Received 13 October 2022, accepted 1 November 2022, date of publication 7 November 2022, date of current version 14 November 2022.

Digital Object Identifier 10.1109/ACCESS.2022.3220240

RESEARCH ARTICLE

Enhanced MPC for Omnidirectional Robot Motion Tracking Using Laguerre Functions and Non-Iterative Linearization

MAHMOUD EL-SAYYAH[1], MOHAMAD R. SAAD[1], (Senior Member, IEEE), AND MAAROUF SAAD[2], (Senior Member, IEEE)
[1]School of Engineering, Université de Québec en Abitibi Témiscamingue, Rouyn-Noranda, QC J9X 5E4, Canada
[2]École de Technologie Supérieure (ETS), Montreal, QC H3C1K3, Canada

Corresponding author: Mahmoud El-Sayyah (mahmoud.sayyah@uqat.ca)

This work was supported by the Natural Science and Engineering Research Council of Canada (NSERC) under Grant RGPIN-2018-04916.

ABSTRACT To cope with the computational complexity of the traditional model predictive control, and to reduce the error of the linearization and prediction processes, this paper presents an improved model predictive control algorithm, based on Laguerre functions, for the motion tracking of an omnidirectional mobile robot with non-iterative linearization. To design the controller, the kinematic modeling of the three-wheeled omnidirectional robot was first performed. Next, the model predictive algorithm was developed using Laguerre functions to parametrize the control signals. At each sampling instant of the online optimization, a linearization along the predicted trajectory, based on the duality principle between optimal control and stochastic filtering, was carried out to deal with the nonlinearities of the system. This non-iterative linearization provides better approximation of the nonlinear behavior which improves the prediction process and the tracking performance, with lower computational burden due to the use of the Laguerre functions. The new controller is applied to solve the trajectory-tracking problem of an omnidirectional robot. A comparative study between the proposed controller, the conventional model predictive control, and the nonlinear model predictive approach is made. Simulation results confirm that the new controller outperform the latter ones regarding tracking accuracy with considerably low computational effort. The feasibility of the controller is demonstrated by real-time experiment on the Robotino-Festo omnidirectional mobile robot.

INDEX TERMS Laguerre functions, linearization, model predictive control (MPC), omnidirectional mobile robot, stochastic filtering, trajectory-following.

I. INTRODUCTION

Nowadays, wheeled mobile robot (WMR), the fruit of combination of the latest sensing technology with advanced control strategies, becomes an important player in modern society [1], [2]. Due to their abilities to increase productivity, and improve work environments safety, WMRs are increasingly entrusted to occupy significant roles in variant sectors such as agriculture, space, surveillance, and mining [3], [4], [5], [6]. In an intelligent mining industry, these robots are expected to act autonomously while completing complex tasks such as automated transportation, automatic inspection and exploring hazardous unsafe areas [7], [8]. One of the key operating conditions of WMR is trajectory tracking which aims to converge the robot's actual position toward a predefined path which can be uploaded as an offline map or generated online by path-planning methods [9]. An omnidirectional mobile robot (OMR) is a special type of WMR with the abilities to move instantly in all directions without any reorientation, which gives it a great advantage to complete the tracking task in such unpredictable dynamic environments [10], [11]. Tracking accuracy and constraints handling are two main criteria in developing the control algorithms along with the ability to deal with nonlinear and multivariable characteristics of the systems. In recent years, many control strategies have been proposed to solve the trajectory-tracking problem of

The associate editor coordinating the review of this manuscript and approving it for publication was Ton Duc Do.

the WMR [12], [13]. In [14], a PI controller tuned by an adaptative fuzzy logic was used as a high-level controller for an OMR. The fuzzy-PI, which corrects the kinematic errors, was paired with a linear quadratic regulator (LQR) as a low-level control of the velocities and accelerations. This combination showed significant improvement over a PI control alone; however, the use of an LQR for low-level control caused a deviation between the desired and the actual paths which greatly increased in the real-time application and led to unsatisfactory results. A control scheme taking into consideration the kinematic and dynamic uncertainties of the OMR was proposed in [15]. A sliding-mode-based observer was used to estimate these uncertainties, then a feedback linearization controller was used to handle such uncertainties. The controller showed a good trajectory-tracking performance. Nonetheless, since these methods cannot handle constraints directly, saturation was used to limit the control signals, which is not acceptable in practice. Another controller for the OMR was presented in [16], which uses a linearizing adaptative algorithm for the kinematic control. This causes some singularities in the control signals. Such singularities are dealt with by switching to a sliding mode controller around them. Indeed, the resulting approach helps to reduce the control effort and eliminate the singularities; however, the risk of chattering appears which may harm the actuators. In [17], a bioinspired backstepping controller was proposed. It performs the tracking task while reducing the large velocity jumps that occur in the traditional backstepping control; however only constant reference speeds were considered. Model-free control schemes were used in [18] and [19]. Using a visual serving strategy in [19] provides a unified algorithm for tracking and regulation. However, these model-free methods ignore useful information from the system model, and they are less adequate compared to systematic methods. In addition, when it comes to harsh environments, physical and operational constraints are expected, and need to be taken into consideration in which the previous methods are limited.

To overcome the above-mentioned problems, one can consider the model-based predictive control (MPC). It is one of the advanced techniques that now has a huge impact on the development of control systems and on research in feedback control areas and has achieved remarkable success in the practical field [20], [21]. This success of the MPC is attributed to many reasons. First, due to the finite control horizon, nonlinear systems dynamics, and process inputs, state and output constraints can be handled directly by the MPC algorithms. Moreover, the prediction aspect of this method over a future time horizon makes it possible to anticipate and remove the effect of disturbances, which leads to better tracking of the future trajectory. Finally, MPC principles and algorithms are relatively easy to understand and to extend to multi-input multi-output systems [22], [23]. The general idea of MPC is to solve an online open-loop optimization problem at each sampling time, and to find a trajectory of future manipulated variables that optimize the future behavior of the system outputs within a limited time window. Traditionally, MPC was only applied to sufficiently slow systems due to the high computational cost required to perform the online optimization, but thanks to increased hardware efficiency, MPC is now applicable to systems with faster dynamics.

The MPC can explicitly handle nonlinearities, and since most systems are inherently nonlinear, many nonlinear model predictive control (NMPC) algorithms have been developed using iterative solutions to solve the optimization problem [24]. In [25], a basic NMPC algorithm that uses the gradient descent method was applied to solve an OMR trajectory tracking with obstacle avoidance. The algorithm gave effective results in both simulation and real-time experiments; however, in the experiments, due to the high computational load, the movement direction and speed were fixed and only the orientation was controlled by NMPC. Using nonlinear systems directly in the NMPC algorithm often leads to undesired complexity and high computational demand. Therefore, studies have been conducted to overcome these problems. In [26], the nonlinear system was modeled in the Weiner model structure, which divides the system into two parts, a linear time invariant system followed by a static nonlinear element. Then, linear MPC was used for the linear part and polynomial representation for the nonlinear part; however, for the Weiner model to properly describe the nonlinear aspects of the system, prior knowledge of these nonlinearities should be available, which is not the case for most systems, and a simple polynomial representation does not give an accurate description of these nonlinearities. Similarly, in [27], a NARMA-Volterra model was selected to represent the brain and used to predict neural activity. In addition, Laguerre functions were introduced to reduce the number of estimation parameters in the Volterra model, and then linear MPC was applied to solve the optimization problem. Nonetheless, Volterra models exhibit high level of complexity, which makes it impractical in modeling strong nonlinearities, and in order to reduce it, prior knowledge of the nonlinear aspect is required. Another popular way to deal with nonlinearities is to linearize the system at each time instant, along the desired trajectory, and to use this linear approximation to compute the predicted future trajectory and then apply the well-known linear MPC [28]; however, when using an approximation at the current time instant to predict the whole future trajectory, the error of the linearization will accumulate, which leads to a poor prediction process. In [29], a duality-based control algorithm has been developed to control a two-wheeled differential robot. The approach uses the duality between optimal control and stochastic filtering to approximate the manipulated variables, and to linearize the nonlinear systems. The linearization and prediction processes were based on the duality without dependence on the future control signals. The algorithm led to better approximation of the nonlinear plants compared to other linearization-based methods; however, the algorithm consists of two passes, forward for linearization and prediction, and backward for smoothing and control signals approximation, which double the computation time. In addition, it cannot explicitly handle constraints.

Even with today's advanced computing technology, reducing the computational cost in both MPC and NMPC remains a challenge, especially when dealing with systems of fast complex dynamics. To cope with this issue, many studies have been carried out in the last few years [30], [31], [32]. A fast NPMC algorithm was presented in [33] for aerial vehicles. The 1-norm was used in the cost function along with the Resilient Propagation to solve the optimization where only the sign of the partial derivative is used. The optimized approach was validated on small aerial vehicles with limited computation capability. Nonetheless, using a simple 1-norm in the evaluation function reduces the tracking performance. Deep-learning-based methods are powerful tools to reduce optimization's time [34], [35]. In [36] Neural-dynamic optimization was considered where the MPC was iteratively transformed to a quadratic programing problem, which is solved using primal-dual neural network. The proposed algorithm was applied to solve the trajectory tracking of a mobile robot and significantly reduced the computation complexity; however, when using a nonconvex cost function, the algorithm can easily get trapped in a local minimum, hence the global solution is hard to obtain. In [37], a learning-by-imitation scheme for mobile medical robot was presented. NMPC was used with the optimization solved by a one-layer projection neural network. Although these methods can reduce the computational cost, a large data set is needed to train and optimize the network, which is hard to get from an unpredicted environment like underground mines. Another method in literature is to introduce a set of discrete orthonormal basis functions, called the Laguerre functions, to parametrize the control signals. This allows the realization of a longer control horizon with fewer optimization parameters, which consequently reduces the computational time [38], [39]. In [40], Laguerre functions were used to parametrize both linear MPC and NMPC. The resulting algorithms were compared to other approaches and showed significant improvement regarding the computational cost and the number of optimization variables; however, only simulation results were given without real-time implementation.

In [41], the effect of the parametrization using Laguerre functions on the feasibility and performance of the MPC was analyzed, and it showed great improvement on the feasibility while maintaining a good performance; however, only dual mode MPC, which uses an infinite horizon for prediction, was considered. Recently in [42], an MPC controller parametrized by Laguerre functions was used to control a fast-switching electronic DC-DC converter allowing the use of a significantly short sampling time. Laguerre functions were first used with MPC in [43], which later has been expanded in [44] where a comprehensive study on the use of Laguerre functions with MPC is given, and which all the above-mentioned studies refer to; however, only linear systems that are supposed to remain constant during the entire prediction process are considered.

Inspired by these ideas and motivated to find a more practical tracking algorithm that deals with nonlinearities accurately while maintaining good performance and low computational demand, this paper proposes an enhanced MPC algorithm, based on Laguerre functions (LMPC), for trajectory tracking of OMR. To ensure a good tracking performance and reduce the linearization's error, this controller deals with nonlinearities by noniteratively linearizing the system along the predicted trajectory, which, to the best of our knowledge, has never been done before. The duality between optimal control and stochastic filtering is used to compute the linearization points, which allows the linearization of the system without dependence on the to be computed control variables. Contrary to existing approaches, the duality is used only for the prediction, then it is combined with an optimizer to compute the optimal solution. This will enhance the prediction process, but also increase the computation cost. To compensate for this increase, Laguerre functions will be used to parametrize the control variables, which will reduce the number of optimization variables and consequently the computational burden. This makes it suitable for real-time implementation allowing the robot to make fast decisions and swiftly adapt to sudden changes in complex environments. The existing Laguerre parametrization method is further developed to consider the change of the linear system at each prediction instant. The performance of the proposed algorithm is evaluated by simulation and by experiment on the Robotino-Festo OMR and a comparative study of accuracy and computational efficiency is carried out with the traditional MPC and NMPC. The main contributions of this paper are summarized as follows:

1. The non-iterative linearization along the future optimal state trajectory prevents the accumulation of the linearization's error, which gives a better approximation of the nonlinear behavior and improves the prediction process and consequently the tracking performance.
2. Parametrization using Laguerre functions reduces the computational cost of the online optimization allowing real-time implementation with longer prediction horizon for better performance.

This paper is organized as follows. In Section II, the kinematic model of the OMR is presented along with the state-space representation and the transition matrix of linearization. An introduction to Laguerre's functions and the LMPC development is done in Section III. In Section IV, a comparison with the NPMC and the traditional linear MPC approaches is illustrated with simulation and experimental results. A conclusion is given in Section V.

II. MODELING OF THE OMNIDIRECTIONAL MOBILE ROBOT

For this study, a three-wheels OMR is considered. This robot has three degrees of freedom and can achieve any translational and rotational movements regardless of its initial orientation. The three omni-wheels, placed at 120° from each other, allow the robot to turn on the spot and to move in any direction (Fig. 1). The method developed in this paper

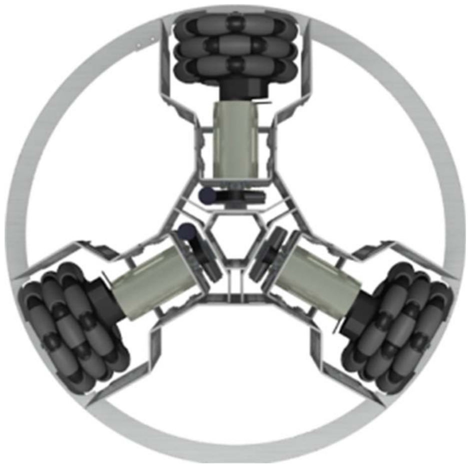


FIGURE 1. Omni-drive of the omnidirectional robot.

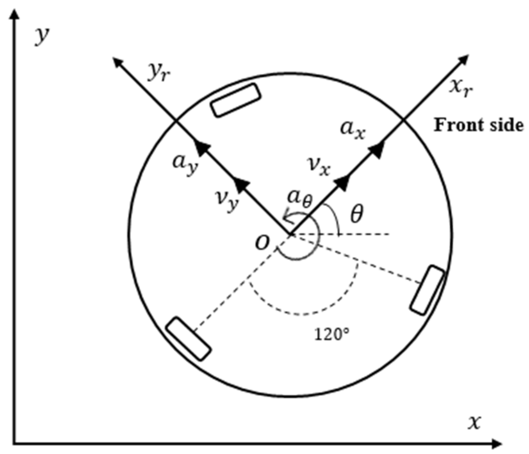


FIGURE 2. Local and global coordinates.

is a model-based algorithm, hence an accurate kinematic model is essential to accurately perform the trajectory tracking tasks. Fig. 2 illustrates the location of the robot using global and local (moving) coordinates. Let (x, y, θ) denote the position and orientation of the robot in the global frame, and (x_r, y_r, θ_r) denote the position and orientation in the local frame. The local coordinates can be transposed into the global coordinates by

$$\begin{pmatrix} x \\ y \\ \theta \end{pmatrix} = T \times \begin{pmatrix} x_r \\ y_r \\ \theta_r \end{pmatrix} \tag{1}$$

where T is the transformation matrix that maps the local coordinates into the global coordinates

$$T = \begin{pmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{pmatrix} \tag{2}$$

Let $q = \begin{pmatrix} x & y & \theta & v_x & v_y & \omega \end{pmatrix}^T$ be the state vector of the vehicle with position (x, y), orientation (θ), the components of translational velocity (v_x, v_y) and rotational velocity (ω). The state evolves through translational accelerations (a_x, a_y) and rotational acceleration (a_θ) which represent the manipulated variables, then the kinematic equations can be written as

$$\begin{aligned} \dot{x} &= v_x \cos\theta - v_y \sin\theta \\ \dot{y} &= v_x \sin\theta + v_y \cos\theta \\ \dot{\theta} &= \omega \\ \dot{v}_x &= a_x \\ \dot{v}_y &= a_y \\ \dot{\omega} &= a_\theta \end{aligned} \tag{3}$$

Remark 1: The relation between the wheels' velocities and the translational and rotational velocities of the OMR is as follows:

$$\begin{bmatrix} wheel_1 \\ wheel_2 \\ wheel_3 \end{bmatrix} = \frac{1}{r} \begin{bmatrix} 0 & 1 & R \\ -\sqrt{3}/2 & 1/2 & R \\ \sqrt{3}/2 & -1/2 & R \end{bmatrix} \begin{bmatrix} V_x \\ V_y \\ \omega \end{bmatrix}$$

where $wheel_i$ is the rotational speed of the *ith* wheel, r is the wheel's radius, and R is the distance from the center of the robot to the wheel. However, since the translational and rotational accelerations of the center of gravity of the robot are our control variables, this relation is not used.

By using the forward differences method to approximate the state and input variables, we obtain the following discrete-time state-space representation of the kinematic model

$$\begin{pmatrix} x_{k+1} \\ y_{k+1} \\ \theta_{k+1} \\ v_{x,k+1} \\ v_{y,k+1} \\ \omega_{k+1} \end{pmatrix} = \begin{pmatrix} x_k + v_{xk}\Delta t \cos\theta_k - v_{yk}\Delta t \sin\theta_k \\ y_k + v_{xk}\Delta t \sin\theta_k + v_{yk}\Delta t \cos\theta_k \\ \theta_k + \omega_k \Delta t \\ v_{xk} \\ v_{yk} \\ \omega_k \end{pmatrix} + \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ \Delta t & 0 & 0 \\ 0 & \Delta t & 0 \\ 0 & 0 & \Delta t \end{pmatrix} \begin{pmatrix} a_{xk} \\ a_{yk} \\ a_{\theta k} \end{pmatrix}$$

which in compact form becomes:

$$\begin{aligned} q_{k+1} &= f(q_k) + Bu_k \\ z_k &= Cq_k \end{aligned} \tag{4}$$

with Δt is the simulation step, $f(q_k)$ is a nonlinear function of the state, B is the input matrix, z_k is the output vector which contains the position, orientation, and velocities of the OMR, and C is the output matrix. The proposed controller is based on linearization, thus the transition matrix to the linearized system, i.e. the Jacobian matrix, is needed and can be obtained by (5), as shown at the bottom of the next page, which will be used to compute the linear approximation of the system at each prediction step.

III. CONTROL SYSTEM ARCHITECTURE

The main objective of robot control is to reach the desired trajectory with the desired orientation and to stay on it for all future time. To achieve this goal, an MPC algorithm based on Laguerre functions denoted as LMPC, is proposed. The LMPC controller consists of two major parts, a non-iterative linearization and a parametrization using Laguerre functions.

A. NON-ITERATIVE LINEARIZATION

Since nonlinear optimal control problems are hard to solve and computationally demanding, linearization is often used. Choosing the linearization points is the main problem of linearization approaches. Using the optimal trajectory as linearization points would be the best solution but since they depend on the control to be computed and the control depends on them, iterative methods are required [29]. Iterative Linear Quadratic Regulator (iLQR) is one of the approaches that uses iterative linearization. It uses the standard linear quadratic regulator to compute the optimal increments of the manipulated variables, re-linearizes the system around the obtained optimal trajectory, and then repeats this process until convergence is reached [45]. Another method based on iterative linearization is NMPC. It can deal with nonlinear cost functions and outputs functions by solving an open loop optimization using a nonlinear optimization algorithm that, in most cases, computes gradients of the cost function which depends on model's gradients [23]. These methods can give high performance; however, they require a high number of iterations and are highly computationally demanding which make them unfeasible in practice. To avoid applying iterative algorithms, we are going to use the duality between stochastic filtering and optimal control to achieve non-iterative linearization by approximating the future optimal trajectory.

To use such duality, we first consider the stochastic dynamics for the control problem as:

$$\begin{aligned} q_{k+1} &= f(q_k) + Bu_k + w_k \\ z_k &= Cq_k + \sigma_k \end{aligned} \tag{6}$$

where w_k and σ_k are fictitious Gaussian noise with covariances V_k and W_k, respectively. The cost function to be minimized at each simulation step is considered quadratic with no final cost and is given by:

$$J_k = \sum_{k=1}^{N_p} e_k^T Q_k e_k + \sum_{k=0}^{N_p} u_k^T R_k u_k \tag{7}$$

with $e_k = s_k - Cq_k$ where s_k are the desired set points, which act as the observation in the duality problem, N_p is the length of the prediction horizon, and Q_k, R_k are known symmetric positive definite matrices.

The stochastic dynamics for the dual estimation problem is considered as:

$$\begin{aligned} \bar{q}_{k+1} &= f(\bar{q}_k) + w_k \\ s_k &= C\bar{q}_k + \sigma_k \end{aligned} \tag{8}$$

The duality between the optimal control problem (6) and (7), and the estimation dynamics (8) is established by choosing $V_k = BR_k^{-1}B^{-1}$ and $W_k = Q_k^{-1}$ [46]. The computation of the optimal linearization points can be done using the following Kalman filter equations:

$$\begin{aligned} K_k &= P_k C^T (CP_k C^T + Q_k^{-1})^{-1} \\ P_{k+1} &= A_k (I - K_k C) P_k A_k^T + BR_k^{-1} B^T \\ \hat{q}_{k+1} &= \hat{q}_k + K_k (s_k - C\hat{q}_k) \end{aligned} \tag{9}$$

with K_k is the Kalman gain matrix, P_k is the estimation error covariance matrix [29], [46] and A_k is the transition matrix (5).

B. CONSTRAINTS

The key advantage of MPC is the capability to handle inequality constraints explicitly. Also, OMR exhibits numerous physical and operational constraints that need to be satisfied by the control algorithm. First, there are limits on the acceleration of the OMR, which in this case represent the control variables and can be expressed as follows:

$$u_{\min} \le u_k \le u_{\max}$$

where $u_{\min}$ and $u_{\max}$ are vectors of the same size as u_k that contain the lower and upper acceleration limits respectively. Furthermore, when tracking a reference trajectory, the robot velocities must not exceed the velocity constraints. In this study, v_x and v_y have the same maximum value denoted $v_{\max}$, and the maximum rotational speed is denoted $\omega_{\max}$, then the speeds constraints can be written as:

$$\begin{bmatrix} -v_{\max} \\ -v_{\max} \\ -\omega_{\max} \end{bmatrix} \le \begin{bmatrix} v_{xk} \\ v_{yk} \\ \omega_k \end{bmatrix} \le \begin{bmatrix} v_{\max} \\ v_{\max} \\ \omega_{\max} \end{bmatrix} \tag{10}$$

$$\begin{aligned} A_k &= \frac{\partial f(q_k)}{\partial q_k} \\ &= \begin{bmatrix} 1 & 0 & -v_{xk}\Delta t \sin\theta_k - v_{yk}\Delta t \cos\theta_k & \Delta t \cos\theta_k & -\Delta t \sin\theta_k & 0 \\ 0 & 1 & v_{xk}\Delta t \cos\theta_k - v_{yk}\Delta t \sin\theta_k & \Delta t \sin\theta_k & \Delta t \cos\theta_k & 0 \\ 0 & 0 & 1 & 0 & 0 & \Delta t \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \end{aligned} \tag{5}$$

Finally, using (4), the operational constraints of the OMR can be written together as follows

$$\begin{aligned} -u_k &\leq -u_{\min} \\ -u_k &\leq \frac{1}{\Delta t}\left(\begin{bmatrix} v_{\max} \\ v_{\max} \\ \omega_{\max} \end{bmatrix} + \begin{bmatrix} v_{xk} \\ v_{yk} \\ \omega_k \end{bmatrix}\right) \\ u_k &\leq \frac{1}{\Delta t}\left(\begin{bmatrix} v_{\max} \\ v_{\max} \\ \omega_{\max} \end{bmatrix} - \begin{bmatrix} v_{xk} \\ v_{yk} \\ \omega_k \end{bmatrix}\right) \\ u_k &\leq u_{\max} \end{aligned} \tag{11}$$

C. MPC WITH LAGUERRE FUNCTIONS

The second part of the LMPC algorithm is to compute the optimal control action by minimizing the quadratic cost function (7) using the MPC method with Laguerre functions. Although linearizing the system at each prediction step will give more accurate approximation, it is more demanding computationally. Therefore, we introduce the Laguerre functions in the problem formulation to solve the open-loop optimization.

1) INTRODUCTION TO LAGUERRE FUNCTIONS

To reduce the computational complexity of the standard MPC, we approximate the future control trajectory by combining a set of orthonormal functions (Laguerre Functions) linearly with few coefficients, which helps to cover the entire control horizon without the need for massive optimization parameters [38]. The Laguerre orthonormal sequence is described by the following z-transforms:

$$\begin{aligned} \Gamma_1(z) &= \frac{\sqrt{1-a^2}}{1-az^{-1}} \\ \Gamma_2(z) &= \frac{\sqrt{1-a^2}}{1-az^{-1}}\frac{z^{-1}-a}{1-az^{-1}} \\ &\vdots \\ \Gamma_N(z) &= \frac{\sqrt{1-a^2}}{1-az^{-1}}\left(\frac{z^{-1}-a}{1-az^{-1}}\right)^{N-1} \end{aligned} \tag{12}$$

where a is the scaling factor of the Laguerre sequence, and $0 \leq a < 1$ for the stability of the sequence [44]. Let $l_i(k)$ be the inverse z-transform of $\Gamma_i(z, a)$, then the set of discrete-time Laguerre functions can be written in vector form as:

$$L(k) = \begin{bmatrix} l_1(k) & l_2(k) & \cdots & l_N(k) \end{bmatrix}^T \tag{13}$$

Taking advantage of the sequence realization

$$\begin{aligned} \Gamma_1(z) &= \left(\sqrt{1-a^2}\right) / \left(1-az^{-1}\right) \\ \Gamma_k(z) &= \Gamma_{k-1}(z)\frac{z^{-1}-a}{1-az^{-1}} \quad k = 2, 3, \ldots N \end{aligned}$$

We can describe the sequence by the following state space representation:

$$L(k+1) = A_l L(k) \tag{14}$$

with $A_l(N \times N)$ and the initial condition $L(0)$ given by:

$$A_l = \begin{bmatrix} a & 0 & 0 & \cdots & 0 \\ \beta & a & 0 & \cdots & 0 \\ -a\beta & \beta & a & \cdots & 0 \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ (-1)^{N-2}a^{N-2}\beta & \cdots & \cdots & \beta & a \end{bmatrix}$$
$$L(0) = \sqrt{\beta}\begin{bmatrix} 1 & -a & a^2 & -a^3 & \ldots & (-1)^{N-1}a^{N-1} \end{bmatrix}^T \tag{15}$$

where $\beta = 1 - a^2$. The orthonormality of Laguerre functions can be expressed in the time domain by:

$$\begin{aligned} \sum_{k=0}^{\infty} l_i(k)l_j(k) &= 0 \quad for\ i \neq j \\ \sum_{k=0}^{\infty} l_i(k)l_j(k) &= 1 \quad for\ i = j \end{aligned} \tag{16}$$

Finally, this set of Laguerre functions can be used to capture the response $H(k)$ of an arbitrary system by:

$$H(k) = c_1 l_1(k) + c_2 l_2(k) + \ldots + c_N l_N(k) \tag{17}$$

where $c_1, c_2, \ldots, c_N$ are the coefficients to be determined using the system data, and N is the number of terms used to capture the response [44].

2) LAGUERRE-BASED MPC

The linear approximation of the kinematic model (4) at a random time instant k is given as:

$$\begin{aligned} q_{k+1} &= A_k q_k + Bu_k \\ z_k &= Cq_k \end{aligned} \tag{18}$$

Since the OMR has three motors, i.e., three control inputs, let the matrix B be partitioned into

$$B = \begin{bmatrix} B_1 & B_2 & B_3 \end{bmatrix} \tag{19}$$

and define

$$A_m = \prod_{i=m}^{1} A_{k,i} \tag{20}$$

where $A_{k,i}$ is the transition matrix computed at the *ith* future instant, and m is the current prediction instant. Here, the linearization at every prediction sample is considered, and $A_{k,i}$ is computed using the duality principle. Using (17), each control variable can be approximated at an arbitrary future instant with

$$u_i(k+m) = L_i(m)^T \eta_i = \sum_{j=1}^{N_i} c_j^i(k) l_j^i(m) \tag{21}$$

where k is the initial time of the moving horizon, m is the future instant where $k \leq m \leq k + N_p - 1$, $i = 1, 2, 3$ implies the *ith* control variable, c_j^i are the coefficients, which are functions of the initial time of the moving horizon, N_i is the number of parameters used to capture the *ith* control variable,

η_i is a vector containing the coefficients, $\eta_i = \begin{bmatrix} c_1^i & \ldots & c_N^i \end{bmatrix}^T$, and N_p is length of the prediction horizon. By using (18) and (21), the prediction of the future state variables can be written as:

$$\begin{aligned} q(k+m) &= A_m q(k) \\ &\quad + \begin{bmatrix} B_1 L_1(m-1)^T & B_2 L_2(m-1)^T & B_3 L_3(m-1)^T \end{bmatrix} \eta \\ &\quad + \sum_{j=0}^{m-2} A_{m-j-1} \begin{bmatrix} B_1 L_1(j)^T & B_2 L_2(j)^T & B_3 L_3(j)^T \end{bmatrix} \eta \\ &= A_m q(k) + \phi(m)^T \eta \end{aligned} \tag{22}$$

with $\eta = \begin{bmatrix} \eta_1^T & \eta_2^T & \eta_3^T \end{bmatrix}^T$ and

$$\begin{aligned} \phi(m)^T &= \begin{bmatrix} B_1 L_1(m-1)^T & B_2 L_2(m-1)^T & B_3 L_3(m-1)^T \end{bmatrix} \\ &\quad + \sum_{j=0}^{m-2} A_{m-j-1} \begin{bmatrix} B_1 L_1(j)^T & B_2 L_2(j)^T & B_3 L_3(j)^T \end{bmatrix} \end{aligned}$$

Here we have taken into consideration that the matrix A_k is not constant during the prediction process, it is changing at each future instant. With a sufficiently large prediction horizon, the orthonormal property becomes

$$\begin{aligned} &\sum_{k=0}^{N_p} l_i(k) l_j(k) = 0 \quad \textit{for } i \neq j \\ &\sum_{k=0}^{N_p} l_i(k) l_j(k) = 1 \quad \textit{for } i = j \end{aligned} \tag{23}$$

Using (21) and (23), the sum of the future control inputs can be computed by:

$$\sum_{m=0}^{N_p} u\,(k+m)^T R_k u(k+m) = \eta^T R_L \eta \tag{24}$$

where R_L is a block-diagonal matrix where each block contains one of the elements of R_k on its diagonal. Define $Q_L = C^T Q_k C$, putting (22) and (24) in the cost function (7) will lead to the following form:

$$\begin{aligned} J &= \eta^T \left(\sum_{m=1}^{N_p} \phi(m) Q_L \phi(m)^T + R_L \right) \eta \\ &\quad + 2\eta^T \left(\sum_{m=1}^{N_p} \phi(m) Q_L A_m \right) q(k) \\ &\quad - 2\eta^T \left(\sum_{m=1}^{N_p} \phi(m) C^T Q_k s(k+m) \right) \\ &\quad + \sum_{m=1}^{N_p} (s(k+m) - CA_m q(k))^T \\ &\quad \times Q_k \,(s(k+m) - CA_m q(k)) \\ &= \eta^T \Omega \eta + 2\eta^T (\Psi q(k) - \xi) \\ &\quad + \sum_{m=1}^{N_p} (s(k+m) - CA_m q(k))^T \\ &\quad \times Q_k \,(s(k+m) - CA_m q(k)) \end{aligned} \tag{25}$$

where

$$\begin{aligned} \Omega &= \sum_{m=1}^{N_p} \phi(m) Q_L \phi(m)^T + R_l \\ \Psi &= \sum_{m=1}^{N_p} \phi(m) Q_L A_m \\ \xi &= \sum_{m=1}^{N_p} \phi(m) C^T Q_k s(k+m) \end{aligned} \tag{26}$$

By setting the partial derivative (relative to η) of the cost function (25) to zero, the optimal solution can be found as:

$$\eta = \Omega^{-1} (\xi - \Psi q(k)) \tag{27}$$

and the first control action can be computed using

$$u_k = \begin{bmatrix} L_1(0)^T & 0_2^T & 0_3^T \\ 0_1^T & L_2(0)^T & 0_3^T \\ 0_1^T & 0_2^T & L_3(0)^T \end{bmatrix} \eta \tag{28}$$

Here $L_i(0)$ is the initial condition for the Laguerre functions of the *ith* input, and 0_i is a zero vector with the same dimension as $L_i(0)$.

Remark 2: We note that the Laguerre functions parameters N and a can be assigned to each input variable independently of the others, which gives more flexibility in the control design.

Remark 3: Scaling factor a and the number of terms needed to approximate u_k are closely related. If we set $a = 0$ and the number of terms $N = N_c$ the control horizon, we obtain the traditional MPC approach, and by choosing $0 < a < 1$, we can achieve similar performance with N far less than N_c and reduce the computational cost [44].

3) CONSTRAINED OPTIMAL SOLUTION

The Laguerre functions can also be introduced in the constraints' description, which gives more flexibility for the designer to force the constraints at any specified future instant. The constraints on the control variables at an arbitrary future time m is:

$$U_{\min} \leq u(k+m) \leq U_{\max}$$

with $m = 0, 1, \ldots, N_p - 1$, and $U_{\min}$, $U_{\max}$ are the control bounds from (11). This can be written in terms of η as:

$$U_{\min} \leq \begin{bmatrix} L_1(m)^T & 0_2^T & 0_3^T \\ 0_1^T & L_2(m)^T & 0_3^T \\ 0_1^T & 0_2^T & L_3(m)^T \end{bmatrix} \eta \leq U_{\max}$$

The constrained optimal solution is obtained by solving a dual-quadratic problem using the Hildreth method [42], [44]. First the active set of the inequalities constraints is selected in matrix M_{act}, then the Lagrange multipliers λ_{act} are found using the Hildreth' algorithm, and finally the optimal constrained solution is computed by

$$\eta = \Omega^{-1}\left(\xi - \Psi q(k) - M_{act}^T \lambda_{act}\right) \tag{29}$$

Algorithm 1 is the resulting algorithm named LMPC. It shows the two steps, the linearization and the control computing. Unlike the algorithm used in [29], which uses the duality to linearize the system and approximate the control inputs, the LMPC uses the duality only for linearization. The control inputs are computed by introducing the Laguerre functions and performing online optimization. Contrary to existing methods, the parameterization using Laguerre functions takes into consideration the linearization at each future instant. In the case of constrained control, the LMPC uses the Hildreth algorithm to identify the active constraints and compute the Lagrange multipliers. All the state variables are considered available for measurement. Therefore, the LMPC is a deterministic state feedback controller.

Algorithm 1 LMPC

```
1:  Initialization
2:  $\hat{q}_k = q_k$; $P_k = 0$
3:  Prediction
4:  for $m = k+1, \ldots, k+N_p$
5:      Linearization
6:      $\bar{q} = f(\hat{q}_{m-1})$
7:      $A_{m-1} = \frac{\partial f}{\partial q_{m-1}}\Big|_{q_{m-1}=\hat{q}_{m-1}}$
8:      $k_m = P_{m-1}C^T\left(CP_{m-1}C^T + Q_k^{-1}\right)^{-1}$
9:        $P_m = A_{m-1}(I - K_m C)P_{m-1}A_{m-1}^T + BR_k^{-1}B^T$
10:     $\hat{q} = \bar{q}_m + k_m(s_m - C\bar{q}_m)$
11:     Compute Convolution Sums
12:     $\Omega$; $\Psi$; $\xi$; (Eq.19)
13: end for
14: Set The Constraints
15: $M_{act}$ and $\lambda_{act}$ using Hildreth Algorithm
16: Compute Optimal Coefficients Vector
17: $\eta_k = \Omega^{-1}(\xi - \Psi q_k - M_{act}^T\lambda_{act})$
18: Compute First Optimal control action
19: $u_k = L_{zero}\eta_k$
```

IV. CASE STUDIES ANS ANALYSES

In this section, the performance of the proposed LMPC algorithm will be analyzed. To show the outstanding performance, the traditional linear MPC and NMPC approaches are introduced for comparisons.

A. SIMULATIONS SETUP

The simulations are carried out using the MATLAB/Simulink software. The aim is to drive the OMR to track a given trajectory by minimizing the cost function (7). All the strategies compared will minimize the same cost function. The following approaches will be compared:

1) LMPC: this algorithm linearizes the system at each future prediction step using the duality principle (9), and then solves the optimality using Laguerre functions (29). The number of terms and the scaling factor will be the same for all input variables.
2) MPC: this method solves the optimization problem using a linearized model and standard optimization algorithm. Implementation is based on [44].
3) NMPC: this algorithm solves the open-loop optimization using the nonlinear model. Implementation is based on the optimized algorithm in [23], where the active-set method is used for the minimization. The convergence threshold is set to 10^{-8} and the maximum number of iterations is 10^3.

To ensure a fair comparison, some unifying conditions need to be set:

1) The prediction horizon N_p is the same for the three strategies and it is set to $N_p = 20$, which ensure the convergence and practical feasibility.
2) For both MPC and NMPC, prediction horizon N_c is the same; however, term N_c does not appear in the LMPC algorithm, since it has been replaced by the number of parameters N and scaling factor a. In [43], it has been shown that for a small N, N_c and a are related by $a \approx e^{-5/N_c}$.
3) The weighting matrices for the strategies compared are set to

$$R_k = diag(0.01, 0.01, 0.01)$$
$$Q_k = diag(25, 25, 25, 0.1, 0.1, 0.1)$$

These tuning parameters were chosen through trial and error, therefore their optimality cannot be guaranteed. Nonetheless, they have demonstrated good performance in this study.

B. TRACKING ANALYSIS AND COMPUTATIONAL RESOURCES

To thoroughly evaluate the tracking performance, we use a desired trajectory given by reference speed components $v_{xd} = 0.5\text{ms}^{-1}$ and $v_{yd} = 0.5\text{ms}^{-1}$, reference positions $x_d = v_{xd}t$; $y_d = v_{yd}t$ m, orientation $\theta_d = 0$ rad, and angular velocity $\omega_d = 0$ rads^{-1}. The trajectory is supposed to be known along the prediction horizon, and all the state variables are considered available for measurement.

We consider a set of $H = 5$ simulations, and a random uniformly distributed initial state

$$q_0^h = Rand([-1, 1], [-1, 1], [-\pi/6, \pi/6], 0, 0, 0)^T$$

For each simulation h, and iteration k, the cost achieved for each approach is denoted $J_r(h, k)$, $r = 1, \ldots, 3$. The desired reference minimum of the cost function is chosen to be 5. On each iteration, the method with the best cost, lower than 5, is taken as reference $J_{best}(h, k)$ to be compared to the other methods [29], i.e, $J_{best} = \min_{1<r<3}(J_r(h, k), 5)$. The average

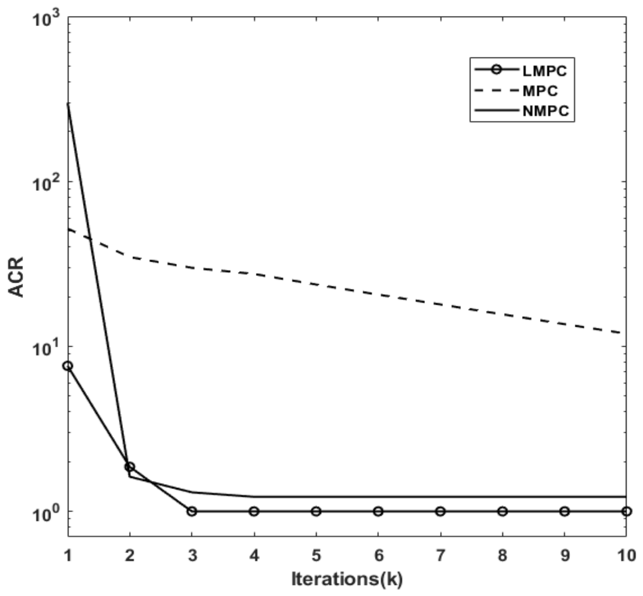


FIGURE 3. Average Cost Ratio (ACR) per iteration.

cost ratio (ACR) over all simulations is computed by:

$$ACR_r = 1/H \sum_{h=1}^{H} \frac{J_r(h,k)}{J_{best}(h,k)}, \quad r = 1,\ldots,3. \tag{30}$$

Furthermore, to compare the tracking performance of different strategies, an additional index is used, which quantifies the tracking quadratic error [38], and it is given by

$$M_i = 1/H \sum_{h=1}^{H} \sqrt{\frac{\sum_{j=1}^{T_{sim}/\Delta t} \left(y_{ref}(j) - y_{sys}(j)\right)^2}{T_{sim}/\Delta t}}; \quad i \in \{x, y, \theta\} \tag{31}$$

with $\Delta t = 0.07$s. In this experiment, the control horizon for MPC and NMPC is set to $N_c = 5$, and for the parameters of LMPC are chosen $a = 0.5$,and $N = 3$.

Fig. 3 shows the ACR of each method for the first 10 iterations in the logarithmic scale. It can be clearly seen that after the first iteration, the initial cost realized by the LMPC is significantly lower than the ones realized by MPC and NMPC, and after 10 iterations, LMPC yields the best performance. Both the LMPC and NMPC converge after the third iteration with LMPC achieving a lower final cost, while MPC took much longer to converge. The numeric values of Table 1 represent the number of variables involved in the optimization process of each strategy, along with the average time for the first ten iterations. For MPC and NMPC, the number optimization variables are computed by multiplying the number of control inputs by the length of the control horizon N_c, whereas for LMPC, it is equal to the number of parameters used to parametrize the manipulated variables. Only 9 variables are involved in the optimization process of the LMPC, while 15 parameters are involved in both MPC and NMPC.

TABLE 1. Number of control parameters and time per iteration.

	LMPC	MPC	NMPC
Number of optimization variables	9	15	15
Time per iteration	0.002	0.001	0.266

TABLE 2. Mean quadratic tracking errors for different strategies.

	LMPC	MPC	NMPC
M_x	0.0083	0.0663	0.0058
M_y	0.0113	0.0631	0.0012
M_ϕ	0.0035	0.0065	0.0217

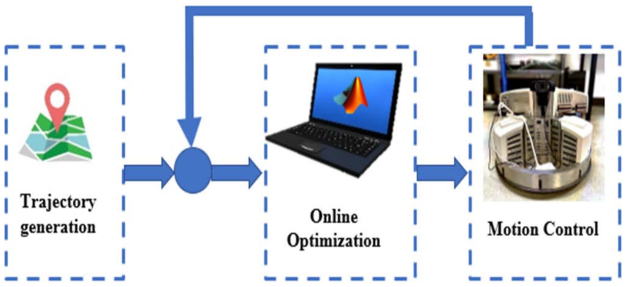


FIGURE 4. Overview of the trajectory tracking experiment.

This gives the LMPC a great computational advantage (100 times faster) over the NMPC while maintaining good performance. Even with linearization performed at each prediction instant, LMPC still managed to keep up with the MPC with only 1ms difference.

Table 2 shows the mean-quadratic tracking errors for the three methods. It can be appreciated that LMPC significantly reduces the tracking error compared to MPC with almost the same computational demand, and it is not that far behind NMPC. In fact, in the orientation tracking, LMPC showed better performance than the other two methods. NMPC showed slightly lower M_x and M_y compared to LMPC, at the cost of higher computation time. This is due to the NMPC iterative aspect. As a result, LMPC can be considered a computationally effective method to obtain optimal results in practice.

C. EXPERIMENTAL RESULTS

In this section, the three algorithms are tested on the Robotino-Festo omnidirectional robot (Fig. 4). The whole system is controlled by an embedded PC to COM Express specifications with Intel i5, 2.4 GHz dual core, 8 GB RAM and 23 GB SSD. For the motor control, a 32-bit microcontroller is used. It generates the PWM signals for actuating the DC motors using a PID controller. The microcontroller

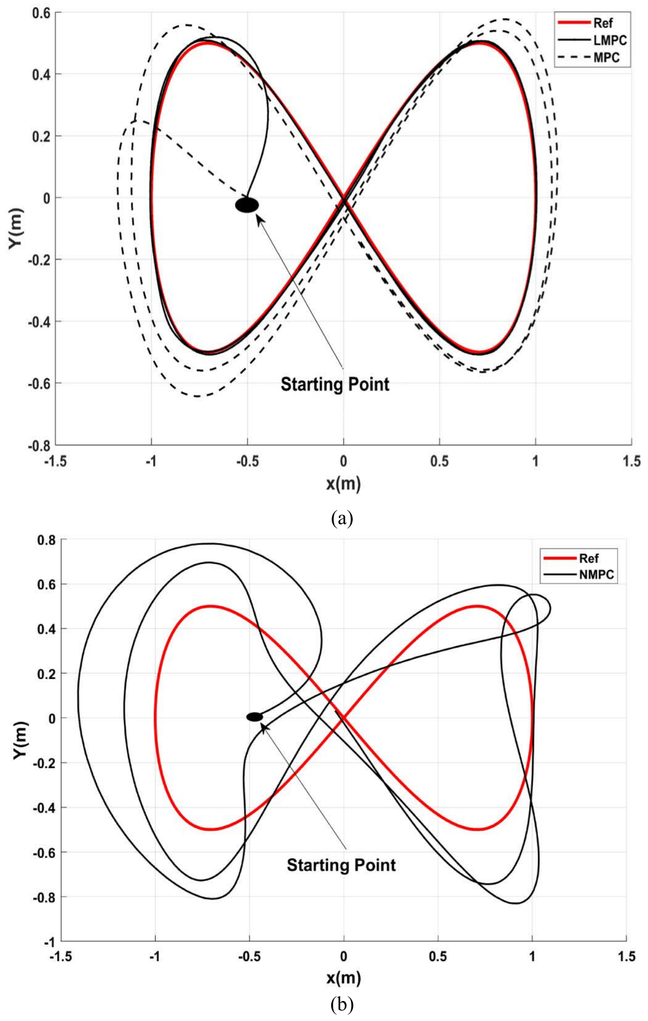


FIGURE 5. Tracking the eight-shaped trajectory in real time: (a) using LMPC and MPC, (b) using NMPC.

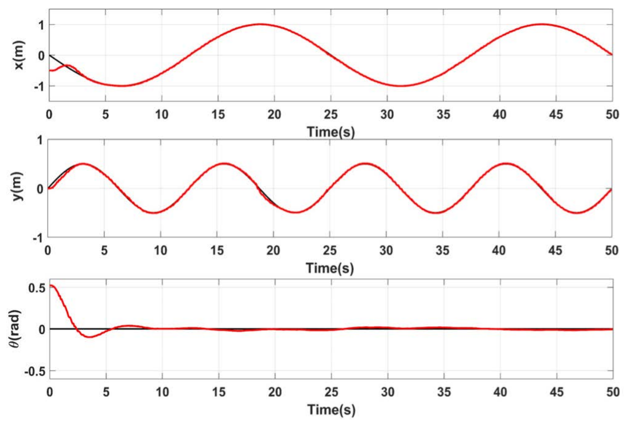


FIGURE 6. Reference positions (black solid line) and real positions (red solid line) using LMPC.

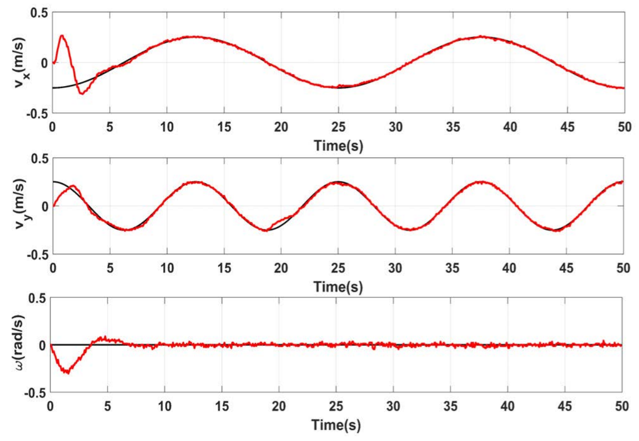


FIGURE 7. Reference speeds (black solid line) and measured speeds (red solid line) using LMPC.

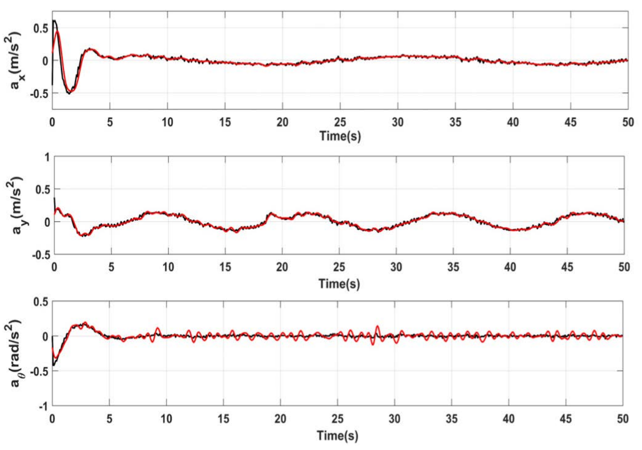


FIGURE 8. Applied control inputs (black solid line) and estimated accelerations (red solid line) using LMPC.

is also used to correct the sensors data. A planetary gear unit with transition ratio 32:1 is used between the drive shafts and omni-wheels [47]. The robot has a maximum translational and rotational speeds of 2m/s and 2rad/s respectively. The algorithms are implemented using the Robotino MATLAB-Simulink toolbox. The robot accepts translational and rotational velocities as inputs, which are expected to be updated every 70ms. Since the compared approaches have accelerations as manipulated variables, integrators are included in the algorithms. The aim is to drive Robotino to follow the eight-shaped trajectory described by the following equations:

$$\begin{aligned} x_d &= -\sin((2\pi/T)t) \\ y_d &= 0.5\sin(2(2\pi/T)t) \\ v_{xd} &= \dot{x}_d; \quad v_{yd} = \dot{y}_d \end{aligned} \tag{32}$$

where $T = 25$ s is the trajectory period, $\theta_d = 0$ and $\omega_d = 0$. The control horizon is chosen 10 for MPC and 5 for NMPC, as for the LMPC, $N = 3$ and $a = 0.8$. The weighting matrices are chosen as

$$\begin{aligned} R_k &= diag(15, 15, 15) \\ Q_k &= diag(80, 80, 80, 0.1, 0.1, 0.1) \end{aligned}$$

and initial position is given as $q_0 = [-0.5, 0, \pi/6, 0, 0, 0]$.

Fig. 5 shows the performance of each method when tracking the eight-shaped trajectory. In Fig. 5(a), the advantage of the LMPC performance over MPC is well illustrated. The NMPC needs longer than the update time of the microcontroller (70ms) to compute the next inputs values, which makes it unsuitable for practice, as shown in Fig. 5(b).

Fig. 6 and Fig. 7 show the state variables of the system over time when using the LMPC: x and y positions, orientation θ, translational velocities v_x and v_y, and angular velocity ω. The computed input accelerations before integration (black lines) and the real robot acceleration (red lines) estimated by differentiating and filtering the odometry data are shown in Fig. 8.

V. CONCLUSION AND FUTURE WORK

In this paper, an enhanced MPC algorithm based on Laguerre functions, LMPC, for trajectory tracking of an OMR has been presented. Non-iterative linearization was performed using the duality between optimal control and stochastic filtering to approximate the nonlinear system, and the Laguerre functions were used to describe the control variables and reduce the number of optimization variables. The method presented provides a way to ameliorate the prediction process, prevent the accumulation of the linearization's error and improve the tracking performance. The computational time was also reduced significantly allowing the algorithm to make fast accurate decisions.

The performance of the proposed algorithm was validated on the trajectory tracking problem of the OMR and compared to the traditional linear MPC algorithm and the NMPC. Experiments show that LMPC can achieve high tracking accuracy, outperforming both MPC and NMPC. Feasibility and suitability for real-time applications were also demonstrated by experiment on Robotino Festo mobile robot.

As future work, we will consider optimizing the tuning parameters using automated tuning algorithms. We will also consider the use of Laguerre Functions directly in the NMPC.

REFERENCES

[1] K. Zhang, J. Wang, X. Xin, X. Li, C. Sun, J. Huang, and W. Kong, "A survey on learning-based model predictive control: Toward path tracking control of mobile platforms," *Appl. Sci.*, vol. 12, no. 4, p. 1995, Feb. 2022.

[2] N. Hassan and A. Saleem, "Analysis of trajectory tracking control algorithms for wheeled mobile robots," in *Proc. IEEE Ind. Electron. Appl. Conf. (IEACon)*, Nov. 2021, pp. 236–241.

[3] B. Fernandez, P. J. Herrera, and J. A. Cerrada, "A simplified optimal path following controller for an agricultural skid-steering robot," *IEEE Access*, vol. 7, pp. 95932–95940, 2019.

[4] H. S. Hewawasam, M. Y. Ibrahim, and G. K. Appuhamillage, "Past, present and future of path-planning algorithms for mobile robot navigation in dynamic environments," *IEEE Open J. Ind. Electron. Soc.*, vol. 3, pp. 353–365, 2022.

[5] I. Rangapur, B. K. S. Prasad, and R. Suresh, "Design and development of spherical spy robot for surveillance operation," *Proc. Comput. Sci.*, vol. 171, pp. 1212–1220, Jan. 2020.

[6] F. Tian, R. Zhou, Z. Li, L. Li, Y. Gao, D. Cao, and L. Chen, "Trajectory planning for autonomous mining trucks considering terrain constraints," *IEEE Trans. Intell. Vehicles*, vol. 6, no. 4, pp. 772–786, Dec. 2021.

[7] D. Topolsky, I. Topolskaya, I. Plaksina, P. Shaburov, N. Yumagulov, D. Fedorov, and E. Zvereva, "Development of a mobile robot for mine exploration," *Processes*, vol. 10, no. 5, p. 865, Apr. 2022.

[8] J. Szrek, J. Jakubiak, and R. Zimroz, "A mobile robot-based system for automatic inspection of belt conveyors in mining industry," *Energies*, vol. 15, no. 1, p. 327, Jan. 2022.

[9] S. Cebollada, L. Payá, M. Flores, A. Peidró, and O. Reinoso, "A state-of-the-art review on mobile robotics tasks using artificial intelligence and visual data," *Expert Syst. Appl.*, vol. 167, Apr. 2021, Art. no. 114195.

[10] H. Eschmann, H. Ebel, and P. Eberhard, "Data-based model of an omnidirectional mobile robot using Gaussian processes," *IFAC-PapersOnLine*, vol. 54, no. 7, pp. 13–18, 2021.

[11] M. Radzak, M. Ali, S. Sha'amri, and A. Azwan, "An overview on real-time control schemes for wheeled mobile robot," *IOP Conf. Ser., Mater. Sci. Eng.*, vol. 342, no. 1, 2018, Art. no. 012059.

[12] S. G. Tzafestas, "Mobile robot control and navigation: A global overview," *J. Intell. Robot. Syst.*, vol. 91, no. 1, pp. 35–58, 2018.

[13] H. Taheri and C. X. Zhao, "Omnidirectional mobile robots, mechanisms and navigation approaches," *Mechanism Mach. Theory*, vol. 153, Nov. 2020, Art. no. 103958.

[14] M. A. Al Mamun, M. T. Nasir, and A. Khayyat, "Embedded system for motion control of an omnidirectional mobile robot," *IEEE Access*, vol. 6, pp. 6722–6739, 2018.

[15] S. Jeong and D. Chwa, "Sliding-mode-disturbance-observer-based robust tracking control for omnidirectional mobile robots with kinematic and dynamic uncertainties," *IEEE/ASME Trans. Mechatronics*, vol. 26, no. 2, pp. 741–752, Apr. 2021.

[16] J.-T. Huang, T. Van Hung, and M.-L. Tseng, "Smooth switching robust adaptive control for omnidirectional mobile robots," *IEEE Trans. Control Syst. Technol.*, vol. 23, no. 5, pp. 1986–1993, Sep. 2015.

[17] Z. Xu, S. X. Yang, and S. A. Gadsden, "Enhanced bioinspired backstepping control for a mobile robot with unscented Kalman filter," *IEEE Access*, vol. 8, pp. 125899–125908, 2020.

[18] R. H. Abiyev, N. Akkaya, and I. Gunsel, "Control of omnidirectional robot using Z-number-based fuzzy system," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 49, no. 1, pp. 238–252, Jan. 2019.

[19] B. Li, Y. Fang, G. Hu, and X. Zhang, "Model-free unified tracking and regulation visual servoing of wheeled mobile robots," *IEEE Trans. Control Syst. Technol.*, vol. 24, no. 4, pp. 1328–1339, Jul. 2016.

[20] T. P. Nascimento, C. E. T. Dórea, and L. M. G. Gonçalves, "Nonholonomic mobile robots' trajectory tracking model predictive control: A survey," *Robotica*, vol. 36, no. 5, pp. 676–696, May 2018.

[21] P. Harasim and M. Trojnacki, "State of the art in predictive control of wheeled mobile robots," *J. Autom., Mobile Robot. Intell. Syst.*, vol. 10, no. 1, pp. 34–42, Feb. 2016.

[22] M. T. Watson, D. T. Gladwin, T. J. Prescott, and S. O. Conran, "Dual-mode model predictive control of an omnidirectional wheeled inverted pendulum," *IEEE/ASME Trans. Mechatronics*, vol. 24, no. 6, pp. 2964–2975, Dec. 2019.

[23] L. Grüne and J. Pannek, *Nonlinear Model Predictive Control*. Berlin, Germany: Springer, 2017.

[24] G. Bai, Y. Meng, L. Liu, W. Luo, Q. Gu, and L. Liu, "Review and comparison of path tracking based on model predictive control," *Electronics*, vol. 8, no. 10, p. 1077, Sep. 2019.

[25] T. A. V. Teatro, J. M. Eklund, and R. Milman, "Nonlinear model predictive control for omnidirectional robot motion planning and tracking with avoidance of moving obstacles," *Can. J. Electr. Comput. Eng.*, vol. 37, no. 3, pp. 151–156, Summer 2014.

[26] I. Aliskan, "Optimized inverse nonlinear function-based Wiener model predictive control for nonlinear systems," *Arabian J. Sci. Eng.*, vol. 46, no. 10, pp. 10217–10230, Oct. 2021.

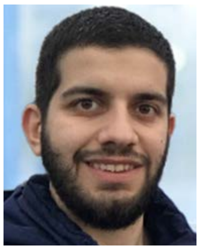
[27] S. Chang, X. Wei, F. Su, C. Liu, G. Yi, J. Wang, C. Han, and Y. Che, "Model predictive control for seizure suppression based on nonlinear auto-regressive moving-average Volterra model," *IEEE Trans. Neural Syst. Rehabil. Eng.*, vol. 28, no. 10, pp. 2173–2183, Oct. 2020.

[28] Z. Zeng, H. Lu, and Z. Zheng, "High-speed trajectory tracking based on model predictive control for omni-directional mobile robots," in *Proc. 25th Chin. Control Decis. Conf. (CCDC)*, May 2013, pp. 3179–3184.

[29] L. Armesto, V. Girbés, A. Sala, M. Zima, and V. Šmídl, "Duality-based nonlinear quadratic control: Application to mobile robot trajectory-following," *IEEE Trans. Control Syst. Technol.*, vol. 23, no. 4, pp. 1494–1504, Jul. 2015.

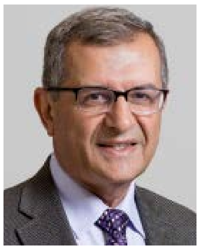
[30] S. Gros and M. Zanon, "Data-driven economic NMPC using reinforcement learning," *IEEE Trans. Autom. Control*, vol. 65, no. 2, pp. 636–648, Feb. 2020.

[31] M. Elsisi, "Optimal design of nonlinear model predictive controller based on new modified multitracker optimization algorithm," *Int. J. Intell. Syst.*, vol. 35, no. 11, pp. 1857–1878, Nov. 2020.
[32] Q. Hu, M. R. Amini, I. Kolmanovsky, J. Sun, A. Wiese, and J. B. Seeds, "Multihorizon model predictive control: An application to integrated power and thermal management of connected hybrid electric vehicles," *IEEE Trans. Control Syst. Technol.*, vol. 30, no. 3, pp. 1052–1064, May 2022.
[33] T. Nascimento and M. Saska, "Embedded fast nonlinear model predictive control for micro aerial vehicles," *J. Intell. Robot. Syst.*, vol. 103, no. 4, pp. 1–11, Dec. 2021.
[34] W. Qi, S. E. Ovur, Z. Li, A. Marzullo, and R. Song, "Multi-sensor guided hand gesture recognition for a teleoperated robot using a recurrent neural network," *IEEE Robot. Autom. Lett.*, vol. 6, no. 3, pp. 6039–6045, Jul. 2021.
[35] Y. Hu, H. Su, L. Zhang, S. Miao, G. Chen, and A. Knoll, "Nonlinear model predictive control for mobile robot using varying-parameter convergent differential neural network," *Robotics*, vol. 8, no. 3, p. 64, Jul. 2019.
[36] Z. Li, J. Deng, R. Lu, Y. Xu, J. Bai, and C.-Y. Su, "Trajectory-tracking control of mobile robot systems incorporating neural-dynamic optimized model predictive approach," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 46, no. 6, pp. 740–749, Jun. 2016.
[37] Y. Hu, H. Su, J. Fu, H. R. Karimi, G. Ferrigno, E. D. Momi, and A. Knoll, "Nonlinear model predictive control for mobile medical robot using neural optimization," *IEEE Trans. Ind. Electron.*, vol. 68, no. 12, pp. 12636–12645, Dec. 2021.
[38] B. Zhang, C. Zong, G. Chen, and B. Zhang, "Electrical vehicle path tracking based model predictive control with a Laguerre function and exponential weight," *IEEE Access*, vol. 7, pp. 17082–17097, 2019.
[39] J. A. Rossiter and L. Wang, "Exploiting Laguerre functions to improve the feasibility/performance compromise in MPC," in *Proc. 47th IEEE Conf. Decis. Control*, Dec. 2008, pp. 4737–4742.
[40] M. Ławryńczuk, "Nonlinear model predictive control for processes with complex dynamics: A parameterisation approach using Laguerre functions," *Int. J. Appl. Math. Comput. Sci.*, vol. 30, no. 1, pp. 35–46, 2020.
[41] J. A. Rossiter, L. Wang, and G. Valencia-Palomo, "Efficient algorithms for trading off feasibility and performance in predictive control," *Int. J. Control*, vol. 83, no. 4, pp. 789–797, Apr. 2010.
[42] J. Saeed, L. Wang, and N. Fernando, "Model predictive control of phase shift full-bridge DC–DC converter using Laguerre functions," *IEEE Trans. Control Syst. Technol.*, vol. 30, no. 2, pp. 819–826, Mar. 2022.
[43] L. Wang, "Discrete model predictive controller design using Laguerre functions," *J. Process Control*, vol. 14, no. 2, pp. 131–142, Mar. 2004.
[44] L. Wang, *Model Predictive Control System Design and Implementation Using MATLAB®*. London, U.K.: Springer, 2009.
[45] J. Chen, W. Zhan, and M. Tomizuka, "Autonomous driving motion planning with constrained iterative LQR," *IEEE Trans. Intell. Veh.*, vol. 4, no. 2, pp. 244–254, Jun. 2019.
[46] E. Todorov, "General duality between optimal control and estimation," in *Proc. 47th IEEE Conf. Decis. Control*, Dec. 2008, pp. 4286–4292.
[47] D. Pršić, V. Stojanović, and V. Dordević, "A constructive approach to teaching with Robotino®," in *Proc. 7th Int. Sci. Conf. Technics Inform. Educ.*, Serbia, Balkans, 2018, pp. 1–6.



MAHMOUD EL-SAYYAH received the B.S. and M.S. degrees in instrumentation, control and cyber-systems technologies form Lebanese University, Tripoli, Lebanon, in 2015 and 2017, respectively. He is currently pursuing the Ph.D. degree in electrical engineering with the Université de Québec en Abitibi-Témiscamingue (UQAT), QC, Canada. His research interests include nonlinear, adaptive, and robust control theories and their application to robotics and autonomous systems.

MOHAMAD R. SAAD (Senior Member, IEEE) received the Ph.D. degree in electrical engineering, control systems from the École Polytechnique de Montréal, in 2001. Then, he joined CAE Inc., Performance Systems Department, as a Flight Control Specialist. In 2006, he joined the School of Engineering, Université du Québec en Abitibi-Témiscamingue (UQAT). His research interest includes nonlinear control applied to robotic systems.



MAAROUF SAAD (Senior Member, IEEE) received the bachelor's and master's degrees in electrical engineering from the Ecole Polytechnique, Montreal, in 1982 and 1984, respectively, and the Ph.D. degree in electrical engineering from McGill University, in 1988. He joined the Ecole de Technologie Superieure, in 1987, where he is currently teaching control theory and robotics courses. His research interests include nonlinear control and optimization applied to robotics and autonomous systems.