A Receding Horizon Trajectory Tracking Strategy for Input-Constrained Differential-Drive Robots via Feedback Linearization

Cristian Tiriolo, Giuseppe Franzè, *Senior Member, IEEE*, and Walter Lucia, *Member, IEEE*

***Abstract*— This brief proposes a novel solution to the trajectory tracking control problem for input-constrained differential-drive robots. In particular, we develop a robust set-based receding horizon tracking scheme capable of dealing with state-dependent input constraints arising when the vehicle's dynamics are approached by a standard feedback linearization (FL) technique. First, offline, we characterize the worst case input constraint set and compute an admissible, although not optimal, controller. Then, online, we leverage the knowledge of the robot's orientation to enlarge the constraint set in a receding horizon fashion and, consequently, improve the tracking performance. Recursive feasibility and constraints' fulfillment are formally proven. The approach's effectiveness is experimentally validated on a Khepera IV differential-drive robot by comparing the control performance with several competitor schemes.**

***Index Terms*— Constrained control, differential-drive robots, feedback linearization (FL), receding horizon control (RHC).**

I. Introduction

AUTONOMOUS vehicles have gained increasing popularity due to their use in industrial, aerospace, and military robotics [1]. In all the applications, the vehicle's control system should be able to solve either a path-following problem or a trajectory tracking problem. Although the two problems are similar, the first requires a sequence of points that the robot tracks sequentially, while the second presupposes the knowledge of the trajectory timing law [2]. The main difficulty in accomplishing such tasks relies on the presence of nonholonomic constraints, i.e., the wheels cannot perform any lateral displacements [3]. Furthermore, due to the Brockett conditions, posture stabilization of such systems cannot be achieved by a static state-feedback law [4], [5]. Extensive research has been conducted on the trajectory tracking problem for wheeled mobile robots [6], and most of the proposed solutions range from Lyapunov-based design [7], [8], [9], adaptive laws [10], and feedback linearization (FL)-based schemes [5], [11]. In more recent years, thanks to the evolution of processing units and numerical algorithms to address optimization problems, several model predictive control (MPC) solutions have been developed; see [4], [12], [13], [14], [15], [16] and references therein. The nonlinear MPC schemes exploit accurate state predictions; however, recursive feasibility and stability properties are usually difficult to prove [14], [17]. Besides, the underlying computational demand might not be affordable in strict real-time contexts or when the robot's computation capabilities are limited [15], [18]. As the latter issue is concerned, an alternative solution, based on a neural network, has been presented in [16] to obtain a computationally affordable algorithm. Conversely, linearized MPC approaches, e.g., [4], [12], [19], have the important advantage of drastically reducing computational burdens at the expense of more conservative control performance.

FL is a popular technique used to recast the nonlinear dynamical models into equivalent linear descriptions by means of a proper change in variables and control input vector [20]. In [21] and [22], FL has been combined with MPC formulations. There, the authors have used FL to obtain exact linear state predictions for an unconstrained vehicle model. Along similar lines is the approach presented in [23], where an unconstrained FL-MPC scheme is derived to address the trajectory tracking problem for a wheeled inverted pendulum. In [24], the nonlinear model of an electric power wheelchair is linearized via transverse FL. Moreover, the inputs' constraints, arising from nonlinear mapping operations between the vehicle and the linear model, are exploited for input prediction purposes. Consequently, the resulting MPC optimization problem is nonlinear and non-convex. This literature analysis puts in light that the combined use of FL arguments and MPC philosophy leads to computationally affordable convex optimizations only for unconstrained models. On the other hand, input constraints are always converted into nonlinear and state-dependent constraints [25], leading to nonlinear MPC formulations. As a matter of fact, it has been shown that non-convex MPC formulations can be avoided if sequential approximations of the constraint set are performed along the prediction horizon. Unfortunately, this strategy turns to be very conservative even for small values of the prediction horizon; see, e.g., [25], [26], [27].

In this brief, we propose a novel trajectory tracking controller for input-constrained differential-drive robots. Specifically, a linear receding horizon control (RHC) strategy, based on set-theoretic arguments, is developed for efficiently dealing with state-dependent input constraints arising when the robot dynamics are linearized using a standard kinematic input–output FL technique [28]. First, the state-dependent input constraints' set is characterized. Then, by considering worst case constraints; realization and the receding horizon controller of [29], [30], we offline design a guaranteed solution to the constrained reference tracking problem. During online operations, by exploiting the knowledge of the robot's orientation and, as a consequence, of the exact input constraint set,

Manuscript received 8 July 2022; accepted 28 October 2022. This work was supported in part by the Natural Sciences and Engineering Research Council of Canada (NSERC). Recommended by Associate Editor S. Galeani. *(Corresponding author: Walter Lucia.)*

Cristian Tiriolo and Walter Lucia are with the Concordia Institute for Information Systems Engineering (CIISE), Concordia University, Montreal, QC H3G 1M8, Canada (e-mail: cristian.tiriolo@concordia.ca; walter.lucia@concordia.ca).

Giuseppe Franzè is with the DIMEG Department, University of Calabria, Rende, 87036 Cosenza, Italy (e-mail: giuseppe.franze@unical.it).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TCST.2022.3219298.

Digital Object Identifier 10.1109/TCST.2022.3219298

we appropriately modify the offline solution to significantly reduce the intrinsic conservativeness.

A preliminary version of such a controller has been discussed by Tiriolo et al. in [31]. Nevertheless, substantial differences and improvements can be underlined. In [31], the proposed controller requires an inner and conservative approximation of the state-dependent input constraint set, while here this requirement has been relaxed with a consequence tangible improvement in terms of control performance (see, e.g., the comparisons provided in Section IV); the proposed tracking controller does not require any ad hoc waypoint switching mechanism to ensure feasibility retention. Moreover, the worst case input constraint set has been analytically characterized, and the controller's design and its properties have been formalized and proved (no proofs are given in [31]); the effectiveness of the proposed solution has been validated on the Khepera IV robot and the strategy experimentally contrasted with competitor schemes.

II. Differential-Drive Robot and Problem Formulation

In the sequel, we consider a differential-drive vehicle equipped with two rear drive wheels and an (optional) front caster for body support. Such a robot belongs to the class of nonholonomic wheeled vehicles and, as a consequence, the wheels can perform only pure rolling motions [28]. The robot's kinematics can be described in the continuous-time domain by the following set of nonlinear differential equations [32]:

$$\begin{aligned}\dot{x}_c(t) &= \frac{R}{2}(\omega_R(t)+\omega_L(t))\cos\theta(t)\\ \dot{y}_c(t) &= \frac{R}{2}(\omega_R(t)+\omega_L(t))\sin\theta(t)\\ \dot{\theta}(t) &= \frac{R}{D}(\omega_R(t)-\omega_L(t))\end{aligned} \tag{1}$$

where [see Fig. 1(a)]: $R \in \mathbb{R}$ is the radius of the wheels, $D \in \mathbb{R}$ is the wheel's axis length, $\omega_R, \omega_L \in \mathbb{R}$ are the right and left wheels' angular velocities, respectively, and $p_c = [x_c, y_c]^T, \theta$ are the 2-D coordinates of the center of mass of the robot and its orientation, respectively. The nonholonomic constraint acting on the vehicle is described by the differential equation $\dot{x}_c \sin\theta = \dot{y}_c \cos\theta$. The physical limitations on the angular velocities of (1) can be described by means of a box-like input set $\mathcal{U}_d \subset \mathbb{R}^m$ [see Fig. 2(a)], i.e., $[\omega_R, \omega_L]^T \in \mathcal{U}_d$, where

$$\mathcal{U}_d = \{[\omega_R, \omega_L]^T \in \mathbb{R}^2 : H_d[\omega_R, \omega_L]^T \leq 1\} \tag{2}$$

$$H_d = \begin{bmatrix} \frac{-1}{\bar{\Omega}} & 0 & \frac{1}{\bar{\Omega}} & 0\\ 0 & \frac{-1}{\bar{\Omega}} & 0 & \frac{1}{\bar{\Omega}}\end{bmatrix}^T \tag{3}$$

and $\bar{\Omega}, \in \mathbb{R}$ is the maximum admissible angular velocities.

Problem 1: Consider the input-constrained differential-drive robot model (1)–(3) and a bounded trajectory $r(t)$. Design a feedback control law $[\omega_R(t), \omega_L(t)]^T = \phi([p_c(t)^T, \theta(t)]^T, r(t))$ such that the tracking error $\xi(t) = r(t) - p_c(t)$ is bounded and $[\omega_R(t), \omega_L(t)]^T \in \mathcal{U}_d, \forall t \geq 0$.

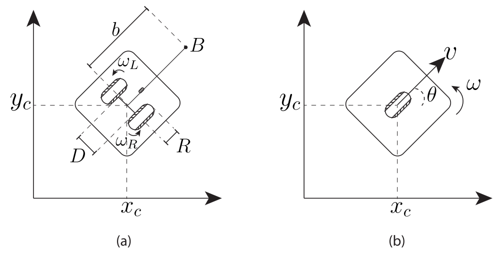

Fig. 1. (a) Differential drive. (b) Unicycle.

III. Proposed Solution

This section is organized as follows. The robot model is recast into a unicycle model and linearized via an input–output FL (Section III-A); the input constraint set associated with the feedback linearized model is characterized and its worst case realization defined (Section III-B); the trajectory tracking problem is formulated as a discrete-time RHC tracking problem and the proposed solution formally derived; the section ends by collecting all the developments into a computable algorithm.

A. Linearized Vehicle Model via FL

Consider the vehicle's physical parameters R (wheels radius) and D (rear axis length). Then, by means of the following coordinate transformation:

$$\begin{bmatrix} v(t)\\ \omega(t)\end{bmatrix} = T\begin{bmatrix}\omega_R(t)\\ \omega_L(t)\end{bmatrix}, \quad T = \begin{bmatrix}\frac{R}{2} & \frac{R}{2}\\ \frac{R}{D} & -\frac{R}{D}\end{bmatrix} \tag{4}$$

the differential-drive model (1) is recast into an equivalent unicycle kinematic model as follows:

$$\begin{aligned}\dot{x}_c(t) &= v(t)\cos(\theta(t))\\ \dot{y}_c(t) &= v(t)\sin(\theta(t))\\ \dot{\theta}(t) &= \omega(t)\end{aligned} \tag{5}$$

where the inputs v and ω denote the linear and angular velocities of the vehicle, respectively [see Fig. 1(b)].

In [28, Sec. 3.3], it has been proven that the unicycle model (5) is feedback linearizable. In fact, by defining a scalar $b > 0$ and two new outputs

$$z_1 = x_c + b\cos\theta, \quad z_2 = y_c + b\sin\theta \tag{6}$$

representing the Cartesian coordinates of a virtual point **B** displaced along the translational directions of the robot [see Fig. 1(b)], it is possible to use the following state-feedback law:

$$\begin{bmatrix} v(t)\\ \omega(t)\end{bmatrix} = T_{\mathrm{FL}}(\theta)\begin{bmatrix}u_1(t)\\ u_2(t)\end{bmatrix}, \quad T_{\mathrm{FL}}(\theta) = \begin{bmatrix}\cos\theta & \sin\theta\\ \frac{-\sin\theta}{b} & \frac{\cos\theta}{b}\end{bmatrix} \tag{7}$$

and recast the unicycle dynamics (5) as a two-single integrator model

$$\dot{z}_1(t) = u_1(t), \quad \dot{z}_2(t) = u_2(t) \tag{8}$$

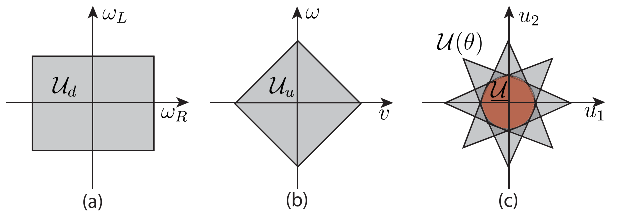

Fig. 2. Input constraints. (a) Differential drive. (b) Unicycle. (c) Feedback linearized model.

where $u(t) = [u_1(t), u_2(t)]^T \in \mathbb{R}^2$ is the control input vector for the **B** point model. Moreover, the internal decoupled dynamics is $\dot{\theta}(t) = ((-\sin(\theta(t))u_1(t) + \cos(\theta(t))u_2(t))/b)$.

Remark 1: A linear tracking controller for (8) allows the point **B** to track any reference trajectory with a stable internal dynamics for $\theta(t)$ [28], [33, Sec. 3.3].

B. Input Constraints Mapping

Under the transformation (4) and the unicycle model (5), the input constraint set (2)–(3) is mapped into a rhombus-like set $\mathcal{U}_u$ on the liner angular velocities of the robot [see Fig. 2(b)], i.e., $[v, \omega]^T \in \mathcal{U}_u$, where

$$\mathcal{U}_u = \{[v, \omega]^T \in \mathbb{R}^2 : H_u[v, \omega]^T \leq 1\}, \quad H_u = H_d T^{-1}. \tag{9}$$

Moreover, by applying (7)–(9), the set $\mathcal{U}_u$ rotates according to the robot's orientation θ. As a consequence, the linearized model (8) is subject to state-dependent input constraints defined as rhombus-like sets $\mathcal{U}(\theta)$ [see Fig. 2(c)], where

$$\mathcal{U}(\theta) = \{[u_1, u_2]^T \in \mathbb{R}^2 : H(\theta)[u_1, u_2]^T \leq 1\}$$
$$H(\theta) = H_u T_{\text{FL}}(\theta) = \begin{bmatrix} \frac{D\sin\theta - 2\cos\theta b}{2\bar{\Omega}Rb} & \frac{-D\cos\theta - 2\sin\theta b}{2\bar{\Omega}Rb} \\ \frac{-D\sin\theta - 2\cos\theta b}{2\bar{\Omega}Rb} & \frac{D\cos\theta - 2\sin\theta b}{2\bar{\Omega}Rb} \\ \frac{-D\sin\theta + 2\cos\theta b}{2\bar{\Omega}Rb} & \frac{D\cos\theta + 2\sin\theta b}{2\bar{\Omega}Rb} \\ \frac{D\sin\theta + 2\cos\theta b}{2\bar{\Omega}Rb} & \frac{-D\cos\theta + 2\sin\theta b}{2\bar{\Omega}Rb} \end{bmatrix}. \tag{10}$$

Note that although (10) is time-varying, a worst case time-invariant input constraint $\underline{\mathcal{U}}$ set such that $\underline{\mathcal{U}} \subset \mathcal{U}(\theta), \forall\theta$, can be defined as follows:

$$\underline{\mathcal{U}} := \bigcap_{\theta=0}^{2\pi} \mathcal{U}(\theta). \tag{11}$$

Lemma 1: The worst case input constraint set (11) is equal to the following ball set:

$$\underline{\mathcal{U}}(r_u) = \{u \in \mathbb{R}^2 | u^T u \leq r_u^2\}, r_u = \frac{2\bar{\Omega}Rb}{\sqrt{4b^2 + D^2}}. \tag{12}$$

Proof: The vertices $V_i(\theta), i = 1, \ldots, 4$ of the polyhedron (10) can be analytically computed intersecting the four hyperplanes shaped by $H_{[i,:]}(\theta)[u_1, u_2]^T = 1$, $i = 1, \ldots 4$, (with $H_{[i,:]}(\theta)$ denoting the ith row of $H(\theta)$), obtaining

$$V_1(\theta) = \left[-\bar{\Omega}R\cos\theta, -\bar{\Omega}R\sin\theta\right]^T, \quad V_3(\theta) = -V_1(\theta)$$
$$V_2(\theta) = \left[\frac{2\bar{\Omega}Rb\sin\theta}{D}, -\frac{2\bar{\Omega}Rb\cos\theta}{D}\right]^T, \quad V_4(\theta) = -V_2(\theta). \tag{13}$$

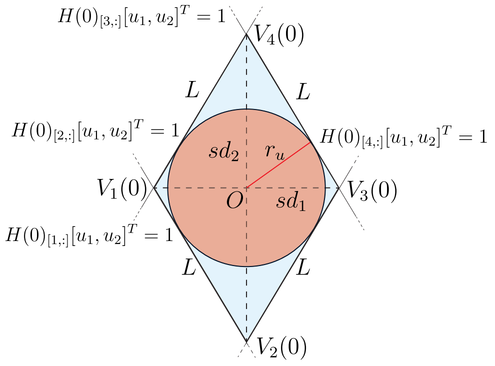
$H(0)_{[3,:]}[u_1, u_2]^T = 1$
$V_4(0)$
L
L
$H(0)_{[2,:]}[u_1, u_2]^T = 1$
sd_2
r_u
$H(0)_{[4,:]}[u_1, u_2]^T = 1$
$V_1(0)$
O
sd_1
$V_3(0)$
$H(0)_{[1,:]}[u_1, u_2]^T = 1$
L
L
$V_2(0)$

Fig. 3. Input constraint set $\mathcal{U}(0)$ and inscribed circle $\underline{\mathcal{U}}(r_u)$.

Then, by computing from (13) the lengths $L_i, i = 1, \ldots, 4$ of each side of $\mathcal{U}(\theta)$, it is straightforward to verify that they are all equal and independent of θ, i.e.,

$$L_i = L = \frac{\bar{\Omega}R}{D}\sqrt{4b^2 + D^2}, \quad i = 1, \ldots, 4. \tag{14}$$

Therefore, the transformation (7) defines a rhombus set $\mathcal{U}(\theta)$ centered in the origin and rigidly rotating into the plane. As a consequence, the set $\underline{\mathcal{U}}$ is equivalent to the biggest circle inscribed into $\mathcal{U}(\theta)$ for any θ. By considering for simplicity and w.l.o.g. only $\theta = 0$ (see Fig. 3), the radius $r_u \in \mathbb{R}$ of the biggest circle within $\mathcal{U}(0)$ can be computed resorting to simple geometric arguments as follows:

$$r_u = \frac{sd_1 sd_2}{\sqrt{sd_1^2 + sd_2^2}} = \frac{2\bar{\Omega}Rb}{\sqrt{4b^2 + D^2}} \tag{15}$$

where $sd_1 = [1, 0]V_3(0)$ and $sd_2 = [0, 1]V_4(0)$ are the length of the two semi-diagonals of $\mathcal{U}(0)$. □

C. Discrete-Time Receding Horizon Controller

Let $T_s > 0$ be a sufficiently small sampling time, the model (8) can be discretized by resorting to the forward Euler method, obtaining

$$z(k+1) = Az(k) + Bu(k) \tag{16}$$

where $k \in \mathbb{Z}_+ := \{0, 1, \ldots\}$ denotes the discrete-time instants, $A = I_{2\times2}$, and $B = T_s I_{2\times2}$. Note that the same state-space description is achieved when the zero-order hold discretization method is used.

Property 1: The input–output FL (6)–(8) and forward Euler discretization commute when the unicycle robot model (5) is considered.

Proof: By resorting to the forward Euler discretization method, the discrete-time characterization of (5) is as follows:

$$\begin{aligned} x_c(k+1) &= x_c(k) + T_s v(k)\cos(\theta(k)) \\ y_c(k+1) &= y_c(k) + T_s v(k)\sin(\theta(k)) \\ \theta(k+1) &= \theta(k) + T_s\omega(k). \end{aligned} \tag{17}$$

By differentiating the outputs $z_1(t)$ and $z_2(t)$ in (6)

$$\begin{aligned} \dot{z}_1(t) &= \dot{x}_c(t) - b\sin(\theta(t))\dot{\theta}(t) \\ \dot{z}_2(t) &= \dot{y}_c(t) + b\cos(\theta(t))\dot{\theta}(t) \end{aligned}$$

one obtains under discretization arguments

$$\frac{z_1(k+1)-z_1(k)}{T_s} = \frac{x_c(k+1)-x_c(k)}{T_s} - b\sin(\theta(k))\frac{\theta(k+1)-\theta(k)}{T_s}$$
$$\frac{z_2(k+1)-z_2(k)}{T_s} = \frac{y_c(k+1)-y_c(k)}{T_s} + b\cos(\theta(k))\frac{\theta(k+1)-\theta(k)}{T_s}.$$

Then, by substituting $x_c(k+1)$, $y_c(k+1)$, and $\theta(k+1)$ with the right-hand sides of (17), the resulting discrete-time evolutions of z_1 and z_2 are

$$z_1(k+1) = z_1(k) + T_s v(k)\cos(\theta(k)) - T_s b\sin(\theta(k))\omega(k)$$
$$z_2(k+1) = z_2(k) + T_s v(k)\sin(\theta(k)) + T_s b\cos(\theta(k))\omega(k)$$

the compact form of which is

$$\begin{bmatrix} z_1(k+1) \\ z_2(k+1) \end{bmatrix} = \begin{bmatrix} z_1(k) \\ z_2(k) \end{bmatrix} + T_s T_{\mathrm{FL}}^{-1}(\theta(k)) \begin{bmatrix} v(k) \\ \omega(k) \end{bmatrix}. \quad (18)$$

Finally, using the input transformation (7), the discrete-time model (18) becomes equal to (16). Hence, input–output linearization and discretization commute. □

By denoting with $r^z(k)$ the reference signal for the **B** point model [i.e., obtained applying the transformation (6) to $r(k)$], and by considering (10) and (16), a solution to Problem 1 can be, in principle, achieved via the following constrained optimization problem over a prediction horizon N_p:

$$\min_{\{u(k+j|k)\}} \sum_{j=0}^{N_p} ||\xi(k+j|k)||_{Q_x} + ||u(k+j|k)||_{Q_u}$$
$$\text{s.t. } z(k+j+1|k) = Az(k+j|k) + Bu(k+j)$$
$$u(k+j|k) \in \mathcal{U}(\theta(k+j|k))$$
$$j = 0, \ldots, N_p - 1 \quad (19)$$

where $\xi(k) = r^z(k) - z(k) = r(k) - p_c(k)$ is the tracking error, and $Q_x = Q_x^T > 0$ and $Q_u = Q_u^T > 0$ are two weight matrices. Moreover, $z(k+j|k)$, $\theta(k+j|k)$ are the jth step ahead state and orientation prediction, respectively. Similarly, $u(k+j|k)$ is the control move at $k+j$.

Remark 2: Note that optimization (19) is nonlinear and non-convex. The latter is due to the presence of the state-dependent input constraint $\mathcal{U}(\theta(k+j|k))$ whose prediction involves, at each step, the nonlinear transformation (7). As a consequence, although appealing, optimization (19) does not present any tangible advantage when compared with the nonlinear MPC counterparts [14], [15], [17]. On the other hand, a viable approach would require to robustly approximate the state-dependent input constraint set over the prediction horizon N_p. Unfortunately, this leads to a computationally demanding solution that becomes very conservative even for small values of the prediction horizon, see, e.g., [26], [27] and references therein. Moreover, for a small horizon, closed-loop stability might not be ensured [34]. □

In the sequel, we tackle the constrained trajectory tracking problem by properly adapting the infinite horizon ($N_p \to \infty$) receding horizon controller developed in [29] and [30], where an upper bound of the cost function in (19) is minimized by means of a constant state-feedback control law. First, by considering the worst case realization (12) of the state-dependent input constraint set $\mathcal{U}(\theta)$, we offline derive a guaranteed, although conservative, solution. Then, during the online phase, the offline controller is updated by exploiting the knowledge of the current robot orientations $\theta(k)$ and proper set inclusions. This allows to drastically reduce its conservativeness and preserve the recursive feasibility property.

Assumption 1: The robot is equipped with an onboard vision module and an online trajectory planner providing a reference signal $r(k)$ within a vision radius $d_r > 0$. The latter implies that $r(k) - p_c(k) \in \mathcal{B}(d_r), \forall k$, with $\mathcal{B}(d_r) = \{\xi \in \mathbb{R}^2 : \xi^T B^{-1}(d_r)\xi \leq 1\}$, $B(d_r) = \begin{bmatrix} d_r^2 & 0 \\ 0 & d_r^2 \end{bmatrix}$.

Definition 1: A set $\Sigma \subset \mathbb{R}^n$ is said control invariant (CI) for (16) under and admissible control law $u(k) \in \mathcal{U}(\theta)$ if $\forall z(k) \in \Sigma \Rightarrow Az(k) + Bu(k) \in \Sigma$ [35].

1) Offline Solution:

Proposition 1: Consider the vehicle linearized model (16) and the worst case input constraint (12). If the following semidefinite programming problem

$$\left[Q_0^*, Y_0^*, \gamma_0^*\right] = \arg\min_{\gamma,Q,Y} \gamma \quad s.t. \quad (20a)$$

$$\begin{bmatrix} 1 & \xi^T(0) \\ \xi(0) & Q \end{bmatrix} \geq 0, \quad Q = Q^T > 0 \quad (20b)$$

$$\begin{bmatrix} Q & QA^T + Y^T B^T & Q Q_x^{\frac{1}{2}} & Y^T Q_u^{\frac{1}{2}} \\ AQ + BY & Q & 0 & 0 \\ Q_x^{\frac{1}{2}} Q & 0 & \gamma I & 0 \\ Q_u^{\frac{1}{2}} Y & 0 & 0 & \gamma I \end{bmatrix} \geq 0 \quad (20c)$$

$$\begin{bmatrix} r_u^2 I & Y \\ Y^T & Q \end{bmatrix} \geq 0 \quad (20d)$$

$$Q \geq B(d_r) \quad (20e)$$

admits a solution and $r(k)$ is a reference complying with Assumption 1, then the state-feedback control law

$$[\omega_r(k), \omega_l(k)] = -T^{-1} T_{\mathrm{FL}}(\theta(k)) Y_0^* Q_0^{*-1} \xi(k) \quad (21)$$

solves the constrained Problem 1. Moreover, $\forall k \geq 0, \xi(k)$ belongs to the ellipsoidal CI set $\mathcal{E}_0 = \{\xi \in \mathbb{R}^2 : \xi^T Q_0^{*-1} \xi \leq 1\}$.

Proof: Initially, consider the scenario $r^z(k) = 0, \forall k$ (i.e., a regulation problem). As shown in [29], the state-feedback control law

$$u(k) = -Y_0^* Q_0^{*-1} \xi(k) \quad (22)$$

obtained solving the optimization problems (20a)–(20d) and guarantees closed-loop asymptotic stability of (16) and worst case input constraint (12) fulfillment for any $\xi(0) \in \mathcal{E}_0$, with $\mathcal{E}_0$ CI. By noting that $(\bar{r}^z, 0_2)$ is the equilibrium pair for (16) under any constant admissible reference $r^z(k) = \bar{r}^z$ [36, Sec. 2], the tracking error dynamics have the same structure of (16): $\xi(k+1) = A\xi(k) + Bu(k)$. Moreover, since the equilibrium input is always zero and no state constraints act on (16), then the tracking error dynamics are also subject to the same worst case input constraint (12). Consequently, an admissible setpoint tracking control law for (16) is given by the solution of (20a)–(20d) if $\xi(k) = \bar{r}^z - z(k) \in \mathcal{E}_0$ (i.e., the error signal remains confined within the controller's domain). By generalizing such a result to a generic reference signal $r(k)$ complying with Assumption 1, the admissibility condition $\xi(k) \in \mathcal{E}_0$ is fulfilled $\forall k$ if $\mathcal{E}_0 \supseteq \mathcal{B}(d_r)$. The latter translates into a further linear matrix inequality (LMI) constraint, i.e., $Q \geq B(d_r)$ [see (20e)] that must be added to (20). Finally, since the control law (22) ensures bounded tracking error for the linearized model (16) and the internal dynamics of $\theta(t)$ is stable (see Remark 1), the nonlinear feedback control law (21), obtained applying the transformations (4) and (7)–(22), solves Problem 1. □

2) Online Receding Horizon Controller:

Proposition 2: If (20) admits a solution at $k = 0$ and $r(k)$ is an admissible reference complying with Assumption 1, then the following semidefinite programming problem:

$$[Q_k^*,\, Y_k^*,\, \gamma_k^*] = \arg \min_{\gamma_k, Q_k, Y_k} \gamma_k \; s.t. \tag{23a}$$

$$\begin{bmatrix} 1 & \xi(k)^T \\ \xi(k) & Q_k \end{bmatrix} \geq 0, \quad Q = Q^T > 0 \tag{23b}$$

$$\begin{bmatrix} Q_k & Q_k A^T + Y_k^T B^T & Q_k Q_x^{\frac{1}{2}} & Y_k^T Q_u^{\frac{1}{2}} \\ A Q_k + B Y_k & Q_k & 0 & 0 \\ Q_x^{\frac{1}{2}} Q_k & 0 & \gamma_k I & 0 \\ Q_u^{\frac{1}{2}} Y_k & 0 & 0 & \gamma_k I \end{bmatrix} \geq 0 \tag{23c}$$

$$\begin{bmatrix} Q_k & Y_k^T H^T(\theta(k))_{[i,:]} \\ H(\theta(k))_{[i,:]} Y_k & 1 \end{bmatrix} \geq 0, \quad i = 1, \ldots, 4 \tag{23d}$$

$$Q_k \leq Q_0^* \tag{23e}$$

admits a solution $\forall k > 0$ and the receding horizon state-feedback control law

$$[\omega_R(k), \omega_L(k)]^T = -T^{-1} T_{\mathrm{FL}}(\theta(k)) Y_k^* Q_k^{*-1} \xi(k) \tag{24}$$

solves the constrained Problem 1.

Proof: First, the LMI (23d) is equivalent to the polyhedral input constraint $\mathcal{U}(\theta(k))$ [29], [30]. Therefore, the RHC law $u(k) = -Y_k^* Q_k^{*-1} \xi(k)$, obtained solving the optimization problems (23a)–(23d) with a constant input constraint set $\mathcal{U}(\theta(k)) = \underline{\mathcal{U}}$, $\forall k$, solves the regulation problem for (16), with $\mathcal{E}_k = \{\xi \in \mathbb{R}^2 : \xi^T Q_k^{*-1} \xi \leq 1\}$ the CI region [30]. Conversely, if $\theta(k)$ varies at each time instant, it is no longer ensured that the one-step evolution $\xi(k+1)$ represents an admissible initial condition for (23a)–(23d) at $k+1$. However, such a drawback can be overcome by forcing the invariant region $\mathcal{E}_k$ to be included in the worst case domain of attraction $\mathcal{E}_0$ [obtained solving (20)]. This translates into the additional LMI constraint (23e) ensuring recursive feasibility of (23) $\forall k$, i.e., the offline solution $[Q_0^*, Y_0^*, \gamma_0^*]$ is always an admissible, although not optimal, solution of the problem. Finally, by resorting to the same arguments used in the proof of Proposition 1, the control law (24) solves Problem 1. □

Remark 3: At each k, the control law (24) exploits the input constraint set $\mathcal{U}(\theta) \supset \underline{\mathcal{U}}$ corresponding to the mapping of the differential-drive input constraint $\mathcal{U}_d$ into the feedback linearized robot model for the current orientation $\theta(k)$. Consequently, $\forall k$, the input constraint set (23d) is a subset of (20d). □

Algorithm 1 Reference Tracking—RHC Strategy

```
Configuration:
  • Robot parameters and planner radius: $\bar{\Omega}, R, D$ and $d_r$
  • LQ cost matrices and B point distance: $Q_x, Q_u, b > 0$
Offline:
  1) Compute $\underline{\mathcal{U}}(r_u)$ as in (12) and solve opt. (20) with
     $\xi(0) = r(0) - p_c(0)$. Store $Q_0^*$
Online ($\forall k$):
  1) Obtain $r(k)$ from the trajectory planner, compute
     $T_{FL}(\theta)$ using (7), and $H(\theta)$ using (10)
  2) Solve opt. (23) with $\xi(k) = r(k) - p_c(k)$
  3) Compute $\omega_R(k), \omega_L(k)$ using (24)
```

Fig. 4. Khepera IV differential-drive robot. (a) Top and (b) bottom views.

Finally, Algorithm 1 summarizes all the above developments.

IV. Experimental Results

In this section, the experimental results, obtained using a Khepera IV robot, are presented to show the effectiveness of the proposed tracking controller and compare it with the solutions developed in [4], [5], [12], [31], and [9]. In particular, [4] and [12] are two popular linear MPC-based controllers developed on the constrained and unconstrained robot's model, respectively; [5] is the landmark control solution for unconstrained robots controlled via FL; [9] is a popular unconstrained Lyapunov-based solution; [31] is the preliminary RHC constrained controller developed by the same authors of this brief.

A. Setup

Khepera IV (Fig. 4) is a differential-drive robot produced by the K-Team Corporation. This wheeled robot presents two independent wheels driven by two DC brushed motors equipped with incremental encoders. Each wheel has a radius $R = 0.021$ m, and the rear axis length is $D = 0.0884$ m. Each wheel has a maximum speed of 1200 steps/s which corresponds to a linear velocity of 0.813 m/s and an angular velocity of 38.7143 rad/s. Nevertheless, in all the performed experiments, the maximum speed of the robot's wheels has been limited to $|\omega_R| = |\omega_L| \leq 250$ steps/s (8.0655 rad/s) to avoid unmodeled dynamic effects and wheel slips. A sampling time $T_s = 0.15$ s is considered, and the robot's position has been estimated resorting to odometric calculations [28]. Moreover, Bluetooth has been used to communicate between the robot and a Linux Ubuntu 16 computer, equipped with an

Intel Core I7 8750H processor, 16 GB of RAM, and running MATLAB R2020a.

1) Configuration of the Proposed Controller: To implement Algorithm 1, the following parameters have been used: $b = 0.1\ m$, $Q_x = I_2$, and $Q_u = 0.001\ I_2$. Moreover, LMIs' optimizations (20) and (23) have been solved using the MATLAB built-in LMI solver *"mincx."*

2) Configuration of the Competitor Schemes: Each competitor scheme has been tuned to obtain the best tracking results in the conducted experiments. Since the algorithms in [5] and [9] do not take into account the wheels' velocity limitations, a saturation is enforced whenever necessary.

For further details on the meaning of the following parameters, one car refer to [4], [5], [9], [12], and [31] as follows.

1) The MPC solutions in [4] and [12] have been tuned to use a prediction horizon $N_p = 10$ with a cost function as in (19), where $Q_x = \begin{bmatrix} I_2 & 0 \\ 0 & 0.05 \end{bmatrix}$ and $Q_u = 0.001\ I_2$. The optimization problem resulting from [4] has been solved in MATLAB using the built-in quadratic programming solver *"quadprog."* Whereas the unconstrained optimization in [12] has been analytically solved. In addition, a reference matrix $A_r = 0.65 I_3$ has been selected for the algorithm of [12].
2) The proportional-derivative controller in [5] has been implemented with $k_{p1} = k_{p2} = 1$, and $k_{d1} = k_{d2} = 0.7$.
3) The Lyapunov-based control law in [9] has been implemented with $k = 5$, $k_x = 0.5$, $k_s = 5$, $n = 2$, $a = 7$.
4) The receding horizon controller in [31] has been implemented using the same parameters of the proposed controller.

3) Evaluation of the Tracking Performance: The tracking performance has been evaluated by resorting to four error-based indices described in [13]. By defining the instantaneous tracking error as $e(t) = ((x_r(t) - x_c(t))^2 + (y_r(t) - y_c(t))^2)^{1/2}$, and by denoting with T_f the duration of the experiment, the used indices are: i) integral absolute error (IAE) ($\int_0^{T_f} |e(t)|dt$); ii) integral square error (ISE) ($\int_0^{T_f} e(t)^2 dt$); iii) integral time-weighted absolute error (ITAE) ($\int_0^{T_f} t|e(t)|dt$); and iv) integral time squared error (ITSE) ($\int_0^{T_f} te(t)^2 dt$).

4) Trajectories: Two trajectories $r(k)$ have been considered as follows.

1) An eight-shaped trajectory, where

$$r(t) = \begin{bmatrix} x_r(t) \\ y_r(t) \end{bmatrix} = \begin{bmatrix} 0.6 \sin\left(\frac{t}{3.5}\right) \\ 0.6 \sin\left(\frac{t}{7}\right) \end{bmatrix}, \quad t \in [0, 44].$$

2) A circular trajectory, where

$$r(t) = \begin{bmatrix} x_r(t) \\ y_r(t) \end{bmatrix} = \begin{bmatrix} 0.3 \sin\left(\frac{t}{3.8}\right) \\ 0.3 \cos\left(\frac{t}{3.8}\right) \end{bmatrix}, \quad t \in [0, 24].$$

Moreover, the vehicle initial's state $[p_c^T(0), \theta(0)]^T$ is $[0.6,\ 0,\ \pi]^T$ for the eight-shaped trajectory and $[0.3,\ 0.3,\ \pi]^T$ for the circular reference.

TABLE I

EXPERIMENT 1 (EIGHT-SHAPED TRAJECTORY): TRACKING PERFORMANCE

	IAE	ISE	ITAE	ITSE
Proposed	2,502	0,390	38,026	2,248
[4]	3,084	0,761	36,805	3,436
[5]	2,002	0,394	16,952	1,042
[9]	2,163	0,454	17,454	1,145
[12]	3,467	0,467	63,512	4,711
[31]	13,505	4,765	317,567	113,990

TABLE II

EXPERIMENT 1 (CIRCULAR TRAJECTORY): TRACKING PERFORMANCE

	IAE	ISE	ITAE	ITSE
Proposed	2,160	0,390	15,913	1,305
[4]	2,298	0,462	14,397	1,330
[5]	2,121	0,408	13,831	1,269
[9]	3,692	0,781	30,919	4,149
[12]	2,634	0,542	21,046	1,855
[31]	8,989	3,625	123,516	53,107

TABLE III

EXPERIMENT 2 (EIGHT-SHAPED TRAJECTORY): TRACKING PERFORMANCE

	IAE	ISE	ITAE	ITSE
Proposed	2,502	0,390	38,026	2,248
[4]	9,452	2,458	162,222	31,162
[5]	6,631	1,247	122,173	17,206
[9]	2,163	0,454	17,454	1,145
[12]	27,730	26,152	707,647	707,830
[31]	13,505	4,765	317,567	113,990

TABLE IV

EXPERIMENT 2 (CIRCULAR TRAJECTORY): TRACKING PERFORMANCE

	IAE	ISE	ITAE	ITSE
Proposed	2,160	0,390	15,913	1,305
[4]	6,298	1,768	72,313	19,100
[5]	6,220	1,691	73,415	18,607
[9]	3,692	0,781	30,919	4,149
[12]	29,949	50,840	486,219	933,627
[31]	8,989	3,625	123,516	53,107

TABLE V

AVERAGE TRACKING PERFORMANCE INDICES

	IAE	ISE	ITAE	ITSE
Proposed	2,331	0,390	26,970	1,776
[4]	5,283	1,362	71,434	13,757
[5]	4,243	0,935	56,593	9,531
[9]	2,927	0,617	24,186	2,647
[12]	15,945	19,500	319,606	412,006
[31]	11,247	4,195	220,541	83,549

B. Results

The obtained results are collected in Figs. 5 and 6 and Tables I–IV. Moreover, the additional Table V shows an averaged comparison based on all the performed experiments. For the interested reader, videos of the performed experiments can be found at the following weblink: https://www.youtube.com/playlist?list=PLh-6B_s-jPuT8RTDOJM96GXu4y1IBIeoC.

Two experiments have been performed as follows.

1) Experiment 1: The operating scenario prescribes that the reference trajectory and its first and second derivatives are available for any desired prediction horizon. The results and comparisons are shown in Fig. 5 (for the eight-shaped trajectory) and Tables I and II. The obtained results show that for both the trajectories, the proposed controller achieves

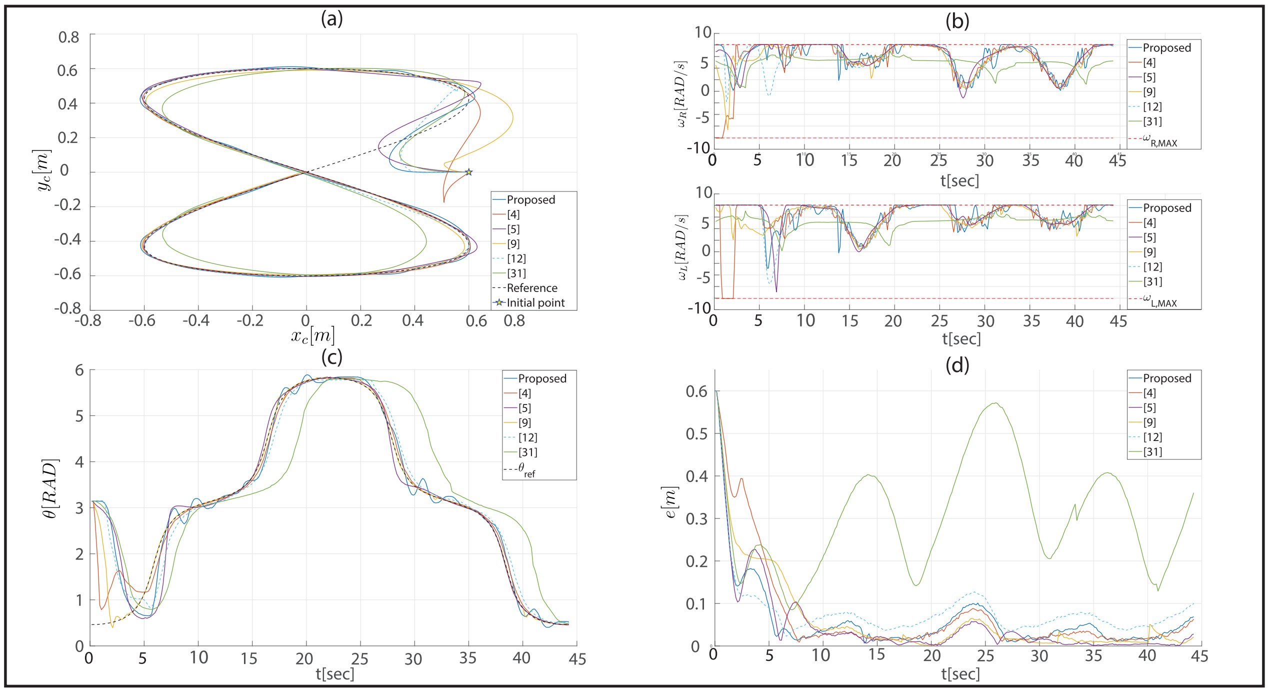

Fig. 5. Experiment 1. (a) Eight-shaped trajectory. (b) Wheel angular velocities. (c) Orientation. (d) Tracking error.

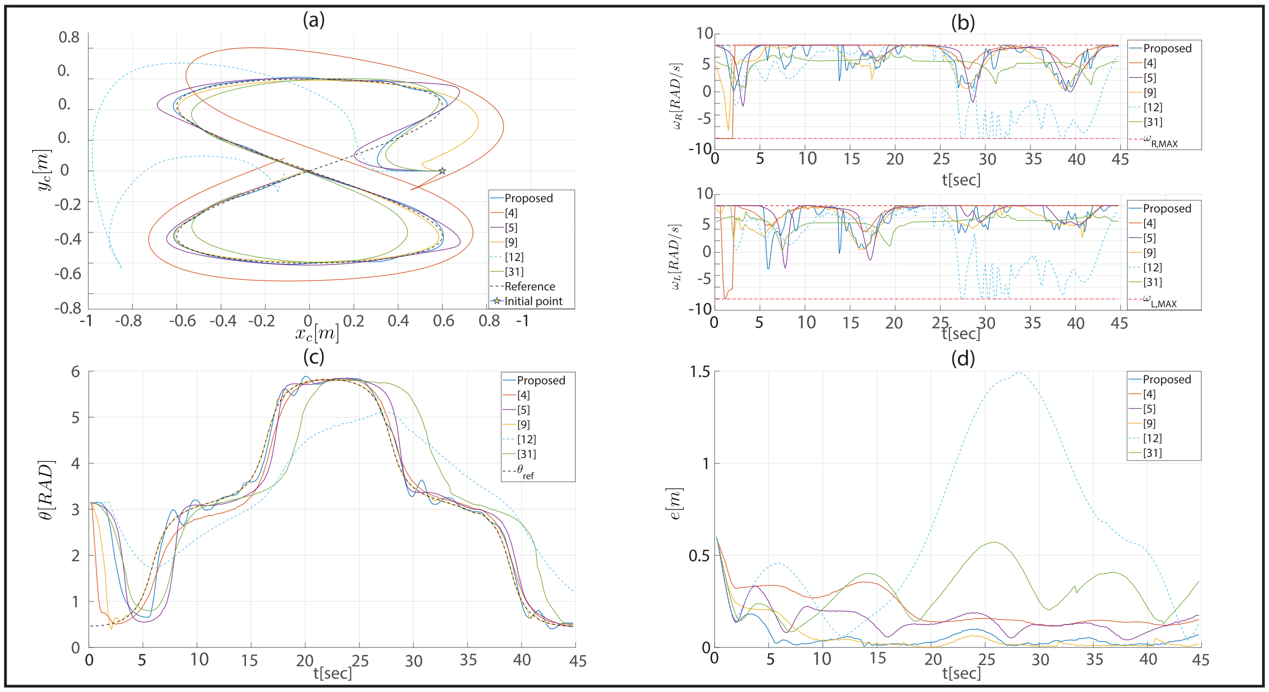

Fig. 6. Experiment 2. (a) Eight-shaped trajectory. (b) Wheel angular velocities. (c) Orientation. (d) Tracking error.

better performance when compared with [4], [12], [31]. On the other hand, the solutions [9] and [5] show slightly better performance with respect to some indices (i.e., IAE, ITAE, ITSE) when the eight-shaped trajectory is under investigation. Indeed, the proposed controller only uses the current reference, while the competitors exploit trajectory predictions or its derivatives.

2) *Experiment 2:* We consider a scenario where an onboard path planner computes the trajectory in real-time. In this setup, only the current reference point $r(k)$ is available, and there is no information on the trajectory derivatives. The obtained results are shown in Fig. 6 (for the eight-shaped trajectory) and Tables III and IV. First, the proposed solution and the strategies in [31] and [9] perform as in *Experiment 1*. This is because such strategies do not use a preview of the reference trajectory nor its derivatives. Moreover, the proposed approach outperforms all the competitor schemes, except [9] that shows slightly better tracking performance (according to the indices IAE, ITAE, and ITSE) when the eight-shaped trajectory is considered. With respect to Experiment 1, the predictive approaches in [4] and [12] worsen because only one-step predictions can be exploited. Moreover, the solution in [5] is affected by the absence of information about the trajectory's derivatives.

3) Overall Performance: In both the experiments, the proposed controller can correctly track both the trajectories while fulfilling the vehicle's velocity constraints. Moreover, the average CPU time required by the proposed control algorithm is $4.1\,ms$, which is much lower than the sampling time. Note that the proposed controller outperforms, in any tracking performance index, the preliminary solution [31]. The latter can be explained by looking at, e.g., the angular velocities plotted in Fig. 5(b), where it is evident that the proposed solution can make use of the maximum robot's velocity, while [31] always remains very conservative. Finally, from the summary Table V, obtained averaging the results in all the experiments, it is interesting to note that for almost all the indices the proposed strategy performs better than the competitor schemes.

V. Conclusion

In this brief, a novel receding horizon reference tracking controller for input-constrained differential-drive robots has been presented. The proposed strategy has been designed by properly leveraging two main ingredients: 1) an FL technique and 2) an RHC framework. The obtained tracking controller has the peculiar capability of efficiently dealing with state-dependent input constraints acting on the feedback linearized vehicle model while ensuring recursive feasibility, stability, and velocity constraints fulfillment. Extensive experimental results and comparisons have been carried out to highlight the features and advantages of the proposed tracking controller.

References

[1] R. Siegwart, I. R. Nourbakhsh, and D. Scaramuzza, *Introduction to Autonomous Mobile Robots*. Cambridge, MA, USA: MIT Press, 2011.

[2] C. Gonzalez and H. B. Schlegel, "An improved algorithm for reaction path following," *J. Chem. Phys.*, vol. 90, no. 4, pp. 2154–2161, Feb. 1989.

[3] A. De Luca and G. Oriolo, "Modelling and control of nonholonomic mechanical systems," in *Kinematics and Dynamics of Multi-Body Systems*. Vienna, Austria: Springer, 1995, pp. 277–342.

[4] F. Kuhne, W. F. Lages, and J. G. da Silva, Jr., "Model predictive control of a mobile robot using linearization," in *Mechatronics and Robotics*. Aachen, Germany: Sascha Eysoldt Verlag, 2004, pp. 525–530.

[5] G. Oriolo, A. De Luca, and M. Vendittelli, "WMR control via dynamic feedback linearization: Design, implementation, and experimental validation," *IEEE Trans. Control Syst. Technol.*, vol. 10, no. 6, pp. 835–852, Nov. 2002.

[6] B. Paden, M. Čáp, S. Z. Yong, D. Yershov, and E. Frazzoli, "A survey of motion planning and control techniques for self-driving urban vehicles," *IEEE Trans. Intell. Veh.*, vol. 1, no. 1, pp. 33–55, Mar. 2016.

[7] Y. Kanayama, Y. Kimura, F. Miyazaki, and T. Noguchi, "A stable tracking control method for an autonomous mobile robot," in *Proc. IEEE Int. Conf. Robot. Autom.*, May 1990, pp. 384–389.

[8] Z. P. Jiang and H. Nijmeijer, "Tracking control of mobile robots: A case study in backstepping," *Automatica*, vol. 33, no. 7, pp. 1393–1399, Jul. 1997.

[9] S. Blažič, "A novel trajectory-tracking control law for wheeled mobile robots," *Robot. Auto. Syst.*, vol. 59, no. 11, pp. 1001–1007, 2011.

[10] K. Shojaei, A. M. Shahri, A. Tarakameh, and B. Tabibian, "Adaptive trajectory tracking control of a differential drive wheeled mobile robot," *Robotica*, vol. 29, no. 3, pp. 391–402, May 2011.

[11] B. d'Andréa-Novel, G. Campion, and G. Bastin, "Control of nonholonomic wheeled mobile robots by state feedback linearization," *Int. J. Robot. Res.*, vol. 14, no. 6, pp. 543–559, Dec. 1995.

[12] G. Klančar and I. Škrjanc, "Tracking-error model-based predictive control for mobile robots in real time," *Robot. Auto. Syst.*, vol. 55, no. 6, pp. 460–469, 2007.

[13] T. P. Nascimento, C. E. T. Dórea, and L. M. G. Gonçalves, "Nonlinear model predictive control for trajectory tracking of nonholonomic mobile robots: A modified approach," *Int. J. Adv. Robotic Syst.*, vol. 15, no. 1, 2018, Art. no. 1729881418760461.

[14] E. F. Camacho and C. B. Alba, *Model Predictive Control*. New York, NY, USA: Springer, 2013.

[15] S. Gros, M. Zanon, R. Quirynen, A. Bemporad, and M. Diehl, "From linear to nonlinear MPC: Bridging the gap via the real-time iteration," *Int. J. Control*, vol. 93, no. 1, pp. 62–80, 2016.

[16] S. J. Yoo, Y. H. Choi, and J. B. Park, "Generalized predictive control based on self-recurrent wavelet neural network for stable path tracking of mobile robots: Adaptive learning rates approach," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 53, no. 6, pp. 1381–1394, Jun. 2006.

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

[18] D. Gu and H. Hu, "Receding horizon tracking control of wheeled mobile robots," *IEEE Trans. Control Syst. Technol.*, vol. 14, no. 4, pp. 743–749, Jul. 2006.

[19] J. Wei and B. Zhu, "Linear time-varying model predictive control for trajectory-tracking of a wheeled mobile robot," in *Proc. Int. Conf. Neural Comput. Adv. Appl.* Singapore: Springer, 2021, pp. 533–544.

[20] H. K. Khalil, *Nonlinear Systems*. Upper Saddle River, NJ, USA: Prentice-Hall, 2002.

[21] M. A. Kamel and Y. Zhang, "Linear model predictive control via feedback linearization for formation control of multiple wheeled mobile robots," in *Proc. IEEE Int. Conf. Inf. Autom.*, Aug. 2015, pp. 1283–1288.

[22] S. Bouzoualegh, E.-H. Guechi, and R. Kelaiaia, "Model predictive control of a differential-drive mobile robot," *Elect. Mech. Eng.*, vol. 10, no. 1, pp. 20–41, 2018.

[23] M. Yue, C. An, and J.-Z. Sun, "An efficient model predictive control for trajectory tracking of wheeled inverted pendulum vehicles with various physical constraints," *Int. J. Control, Autom. Syst.*, vol. 16, no. 1, pp. 265–274, Feb. 2018.

[24] V. T. Nguyen, C. Sentouh, P. Pudlo, and J.-C. Popieul, "Path following controller for electric power wheelchair using model predictive control and transverse feedback linearization," in *Proc. IEEE Int. Conf. Syst., Man, Cybern. (SMC)*, Oct. 2018, pp. 4319–4325.

[25] J. Deng, V. Becerra, and R. Stobart, "Input constraints handling in an MPC/feedback linearization scheme," *Int. J. Appl. Math. Comput. Sci.*, vol. 19, no. 2, pp. 219–232, Jun. 2009.

[26] D. Simon, J. Lofberg, and T. Glad, "Nonlinear model predictive control using feedback linearization and local inner convex constraint approximations," in *Proc. Eur. Control Conf. (ECC)*, Jul. 2013, pp. 2056–2061.

[27] K. Margellos and J. Lygeros, "A simulation based MPC technique for feedback linearizable systems with input constraints," in *Proc. 49th IEEE Conf. Decis. Control (CDC)*, Dec. 2010, pp. 7539–7544.

[28] A. De Luca, G. Oriolo, and M. Vendittelli, "Control of wheeled mobile robots: An experimental overview," in *Ramsete*. Berlin, Germany: Springer, 2001, pp. 181–226.

[29] M. V. Kothare, V. Balakrishnan, and M. Morari, "Robust constrained model predictive control using linear matrix inequalities," *Automatica*, vol. 32, no. 10, pp. 1361–1379, 1996.

[30] B. Pluymers, M. V. Kothare, J. A. K. Suykens, and B. De Moor, "Robust synthesis of constrained linear state feedback using LMIs and polyhedral invariant sets," in *Proc. Amer. Control Conf.*, 2006, p. 6.

[31] C. Tiriolo, G. Franze, and W. Lucia, "A receding horizon control strategy for constrained differential-drive robots moving in static unknown environments," in *Proc. IEEE Conf. Control Technol. Appl. (CCTA)*, Aug. 2020, pp. 261–266.

[32] L. D'Alfonso, W. Lucia, P. Muraca, and P. Pugliese, "Mobile robot localization via EKF and UKF: A comparison based on real data," *Robot. Auto. Syst.*, vol. 74, pp. 122–127, Dec. 2015.

[33] D. Wang and G. Xu, "Full-state tracking and internal dynamics of nonholonomic wheeled mobile robots," *IEEE/ASME Trans. Mechatronics*, vol. 8, no. 2, pp. 203–214, Jun. 2003.

[34] D. Q. Mayne, J. B. Rawlings, C. V. Rao, and P. O. M. Scokaert, "Constrained model predictive control: Stability and optimality," *Automatica*, vol. 36, no. 6, pp. 789–814, 2000.

[35] F. Borrelli, A. Bemporad, and M. Morari, *Predictive Control for Linear and Hybrid Systems*. Cambridge, U.K.: Cambridge Univ. Press, 2017.

[36] F. Blanchini and S. Miani, "Any domain of attraction for a linear constrained system is a tracking domain of attraction," *SIAM J. Control Optim.*, vol. 38, no. 3, pp. 971–994, Jan. 2000.