Information Sciences 501 (2019) 436–459

Contents lists available at ScienceDirect

Information Sciences

journal homepage: www.elsevier.com/locate/ins

Synthetic-analytic behavior-based control framework: Constraining velocity in tracking for nonholonomic wheeled mobile robots

Marlen Meza-Sánchez[b], Eddie Clemente[a,*], M.C. Rodríguez-Liñán[b], Gustavo Olague[c]

[a] Tecnológico Nacional de México/ I. T. Ensenada, Blvd. Tecnologico 150, Ex-ejido Chapultepec, Ensenada 22780, B.C., Mexico
[b] CONACYT-TecNM/I.T. Ensenada, México
[c] Department of Computer Science, CICESE, Carretera Tijuana-Ensenada No. 3918, Zona Playitas, Ensenada, B. C., México

ARTICLE INFO

Article history:
Received 8 February 2019
Revised 8 May 2019
Accepted 8 June 2019
Available online 8 June 2019

Keywords:
Synthetic-analytic behaviors
Bounded velocity
Evolutionary robotics
Genetic programming
Nonholonomic mobile robots
Nonlinear tracking control

ABSTRACT

This work presents a genetic programming control design methodology that extends the traditional behavior–based control strategy towards a synthetic-analytic perspective. The proposed approach considers the internal and external dynamics of the system, providing solutions to a general structure, and including analytic functions, which can be studied within the Control Theory framework. The method is illustrated for the tracking control problem under bounded velocity restrictions of a nonholonomic wheeled mobile robot. A classic Control Theory (CT) based controller that solves the tracking problem (but not the velocity constraint requirement) is chosen from the literature; based on its stability properties, a modified structure where the search of suitable analytic basis behaviors, fulfilling both control objectives simultaneously, can be introduced. The proposed framework takes the form of a learning process based on Genetic Programming (GP) which generates a set of nonlinear tracking controllers satisfying pre-specified velocity bounds. A collection of 9113 suitable nonlinear solutions were obtained to augment the ground controller. Simulations and real–time experiments are performed to illustrate the effectiveness of the methodology through the testing of the models with the best performance, as well as those with lower structural complexity.

1. Introduction

The synthesis of nonlinear controllers where multiple objectives are pursued constitutes a significant challenge to the control research community. Here, the aim is to make a system meet additional constraints without compromising the achievement of a desired motion or behavior. While most state-of-the-art proposals addressing control problems can be classified in two main broad approaches (Control Theory and Soft Computing techniques), there are successful results where these are combined to take advantage of the strengths of each method, [7,14,26,27,31]. In this work, a methodology for the automated synthesis of nonlinear controllers through an evolutionary process is proposed. The objective is to solve the tracking control problem in nonholonomic wheeled mobile robots while fulfilling a bounded velocity policy.

* Corresponding author.
E-mail addresses: eclemente@ite.edu.mx (E. Clemente), mcrodriguez@conacyt.mx (M.C. Rodríguez-Liñán), olague@cicese.mx (G. Olague).

https://doi.org/10.1016/j.ins.2019.06.025
0020-0255/

Control of nonholonomic wheeled mobile robots is challenging since these mechanisms possess more degrees of freedom than available control variables. By definition, they have motion constraints related to their wheels configuration. In other words, this kind of robots is unable to move simultaneously and independently in any arbitrary direction in the horizontal plane and to instantaneously change their orientation (i.e., translational and rotational motion). Therefore control of nonholonomic mobile robots has gathered the attention from various research groups, as evidenced by the literature, [5,9–11,13,15,20,21,25,28,29].

From a practical point of view, kinematic control laws are the most common way to develop motion controllers for wheeled mobile robots. While many proposals from the state-of-the-art provide designs using torque or voltage as the control signal, most commercial and academic prototypes use velocity inputs instead. Such velocity inputs are either specified as velocities for each of the robot's wheels, or as linear and angular velocities in the kinematic model. However, independently of which speeds are considered in the design, for safety purposes, it is necessary to comply with predefined boundaries in the velocities to avoid skidding and slipping effects in the robot's motion [24].

To assume that the robot actuators can supply any demanded torque, voltage, or velocity simplifies the design stage of the controllers but, in practice, this supposition is not true. When the signal demanded by the controller exceeds the physical capacities of the actuators, they operate in saturated mode. Keeping the actuators working at the saturation limit decreases their lifespan since they work at their maximum capacity. To preserve the actuators' integrity, the saturation mode should be avoided whenever it is possible. Then, it follows that, in general, it is necessary to derive controllers such that actuator constraints are considered in the design.

Saturation of the reachable velocities can be addressed by the specific use of saturation functions (such as the hyperbolic tangent or the sign) in combination with CT or soft computing methods to design the controllers. For example, from a CT approach, many works have been proposed to solve one or various of the control problems related to nonholonomic mobile robots (tracking, path following, stabilization, flocking, or formation [5]), take for example, [5,10,11,20,21,28]. In [5], the tracking control problem subject to bounded velocity and torque is addressed using two first-order filters. The filters ensure that the torque and velocity constraints are satisfied producing uniformly continuous feedback signals. A backstepping method is used in [10] to solve the path following problem when the actuator velocities are limited. Notably, this strategy is directed at high-speed applications. Jiang et al. [11] present an adaptive controller, based on passivity and normalization, to achieve global stabilization and tracking for a mobile robot subject to input velocity constraints. Nonlinear Model Predictive Control (NMPC) is employed in [20] for trajectory tracking in nonholonomic mobile wheeled robots. NMPC can naturally deal with restrictions; then, the proposal achieves the tracking objective while restricting input velocities. Serrano et al. [28] propose a tracking controller for a wheeled mobile robot subject to saturation in the angular and linear speeds. The approach uses nonlinear programming methods to calculate the controller parameters. A generalized framework for several types of wheeled mobile robots to satisfy the path following requirement while keeping the velocities within acceptable bounds is presented in [21]. The authors use explicit expressions for the input velocities to meet the tracking objective.

Alternatively, Soft Computing techniques can be used to achieve an adequate performance of the robot while enforcing limits in the velocity signals. In [29], formation control is dealt with using adaptive Radial Based Function Neural Networks (RBFNN) to approximate the actuator's saturation, where the saturated input is the torque. Additionally, the estimation and tracking errors are bounded by saturation functions. It should be noted that the saturation bounds are explicitly incorporated in the controller design. A Takagi–Sugeno fuzzy controller is proposed in [25] to solve the trajectory tracking problem of a unicycle mobile robot which is subject to velocity and actuator saturation. In their work, the saturation nonlinearity is approximated through a set of fuzzy rules, rather than by specific saturation functions. The authors demonstrate the stability of the closed-loop system employing Lyapunov functions. They present three experiments to demonstrate the approach: one positioning challenge, and two tracking experiments following a line and an eight shape curve.

In all of the works mentioned above, it is assumed that the control signal is either the angular or the linear velocity of the robot, with its corresponding boundary. This restriction means that this boundary must be calculated for the wheel configuration and the physical parameters of the actuators. This work, in particular, focuses on solving the control problem where constraints in the velocity of each of the robot's wheels are considered.

Under the requirements stated above (kinematic restrictions and input velocity bounds), it becomes clear that a hybrid framework will be best to exploit the advantages of different approaches to solve such a complex task. Specifically, the tracking control problem can greatly benefit from the neural network framework of Evolutionary Robotics [8], where the aim is to endow the robot with natural language processing or deductive reasoning to generate behaviors that can solve complex tasks. A behavior can be defined as an independent action resulting from the direct interaction of the system with its environment. This concept was introduced to represent intelligence in artificial systems [3]. Furthermore, a system can exhibit complex behaviors when such activity producers are intertwined and executed in parallel. The works of Matarić et al. [17,18] and the developments proposed by Arkin [1], led to the evolution of the concept of behaviors into an approach called Behavior–based control. This approach aims to solve control problems within the robotics field; it proposes the development of a process where a set of actions, or modules, named *basis* behaviors, are combined to achieve desired features of the system. This method was originally developed for situated robots that need to adapt to the dynamics of real-world environments without considering (a) the internal dynamics of the system, or (b) abstract representations of knowledge and reality [19].

Recently, an analytic behavior–based framework was proposed by Clemente et. al [6] for obstacle avoidance with bounded velocity for the position control problem in omnidirectional mobile robots. The proposed methodology takes advantage of

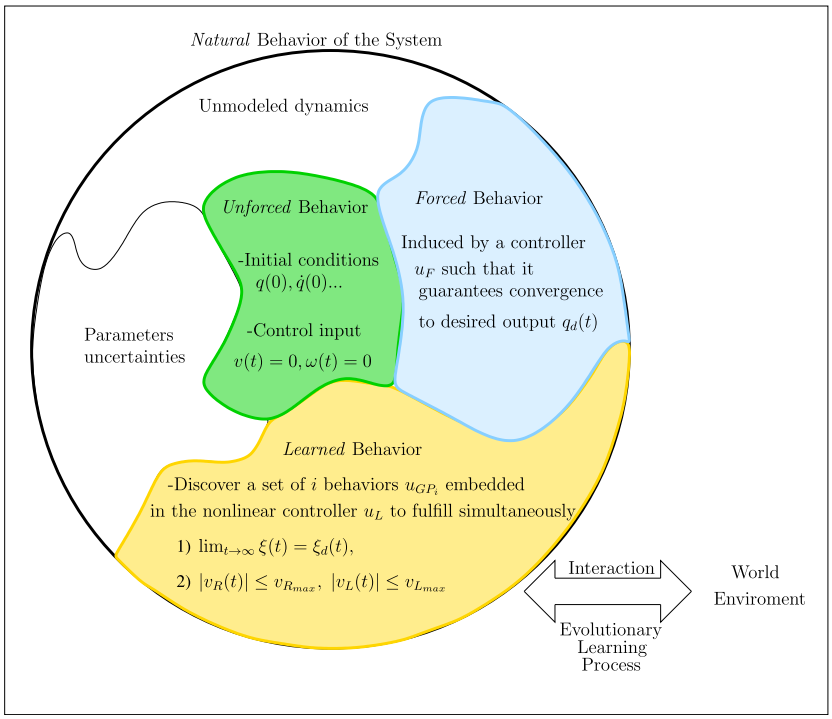


Fig. 1. Conceptual framework of the learning process of a nonholonomic mobile robot towards the development of analytic nonlinear tracking controllers with constrained velocities.

the Potential Fields approach to derive a *forced* behavior and an evolutionary search for *adaptive* behaviors as the attractive and the repulsion functions. In [22], this methodology was extended to solve the tracking problem in a double integrator system; and in [23] for second-order dynamical systems. A traditional PD with inverse dynamics is proposed to generate the *forced* behavior, while the *learned* responses are searched by an evolutionary approach for a bounded flow variable. The present work extends and develops the ideas introduced by the authors in [6,22,23]. It aims to control a nonholonomic wheeled mobile robot under velocity constraints using a PI-type controller.

The analytic behavior–based framework has three significant features that distinguish it from traditional Behavior–based control. First, it defines the *basic* behaviors as analytic functions, which can also be nonlinear; second, it uses of a model of the system to include the internal dynamics; and third, it integrates a CT-based controller with a learning process applying GP to generate the *learned* behaviors. Moreover, the advantage of employing a CT-based controller is that it can be analyzed in terms of the control theory framework, thus guaranteeing the performance of the system; this is possible through the concept of stability of the equilibrium points. An equilibrium point is a coordinate of the state space such that, whenever the mobile robot starts at it, it will remain at that point for all future time. The stability of a system is proven through a qualitative analysis of the trajectories, or solution curves of the system. The most critical stability criterion within the CT approach is the Lyapunov stability theorem.

The integration of the CT method with the GP approach allows the implementation of a learning stage seeking fulfillment of additional features in the behavior of the robot. The GP technique allows for the construction of a syntactical tree to represent a solution given in the form of nonlinear controllers; such solutions are composed of mathematical operators and analytic functions derived from the CT approach.

In this work, we extend the application of the analytic behavior-based framework by addressing the tracking problem in nonholonomic mobile robots. Constrained velocities in the robot's wheels are also considered. In contrast to previous works using this framework, this proposal takes advantage of the Lyapunov stability conditions to derive the learning stage in the mobile robot. A modified structure of a classical state-of-the-art tracking controller is introduced to seek behavior modifiers. Such modifiers simultaneously fulfill the convergence to the desired trajectory while keeping the velocities within some constant boundary value. Moreover, the GP approach guarantees that the speeds in each wheel will never reach the saturation bounds for several given scenarios. Its strength lies in the automation of the synthesis of nonlinear controllers, giving rise to a big set of solutions that can be studied by the CT approach.

The methodology presented in this work permits the automatic design of nonlinear controllers, which are in general hard to derive. This approach could be easily extended to the design of controllers for plants with multiple constraints. The intrinsic characteristics of the method allow the user to explore alternative solutions to the task at hand. The solutions found by the proposed method can be more effective than those found by traditional means by optimizing variables like

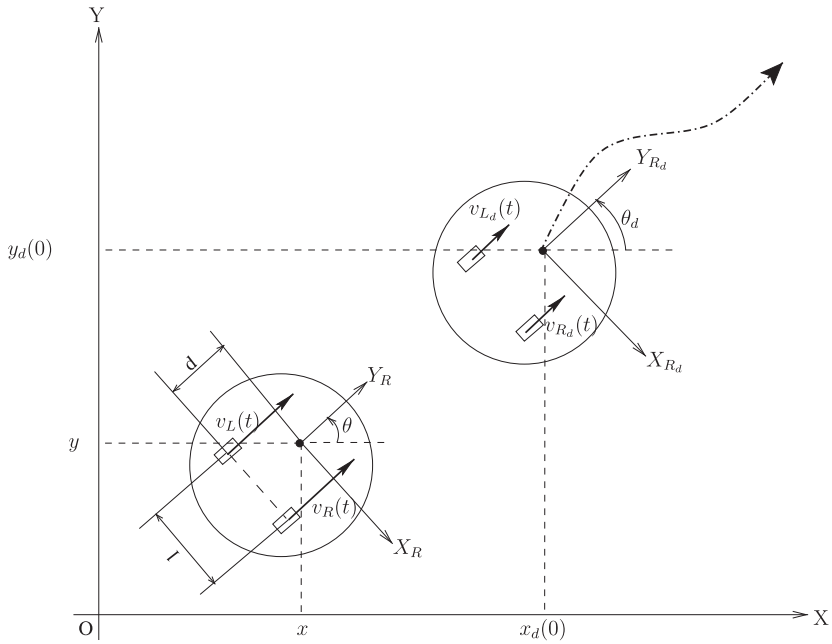


Fig. 2. Motion problem setup for the tracking control problem of the desired trajectory in a nonholonomic mobile robot within a normalized workspace in the *XOY* plane.

total energy, settling time, convergence time, etc. These characteristics can be advantageous in a variety of systems, such as in robotics, mechanics, electronics, and in general, in multi-parametric plants, to mention a few.

The rest of the paper is organized as follows. Section 2 contains the problem statement. Section 3 describes the synthesis of the nonlinear controllers for the basic behaviors, the parameters for the GP, and the statistical analysis of the attained solutions. Section 4 describes the discovered nonlinear controllers together with numerical results. Section 5 presents an application example. Finally, the conclusions are given in Section 6.

1.1. Contributions

This paper introduces the development of nonlinear control laws based on an analytic behavior-based framework which integrates Control Theory with the Genetic Programming approach. The aim is to solve the tracking control problem in nonholonomic mobile robots operating with bounded velocity. In contrast with other works found in the literature, this paper seeks the satisfaction of the velocity bounds, individually, for each of the mobile robot's wheels. The proposed method relies on the Genetic Programming approach to keep the velocity inputs within the saturation bounds by creating control laws that implicitly enforce this behavior.

The contributions of this work can be summarized as follows.

1. The present work proposes a novel approach that combines Control Theory and Genetic Programming concepts to deliver a controller design methodology. The proposed strategy generates analytic solutions, in contrast with those obtained from other soft computing techniques, like neural network or fuzzy control systems, such as Mamdami models. The advantage of dealing with analytic solutions is that, on the one hand, the cost associated to the implementation of an analytic solution is lower than that of nonanalytic ones, this means that the solutions obtained using the proposed methodology can be implemented in real-time. On the other hand, the proposal in this paper makes it possible to utilize existent analysis techniques developed within the Control Theory framework, by its analytic properties.
2. A set of 9113 solutions were found by the proposed algorithm, thus automating the design process. The automation of such a process implies a reduction in manual design time.
3. The methodology in this work produces several solutions of different complexity. The user can choose among them, according to specific requirements of the controlled system, or the computational and implementation cost associated with each application. Also, the automation of the learning process aiming to discover suitable nonlinear controllers reveals insights for the synthesis of new controllers.
4. In this paper, the proposed scheme is demonstrated for a nonholonomic wheeled mobile robot. However, it can be generalized to other applications and control problems.

Table 1
Terminals used in the evolutionary process to find the *learned* robot behaviors satisfying the bounded velocity condition.

ID	Term	Description	ID	Term	Description
1	$x(t)$	Robot position in X-axis	10	$\dot{x}_d(t)$	Linear velocity of the robot in X-axis
2	$y(t)$	Robot position in Y-axis	11	$\dot{y}_d(t)$	Linear velocity of the robot in Y-axis
3	$x_d(t)$	Desired robot position in X-axis	12	$v_{Rd}(t)$	Desired velocity in the right wheel
4	$y_d(t)$	Desired robot position in the Y-axis	13	$v_{Ld}(t)$	Desired velocity in the left wheel
5	$\theta(t)$	Robot orientation	14	$x_d(t) - x(t)$	Negative error position in x
6	$\theta_d(t)$	Desired robot orientation	15	$y_d(t) - y(t)$	Negative error position in y
7	$v_d(t)$	Desired linear velocity for the robot	16	x_e	Error position in X-axis on the inertial frame
8	$\omega_d(t)$	Desired angular velocity for the robot	17	y_e	Error position in Y-axis on the inertial frame
9	$\theta_e(t)$	Error in orientation on the inertial frame			

Table 2
Functions used in the evolutionary process to find the *learned* robot behaviors that achieve bounded velocity.

ID	Expression	Definition	ID	Expression	Definition
1	$+$	Addition	18	$ct_h(\cdot)$	Hyperbolic cotangent
2	$-$	Subtraction	19	$\mathrm{sgn}(\cdot)$	Signum function
3	$/$	Division	20	$t_{hi}(\cdot)$	Inverse hyperbolic tangent
4	$*$	Multiplication	21	$s_{hi}(\cdot)$	Inverse hyperbolic sine
5	$(\cdot)^{(\cdot)}$	Exponentiation	22	$\mathrm{erf}(\cdot)$	Gauss error function of the real part of the argument
6	$\max(\cdot,\ \cdot)$	Maximum	23	$c(\cdot)$	Cosine
7	$\min(\cdot,\ \cdot)$	Minimum	24	$s(\cdot)$	Sine
8	$\mathrm{atan2r}(\cdot,\ \cdot)$	Inverse tangent of the real part of each argument	25	$t(\cdot)$	Tangent
9	$\sqrt{\cdot}$	Square root	26	$cs(\cdot)$	Cosecant
10	$(\cdot)^2$	Square	27	$sc(\cdot)$	Secant
11	$\ln(\cdot)$	Natural logarithm	28	$ct(\cdot)$	Cotangent
12	$e^{(\cdot)}$	Exponential	29	s_i	Inverse sine
13	$s_h(\cdot)$	Hyperbolic sine	30	c_i	Inverse cosine
14	$c_h(\cdot)$	Hyperbolic cosine	31	$\|\cdot\|$	Euclidean norm
15	$t_h(\cdot)$	Hyperbolic tangent	32	$\lvert\cdot\rvert$	Absolute value
16	$cs_h(\cdot)$	Hyperbolic cosecant	33	$\mathrm{Re}(\cdot)$	Real part of the argument
17	$sc_h(\cdot)$	Hyperbolic secant	34	$\mathrm{atanr}(\cdot)$	Inverse tangent of the real part of the argument

5. The learning process was performed in simulation using only one desired trajectory, and a given set of parameters and initial conditions for the robot. In practice, the controllers were tested on a robot with different settings and paths, which demonstrates the applicability of the proposed strategy.

2. Behaviors in tracking control for nonholonomic mobile robots

An overview of the critical aspects of the applied analytic behavior-based framework for this specific control problem is shown in Fig. 1. Let us define the *unforced* behavior of the nonholonomic mobile robot as its kinematic model, given some initial position at time t_0, and without considering the action of the controller. The kinematic model, described by a vector of generalized coordinates as shown in Fig. 2, is given as [4,32]

$$\dot{\boldsymbol{\xi}}(t) = \begin{bmatrix} \cos(\theta(t)) & 0 \\ \sin(\theta(t)) & 0 \\ 0 & 1 \end{bmatrix} \mathbf{u}(t), \tag{1}$$

where $\boldsymbol{\xi}(t) = [x(t),\ y(t),\ \theta(t)]^T$ refers to its position coordinates of a point of contact with the ground in a fixed coordinate frame XOY, and its orientation angle with respect to the X axis; the vector of control inputs $\boldsymbol{u}(t) = [v(t),\ \omega(t)]^T$ corresponds with its linear velocity and with the angular velocity about its instantaneous rotational axis. This model describes cars, carts, and mobile robots with parallel driven wheels, such as those with differential drive locomotion type, and some four-wheeled robots. This class of mobile robots cannot move sideways since they cannot roll in such direction unless they turn around first. In mobile robots with parallel driven wheels, such as the one portrayed in Fig. 2, the motion is only allowed in the normal direction to its wheels axis.

Assuming that the condition of *pure rolling* and *non-slipping* is fulfilled, the nonholonomic constraint for the model (1) is given as

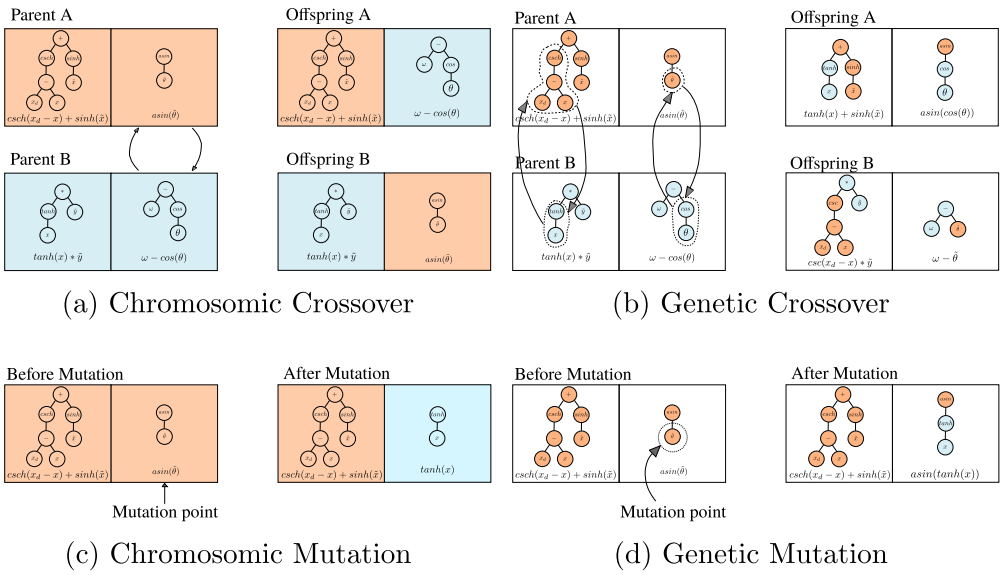


Fig. 3. Recombination operators applied to the proposed representation.

$$\dot{y}(t)\cos(\theta(t)) - \dot{x}(t)\sin(\theta(t)) - \omega(t) = 0. \tag{2}$$

The *pure rolling* condition supposes that the rolling motion of the mobile robot is the sum of pure rotation of the wheel and pure translation ahead. The *non-slipping* requirement means that the net instantaneous velocity of the wheel on the contact surface is zero, i.e., the point of contact of the wheels with the plane does not move.

Since the *unforced* behavior is characterized by the kinematic model (1) subject to the restriction (2), where $\boldsymbol{u}(t) = [0, 0]^T$. The *forced* and the *learned* behaviors arise from closing the control loop. Let us redefine the vector of control inputs as

$$\boldsymbol{u}(t) = \boldsymbol{u}_F + \boldsymbol{u}_L, \tag{3}$$

where $\boldsymbol{u}_F$ is the control law which generates the *forced* behavior, and $\boldsymbol{u}_L$ is the controller that produces the *learned* behaviors in the mobile robot. Let the *forced* behavior be imposed using a selected CT-based tracking controller. This induced behavior guarantees a consistent response of the system under certain conditions, applying the Lyapunov stability criterion. The second part of the controller denoted as $\boldsymbol{u}_L$ in (3) is discovered through the implementation of an automatic search of behavior modifiers. Such behavior modifiers must lead the system to exhibit additional desired features without compromising the achievement of the *forced* behaviors. For the control problem as mentioned earlier, the GP approach is used to discover the set of nonlinear controllers aiming to constraint the velocities of each wheel in the nonholonomic mobile robot.

3. Synthesis of nonlinear tracking controllers enhancing bounded velocity motion

The objective of this work is to synthesize nonlinear controllers addressing the autonomous navigation problem in nonholonomic mobile robots. Besides, the convergence to the desired motion must be constrained to exhibit bounded velocities chosen accordingly to meet safe operation values of a real wheeled mobile robot.

Consider the locomotion problem of the differential drives mobile robot shown in Fig. 2. The control input $\boldsymbol{u}(t)$ is defined in terms of the linear and angular velocities, $v(t)$ and $\omega(t)$, respectively; in practice, both speeds are related to the velocities of each wheel. Since the distance between both wheels (denoted as l) is constant, the velocities of the right and left wheels of the mobile robot can be computed as

$$v_R(t) = v(t) + \frac{l\omega(t)}{2}, \quad v_L(t) = v(t) - \frac{l\omega(t)}{2}. \tag{4}$$

Assuming perfect velocity tracking of the wheels (i.e., the wheels of the mobile robot display the demanded velocities $v_R(t)$ and $v_L(t)$) according to relation (4), and that there is a smooth desired trajectory $\boldsymbol{\xi}_d(t) = [x_d, y_d, \theta_d]^T$ to be tracked, for the system (1)–(2), the following control objectives must be ensured

1. To solve the autonomous navigation problem, such that the nonholonomic mobile robot reaches a desired behavior in the predefined workspace; this is defined as

$$\lim_{t\to\infty} \boldsymbol{\xi}(t) = \boldsymbol{\xi}_d(t). \tag{5}$$

Table 3
Settings for the parameters used by the evolutionary process.

Parameter	Value
Number of Generations	100
Population size	400
Crossover rate	80%
Mutation rate	20%
Maximum tree depth	11
Sampling	Lexicographic
Elitism	Keep best

2. The linear velocity of both wheels, $v_R(t)$ and $v_L(t)$, are kept within a constant boundary interval; this is,

$$|v_R(t)| \leq v_{R_{max}}, \quad |v_L(t)| \leq v_{L_{max}} \quad \forall \quad t > 0, \tag{6}$$

where $v_{R_{\max}}$, $v_{L_{\max}}$ are positive constants corresponding to the maximum value on the wheels' velocities. This limits can be freely set to meet specific physical constraints. Notice that if the mobile robot wheels have identical specifications, then the values $v_{R_{\max}}$ and $v_{L_{\max}}$ can be the same.

3.1. The forced behavior: CT-based tracking controller

The *forced* behavior proposed for the mobile robot is the fulfilling of the control objective given in (5); that is, it must guarantee the convergence of the robot to the desired reference $\boldsymbol{\xi}_d(t)$. Let us define the tracking errors for the mobile robot modeled by (1)–(2) with respect to the desired trajectory $\boldsymbol{\xi}_d(t)$, as

$$\tilde{\boldsymbol{\xi}}(t) = T(\boldsymbol{\xi}(t) - \boldsymbol{\xi}_d(t)), \tag{7}$$

where $\tilde{\boldsymbol{\xi}}(t) = [x_e,\ y_e,\ \theta_e]^T$ is the error vector for each position in the coordinated workspace, and

$$\boldsymbol{T} = \begin{bmatrix} \cos(\theta) & \sin(\theta) & 0 \\ -\sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix}, \tag{8}$$

since the position error is not given in the global coordinate frame but rather as an error in the local coordinate frame of the robot, denoted as $X_R O Y_R$.

Let us choose a state-of-the-art controller, presented in [12], as the CT-based controller, which is given by the following expression

$$\boldsymbol{u}_F = \boldsymbol{u}_{ct} = \begin{bmatrix} v_d \cos(\theta_e) \\ \omega_d + k_\theta \sin(\theta_e) \end{bmatrix} + \boldsymbol{K} \begin{bmatrix} x_e \\ v_d y_e \end{bmatrix} \tag{9}$$

where $\boldsymbol{u_{ct}} = [v_{ct}\ \omega_{ct}]^T$ is the control input which generates the *forced* behavior, and the matrix $\boldsymbol{K} =$ diag $\{k_x,\ k_y\}$ and variable k_θ are the constant gains of the controller. Notice that the subindex *ct* in (9) denotes the application of the Control Theory approach in the first part of the controller, as defined in (3). The closed-loop dynamics obtained after applying controller (9) to the kinematic model (1) are uniformly asymptotically stable around $\tilde{\boldsymbol{\xi}} = \boldsymbol{0}$. The gain tuning process requires that $k_x \neq T_s^{-1}$, where T_s is the sampling time, and the fulfillment of the relation $4k_y \geq k_\theta^2$. For further details, refer to the Lyapunov stability proof in [12]. The selection of controller $\boldsymbol{u_{ct}}$ given in [12] is arbitrary. The only requirement is that the chosen controller converges to solve the tracking control problem for this class of wheeled mobile robots. We aim to derive a learning process for the nonholonomic mobile robot to simultaneously fulfill a bounded velocity condition, as stated in the control objective given in (6).

3.2. The learned behaviors for bounded velocity: a GP-based set of nonlinear controllers

The Genetic Programming paradigm is now applied to derive a scheme for the construction of analytic solutions. The implemented evolutionary process automates the synthesis of nonlinear controllers, causing the mobile robot to simultaneously fulfill the control objectives stated in (5) and (6). Let us define

$$\boldsymbol{u}_L = -\boldsymbol{K}\ \boldsymbol{u}_{gp} \begin{bmatrix} x_e \\ v_d y_e \end{bmatrix}, \tag{10}$$

where $\boldsymbol{K} \in \mathbf{R}^{2\times 2}$ is the same gain matrix as in the CT-based controller (9) and $\boldsymbol{u}_{gp} =$ diag $\{v_{gp},\ \omega_{gp}\}$ is the control input, obtained by an evolutionary process which produces the *learned* behaviors in the mobile robot. Recall that the full nonlinear controller $\boldsymbol{u}$ that is applied is given as the summation of the selected CT-based controller $\boldsymbol{u}_{ct}$ in (9), and the proposed GP-based control input $\boldsymbol{u}_L$ in (10).

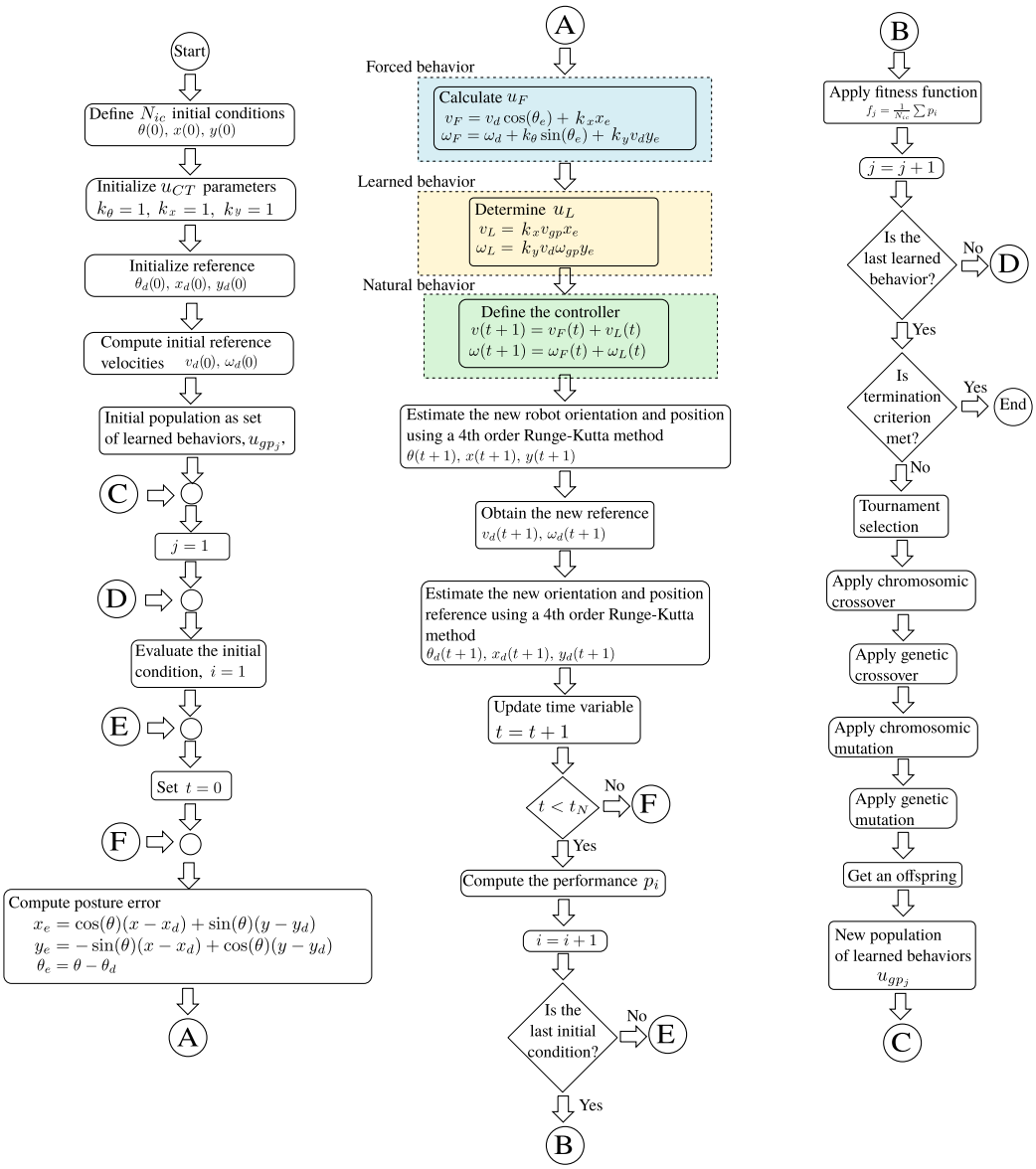


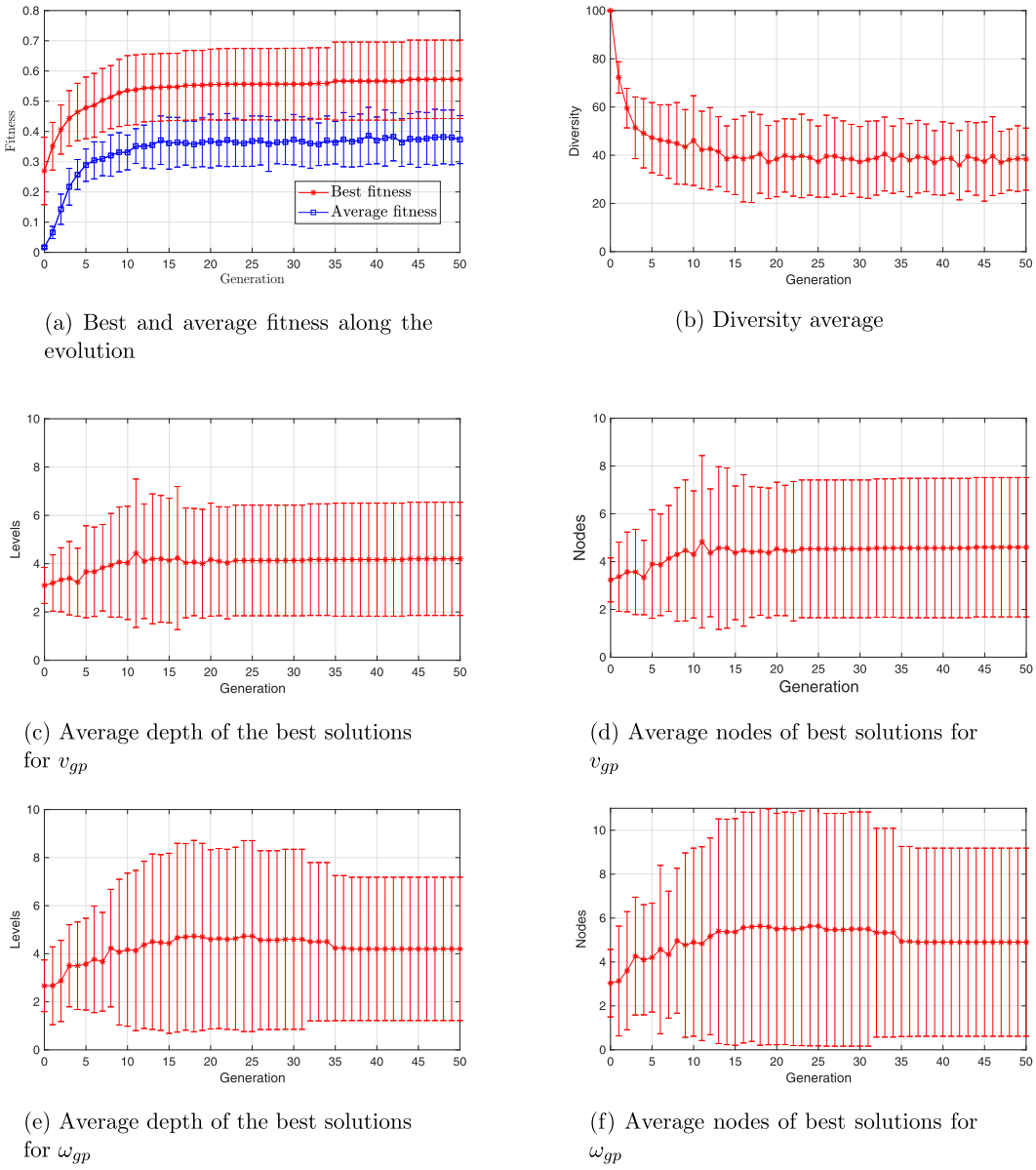
Fig. 4. Evolutionary process for the synthesis of nonlinear tracking controllers subject to velocity constraints.

The proposed approach applies an evolutionary process for the optimization of two different functions v_{gp} and ω_{gp}. Notice that v_{gp} and ω_{gp} are meant to modify the *forced* behavior induced by (9). Specifically, v_{gp} modifies the linear velocity along the *X*-axis, since it is multiplied by the error x_e; similarly, ω_{gp} modifies the motion along the *Y*-axis. A set of functions and terminals are selected for each behavior modifier focusing in its own specific sought behavior. Thus, each expression has it own set of functions and terminals.

3.2.1. Functions and terminals

The functions and terminals are defined according to the most used mathematical expressions in Control Theory. These expressions represent the search space where the *learned* behaviors will be discovered. Tables 1 and 2 show the set of functions and terminals that can be used to build the v_{gp} and ω_{gp} operators. Given the common domain, and the dependence between v_{gp} and ω_{gp}, the set of functions and terminals are similarly defined; nevertheless, each feature is independent, and is associated with a specific performance.

The terminal set is defined in terms of the robot's state variables, given as position, orientation and velocity, and in terms of the error between the desired value and the system behavior. The function set is proposed by arithmetic, trigonometric, hyperbolic, and some special functions. Thus, operators frequently used in the state-of-art, such as signum and hyperbolic



(a) Best and average fitness along the evolution

(b) Diversity average

(c) Average depth of the best solutions for v_{gp}

(d) Average nodes of best solutions for v_{gp}

(e) Average depth of the best solutions for ω_{gp}

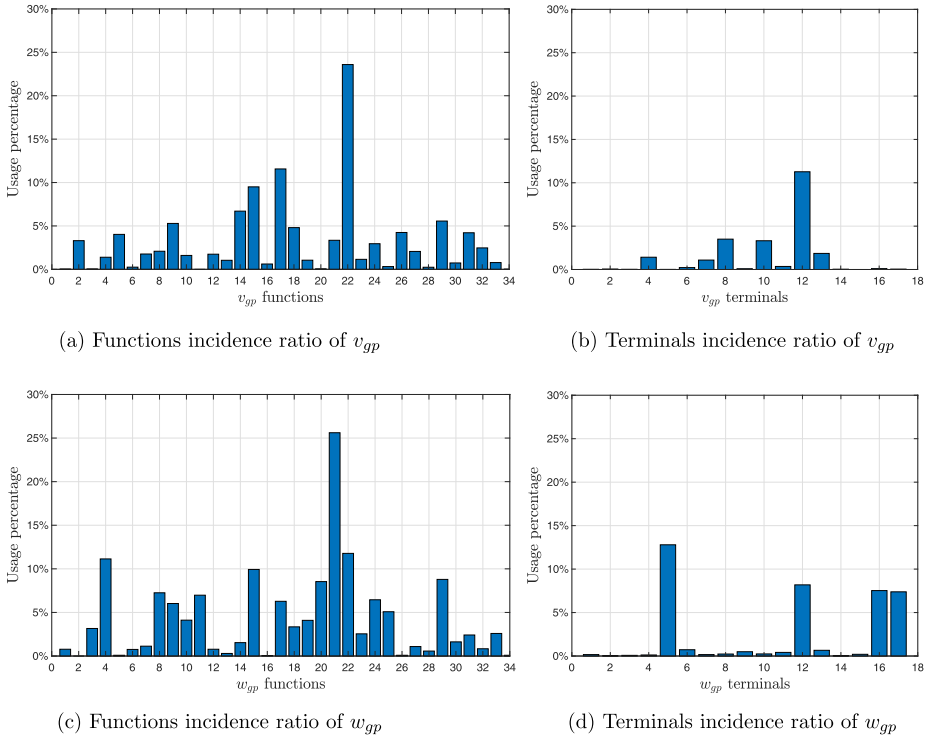
(f) Average nodes of best solutions for ω_{gp}

Fig. 5. Statistics of the evolutionary GP process showing the average fitness and complexity of learned behaviors.

tangent functions, are included. Additional operations, such as exponential, natural logarithm, maximum, and minimum, are also considered in the search space.

3.2.2. Genotype and evolutionary operators

The *genotype* is defined as the representation of a possible solution. In Genetic Programming it is commonly codified by a syntactic tree, encoding a single function or program. Two different operators are used in the present approach. Each operation is represented as a syntactic tree array as shown in Fig. 3. The first syntactic tree codifies v_{gp}, while the second tree typifies the ω_{gp} function. This atypical representation is inspired by [6]. It implies genetic recombination taking into account the syntactic array, and the elements that compose them. Hence, the genetic operators are defined in two levels. In a higher level, the syntactic tree level (called chromosome), involves similar genetic operators as defined in *single point crossover*, and common mutation in Genetic Algorithms. In a lower level, the recombination operators are defined over each element of the chromosome. Thus, the crossover and mutation operators are implemented using a typical setting, as in the classical Genetic Programming approach. Fig. 3 depicts an example of these operators, where the user defines the probability selection of each genetic operator.



(a) Functions incidence ratio of v_{gp} (b) Terminals incidence ratio of v_{gp}

(c) Functions incidence ratio of w_{gp} (d) Terminals incidence ratio of w_{gp}

Fig. 6. Frequency of use of functions and terminals for v_{gp} and w_{gp}.

In this case, the *phenotype* is the behavior showed by the robot. Note, that this performance is given by the *forced* and *learned* behavior interaction. Thus, the *genotype* only codifies an intrinsic part of the robot behavior.

3.2.3. Fitness function

The fitness function is a metric that describes the performance of a given design solution. The goal is to guide a search for a suitable solution where the mobile robot follows the desired path while keeping its linear velocity $v(t)$ within defined safe operation bounds. This score is given in terms of the mean squared error in each direction, and a set of weighted metrics, defined for each of the robot's wheels, as follows

$$F \propto MSE_x + MSE_y + \gamma_R + \gamma_L + \phi_R + \phi_L \,, \tag{11}$$

where MSE_k is the mean squared error between the desired position, and the robot position over each axis, $k = \{x, y\}$. Thus, MSE_x is expressed as

$$MSE_x = \frac{1}{t_f} \sum_{t=0}^{t_f} (x(t) - x_d(t))^2 \,, \tag{12}$$

and similarly for MSE_y. The term γ_R represents a weighted metric between $v_{R_{\max}}$ and the maximum velocity $max(|v_R(t)|)$ reached by the right wheel of the robot as

$$\gamma_R = e^{max(|v_R(t)|) - v_{R_{max}}} \,, \tag{13}$$

in the same way, $\gamma_L = e^{max(|v_L(t)|) - v_{L_{\max}}}$ is defined in terms of the velocity of the left robot wheel. Finally, ϕ_R, also a weighted metric, is defined for the robot's right wheel in terms of the time $t_{outside}$ during which the robot's velocity is outside of the chosen boundaries as

$$\phi_R = e^{t_{outside}} \,, \tag{14}$$

where $t_{outside}$ is given in milliseconds. The corresponding metric ϕ_L for the left wheel is defined accordingly.

The performance p of a given trajectory is then computed applying the metrics described above, as follows

$$p = MSE_x + MSE_y + \gamma_R + \gamma_L + \phi_R + \phi_L \,. \tag{15}$$

Given a desired trajectory, the robot performance is evaluated for different initial conditions, resulting in a set of p_i performances; the average value over such p_i, defines the fitness of the solution. This is

$$F = \frac{1}{N_{ic}} \sum_{i=1}^{N_{ic}} p_i \,, \tag{16}$$

Table 4
A set of the *learned* behavior modifiers $\boldsymbol{u}_{gp_i}$, $i = 1, \ldots, 15$, with the best performance is shown alongside their fitness value, computed according to (11) with $t_f = 100\,[s]$, $T_s = 1 \times 10^{-3}\,[s]$. Each behavior modifier is applied to the mobile robot in the form of a nonlinear controller as $\boldsymbol{u}(t) = \boldsymbol{u}_{ct} + \boldsymbol{u}_L$.

Sol	v_{gp}	ω_{gp}	F
$\boldsymbol{u}_{gp_1}$	$\mathrm{erf}(e^{\mathrm{erf}(\mathrm{erf}(\mathrm{erf}(s(v_{Ld}))))})$	$\ln\left(\frac{\sqrt{y_e}}{c(v_{Rd})}\right)$	0.869
$\boldsymbol{u}_{gp_2}$	$\mathrm{erf}(e^{\mathrm{erf}(\mathrm{erf}(\mathrm{erf}(s(v_{Ld}))))})$	$\ln\left(\frac{\sqrt{y_e}}{c^2(v_{Rd})}\right)$	0.866
$\boldsymbol{u}_{gp_3}$	$\mathrm{erf}(e^{\mathrm{erf}(\mathrm{erf}(\mathrm{erf}(v_{Ld})))})$	$\ln\left(\frac{\sqrt{y_e}}{c(v_{Rd})}\right)$	0.863
$\boldsymbol{u}_{gp_4}$	$\mathrm{erf}(cs(\mathrm{erf}(\mathrm{erf}(s(y_d)))))$	$s_{hi}(s_{hi}(s_{hi}(\mathrm{Re}(\frac{t_h(\ln(v_{Rd}))}{\ln(s_{hi}(v_{Rd}))}))))$	0.790
$\boldsymbol{u}_{gp_5}$	$\mathrm{erf}(cs(\mathrm{erf}(\mathrm{erf}(sc(y_d)))))$	$s_{hi}(s_{hi}(s_{hi}(\frac{t_h(\ln(s_{hi}(v_{Rd})))}{\ln(s_{hi}(v_{Rd}))})))$	0.789
$\boldsymbol{u}_{gp_6}$	$\mathrm{erf}(cs(\mathrm{erf}(\mathrm{erf}(sc(y_d)))))$	$s_{hi}(s_{hi}(s_{hi}(\frac{t_h(\ln(s_{hi}(s_{hi}(v_{Rd}))))}{\ln(s_{hi}(v_{Rd}))})))$	0.789
$\boldsymbol{u}_{gp_7}$	$\mathrm{erf}(c_h(\mathrm{erf}(\sqrt{c_h(v_d)})))$	$\sqrt{\sqrt{\dot{y}_d}}$	0.780
$\boldsymbol{u}_{gp_8}$	$th(ct_h(t_h(t_h(ct_h(ct_h(t_h(t_h(ct_h(ct_h(ch(x)))))))))))$	$s(\theta_d)$	0.772
$\boldsymbol{u}_{gp_9}$	$\sqrt{cs_h(sc_h(t(v_{Ld})))}$	$sc_h(c(v_{Ld}))$	0.767
$\boldsymbol{u}_{gp_{10}}$	$\mathrm{erf}(c_h(\mathrm{erf}(c_h(v_{Ld}))))$	$\mathrm{erf}(\mathrm{erf}(c_i(c(c(x_e)))))$	0.766
$\boldsymbol{u}_{gp_{11}}$	$\mathrm{erf}(cs(\mathrm{erf}(\mathrm{erf}(sc(y_d)))))$	$\frac{t_h(\ln(v_{Rd}))}{\ln(s_{hi}(v_{Rd}))}$	0.760
$\boldsymbol{u}_{gp_{12}}$	$\sqrt{cs_h(sc(t(t(v_{Ld}))))}$	$sc_h(c(v_{Ld}))$	0.758
$\boldsymbol{u}_{gp_{13}}$	$\mathrm{erf}(c_h(\mathrm{erf}(c_h(\mathrm{erf}(\sqrt{v_d}))))$	$t_h(\sqrt{\sqrt{\dot{y}_d}})$	0.730
$\boldsymbol{u}_{gp_{14}}$	$s(\mathrm{sgn}(v_{Rd}))$	$s_h(s_h(\min(s_i(y_d - y), \dot{y}_d)))$	0.573
$\boldsymbol{u}_{gp_{15}}$	$\sqrt{cs_h(cs_h(\sqrt{cs_h(sc_h(v_{Ld}))}))}$	$sc_h(c(x_e))$	0.411

Table 5
The fuzzy controller, $\boldsymbol{u}_{fuzzy}$ [25], the CT–based controller, $\boldsymbol{u}_{ct}$ [12], and its saturated version, $\boldsymbol{u}_{ct-sat}$, are listed along with their corresponding fitness values. These can be compared with the fitness of the $\boldsymbol{u}_{gp_i}$ controllers in Table 4.

Control	Expression	F
$\boldsymbol{u}_{fuzzy}$	$v_{fuzzy} = c(\theta)z_1 + s(\theta)z_2,$ $\omega_{fuzzy} = -d^{-1}s(\theta)z_1 + d\,c(\theta)z_2;$ where $z_1 = \dot{x}_d + \nu(\dot{x}_d, \tilde{x})\tilde{x}$, and $z_2 = \dot{y}_d + \nu(\dot{y}_d, \tilde{y})\tilde{y}$.	0.053
$\boldsymbol{u}_{ct}$	$\begin{bmatrix} v_{ct} \\ \omega_{ct} \end{bmatrix} = \begin{bmatrix} v_d \cos(\theta_e) \\ \omega_d + k_\theta \sin(\theta_e) \end{bmatrix} + \boldsymbol{K}\begin{bmatrix} x_e \\ v_d y_e \end{bmatrix}$	0.033
$\boldsymbol{u}_{ct-sat}$	$v_{ct-sat} = \begin{cases} v_{\max} & \text{if } v_{ct} \geq v_{\max} \\ v_{ct} & \text{if } v_{\min} \leq v_{ct} \leq v_{\max} \\ v_{\min} & \text{if } v_{ct} \leq v_{\min} \end{cases}$ $\omega_{ct-sat} = \begin{cases} \omega_{\max} & \text{if } \omega_{ct} \geq \omega_{\max} \\ \omega_{ct} & \text{if } \omega_{\min} \leq \omega_{ct} \leq \omega_{\max} \\ \omega_{\min} & \text{if } \omega_{ct} \leq \omega_{\min} \end{cases}$	0.21

where N_{ic} is the number of initial conditions used.

3.2.4. The Genetic Programming setup

The proposed computational process employed to discover and test the arising *learned* behaviors is depicted in Fig. 4. The desired trajectory to test the performance of the mobile robot is generated by a lemniscate of Huygens, Gerono, or figure-eight curve; its parametric equations are given by

$$x_d(t) = a\sin(\omega t), \tag{17}$$

$$y_d(t) = b\sin(\omega t)\cos(\omega t), \tag{18}$$

where a and b define the length and width of the lemniscate's lobes, and $\omega = 2\pi/s_t$.

The lemniscate of Gerono is the antihyperbolism of a circle with respect to its center and a tangent. This trajectory has vertical tangents at $(\pm a, 0)$ and horizontal tangents at $\left(\pm\frac{1}{2}\sqrt{2}a, \pm\frac{1}{2}a\right)$. Let us redefine the parameterized reference given in (17), (18), into an equivalent structure

$$\dot{\boldsymbol{\xi}}_d(t) = \begin{bmatrix} \cos(\theta_d(t)) & 0 \\ \sin(\theta_d(t)) & 0 \\ 0 & 1 \end{bmatrix}\begin{bmatrix} v_d(t) \\ \omega_d(t) \end{bmatrix}, \tag{19}$$

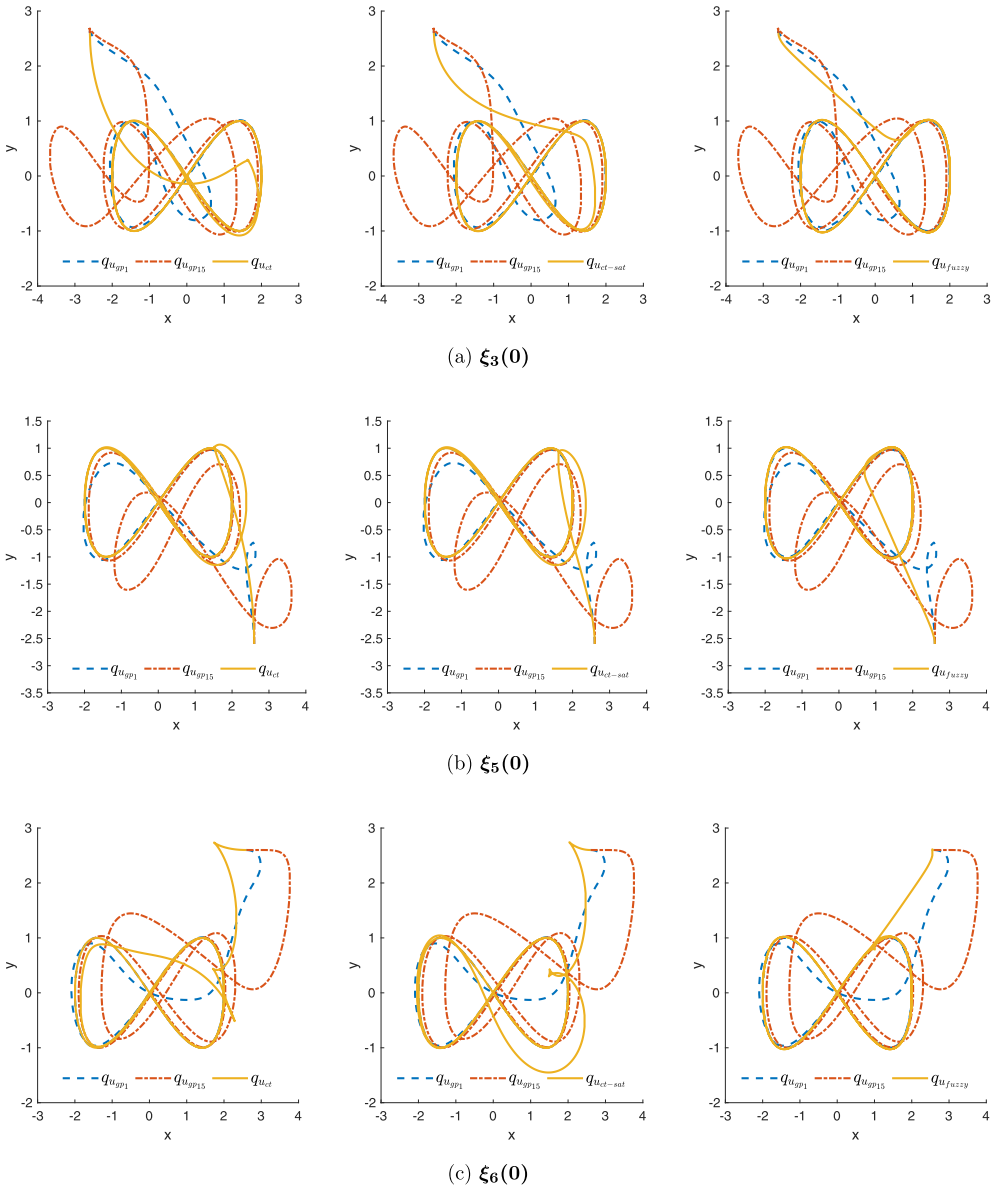


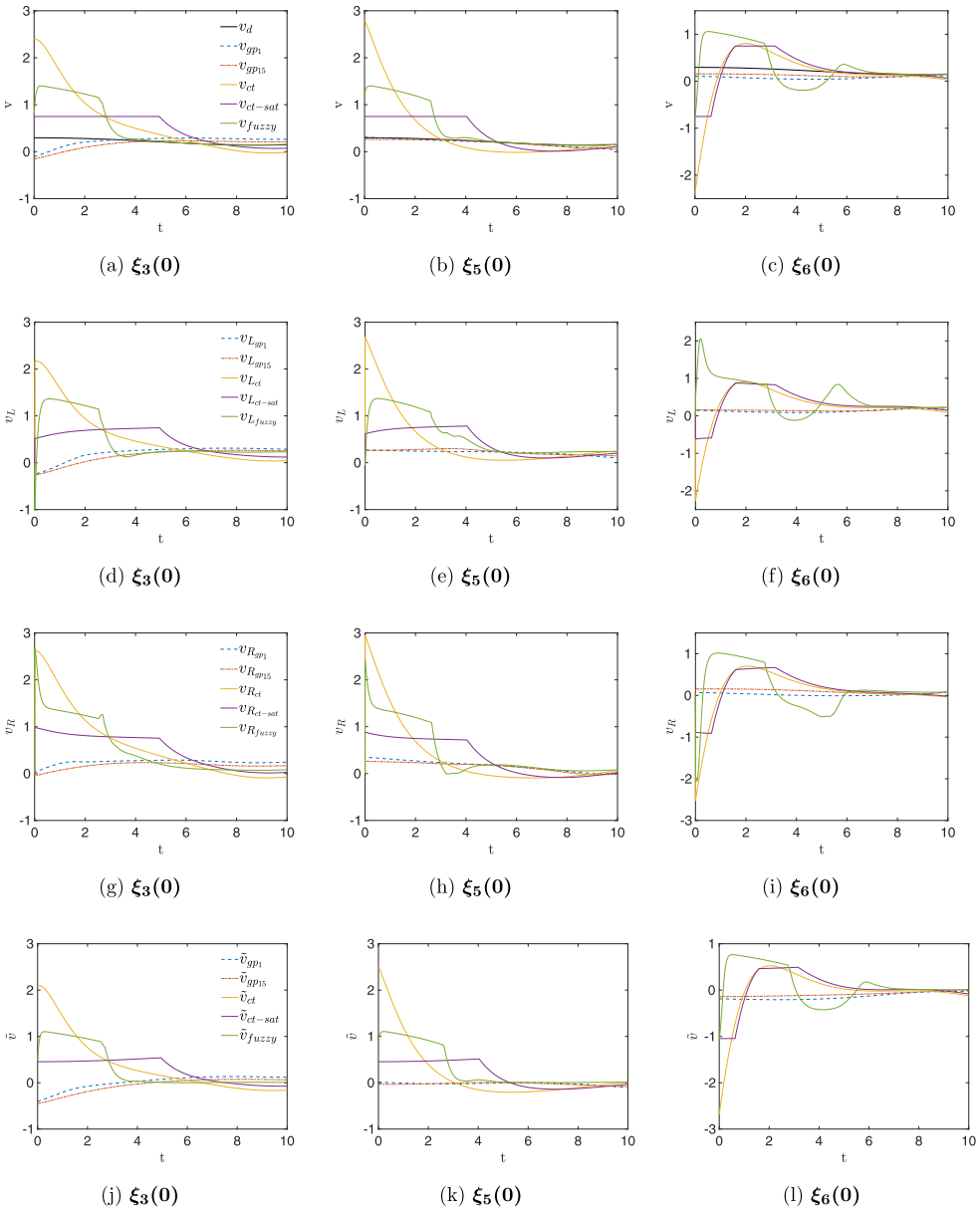
Fig. 7. Position tracking performance comparison between the proposed solutions $\boldsymbol{u}_{GP1}$ (dashed blue line), $\boldsymbol{u}_{GP15}$ (dashed-dotted red line), and the $\boldsymbol{u}_{ct}$ (leftmost column, solid yellow line), $\boldsymbol{u}_{ct-sat}$ (center column, solid yellow line), and $\boldsymbol{u}_{fuzzy}$ (rightmost column, solid yellow line) controllers. Three initial conditions are presented, in quadrants I, II and IV. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

where $\dot{\boldsymbol{\xi}}_d(t) = [\dot{x}_d(t),\ \dot{y}_d(t),\ \dot{\theta}_d(t)]^T$, corresponds to the velocity vector of each axis of the Cartesian plane, and to the orientation angle. The linear and angular velocities of the desired trajectory (denoted as $v_d(t)$ and $\omega_d(t)$, respectively) are computed as in [2]

$$v_d(t) = \sqrt{\dot{x}_d^2(t) + \dot{y}_d^2(t)},\quad \omega_d(t) = \frac{\dot{x}(t)\ddot{y}_d(t) - \dot{y}_d(t)\ddot{x}_d(t)}{\dot{x}_d^2(t) + \dot{y}_d^2(t)},$$
$$\theta_d(t) = \operatorname{atan2}\left(\frac{\dot{y}_d(t)}{\dot{x}_d(t)}\right), \tag{20}$$

where $\ddot{x}_d(t)$, $\ddot{y}_d(t)$ represent the second temporal derivatives of (17) and (18), respectively. In addition, the required velocity of the wheels for the desired trajectory is given by

$$v_{Rd}(t) = v_d(t) + \frac{l\,\omega_d(t)}{2},\quad v_{Ld}(t) = v_d(t) - \frac{l\,\omega_d(t)}{2}. \tag{21}$$



(a) $\boldsymbol{\xi_3(0)}$ (b) $\boldsymbol{\xi_5(0)}$ (c) $\boldsymbol{\xi_6(0)}$

(d) $\boldsymbol{\xi_3(0)}$ (e) $\boldsymbol{\xi_5(0)}$ (f) $\boldsymbol{\xi_6(0)}$

(g) $\boldsymbol{\xi_3(0)}$ (h) $\boldsymbol{\xi_5(0)}$ (i) $\boldsymbol{\xi_6(0)}$

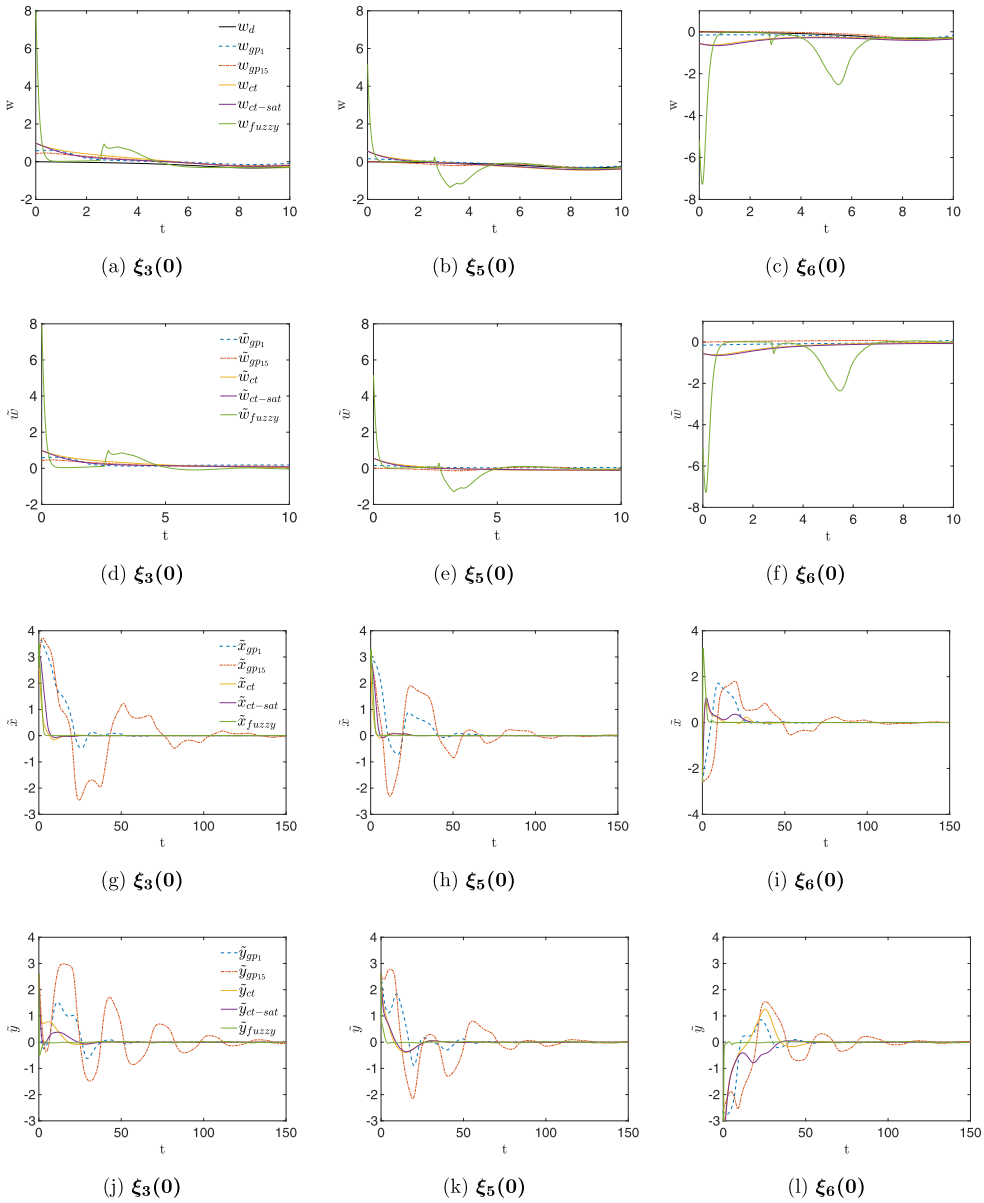
(j) $\boldsymbol{\xi_3(0)}$ (k) $\boldsymbol{\xi_5(0)}$ (l) $\boldsymbol{\xi_6(0)}$

Fig. 8. Linear velocity tracking performance comparison obtained by the proposed solutions $\boldsymbol{u}_{GP1}$ (dashed blue line), $\boldsymbol{u}_{GP15}$ (dashed-dotted red line), and the $\boldsymbol{u}_{ct}$ (solid yellow line), $\boldsymbol{u}_{ct-sat}$ (solid purple line), and $\boldsymbol{u}_{fuzzy}$ (solid green line) controllers. The velocities obtained from the found GP controllers fully respect the saturation bounds, in contrast with the CT and fuzzy controllers. The last row illustrates the linear velocity errors for each control law. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

The initial conditions $x_d(0) =$, $y_d(0) = 0\,[m]$, and $\theta_d(0) = 0\,[rad]$ are chosen for the reference system (19), (20). The parameters of the Lemniscate of Gerono are set as $a = 2$, $\omega = \pi/30\,[\text{rad/s}]$.

The performance evaluation is applied to the differential equations of the nonholonomic mobile robot in (1), subject to the control input $\boldsymbol{u}(t)$ as defined in (3), where $\boldsymbol{u}_F$ and $\boldsymbol{u}_L$ are given as in (9) and (10), respectively. A fourth-order Runge–Kutta method with a fixed-step size $T_s = 1 \times 10^{-3}\,[s]$ is used for a total simulation time $t_f = 100\,[s]$.

The *unforced* behavior of the nonholonomic mobile robot is given solely by its initial conditions, without control input (i.e., $\boldsymbol{u}(t) = 0$). The selected set of initial conditions is defined as $\boldsymbol{\xi}_i(0) = [x(0),\ y(0),\ \theta(0)]^T = \{[2.6, 2.6, 3\pi/2]^T,\ [-2.6,\ 2.6,\ 0]^T,\ [-2.6,\ 2.6,\ 3\pi/2]^T,\ [2.6,\ -2.6,\ \pi]^T,\ [2.6,\ -2.6,\ \pi/2]^T,\ [2.6, 2.6, 0]^T\}[m, m, rad]^T$, for $i = 1, \ldots, 6$. Notice that initial conditions in all quadrants of the inertial frame have been chosen. Some elements of the set of initial conditions are selected taking into account those values where the linear velocities of each wheel exceed the defined maximum values of $v_{R_{\max}}$ and $v_{L_{\max}}$ when applying the CT-based controller $\boldsymbol{u}_{ct}$ solely. The maximum allowable velocities for each wheel



(a) $\boldsymbol{\xi_3(0)}$ (b) $\boldsymbol{\xi_5(0)}$ (c) $\boldsymbol{\xi_6(0)}$

(d) $\boldsymbol{\xi_3(0)}$ (e) $\boldsymbol{\xi_5(0)}$ (f) $\boldsymbol{\xi_6(0)}$

(g) $\boldsymbol{\xi_3(0)}$ (h) $\boldsymbol{\xi_5(0)}$ (i) $\boldsymbol{\xi_6(0)}$

(j) $\boldsymbol{\xi_3(0)}$ (k) $\boldsymbol{\xi_5(0)}$ (l) $\boldsymbol{\xi_6(0)}$

Fig. 9. Angular velocity tracking performance comparison (Fig. 9a–c) obtained by the proposed solutions $\boldsymbol{u}_{GP1}$ (dashed blue line), $\boldsymbol{u}_{GP15}$ (dashed-dotted red line), and the $\boldsymbol{u}_{ct}$ (solid yellow line), $\boldsymbol{u}_{ct-sat}$ (solid purple line), and $\boldsymbol{u}_{fuzzy}$ (solid green line) controllers. For each initial condition, the errors with respect to the desired angular velocity ω_d are shown in Fig. 9d–f; the errors in the x and y coordinates are, respectively, shown in Fig. 9g–i and 9 j–l. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

are set as $v_{R_{\max}} = v_{L_{\max}} = 0.35\,[\mathrm{m/s}]$. Let us remark that a trade-off between the number of selected scenarios against the computational load must be considered.

The gain values for the *forced* behavior induced by the CT-based controller (9) are selected as $\boldsymbol{K} = \mathrm{diag}\{1, 1\}$ and $k_\theta = 1$. These values are chosen such that they fulfill the Lyapunov stability criterion that guarantees convergence to the desired trajectory. The generation of the *learned* behaviors is induced by each pair of individuals proposed in the evolutionary process (denoted as v_{gp} and ω_{gp}). They are embedded as part of the controller $\boldsymbol{u}_L$, and ranked according to their performance by applying the fitness function described by (11).

The tuning of the evolutionary process is empirical, that is, it is a trial-and-error process based on the designer's expertise, then the best parameters are chosen according to each application. In this case, the parameters for the evolutionary process are chosen as indicated in Table 3. Hence, the process is executed with 100 generations and a population size of 400 individuals for 30 runs. The ramped half-and-half method is applied, as proposed by [16]. The roulette wheel selection method is employed for the selection of the individuals. Besides, the crossover operator is applied with a higher probability

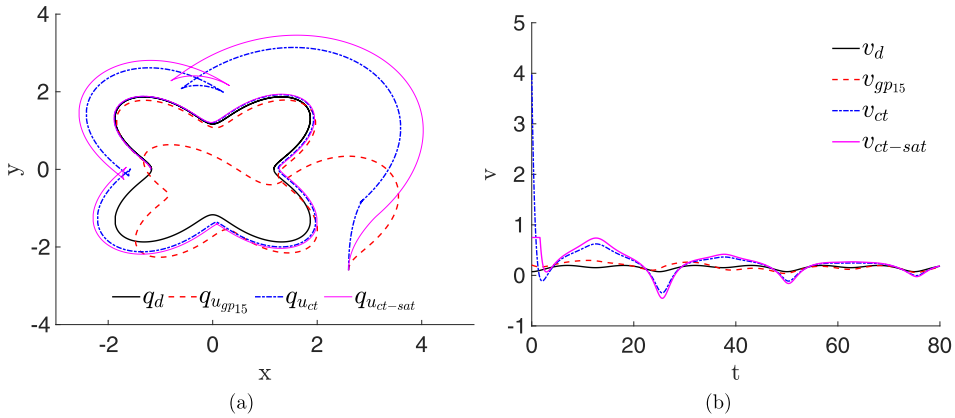


Fig. 10. Fig. (a) shows the position tracking performance comparison between the proposed solution $\boldsymbol{u}_{GP15}$ (dashed red line), the $\boldsymbol{u}_{ct}$ (dashed-dotted blue line), and $\boldsymbol{u}_{ct-sat}$ (solid magenta line) controllers under Daisy-like trajectory. Fig. (b) depicts the linear velocities demanded by each controller. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

Fig. 11. Experimental platform Pioneer 2-AT mobile robot.

than that of the mutation operator. A lexicographic parsimony pressure, seeking for smaller syntactical trees, is used and an elitism method is applied to select the best individual of each generation.

3.2.5. Statistics

In this section, a statistical analysis of the evolutionary process is presented. The aim is to show the convergence rate of the proposed algorithm. A solution is considered suitable when its fitness is higher than 0.21, which is the score given by the fitness function to the saturated version of the CT controller $\boldsymbol{u}_{ct-sat}$ described in Table 5.

The best and average fitness over the 30 runs, together with their standard deviation, are shown in Fig. 5a. Note that, suitable solutions are discovered as early as in the 1st generation, moreover, from the 3rd generation the average population performance is similar to that of the saturated $\boldsymbol{u}_{ct}$ solution. The maximum fitness is achieved around the 45th generation, and the average fitness converges to 0.4 from the 14th generation.

Fig. 5 b depicts the average variety of solutions along with the generations. One can observe that from iteration 15 the diversity converges to around 40%, which is congruent with the data in Fig. 5a.

The structural complexity of the solutions is determined by the number of levels and nodes in their syntactical tree. Fig. 5c to f show the average structural complexity for the first and the second syntactical trees. In Fig. 5c and e one can observe that the average number of levels, in both syntactical trees, converge to 4. Also, the number of nodes in both cases converges to 5; convergence in the first tree is achieved around the 15th generation, while in the case of the second tree, it is attained near the 34th generation.

Fig. 6 illustrates the usage rate of the terminals and functions, defined in Tables 1 and 2, to construct the proposed solutions. Fig. 6a and b show the usage percentage of functions and terminals for the v_{gp} function, while Fig. 6c and d do the same for w_{gp}. The x-axis shows the index of the terminals or functions, according to Tables 1 and 2. For instance, in

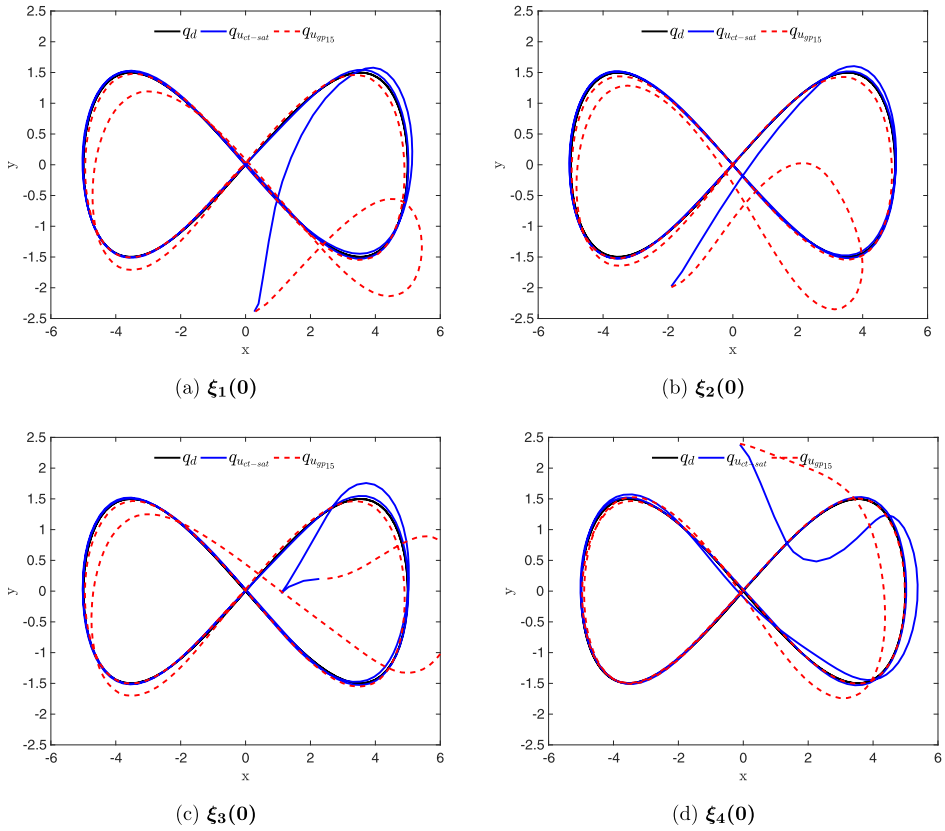


Fig. 12. Experimental results of position tracking performance comparison between the proposed solution, $\boldsymbol{u}_{gp_{15}}$ (dashed-dotted red line), and the $\boldsymbol{u}_{ct}$ (solid blue line) controller. Four initial conditions are presented, in quadrants I, III and IV. The proposed controller shows a smoother response than $\boldsymbol{u}_{ct-sat}$, in both cases convergence to the reference (solid black line) is achieved. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

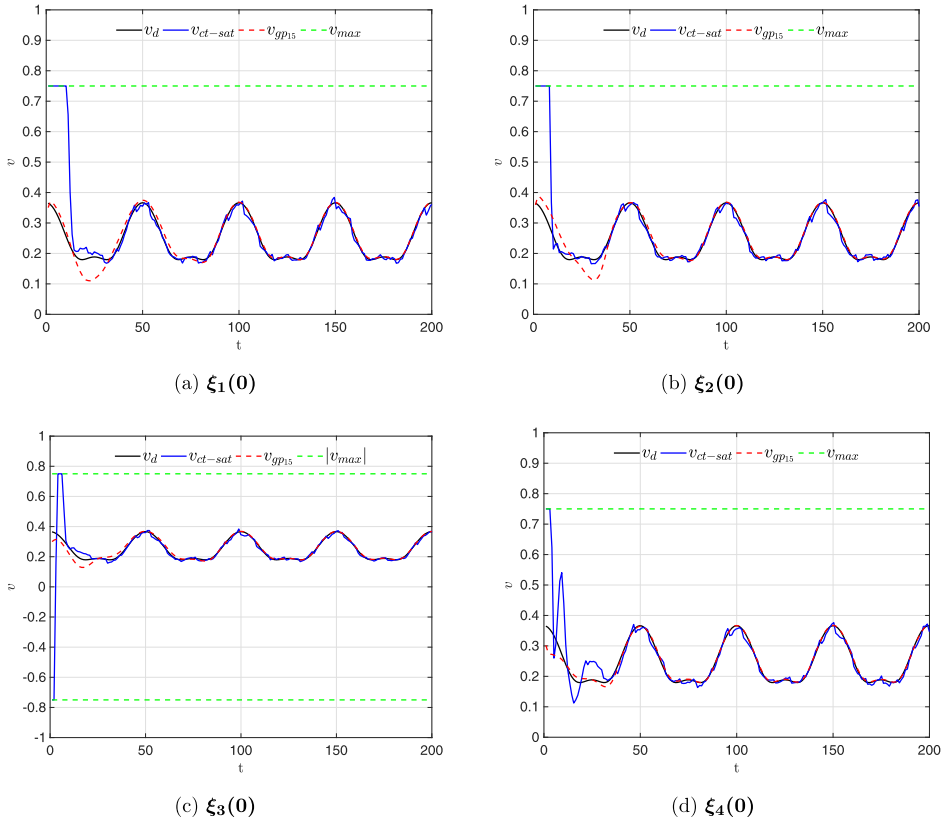
Fig. 6a, one can appreciate that the most used function to generate v_{gp} is function number 22, which corresponds to the error function ($erf(\cdot)$); the most used terminal, in this case, is terminal number 12 which is the desired velocity in the right wheel v_{Rd}, see Fig. 6b. It is interesting to note that the GP found $erf(\cdot)$ as an alternative to $\tanh(\cdot)$, which is commonly used in CT schemes to approximate a saturation nonlinearity. As mentioned above, the most used terminal for v_{gp} is v_{Rd}, and this could be because the velocity of both left and right wheels is related to one another, thus, restricting one of them would necessarily limit the other. A similar analysis can be made for w_{gp}; the most used function, in this case, is $\sinh(\cdot)$ and the most used terminal is θ, the angular position of the robot. The $\sinh(\cdot)$ function is passive, which in the CT approach can be used to describe a stability condition. Notice as well that θ was the GP's first choice for controlling the angular velocity, which is a natural result.

4. Simulation and comparison of discovered behaviors

Let $\boldsymbol{u}_{gp}$ be defined as a set of N *learned* fittest behavior modifiers discovered through the evolutionary process. This set of behavior modifiers are part of the nonlinear controllers' $\boldsymbol{u}_L$ defined in (10). They aim to fulfill simultaneously the desired motion $\boldsymbol{\xi}_d(t)$ in the nonholonomic mobile robot and the boundedness of the velocity of its wheels to a suitable value within its physical specifications.

Table 4 lists fifteen solutions that were selected for performance comparison. They were chosen according to the following criteria: first, all solutions are ordered according to their fitness value, from highest to lowest; then, thirteen solutions are selected from this set. The first seven solutions ($\boldsymbol{u}_{gp_1}$ – $\boldsymbol{u}_{gp_7}$) are those with the upmost fitness among all; the next six ($\boldsymbol{u}_{gp_8}$ – $\boldsymbol{u}_{gp_{13}}$), are selected to show the diversity of the partial controllers obtained in the learning process. Additionally, solutions $\boldsymbol{u}_{gp_{14}}$ and $\boldsymbol{u}_{gp_{15}}$ are chosen because of their low structural complexity; note that they are composed of functions that are different from those in the first thirteen solutions. Also, none of the controllers in Table 4 reach the limits established by the velocity bound.

Controllers $\boldsymbol{u}_{fuzzy}$ and $\boldsymbol{u}_{ct}$ are presented in Table 5, together with their corresponding fitness. Here, $\boldsymbol{u}_{fuzzy}$ represents the fuzzy controller from [25] and $\boldsymbol{u}_{ct}$ is the CT–based controller from [12]. These are chosen to make a performance comparison



(a) $\boldsymbol{\xi_1(0)}$ (b) $\boldsymbol{\xi_2(0)}$

(c) $\boldsymbol{\xi_3(0)}$ (d) $\boldsymbol{\xi_4(0)}$

Fig. 13. Experimental results of linear velocity tracking performance comparison obtained by the proposed solution $\boldsymbol{u}_{gp_{15}}$ (dashed-dotted red line), and the $\boldsymbol{u}_{ct}$ (solid blue line) controller. The velocities obtained by the found GP controller remain close to the desired velocity v_d (solid black line), whereas the CT controller reaches the saturation bounds during the transient time for all tested initial conditions. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

of the proposed approach against a soft computing and a CT–based controller. The last function in the table, $\boldsymbol{u}_{ct-sat}$, is the saturated $\boldsymbol{u}_{ct}$ controller. This saturated version of the CT controller is also evaluated since the actuators are implicitly limited in practice.

The velocity constrained fuzzy controller $\boldsymbol{u}_{fuzzy}$ used in the comparison is taken from [25]. This controller depends on two so-called "Fuzzy Velocity Compensators" (FVC), denoted by $\nu(\dot{x}_d, \tilde{x})$ and $\nu(\dot{y}_d, \tilde{y})$. The FVCs limit the magnitude of the linear velocities ($\dot{x}$ and $\dot{y}$) in the mobile robot. Each compensator is composed of six rules and six gain values. Same configuration, scenarios, and initial conditions are used to evaluate its fitness.

Next, the numerical simulations that were realized to illustrate the performance of two found GP controllers are presented. These are compared against two controllers taken from the literature. In particular, the selected GP controllers $\boldsymbol{u}_{gp_1}$ and $\boldsymbol{u}_{gp_{15}}$ are compared against the $\boldsymbol{u}_{ct}$ controller [12], its saturated version, $\boldsymbol{u}_{ct-sat}$, and Resende's fuzzy controller $\boldsymbol{u}_{fuzzy}$ [25]. The above controllers are given in Tables 4 and 5.

The performance of these controllers was evaluated for three different initial conditions, each located in one quadrant of the Cartesian plane. These conditions were selected from the bank of initial conditions that were given to the GP during the learning process. The simulation results show the tracking performance, the linear and angular velocities for the robot aand for each wheel, and the position and velocity errors, obtained for each initial condition. The initial conditions (defined previously) are $\boldsymbol{\xi}_3(0) = [-2.6,\ 2.6,\ 3\pi/2]^T$, $\boldsymbol{\xi}_5(0) = [2.6,\ -2.6,\ \pi/2]^T$, $\boldsymbol{\xi}_6(0) = [2.6,\ 2.6,\ 0]^T$.

Fig. 7 presents the tracking results for all of the considered controllers, and each selected initial condition. In this example, the trajectory to be followed is a Gerono lemniscate, given by Eqs. (17) and (18), with parameters $a = b = 2$, $\omega = \pi/30\,[\text{rad/s}]$. The dashed, blue trajectory represents the results obtained with controller $\boldsymbol{u}_{gp_1}$, and the dashed-dotted, red line shows the behavior under $\boldsymbol{u}_{gp_{15}}$. The solid, yellow line represents either $\boldsymbol{u}_{ct}$ (leftmost column), $\boldsymbol{u}_{ct-sat}$ (center column), or $\boldsymbol{u}_{fuzzy}$ (rightmost column). It should be noted that while the CT or fuzzy controllers attain the reference earlier than the GP-based controllers, they do so by demanding harsh turns on the robot. This behavior can be more clearly appreciated for condition $\boldsymbol{\xi_6(0)}$, (Fig. 7c). The GP-based controllers, on the other hand, generate smooth trajectories, that reach the reference, while avoiding turns that are too sharp. It is true that the GP controllers take longer to bring the robot's position to the reference, however, the reason for this, is that the velocities demanded by the GP controllers are much lower than

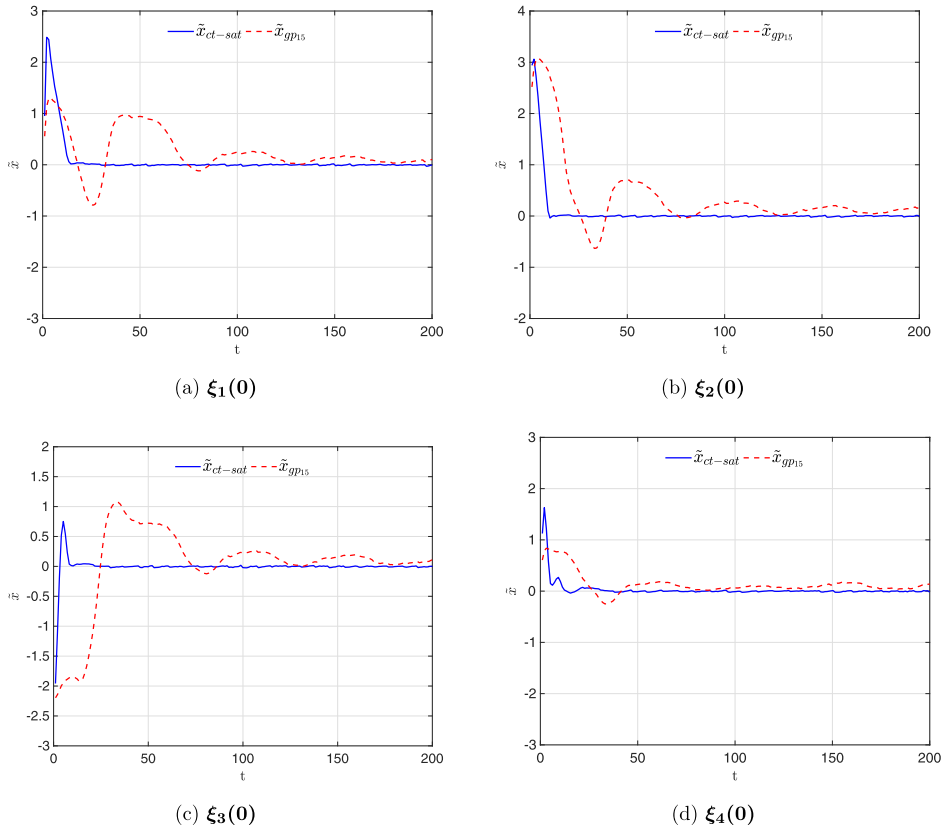


Fig. 14. Experimental results of the tracking errors in the x-coordinate are illustrated for the found solution $\boldsymbol{u}_{gp_{15}}$ (dashed red line), and for the CT controller $\boldsymbol{u}_{ct-sat}$ (solid blue line). In both cases, the errors converge asymptotically to zero. In accordance with the demanded velocities, the settling time is longer for the GP controller than the CT controller. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

those demanded by the CT or the fuzzy-based controllers. This can be observed in the graphs in Fig. 8 (for the linear velocities), and in Fig. 9a–c (for the angular velocity). Notice that the velocities obtained from the GP approach lie well below the saturation constraints of the robot and that they reach the desired velocity faster than their counterparts, for all three conditions. Finally, Fig. 9g-l show the position tracking error in the x and y coordinates.

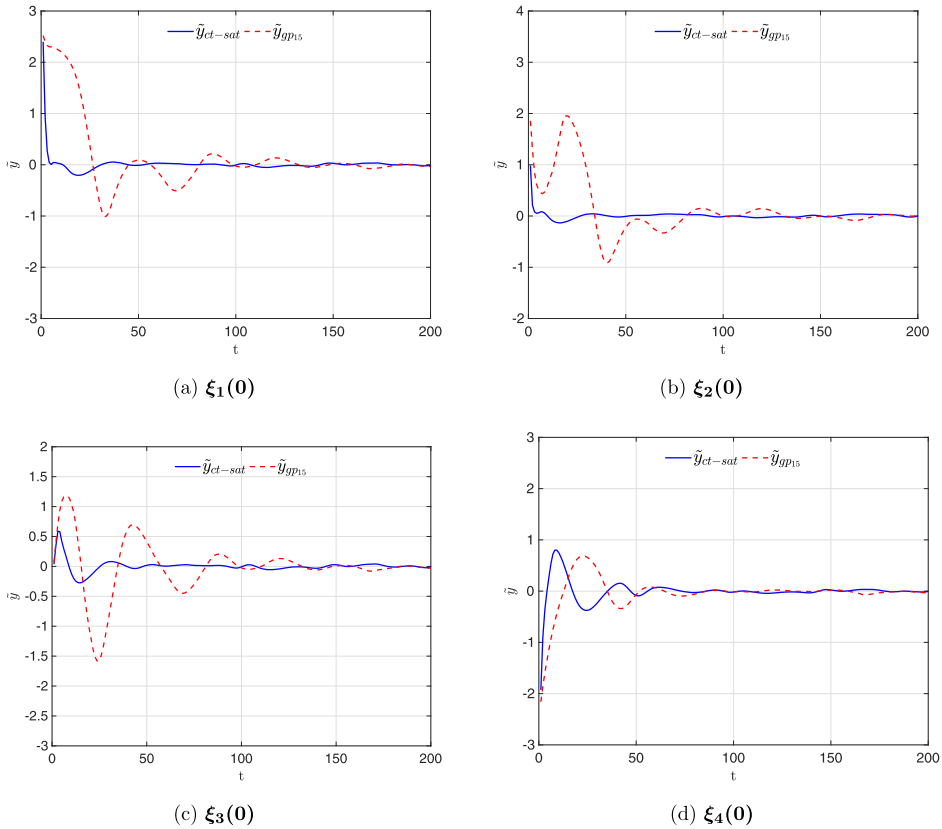
A new trajectory is proposed to evaluate the performance of the found solutions. This trajectory is defined by the parametric equation of a Daisy-like shape, as follows [30]

$$x_d(t) = x_0 + ab\cos(\omega t) - br\cos(m\omega t)\cos(\omega t), \tag{22}$$

$$y_d(t) = y_0 + ab\sin(\omega t) - br\cos(m\omega t)\sin(\omega t). \tag{23}$$

The trajectory of reference is centered at the coordinates (α, β) [m] in the Cartesian space with an external radius of $b_t(a_t + r_t)$. The number of petals for the trajectory is given by the parameter m, and $\omega = \frac{2\pi}{t_t}$, where t_t equals the time in seconds that it takes to make a whole turn. As in the Lemniscate of Gerono from the evolutionary process, the computation of a reference system in the form (19) and (20) for this trajectory is also used.

Given the behavior shown by the $\boldsymbol{u}_{gp_{15}}$ solution in the previous simulations, it is chosen as an example to show the performance of the found solutions against the CT and the saturated CT controllers. Fig. 10 shows a comparison of the position tracking capabilities of the controllers. It can be observed that the GP controller provides a smooth path, while the CT-based controllers generate sharp turns due to the high velocities that the controllers demand to the actuators, see Fig. 10b. In this case, the GP controller accomplishes a linear velocity that is always within the imposed limits. This simulation demonstrates that, even when the Daisy-like shape trajectory was not considered as a reference during training, the found solutions are still capable of achieving the performance requirements of tracking and velocity boundedness.



(a) $\boldsymbol{\xi_1(0)}$ (b) $\boldsymbol{\xi_2(0)}$

(c) $\boldsymbol{\xi_3(0)}$ (d) $\boldsymbol{\xi_4(0)}$

Fig. 15. Experimental results of the tracking errors in the y-coordinate are illustrated for the found solution $\boldsymbol{u}_{gp_{15}}$ (dashed red line), and for the CT controller $\boldsymbol{u}_{ct-sat}$ (solid blue line). In both cases, the errors converge asymptotically to zero. In accordance with the demanded velocities, the settling time is longer for the GP controller than the CT controller. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

5. Application of discovered behaviors in a pioneer 2-AT mobile robot

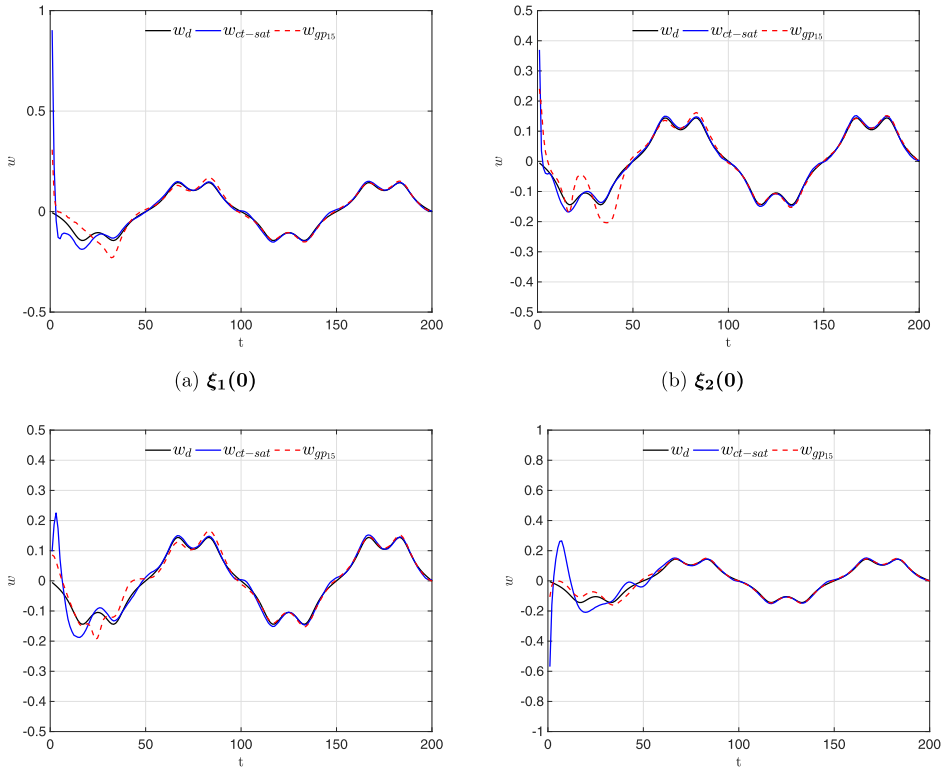
Real-time experiments are performed in a Pioneer 2-AT Mobile Robot for tracking control with constrained velocity. The Pioneer 2-AT is a four-wheel-drive mobile robot, shown in Fig. 11, manufactured by *Adept Mobile Robots*. This prototype can be modeled as a nonholonomic mobile robot with an internal band coupling each pair of wheels, which are located on either side of the robot. The values of its physical parameters and characteristics are extracted from the user manual. The radius of each wheel is $r = 0.105$ [m], and the distance between both wheels, which defines the center of the mobile robot, is $l = 0.48$ [m]. The measurement of the displacement of each wheel is done with encoder sensors with a resolution of 36,000 pulses per revolution [ppr]. Odometry is used to compute the relative position of the robot in the inertial frame *XOY*.

For the application in the Pioneer 2-AT mobile robot, the tracking problem is addressed using the reference system with different parameters to generate the desired trajectories, and changing initial conditions of the mobile robot. Hence, the robustness, scalability, and generality of the discovered behaviors are also illustrated experimentally.

The selected parameter values for the desired trajectory must consider that the velocities required to track it must be physically reachable by the mobile robot. Additionally, special attention is put in the width of the curve for the desired lemniscate of Gerono trajectory. The presence of skid-slip effects was not considered in this work, and the size of the mobile robot must be taken into account given the sharp curves of the trajectories.

The experiments compare the performance of the saturated version of the original CT-based controller u_{ct} [12], and a selected discovered *learned* behavior induced by a control law $u_b = u_{ct} + u_L$. The description of the compared tracking controllers is summarized below.

- **Case 1.** The CT-based controller $\boldsymbol{u}_{ct}$ is defined as in (9), subject to the velocity constraints set by the robot's physical limits. This model is the arbitrarily selected state-of-the-art nonlinear controller, used individually for solving the tracking control problem in the nonholonomic mobile robot. This controller only induces the *forced* behavior of the system, where convergence to the desired trajectory is guaranteed. Tuning considerations are taken into account according to [12].



(a) $\boldsymbol{\xi_1(0)}$ (b) $\boldsymbol{\xi_2(0)}$

(c) $\boldsymbol{\xi_3(0)}$ (d) $\boldsymbol{\xi_4(0)}$

Fig. 16. Experimental results of the angular velocity demanded by the $\boldsymbol{u}_{ct-sat}$ controller (solid blue line) is higher than the velocity required by the GP solution, $\boldsymbol{u}_{gp_{15}}$, for all tested initial conditions. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

Notice that this control input will be saturated as in $\boldsymbol{u}_{ct-sat}$ by the Pioneer 2-AT Mobile Robot according to its datasheet. Hence, the maximum operational values for its velocities are $v_{\max} = \pm 0.75$ [m/s] and $\omega_{\max} = \pm 1.72$ [rad/s], and $\boldsymbol{u}_{ct-sat}$ will be implemented for safety reasons.

- **Case 2.** The synthetic-analytic behavior-based nonlinear tracking controller applies one discovered solution from the evolutionary process. It is implemented in the form of (3) as $\boldsymbol{u}_b(t)$, where $\boldsymbol{u}_{ct}$ and $\boldsymbol{u}_L$ are defined by (9) and (10), respectively. The controller $\boldsymbol{u}_L$ uses solution $\boldsymbol{u}_{gp_{15}}$ from Table 4. This discovered behavior modifier is selected due to its balance in fitness and structural simplicity.

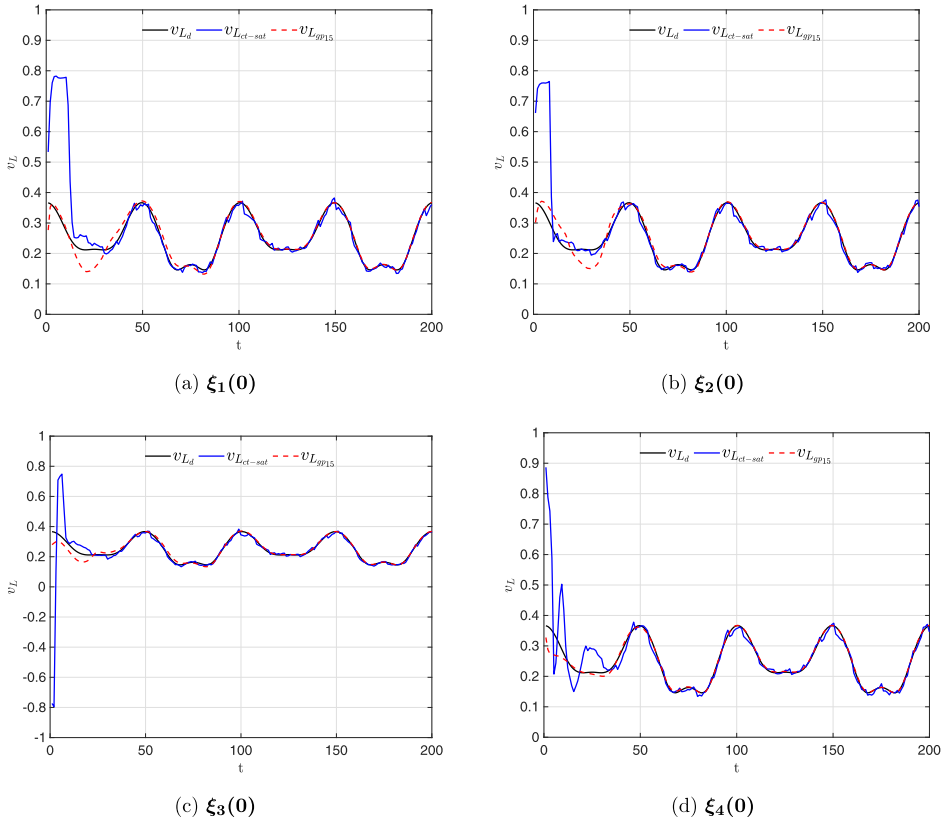
5.1. Experiments

The controllers are implemented in the MobileRobots' Advanced Robot Interface for Applications (ARIA), which is a C library specially designed for the Pioneer 2-AT robot. Both the $\boldsymbol{u}_{ct-sat}$ and the $\boldsymbol{u}_L$ solutions are represented as analytic expressions, making them feasible to be implemented in a real robot. Note, that the computation time of the implementation depends on the number of operations in each controller; for example, for the $\boldsymbol{u}_{ct-sat}$, the system performs at most ten operations, while for the selected $\boldsymbol{u}_L$ controller, it performs 8 operations in total. These values mean that both controllers are feasible and have a constant running time.

The implementation in the Pioneer robot was realized using a sampling time $T_s = 1 \times 10^{-2}$ [s]. The system should follow the reference trajectory defined by Gerono's lemniscate, given in equations (17)–(18), with parameters $a = 5$, $b = 3$, and $s_t = 100$. Then, the maximum values required by the reference linear and angular velocities are $v_d(t) = [0.1795,\ 0.3664]^T$ [m/s], $\omega_d(t) = \pm 0.1442$ [rad/s], with $[v_{Rd}(t),\ v_{Ld}(t)] = [0.1457,\ 0.3667]$ [m/s], for the respective wheel.

The control gains for $\boldsymbol{u}_{ct-sat}$ are chosen as $K = diag\{1, 1\}$ and k_θ=0.5 considering the robot's specifications. Note that these gains were also used for $\boldsymbol{u}_{ct}$ in $\boldsymbol{u}_b$.

The selected set of initial conditions is defined as $\{\boldsymbol{\xi}_1(0) = [0.2,\ -2.4,\ 0]^T,\ \boldsymbol{\xi}_2(0) = [-2,\ -2,\ 0]^T,\ \boldsymbol{\xi}_3(0) = [2.4,\ 0.2,\ 0]^T,\ \boldsymbol{\xi}_4(0) = [0,\ 2.4,\ 0]^T\}$. The experimental results obtained are shown in Figs. 12–18.



(a) $\boldsymbol{\xi_1(0)}$ (b) $\boldsymbol{\xi_2(0)}$

(c) $\boldsymbol{\xi_3(0)}$ (d) $\boldsymbol{\xi_4(0)}$

Fig. 17. Experimental results of the left wheel velocities for the proposed solution $\boldsymbol{u}_{gp_{15}}$ (dashed-dotted red line), and the $\boldsymbol{u}_{ct-sat}$ (solid blue line) controller are shown; the desired velocity v_{L_d} is shown by a solid line black. The CT controller is saturated and demands much higher velocities than the GP controller during transient time for all tested initial conditions. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

Notice that both controllers converge to the desired trajectory defined by the reference system, see Fig. 12. The controller augmented by the GP solution shows a smoother response when it is compared to the CT controller. This is desirable in practice since it can help reduce the effects of nonlinear phenomena, unmodeled dynamics, parameter uncertainties, and unwanted behaviors, like those caused by backlash and friction in the wheels' actuators.

Regarding its performance, the evolved nonlinear controller $\boldsymbol{u}_b$ prevents the robot from operating in saturated mode. This occurs even when the initial conditions and the lemniscate of Gerono's parameters differ from those used during the training stage. This behavior can be observed in Fig. 13, where the linear velocities for both the GP and the saturated CT-based controllers are shown. The velocity generated by $\boldsymbol{u}_b$ never reaches the saturation limits. In contrast, when the CT-based controller is used, the linear velocity saturates, specifically during the transient time. Another undesired phenomenon observed when using the saturated CT-based controller is chattering around the set point, notice that this is avoided when the GP controller is used. Moreover, for initial condition $\boldsymbol{\xi}_3(0)$ (see Fig. 13c), the velocity demanded by the CT controller shows a commutation between its lowest and highest allowable values. For the position tracking task, this implies a sharp change of direction (see Fig. 12c). In contrast, when the GP controller is active, the highest linear velocity is obtained for the second initial condition, $\boldsymbol{\xi}_2(0)$, with $v(t) = 0.386$ [m/s].

As expected, when using $\boldsymbol{u}_b$, it takes the system more time to settle and achieve the desired trajectory since it produces velocities which are lower than that of the CT controller, $\boldsymbol{u}_{ct-sat}$, see Figs. 14 and 15. However, $\boldsymbol{u}_{ct-sat}$ needs higher velocities to achieve tracking, while $\boldsymbol{u}_b$ satisfies the tracking objective while keeping the velocities close to their desired values. This behavior allows for smoother performance in comparison with the trajectories obtained by $\boldsymbol{u}_{ct-sat}$. More specifically, for $\boldsymbol{u}_b$, the highest angular velocity (in each direction) is obtained for the first initial condition as $\omega(t) = 0.3078$ [rad/s] (and $\omega(t) = -0.2298$ [rad/s], for the negative direction), see Fig. 16. Similarly, the highest velocity in the left wheel is obtained for initial condition $\xi_1(0)$ as $v_L(t) = 0.3725$ [m/s], and the highest right wheel velocity is obtained for the second initial condition, $\boldsymbol{\xi}_2(0)$, with $v_R(t) = 0.4262$ [m/s], see Figs. 17 and 18. Note that all of these values are well under the defined saturation level. In contrast, the implementation of u_{ct-sat} always initiates at saturated mode, that is, for all of the initial tested conditions.

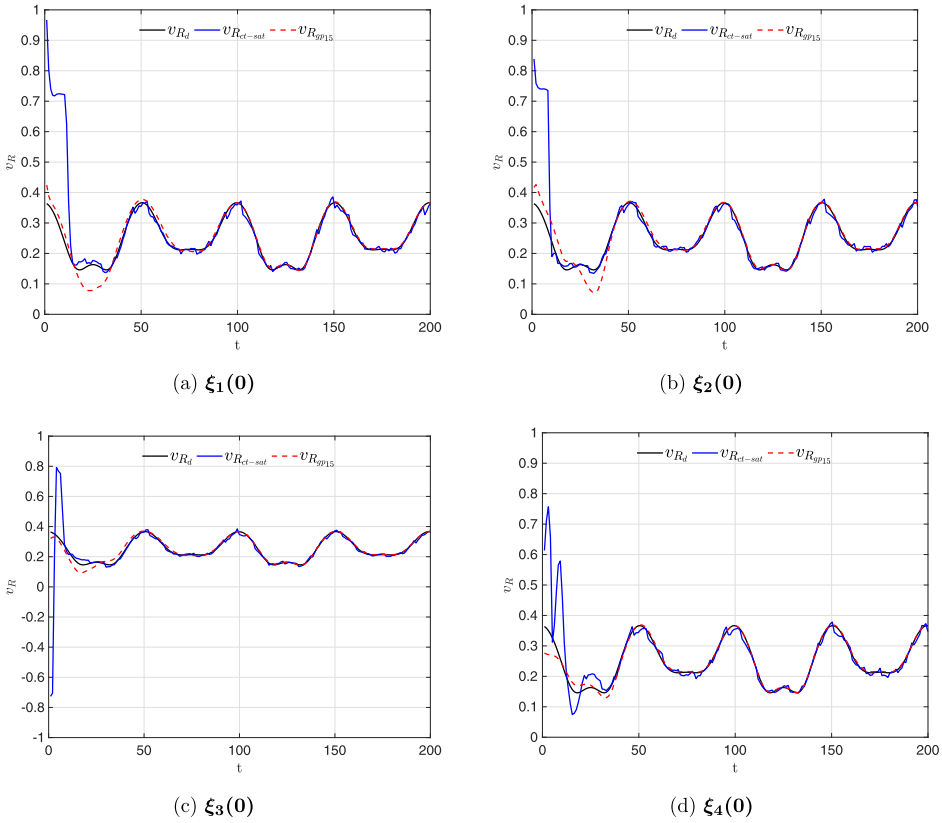


Fig. 18. Experimental results of the right wheel velocities for the proposed solution $\boldsymbol{u}_{gp_{15}}$ (dashed-dotted red line), and the $\boldsymbol{u}_{ct-sat}$ (solid blue line) controller are shown; the desired velocity v_{R_d} is shown by a solid line black. The CT controller operates in saturation mode and demands much higher velocities than the GP controller during transient time for all tested initial conditions. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

6. Conclusions

This work proposed the design of nonlinear controllers for tracking in nonholonomic wheeled mobile robots subject to velocity constraints, using the novel synthetic-analytic behavior-based control framework. This methodology uses Genetic Programming tools and the Control Theory approach to generate analytic solutions, in contrast with those obtained by other Soft Computing techniques. One advantage of the proposed framework is that it allows the evaluation of the solutions' performance, as their stability, using analytic CT tools, not available to the majority of the soft computing techniques.

The proposed strategy automates the controller design task. The most important stages are the definition of the policies for the evolutionary process, and the selection of the function to evaluate the fitness of each proposed solution; this strategy is straightforward. In this work, the CT approach is used to state such policies taking advantage of its analytic nature, which is its greatest strength. Then, the training scenarios are selected, and the GP algorithm is defined according to sought performance and computational load. The selection of the implemented solution among the best is left to the practitioner in terms of task requirements and physical specifications.

Regarding the tracking control problem at hand, and as a result of applying this methodology, 9113 solutions were obtained that better the performance of state–of–the–art controllers. The best fifteen solutions were selected, giving priority to those with the highest fitness, the lower structural complexity, and the variety of the functions that integrate them. This last criterion was considered to show the diversity of the found solutions. In this case, the search for the results was guided such that the linear and angular velocities remained within the desired velocity bounds for each wheel of the robot.

The selected controller, denoted as $\boldsymbol{u}_b$, which uses solutions $\boldsymbol{u}_{gp_1}$ and $\boldsymbol{u}_{gp_{15}}$, was compared in simulation against a CT-based controller, a saturated version of the CT-based controller, and a fuzzy controller. For the experimental setup, the comparison was performed using the saturated CT controller. In both cases, the controllers were evaluated using several initial conditions, which were different from the ones used in the learning process. In all cases, only the proposed GP solutions accomplished the tracking task under the defined velocity restrictions. Moreover, the u_b performance with $\boldsymbol{u}_{gp_{15}}$ was compared in simulation assuming a trajectory different than the training stage reference. This was done to show how some solutions could be generalized to other scenarios. The stability analysis of the selected solutions using the CT approach is left as future work, as this would demonstrate the generality of the solutions.

The reach of the methodology proposed in this paper goes further than the particular problem presented here. This method can give insight into the development of new control structures, that differ from those commonly reported in the literature according to the variety of the functions in the discovered solutions.

Declaration of Competing Interest

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

Acknowledgments

This work was partially supported by TecNM project 6474.18-P. The authors would like to thank to Consejo Nacional de Ciencia y Tecnología, Tecnológico Nacional de México/Instituto Tecnológico de Ensenada, Centro de Investigación Científica y de Educación Superior de Ensenada, and the bachelor student Rodrigo Alexandro Villalvazo Covián for his assistance with the experiments.

References

[1] R.C. Arkin, Behavior-Based Robotics (Intelligent Robotics and Autonomous Agents), The MIT Press, 1998. ISBN 0262011654
[2] S. Blažič, On periodic control laws for mobile robots, IEEE Trans. Ind. Electron. 61 (7) (2014) 3660–3670. ISSN 0278-0046. doi: 10.1109/TIE.2013.2287222.
[3] R.A. Brooks, Intelligence without representation, Artif. Intell. 47 (1) (1991) 139–159. ISSN 0004-3702. doi: 10.1016/0004-3702(91)90053-M.
[4] G. Campion, G. Bastin, B. Dandrea-Novel, Structural properties and classification of kinematic and dynamic models of wheeled mobile robots, IEEE Trans. Robot. Autom. 12 (1) (1996) 47–62. ISSN 1042-296X. doi: 10.1109/70.481750
[5] X. Chen, Y. Jia, Simple tracking controller for unicycle-type mobile robots with velocity and torque constraints, Trans. Inst. Meas. Control 37 (2) (2015) 211–218, doi:10.1177/0142331214537294.
[6] E. Clemente, M. Meza-Sánchez, E. Bugarin, A.Y. Aguilar-Bustos, Adaptive behaviors in autonomous navigation with collision avoidance and bounded velocity of an omnidirectional mobile robot, J. Intell. Robot. Syst. 92 (2) (2018) 359–380. ISSN 1573-0409. doi: 10.1007/s10846-017-0751-y.
[7] M.C. Fang, Y.Z. Zhuo, Z.Y. Lee, The application of the self-tuning neural network PID controller on the ship roll reduction in random waves, Ocean Eng. 37 (7) (2010) 529–538. ISSN 0029-8018. doi: 10.1016/j.oceaneng.2010.02.013.
[8] D. Floreano, P. Husbands, S. Nolfi, Evolutionary robotics, in: B. Siciliano, O. Khatib (Eds.), Springer Handbook of Robotics, Springer Berlin Heidelberg, 2008, pp. 1423–1451. ISBN 978-3-540-30301-5. doi: 10.1007/978-3-540-30301-5_39.
[9] J. Huang, C. Wen, W. Wang, Z.P. Jiang, Adaptive stabilization and tracking control of a nonholonomic mobile robot with input saturation and disturbance, Syst. Control Lett. 62 (3) (2013) 234–241. ISSN 0167-6911. doi: 10.1016/j.sysconle.2012.11.020.
[10] G. Indiveri, A. Nuchter, K. Lingemann, High speed differential drive mobile robot path following control with bounded wheel speed commands, in: Proceedings 2007 IEEE International Conference on Robotics and Automation, 2007, pp. 2202–2207. ISSN 1050-4729. doi: 10.1109/ROBOT.2007.363647.
[11] Z.P. Jiang, E. Lefeber, H. Nijmeijer, Saturated stabilization and tracking of a nonholonomic mobile robot, Syst. Control Lett. 42 (5) (2001) 327–332. ISSN 0167-6911. doi: 10.1016/S0167-6911(00)00104-3.
[12] Y. Kanayama, Y. Kimura, F. Miyazaki, T. Noguchi, A Stable Tracking Control Method for an Autonomous Mobile Robot, in: Proceedings., IEEE International Conference on Robotics and Automation, 1, 1990, pp. 384–389. doi: 10.1109/ROBOT.1990.126006
[13] A.K. Khalaji, M. Jalalnezhad, Dynamic feedback linearizing controller for a wheeled vehicle, in: 2017 IEEE 4th International Conference on Knowledge-Based Engineering and Innovation (KBEI), 2017, pp. 0088–0094. doi: 10.1109/KBEI.2017.8324940.
[14] D.H. Kim, J.I. Park, Intelligent PID controller tuning of AVR system using GA and PSO, in: H. De-Shuang, Z. Xiao-Ping, H. Guang-Bin (Eds.), Advances in Intelligent Computing: International Conference on Intelligent Computing, ICIC 2005, Hefei, China, August 23–26, 2005, Proceedings, Part II, 2005, pp. 366–375. ISBN 978-3-540-31907-8
[15] Y. Koubaa, M. Boukattaya, T. Damak, Adaptive sliding mode control for trajectory tracking of nonholonomic mobile robot with uncertain kinematics and dynamics, Appl. Artif. Intell. (2018) 1–15, doi:10.1080/08839514.2018.1519100.
[16] J. Koza, M. Keane, J. Yu, F.H. Bennett, W. Mydlowec, Automatic creation of human-competitive programs and controllers by means of genetic programming, Gen. Program. Evol. Mach. 1 (1) (2000) 121–164. ISSN 1573-7632. doi: 10.1023/A:1010076532029.
[17] M. Matarić, Interaction and Intelligent Behavior, Cambridge, MA, USA, 1994 Ph.d. thesis.
[18] M. Matarić, Designing and understanding adaptive group behavior, Adapt. Behav. 4 (1995) 51–80.
[19] M. Matarić, F. Michaud, Behavior-based systems, in: B. Siciliano, O. Khatib (Eds.), Springer Handbook of Robotics, Springer Berlin Heidelberg, ISBN 978-3-540-30301-5, 2008, pp. 891–909. doi: 10.1007/978-3-540-30301-5_39.
[20] T.P. Nascimento, C.E.T. Dórea, L.M.G. Goncalves, Nonlinear model predictive control for trajectory tracking of nonholonomic mobile robots: a modified approach, Int. J. Adv. Robot. Syst. 15 (1) (2018) 1–14, doi:10.1177/1729881418760461.
[21] R. Oftadeh, R. Ghabcheloo, J. Mattila, A Time-optimal Bounded Velocity Path-following Controller for Generic Wheeled Mobile Robots, in: 2015 IEEE International Conference on Robotics and Automation (ICRA), 2015, pp. 676–683. ISSN 1050-4729. doi: 10.1109/ICRA.2015.7139252.
[22] O. Peñaloza Mejía, E. Clemente, M. Meza-Sánchez, C. Pérez, F. Chavez, GP-Based Motion Control Design for the Double-integrator System Subject to Velocity Constraint, GECCO '17 Companion, 2017. Berlin, Germany
[23] O. Peñaloza Mejía, E. Clemente, M. Meza-Sánchez, C.B. Pérez, Evolving behaviors for bounded-flow tracking control of second-order dynamical systems, Eng. Appl. Artif. Intell. 78 (2019) 12–27. ISSN 0952-1976. doi: 10.1016/j.engappai.2018.10.001.
[24] R. Rajamani, Lateral and longitudinal tire forces, in: R. Rajamani (Ed.), Vehicle Dynamics and Control, 2nd, Springer US, Boston, MA, 2012, pp. 355–396. ISBN 978-1-4614-1432-2. doi: 10.1007/978-1-4614-1433-9.
[25] C.Z. Resende, R. Carelli, M. Sarcinelli-Filho, A nonlinear trajectory tracking controller for mobile robots with velocity limitation via fuzzy gains, Control Eng. Pract. 21 (10) (2013) 1302–1309. ISSN 0967-0661. doi: 10.1016/j.conengprac.2013.05.012.
[26] G. Sandou, Metaheuristic Optimization for the Design of Automatic Control Laws, Wiley-ISTE, 2013. ISBN 978-1-84821-590-0
[27] R. Saravanakumar, G. Rajchakit, .C.K. Ahn, .H.R. Karimi, Exponential stability, passivity, and dissipativity analysis of generalized neural networks with mixed time-varying delays, IEEE Trans. Syst. Man Cybern. 49 (2) (2019) 395–405, doi:10.1109/TSMC.2017.2719899.
[28] M.E. Serrano, G.J.E. Scaglia, F.A. Cheein, V. Mut, O.A. Ortiz, Trajectory-tracking controller design with constraints in the control signals: a case study in mobile robots, Robotica 33 (10) (2015) 2186–2203, doi:10.1017/S0263574714001325.
[29] K. Shojaei, Output-feedback formation control of wheeled mobile robots with actuators saturation compensation, Nonlinear Dyn. 89 (4) (2017) 2867–2878. ISSN 1573-269X. doi: 10.1007/s11071-017-3631-x.
[30] H. Sira-Ramírez, C. López-Uribe, M. Velasco-Villa, Linear observer-based active disturbance rejection control of the omnidirectional mobile robot, Asian J. Control 15 (1) (2013) 51–63, doi:10.1002/asjc.523.

[31] C. Sowmiya, R. Raja, Q. Zhu, G. Rajchakit, Further mean-square asymptotic stability of impulsive discrete-time stochastic BAM neural networks with markovian jumping and multiple time-varying delays, J. Frankl. Inst. 356 (1) (2019) 561–591, doi:10.1016/j.jfranklin.2018.09.037.
[32] S.G. Tzafestas, 2 - mobile robot kinematics, in: S.G. Tzafestas (Ed.), Introduction to Mobile Robot Control, Elsevier, Oxford, 2014, pp. 429–478. ISBN 978-0-12-417049-0. doi: 10.1016/B978-0-12-417049-0.00011-0.