ISSN 2007-9737

Prescribed-Time Trajectory Tracking Control of Wheeled Mobile Robots Using Neural Networks and Robust Control Techniques

Jesus A. Rodríguez-Arellano[1], Víctor D. Cruz[1], Luis T. Aguilar[1,*], Roger Miranda-Colorado[2,3]

[1] Instituto Politécnico Nacional, Tijuana, Mexico

[2] Instituto Politécnico Nacional, CITEDI, Mexico

[3] Cátedras CONAHCyT, Mexico

jrodriguez@citedi.mx, vcruz@citedi.mx, laguilarb@ipn.mx, rmirandaco@conacyt.mx

Abstract. This research presents a novel trajectory generation algorithm and the design of a prescribed time controller for trajectory tracking tasks for autonomous vehicles. The trajectory generation algorithm uses a hybrid combination of computer vision techniques and intelligent rail detection methods using an on-board camera. Based on the previous information, a possible trajectory is then generated that the vehicle should follow. A time-prescribed controller is then developed and implemented to track the trajectory generated by the proposed methodology. The controller uses a hybrid structure in which a time-varying feedback controller transitions into a fixed-time controller. This approach achieves stabilization in the prescribed time despite the initial conditions. To address the trajectory design, a scaled autonomous vehicle simulator was used to then evaluate the prescribed time controller compared to a finite time controller and a dynamic feedback controller. The simulation results demonstrate the effectiveness of trajectory generation and trajectory tracking control algorithms in addressing these challenges in real-world scenarios by examining two situations: unperturbed and perturbed cases.

Keywords. Prescribed time stabilization, trajectory generation, neural networks.

1 Introduction

Research on autonomous vehicles is intensively explored due to their increasing scope of application, such as surveillance tasks, space exploration, delivery, transportation, and others [1, 2, 3, 4]. These vehicle systems require information about the environment to correctly perform the various tasks and avoid collisions with various obstacles in the environment. The various tasks that these vehicles perform are classified into three main problems: trajectory tracking, path tracking, and point stabilization [5,6].

All information is collected by the sensors these systems are equipped with, such as B. LiDAR, ultrasonic sensors, GPS, and cameras. The information of the environment is processed and interpreted to perform the above-mentioned navigation tasks and avoid accidents and collisions [4, 7]. Therefore, the main focus of this study consists of the precise control of the vehicle's movement along a desired path, known as trajectory tracking, as well as the generation of such paths, known as trajectory generation. These topics have attracted significant attention in the field of autonomous vehicles [8]. A wide range of studies have been carried out on the generation of trajectories.

Prior studies [9, 10, 11, 12] have shown that it is possible to implement intelligent techniques for generating trajectories and segmenting lanes. A technique for lane line identification is proposed by the authors in [13]. This approach utilizes the RANSAC algorithm to aid in the detection of lane lines and is predicated on an adaptive region of interest extraction strategy.

In addition, convolutional neural network-based algorithms for lane line detection are proposed by Haixia and colleagues [14]. For training and validation purposes, their algorithms employ the TuSimple dataset. The implementation of a Neural Network (NN) using an on-board camera in [9] leads to favorable outcomes in the trajectory generation process.

Hart et al. [10] describe the implementation of intelligent methodologies to create a viable trajectory generator. The study's findings indicate that it is feasible to create trajectories using intelligent methodologies. However, Bellusci et al. [11] present a new approach to lane segmentation using neural networks and computer vision techniques to create a map of the surroundings. Nevertheless, it does not explain the process of generating trajectories.

Besides, in [15] the system utilizes a convolutional neural network along with an auxiliary layer to detect the borders of lanes. In addition, the authors suggest a straightforward algorithm to rectify the vehicle's orientation by utilizing the centroid of the drivable area.

Unfortunately, there is no process for generating trajectories. In addition, in [16] was developed an independent navigation system using machine learning and computer vision techniques on a scaled vehicle.

The system also incorporates a depth camera for localization. However, the algorithm's performance is reduced in low-brightness scenarios. Furthermore, a study conducted by Neven et al. [17] focuses on the development of a control system for a Car-Like robot equipped with a vision system.

This system enables the robot to detect and monitor lanes on a road. The study's findings indicate that the utilization of vision techniques leads to effective lane detection in practical situations.

After addressing the trajectory generation problem, the subsequent critical task is solving trajectory tracking. Numerous studies have tackled this challenge, but there are lingering issues in the field. Various control schemes, including feedback control strategies [18], Sliding Mode Control (SMC) [19], and decoupled approaches [20], have been explored.

Furthermore, these controllers use different scenarios that encompass diverse convergence rates [20, 21], the effect of disturbances [19, 22, 23], and different kinematic models [18]. Cui et al. [24] introduced an adaptive control law within SMC, demonstrating exponential convergence to the trajectory.

The proposed methodology implements a decoupled approach to position and orientation tracking. Experimental results indicated its effectiveness for trajectory tracking despite disturbances. Nevertheless, the proposed methodology implements a simplified kinematic model, and the convergence rate is low. Qun Lu et al [21] proposed a fixed-time controller coupled with an observer under kinematic disturbances.

The controller accounted for signal saturation to prevent slipping, yielding satisfactory tracking results. However, it is noteworthy that the convergence exhibited a gradual pace. The employed kinematic model was the simplified version, and the control structure is complex. In [25], the authors presented a prescribed-time containment controller coupled with a prescribed-time observer to achieve leader-follower tasks.

This study employs the effect of uncertainties and external disturbances by using a chain of integrators for the model. The results demonstrated good tracking performance and disturbance rejection. The research on trajectory tracking has been tackled from different perspectives, like finite-time stability, fixed-time stability, and simplified kinematic models.

However, prescribed-time stability has not been widely studied on WMRs, but in [25] it was proved that this methodology can be implemented in these systems by achieving a fast convergence rate and low tracking errors. Thus, it is important to tackle this problem in WMRs because these systems need to attain a fast response in different scenarios where convergence time is crucial.

Furthermore, trajectory generation is a problem that has been studied from different perspectives, however, there are not many studies with onboard cameras on this crucial task.

ISSN 2007-9737

1.1 Contribution

Based on the previous literature review, the contribution of this research is to address the trajectory generation and trajectory tracking tasks for autonomous vehicles.

To attain the trajectory generation problem, we develop a novel algorithm that combines computer vision techniques and NNs that enable us to segment rails and then design a feasible trajectory using an onboard camera.

We generate the trajectory using the Autominy simulator and propose a novel methodology. Furthermore, we design a new prescribed-time controller that drives the vehicle to the desired trajectory despite the effects of the disturbances.

This controller is composed of two stages: initially, a time-varying feedback control drives the system to a neighborhood of the origin; then it switches to a twisting controller that converges in fixed time to the origin.

To implement the proposed controller, we perform a coordinate transformation to the complete kinematic model of a Car-Like robot. Then, a series of simulations are performed between the proposed controller against a finite-time controller and a feedback controller. The trajectory tracked is the reference signal generated by the proposed algorithm.

The results demonstrate that the desired trajectory is a good option for trajectory generation problems, and the proposed controller is also a feasible option for trajectory tracking by demonstrating its superior performance against the compared control schemes. Then, the main contributions are:

– Develop a novel trajectory generation algorithm by combining NNs and computer vision techniques for WMRs using an on-board camera.

– Design of a novel prescribed time controller using the complete kinematic model of a WMR, that attains the trajectory tracking problem despite the effect of kinematic disturbances.

– Validation of the trajectory generated by the proposed algorithm by comparing the proposed controller and control schemes from the literature.

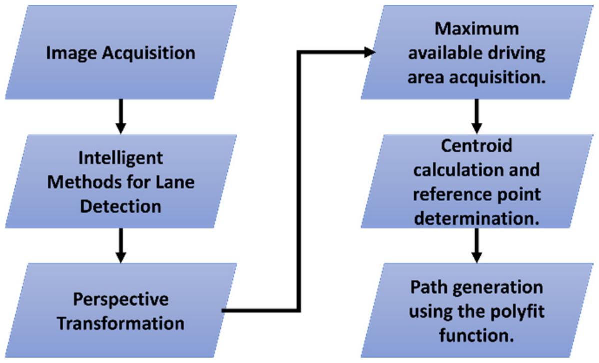
Image Acquisition
Intelligent Methods for Lane Detection
Perspective Transformation
Maximum available driving area acquisition.
Centroid calculation and reference point determination.
Path generation using the polyfit function.

Fig. 1. Trajectory generation methodology

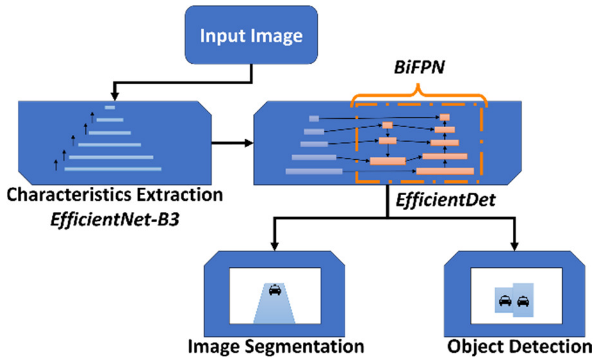

Fig. 2. HybridNets architecture [29]

– Exhaustive qualitative and quantitative study that demonstrates the superiority of the proposed controller against the finite time and dynamic feedback controllers.

1.2 Organization

The subsequent sections are arranged in the following order: Section 2 describes the novel approach for generating trajectories, detailing both the intelligent method and the vision techniques. Section 3 provides a detailed explanation of the kinematic model of a WMR, including a coordinate transformation that allows for the implementation of a hybrid control scheme.

Section 4 develops the controller design that attains the prescribed time stabilization. The outcomes of the trajectory generation methodology are showcased in Section 5, alongside the evaluation of the suggested controller formulated in Section 4. Section 6, ultimately, provides the final findings and results of this manuscript.

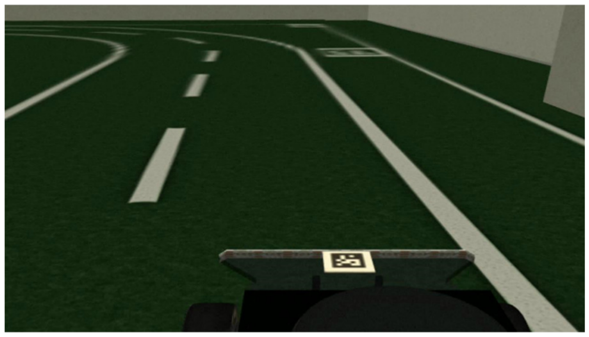

Fig. 3. Image acquired from the on-board camera of the Autominy simulator

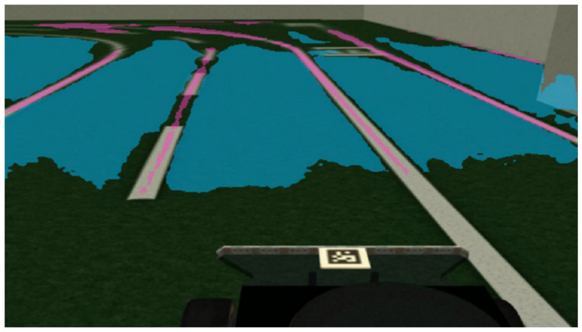

Fig. 4. Output image with HybridNets NN

1.3 Notation

Trigonometric functions are described as c_ψ, s_ψ, t_ψ which corresponds to $\cos(\psi), \sin(\psi)$ and $\tan(\psi)$ correspondingly. The function $\operatorname{sign}(\sigma) = \sigma/|\sigma|$ if $\sigma \neq 0$, and $\operatorname{sign}(0) \in [-1,1]$. $\mathbb{R}_+$ represents positive real numbers.

2 Trajectory Generation

The generation of trajectories is an essential undertaking for WMRs owing to the diverse outcomes it has in real-world situations and the limitations it must satisfy to ensure its tracking is feasible. To achieve this goal, we propose the use of the methodology presented in Fig. 1, which integrates computer vision techniques and neural networks. To achieve this objective, we provide a detailed description of the proposed strategy.

2.1 HybridNets Neural Network

The Neural Network employed is HybridNets1[1] [27, 28], which is an end-to-end perception neural network based on PyTorch. The objective is to address the multi-task issue by employing segmentation and box detection classification networks. Its main architecture comprises two networks, as depicted in Fig. 2.

The initial part of the system is the backbone, which uses the EfficientNet-B3 convolutional neural network architecture to extract characteristics from the input. This architecture scales the dimensions of depth, width, and resolution using a composite coefficient to obtain feature maps of the image.

The information extracted by the backbone is then passed on to the neck network, called EfficientDet, which uses a Weighted Bi-directional Feature Pyramid Network (BiFPN) module for image segmentation and object detection. The BiFPN module achieves this by creating bidirectional interconnections between network nodes. Each input feature is assigned an additional weight, allowing the network to determine the individual significance of each feature[2].

2.2 Proposed Algorithm

To attain the trajectory generation task, we perform a series of steps detailed in Fig. 1, which allows us to finally obtain a feasible trajectory for a WMR using an on-board camera.

To this end, we provide a deep description of this algorithm in this section by using the Gazebo simulator of the Autominy vehicle [31].

Image acquisition. The first step entails reading the input image captured by the vehicle's built-in camera, as shown in Fig. 3.

This step can be realized through the application of the available topics and the Robotic Operating System (ROS) to control the vehicle.

[1] The TuSimple dataset, comprising 6408 images of highways in the United States presented at a resolution of 1280x720, was utilized to train HybridNets [26].

[2] To implement the intelligent method, the NN was converted to an *Open Neural Network Exchange* (ONNX) model to enhance its inference. The codes can be found in [29], while the weights can be found in [30].

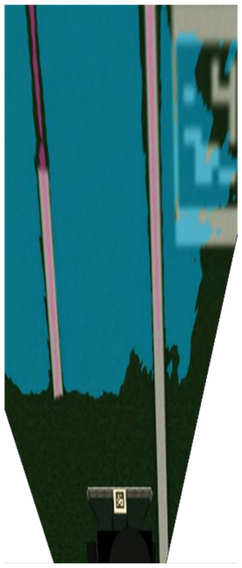

Fig. 5. Bird-Eye-View perspective transformation

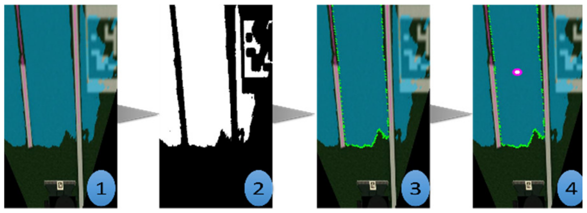

Fig. 6. Procedure for determining the centroid of the available drivable area

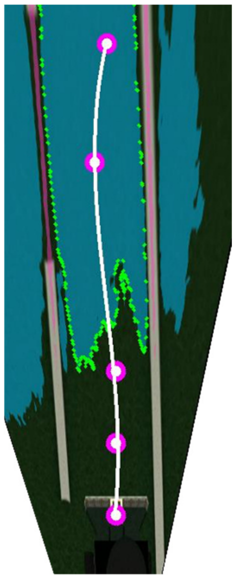

Fig. 7. Reference points for path visualization

Intelligent Methods for Lane Detection. HybridNets NN application is used for lane segmentation and lane detection. This scheme identifies and separates the lane observed by the vehicle's on-board camera. An important advantage of this NN is its ability to accurately identify the lane directly above the vehicle, as shown in Fig. 4.

Perspective Transformation. The output NN image undergoes a perspective transformation technique called "bird's eye view". This technique simulates the image being viewed from a higher angle, similar to a bird's viewpoint [32].

Leveraging this transformation provides a comprehensive perspective of the lane, ensuring both lanes remain parallel to facilitate the implementation of lane detection and segmentation processes. This simplifies the detection and segmentation processes. Figure 5 illustrates the use of the technique from a bird's eye view.

Maximum Drivable Area Acquisition. In our pursuit to determine the largest navigable region, we first examine the area divided by the NN and thus determine the track with the largest extent to generate the desired path. To achieve this, a mask is implemented, which makes use of the segmentation color indicated by the intelligent method results (blue area).

All resulting available segmented areas are then found and the maximum available area is calculated. Subsequent to this phase, the largest area is selected and highlighted in green to proceed with the trajectory generation method.

The algorithm then identifies all currently available segmented areas and uses them to calculate the largest possible area. Next, the algorithm identifies the largest area and marks it with a green marker to proceed with trajectory creation.

Centroid Calculation and Reference Point Determination. The subsequent procedure entails determining the centroid of the drivable area that has been identified through the green contour in the segmented results, as depicted in Fig. 6. Afterward, five points are positioned to create the intended trajectory. Initially, two immobile points are positioned at an equivalent elevation as the ArUco marker on the Autominy vehicle. These points are fixed and serve as the initial positions for displaying the trajectory. In addition, two extra points are included, one located blow, and one above the calculated centroid. The reference points are adjusted according to the area determined by the NN, and they help to determine the trajectory when navigating a curved lane based on the captured image.

Afterward, the visual representation of the trajectory that corresponds to the five points mentioned earlier is displayed in Fig. 7. Additionally, the homography transformation converts the image representing the aerial perspective into the initial image perspective [33].

This transformation is dependent on the ArUco position, which serves as a reference point. This procedure utilizes the marker location and its position within the pixel coordinates on the bird-eye view image to accurately convert them into the corrected perspective pixel coordinates.

Figure 8 illustrates the process of translating the first trajectory to its equivalent on the vehicle's perspective.

Path generation. Ultimately, we derive the equation that defines the reference trajectory. The outcomes of the trajectory generation phase utilizing our suggested approach are depicted in Fig. 9. The trajectory is determined by analyzing the image captured by the on-board camera.

This visual observation is used to determine the trajectory. Figure 10 illustrates the reference trajectory produced using the proposed methodology in conjunction with the Autominy simulator. The controller will receive this trajectory as the reference input.

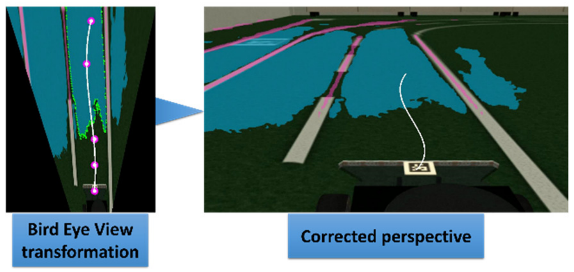

Fig. 8. Perspective correction

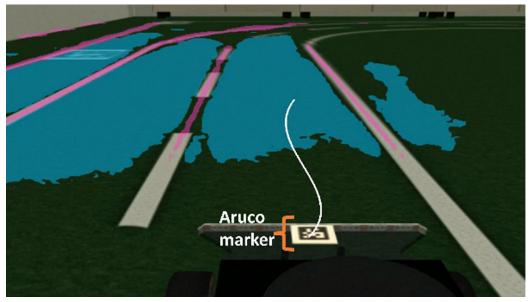

Fig. 9. Results for trajectory generation

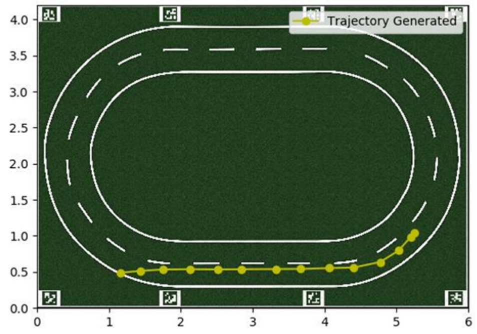

Fig. 10. Path generated using the proposed trajectory generation method

3 Kinematic Model and Control Design

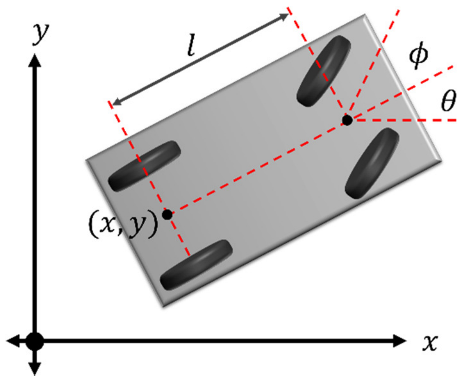

Fig. 11. Description of the WMR's kinematic model in the x, y plane

To tackle the trajectory tracking problem, we aim to design a controller that achieves prescribed-time stability to the desired trajectory. To this end, let us consider the full kinematic model a Car-Like robot depicted in Fig. 11, where $\boldsymbol{q}(t) = [x(t),\ y(t),\ \theta(t),$ $\phi(t)] \in \mathbb{R}^4$ is the system's vector configuration, the $x(t), y(t)$ is the position on the plane with respect to the world frame $\{x, y\}$, $\theta(t)$ is the orientation of the vehicle with respect to the x axis, $\phi(t)$ is the

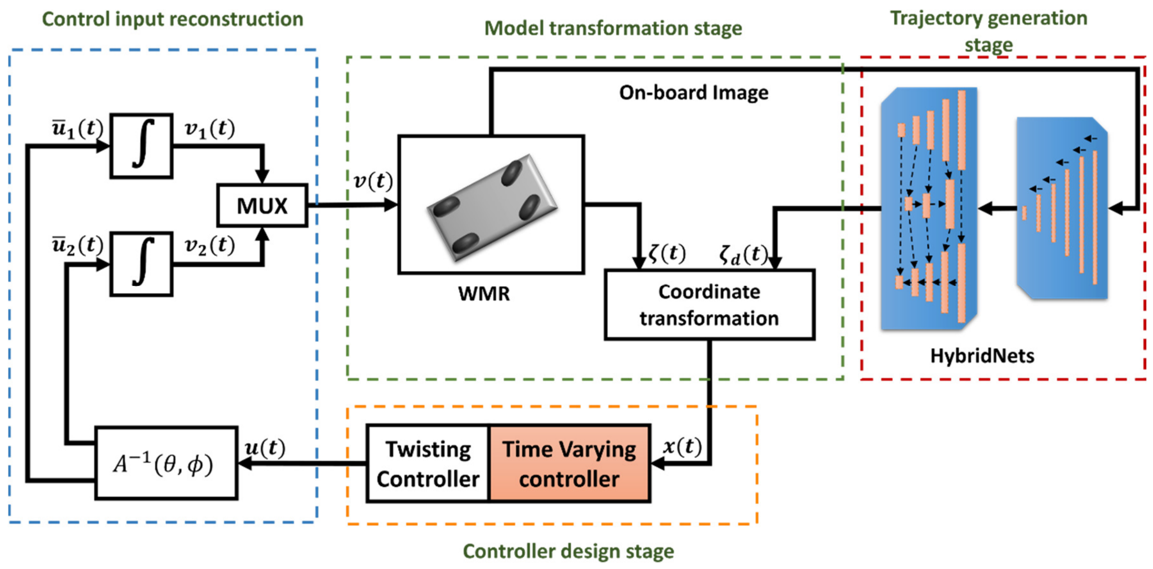

Fig. 12. General scheme of the proposed methodology

steering angle of the front wheels. Now, the kinematic model of the WMR is described as:

$$\dot{\boldsymbol{q}}(t) = S(q)\boldsymbol{v}(t) + \boldsymbol{d}(t), \tag{1}$$

$$S(\boldsymbol{q}) = \begin{bmatrix} c_\theta & s_\theta & \frac{t_\phi}{l} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \quad \boldsymbol{v}(t) = \begin{bmatrix} v_1(t) \\ v_2(t) \end{bmatrix}, \tag{2}$$

With $v(t) = [v_1(t), v_2(t)]^T \in \mathbb{R}^2$ is the control input vector with $v_1(t), v_2(t)$ being the linear and angular velocities, and $\boldsymbol{d}(t) = [d_1(t),\ d_2(t),\ d_3(t),\ d_4(t)]^T \in \mathbb{R}^4$ encompasses the disturbances, which are bounded and smooth until its first time derivative [34].

Furthermore, the actuators generate bounded control signals. Hence, based on the previous statement we are able to consider the following assumption.

Assumption 1. There exist some positive constants $\bar{\mathrm{D}}, \bar{\mathrm{d}}, \mathrm{V}_1^+, \mathrm{V}_2^+$, and Φ such that:

$$||\boldsymbol{d}(t)|| \leq \bar{D}, ||\dot{\boldsymbol{d}}(t)|| \leq \bar{d}, |v_1(t)| \leq V_1^+,$$
$$|v_2(t)| \leq V_2^+, |\phi(t)| \leq \Phi < \pi/2. \tag{3}$$

Then, we perform a coordinate transformation by defining the new output variable:

$$\zeta(\mathrm{t}) = \begin{bmatrix} \zeta_1(t) \\ \zeta_2(t) \end{bmatrix} = \begin{bmatrix} x + l\, c_\theta + \delta c_{(\theta+\phi)} \\ y + l\, s_\theta + \delta\, s_{(\theta+\phi)} \end{bmatrix}. \tag{4}$$

With an arbitrary $\delta \neq 0$, which will be used to design a controller. Furthermore, to attain the trajectory tracking problem, a reference kinematic model is required, which is described by:

$$\dot{\boldsymbol{q}}_{\boldsymbol{d}}(t) = S(q_d)\boldsymbol{v}_{\boldsymbol{d}}(t),$$

$$S(\boldsymbol{q}_{\boldsymbol{d}}) = \begin{bmatrix} c_{\theta_d} & s_{\theta_d} & \frac{t_{\phi_d}}{l} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \tag{5}$$
$$\boldsymbol{v}_{\boldsymbol{d}}(t) = \begin{bmatrix} v_{d1}(t) \\ v_{d2}(t) \end{bmatrix},$$

where $\boldsymbol{q}_{\boldsymbol{d}}\,(t) = [x_d(t),\ y_d(t),\ \theta_d(t),\ {}_d(t)]^T \in \mathbb{R}^4$, $v_{d1}(t), v_{d2}(t) \in \mathbb{R}$ are the reference signals of $\boldsymbol{q}(t)$ and $v_1(t), v_2(t)$. Moreover, we perform the same coordinate transformation as in (4):

$$\boldsymbol{\zeta}_{\boldsymbol{d}}(\mathrm{t}) = \begin{bmatrix} \zeta_{d1}(t) \\ \zeta_{d2}(t) \end{bmatrix} = \begin{bmatrix} x_d + l\, c_{\theta_d} + \delta c_{(\theta_d+\phi_d)} \\ y_d + l\, s_{\theta_d} + \delta\, s_{(\theta_d+\phi_d)} \end{bmatrix}. \tag{6}$$

By following the procedure described in [20] along the coordinate transformation (5) and the reference model (9), we define the tracking error $\tilde{\boldsymbol{\xi}}(t) = \boldsymbol{\xi}(t) - \boldsymbol{\xi}_{\boldsymbol{d}}(t)$, which yields to the following dynamic error structure:

$$\ddot{\tilde{\boldsymbol{\xi}}}(t) = A(\theta,\phi)\bar{\boldsymbol{u}}(t) + \bar{A}(\theta,\phi)\boldsymbol{v}(t) + \dot{\boldsymbol{\Gamma}}(\boldsymbol{t}) - A(\theta_d,\phi_d)\bar{\boldsymbol{u}}_{\boldsymbol{d}}(t) - \dot{A}(\theta_d,\phi_d)\boldsymbol{v}_{\boldsymbol{d}}, \quad (7)$$

where the structure of $A(\theta,\phi)$, $\bar{A}(\theta,\phi)$, $\bar{\boldsymbol{u}}(t)$, $\dot{\boldsymbol{\Gamma}}(t)$, $\bar{\boldsymbol{u}}_{\boldsymbol{d}}(t)$, $\dot{A}(\theta_d,\phi_d)$, and $\boldsymbol{v}_{\boldsymbol{d}}(t)$ can be consulted in [23].

Now we can state the control objective by implementing a hybrid control scheme that makes the tracking error signal $\tilde{\boldsymbol{\xi}}(t)$ to converge to zero in a prescribed time despite disturbances.

Control objective. Considering the WMR's kinematic model (1), the coordinate transformation (4), and the error's dynamics (7), design a control input $\boldsymbol{v}(t)$ such that the tracking error converges to zero in a prescribed time.

4 Control Design

In order to design the proposed controller, depicted in Fig. 12, we begin by presenting a general structure for the controller, which is then applied to the kinematic model (1) by employing the tracking error $\tilde{\boldsymbol{\xi}}(t)$, and the error's dynamics (7) to achieve prescribed time stability to the desired trajectory.

4.1 General Structure

The control problem encompasses two stages. First, the system is directed towards an arbitrary small attraction zone by means of a time-varying state feedback control law.

Once the system enters the attraction zone, a twisting controller is executed to ensure that it converges to the equilibrium point in prescribed time [35].

In order to address this issue, we begin by formulating a general second-order system with the following structure:

$$\begin{aligned} \dot{x}_1(t) &= x_2(t), \\ \dot{x}_2(t) &= u(t,x) + w(t,x), \end{aligned} \quad (8)$$

where $\boldsymbol{x}(t) = [x_1, x_2]^T \in \mathbb{R}^2$ is the state vector, $u(t,\boldsymbol{x}) \in \mathbb{R}$ is the control input, and $w(t,\ \boldsymbol{x}) \in \mathbb{R}$ encompasses smooth and bounded disturbances, such that $|w(t,\ x)| \leq W^+$, with $W^+ \in \mathbb{R}$ being the upper bound of the disturbances. To attain prescribed time stability, we structure the control input as follows:

$$u = \begin{cases} u_1(t,\boldsymbol{x}), & t < T_1 \text{ and } \boldsymbol{x} \in \mathbb{R}^2 \setminus G_R, \\ u_2(\boldsymbol{x}), & \text{otherwise} \end{cases} \quad (9)$$

where $G_R = \{\boldsymbol{x}: V(\boldsymbol{x}) \leq R\}$ is the attraction domain, and:

$$u_1(t,x) = -l_1(t)x_1(t) - l_2(t)x_2(t), \quad (10)$$

$$u_2(x) = -c_1\text{sign}(x_1) - c_2\text{sign}(x_2), \quad (11)$$

$$\begin{aligned} l_1(\text{t}) &= \mu^2(\text{t})[2+m][3+m] + k_1\mu(\text{t})[2+m] \\ &\quad + k_2\mu^{3+m}(\text{t})[2+m] \\ &\quad + k_1k_2\mu^{2+m}(\text{t}), \end{aligned} \quad (12)$$

$$l_2(\text{t}) = 2\mu(\text{t})[2+m] + k_1 + k_2\mu(\text{t})^{2+m}.$$

With $m \in \mathbb{N}$, $k_1, k_2 \in \mathbb{R}_+$, and the function:

$$\mu(t) = \frac{1}{T_1 + t_0 - t}. \quad (13)$$

The control signal $u_1(t,\boldsymbol{x})$ employs some time-varying gains $l_1(t)$ and $l_2(t)$, which act in the first stage of the control strategy in $t < T$. Then the twisting controller, represented by $\text{u}_2(\text{t})$ with some gains $c_1 > c_2 > W^+ > 0$, is introduced once the trajectories of the system enter the attraction domain G_R, defined by the parameter R and the time T. Furthermore, this domain is set by the non-strict Lyapunov function:

$$V(\boldsymbol{x}) = c_1|x_1(t)| + \frac{1}{2}x_2^2(t). \quad (14)$$

Which ensures that the equilibrium point is finite-time stable, according to [36, Theorem 5.1]. The proposed controller attains prescribed-time stabilization by combining two methodologies; the first stage consists in a control structure encompassed by time-varying gains (12) that drives the trajectories of the system to a vicinity of the origin defined by the time T and the attraction domain G_R.

Once the trajectories reach this vicinity, the twisting controller (11) is introduced to reach the origin in fixed time with an admissible disturbance (for more details, refer [35]). Based on the previous information, the controller's implementation for the WMR is described in the following subsection.

4.2 Wheeled Mobile Robot Controller

To implement the hybrid control scheme (9), we first define the following state variables:

$$x_1(t) = \tilde{\xi}_1(t),\ x_2(t) = \dot{\tilde{\xi}}_1(t), \tag{15}$$

$$x_3(t) = \tilde{\xi}_2(t),\ x_4(t) = \dot{\tilde{\xi}}_2(t). \tag{16}$$

Thus, we can rewrite the error's dynamics (7) by using definitions (15, 16), which yields to two decoupled second-order systems:

$$\Upsilon_1 \begin{cases} \dot{x}_1(t) = x_2(t), \\ \dot{x}_2(t) = u_1(t,x) + w_1(t,x), \end{cases} \tag{17}$$

$$\Upsilon_2 \begin{cases} \dot{x}_3(t) = x_4(t), \\ \dot{x}_4(t) = u_2(t,x) + w_2(t,x), \end{cases} \tag{18}$$

where $u_1(t,\boldsymbol{x})$ and $u_2(\boldsymbol{x})$ are the control inputs retrieved from:

$$\boldsymbol{u}(\mathrm{t}) = A(\theta,\phi)\bar{\boldsymbol{u}}(\mathrm{t}). \tag{19}$$

Being:

$$\boldsymbol{u} = [u_1(t,x), u_2(t,x)]^T,$$

$$w_1 = \dot{a}_{11}v_1 + \dot{a}_{12}v_2 + \dot{\lambda}_1 - a_{d11}\dot{v}_{d1} - a_{d12}\dot{v}_{d2} - \dot{a}_{d11}v_{d1} - \dot{a}_{d12}v_{d2}, \tag{20}$$

$$w_2 = \dot{a}_{21}v_1 + \dot{a}_{22}v_2 + \dot{\lambda}_2 - a_{d21}\dot{v}_{d1} - a_{d22}\dot{v}_{d2} - \dot{a}_{d21}v_{d1} - \dot{a}_{d22}v_{d2},$$

where $w_i(t)$ are considered smooth and bounded disturbances, such as $|w_i(x,t)| \leq M_i^+$ with $i \in \{1,2\}$. According to (9), the hybrid controller is structured as [35]:

$$u_i = \begin{cases} u_{1i}, & t < T_{1i} \text{ and } \boldsymbol{x} \in \mathbb{R}^2 \setminus G_{Ri} \\ u_{2i}, & \text{otherwise,} \end{cases} \tag{21}$$

where $T_{1i} > 0$ is a design parameter, and:

$$u_{11} - l_{11}(t)x_1(t) - l_{12}x_2(t), \tag{22}$$

$$u_{12} = -l_{21}(t)x_3(t) - l_{22}x_4(t), \tag{23}$$

$$u_{21} = -c_{11}\mathrm{sign}(x_1) - c_{12}\mathrm{sign}(x_2), \tag{24}$$

$$u_{22} = -c_{21}\mathrm{sign}(x_3) - c_{22}\mathrm{sign}(x_4), \tag{25}$$

$$G_{R1} = \{\boldsymbol{x}: V(x_1,x_2) \leq R_1\}, \tag{26}$$

$$G_{R2} = \{\boldsymbol{x}: V(x_3,x_4) \leq R_2\}. \tag{27}$$

The control $u_{1i}(t,x)$ is a linear time-varying state feedback with positive time-varying gains $l_{1i}(t),\ l_{2i}(t)$ with the structure defined in (12), and the control structure $u_{2i}(x)$ attains stability in finite time to the origin according to the stability analysis developed with the non-strict Lyapunov function (14) [36].

Finally, in order to recover the control inputs $v_1(t)$, and $v_2(t)$ we use the definitions $A(\theta,\phi)$ and $\dot{\boldsymbol{v}}(t)$ from (19), which yields to:

$$\boldsymbol{v}(t) = \begin{bmatrix} v_1(t) \\ v_2(t) \end{bmatrix} = \begin{bmatrix} \int_0^\tau \dot{v}_1(\tau)\,d\tau \\ \int_0^\tau \dot{v}_2(\tau)d\tau \end{bmatrix}, \tag{28}$$

where $\bar{\boldsymbol{u}}(t) = A^{-1}(\theta,\phi)\boldsymbol{u}(t)$. Therefore, we synthesize the stated controller in the following Theorem.

Theorem 1. [35] Let the dynamic's error (7) and assume that **A1** holds. Then, the controller (19), (21)-(28), provided (12, 13), ensures the trajectory tracking in prescribed time.

5 Numerical Results

To assess the proposed approaches for trajectory generation and trajectory tracking tasks, we conducted simulations using MATLAB/SIMULINK® for trajectory tracking and the Autominy simulator for trajectory generation [4]. The trajectory generated via the methodology described in Section 2 is illustrated in Fig. 13.

To assess the performance of the proposed methodology, named **PTC**, we consider the controllers described in references [18] and [20]. We choose the dynamic feedback controller [18] (named **DFC**), due its simple structure and because it implements the complete kinematic model of a Car-Like robot.

Furthermore, we also employ [20] (referred to as **FTC**) because it utilizes the full kinematic model of the Car-Like robot, attains the trajectory tracking problem by using a decoupling approach, and achieves Finite-time stability. The following cases are considered for the validation of the PTC controller:

– **C1**: There is no effect of kinematic disturbances in the model, considering the

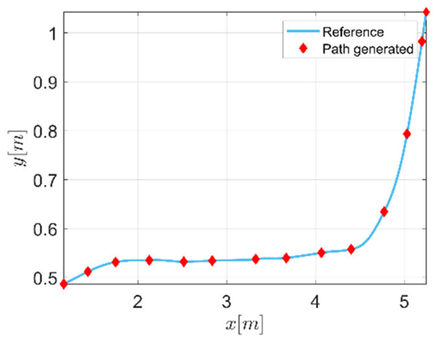

Fig. 13. Reference points for path visualization

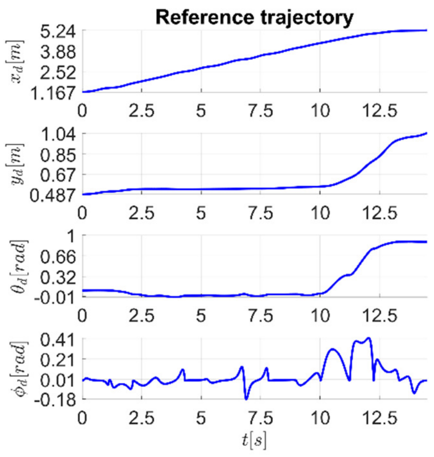

Fig. 14. Desired trajectory represented in generalized coordinates $x(t)$, $y(t)$, $\theta(t)$ and $\phi(t)$

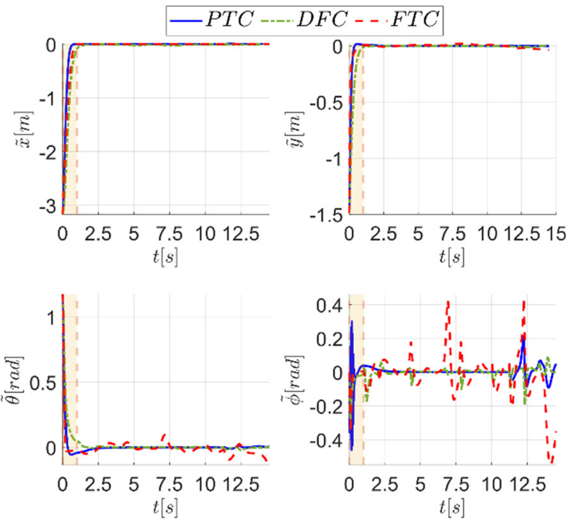

Fig. 15. Tracking errors in $\tilde{x}, \tilde{y}, \tilde{\theta}$and $\tilde{\phi}$ for case **C1**

initial conditions $[x(0),\ y(0),\ \theta(0),\ \phi(0)]^T = [-1,\ -2,\ \arctan(\pi/2)\,,0]^T$ and $l = 0.255$ [m].

– **C2**: A The effect of kinematic disturbances is introduced with $d_1(t) = 0.05 + 0.05\sin(2t)$, $d_2(t) = -0.05 - 0.05\cos(2t)$, $d_3(t) = 0.05$, $d_4(t) = -0.05$, with the initial conditions considered in case **C1**.

The gains considered for the DFC are $k_{pi} = 343$, $k_{vi} = 147$, $k_{ai} = 21$, and for the FTC are $k_1 = 35$, $k_2 = 25$, $k_3 = 5$ and $k_4 = 7$. Finally, the parameters for the PTC's time-varying gains are $T_{1i} = 1$, $k_{11} = 5$, $k_{12} = 5$, $k_{21} = 25$, $k_{22} = 15$, $m\_i = 1$, and for the twisting controller are $c_{11} = 275$, $c_{12} = 495$, $c_{21} = 770$, $c_{22} = 715$ with $\delta = 0.03$. The simulation in Matlab/Simulink was performed using the sampling time of 1×10^{-4} seconds and Runge-Kutta algorithm as the solver.

In Fig. 13 is depicted the final trajectory generated with the procedure detailed in Section 2. The red marks are the samples gathered from the Autominy simulator, and the smooth trajectory generated with the time intervals and the red marks is represented by the blue line. The equations that represent the motion at each time interval are shown in Table 3 (see Appendix A) and are depicted in generalized coordinates in Fig. 14.

In Fig. 15 are depicted the tracking errors in generalized coordinates. It can be observed that the FTC and the PTC controllers converge faster than the DFC. Also, the proposed controller reaches the vicinity near the origin in $t < 1[s]$, as highlighted with the orange area; during this transitory stage the time-varying feedback control scheme is used, which afterwards is switched to the twisting controller.

This behavior is also present in $\tilde{y}$, where the FTC and the PTC present the fastest responses, followed by the DFC. The PTC controller exhibits a slight overshoot; however, it approaches the vicinity of the origin at $t < 1[s]$ within the orange region.

Finally, in $\tilde{\phi}$ the *PTC* controller obtains the fastest convergence compared with the DFC and FTC controllers; however, it also exhibits the highest level of overshoot. Furthermore, in $10 \leq t \leq 14.15$ the proposed controller achieves the lowest overshoots compared with the FTC and DFC controllers; this behavior can be attributed to the alterations in the reference signal for $\phi(t)$ as depicted in Fig.14 during this specific period.

Furthermore, in Fig.16 are depicted the control signals generated by the controllers in comparison to the desired control inputs $v_1(t)$ and $v_2(t)$, which are represented by the black-*colored* dashed lines. In $v_1(t)$, it is evident that the FTC controller exhibits the highest overshoot when compared to the PTC and DFC controllers; this behavior is associated with the fastest convergence of the FTC's tracking $\tilde{x}$ in Fig. 15, which is attributed to the presence of the SMC in the x coordinate.

However, the PTC achieves the fastest response, which implies the fastest tracking response. For $v_2(t)$, the PTC controller generates a large control signal before the commutation of the twisting controller, then it generates a noisy control signal, which is directly related to the tracking error of $\phi(t)$ in Fig. 15. Moreover, it is evident that the PTC controller transitions to the twisting control scheme in a time period less than 1 second.

As observed in the signal, the PTC controller generates the effect of chattering due to the presence of the twisting controller. In addition, all the control signals remain with similar behaviors after the transient stage.

Figure 17 depicts the tracking errors for case **C2** in the x, y, θ and ϕ coordinates. It can be observed that $\tilde{x}$ behaves similarly, with the FTC and PTC controllers achieving the fastest response, even when disturbances are considered. Regarding $\tilde{y}(t)$, it is evident that the FTC and PTC controllers also achieve the fastest response; nevertheless, the performance of the FTC and DFC controllers is diminished due to the generation of greater oscillations.

Furthermore, it is noteworthy that the time-varying stage of the PTC commutes to the twisting controller before the orange-colored area ends, where the twisting control signal is introduced, and thereafter the error signal keeps a neighborhood in zero.

For $\tilde{\theta}$, the fastest response is achieved by the PTC and FTC control schemes, but the PTC controller keep less oscillations. It can also be observed that the DFC controller generates a greater error at the end of the simulation.

For $\tilde{\phi}(t)$, the largest overshoot was generated by the PTC due to the time-varying stage and the

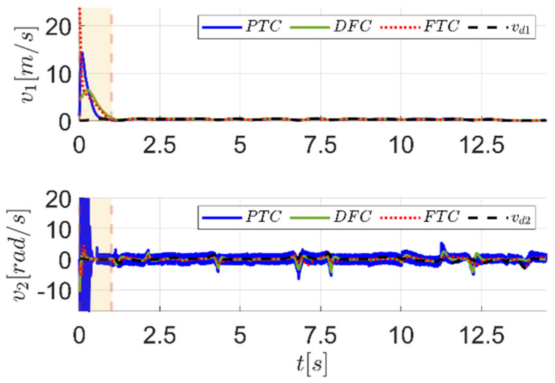

Fig. 16. Control signals $v_1(t)$ and $v_2(t)$ generated for case **C1**

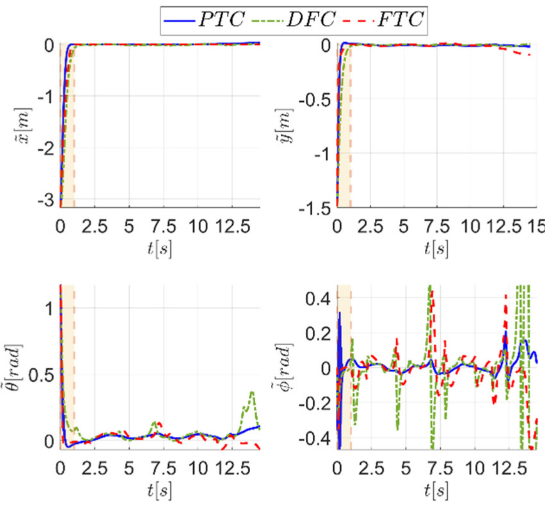

Fig. 17. Tracking errors in $\tilde{x}, \tilde{y}, \tilde{\theta}$and $\tilde{\phi}$ for case **C2**

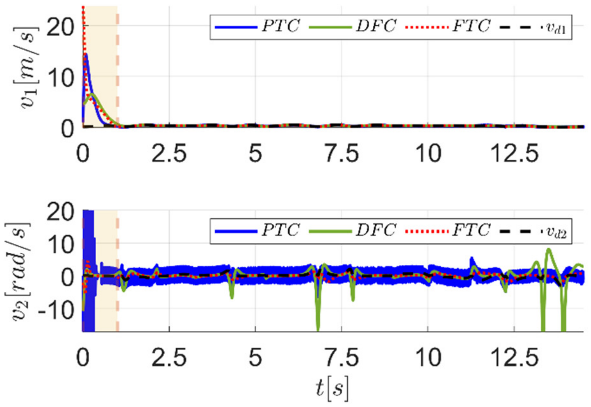

Fig. 18. Control signals $v_1(t)$ and $v_2(t)$ generated for case **C2**

Table 1. Performance indexes IAE, ITSE and ISV for the WMR for case **C1**

Controller	IAE				ISV
	$\tilde{x}$	$\tilde{y}$	$\tilde{\theta}$	$\tilde{\phi}$	
PTC	**0.653**	**0.168**	**0.21**	**0.245**	149.2
DFC	1.415	0.392	0.314	0.299	**28.44**
FTC	0.939	0.26	0.522	1.18	45.03
	ITSE				
	$\tilde{x}$	$\tilde{y}$	$\tilde{\theta}$	$\tilde{\phi}$	
PTC	**0.119**	**0.006**	**0.008**	**0.148**	
DFC	0.29	0.038	0.019	0.111	
FTC	0.264	0.033	0.207	3.242	

Table 2. Performance indexes IAE, ITSE and ISV for the WMR for case **C2**

Controller	IAE				ISV
	$\tilde{x}$	$\tilde{y}$	$\tilde{\theta}$	$\tilde{\phi}$	
PTC	**0.697**	**0.268**	0.62	**0.455**	190.1
DFC	1.373	0.447	1.01	2.03	169.5
FTC	0.936	0.371	**0.544**	1.147	**44.47**
	ITSE				
	$\tilde{x}$	$\tilde{y}$	$\tilde{\theta}$	$\tilde{\phi}$	
PTC	**0.133**	**0.017**	0.253	**0.312**	
DFC	0.49	0.04	1.35	16.26	
FTC	0.263	0.171	**0.159**	2.59	

control signal $v_2(t)$, which is observed at the beginning of the orange-colored area.

The PTC maintains the lowest tracking errors in this coordinate with the lowest oscillations even in the presence of disturbances.

Conversely, the DFC and FTC controllers present the large oscillations during the trajectory tracking. The tracking error $\tilde{\phi}(t)$ is associated with the control signal $v_2(t)$ depicted in Fig. 18, where it is corroborated that the DFC controller generated large control signals. On the other hand, the PTC controller generated the largest overshoot while the time-varying gains were active, i.e., in the orange-colored area at $t < 1[s]$; afterwards, the proposed scheme presents the effect of chattering due to the properties of the twisting controller and the SMC, however, its magnitude is small.

For $v_1(t)$, the PTC controller achieved the fastest response with a slight overshoot compared with the FTC controller. The DFC controller generated the lowest overshot and achieved the lowest response. In addition, it is clear that the proposed PTC controller achieves the fastest response and keeps the lowest oscillations in comparison with the DFC and FTC controllers.

Hence, the aforementioned findings illustrate that the PTC controller achieves the fastest response and the lowest oscillations, despite the existence of disturbances. Therefore, it can be inferred that the PTC controller excels over the DFC and FTC controllers.

In addition, a quantitative analysis was conducted to supplement the qualitative analysis. Tables 1 and 2 expose the IAE (Integral of Absolute Error) and ITSE (Integral of Time-weighted Squared Error) for evaluating the tracking error of the controllers, while the ISV (Integral of Squared Control Signal) is used to quantify the control signals [20]. Table 1 displays the quantitative tracking errors and ISV for case **C1** while Table 2 showcases the performance indexes for case **C2**.

Regarding case **C1**, it is evident that the PTC controller achieves the lowest tracking errors for each coordinate, except for $\phi(t)$, where the FTC controller achieves the lowest tracking errors in terms of ITSE. These quantities indicate that the PTC controller outstands with the best performance by achieving the lowest tracking errors with respect to the DFC and FTC controllers.

In addition, for case **C2**, the FTC controller demonstrated the smallest tracking error in $\theta(t)$, while the PTC controller exhibited the smallest tracking errors in the remaining coordinates, thus demonstrating its superior performance in comparison with the other controllers.

Based on the preceding qualitative and quantitative analysis, we can deduce that the PTC controller demonstrates superior performance in both disturbed and undisturbed scenarios by achieving the fastest response and lowest oscillations with respect to the DFC and PTC controllers. Moreover, the proposed control strategy achieved the lowest tracking errors with ITSE and IAE in general, thus outstanding its performance. Furthermore, we have observed that the trajectory generation algorithm is an effective solution for the task of generating trajectories.

This is because it successfully fulfills the non-holonomic constraints of WMRs. The simulations conducted with the controllers have demonstrated the algorithm's ability to track trajectories in real-world scenarios, even in the presence of disturbances.

Table 3. Polynomials for trajectory reference

Time interval	Reference trajectories
$0 \leq t < 1.13$	$x_d(t) = 0.0776t^7 + 0.4971t^6 - 1.7644t^5 + 1.3442t^4 + 0.1t + 1.668$ $y_d(t) = 0.0074t^7 + 0.0471t^6 - 0.167t^5 + 0.1274t^4 + 9e^{-3}t + 0.4869$
$1.13 \leq t < 2.14$	$x_d(t) = 0.004t^7 + 0.88t^6 - 8.37t^5 + 31.73t^4 - 62.0045t^3 + 66.23t^2 - 36.73t + 9.68$ $y_d(t) = 5e^{-4}t^7 + 752e^{-4}t^6 - 707.7e^{-3}t^5 + 2.638t^4 - 5.069t^3 + 5.328t^2 - 2.909t + 1.1545$
$2.14 \leq t < 3.23$	$x_d(t) = -0.01t^6 + 0.1552t^5 - 0.97t^4 + 3.187t^3 - 5.806t^2 + 5.92t - 1.2013$ $y_d(t) = -15.8e^{-3}t^6 + 246e^{-3}t^5 - 1.569t^4 + 5.262t^3 - 9.797t^2 + 9.625t - 3.38$
$3.21 \leq t < 4.32$	$x_d(t) = 2e^{-4}t^7 + 0.175t^6 - 3.68t^5 + 31.51t^4 - 141.87t^3 + 353.27t^2 - 464.53t + 251.09$ $y_d(t) = -15.2e^{-3}t^6 + 0.327t^5 - 2.91t^4 + 13.672t^3 - 35.83t^2 + 49.682t - 27.951$
$4.32 \leq t < 5.31$	$x_d(t) = 0.5t^6 + 13.8t^5 + 156.6t^4 - 942.8t^3 + 3172.7t^2 - 5659.4t + 4181.8$ $y_d(t) = 36e^{-4}t^6 - 0.1t^5 + 1.161t^4 - 7.117t^3 + 24.431t^2 - 44.536t + 34.212$
$5.31 \leq t < 6.81$	$x_d(t) = 0.1t^6 - 2.9t^5 + 41.1t^4 - 308.7t^3 + 1294.2t^2 - 2870.9t + 2635.1$ $y_d(t) = -4e^{-4}t^6 + 12.2e^{-3}t^5 - 0.1618t^4 + 1.1245t^3 - 4.292t^2 + 8.483t - 6.1832$
$6.81 \leq t < 7.83$	$x_d(t) = -19t^5 + 324t^4 - 2994t^3 + 15515t^2 - 42699t + 48755$ $y_d(t) = 19t^5 + 324t^4 - 2994t^3 + 15515t^2 - 42699t + 48755$
$7.83 \leq t < 9$	$x_d(t) = -8t^5 + 152t^4 - 1638t^3 + 9883t^2 - 31714t + 42282$ $y_d(t) = -0.2t^5 + 4.4t^4 - 48.1t^3 + 292.1t^2 - 944.8t + 1270.5$
$9 \leq t < 10.03$	$x_d(t) = -3t^5 + 57t^4 - 672t^3 + 4435t^2 - 15499t + 22397$ $y_d(t) = 0.2t^5 - 4.1t^4 + 47.1t^3 - 296.7t^2 + 980.6t - 1321$
$10.03 \leq t < 11.25$	$x_d(t) = -0.8t^5 + 19.1t^4 - 242.5t^3 + 1712.4t^2 - 6342.8t + 9588.1$ $y_d(t) = -2t^5 + 64t^4 - 896t^3 + 7053t^2 - 29575t + 51615$
$11.25 \leq t < 12.26$	$x_d(t) = -80t^4 + 1460t^3 - 14040t^2 + 71590t - 150960$ $y_d(t) = -10t^5 + 210t^4 - 3150t^3 + 26780t^2 - 121230t + 228190$
$12.26 \leq t < 13.34$	$x_d(t) = 3t^4 + 79t^3 - 2077t^2 + 17320t - 51070$ $y_d(t) = -2t^5 + 64t^4 - 915t^3 + 7138t^2 - 28147t + 42243$
$13.34 \leq t < 14.51$	$x_d(t) = t^5 - 21t^4 + 344t^3 - 3102t^2 + 14569t - 27559$ $y_d(t) = x_d(t) = t^5 - 21t^4 + 344t^3 - 3102t^2 + 14569t - 27559 + 18543t - 35323$

6 Trajectory Generation

The trajectory generation algorithm obtained specific points from the rail detections, which were subsequently analyzed to calculate two polynomials in the x, y plane. In order to achieve a feasible trajectory for a Car-Like robot [17], it is necessary for these polynomials to be both smooth and continuous. The procedure for designing the equations is described in [37], resulting in the equation's trims presented in Table 3.

7 Conclusions

The manuscript introduced an innovative approach for generating trajectories in vehicular systems by utilizing an on-board camera, computer vision techniques, and intelligent algorithms.

The proposed methodology was evaluated by employing three controllers that successfully achieved the trajectory tracking task. In addition, a prescribed-time controller was introduced for the purpose of trajectory tracking tasks. The performance of this controller was evaluated by comparing it to two controllers previously discussed in the literature.

Employing the simulation capabilities of MATLAB/SIMULINK, the prescribed-time controller demonstrated be superior, showcasing its resilience and adaptability in maintaining the desired trajectory, irrespective of the presence of disturbances.

The proposed methodology demonstrated its key feature of adjusting the convergence rate and its ability to withstand disturbances. Therefore, this study introduced a novel approach that integrates two approaches to tackle significant challenges in vehicular systems: the prescribed-time controller and the trajectory generation algorithm.

Future research ought to emphasize on enhancing the prescribed time controller by means of reducing or modifying the control structure to mitigate chattering and minimize the overshoot of the control signal of $v_2(t)$.

Furthermore, the trajectory generation algorithm can be enhanced by utilizing optimization algorithms to enhance its desired control signals in relation to generalized coordinates.

Acknowledgments

This work was supported in part by Instituto Politécnico Nacional under Grant SIP 2024-0018 and the program “Investigadoras e Investigadores por México”, Cátedras CONAHCYT, Project No. 537.

References

1. **Zhang, J., Li, S., Meng, H., Li, Z., Sun, Z. (2023).** Variable gain based composite trajectory tracking control for 4-wheel skid-steering mobile robots with unknown disturbances. Control Engineering Practice, Vol. 132, pp. 105428. DOI: 10.1016/j.coneng prac.2022.105428.
2. **Li, L., Cao, W., Yang, H., Geng, Q. (2022).** Trajectory tracking control for a wheel mobile robot on rough and uneven ground. Mechatronics, Vol. 83, pp. 102741. DOI: 10.10 16/j.mechatronics.2022.102741.
3. **Gao, H., Chen, C., Ding, L., Li, W., Yu, H., Xia, K., Liu, Z. (2017).** Tracking control of WMRS on loose soil based on mixed H2/H∞ control with longitudinal slip ratio estimation. Acta Astronautica, Vol. 140, pp. 49–58. DOI: 10.1016/j.actaastro.2017.07.037.
4. **Olayode, I. O., Du, B., Severino, A., Campisi, T., Alex, F. J. (2023).** Systematic literature review on the applications, impacts, and public perceptions of autonomous vehicles in road transportation system. Journal of Traffic and Transportation Engineering (English Edition), Vol. 10, No. 6, pp. 1037–1060. DOI: 10.1016/ j.jtte.2023.07.006.
5. **Wang, C. (2011).** A novel variable structure theory applied in design for wheeled mobile robots. Artificial Life and Robotics, Vol. 16, No. 3, pp. 378–382. DOI: 10.1007/s10015-011-0955-3.
6. **Yan, K., Ma, B. (2023).** Global posture stabilization for the kinematic model of a rear-axle driven car-like mobile robot considering obstacle avoidance. IEEE Robotics and Automation Letters, Vol. 8, No. 9, pp. 5568–5575. DOI: 10.1109/lra.2023.3296351.
7. **Kocsis, M., Schultz, A., Zöllner, R., Mogan, G. L. (2016).** A method for transforming electric vehicles to become autonomous vehicles. CONAT 2016 International Congress of Automotive and Transport Engineering, pp. 752–761. DOI: 10.1007/978-3-319-45447- 4_83.
8. **Han, X., Zhao, X., Xu, X., Mei, C., Xing, W., Wang, X. (2024).** Trajectory tracking control for underactuated autonomous vehicles via adaptive dynamic programming. Journal of the Franklin Institute, Vol. 361, No. 1, pp. 474–488. DOI: 10.1016/j.jfranklin.2023.12.003.
9. **Cruz, V. D., Rodriguez, J. A., Aguilar, L. T., Colorado, R. M. (2023).** Trajectory tracking control of wheeled mobile robots using neural networks and feedback control techniques. Studies in Computational Intelligence, pp. 381–393. DOI: 10.1007/978-3-031-28999-6_24.
10. **Hart, P., Rychly, L., Knoll, A. (2019).** Lane-merging using policy-based reinforcement learning and post-optimization. IEEE Intelligent Transportation Systems

Conference, pp. 3176–3181. DOI: 10.1109/ itsc.2019.8917002.

11. **Bellusci, M., Cudrano, P., Mentasti, S., Cortelazzo, R. E. F., Matteucci, M. (2024).** Semantic interpretation of raw survey vehicle sensory data for lane-level hd map generation. Robotics and Autonomous Systems, Vol. 172, pp. 104513. DOI: 10.1016/j.robot.2023. 104513.

12. **Han, Z., Gu, J., Feng, Y. (2023).** Blind lane detection and following for assistive navigation of vision impaired people. International Conference on Advanced Robotics and Mechatronics, pp. 721–726. DOI: 10.1109/ icarm58088.2023.10218843.

13. **Chen, Y., Wong, P. K., Yang, Z. (2021).** A new adaptive region of interest extraction method for two-lane detection. International Journal of Automotive Technology, Vol. 22, No. 6, pp. 1631–1649. DOI: 10.1007/s12239-021- 0141- 0.

14. **Haixia, L., Xizhou, L. (2021).** Flexible lane detection using CNNs. International Conference on Computer Technology and Media Convergence Design. DOI: 10.1109/ ctmcd53128.2021.00057.

15. **Khan, M. A., Kee, S., Sikder, N., Mamun, M. A. A., Zohora, F. T., Hasan, M. T., Bairagi, A. K., Nahid, A. (2021).** A vision-based lane detection approach for autonomous vehicles using a convolutional neural network architecture. Joint 10th International Conference on Informatics, Electronics and Vision, pp. 1-10. DOI: 10.1109/ICIEVicIV PR52578.2021.9564229.

16. **Peregrina-Ochoa, S. A. (2019).** Sistema de navegación autónomo para un vehículo a escala mediante aprendizaje automático y visión por computadora. Tesis de Maestría en Ciencias en Ingeniería de Cómputo, Centro de Investigación en Computación, Instituto Politécnico Nacional.

17. **Neven, D., Brabandere, B. D., Georgoulis, S., Proesmans, M., Gool, L. V. (2018).** Towards end-to-end lane detection: an instance segmentation approach. IEEE Intelligent Vehicles Symposium, pp. 286–291. DOI: 10.1109/IVS.2018.8500547.

18. **Luca, A. D., Oriolo, G., Samson, C. (1998).** Feedback control of a nonholonomic car-like robot. Robot Motion Planning and Control, pp. 171–253. DOI: 10.1007/bfb0036073.

19. **Liu, D., Tang, M., Fu, J. (2022).** Robust adaptive trajectory tracking for wheeled mobile robots based on gaussian process regression. Systems and Control Letters, Vol. 163, pp. 105210. DOI: 10.1016/j.sysconle.2022. 105210.

20. **Rosas-Vilchis, A. J. (2021).** Algoritmos de observación y control robustos para el vehículo autónomo. Tesis de Maestría en Ciencias en Sistemas Digitales, Centro de Investigación en Computación, Instituto Politécnico Nacional.

21. **Lu, Q., Chen, J., Wang, Q., Zhang, D., Sun, M., Su, C. (2022).** Practical fixed-time trajectory tracking control of constrained wheeled mobile robots with kinematic disturbances. ISA Transactions, Vol. 129, pp. 273–286. DOI: 10.1016/j.isatra.2021.12.039.

22. **Dixon, W. E., Dawson, D. M., Zergeroglu, E. (2000).** Tracking and regulation control of a mobile robot system with kinematic disturbances: a variable structure-like approach. Journal of Dynamic Systems, Measurement, and Control, Vol. 122, No. 4, pp. 616–623. DOI: 10.1115/1.1316795.

23. **Rodríguez-Arellano, J. A., Miranda-Colorado, R., Aguilar, L. T., Negrete-Villanueva, M. (2023).** Trajectory tracking nonlinear H^∞ controller for wheeled mobile robots with disturbances observer. ISA Transactions, Vol. 142, pp. 372–385. DOI: 10.1016/j.isatra.2023.07.037.

24. **Cui, M., Liu, H., Wang, X., Liu, W. (2023).** Adaptive control for simultaneous tracking and stabilization of wheeled mobile robot with uncertainties. Journal of Intelligent and Robotic Systems, Vol. 108, No. 3 DOI: 10.10 07/s10846-023-01908-0.

25. **Chang, S., Wang, Y., Zuo, Z., Yang, H., Luo, X. (2023).** Robust prescribed-time containment control for high-order uncertain multi-agent systems with extended state observer. Neurocomputing, Vol. 559, pp. 126782. DOI:10.16/j.neucom.2023.126782.

26. **Github (2004).** TuSimple Dataset. github.com/TuSimple/tusimple-benchmark.
27. **Vu, D., Ngo, B., Phan, H. (2022).** Hybridnets: end-to-end perception network. arXiv. DOI: 10.48550/ARXIV.2203.09035.
28. **GitHub (2022).** ONNX-hybridnets-multitaskroad- detection: Python scripts for performing road segmentation and car detection using the hybridnets multitask model in ONNX. github.com/ibaiGorordo/ONNX-HybridNets-Multitask-Road-Detection.
29. **GitHub (2020).** Pinto model repository for storing models that have been inter-converted between various models. github.com/PINTO 0309/PINTO_model_zoo
30. **Marutotamtama, J. C., Setyawan, I. (2021).** Physical distancing detection using YOLO v3 and bird's eye view transform. Proceedings of the 2nd International Conference on Innovative and Creative Information Technology, pp. 50–56. DOI: 10.1109/ICITech 50181.2021.9590157.
31. **Sihombing, D. P., Nugroho, H. A., Wibirama, S. (2015).** Perspective rectification in vehicle number plate recognition using 2d-2d transformation of planar homography. Proceedings of the International Conference on Science in Information Technology, pp. 237–240. DOI: 10.1109/ICSITech.2015. 7407810.
32. **Yoo, S. J. (2013).** Adaptive neural tracking and obstacle avoidance of uncertain mobile robots with unknown skidding and slipping. Information Sciences, Vol. 238, pp. 176–189. DOI: 10.1016/j.ins.2013.03.013.
33. **Kairuz, R. I. V., Orlov, Y., Aguilar, L. T. (2021).** Prescribed-time stabilization of controllable planar systems using switched state feedback. IEEE Control Systems Letters, Vol. 5, No. 6, pp. 2048–2053. DOI: 10.1109/ LCSYS.2020.3046682.
34. **Orlov, Y. (2020).** Nonsmooth Lyapunov analysis in finite and infinite dimensions. Springer Cham. DOI: 10.1007/978-3-030-37625-3.
35. **Biagiotti, L., Melchiorri, C. (2009).** Trajectory planning for automatic machines and robots. Springer Berlin.

Article received on 22/01/2023; accepted on 24/04/2024.
**Corresponding author is Luis T. Aguilar.*