World Electric Vehicle Journal

MDPI

Article

Physics-Informed Neural Network-Based Nonlinear Model Predictive Control for Automated Guided Vehicle Trajectory Tracking

Yinping Li * and Li Liu

School of Mechanical Engineering, Beijing University of Science and Technology, Beijing 100083, China; liliu@ustb.edu.cn

* Correspondence: liyinping0001@163.com

Abstract: This paper proposes a nonlinear Model Predictive Control (MPC) method based on Physics-Informed Neural Networks (PINNs), aimed at enhancing the trajectory tracking performance of Automated Guided Vehicles (AGVs) in complex dynamic environments. Traditional physical models often face the challenges of computational inefficiency and insufficient control precision when dealing with complex dynamic systems. However, by integrating physical laws directly into the training process of neural networks, PINNs can effectively learn and capture the kinematic characteristics of vehicles, replacing traditional nonlinear ordinary differential equation models and thus significantly enhancing computational efficiency and control performance. During the model-training phase, this study further incorporates the Theory of Functional Connections (TFC) and adaptive loss balancing strategies to efficiently solve ODE problems without relying on numerical integration and optimize the control strategy. This combined approach not only reduces computational complexity, but also improves the robustness and precision of the control strategy in varying environments. Numerical simulations demonstrate that this method offers significant advantages in AGV trajectory-tracking tasks, manifested in higher computational efficiency and precise control performance. The proposal of the PINN-MPC method provides new theoretical support and innovative methods for real-time complex system control, with important research and application potential, and is expected to play a key role in future intelligent control systems.

Keywords: physics-informed neural networks; model predictive control; machine learning; ordinary differential equations

check for updates

Citation: Li, Y.; Liu, L. Physics-Informed Neural Network-Based Nonlinear Model Predictive Control for Automated Guided Vehicle Trajectory Tracking. *World Electr. Veh. J.* **2024**, *15*, 460. https://doi.org/10.3390/wevj15100460

Academic Editor: Michael Fowler

Received: 7 September 2024
Revised: 4 October 2024
Accepted: 8 October 2024
Published: 10 October 2024

Copyright: © 2024 by the authors. Published by MDPI on behalf of the World Electric Vehicle Association. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

In recent years, Automated Guided Vehicles (AGVs) have become a key component of modern industrial automation as a core technology in intelligent logistics and manufacturing systems. The trajectory planning and control technologies of AGVs are particularly important in the process of achieving efficient and precise operations. However, with the increasing complexity of application scenarios, traditional control methods based on physical models face severe challenges [1]. These methods usually rely on simplified dynamics and kinematics models to describe the behavior of AGVs, but in complex and variable environments, such simplified models often fail to accurately capture the nonlinear characteristics of the system, leading to insufficient control precision and robustness. Additionally, the complex dynamic characteristics of AGV systems and the diversity of operational environments further increase the difficulty of control system design, making it difficult for traditional physical models to effectively meet control requirements under complex scenarios [2,3]. This limitation has prompted researchers to seek new methods to improve the performance of control systems to better meet the demands of complex application environments. Caban and colleagues conducted a detailed analysis of the application of autonomous trucks in the logistics sector, with a particular emphasis on the

critical role of artificial intelligence. The article first introduced the latest advancements in automated transportation technology, focusing specifically on the central role of Automated Guided Vehicles (AGVs) in logistics systems. The study explored the efficiency of deep learning algorithms and neural networks in data processing and decision support. It also covered specialized mathematical models for autonomous driving systems, which have been validated through simulation tests to demonstrate their adaptability and optimization performance under complex conditions. Furthermore, the paper evaluated the economic and safety benefits of autonomous trucks and projected their broad potential for future applications in the logistics industry. Additionally, it discussed the potential impact of AI in the wider transportation sector [4].

To address the limitations of traditional control methods in complex environments, researchers have gradually shifted toward integrating modern deep learning with advanced control theories to enhance the control precision and reliability of AGV systems in complex environments. Among these, Model Predictive Control (MPC), as a cutting-edge method that optimizes future control inputs to achieve optimal system control, has been widely applied in the field of vehicle engineering. Although linear MPC is commonly used in scenarios requiring high real-time control due to its low computational complexity and strong real-time performance, its effectiveness in handling complex nonlinear dynamic systems is relatively limited [5]. In contrast, nonlinear MPC can utilize nonlinear models to more accurately capture the dynamic characteristics of the system, thereby exhibiting stronger adaptability in dealing with complex working conditions. Despite the higher computational complexity of nonlinear MPC, with the enhancement of computing capabilities and advancements in optimization algorithms, its application is increasingly widespread in demanding applications, such as autonomous driving and vehicle dynamics control, demonstrating significant potential [6–8].

Simultaneously, data-driven Model Predictive Control (MPC) methods have gradually become an important means in complex system control due to their ability to leverage large volumes of historical data for model training, thus reducing reliance on traditional physical models. This approach is particularly suited to systems with high dimensions and significant nonlinear characteristics. However, data-driven MPC still faces many challenges in big data processing, model training, and algorithm design, especially in terms of model generalization and robustness, which need further enhancement [9–11]. Overall, the application of MPC technology in the field of vehicle engineering is exhibiting a trend toward diversification and deepening: linear MPC has made significant progress in enhancing robustness, nonlinear MPC continues to improve in handling complexity and nonlinear dynamic systems, and data-driven MPC displays unique innovative potential in dealing with high-dimensional complex systems. These developments indicate that future control systems will achieve greater breakthroughs in intelligence and autonomous decision-making capabilities, providing more effective solutions for increasingly complex engineering applications [12,13].

In the exploration of MPC technology, researchers have proposed an innovative method that combines Physics-Informed Neural Networks (PINNs) with Model Predictive Control (MPC) to further enhance the control performance of complex systems. PINNs integrate physical laws directly into the training process of neural networks, allowing the model to strictly adhere to physical constraints while learning complex dynamic systems, thereby significantly improving the model's accuracy and generalization capability. Especially in the field of vehicle engineering, the application of PINNs has gradually extended to key technological areas, such as dynamic simulation, path planning, and fault diagnosis. By deeply integrating physical laws with data-driven models, PINNs not only improve the system's predictive accuracy, but also enhance the robustness of autonomous driving systems in handling complex road conditions and environmental uncertainties. Reissa et al.'s research has laid the theoretical foundation for solving complex nonlinear partial differential equations using PINNs, advancing the technology's development in the field of scientific computing. With the rapid progress of deep learning technology, PINNs have

shown significant advantages in various engineering fields, such as fluid dynamics, electromagnetic field simulation, and vehicle dynamics [14]. Karniadakis et al. systematically reviewed the latest advances in PINNs in the scientific and engineering fields, particularly highlighting their enormous potential in solving high-dimensional nonlinear problems [15]. Sun et al. further explored the application of PINNs in fluid dynamics, and proposed new methods for optimizing model training and handling complex boundary conditions. These advances have promoted the widespread application of PINNs in multiple domains [16]. Overall, the application of PINNs in vehicle engineering has gradually deepened, bringing higher accuracy and reliability to dynamic simulation, path planning, and fault diagnosis.

The Theory of Functional Connections (TFC) serves as a cutting-edge mathematical tool that simplifies the solving process of partial differential equations and optimization problems by embedding constraints directly into the solution expression, significantly reducing computational complexity. TFC has shown broad application prospects in fields such as physical modeling, engineering optimization, and control system design, particularly in vehicle engineering, where the introduction of TFC has greatly enhanced the precision and response speed of path planning, dynamic simulation, and intelligent control. TFC has been proven to possess strong solving capabilities in complex physical systems, with its application scope continuously expanding, especially showing unique advantages in solving complex dynamic problems in vehicle engineering. Researchers are actively exploring the integration of TFC with machine learning and neural network technologies to further enhance its adaptability in high-dimensional nonlinear systems. This interdisciplinary combination not only helps to enhance the application efficiency of TFC in intelligent control and autonomous driving, but also provides a new pathway for developing more efficient and intelligent control algorithms [17–19]. With the deepening of research, TFC is expected to play an increasingly critical role in future intelligent control systems, providing more advanced solutions to meet the challenges of highly complex engineering tasks.

In the deep machine learning training process, the reasonable allocation of loss function weights is crucial for the convergence and overall performance of the model. Especially in multi-task learning, the adaptive weight-adjustment strategies that dynamically balance the weights of various tasks significantly enhance the model's performance in dealing with class imbalance and multi-objective optimization. Such techniques have practical application value, particularly in vehicle engineering fields such as autonomous driving and fault diagnosis, providing important support for the optimization of complex system control strategies. Cao et al. proposed an adaptive loss weight adjustment method that has shown significant improvement effects in multi-task learning for autonomous driving. By dynamically adjusting the weights of various tasks, the model can still maintain efficient learning ability when facing diverse tasks [20]. Kendall et al. further optimized the handling of task imbalance problems by introducing uncertainty estimation, enhancing the coordination ability of the model between different tasks [21]. Additionally, Wang et al.'s research strategy performed excellently in handling imbalanced data, further enhancing the model's generalization ability and robustness [22]. The combination of these methods has propelled the application of deep learning technology in complex system control, providing new theoretical support and practical pathways for the development of intelligent control systems. The theoretical research and practical application results of adaptive weight adjustment strategies show that it has important academic and engineering significance in improving model performance and adaptability.

In order to address the limitations of traditional control methods in complex nonlinear systems, researchers have actively explored an innovative approach that combines Physics-Informed Neural Networks (PINNs) with Model Predictive Control (MPC) in recent years [23]. PINNs embed physical laws directly into the deep learning framework, ensuring that the model maintains physical consistency during the learning process while enhancing its generalization ability in complex dynamic environments. MPC optimizes control inputs over a finite time horizon to ensure that the system can achieve the expected

dynamic objectives. The application of PINNs integrated with MPC for trajectory control of Automated Guided Vehicles (AGVs) not only effectively addresses the challenges posed by nonlinear systems, but also provides higher precision and stronger robustness control strategies in complex environments. This fusion method provides a new solution for AGV to achieve efficient and accurate trajectory control in dynamic and highly uncertain scenarios [24,25], and lays a solid foundation for the development of future intelligent control systems.

This study delves into the integration of Physics-Informed Neural Networks (PINNs) with Model Predictive Control (MPC), which significantly improves the accuracy and robustness of control systems by embedding physical laws into deep learning processes. Based on this, this paper proposes an innovative PINN-MPC method, which is particularly suitable for complex systems, such as autonomous driving, utilizing PINNs to replace traditional physical models and thereby improving control precision without relying on explicit physical modeling. Additionally, this research focuses on the application of the Theory of Functional Connections (TFC) and optimization algorithms in vehicle path planning, demonstrating their significant advantages in simplifying the solution process and enhancing computational efficiency. At the same time, this study also explores the potential of multi-task learning and imbalanced data processing techniques to improve model performance, especially in complex task scenarios. Through these studies, this paper provides an important theoretical basis and practical guidance for the field of intelligent control and optimization, promoting the further development of complex system control strategies.

2. AGV Model and Nonlinear MPC

Figure 1 shows the kinematic model of the Automated Guided Vehicle (AGV), where OXY denotes a stationary Cartesian coordinate system on the ground [26]. Within this coordinate system, the kinematic behavior of the AGV can be described by a set of kinematic Equations (1). These equations precisely represent the motion trajectory of AGV in the plane, covering changes in its position, velocity, and direction. This model not only allows for an in-depth analysis of the AGV's motion characteristics, but also provides solid theoretical support for the design of control strategies. Through the study of kinematic equations, the study of the kinematic equations not only reveals the fundamental motion laws of the AGV, but also provides a crucial foundation for the further optimization of control algorithms.

$$\begin{aligned} \dot{x}(t) &= v(t)\cos(\theta(t)) \\ \dot{y}(t) &= v(t)\sin(\theta(t)) \\ \dot{\theta}(t) &= v(t)\frac{\tan(\varphi(t))}{l} \end{aligned}, \tag{1}$$

In this model, x and y respectively denote the lateral and longitudinal position coordinates of the vehicle in the Cartesian coordinate system OXY, θ represents the vehicle's yaw angle, φ denotes the steering angle of the front wheel, v is the vehicle's velocity, and l is the wheelbase between the front and rear wheels. These key variables allow for a precise description of the vehicle's motion state and directional changes within the plane [27]. These parameters collectively determine the AGV's motion trajectory and dynamic characteristics, providing an important data foundation for the derivation of kinematic equations and the formulation of control strategies.

The nonlinear ordinary differential equations in Equation (1) can be further organized and expressed in the following standard form (2). Through this general form of expression, the dynamic behavior and inherent nonlinear characteristics of the system can be more clearly revealed. This form of expression not only facilitates deeper theoretical analysis, but also provides a more convenient mathematical framework for numerical solutions and optimization of control strategies.

$$\dot{\boldsymbol{z}}(t) = \boldsymbol{F}(\boldsymbol{z}(t), \boldsymbol{u}(t)), \tag{2}$$

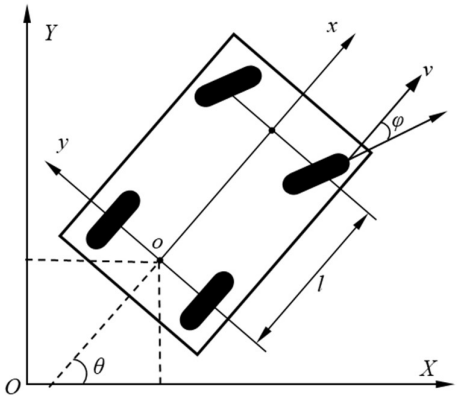


Figure 1. Configuration of the AGV platform.

In this context, $z(t) = [x(t), y(t), \theta(t)]^{\mathrm{T}}$ represents the vehicle's state vector and $\boldsymbol{u}(t) = [v(t), \varphi(t)]^{\mathrm{T}}$ represents the vehicle's control vector. In practical applications, to ensure the effectiveness of digital control, it is usually necessary to discretize the continuous-time control systems [28]. After discretization, the form of Equation (2) can be re-expressed as Formula (3):

$$z_{k+1} = \boldsymbol{f}(\tau, z_k, \boldsymbol{u}_k)\ k = 0, 1, 2, \ldots, \tag{3}$$

where $z_k = z(t_k)$ and $u_k = u(t_k)$ respectively represent the state vector and control vector at time $t_k = t_0 + k\tau$, and τ is the fixed sampling time interval. It is further assumed that the control input employs a zero-order hold strategy, which means that the control input remains constant and unchanged during each sampling period. This assumption simplifies the analysis and design of the control system, allowing the controller to perform computations at each sampling interval, thereby ensuring the system's stability and responsiveness in the discrete-time domain [29].

This article discusses the application of nonlinear Model Predictive Control (MPC) in trajectory tracking tasks. It should be clarified that MPC is not a specific control strategy, but rather a class of methods that compute optimal control sequences by optimizing a cost function of a constrained dynamic system within a finite time horizon. For the control systems as described by Equation (3), when the current state z_k is measured at time t_k, the core idea of MPC is to determine a set of optimal control inputs $u_k, \ldots, u_{k+N-1}$ to minimize the deviation between the predicted future states $z_{k+1}, \ldots, z_{k+N}$ and a given reference trajectory $z_{k+1}^{ref}, \ldots, z_{k+N}^{ref}$ within the prediction range. The discrete nonlinear MPC formulations in this article are as follows ((4) and (5)):

$$\min_{\Delta \boldsymbol{u}_k, \ldots, \Delta \boldsymbol{u}_{k+N-1}} J = \sum_{i=1}^{N} \left(\boldsymbol{z}_{k+i} - \boldsymbol{z}_{k+i}^{ref}\right)^{\mathrm{T}} \boldsymbol{Q} \left(\boldsymbol{z}_{k+i} - \boldsymbol{z}_{k+i}^{ref}\right) + \sum_{j=0}^{N-1} \Delta \boldsymbol{u}_{k+j}^{\mathrm{T}} \boldsymbol{R} \Delta \boldsymbol{u}_{k+j}, \tag{4}$$

$$s.t. \quad \left.\begin{array}{l} \boldsymbol{z}_{k+i+1} = \boldsymbol{f}(\tau, \boldsymbol{z}_{k+i}, \boldsymbol{u}_{k+i}) \\ \boldsymbol{u}_{k+i} = \boldsymbol{u}_{k-1} + \sum\limits_{j=0}^{i} \Delta \boldsymbol{u}_{k+j} \\ \boldsymbol{u}_{\min} \le \boldsymbol{u}_{k+i} \le \boldsymbol{u}_{\max} \\ \Delta \boldsymbol{u}_{\min} \le \Delta \boldsymbol{u}_{k+i} \le \Delta \boldsymbol{u}_{\max} \\ \boldsymbol{z}_{\min} \le \boldsymbol{z}_{k+i+1} \le \boldsymbol{z}_{\max} \end{array}\right\} \forall i = 0, \ldots, N-1, \tag{5}$$

In this process, diagonal matrices Q and R serve as weights to convert the tracking problem into a Nonlinear Programming (NLP) challenge at each time point k^{th}. The objective is to identify a sequence of control increments $\Delta u_k, \ldots, \Delta u_{k+N-1}$ that minimize the cost function, outlined in Equation (4), while complying with the constraints detailed in Equation (5). Once optimization is finalized, the initial control increment Δu_k is executed to generate the control input u_k, which is applied and sustained until the next point $(k+1)^{th}$.

This optimization and implementation cycle is methodically repeated at regular intervals throughout the process, persisting until the completion of the entire time series, ensuring consistent application of the control inputs.

In this study, to address the Nonlinear Programming (NLP) challenges inherent in nonlinear Model Predictive Control (MPC), the mature and efficient Sequential Quadratic Programming (SQP) method was employed [30]. The SQP method is widely utilized for its superiority in handling nonlinear constraint optimization problems. By decomposing the original nonlinear problem into a series of more easily solvable quadratic programming problems, it gradually approximates the global optimum and effectively addresses the complexity and nonlinear characteristics of MPC problems [31].

3. Physics-Informed Neural Networks

In this study, Physics-Informed Neural Networks (PINNs) are introduced as an alternative tool to traditional nonlinear ordinary differential equation (ODE) models for the modeling and state prediction of Automated Guided Vehicle (AGV) systems. Trained PINNs can directly predict system states, replacing the traditional nonlinear ODE models used for state prediction at time points in Nonlinear Model Predictive Control (MPC). Compared with traditional numerical integration methods, a significant advantage of PINNs is that they do not require heavy numerical integration operations, thereby substantially reducing the computational complexity of solving ODEs. This efficient modeling approach not only significantly enhances computational efficiency, but also makes real-time prediction in complex systems more feasible, providing strong support for the effectiveness and operability of nonlinear MPC in practical applications.

However, traditional Physics-Informed Neural Networks (PINNs) exhibit limitations in Model Predictive Control (MPC) tasks, mainly because they typically assume fixed inputs and conditions, making it difficult to cope with dynamically changing initial conditions and control inputs [32]. Specifically, in the conventional PINN framework, the network usually takes continuous time as the input and outputs the system state variables x, y, and θ, which limits its flexibility in handling variable conditions. In order for PINN to adapt to MPC tasks, it is necessary to explicitly include initial conditions and control inputs in the network's input layer to reflect the system's dynamic characteristics. Antonelo et al. [33] proposed an improved PINN framework that takes control actions and initial conditions as additional network inputs, effectively achieving long-term simulation of dynamic systems through a self-recycling mechanism. This method overcomes the limitations of traditional PINNs, making them more suitable for model predictive control tasks in complex dynamic systems.

Traditional Physics-Informed Neural Network (PINN) frameworks exhibit a significant limitation when solving Ordinary Differential Equations (ODEs), which is that they cannot strictly satisfy the initial conditions at the analytical level. Additionally, PINNs face two conflicting objectives during the training process: on the one hand, they need to accurately learn the solution of ODE in the time domain, and on the other hand, it also needs to satisfy the initial conditions simultaneously. When optimizing based on gradient methods, such objective conflicts may lead to gradient imbalance, which in turn affects the approximation accuracy of PINN to ODE solutions. To address this issue, this paper introduces a method based on the Theory of Functional Connections (TFC) [34], which directly satisfies initial conditions at the analytical level, successfully avoiding gradient imbalances caused by competitive objectives during training process. This improvement not only significantly enhances model training stability, but also effectively mitigates common accuracy issues in the traditional PINN framework. However, another challenge in PINN training is the sensitivity of the loss function's weightings, which significantly impacts training efficiency. To further optimize the training process, this paper uses the TFC method to remove the initial condition loss term from the overall loss function, retaining only the three loss terms related to the ODE residuals, corresponding to the three ODEs in the system (Equation (1)). To ensure effective and efficient training, the weights of these loss terms must be finely adjusted. For this purpose, this study adopts the adaptive loss balancing

technique proposed by Xiang et al. [35], which can dynamically adjust the weight of the loss term, thereby improving training efficiency and ensuring the accuracy of the model. Through this series of improvements, significant progress has been made in the stability and accuracy of PINN training in this study, providing a more reliable tool for solving complex dynamic systems.

Here, based on the research of Antonello et al. [33], in order to perform short-term simulations within a specific time interval $[0, \tau]$, a physics-informed neural network (PINN) was designed, incorporating control actions and initial conditions as additional inputs, as shown in Figure 2.

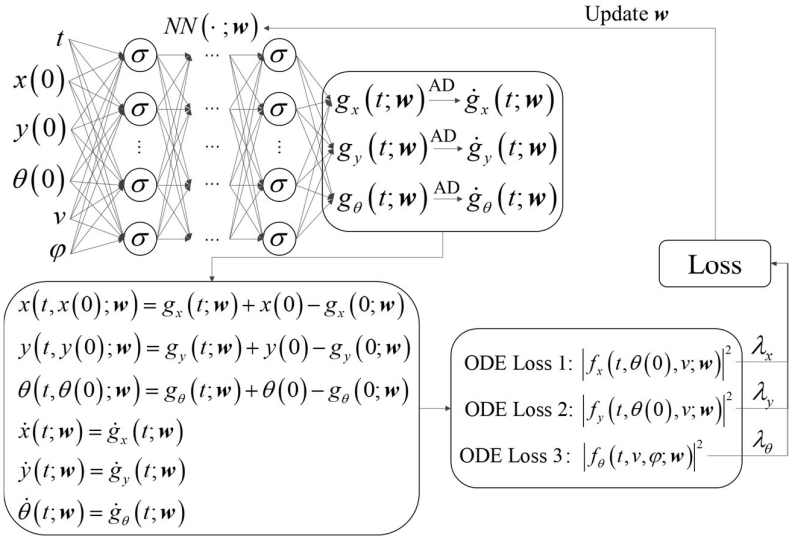


Figure 2. A schematic diagram of the LB-TFC-PINN framework.

Here, w represents the network's weights and σ denotes commonly used activation functions, such as sigmoid and tanh [36]. The inputs of the network include time t, initial states $x(0)$, $y(0)$, and $\theta(0)$, and control actions v and φ. Among them, the time range t is $[0, \tau]$, and the values of the initial states and control actions are determined by the system's constraints (see Equation (5)). According to the zero-order hold assumption, the control inputs v and φ remain constant within the time interval $[0, \tau]$. It is worth noting that the network output is not the system's state variables x, y, and θ, but rather the so-called free functions g_x, g_y, g_θ. The derivatives of these free functions with respect to time can be obtained through automatic differentiation (AD) techniques [37]. Based on the Theory of Functional Connections (TFC), the unknown solution of the system's ordinary differential equations (ODEs) (see Equation (1)) can be approximated by a constrained expression (6) containing these free functions, thereby achieving effective prediction of the system's behavior. This method not only improves the expression ability of the model, but also better meets the control requirements of complex systems.

$$\begin{cases} x(t) = g_x(t) + x(0) - g_x(0) \\ y(t) = g_y(t) + y(0) - g_y(0) \\ \theta(t) = g_\theta(t) + \theta(0) - g_\theta(0) \end{cases}, \tag{6}$$

It is worth noting that, regardless of the forms of the free functions g_x, g_y, g_θ, the solutions x, y, θ always satisfy the initial conditions, with their derivative expressions given by Equation (7) as follows:

$$\begin{cases} \dot{x}(t) = \dot{g}_x(t) \\ \dot{y}(t) = \dot{g}_y(t) \\ \dot{\theta}(t) = \dot{g}_\theta(t) \end{cases}, \tag{7}$$

In this study, neural networks are used as the free functions. By substituting Equations (6) and (7) into Equation (1), three Ordinary Differential Equation (ODE) residuals can be defined [38], denoted as f_x, f_y, and f_θ, as shown in Equation (8). These residuals are used to evaluate the performance of the neural network in approximating the system's dynamic behavior, thereby guiding the optimization process of the model.

$$\begin{aligned} f_x(t,\theta(0),v;\boldsymbol{w}) &= \dot{g}_x(t;\boldsymbol{w}) - v\cos(g_\theta(t;\boldsymbol{w}) + \theta(0) - g_\theta(0;\boldsymbol{w})) \\ f_y(t,\theta(0),v;\boldsymbol{w}) &= \dot{g}_y(t;\boldsymbol{w}) - v\sin(g_\theta(t;\boldsymbol{w}) + \theta(0) - g_\theta(0;\boldsymbol{w})) \\ f_\theta(t,v,\varphi;\boldsymbol{w}) &= \dot{g}_\theta(t;\boldsymbol{w}) - v\frac{\tan\varphi}{l} \end{aligned}, \tag{8}$$

Physics-Informed Neural Networks (PINNs) have the capability to be trained without labeled data because the laws of physics are embedded in the loss function in the form of Ordinary Differential Equations (ODEs). The output of the network, which represents the numerical solution of the ODE, is constrained by these physical laws during the training process to ensure that the solution satisfies the ODE, even if relying only on unlabeled sampled points as inputs. As shown in Figure 2, the network has six inputs, and the sampling points required for training are generated through uniform sampling in the six-dimensional space. These sampling points can be generated using the Sobol sequence or Latin hypercube sampling methods [39] to ensure comprehensive coverage of all regions of the input space. During training, the loss function consists of three main terms, each corresponding to the residuals of the ODEs, as shown in Equation (9) below, to ensure strict consistency between the network output and the physical laws. This method enables the PINN to effectively capture the system's dynamic behavior and accurately predict the system's states, even in the absence of labeled data. This physics-constrained training method significantly enhances the potential application of PINNs in modeling complex systems.

$$\begin{aligned} L_x(\boldsymbol{w}) &= \frac{1}{N_f}\sum_{i=1}^{N_f}\left|f_x(t_i,\theta_i(0),v_i;\boldsymbol{w})\right|^2 \\ L_y(\boldsymbol{w}) &= \frac{1}{N_f}\sum_{i=1}^{N_f}\left|f_y(t_i,\theta_i(0),v_i;\boldsymbol{w})\right|^2 \\ L_\theta(\boldsymbol{w}) &= \frac{1}{N_f}\sum_{i=1}^{N_f}\left|f_\theta(t_i,v_i,\varphi_i;\boldsymbol{w})\right|^2 \end{aligned}, \tag{9}$$

where N_f represents the total number of collocation points, and index i refers to the i^{th} collocation point. Therefore, the loss function can be decomposed into the following three main components, with each part described specifically in Equation (10). These loss terms correspond to the contribution of each Ordinary Differential Equation (ODE) residual in the Physics-Informed Neural Network (PINN), ensuring that the network fully adheres to the physical laws during the training process. By accumulating the residuals of each collocation point into the overall loss function, PINN can gradually approximate the true dynamic behavior of the system, thereby improving the accuracy and consistency of the predictions.

$$L(\boldsymbol{w}) = \lambda_x L_x(\boldsymbol{w}) + \lambda_y L_y(\boldsymbol{w}) + \lambda_\theta L_\theta(\boldsymbol{w}), \tag{10}$$

where λ_x, λ_y, and λ_θ represent the weights of the three loss terms respectively. In this study, an adaptive loss balancing technique is employed, which dynamically adjusts the weights of these three loss terms. This method automatically updates the weights during each training epoch using Maximum Likelihood Estimation (MLE) [40], ensuring a reasonable balance between the contributions of each loss term throughout the training process. This strategy not only improves the training efficiency and accuracy of the model, but also effectively prevents performance degradation caused by improper weight allocation. For a detailed description of this technology, refer to the work of Xiang et al. [35]. Let $\lambda = \{\lambda_x, \lambda_y, \lambda_\theta\}$

represent the adaptive weights of the three loss terms; then, the loss function can be expressed in the form of the following Formula (11).

$$L(\boldsymbol{\lambda}, \boldsymbol{w}) = \frac{1}{2}e^{-\log \lambda_x^2} L_x(\boldsymbol{w}) + \frac{1}{2}e^{-\log \lambda_y^2} L_y(\boldsymbol{w}) + \frac{1}{2}e^{-\log \lambda_\theta^2} L_\theta(\boldsymbol{w}), \\ + \log \lambda_x^2 + \log \lambda_y^2 + \log \lambda_\theta^2 \tag{11}$$

During network training, model parameters are optimized by minimizing the loss function, which can be achieved using either the ADAM optimizer [41] or the L-BFGS optimizer [42]. In this process, $\boldsymbol{\lambda}$ is treated as an additional network parameter and is optimized together with the network weights $\boldsymbol{\lambda}$ and $\boldsymbol{w}$. These parameters are synchronously updated in each training epoch by minimizing the loss function $L(\boldsymbol{\lambda}, \boldsymbol{w})$, ensuring that both the network weights and the adaptive weights can be adjusted collaboratively to improve the overall performance of the model. This joint optimization strategy not only optimizes the expression ability of the network, but also enhances the adaptability and prediction accuracy of the model to complex systems.

4. Nonlinear Model Predictive Control (MPC) Based on LB-TFC-PINN

In the following discussion, the proposed method is simplified as the LB-TFC-PINN model. After training, this model can be represented in the form of Equation (12).

$$\boldsymbol{z}(t) = NN(t, \boldsymbol{z}(0), \boldsymbol{u}(0); \boldsymbol{w}) \; t \in [0, \tau], \tag{12}$$

where $\boldsymbol{z}(t) = [x(t), y(t), \theta(t)]^{\mathrm{T}}$ represents the output state vector, while $\boldsymbol{u}(0) = [v(0), \varphi(0)]^{\mathrm{T}}$ denotes the control vector within the time interval $[0, \tau]$. Thus, the behavior of the system during this time interval $[0, \tau]$ can be simulated using arbitrary initial states and control actions. For the case of $t > \tau$, for instance, $t \in [\tau, 2\tau]$, the prediction of LB-TFC-PINN at time $t = \tau$, denoted as $\boldsymbol{z}(\tau) = NN(\tau, \boldsymbol{z}(0), \boldsymbol{u}(0); \boldsymbol{w})$, $[\tau,\ 2\tau]$ is used as the new initial state for the next time interval, $\boldsymbol{z}(t) = NN(t, \boldsymbol{z}(\tau), \boldsymbol{u}(\tau); \boldsymbol{w})$, $t \in [\tau, 2\tau]$. By iteratively repeating this process, long-term simulation of the system can be achieved. This method is called the self-looping mode, proposed by Antonelo et al. [33]. The self-cycling continuously updates the initial state and control vector, enabling the LB-TFC-PINN to effectively capture the system's dynamic changes over a longer period of time, thereby providing accurate predictions. The advantage of this method lies in its flexibility and adaptability to complex systems, making it widely applicable in practical applications.

The nonlinear MPC scheme based on LB-TFC-PINN is illustrated in Figure 3. At each time step, LB-TFC-PINN replaces the traditional nonlinear AGV kinematic model with MPC prediction. At time k^{th}, the measured real state z_k is used as the initial state input for LB-TFC-PINN. Subsequently, MPC calculates a series of control increments $\Delta u_k, \ldots, \Delta u_{k+N-1}$ to minimize the cost function J (see Equation (4)) and satisfy the predefined constraints (see Equation (5)). When solving this nonlinear optimization problem, the states $z_{k+1}, \ldots, z_{k+N}$ involved in the cost function are predicted by LB-TFC-PINN. Next, the first calculated control increment Δu_k is applied to the system as the control input u_k, and it is held until the next time step $(k+1)^{th}$. At time $(k+1)^{th}$, the actual state z_{k+1} is measured again and used as the new initial state input for LB-TFC-PINN, and the process is repeated. By continuously updating the control inputs at each time step, MPC can dynamically adjust the system's behavior to ensure optimal control performance in complex environments. This LB-TFC-PINN-based MPC approach significantly improves the response speed and control accuracy of the AGV system in complex nonlinear motion control tasks through precise state prediction and dynamic control input adjustment, demonstrating stronger adaptability and stability in practical applications.

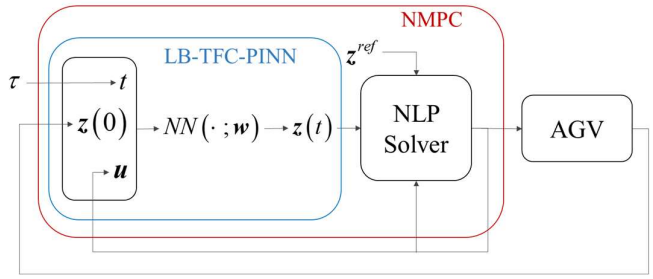


Figure 3. LB-TFC-PINN-based nonlinear MPC scheme.

As previously mentioned, LB-TFC-PINN is trained for state predictions within the time interval $[0, \tau]$. In practical applications, τ is typically set as the sampling interval. For example, when the sampling interval is $T_s = 0.5\text{ s} = \tau$ and the prediction horizon spans 10 time steps, LB-TFC-PINN needs to perform ten self-looping iterations at each MPC time step to predict the system's states over the next ten time steps. This prediction method ensures that the dynamic behavior of the system over multiple time steps in the future can be accurately captured, significantly enhancing MPC's control effectiveness and the overall performance of the system. Through this approach, LB-TFC-PINN provides strong support for the long-term stability and control accuracy of complex dynamic systems.

5. Results and Discussion

The simulation experiment in this section was conducted on the MATLAB R2023b platform. First, the solution of Equation (1) with arbitrary initial conditions and control actions is learned through LB-TFC-PINN. The designed network architecture consists of four hidden layers, each layer consisting of 64 neurons, and utilizes the tanh activation function. The training process was conducted over the time interval $[0, \tau]$, $\tau = 0.5$ s. The initial conditions were set as $x, y \in [-15\text{ m},\ 15\text{ m}]$, $\theta \in [-2\pi,\ 2\pi]$, $v \in [-2\text{ m/s},\ 2\text{ m/s}]$, and $\varphi \in [-\pi/6,\ \pi/6]$. A total of Nf = 100,000 collocation points were generated using Sobol sampling to cover diverse initial conditions and control actions. During the network training, i.e., the optimization of the loss function, the ADAM optimizer was used, and the network was trained for 10,000 epochs. Each epoch contained 1000 samples and was trained over 100 iterations. The initial adaptive weights $\lambda = \{1,\ 1,\ 1\}$ were set to balance the contributions of different loss terms. To ensure the stability and convergence of the training, the learning rate was set as shown in Equation (13), which gradually decreased as the training progressed. The initial learning rate was 0.01 with a decay rate of 0.005. The specific learning rate decay strategy was as follows: as the number of training iterations increased, the learning rate was progressively reduced according to the predefined decay rate, allowing for finer adjustments of the model parameters in later stages of training. Through this training strategy, LB-TFC-PINN can accurately learn the dynamic behavior of the system under diverse initial conditions and control actions, providing effective support for modeling and predicting complex dynamic systems.

$$\text{learning rate} = \frac{\text{initial learning rate}}{1 + \text{ decaying rate } \times \text{ iteration}}, \tag{13}$$

For comparison, a PINN model without the use of TFC and adaptive loss balancing techniques was also trained in this study, based on the method proposed by Antonelo et al. [33]. Unlike the TFC-based PINN, this model directly outputs the system's state variables x, y, and θ, instead of free functions. To ensure the accuracy of the initial conditions, a data-driven loss term was introduced into the loss function. Additionally, the loss function still included the ODE residual term to ensure the physical consistency of the system's dynamic behavior. However, due to the absence of TFC and adaptive loss balancing techniques, this model may face greater challenges in handling initial conditions and assigning loss weights.

LB-TFC-PINN adopted the same network architecture and training method as the traditional PINN. In PINN, an additional 1000 data points are introduced to enforce the initial conditions. Figure 4 presents the performance of LB-TFC-PINN and traditional PINN during the training process. It is particularly noteworthy that the loss function of LB-TFC-PINN included adaptive weights (see Equation (11)) to dynamically adjust the impact of each loss term. However, to ensure a fair comparison between the two models, the loss values shown in Figure 4 represents the total sum of the loss terms without applying any adaptive weights. Therefore, the loss values in Figure 4 do not reflect the impact of adaptive weights on model training.

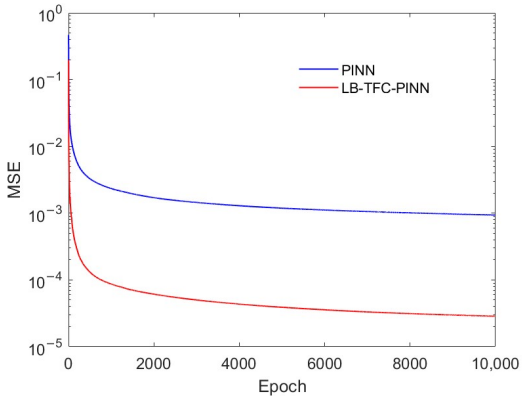


Figure 4. Training Performance.

It can be clearly seen from Figure 4 that LB-TFC-PINN exhibited significantly faster convergence speed during the training process. After only 1000 epochs, the loss value of LB-TFC-PINN decreased to less than 10^{-4}. In contrast, after 10,000 epochs of training, the loss of LB-TFC-PINN eventually stabilized at 2.8×10^{-5}, while the loss of traditional PINN could only converge to 10^{-3}. This result indicated that LB-TFC-PINN not only significantly improved the convergence speed, but also notably reduced the final loss value when dealing with complex dynamic systems, demonstrating superior training performance. This outcome further demonstrated the critical role of the TFC and adaptive loss balancing techniques in enhancing the training efficiency and accuracy of the PINN model.

Figure 5 shows the results of solving Equation (1) using LB-TFC-PINN, traditional PINN, and the fourth-order Runge–Kutta method (RK4). The initial conditions were set as $[x(0), y(0), \theta(0)]^{\mathrm{T}} = [0, 0, 0]^{\mathrm{T}}$, and the control input was fixed at $[v, \varphi]^{\mathrm{T}} = [1\ \mathrm{m/s}, \pi/24]^{\mathrm{T}}$. As shown in the figure, the solution obtained by LB-TFC-PINN was almost identical to the results from the RK4 method, demonstrating extremely high computational accuracy. In contrast, the traditional PINN failed to achieve satisfactory accuracy in approximating the solution of this ordinary differential equation (ODE). This comparison further demonstrates the outstanding performance of LB-TFC-PINN in simulating complex dynamic systems; especially when dealing with initial conditions and control inputs, its performance is significantly better than that of traditional PINN methods.

Next, the LB-TFC-PINN nonlinear MPC scheme discussed in Section 4 will be applied to the trajectory tracking problem of AGVs. In this scheme, the trained LB-TFC-PINN is used as a substitute for the AGV kinematic model within the MPC. To solve the Nonlinear Programming (NLP) problem, the fmincon function in MATLAB is utilized, where the Sequential Quadratic Programming (SQP) method is selected as the optimization algorithm, and the maximum number of iterations is set to 15. Table 1 lists the MPC parameters used in the simulation.

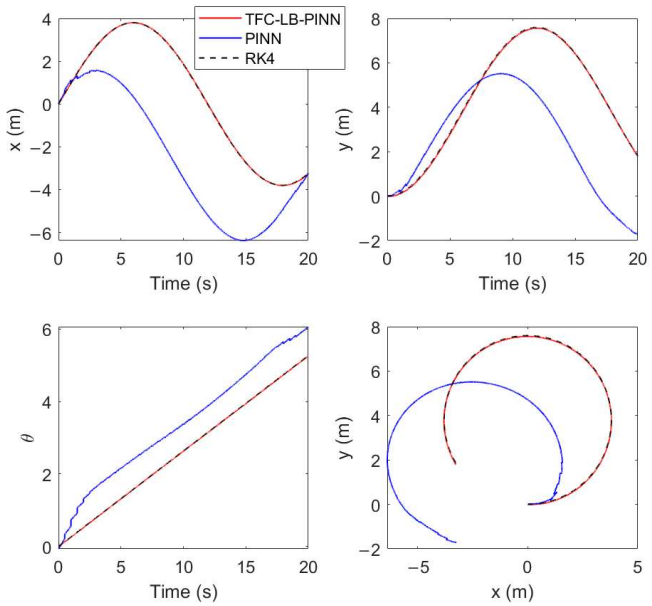


Figure 5. Simulation results from LB-TFC-PINN, PINN, and RK4.

Table 1. MPC parameters configured in the simulation.

Variable Name	Variable Values and Units
Sampling interval T_s	0.5 s
Prediction horizon N	10
Horizontal position coordinates x	$[-15\text{ m}, 15\text{ m}]$
Vertical position coordinates y	$[-15\text{ m}, 15\text{ m}]$
Vehicle speed	$[-2\text{ m/s}, 2\text{ m/s}]$
Vehicle yaw angle	$[-\pi/6, \pi/6]$
Vehicle speed variation	$[-0.2\text{ m/s}, 0.2\text{ m/s}]$
Vehicle yaw angle variation	$[-\pi/60, \pi/60]$

This paper explores two trajectory tracking tasks. In the first task (Task 1), the vehicle starts from a specific position $[x(0), y(0), \theta(0)]^{\mathrm{T}} = [0, 1.5, 0]^{\mathrm{T}}$, with the reference path defined by $x^{ref}(t) = 1.3t^{0.5}$ and $y^{ref}(t) = 3.5\sin(0.025t)$. For the second task (Task 2), the vehicle starts from a different position $[x(0), y(0), \theta(0)]^{\mathrm{T}} = [0, 6, 0]^{\mathrm{T}}$, with the reference path defined by $x^{ref}(t) = 11.5\sin(0.1t)$ and $y^{ref}(t) = 10\cos(0.05t)$. In practical scenarios, vehicles may deviate from their predetermined trajectory due to interference or uncertain factors. Therefore, intentionally setting the initial position of the vehicle to deviate from the reference trajectory is used to evaluate whether the designed control algorithm can effectively adjust the vehicle back to the trajectory quickly in the event of deviation. The purpose of this test is to verify the robustness and response speed of the algorithm, thereby ensuring the reliability and stability of the system in complex environments. In both tasks, the reference yaw angle θ^{ref} is not specified, and the task requirements are limited to position tracking. Therefore, the position states x and y are regarded as the primary outputs of the AGV system. To optimize the position tracking accuracy, the weight matrices Q and R are defined as key parameters, as shown in Equation (14). In the simulations for both tasks, the effectiveness of the LB-TFC-PINN-based nonlinear MPC scheme is validated, demonstrating its significant advantages in precise tracking and dynamic adjustments. During the simulation process, the AGV was treated as the controlled object, and the fourth-order Runge–Kutta (RK4) method with a fixed time step of 0.01 s was used for

the simulation. This simulation method ensured consistency in the time step, thereby improving the overall simulation accuracy.

$$\boldsymbol{Q} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \boldsymbol{R} = \begin{bmatrix} 0.01 & 0 \\ 0 & 0.01 \end{bmatrix}, \tag{14}$$

For comparative analysis, this paper also provides simulation results where the MPC uses the Ordinary Differential Equation (ODE) model (Equation (1)) as the prediction model instead of the LB-TFC-PINN. In this case, the system states in the MPC were computed using the fourth-order Runge–Kutta (RK4) method with a fixed time step of 0.01 s. Figures 6–10 present the trajectory tracking simulation results and the control actions generated by the MPC in these two tracking tasks.

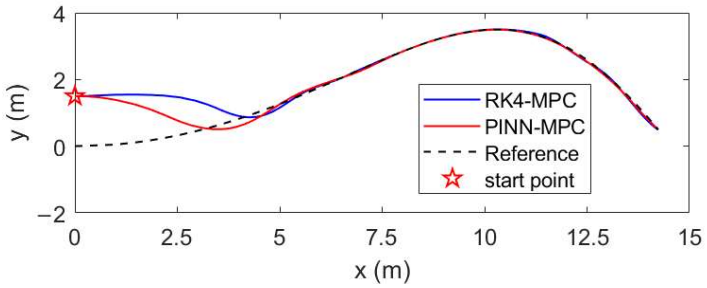


Figure 6. Trajectory tracking simulation results (Task 1).

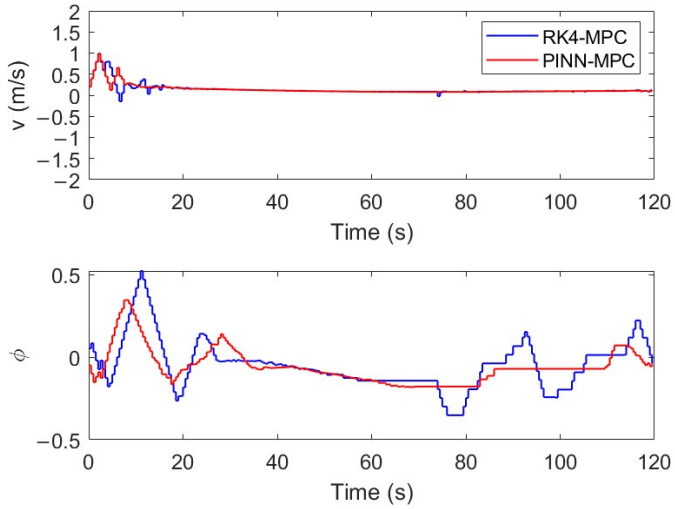


Figure 7. Control variables change over time (Task 1).

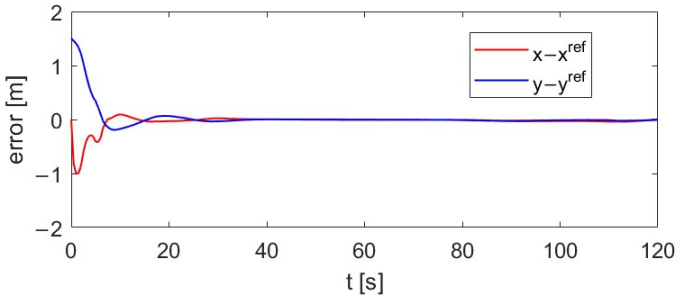


Figure 8. Error over time of the solution of LB-TFC-PINN-based nonlinear MPC for the tracking problem (Task 1).

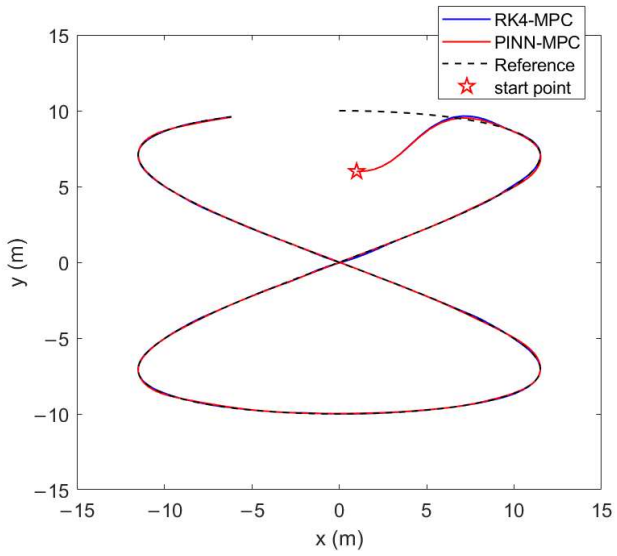


Figure 9. Trajectory tracking simulation results (Task 2).

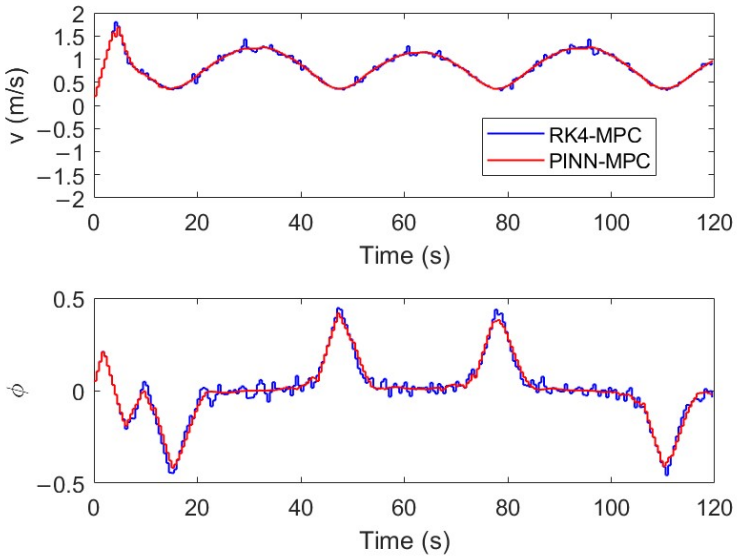


Figure 10. Control variable changes over time (Task 2).

In both tracking tasks, the performance of the LB-TFC-PINN-based nonlinear MPC was nearly identical to that of the RK4-based nonlinear MPC. While the RK4 method, as a classical numerical solving technique, offers high accuracy, the LB-TFC-PINN is also capable of providing precise predictions while significantly reducing the computational burden. As previously mentioned, to lower computational costs, the maximum number of iterations for the NLP solver was limited to 15. Although increasing the number of iterations could potentially improve the solution accuracy, as shown in the results from Figures 6 and 9, the current solution has already reached an acceptable level of accuracy, and further iterations are unnecessary and would increase computation time.

In this study, all simulations were conducted on a desktop computer equipped with a quad-core 2.9 GHz CPU and 16 GB of RAM. The results showed that the average computation time for the complete simulation using the LB-TFC-PINN-based nonlinear MPC scheme was 35.2 s, while the RK4-based nonlinear MPC scheme required 101.6 s. Figures 8 and 11 present the error analysis for Task 1 and Task 2, respectively. The error was defined as the difference between the vehicle's actual position and the reference position (i.e., $x(t) - x^{ref}(t)$ and $y(t) - y^{ref}(t)$). As shown in both tasks, the error quickly converged to zero within a short period, indicating that the vehicle could promptly return to the reference trajectory

and follow it stably. This demonstrated the feasibility and effectiveness of the method proposed in this paper. Additionally, TFC and the self-adaptive loss balancing technique significantly improved the training efficiency and performance of the PINN. Once the PINN training is completed, the model can directly predict any state within the time interval $[0, \tau]$ without requiring numerical integration. Therefore, using PINN instead of the AGV model can effectively enhance the computational efficiency of MPC. This demonstrates that the LB-TFC-PINN not only matches traditional methods in terms of accuracy, but also exhibits significant advantages in computational efficiency. This efficiency improvement is particularly critical in real-world applications, especially in scenarios that demand real-time control and rapid responses. The LB-TFC-PINN offers a more efficient and feasible solution by greatly reducing computation time, enabling real-time control of complex systems while ensuring the reliability and stability of the system's performance.

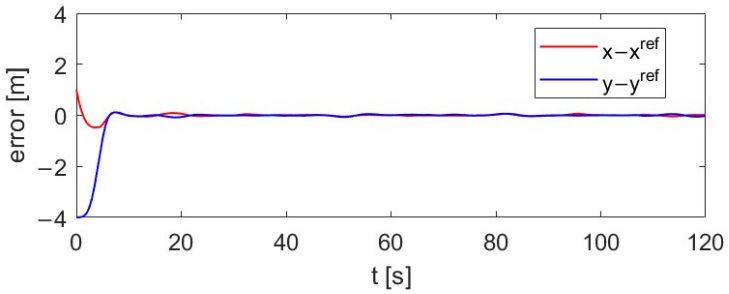


Figure 11. Error over time of the solution of LB-TFC-PINN-based nonlinear MPC for the tracking problem (Task 2).

6. Conclusions

This paper explores the potential of Physics-Informed Neural Networks (PINNs) in applying nonlinear Model Predictive Control (MPC) for AGV trajectory tracking. This study first trains a PINN to learn the kinematic model of the AGV, and then replaces the traditional kinematic model in the nonlinear MPC with the trained PINN model. In order to meet the requirements of Model Predictive Control, the input layer of the PINN is specifically designed to include initial conditions and control inputs, in order to accurately simulate the dynamic behavior of the system under different initial states and control actions.

To improve training efficiency and avoid competition between objective functions during the training process, this paper introduces the Theory of Functional Connections (TFC), which can analytically satisfy the initial conditions and avoid the phenomenon of objective function conflicts in traditional PINNs. Additionally, an adaptive loss balancing technique is employed to address the sensitivity of PINNs to the weights of different loss terms, ensuring the stability and efficiency of the training process. Through numerical simulations, it is verified that the proposed training method enables the PINN to accurately and efficiently learn the solutions of ordinary differential equations (ODEs), further validating the effectiveness of the PINN-based MPC scheme in trajectory tracking tasks.

It is particularly noteworthy that the introduction of the PINN model to nonlinear MPC significantly reduces the computational burden of solving ODEs. This is because the trained PINN can directly output the system's predicted results, without relying on the numerical integration process in traditional methods. This feature not only greatly improves computational efficiency, but also provides a more feasible solution for real-time control of complex dynamic systems. This study demonstrates the potential application of PINN-based MPC in AGV trajectory tracking, laying a solid foundation for future research and applications in related fields. Next, efforts are being made to assemble a real car model, which will be used to experimentally validate the method proposed in this article. We will also further support our research hypotheses and conclusions through real vehicle testing. In addition, future real vehicle experiments will record the experimental process and data in detail, providing reliable experimental basis for further research.

Author Contributions: Conceptualization, Y.L. and L.L.; methodology, Y.L. and L.L.; software, validation, Y.L. and L.L.; formal analysis, Y.L.; investigation, Y.L.; data curation, Y.L. and L.L.; writing—original draft preparation, writing—review and editing, Y.L.; supervision, L.L.; project administration, funding acquisition, L.L. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the technology development project "Research on Intelligent Vehicle Path Planning and Control System" jointly established by the Qilu University of Technology and Beijing Institute of Technology Frontier Technology.

Data Availability Statement: The original contributions presented in the study are included in the article, further inquiries can be directed to the corresponding author.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Chen, W.; Guo, C.; Li, J. Trajectory planning and tracking control for AGV in complex environments. *J. Intell. Robot. Syst.* **2020**, *99*, 21–34.
2. Liu, S.; Feng, X. A review on trajectory optimization and control for autonomous ground vehicles. *IEEE Trans. Intell. Transp. Syst.* **2021**, *22*, 3495–3509.
3. Wang, J.; Zhao, Z. Robust control methods for AGV systems in logistics applications. *Int. J. Control. Autom. Syst.* **2023**, *21*, 1127–1136.
4. Caban, J.; Nieoczym, A.; Dudziak, A.; Krajka, T.; Stopková, M. The Planning Process of Transport Tasks for Autonomous Vans—Case Study. *Appl. Sci* **2022**, *12*, 2993. [CrossRef]
5. Chen, B.; Sun, J. Robust linear MPC for autonomous vehicle path following under parameter uncertainties. *IEEE Trans. Control Syst. Technol.* **2019**, *27*, 1991–2000.
6. Nazat, S.; Osvaldo, A.; Mustafa, A. On Evaluating Black-Box Explainable AI Methods for Enhancing Anomaly Detection in Autonomous Driving Systems. *Sensors* **2024**, *24*, 3515. [CrossRef]
7. Zhou, D.; Li, P. Nonlinear model predictive control of hybrid electric vehicles for energy management. *IEEE Trans. Veh. Technol.* **2021**, *70*, 8833–8845.
8. Zhang, T.; Zhao, Y. Real-time nonlinear MPC for vehicle path tracking with actuator dynamics. *IEEE Control Syst. Lett.* **2022**, *7*, 1516–1521.
9. Guo, Q.; He, Z. A data-driven approach to model predictive control for vehicle stability enhancement. *Automob. Eng. J.* **2020**, *52*, 1123–1135.
10. Wang, F.; Ma, X. Data-driven model predictive control for autonomous vehicle overtaking maneuvers. *J. Adv. Veh. Syst.* **2023**, *75*, 45–56.
11. Xu, J.; Shi, L. Model predictive control with reinforcement learning for vehicle longitudinal control. *IEEE Trans. Neural Netw. Learn. Syst.* **2021**, *32*, 2898–2910.
12. Liu, Y.; Xu, H. Model predictive control for connected vehicles in mixed traffic environments. *J. Intell. Transp. Syst.* **2022**, *26*, 395–408.
13. Li, S.; Chen, G. Adaptive model predictive control for lane keeping in autonomous vehicles. *J. Control Theory Appl.* **2020**, *18*, 333–342.
14. Parish Eric, J.; Karthik, D. A paradigm for data-driven predictive modeling using field inversion and machine learning. *J. Comput. Phys.* **2016**, *305*, 758–774. [CrossRef]
15. Karniadakis, G.E.; Kevrekidis, I.G.; Lu, L.; Perdikaris, P.; Wang, S.; Yang, L. Physics-informed machine learning. *Nat. Rev. Phys.* **2021**, *3*, 422–440. [CrossRef]
16. Sun, L.; Wang, Z. Physics-informed neural networks for fluid dynamics: A review. *Comput. Fluids* **2022**, *235*, 105278.
17. Geyer, J.; Kassahun, Y.; Mahmudi, M.; Ricou, X.; Durgesh, R.; Chung, A.S.; Hauswald, L.; Pham, V.H.; Mühlegg, M.; Dorn, S.; et al. A2D2: Audi Autonomous Driving Dataset. *arXiv* **2020**, arXiv:2004.06320.
18. Alkhaldi, A.; Mortari, D. Solving nonlinear differential equations using the Theory of Functional Connections. *J. Comput. Appl. Math.* **2020**, *368*, 112493.
19. Franzini, G.R.; Mortari, D. Theory of Functional Connections for constrained optimization problems. *Optim. Methods Softw.* **2021**, *36*, 1012–1033.
20. Cao, B.; Li, Y.; Zhang, C.; Xie, Y. Adaptive loss weight balancing for multi-task learning with applications in autonomous driving. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Long Beach, CA, USA, 15–20 June 2019; pp. 11983–11992.
21. Kendall, A.; Gal, Y.; Cipolla, R. Multi-task learning using uncertainty to weigh losses for scene geometry and semantics. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Salt Lake City, UT, USA, 18–23 June 2018; pp. 7482–7491.

22. Wang, Z.; Yang, Z.; Zhou, D. A novel adaptive loss weight adjustment strategy for deep learning models applied to imbalanced data. *J. Artif. Intell. Res.* **2021**, *70*, 123–145.
23. Raissi, M.; Perdikaris, P.; Karniadakis, G.E. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *J. Comput. Phys.* **2019**, *378*, 686–707. [CrossRef]
24. Koller, D.; Buchli, J. Model predictive control for trajectory tracking of unmanned aerial vehicles using nonlinear model learning. *IEEE Trans. Robot.* **2019**, *35*, 62–77.
25. Lee, D.; Hwang, K. A review on the recent advances in model predictive control for automotive applications. *IEEE Trans. Veh. Technol.* **2021**, *70*, 1032–1046.
26. Lefeber, R.; Pappalardo, C.M. Dynamics and Control of Automated Guided Vehicles. *J. Control Eng. Pract.* **2020**, *29*, 123–133.
27. Wang, P. A Novel Kinematic Model for Predicting Vehicle Motion in Dynamic Environments. *J. Veh. Dyn. Control* **2023**, *15*, 201–212.
28. Smith, J. Discretization Techniques for Continuous-Time Control Systems in Digital Applications. *IEEE Trans. Control Syst. Technol.* **2022**, *30*, 1023–1034.
29. Leva, A.; Cimino, C.; Seva, S. A control education software suite to bridge methodological and engineering aspects. *IFAC-Pap. OnLine* **2020**, *53*, 17179–17184. [CrossRef]
30. Gould, N.I.M.; Robinson, D.P. A second derivative SQP method: A numerical investigation. *Comput. Optim. Appl.* **2021**, *79*, 505–522.
31. Gros, S.; Zanon, M.; Quirynen, R.; Bemporad, A.; Diehl, M. From linear to nonlinear MPC: Bridging the gap via the real-time iteration. *Int. J. Control* **2016**, *l88*, 1577–1601. [CrossRef]
32. Farea, A.; Yli-Harja, O.; Emmert-Streib, F. Understanding Physics-Inform-ed Neural Networks: Techniques, Applications, Trends, and Challenges. *AI* **2024**, *5*, 1534–1557. [CrossRef]
33. Antonelo, E.A.; Camponogara, E. Physics-informed neural nets for control of dynamical systems. *Neurocomputing* **2024**, *579*, 127419. [CrossRef]
34. Johnston, H.; Mortari, D. Least-squares solutions of eighth-order boundary value problems using the Theory of Functional Connections. *Mathematics* **2020**, *8*, 397. [CrossRef] [PubMed]
35. Xiang, Z.X.; Peng, W.; Liu, X.; Yao, W. Self-adaptive loss balanced Physics-informed neural networks. *Neurocomputing* **2022**, *496*, 11–34. [CrossRef]
36. Jagtap, A.D.; Kawaguchi, K.; Karniadakis, G.E. Locally adaptive activation functions with slope recovery for deep and physics-informed neural networks. *Proc. R. Soc. A Math. Phys. Eng. Sci.* **2020**, *476*, 20200334. [CrossRef] [PubMed]
37. Kaminski, B.; Lagos, G. New approaches in automatic differentiation for large-scale dynamic optimization. *Optim. Methods Softw.* **2020**, *35*, 191–219.
38. Beck, C.; Weinan, E. Machine Learning Approximation Algorithms for High-Dimensional Fully Nonlinear Partial Differential Equations and Second-Order Backward Stochastic Differential Equations. *J. Nonlinear Sci.* **2019**, *29*, 1563–1619. [CrossRef]
39. Borisut, P.; Nuchitprasittichai, A. Adaptive Latin Hypercube Sampling for a Surrogate-Based Optimization with Artificial Neural Network. *Processes* **2023**, *11*, 3232. [CrossRef]
40. Narayanan, A.; Mantiuk, R.K. Maximum Likelihood Estimation for Adaptive Image Processing. *ACM Trans. Graph.* **2022**, *41*, 1–13.
41. Schneider, M.; Hennig, P. A modified Adam algorithm for deep neural network optimization. *Neural Comput. Appl.* **2023**, *35*, 3891–3902.
42. Baker, J.; Kumar, R.; Lopez, G. Large-scale distributed L-BFGS. *J. Big Data* **2023**, *10*, 89–102.

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.