Journal of
Marine Science and Engineering

MDPI

Article

Optimized Trajectory Tracking for ROVs Using DNN + ENMPC Strategy

Guanghao Yang [1], Weidong Liu [1], Le Li [1,*], Jingming Xu [1], Liwei Guo [1] and Kang Zhang [2]

[1] School of Marine Science and Technology, Northwestern Polytechnical University, Xi'an 710072, China
[2] Wuhan Second Ship Design and Research Institute, Wuhan 430223, China
* Correspondence: leli@nwpu.edu.cn

Abstract: This study introduces an innovative double closed-loop 3D trajectory tracking approach, integrating deep neural networks (DNN) with event-triggered nonlinear model predictive control (ENMPC), specifically designed for remotely operated vehicles (ROVs) under external disturbance conditions. In contrast to single-loop model predictive control, the proposed double closed-loop control system operates in two distinct phases: (1) The outer loop controller uses a DNN controller to replace the LMPC controller, overcoming the uncertainties in the kinematic model while reducing the computational burden. (2) The inner loop velocity controller is designed using a nonlinear model predictive control (NMPC) algorithm with its closed-loop stability proven. A DNN + ENMPC 3D trajectory tracking method is proposed, integrating a velocity threshold-triggered mechanism into the inner-loop NMPC controller to reduce computational iterations while sacrificing only a small amount of tracking control performance. Finally, simulation results indicate that compared with the ENMPC algorithm, NMPC + ENMPC can better track the desired trajectory, reduce thruster oscillations, and further minimize the computational load.

Keywords: remotelyoperated vehicles; deep neural network; event-triggered nonlinear model predictive control; trajectory tracking

check for updates

Citation: Yang, G.; Liu, W.; Li, L.; Xu, J.; Guo, L.; Zhang, K. Optimized Trajectory Tracking for ROVs Using DNN + ENMPC Strategy. *J. Mar. Sci. Eng.* **2024**, *12*, 1827. https://doi.org/10.3390/jmse12101827

Academic Editor: Rafael Morales

Received: 4 September 2024
Revised: 3 October 2024
Accepted: 9 October 2024
Published: 13 October 2024

1. Introduction

Remotely operated vehicles (ROVs) are advanced underwater equipment that have been increasingly used in various marine applications, such as seabed exploration, aquaculture, underwater structure maintenance, and rescue operations [1,2]. With the increasing development of ROV technology, 3D trajectory tracking has become a highly important area of research [3].

The 3D trajectory tracking problem for ROVs involves developing a control system that enables the vehicle to accurately follow a predetermined path from its initial position while ensuring the trajectory remains smooth and well-constrained throughout the entire process [4]. The inherently complex nonlinear dynamics, external disturbances, and uncertainties of ROV systems make the design of robust 3D trajectory tracking controllers particularly challenging. These factors significantly increase the difficulty of achieving precise and reliable control in underwater environments. To address the challenges posed by ROV systems, a variety of control methods have been proposed in recent years, including proportional–integral–derivative (PID) control [5,6], sliding mode control [7–9], and neural network control [10].

Among the above control methods, model predictive control (MPC) excels at managing multi-input, multi-output systems with constraints. The primary concept behind MPC involves resolving a discrete-time optimal control problem subject to constraints for the current state and using the first control action from the optimal control sequence to adjust the system state. Through the inclusion of terminal costs and constraints, MPC is capable of maintaining the stability of nonlinear systems under control restrictions [11].

When addressing the challenges of nonlinear models, MPC typically employs two approaches. The first approach is to linearize the nonlinear dynamics and reformulate the control problem as a standard quadratic programming task [12]. Liu et al. [13] proposed an adaptive MPC method based on Gaussian process regression for underwater robotic arms. This approach effectively transforms the nonlinear system into a linearized form and addresses the constraint control problem through optimization, thereby enhancing the overall control performance of underwater robotic arms. The second approach is to use nonlinear model predictive control (NMPC) to directly solve the nonlinear model. For instance, Wang et al. [14] applied NMPC to multi-rate networked industrial process control, and Ellis et al. [15] demonstrated its effectiveness in economic models. Qin [16] proposed a fuzzy NMPC approach, integrating a fixed-time extended state observer (FXESO) to enhance path following and speed maintenance in underactuated surface vessels with four degrees-of-freedom. This method was validated through simulations, demonstrating that NMPC is a robust and effective solution for addressing complex control challenges in nonlinear systems.

Building upon single-loop MPC, dual-loop MPC has gained significant attention to further enhance control performance. Yan et al. [17] introduced a dual-loop MPC algorithm specifically designed for AUVs, employing linear model predictive control (LMPC) strategies for both the inner and outer control loops. Experimental results demonstrated that dual-loop MPC outperforms single-loop control in terms of constraint handling and overall control effectiveness, particularly under dynamic underwater conditions. Yan et al. [18] proposed a dual-loop MPC-based control strategy for a five-degrees-of-freedom Autonomous Underwater Vehicle (AUV), utilizing a finite-time extended state observer to compensate for model uncertainties and a nonlinear auxiliary control law to enhance robustness against external disturbances, which was validated through comparative simulations. In [19], an enhancement to dual-loop MPC was proposed by incorporating an ocean current observer into the outer loop, which significantly improved the stability and robustness of AUVs in the presence of ocean current disturbances. While dual-loop MPC provides superior control capabilities, its main challenge lies in the substantial computational burden required for online optimization, a critical issue that this paper seeks to address.

Dual-loop MPC requires simultaneous online optimization for both the inner and outer control loops, resulting in a significant computational burden. MPC not only demands precise mathematical models but also incurs substantial computational costs. However, machine learning offers an alternative by learning the system's dynamic behavior from training data without the need for explicitly defined models. Once training is complete, machine learning models can predict the system's future behavior, reducing reliance on exact models and thus alleviating both modeling complexity and computational overhead [20].

Liu et al. [21] developed a Multi-Layer Perceptron (MLP) neural network for a five-degrees-of-freedom underwater vehicle to effectively approximate model uncertainties and compensate for external disturbances. This method not only reduced the computational burden significantly but also improved overall system performance, demonstrating the neural network's efficiency and practicality in controlling five-degrees-of-freedom underwater robots. Additionally, Liu et al. [22] proposed a finite-time self-structuring neural network (SSNN) control scheme for five-degrees-of-freedom underwater vehicles. This approach optimally determines the number of neurons through simple computations, significantly improving the approximation and estimation of uncertainties in the dynamic model, which is critical for robust trajectory tracking. Moreover, Yang et al. [23] introduced a recurrent neural network (RNN) to address model uncertainties in four-degrees-of-freedom underwater vehicles, effectively compensating for the dynamic uncertainties and enhancing the system's stability and control accuracy. Guo et al. [24] employed Radial Basis Function (RBF) neural networks to mitigate external disturbances in ROVs. Inspired by these advancements, this paper proposes, for the first time in the context of ROV dual-loop control, the use of a DNN controller to replace the outer loop LMPC. The DNN can approximate

implicit solutions to optimization problems through offline learning, eliminating the need for real-time online optimization typical of MPC.

The dynamics of ROVs are inherently nonlinear, making the use of NMPC in the inner loop ideal for solving constrained nonlinear equations. Despite its advantages, NMPC carries a significant computational burden. To mitigate this, an event-triggered strategy is introduced in the inner loop controller. This strategy computes and transmits control inputs only when specific triggering conditions are met, thereby reducing computational frequency while maintaining stable control performance [25]. Li and Shi [26] demonstrated the feasibility and stability of event-triggered mechanisms in NMPC. Xu et al. [27] proposed an adaptive neural network event-triggered control scheme for the target tracking problem in underactuated AUVs. Simulations demonstrate that the neural network, combined with the event-triggered mechanism, achieves superior target tracking performance in a five-degree-of-freedom underwater robot. Zhang et al. [28] introduced an event-triggered strategy for a five-degrees-of-freedom underwater robot and achieved good experimental results, demonstrating the feasibility of the event-triggered mechanism in the five-degrees-of-freedom model. Several studies, including [29–31], have validated the effectiveness of event-triggered strategies through experimental results, showing that they can reduce computational resources while sacrificing only a small amount of control performance. In this paper, an event-triggered strategy based on the velocity deviations is applied to the inner loop velocity control of the ROV system, successfully reducing the computational burden while ensuring stable control performance.Both this paper and the literature [28] investigate the 3D trajectory tracking problem for underwater robots, focusing on the nonlinear model predictive control approaches and event-triggered mechanisms to reduce computational resources. Compared with the work presented by Zhang et al. [28], the DNN + ENMPC strategy differs in the following three aspects, i.e., using the double-loop closed control framework, adopting a DNN controller for the outer loop control, and proposing the velocity deviation-based event-triggering mechanism to reduce the number of triggers, thereby saving computational resources in the inner loop.

This paper proposes a dual closed-loop DNN + ENMPC strategy for the 3D trajectory tracking of the five-degree-of-freedom ROV. Given that the kinematic equations are relatively simple, a deep neural network (DNN) controller is adopted in the outer loop. The velocity threshold-based event-triggered NMPC strategy is proposed for the inner loop control. This design leverages the rapid computation of DNN in the outer loop and the precise control of NMPC in the inner loop. The introduction of the velocity threshold event-triggered strategy reduces computational demands, thereby improving the system's real-time performance and control efficacy.

The main contributions of this work are three-fold:

- A novel dual-loop control strategy has been designed that exhibits good control performance under external disturbances and ocean current conditions.
- Replace the outer-loop LMPC with an DNN, which improves the outer loop of the general dual-loop MPC. This approach addresses kinematic model uncertainties and reduces the computational burden to some extent.
- Enhance the inner loop of the dual-loop MPC by introducing an event-triggered strategy based on the speed deviation instead of the position deviation for the inner loop NMPC, further reducing the trigger frequency and the computational burden while sacrificing a small amount of tracking control performance.

The organization of this paper is as follows: Section 2 presents the mathematical model for the ROV. Section 3 introduces the proposed DNN + ENMPC strategy, including the design of the outer DNN controller and the inner ENMPC controller. Section 4 presents numerical simulations performed in MATLAB R2022b to validate the DNN + ENMPC strategy proposed herein. Lastly, Section 5 provides a summary of the experimental findings and lists the potential directions for further research.

2. Mathematical Model of ROV

Sections 2.1 and 2.2 provide detailed introductions to the kinematic model of the ROV and the dynamic model of the ROV, respectively.

2.1. Kinematic Model

To analyze the kinematic model of the ROV, two coordinate systems are introduced: the fixed coordinate system $E-\xi\eta\zeta$ and the moving coordinate system $O-xyz$, as illustrated in Figure 1. The fixed system represents a stationary point on Earth, while the moving system tracks the ROV's motion. The position and attitude in the Earth-fixed system $E-\xi\eta\zeta$ are expressed as $\boldsymbol{\eta}=[\xi,\eta,\zeta,\phi,\theta,\psi]^T$, where ξ,η,ζ denote the position and θ,ψ,ϕ represent the attitude. The linear and angular velocities in the moving system $O-xyz$ are represented by $\boldsymbol{v}=[u,v,w,p,q,r]^T$, where u,v,w correspond to forward, lateral, and vertical velocities, and p,q,r correspond to roll, yaw, and pitch angular velocities, respectively.

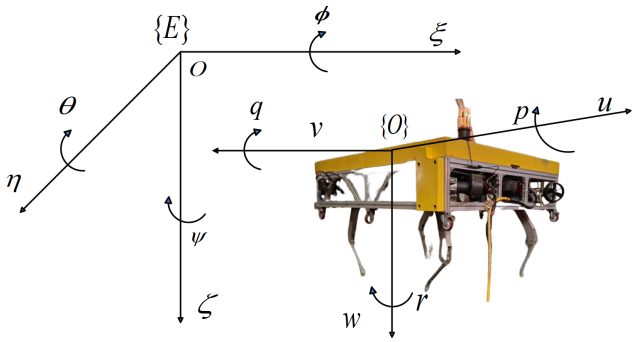

Figure 1. ROV coordinate system.

The design of ROVs is typically symmetrical in both the front–rear and left–right directions. During the design process, the gravity and buoyancy of the ROV are usually made approximately equal, ensuring relatively stable underwater motion, which allows the rolling degree of freedom to be neglected.

In this model, matrices are represented in boldface. The transformation equation from the ROV body frame to the earth-fixed frame is as follows [32]:

$$\dot{\boldsymbol{\eta}}=\boldsymbol{J}(\boldsymbol{\eta})\boldsymbol{v} \tag{1}$$

The ROV's coordinates in the earth-fixed frame are given by $\boldsymbol{\eta}=[\xi,\eta,\zeta,\theta,\psi]^T$, and in the body-fixed frame by $\boldsymbol{v}=[u,v,w,q,r]^T$.

$$\boldsymbol{J}(\boldsymbol{\eta})=\begin{bmatrix}\boldsymbol{J_1}(\boldsymbol{\eta}) & \boldsymbol{0_{3\times2}}\\ \boldsymbol{0_{2\times3}} & \boldsymbol{J_2}(\boldsymbol{\eta})\end{bmatrix} \tag{2}$$

$$\boldsymbol{J_1}(\boldsymbol{\eta})=\begin{bmatrix}\cos\theta\cos\psi & -\sin\psi & \sin\theta\cos\psi\\ \cos\theta\sin\psi & \cos\psi & \sin\theta\sin\psi\\ -\sin\theta & 0 & \cos\theta\end{bmatrix} \tag{3}$$

$$\boldsymbol{J_2}(\boldsymbol{\eta})=\begin{bmatrix}1 & 0\\ 0 & 1/\cos\theta\end{bmatrix} \tag{4}$$

The transformation matrix from the body-fixed frame to the earth-fixed frame is denoted as $J(\eta)$. To ensure the validity of Equation (1), the pitch angle θ and yaw angle ψ are subject to the following constraints: $-\pi/6\le\theta\le\pi/6$ and $-\pi\le\psi\le\pi$ [33].

2.2. Dynamic Model

The dynamic model of the ROV is established based on a set of assumptions as follows [34]. The vehicle is symmetrical in the x-z and x-y planes, and the length-to-width ratio of the vehicle is large, which simplifies some dynamic couplings. The model is

expressed in the vehicle's body-fixed frame, which is located on the vehicle's centerline, and the vehicle's center of gravity is also on this centerline. The center of buoyancy and the center of gravity lie on the same vertical axis, so the model does not account for lateral deviations related to buoyancy. Nonlinear damping is neglected, and only linear damping is considered, simplifying the calculations. The roll degree of freedom is ignored because the vehicle is passively stable in the roll direction. The vehicle is neutrally buoyant, so there is no need to account for changes in buoyancy. Therefore, the dynamic model of the five-degrees-of-freedom ROV can be expressed in vector form as follows:

$$\boldsymbol{M}\dot{\boldsymbol{v}} + \boldsymbol{C}(\boldsymbol{v})\boldsymbol{v} + \boldsymbol{D}(\boldsymbol{v})\boldsymbol{v} + \boldsymbol{g}(\boldsymbol{\eta}) = \boldsymbol{\tau} \tag{5}$$

The added mass matrix is symmetric along the diagonal. When the ROV exhibits one or more planes of symmetry, the added mass matrix can be simplified. Generally, for an ROV with symmetry in the left–right, top–bottom, and front–back planes, the off-diagonal elements of the added mass matrix can be neglected.

Here, $\boldsymbol{M} \in \mathbb{R}^{5\times5}$ denotes the inertia matrix, where $\boldsymbol{M} = \boldsymbol{M_{RB}} + \boldsymbol{M_{AM}}$. $\boldsymbol{M_{RB}}$ represents the rigid-body mass matrix, expressed as $\boldsymbol{M_{RB}} = \boldsymbol{diag\{m, m, m, I_y, I_z\}}$, and $\boldsymbol{M_{AM}}$ is the added mass matrix, given by $\boldsymbol{M_{AM}} = \boldsymbol{diag\{-X_{\dot{u}}, -Y_{\dot{v}}, -Z_{\dot{w}}, -M_{\dot{q}}, -N_{\dot{r}}\}}$. Here, m is the mass of the ROV, while I_y and I_z are the inertia tensors. The coefficients $X_{\dot{u}}$, $Y_{\dot{v}}$, $Z_{\dot{w}}$, $M_{\dot{q}}$, and $N_{\dot{r}}$ are hydrodynamic constants that are measurable.

The Coriolis and centripetal matrix $\boldsymbol{C}(\boldsymbol{v}) \in \mathbb{R}^{5\times5}$ is decomposed into the rigid-body Coriolis matrix $\boldsymbol{C_{RB}}(\boldsymbol{v})$ and the hydrodynamic Coriolis matrix $\boldsymbol{C_{AM}}(\boldsymbol{v})$, with $\boldsymbol{C}(\boldsymbol{v}) = \boldsymbol{C_{RB}}(\boldsymbol{v}) + \boldsymbol{C_{AM}}(\boldsymbol{v})$.

The matrices are defined as follows:

$$\boldsymbol{C_{RB}}(\boldsymbol{v}) = \begin{bmatrix} 0 & 0 & 0 & mw & -mv \\ 0 & 0 & 0 & 0 & mu \\ 0 & 0 & 0 & -mu & 0 \\ -mw & 0 & mu & 0 & 0 \\ mv & -mu & 0 & 0 & 0 \end{bmatrix} \tag{6}$$

$$\boldsymbol{C_{AM}}(\boldsymbol{v}) = \begin{bmatrix} 0 & 0 & 0 & -Z_{\dot{w}}w & Y_{\dot{v}}v \\ 0 & 0 & 0 & 0 & -X_{\dot{u}}u \\ 0 & 0 & 0 & X_{\dot{u}}u & 0 \\ Z_{\dot{w}}w & 0 & -X_{\dot{u}}u & 0 & 0 \\ -Y_{\dot{v}}v & X_{\dot{u}}u & 0 & 0 & 0 \end{bmatrix} \tag{7}$$

The damping term $\boldsymbol{D}(\boldsymbol{v})$ for the ROV in this paper is derived without accounting for coupling or higher-order terms. It primarily includes a linear damping matrix $\boldsymbol{D_L}(\boldsymbol{v})$ and a nonlinear damping matrix $\boldsymbol{D_N}(\boldsymbol{v})$. The equation is as follows:

$$\boldsymbol{D}(\boldsymbol{v}) = \boldsymbol{D_L}(\boldsymbol{v}) + \boldsymbol{D_N}(\boldsymbol{v}) \tag{8}$$

The linear damping matrix $\boldsymbol{D_L}(v)$ is defined as follows:

$$\boldsymbol{D_L}(\boldsymbol{v}) = \mathrm{diag}[X_u, Y_v, Z_w, K_p, M_q, N_r] \tag{9}$$

The nonlinear damping matrix is expressed as follows:

$$\boldsymbol{D_N}(v) = \mathrm{diag}[X_{|u|u}|u|, Y_{v|v|}|v|, Z_{w|w|}|w|, K_{p|p|}|p|, M_{q|q|}|q|, N_{r|r|}|r|] \tag{10}$$

The terms $X_u, Y_v, Z_w, M_q, N_r, X_{|u|u}|u|, Y_{|v|v}|v|, Z_{|w|w}|w|$, and $M_{|q|q}|q|, N_{|r|r}|r|$ represent measurable dynamic coefficients. The vector of restoring forces and moments due to gravity and buoyancy is given by $\boldsymbol{g}(\boldsymbol{\eta})$.

$$\boldsymbol{g}(\boldsymbol{\eta}) = \left[0, 0, 0, \rho v g \overline{GM_L} \sin\theta, 0\right]^T \tag{11}$$

For a neutrally buoyant system, $W = mg$, $B = \rho g v$, and $W = B$. Here, W refers to the weight, B denotes buoyancy, and $\overline{GM_L}$ is the vertical center of gravity. The input forces for each degree of freedom are represented as $\boldsymbol{\tau} = [\tau_u, \tau_v, \tau_w, \tau_q, \tau_r]$.

3. Design of a Dual-Loop Control System

This section focuses on the control system design for the ROV, utilizing a dual closed-loop structure. The outer loop is managed by a deep neural network (DNN), while the inner loop operates under an event-triggered nonlinear model predictive control (ENMPC) framework. The outer loop receives inputs such as the current velocity, previous velocity, and positional deviations from both the current and previous time steps and produces the desired velocity increment. The inner loop processes this velocity increment along with the ROV's actual current velocity to generate the necessary control input. A schematic diagram of this dual closed-loop system is presented in Figure 2.

In this section, the ROV executes 3D trajectory tracking, adhering to a smooth and continuous path. The expression for the desired trajectory is shown below:

$$\boldsymbol{\eta_d} = [\xi_d, \eta_d, \zeta_d, \theta_d, \psi_d]^T \tag{12}$$

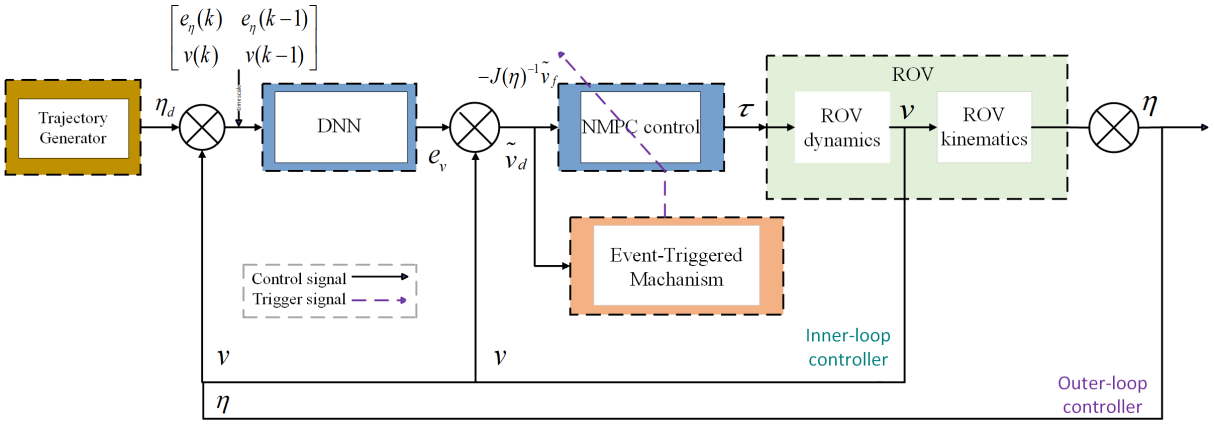

Figure 2. Diagram of the ROV control framework.

3.1. The Outer Loop DNN Design

In conventional dual-loop control systems, the outer loop controller typically uses positional deviation as input and generates the desired velocity as output. Yan enhanced the outer loop LMPC controller by modifying the inputs to include both positional deviation and velocity, with the output being the increment in desired velocity. Yan's study demonstrated that this approach, which accounts for the velocity rate of change, achieves superior control performance compared with single-loop MPC [17].

The primary goal of using a DNN in this paper is to address the kinematic model uncertainties of the ROV while also mitigating the computational challenges inherent in LMPC design. Although offline training of the DNN requires significant computational resources, its online implementation significantly reduces the computational load compared with LMPC. The training data for the DNN are generated using the LMPC method implemented in MATLAB R2022b . To aid the reader's understanding, this section begins by introducing the dynamic model used in Yan's LMPC controller. Section 3.1.1 outlines the structure of the DNN, and Section 3.1.2 presents the training and testing results.

The content up to Section 3.1.1 provides an introduction to the kinematic model of Yan's LMPC controller. By discretizing Equation (1), the corresponding discretized kinematic equations are derived, where T_s represents the sampling time.

$$\boldsymbol{\eta(k+1) = \eta(k) + J(k)v(k)}T_s \tag{13}$$

To improve control performance, we reformulated the kinematic Equation (13) and derived the subsequent Equations (14)–(20). The state variables in these equations represent both position and velocity, while the control input corresponds to changes in velocity. By imposing constraints on the control input, we can regulate the rate of velocity change, preventing the ROV from accelerating or decelerating too quickly, which ultimately enhances the overall control system performance.

$$x_\eta(k+1) = A_\eta(k)x_\eta(k) + B_\eta(k)\Delta v \tag{14}$$

$$y_\eta(k) = C_\eta(k)x_\eta(k) \tag{15}$$

$$x_\eta(k) = \begin{bmatrix} \eta(k) & v(k-1) \end{bmatrix}^T \tag{16}$$

$$\Delta v = v(k) - v(k-1) \tag{17}$$

with:

$$A_\eta(k) = \begin{bmatrix} I_{5\times5} & J(k)T_s \\ 0_{5\times5} & I_{5\times5} \end{bmatrix} \tag{18}$$

$$B_\eta(k) = \begin{bmatrix} J(k)T_s \\ I_{5\times5} \end{bmatrix} \tag{19}$$

$$C_\eta(k) = \begin{bmatrix} I_{5\times5} & 0_{5\times5} \end{bmatrix} \tag{20}$$

3.1.1. DNN Structure

This section presents the DNN as a position controller, describing its inputs, outputs, and internal structure. The outer-loop LMPC position controller receives 10 input elements, including the current position error and velocity, both represented as 5×1 vectors. The control output consists of five elements, representing velocity changes. While the DNN's input and output are theoretically similar to the LMPC, to further improve the DNN's performance, we also incorporate the position deviation and velocity from the previous time step into the input.

Figure 3 shows the structure of the DNN as the outer-loop controller. From the diagram, it can be seen that there are 20 control inputs. These 20 inputs consist of four 5×1 column vectors, including velocity at the previous sampling instant $v(s, t_k - \Delta t)$, position deviation at the previous time instant $e(s, t_k - \Delta t)$, current position deviation $e(s, t_k)$, and current velocity $v(s, t_k)$, where t_k is the current sampling instant and Δt is the sampling interval. The output layer consists of five nodes, representing the velocity increment Δv.

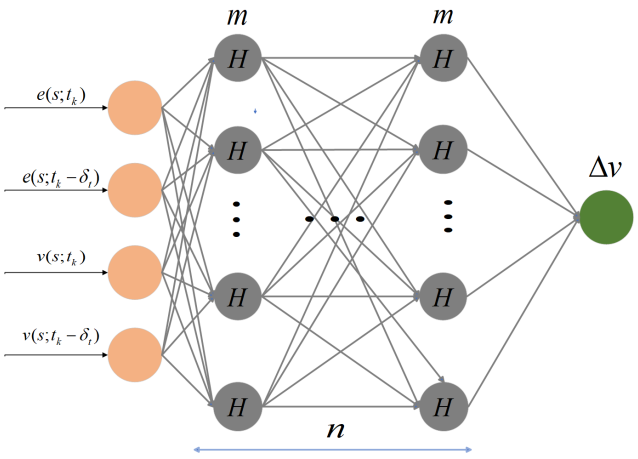

Figure 3. Representation of a DNN with n hidden layers and m neurons per layer.

A simple linear model with only input and output cannot adequately address the design challenges of the outer loop controller. By introducing hidden layers, the model can overcome these limitations and handle more complex functions. Mathematically, H represents the hidden layer variable. The hidden layers are fully connected, with parameters for the weights $W^{(1)}$ and biases $b^{(1)}$. Similarly, the output layer is fully connected, with weights $W^{(2)}$ and biases $b^{(2)}$. The formula for the hidden layer is given in Equation (21), where i denotes the number of the hidden layer.

The formula for the hidden layer can be represented as Equation (21), where i represents the number of the hidden layer.

$$\begin{cases} H^{(1)} = XW^{(1)} + b^{(1)} i = 1 \\ H^{(i)} = H^{(i-1)}W^{(1)} + b^{(1)} 2 \leq i \leq n \end{cases} \tag{21}$$

The formula for the output layer can be represented as Equation (22).

$$O = H(n)W^{(2)} + b^{(2)} \tag{22}$$

To leverage the potential of a multi-layer architecture, we apply a non-linear activation function σ to each hidden unit after the affine transformation. After incorporating the activation function, the formula for the hidden layer becomes Equation (23). With the activation function, it is no longer possible for our multi-layer perceptron to degenerate into a linear model.

$$\begin{cases} H^{(1)} = \sigma(XW^{(1)} + b^{(1)}) i = 1 \\ H^{(i)} = \sigma(H^{(i-1)}W^{(1)} + b^{(1)}) 2 \leq i \leq n \end{cases} \tag{23}$$

3.1.2. Training DNN

The DNN is trained using supervised learning, which necessitates a pre-recorded dataset. Collecting real-world ROV data is expensive, and running the MPC algorithm directly on ROV poses additional challenges. To address this, data are generated through MATLAB R2022b simulations, which is more time-efficient than using a physical ROV. The simulations incorporate both fully random and typical trajectories.

The DNN has been trained using a workstation with an AMD Ryzen 7 6800H processor (3.20 GHz), Thinkstation P430, Lenovo, Beijing, China, and Radeon Graphics, equipped with 16 GB of RAM (15.2 GB usable), Thinkstation P430, Lenovo, Beijing, China, and running a 64-bit operating system. Python version 3.10 is used for training, with the network consisting of four hidden layers ($n = 4$) and a variable number of neurons per layer (m). A learning rate of 0.001 is selected to ensure a balance between convergence speed and training stability, avoiding slow convergence or divergence. The Rectified Linear Unit (ReLU) activation function is applied to the hidden layers, with the mean square error used as the loss function and the ADAM optimizer employed for optimization.

Based on the data in Figure 4, increasing the number of neurons in each hidden layer enhances the DNN's ability to model nonlinearity in the training set, leading to better control performance. However, as shown in Figure 5, when the number of neurons increases from $m = 30$ to $m = 40$, performance on the testing set deteriorates due to overfitting, which impacts the controller's efficiency. Consequently, the final decision is to train the model with $m = 30$ neurons.

The trained neural network is switched to test mode. Using the same input data $e(s, t_k - \Delta t)$, $e(s, t_k)$, $v(s, t_k - \Delta t)$, and $v(s, t_k)$ for both the LMPC controller and the DNN, the effectiveness of the DNN's training is evaluated. As shown in Figure 6, the DNN's output closely matches that of the LMPC, demonstrating that the DNN achieves good control performance without relying on an accurate kinematic model.

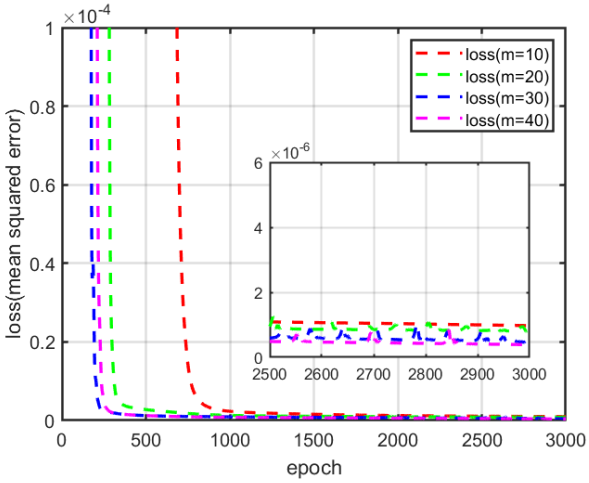

Figure 4. The loss function for training a DNN on a training set.

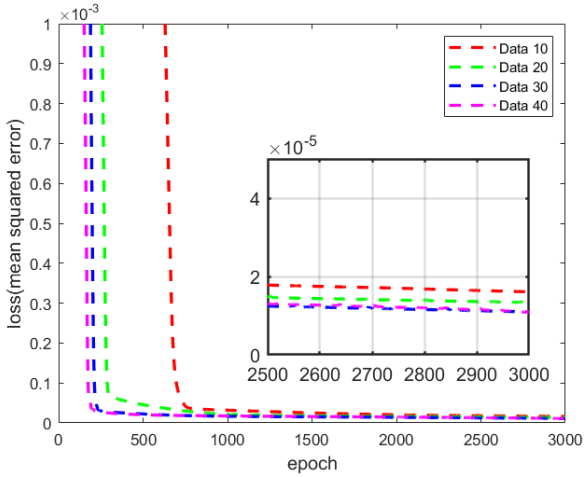

Figure 5. The loss function for testing a DNN on a test set.

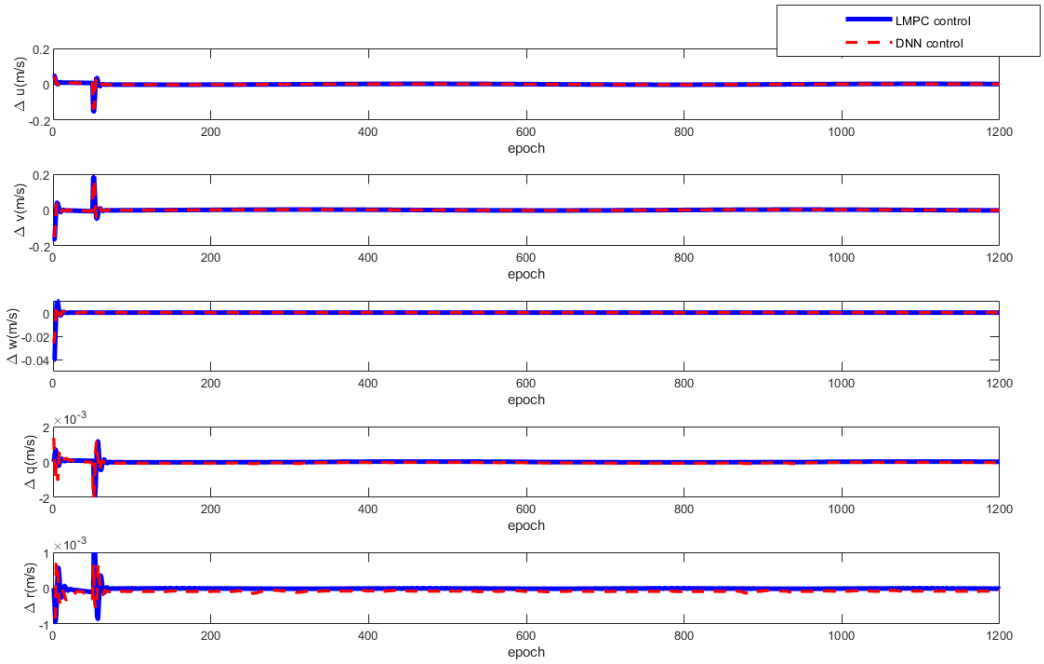

Figure 6. Comparison of outputs between LMPC and DNN under the same input conditions.

To compare the computational efficiency of the LMPC and DNN in ROV trajectory control, 20 sets of data were used. Each set contained 800 inputs, and both the LMPC and DNN were executed using Python on a system with an Intel(R) Core(TM) i7-12700F CPU @ 2.1 GHz and 16 GB of RAM, Huashuo BNH-20230215DZH, ASUS, Shanghai, China. The results showed that the LMPC's execution time ranged between 6.2 s and 6.8 s, while the DNN's time ranged from 4.9 s to 5.6 s. The average execution times for the LMPC and DNN were 6.4976 s and 5.3253 s, respectively.

In summary, DNN can achieve good control performance without the need for a kinematic model while reducing computation time.

3.2. The Inner Loop ENMPC Controller Design

This section discussed the inner loop velocity controller, starting with the introduction of the NMPC. The theoretical analysis covered the stability and feasibility of this approach. Additionally, introducing an event-triggered mechanism in ROV control is motivated by the limited computational capabilities of underwater vehicles. This mechanism reduces computational burden by sacrificing a small amount of control performance.

3.2.1. NMPC Inner Loop Controller

This section discusses the NMPC, where the inputs are the desired velocity and the deviation from the actual velocity, and the output is the corresponding control force. It introduces the control problem and highlights the algorithm's feasibility and stability.

By transforming Equation (5), we derive a nonlinear function of e, where $e = v - v_d$, with v representing the actual velocity and v_d the desired velocity of the ROV.

$$\begin{gathered} M\dot{v} + C(v)v + D(v)v + g(\eta) = \tau \\ \Downarrow \\ M\dot{e} + C(v)e + D(v)e + g(\eta) + (C(v) + D(v))v_d = \tau \end{gathered} \tag{24}$$

For the convenience of proving the inner-loop controller later, I discretized Equation (24) to obtain Equation (25).

$$\begin{cases} e(k+1) = F(e(k), \tau(k)) \\ y(k) = G\tau(k) \end{cases} \tag{25}$$

The predictive range of MPC is denoted as N_P and the control range as N_C. In this paper, N_C is consistently set to N_P, meaning $N_C = N_P = N$. Using Equation (25), the control sequence is derived as follows:

$$\begin{gathered} e(k+1|k) = F(e(k|k), \tau(k)) \\ e(k+2|k) = F(e(k+1|k), \tau(k+1)) \\ \vdots \\ e(k+N|k) = F(e(k+N-1|k), \tau(k+N-1)) \end{gathered} \tag{26}$$

Problem 1. *At sampling time k, the velocity error of the inner-loop NMPC controller is denoted as $e(k)$. The optimization problem for solving the optimal control of the inner-loop NMPC in discrete time can be described by expression (27).*

$$\begin{aligned} \min_{\bar{\tau}(\cdot)} J(e(k), \bar{\tau}(\cdot)) = & \sum_{i=0}^{N-1} (\|e(k+i|k)\|_Q^2 + \|\tau(k+i)\|_R^2) \\ & + \|e(k+N|k)\|_P^2 \end{aligned} \tag{27}$$

The inner loop control system satisfies both control constraints and state constraints:

$$\begin{gathered} e(k+i+1|k) = F(e(k+i|k+i), \tau(k+i)), i \in [0, N-1] \\ \tau(k+i) \in U, i \in [0, N-1] \\ e(k+i+1|k) \in X, i \in [0, N-1] \\ e(k+N|k) \in \Omega \end{gathered} \tag{28}$$

where $\|e(k+i|k)\|_Q^2$ *represents the velocity error stage cost function, and* $\|\tau(k+i)\|_R^2$ *represents the control input stage cost function. , and* $\|e(k+N|k)\|_P^2$ *represents the terminal constraint for the velocity error. Here,* Q*,* P*, and* R *are the relevant parameters.* N *is the finite prediction horizon, and* X *represents the state constraint set. If Problem 1 has a solution, it is expressed as follows:*

$$\tau(k) = \begin{bmatrix} 1 & 0 & \cdots & 0 \end{bmatrix} U^*(k) \tag{29}$$

where $U(k)$ *represents the predicted sequence:*

$$U^*(k) = \begin{bmatrix} \tau^*(k) \\ \tau^*(k+1) \\ \vdots \\ \tau^*(k+N-1) \end{bmatrix} \tag{30}$$

As illustrated in Figure 2, the inner loop velocity controller generates control forces, creating a closed-loop system. Through continuous iterations, the ROV tracks both position and velocity.

Theorem 1. *Linearizing the nonlinear system described by Equation (25) around the equilibrium point* $(0,0)$ *yields the following expression:*

$$e(k+1) = Ae(k) + B\tau(k) \tag{31}$$

where $A = \frac{\partial f}{\partial e}(0,0)$ *and* $B = \frac{\partial f}{\partial \tau}(0,0)$*. If Equation (31) is stable, a linear state feedback* $\tau = Ke$ *can be determined, ensuring* $A_K = A + BK$ *is asymptotically stable. For any such* K*, the following lemma holds:*

(1) If $Q^* = Q + K^T R K \in \mathbb{R}^{n\times n}$ *and* ε *is a constant with* $\varepsilon > 1$*, then the discrete Lyapunov equation*

$$A_k^T P A_k - P + \varepsilon Q^* = 0 \tag{32}$$

admits a unique positive definite solution P*.*

(2) For any constant $\alpha \in (0, \infty)$*, the neighborhood around the equilibrium point* Ω *is defined as follows:*

$$\Omega = \{e \in R^n | e^T P e < \alpha\} \tag{33}$$

For all $e \in \Omega$*, there exists* $\Omega \in X$ *and* $Ke \in U$*, such that the linear state feedback* $\tau = Ke$ *satisfies the control constraints within the Euclidean neighborhood. Simultaneously, the nonlinear system under the local linear feedback control law* $\tau = Ke$ *can be expressed as follows:*

$$e(k+1) = F(e(k), Ke(k)) \tag{34}$$

Under the influence of $\tau = Ke$*,* Ω *remains invariant. Theorem 1 has been proven in the literature [35].*

Based on the above groundwork, the feasibility and stability of the inner-loop control system are discussed.

Theorem 2. *For the nonlinear Equation (25), the controller* $\tau = Ke$ *is locally asymptotically stable, with the positive definite function* $E(x) = e^T P e$ *and the neighborhood* $\Omega = \{e \in X | e^T P e \leq \alpha\}$ *defined as the terminal control, penalty, and domain in the quasi-infinite time-domain NMPC.*

The open-loop optimization problem in Problem 1 has a feasible solution at $k = 0$*. In the absence of disturbances and model errors, the optimization problem remains feasible for any* $k > 0$*, ensuring closed-loop system stability.*

A detailed theoretical proof of the feasibility and stability of Theorem 2 can be found in the paper [28], from which the feasibility and stability of the inner-loop NMPC controller can be concluded.

3.2.2. ENMPC Inner Loop Controller

The onboard controller of ROVs has limited computing capability. The inner-loop controller employs online ENMPC for optimization, which reduces the computational load. The implementation procedure of the ENMPC is given in Algorithm 1. The controller only recalculates and updates the control input when specific triggering conditions are met. The tracking error of the inner-loop velocity controller for speed tracking is expressed as follows:

$$e = \sqrt{(u(k) - u_d(k))^2 + (v(k) - v_d(k))^2 + (w(k) - w_d(k))^2} \tag{35}$$

The actual velocity of the ROV is represented as $v = [u, v, w, q, r]^T$, while the desired velocity is expressed as $v_d = [u_d, v_d, w_d, q_d, r_d]^T$.

The event-triggering mechanism is formulated as follows:

$$e \geq \sigma \tag{36}$$

Here, σ represents the triggering threshold. The NMPC is recalculated only when the specified conditions are satisfied.

Remark 1. *The inner loop velocity controller can effectively reduce computational load by selecting appropriate event triggering conditions while sacrificing a minimal amount of control performance. However, a large trigger threshold may lead to instability, while a small threshold may not effectively reduce computational burden. It is important to find a suitable trigger threshold that balances stability and energy consumption.*

The stability and feasibility of event-triggering mechanisms are discussed in [26], while the selection criteria for the triggering threshold can be found in [28]. The event-triggering process is outlined in Table 1.

Algorithm 1 Algorithm Description

1: Initialize system's velocity and control input
2: Set prediction range N_P and control range N_C to be equal ($N_P = N_C$)
3: **while** True **do**
4: **if** Event-triggering mechanism Equation (36) is satisfied **then**
5: Solve the corresponding optimal control problem for Problem 1 and obtain the optimal control solution
6: **else**
7: Maintain the optimal control solution from the previous iteration.
8: **end if**
9: Update the state of the ROV in the control system by applying the first element, u, of the optimal solution
10: **end while**

Table 1. Parameters of the ROV.

Parameters	Value	Parameters	Value
m	4187.5 kg	$N_{\dot{r}}$	-3708 kg· m^2
W	41,079.38 kg	X_u	-3610 kg/s
B	41,079.38 kg	Y_v	-2463 kg/s
I_x	2038 kg· m^2	Z_w	-4567 kg/s
I_y	3587 kg· m^2	M_q	-5221 kg· m^2/(s ·rad)
I_z	3587 kg· m^2	N_r	-5842 kg· m^2/(s ·rad)
$X_{\dot{u}}$	-3261 kg	$X_{u\|u\|}$	-952 kg/m
$Y_{\dot{v}}$	-4664 kg	$Y_{v\|v\|}$	-2443 kg/m
$Z_{\dot{w}}$	-7472 kg	$Z_{w\|w\|}$	-530 kg/m
$K_{\dot{p}}$	-1664 kg· m^2	$M_{q\|q\|}$	-1876 kg· m^2/rad
$M_{\dot{q}}$	-4118 kg· m^2	$N_{r\|r\|}$	-2086 kg· m^2/rad

4. Simulation Analysis

In this section, simulations are conducted for the 3D trajectory tracking of DNN + NMPC due to the limitations in performing physical experiments. While physical testing is constrained by practical conditions at this stage, the trajectory tracking control using the proposed DNN + ENMPC approach was conducted on a laptop equipped with an Intel® Core™ i7-12700F CPU running at 2.1 GHz and 16 GB of RAM, Huashuo BNH-20230215DZH, ASUS, Shanghai, China. The computing power of the board is sufficient to meet the computational demands of the DNN + ENMPC algorithm. MATLAB simulations currently serve to validate the algorithm's performance. A comparison is made between the control algorithms DNN + ENMPC (which includes an event-triggered mechanism), ENMPC, NMPC, and DNN + NMPC to assess their effectiveness. The specific parameters of the ROV are provided in Table 1, as referenced in [36,37].

The thruster force allocation diagram is presented in Figure 7. As this study focuses on five-degree-of-freedom control, two vertical thrusters are excluded compared with the model in the aforementioned article. Nonetheless, in the above-mentioned reference, the actual thrust limit of the thruster is 4000 N. To make the simulation more realistic, this paper applies a more stringent constraint on the thruster's thrust. The thrust allocation formula is provided below, where τ_w denotes the thruster force as a 6×1 vector.

$$\tau_w = {B_t}^{-1} \times \tau \tag{37}$$

$$B_t = \begin{bmatrix} \sin(45^\circ) & \sin(45^\circ) & -\sin(45^\circ) & -\sin(45^\circ) & 0 & 0 \\ \cos(45^\circ) & -\cos(45^\circ) & \cos(45^\circ) & -\cos(45^\circ) & 0 & 0 \\ 0 & 0 & 0 & 0 & \frac{1}{2} & \frac{1}{2} \\ 1.79 * \sin(75.14^\circ) & 1.79 * \sin(75.14^\circ) & 1.79 * \sin(75.14^\circ) & 1.79 * \sin(75.14^\circ) & 0 & 0 \\ 0 & 0 & 0 & 0 & -\frac{3.1}{2} & \frac{3.1}{2} \end{bmatrix} \tag{38}$$

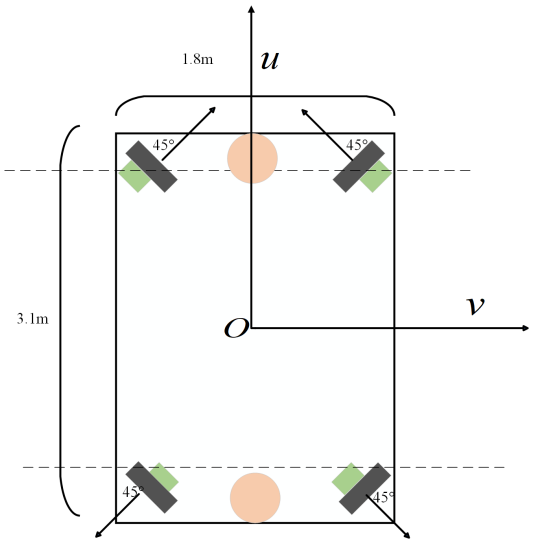

Figure 7. ROV thruster allocation diagram.

The thrust constraints specify that the horizontal thruster force is limited to an absolute value of 3600 N, while the vertical thruster force is capped at 2600 N. Therefore, the thrust constraints for the thrusters in this paper can be derived.During the simulation process, the thrust constraint of the thruster is represented by a black dashed line.

4.1. Analysis of 3D Trajectory Tracking Using DNN + NMPC

The DNN + NMPC approach is utilized to follow the specified trajectory, initiating from multiple starting points. The reference equation for the specified trajectory is given below:

$$\begin{cases} \xi_d = 5\sin(0.021t) \\ \eta_d = 5\cos(0.006t) \\ \zeta_d = -0.04t \end{cases} \tag{39}$$

DNN + NMPC-based 3D trajectory tracking with initial starting points $x_1 = [0, 4, 0, 0, 0]$, $x_2 = [1, 4, 0, 0, 0]$, $x_3 = [0, 4, -1, 0, 0]$, and $x_4 = [0, 4, 2, 0, 0]$, DNN + NMPC-based 3D trajectory tracking with initial velocities all set to 0, $v = [0, 0, 0, 0, 0]$.The simulation time is set to 400 s with a sampling interval of 0.5 s.

The inner loop controller has an equal prediction and control horizon, with $N_C = N_P = 5$. This setup strikes a balance between computational burden and control performance, providing sufficient future prediction capabilities while maintaining system stability and responsiveness. Control constraints are set to the physical maximum thrust of the thrusters to prevent thrust overflow. The simultaneous control constraints are designed as $u_{\max} = {B_t}^{-1}[3600, 3600, 3600, 2600, 2600]$ and $u_{\min} = {B_t}^{-1}[-3600, -3600, -3600, -2600, -2600]$.

Where $Q = diag\{300, 300, 300, 300, 300\}$ and $R = diag\{0.01, 0.01, 0.01, 0.01, 0.01\}$ are also defined. The choice of Q reflects the importance of minimizing state errors, with higher values prioritizing accurate control across the five degrees-of-freedom, while R is selected to allow flexibility in the control inputs, ensuring that the system can provide sufficient thrust response without being overly constrained by input costs.

The parameters of the cost function for the NMPC, including the terminal penalty matrix P, the infinite horizon control gain K, and the terminal set Ω, are determined following the procedure outlined in Section 3.2. The terminal penalty matrix P is designed to ensure system stability as it approaches the terminal state, effectively penalizing any deviation from the desired terminal configuration. The infinite horizon control gain K is selected to optimize long-term control performance, ensuring the system maintains stability while minimizing the cost beyond the prediction horizon. Lastly, the terminal set Ω is defined to guarantee that the system remains within a safe and feasible operating region at the end of the prediction horizon, providing the necessary conditions for the control law to regulate the system effectively.

From Figures 8 and 9, the proposed DNN + NMPC algorithm can quickly and effectively track the desired 3D trajectory from various initial states, demonstrating the effectiveness of the DNN + NMPC approach.

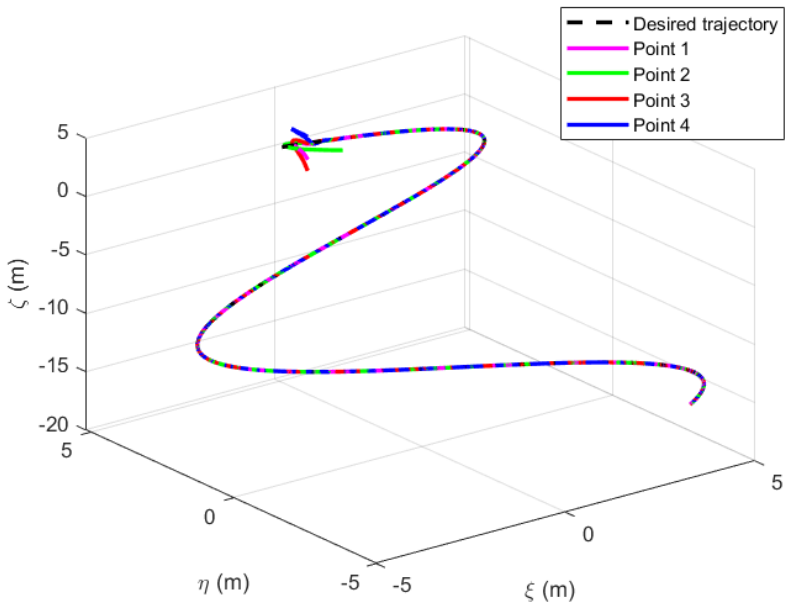

Figure 8. DNN + NMPC-based 3D trajectory tracking performance.

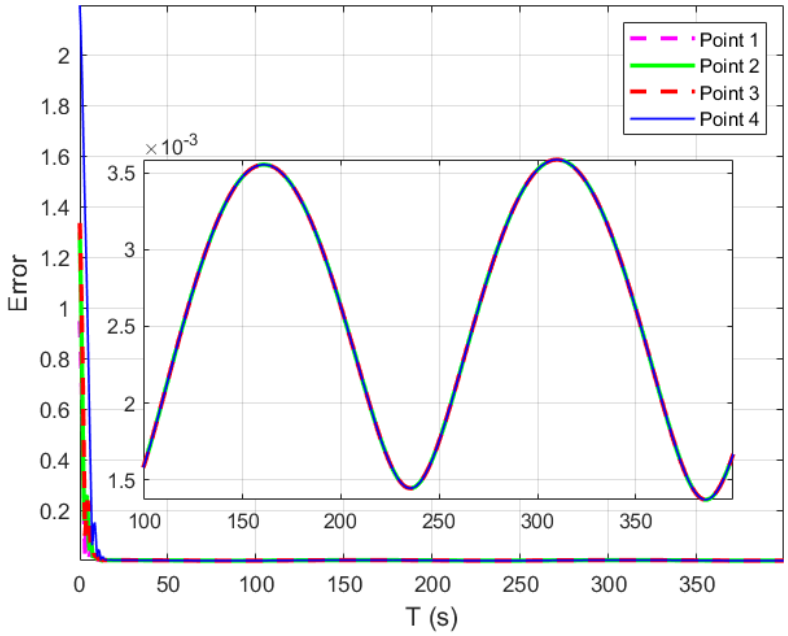

Figure 9. The position tracking errors based on DNN + NMPC.

4.2. Comparative Analysis of Four Control Strategies

This section analyzes the performance of NMPC, ENMPC, DNN + NMPC, and DNN + ENMPC in following 3D trajectories under three scenarios: no external disturbances, Gaussian noise, and ocean current disturbances. The goal is to verify the efficiency of the DNN + ENMPC control method. The control settings are as follows: The simulation time step is $T_s = 0.5$ s, with a total simulation time of 400 s. The initial position of the motion is $x = [-1, 7, 0, 0, 0]$, with the motion trajectory determined by Equation (40). To ensure a fair comparison of experimental results, we choose the same control parameters for NMPC, ENMPC, DNN + NMPC, and DNN + ENMPC. The control range and prediction horizon are the same, with $N_C = N_P = 5$. The weighting factors are designed as follows: $Q = diag\{300, 300, 300, 300, 300\}$ and $R = diag\{0.01, 0.01, 0.01, 0.01, 0.01\}$, terminal cost P, control gain K, and terminal domain Ω will be determined according to the steps described in Section 3.2. The control constraints are defined by $u_{\max} = {B_t}^{-1}[3600, 3600, 3600, 2600, 2600]$ and $u_{\min} = {B_t}^{-1}[-3600, -3600, -3600, -2600, -2600]$.

$$\begin{cases} \xi_d = 6\sin(0.02t) \\ \eta_d = 6\cos(0.02t) \\ \zeta_d = -0.04t \end{cases} \tag{40}$$

4.2.1. Three-Dimensional Trajectory Tracking without Disturbance

The DNN + ENMPC triggering threshold selection method is shown in Section 3.2.2. As illustrated in Figure 10, when the threshold is too small, the event triggering does not function effectively. As the threshold increases from greater than 0.006 to less than 0.01, the triggering frequency decreases. Beyond a threshold of 0.01, the triggering frequency basic stabilizes, but exceeding 0.016 destabilizes the ROV. Introducing the event-triggered mechanism, as depicted in Figure 11, impacts control precision to some extent, but the loss is minimal. Under disturbance-free conditions, a threshold of 0.01 is selected, where errors and triggering occurrences are minimized. The same method is applied under ocean currents and random disturbances; threshold selection is no longer discussed in ocean currents and random disturbances. Similarly, the threshold selection for ENMPC follows the same approach, with the threshold set to 0.012.

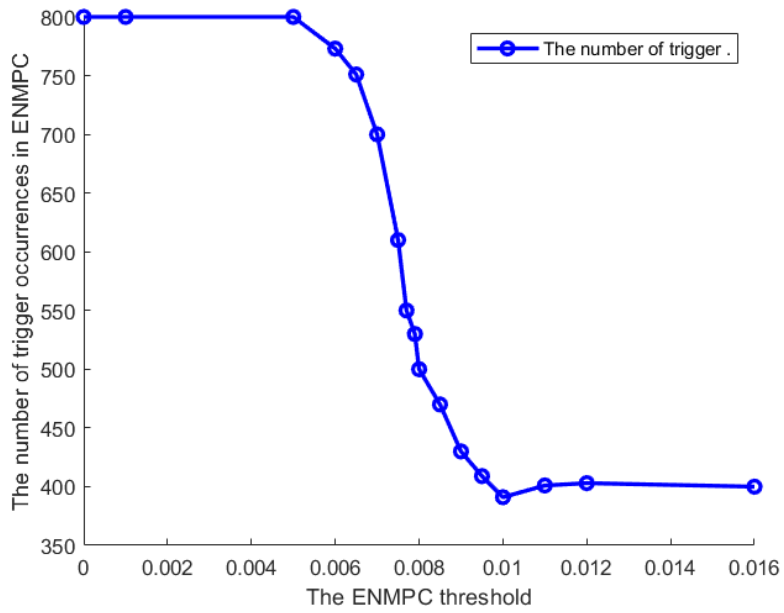

Figure 10. Illustrates the relationship between ENMPC thresholds and output occurrences.

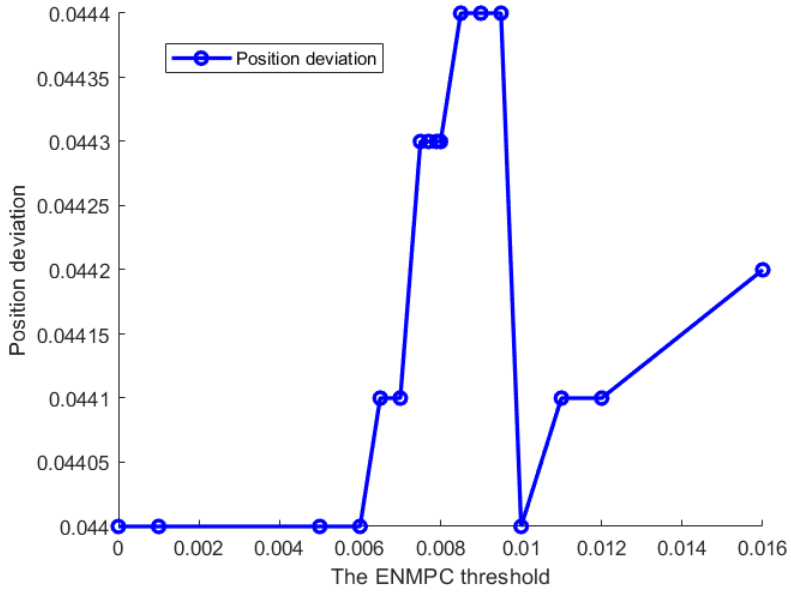

Figure 11. Shows the relationship between thresholds and position error.

As demonstrated in Figures 12 and 13, the NMPC trajectory tracking curve is shown in green, the ENMPC tracking curve is displayed in red, the DNN + NMPC tracking curve is presented in blue, and the DNN + ENMPC tracking curve is highlighted in pink. Without external disturbances, all four control algorithms achieve strong tracking performance. Additionally, as indicated in Figure 14, in disturbance-free conditions, the DNN + NMPC and DNN + ENMPC controllers outperform their single-loop counterparts, NMPC and ENMPC. Both ENMPC and DNN + ENMPC employ event-triggered control strategies, which introduce tracking error oscillations. However, the oscillations in DNN + ENMPC are significantly smaller compared with those in ENMPC, highlighting the improved stability and tracking performance of the DNN + ENMPC approach.

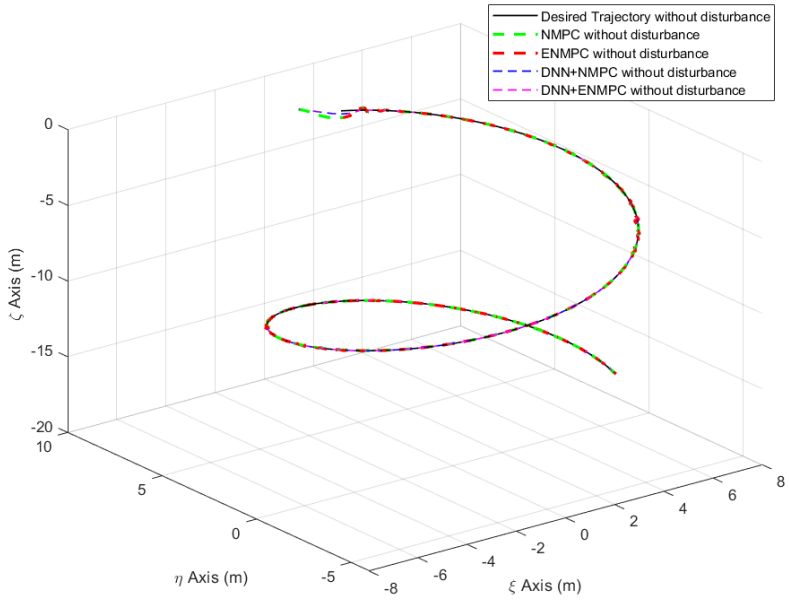

Figure 12. Three-dimensional trajectory tracking curve without external interference.

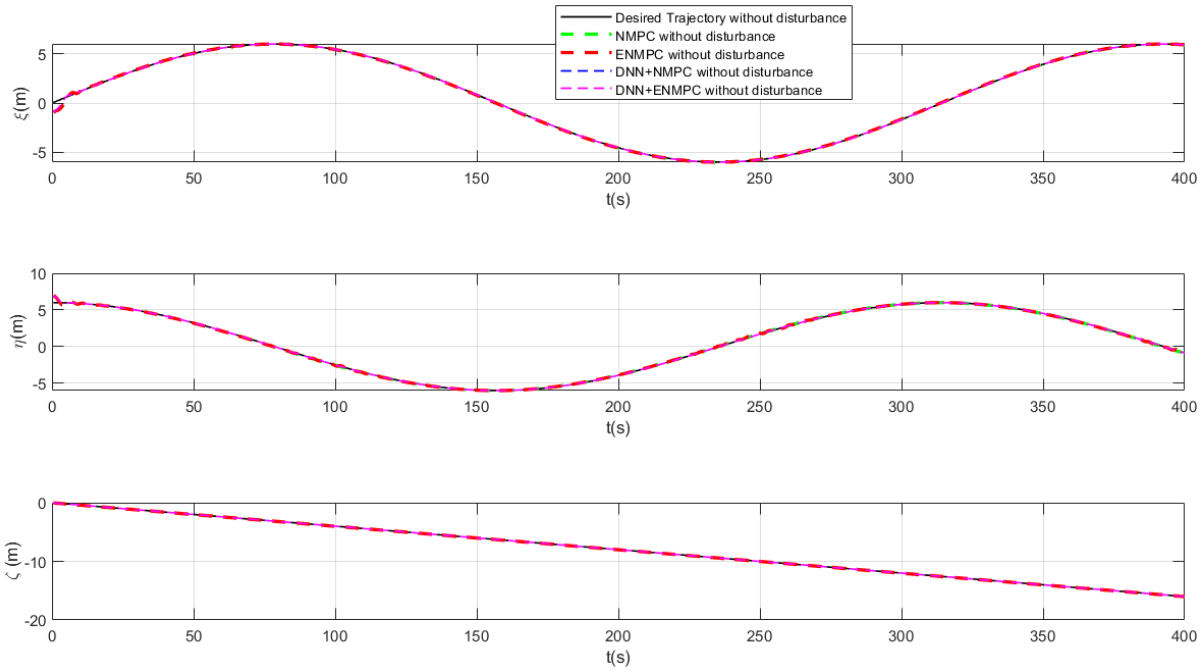

Figure 13. Position tracking performances without disturbances.

As indicated in Figure 15, all four control algorithms comply with the maximum thrust limitations on the thrusters. Nonetheless, the ENMPC strategy exhibits significant oscillations in thruster output due to its event-triggered mechanism. On the other hand, the DNN + ENMPC method effectively minimizes these oscillations, providing improved stability and smoother control performance when compared with ENMPC alone.

As shown in Figure 16, with a simulation interval of 0.5 s, the vertical axis represents the time interval between the current and previous trigger events, with a value of 0 if no trigger occurs. The trigger frequency of DNN + ENMPC is significantly lower than that of ENMPC, effectively reducing the computational load required by the controller. This demonstrates that the DNN + ENMPC approach introduces a more efficient event-triggered mechanism, leading to fewer trigger events and a reduced computational burden.

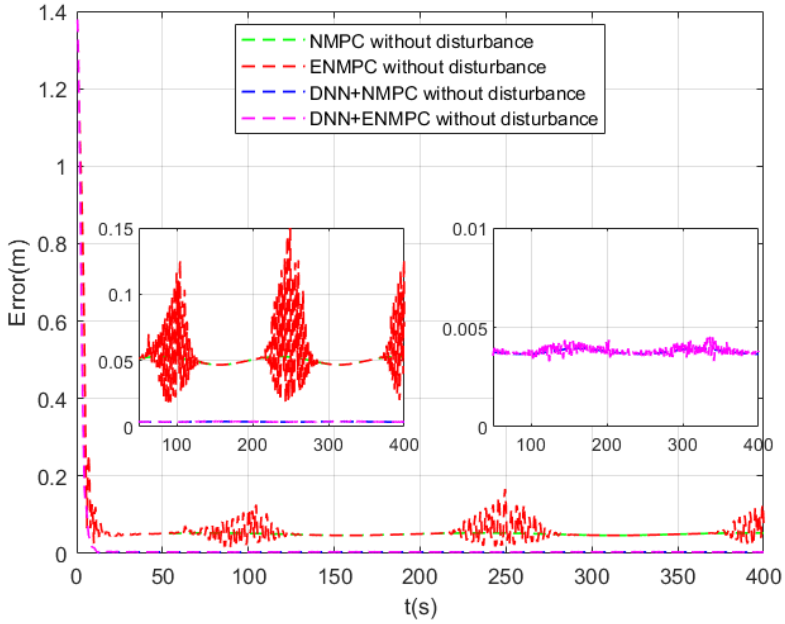

Figure 14. Position tracking errors in the absence of disturbances.

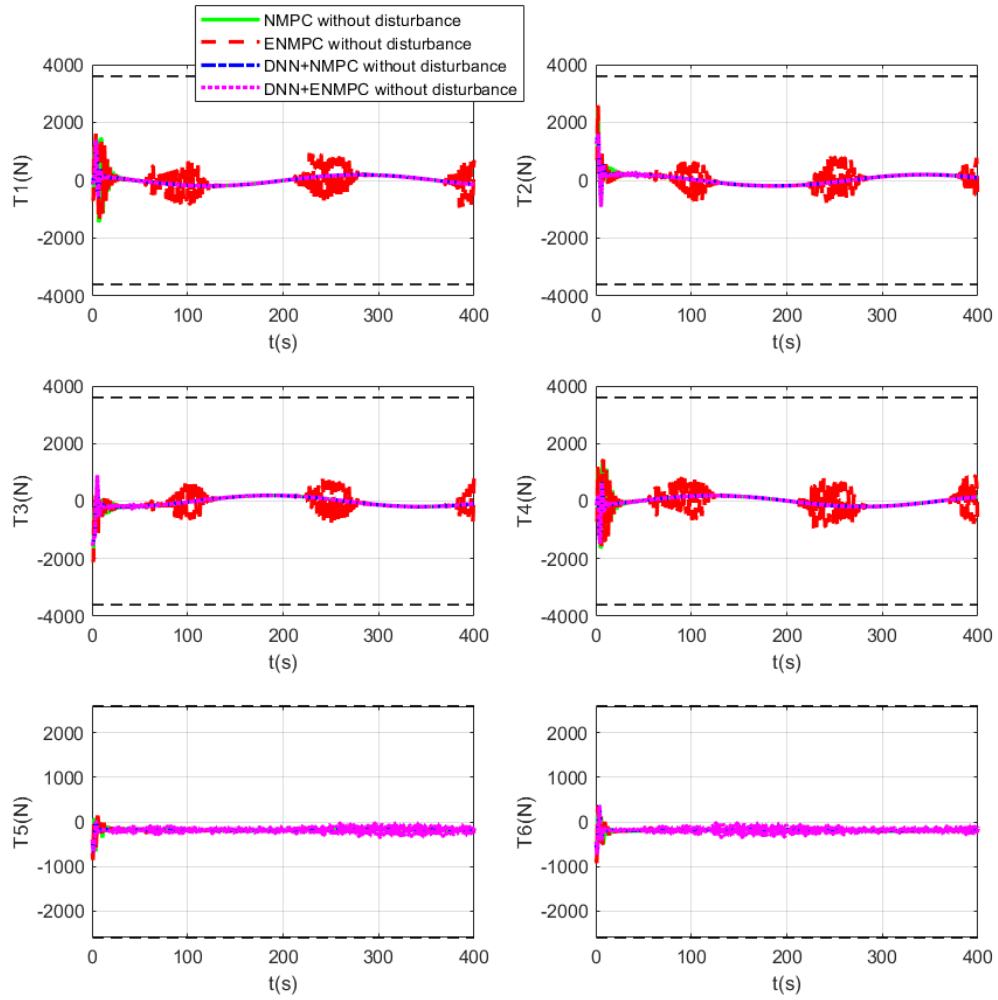

Figure 15. ROV control inputs in the absence of disturbances.

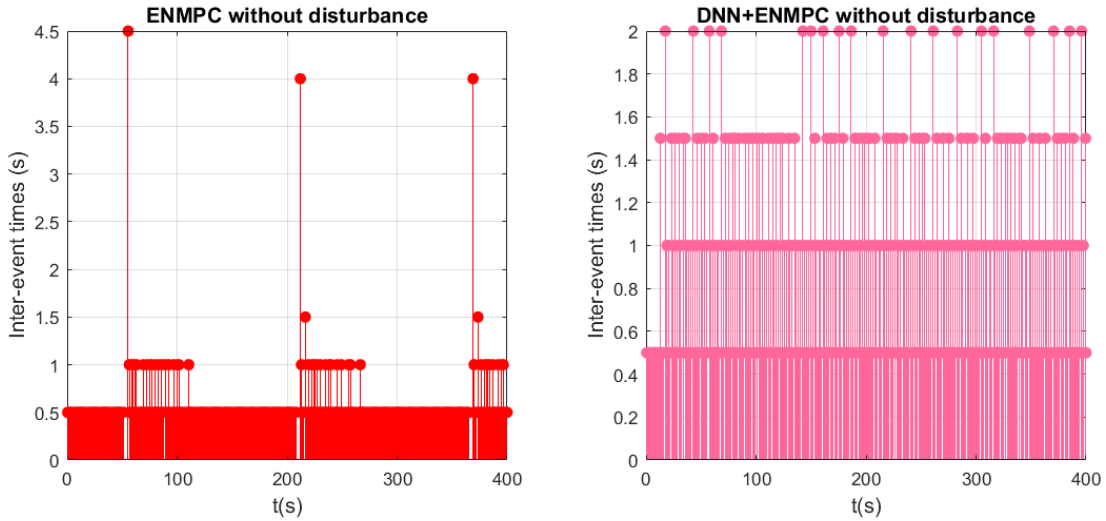

Figure 16. The triggering interval of DNN + ENMPC under conditions of no external interference.

As shown in Table 2, the average, maximum, and minimum errors of both DNN + NMPC and DNN + ENMPC are significantly lower than those of NMPC and ENMPC, demonstrating the superior tracking performance of the DNN-ENMPC controllers. Specifically, DNN + NMPC and DNN + ENMPC achieve an average error of 0.0038 m and 0.0039 m, respectively, compared with 0.05 m for NMPC and 0.0538 m for ENMPC. Furthermore, the RMSE values for DNN + NMPC and DNN + ENMPC are 0.0038 m and 0.0040 m, respectively, which are much smaller than the RMSE values for NMPC (0.0499 m) and ENMPC (0.0571 m). Additionally, DNN + ENMPC significantly reduces the trigger frequency, with only 479 iterations compared with 731 for ENMPC, highlighting the efficiency of the event-triggered mechanism in reducing computational load while maintaining high accuracy.

Table 2. Trajectory tracking comparison under no external disturbances.

Methods	Avg (m)	Max (m)	Min (m)	RMSE (m)	Iteration Counter
NMPC	0.05	0.0529	0.0467	0.0499	800
ENMPC	0.0538	0.1667	0.0183	0.0571	731
DNN + NMPC	0.0038	0.0039	0.0037	0.0038	800
DNN + ENMPC	0.0039	0.0045	0.0033	0.0040	479

4.2.2. Three-Dimensional Trajectory Tracking in the Presence of Gaussian Noise

This section presents a comparative analysis of trajectory tracking performance for NMPC, ENMPC, DNN + NMPC, and DNN + ENMPC in the presence of Gaussian noise. The Gaussian noise disturbance is defined in Equation (41), where $randn(1)$ denotes a Gaussian noise signal with zero mean and unit variance.

$$\begin{cases} \tau_{d\xi} = 40 * randn(1) \\ \tau_{d\eta} = 40 * randn(1) \\ \tau_{d\zeta} = 40 * randn(1) \end{cases} \tag{41}$$

As shown in Figures 17 and 18, all four control methods exhibit good trajectory tracking performance under Gaussian noise. As shown in Figure 19, under the influence of Gaussian noise, the controls performance of both DNN + NMPC and DNN + ENMPC surpass those of NMPC and ENMPC. Although the introduction of the event-triggered mechanism and Gaussian noise induces oscillations in both ENMPC and DNN + ENMPC, the oscillations in DNN + ENMPC are significantly smaller compared with ENMPC. This combination of reduced oscillations and improved noise handling capability highlights the superiority of DNN + ENMPC in three-dimensional trajectory tracking.

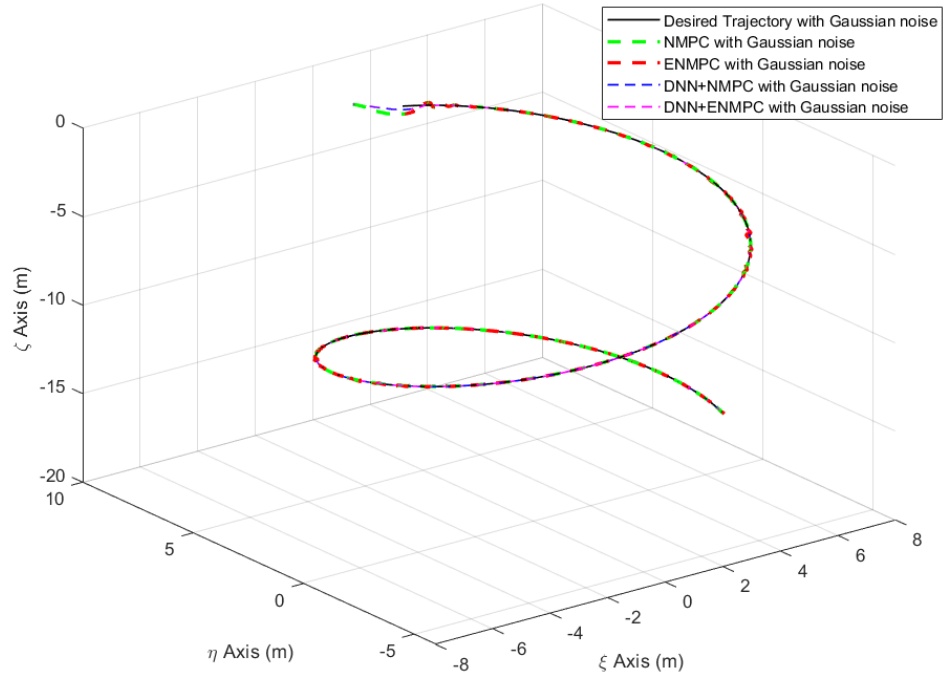

Figure 17. Performance analysis of 3D trajectory tracking under gaussian noise.

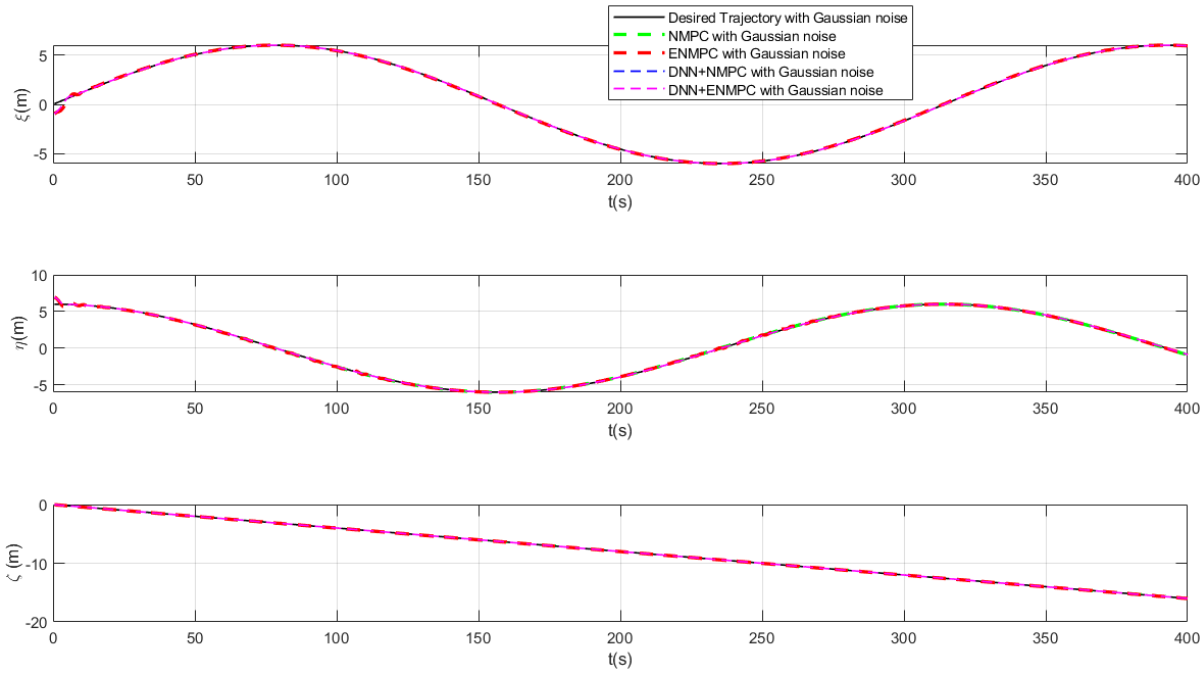

Figure 18. Position tracking performances with gaussian noise.

As shown in Figure 20, all four control strategies adhere to the maximum thrust constraints of the thrusters under Gaussian noise. However, the oscillations produced by ENMPC are significantly larger compared with those generated by DNN + ENMPC, further demonstrating the superior performance and stability of the DNN + ENMPC control strategy under noisy conditions.

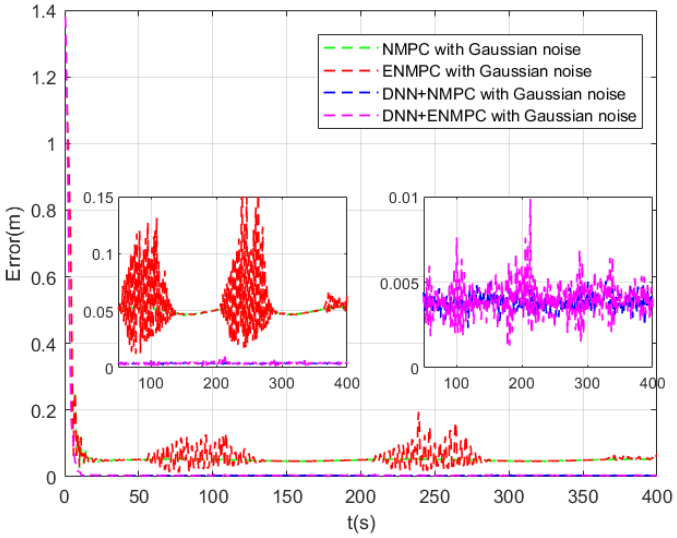

Figure 19. The position tracking errors with gaussian noise.

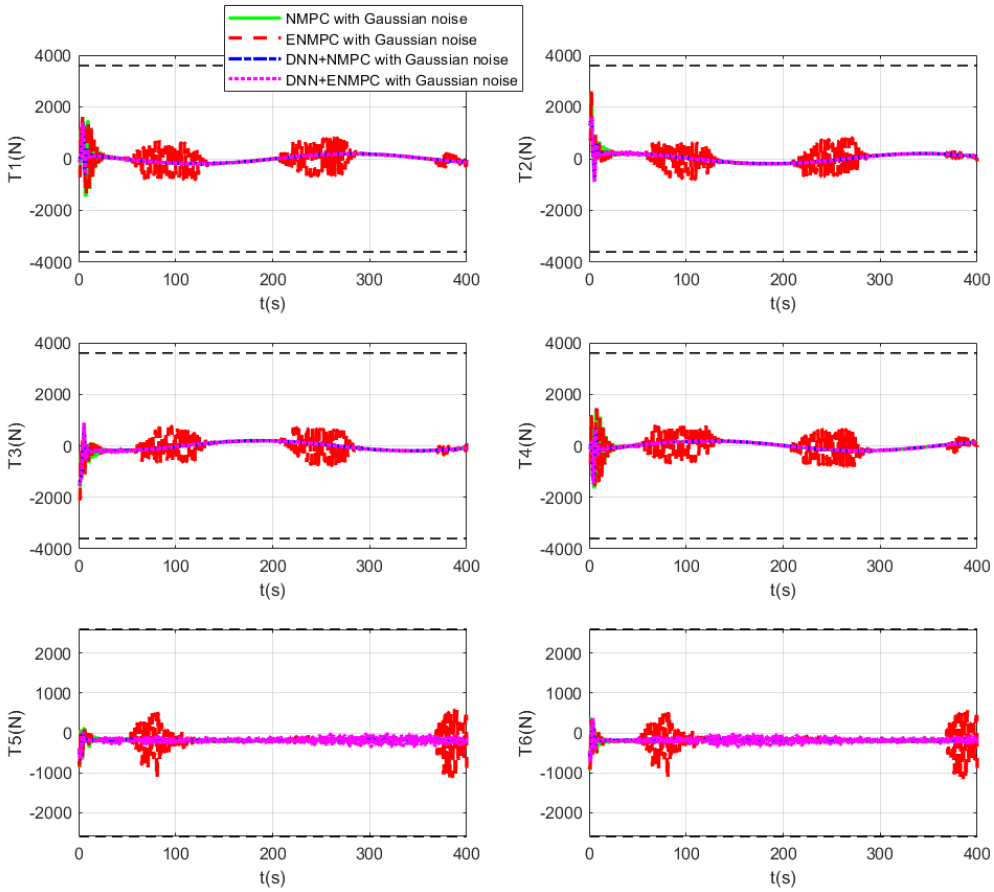

Figure 20. Control inputs of the ROV with gaussian noise.

As shown in Figure 21, under Gaussian noise, the trigger frequency of DNN + ENMPC is significantly lower than that of ENMPC. This demonstrates that DNN + ENMPC is more efficient in terms of reducing the computational load and requiring fewer trigger events while maintaining performance. This result further highlights the superiority of DNN + ENMPC in terms of both control stability and computational efficiency.

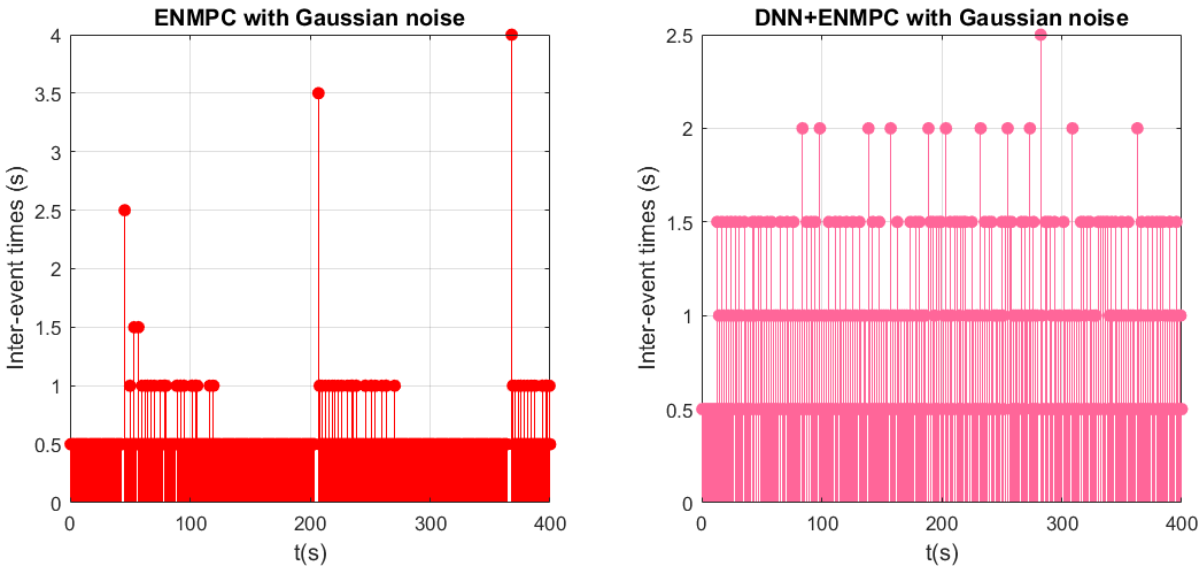

Figure 21. The triggering interval of DNN + ENMPC under conditions of gaussian noise.

As shown in Table 3, the DNN + ENMPC method significantly outperforms both ENMPC and NMPC in terms of average error, maximum error, and minimum error. Specifically, the DNN + ENMPC demonstrates a much smaller average error (0.0041 m) compared with NMPC (0.05 m) and ENMPC (0.0542 m). Additionally, the iteration count for DNN + ENMPC is notably lower than ENMPC, at 481 iterations compared with 732, highlighting the method's efficiency and reduced computational burden under Gaussian noise conditions.

Table 3. Trajectory tracking comparison under gaussian noise.

Methods	Avg (m)	Max (m)	Min (m)	RMSE (m)	Iteration Counter
NMPC	0.05	0.0539	0.0455	0.0499	800
ENMPC	0.0542	0.1946	0.0174	0.0579	732
DNN + NMPC	0.0039	0.0051	0.0023	0.0039	800
DNN + ENMPC	0.0041	0.0099	0.0012	0.0042	481

4.2.3. Three-Dimensional Trajectory Tracking in the Presence of Ocean Current Disturbances

This section introduces a comparative analysis of trajectory tracking performance among NMPC, ENMPC, DNN + NMPC, and DNN + ENMPC under external ocean current disturbances. The ocean current disturbance is represented by Equation (42).

$$\begin{cases} v_{fu} = 0.1\sin(0.03t) \\ v_{fv} = 0.1\sin(0.04t) \\ v_{fw} = 0.05\sin(0.02t) \end{cases} \tag{42}$$

As depicted in Figures 22 and 23, all three control algorithms demonstrate effective trajectory tracking under ocean current conditions. However, as shown in Figure 24, the tracking errors of DNN + NMPC and DNN + ENMPC are consistently smaller than those of NMPC and ENMPC for most of the time.

As shown in Figure 25, under ocean current disturbances, the oscillations in ENMPC are much larger than those in DNN + ENMPC, demonstrating the superior control performance of DNN + ENMPC under ocean current conditions.

As clearly shown in Figure 26, the trigger frequency of DNN + ENMPC is much lower than that of ENMPC, proving that DNN + ENMPC can more effectively reduce computational resource usage.

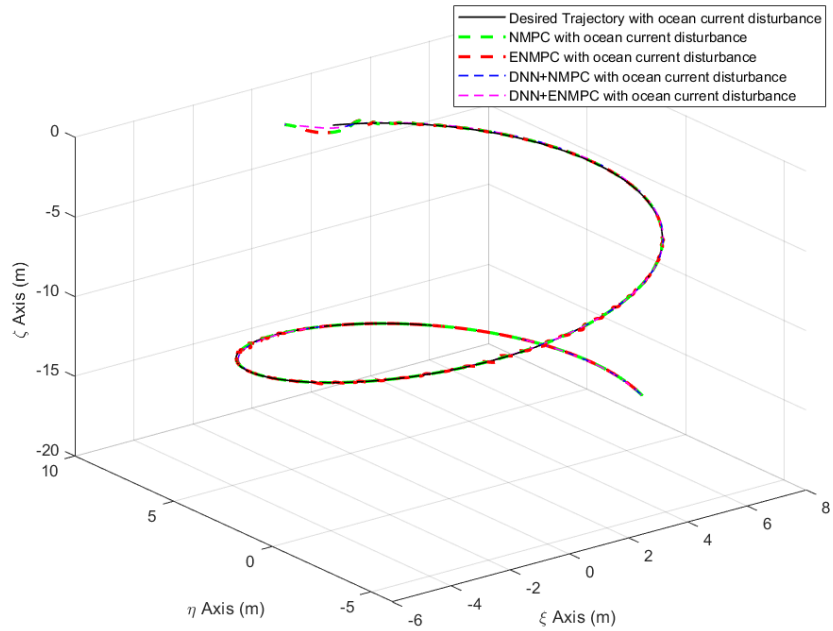

Figure 22. Three-dimensional trajectory tracking performance under ocean current disturbances.

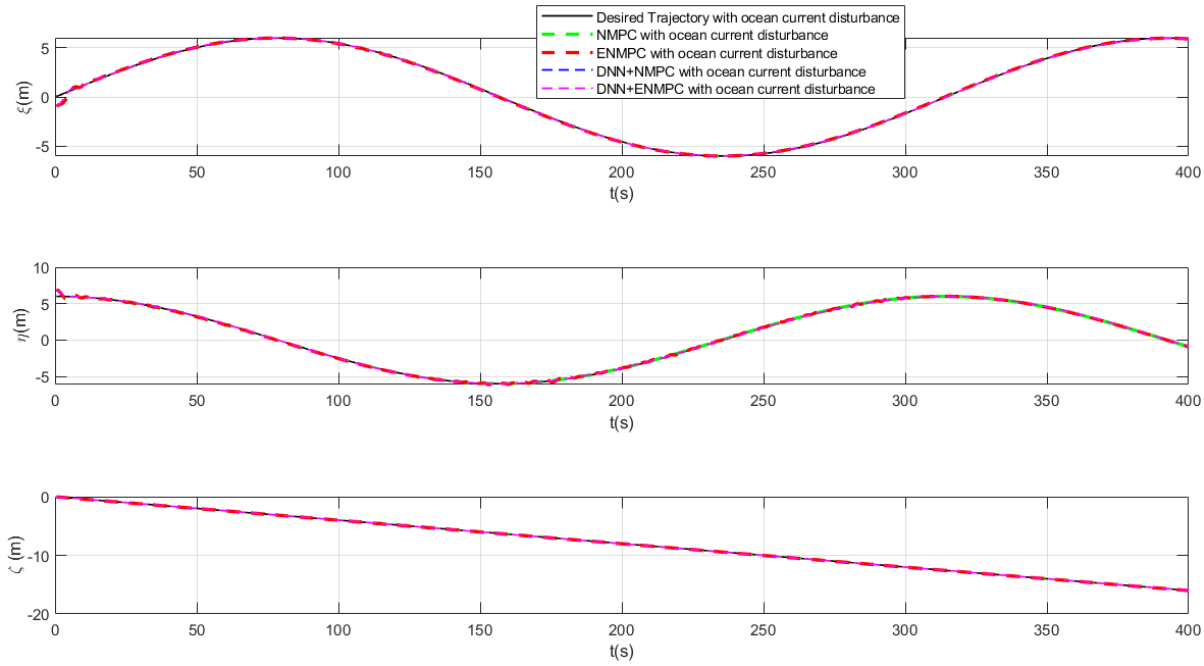

Figure 23. Position tracking performance under ocean current disturbances.

As clearly demonstrated in Table 4, the DNN + ENMPC method achieves a lower average tracking error and RMSE under ocean current conditions compared with both NMPC and ENMPC. This highlights the superior tracking control performance of DNN + ENMPC in ocean current conditions. Specifically, the average tracking error of DNN + ENMPC is smaller than that of NMPC and significantly lower than ENMPC, while the RMSE further underscores its robustness and precision. Moreover, the iteration count (or trigger frequency) for DNN + ENMPC is much lower than that of ENMPC, clearly illustrating its enhanced computational efficiency alongside improved control accuracy.

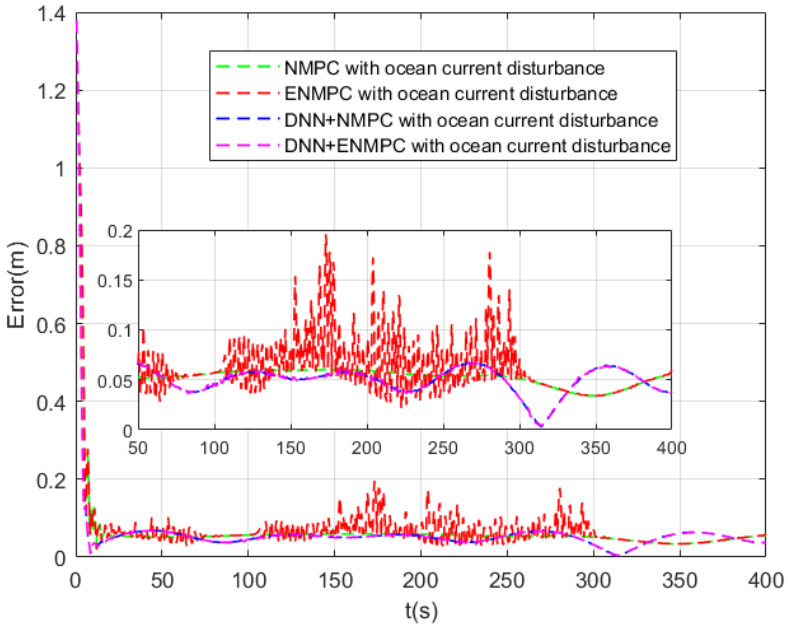

Figure 24. Position tracking errors under ocean current disturbances.

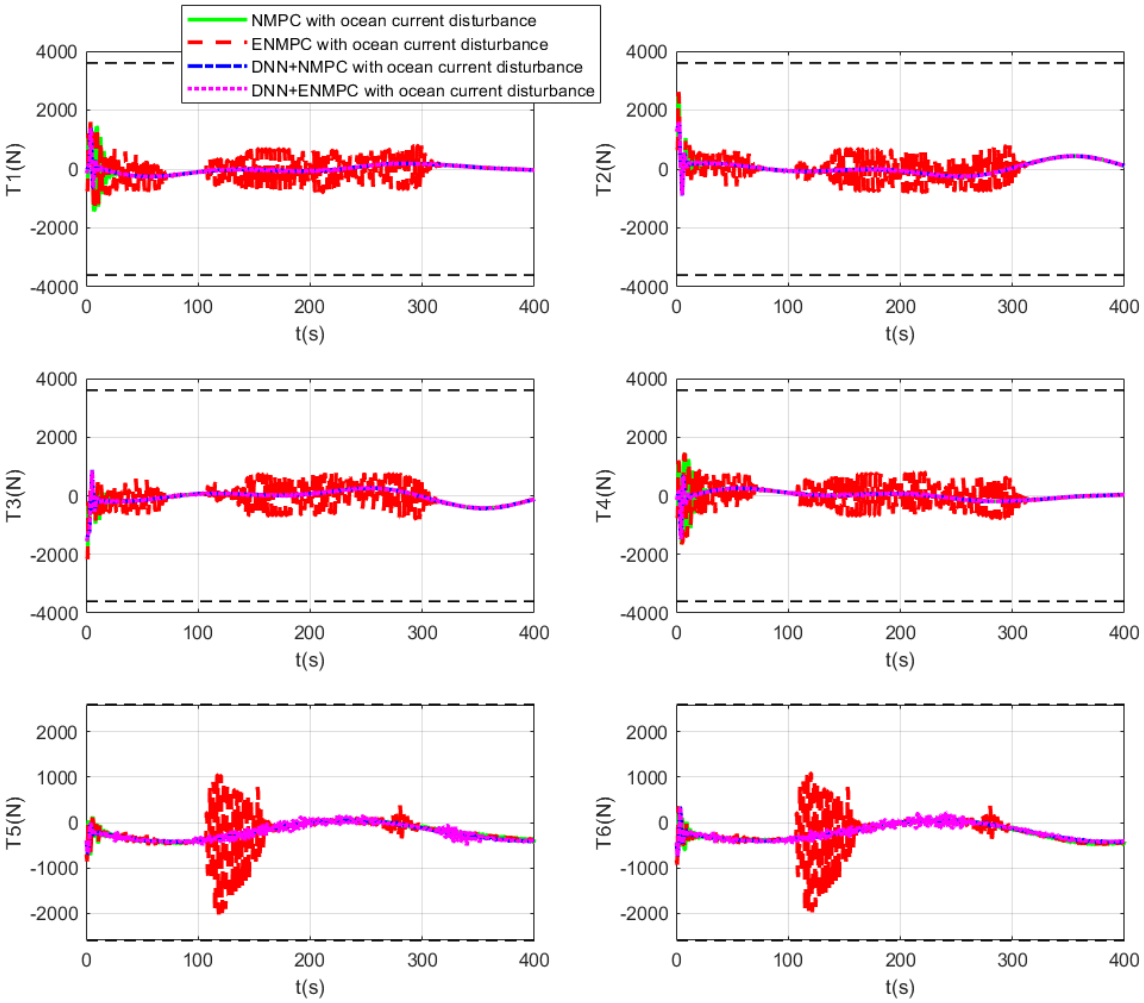

Figure 25. Control inputs of the ROV with ocean current disturbances.

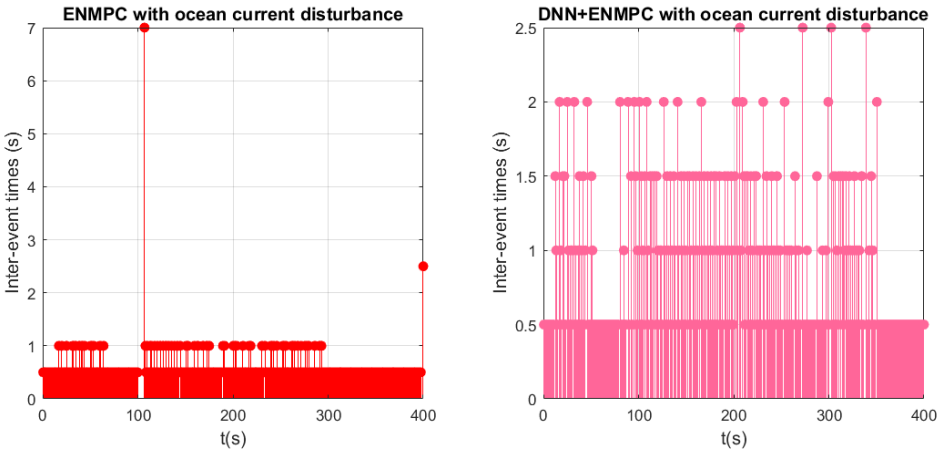

Figure 26. The triggering interval of DNN + ENMPC under conditions of ocean current disturbances.

Table 4. Trajectory tracking comparison under ocean current disturbances.

Methods	Avg (m)	Max (m)	Min (m)	RMSE (m)	Iteration Countert
NMPC	0.0524	0.0612	0.0336	0.0529	800
ENMPC	0.0627	0.1957	0.0218	0.0689	718
DNN + NMPC	0.0492	0.0671	0.0034	0.0509	800
DNN + ENMPC	0.0495	0.0676	0.0039	0.0512	546

4.2.4. Three-Dimensional Trajectory Tracking with the Center of Mass Shift

This section analyzes the nonlinearity of having the center of mass shift in the ROV. Adopting the approach in [38], the center of mass shift means that the ROV's center of mass is at $(x_g, 0, z_g)$ instead of $(0, 0, 0)$. This section analyzes the performance of the DNN + ENMPC strategy under two conditions, i.e., without the center of mass shift with $x_g = 0, z_g = 0$, and having the center of mass shift with $x_g = 0.05\,\text{m}$, $z_g = 0.1\,\text{m}$. According to the aforementioned literature [38], the inertia matrix is denoted as

$$M = \begin{bmatrix} m - X_{\dot{u}} & 0 & 0 & mz_g & 0 \\ 0 & m - Y_{\dot{v}} & 0 & 0 & mx_g \\ 0 & 0 & m - Z_{\dot{w}} & -mx_g & 0 \\ mz_g & 0 & -mx_g & J_y - M_{\dot{q}} & 0 \\ 0 & mx_g & 0 & 0 & J_z - N_{\dot{r}} \end{bmatrix},$$

and the restoring forces and moment vectors caused by gravity and buoyancy are represented as $g(\eta) = [0, 0, 0, g_4, 0]$, $g_4 = Z_g w \sin\theta + X_g W \cos\theta$. As shown in Figure 27 the 3D trajectory tracking errors for the DNN + ENMPC strategy under both having the center of mass shift and without the center of mass shift remain within the magnitude of less than 0.02 m. While the DNN + ENMPC strategy demonstrates good trajectory tracking capability when having the center of mass shift, its performance is worse compared with the case when there is no center of mass shift. As shown in Figure 28, under both the center of mass shift and without the center of mass shift conditions, the thruster forces satisfy the actual constraints. When having the center of mass shift condition, the vertical thrusters consistently require greater thrust to stabilize the pitch angle caused by the center of mass shift $X_g = 0.05$ m. As shown in Figure 29, the DNN + ENMPC strategy requires a relatively larger number of triggers when having the center of mass shift than the number of triggers when without the center of mass shift, i.e., 505 and 476, respectively. In conclusion, the DNN + ENMPC strategy demonstrates good control performance under both having the center of mass shift and without the center of mass shift conditions. However, research efforts are needed in the future to implement comprehensive theoretical and practical investigations of the nonlinearity of having the center of mass shit in the underwater robot control.

The simulation experiment results demonstrate that the dual-loop DNN + ENMPC strategy shows better three-dimensional trajectory tracking performance compared with the existing single-loop NMPC and ENMPC strategies, mitigates effectively the thruster oscillation phenomenon observed in the ENMPC strategy, and reduces computational resource usage more efficiently compared with ENMPC. Meanwhile, there are issues worthy to be further investigated, e.g., the influence of the nonlinearity of the system model.

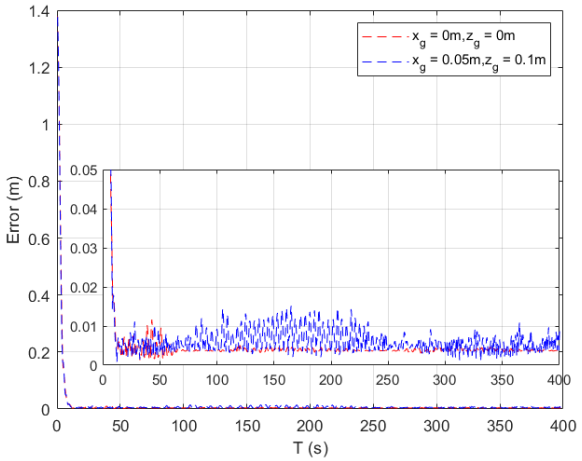

Figure 27. Comparison of 3D trajectory tracking errors of the DNN + ENMPC strategy under having the center of mass shift and without the center of mass shift conditions.

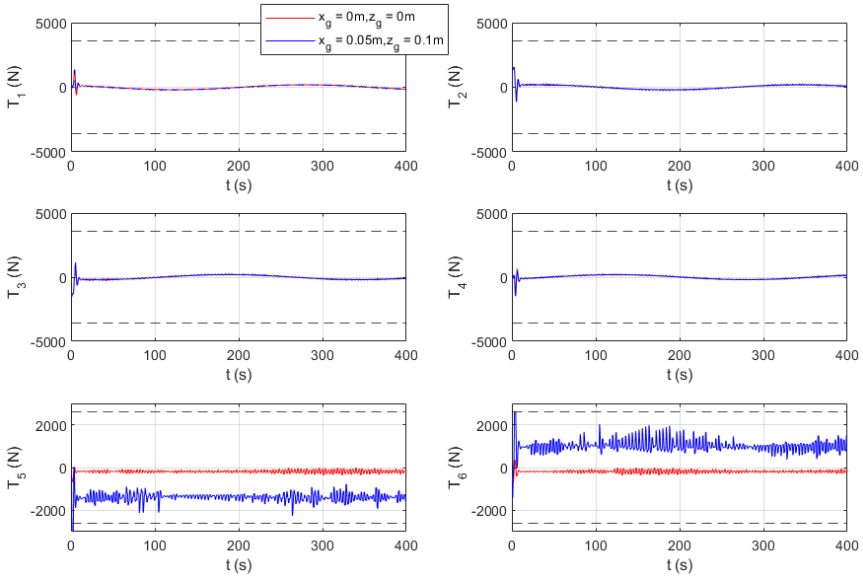

Figure 28. Comparison of thruster force response under having the center of mass shift and without the center of mass shift conditions.

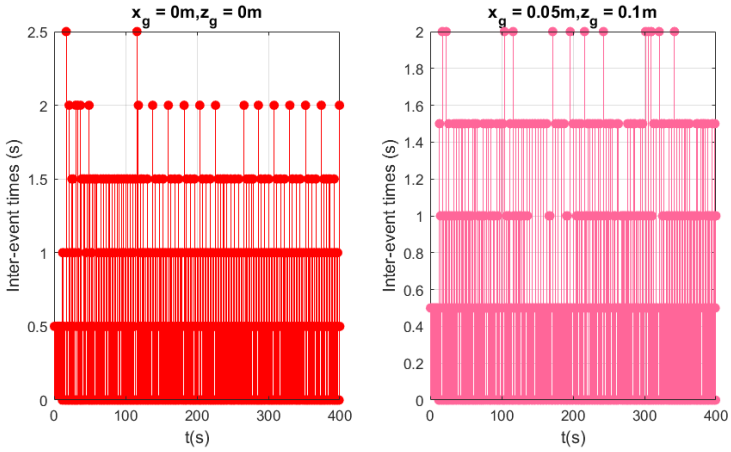

Figure 29. Comparison of inter-event times under having the center of mass shift and without the center of mass shift conditions.

5. Conclusions

This paper presents a dual closed-loop trajectory tracking strategy utilizing DNN in the outer loop and ENMPC in the inner loop for a remotely operated vehicle with a five-degrees-of-freedom model. The presented DNN + ENMPC strategy has demonstrated efficient trajectory tracking control performance in the numerical simulation experiments in terms of improving control performance and minimizing computational burden in comparison with the NMPC, ENMPC, and DNN + NMPC methods.

For further research, we will further investigate the nonlinearity of the ROV model for both achieving an efficient practical control performance and contributing to the nonlinear control method theories. The integration of extended Kalman filter and fuzzy logic with the proposed DNN + ENMPC is potentially to improve the robustness and adaptability to dynamic environments and will be explored in future work. Moreover, it is also worthy to update the physical ROV prototype and conduct practical experiments in the water pool or the lake to validate and optimize the trajectory tracking performance of the presented DNN + ENMPC strategy.

Author Contributions: Conceptualization and methodology, G.Y., W.L., and L.L.; Testing setpub, G.Y. and L.L.; testing conduction and data analysis, G.Y. and W.L.; writing—original draft preparation, all authors; writing—review and editing, all authors; funding acquisition, L.L. All authors have read and agreed to the published version of the manuscript.

Funding: This work is supported in part by the National Natural Science Foundation of China (project numbers 52102469, 61903304), in part by the Science and Technology Major Project of Guangxi under Grants of AB21196029, and in part by the Fundamental Research Funds for the Central Universities under project number 3102020HHZY030010.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data presented in this study are available through sending a request to 2022260761@mail.nwpu.edu.cn.

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

References

1. Teague, J.; Allen, M.J.; Scott, T.B. The potential of low-cost ROV for use in deep-sea mineral, ore prospecting and monitoring. *Ocean. Eng.* **2018**, *147*, 333–339. [CrossRef]
2. Osen, O.L.; Sandvik, R.-I.; Rogne, V.; Zhang, H. A novel low cost ROV for aquaculture application. In Proceedings of the OCEANS 2017, Anchorage, AK, USA, 18–21 September 2017; pp. 1–7. Available online: https://ieeexplore.ieee.org/abstract/document/8232180 (accessed on 25 March 2024).

3. Thanh, P.N.N.; Tam, P.M.; Anh, H.P.H. A new approach for three-dimensional trajectory tracking control of under-actuated AUVs with model uncertainties. *Ocean. Eng.* **2021**, *228*, 108951. [CrossRef]
4. Yan, Z.; Wang, M.; Xu, J. Integrated guidance and control strategy for homing of unmanned underwater vehicles. *J. Frankl. Inst.* **2019**, *356*, 3831–3848. [CrossRef]
5. Gayvoronskiy, S.A.; Ezangina, T.; Khozhaev, I. Providing a robust aperiodic transient process in motion control system unmanned underwater vehicle with interval parameters. *IFAC-PapersOnLine* **2018**, *51*, 220–225. [CrossRef]
6. Bingul, Z.; Gul, K. Intelligent-PID with PD feedforward trajectory tracking control of an autonomous underwater vehicle. *Machines* **2023**, *11*, 300. [CrossRef]
7. Rahmani, M.; Redkar, S. Enhanced Koopman operator-based robust data-driven control for 3 degree of freedom autonomous underwater vehicles: A novel approach. *Ocean. Eng.* **2024**, *307*, 118227. [CrossRef]
8. Zhou, H.; Wei, Z.; Zeng, Z.; Yu, C.; Yao, B.; Lian, L. Adaptive robust sliding mode control of autonomous underwater glider with input constraints for persistent virtual mooring. *Appl. Ocean. Res.* **2020**, *95*, 102027. [CrossRef]
9. Xia, G.; Zhang, Y.; Zhang, W.; Chen, X.; Yang, H. Dual closed-loop robust adaptive fast integral terminal sliding mode formation finite-time control for multi-underactuated AUV system in three dimensional space. *Ocean. Eng.* **2021**, *233*, 108903. [CrossRef]
10. Li, J.; Du, J.; Chen, C.P. Command-filtered robust adaptive NN control with the prescribed performance for the 3D trajectory tracking of underactuated AUVs. *IEEE Trans. Neural Netw. Learn. Syst.* **2021**, *33*, 6545–6557. [CrossRef]
11. Mayne, D.Q. Model predictive control: Recent developments and future promise. *Automatica* **2014**, *50*, 2967–2986. [CrossRef]
12. Zhang, Y.; Liu, X.; Luo, M.; Yang, C. MPC-based 3D trajectory tracking for an autonomous underwater vehicle with constraints in complex ocean environments. *Ocean. Eng.* **2019**,*189*, 106309. [CrossRef]
13. Liu, W.; Xu, J.; Li, L.; Zhang, K.; Zhang, H. Adaptive model predictive control for underwater manipulators using Gaussian process regression. *J. Mar. Sci. Eng.* **2023**, *11*, 1641. [CrossRef]
14. Wang, T.; Gao, H.; Qiu, J. A combined adaptive neural network and nonlinear model predictive control for multirate networked industrial process control. *IEEE Trans. Neural Netw. Learn. Syst.* **2015**, *27*, 416–425. [CrossRef] [PubMed]
15. Ellis, M.; Durand, H.; Christofides, P.D. A tutorial review of economic model predictive control methods. *J. Process. Control.* **2014**, *24*, 1156–1178. [CrossRef]
16. Qin, Y.; Liu, Z.FXESO based FNMPC path following control for underactuated surface vessels with roll stabilisation. *Ocean. Eng.* **2023**, *280*, 114855. [CrossRef]
17. Yan, Z.; Gong, P.; Zhang, W.; Wu, W. Model predictive control of autonomous underwater vehicles for trajectory tracking with external disturbances. *Ocean. Eng.* **2020**, *217*, 107884. [CrossRef]
18. Yan, Z.; Yan, J.; Cai, S.; Yu, Y.; Wu, Y. Robust MPC-based trajectory tracking of autonomous underwater vehicles with model uncertainty. *Ocean. Eng.* **2023**, *286*, 115617. [CrossRef]
19. Long, C.; Hu, M.; Qin, X.; Bian, Y. Hierarchical trajectory tracking control for ROVs subject to disturbances and parametric uncertainties. *Ocean. Eng.* **2022**, *266*, 112733. [CrossRef]
20. Dörfler, F.; Coulson, J.; Markovsky, I. Bridging direct and indirect data-driven control formulations via regularizations and relaxations. *IEEE Trans. Autom. Control.* **2022**, *68*, 883–897. [CrossRef]
21. Liu, H.; Meng, B.; Tian, X. Finite-time prescribed performance trajectory tracking control for underactuated autonomous underwater vehicles based on a tan-type barrier Lyapunov function. *IEEE Access* **2022**, *10*, 53664–53675. [CrossRef]
22. Liu, H.; Zhuo, J.; Tian, X.; Mai, Q. Finite-time self-structuring neural network trajectory tracking control of underactuated autonomous underwater vehicles. *Ocean. Eng.* **2023**, *268*, 113450. [CrossRef]
23. Yang, M.; Sheng, Z.; Yin, G.; Wang, H. A recurrent neural network based fuzzy sliding mode control for 4-DOF ROV movements. *Ocean. Eng.* **2022**, *256*, 111509. [CrossRef]
24. Guo, L.; Liu, W.; Li, L.; Lou, Y.; Wang, X.; Liu, Z. Neural network non-singular terminal sliding mode control for target tracking of underactuated underwater robots with prescribed performance. *J. Mar. Sci. Eng.* **2022**, *10*, 252. [CrossRef]
25. Astrom, K.J.; Bernhardsson, B.M. Comparison of Riemann and Lebesgue sampling for first order stochastic systems. In Proceedings of the 41st IEEE Conference on Decision and Control, Las Vegas, NV, USA, 10–13 December 2002; pp. 2011–2016. Available online: https://ieeexplore.ieee.org/abstract/document/1184824 (accessed on 13 May 2024).
26. Li, H.; Shi, Y. Event-triggered robust model predictive control of continuous-time nonlinear systems. *Automatica* **2014**, *50*, 1507–1513. [CrossRef]
27. Xu, J.; Cui, Y.; Yan, Z.; Huang, F.; Du, X.; Wu, D. Event-triggered adaptive target tracking control for an underactuated autonomous underwater vehicle with actuator faults. *J. Frankl. Inst.* **2023**, *360*, 2867–2892. [CrossRef]
28. Zhang, W.; Wang, Q.; Wu, W.; Du, X.; Zhang, Y.; Han, P. Event-trigger NMPC for 3D trajectory tracking of UUV with external disturbances. *Ocean. Eng.* **2023**, *283*, 115050. [CrossRef]
29. Li, S.; Zhu, Y.; Bai, J.; Guo, G. Dynamic obstacle avoidance of unmanned ship based on event-triggered adaptive nonlinear model predictive control. *Ocean. Eng.* **2023**, *286*, 115626. [CrossRef]
30. Liu, C.; Hu, Q.; Wang, X.; Yin, J. Event-triggered-based nonlinear model predictive control for trajectory tracking of underactuated ship with multi-obstacle avoidance. *Ocean. Eng.* **2022**, *253*, 111278. [CrossRef]
31. Wu, W.; Zhang, W.; Du, X.; Li, Z.; Wang, Q. Homing tracking control of autonomous underwater vehicle based on adaptive integral event-triggered nonlinear model predictive control. *Ocean. Eng.* **2023**, *277*, 114243. [CrossRef]

32. Chen, H.; Tang, G.; Wang, S.; Guo, W.; Huang, H. Adaptive fixed-time backstepping control for three-dimensional trajectory tracking of underactuated autonomous underwater vehicles. *Ocean. Eng.* **2023**, *275*, 114109. [CrossRef]
33. Do, K.D.; Pan, J. *Control of Ships and Underwater Vehicles: Design for Underactuated and Nonlinear Marine Systems*; Springer Science & Business Media: New York, NY, USA, 2009.
34. Wiig, M.S.; Pettersen, K.Y.; Krogstad, T.R.A 3D reactive collision avoidance algorithm for underactuated underwater vehicles. *J. Field Robot.* **2020**, *37*, 1094–1122. [CrossRef]
35. Chen, H.; Allgöwer, F. A quasi-infinite horizon nonlinear model predictive control scheme with guaranteed stability. *Automatica* **1998**, *34*, 1205–1217. [CrossRef]
36. Liu, W.; Guo, L.; Li, L.; Xu, J.; Yang, G.Fractional Active Disturbance Rejection Positioning and Docking Control of Remotely Operated Vehicles: Analysis and Experimental Validation. *Fractal Fract.* **2024**, *8*, 354. [CrossRef]
37. Fan, S. Hydrodynamic Test and Motion Control Technology Research of Deep-Sea Operation ROV. Ph.D. Dissertation, Shanghai Jiao Tong University, Shanghai, China, 2013.
38. Herman, P. Nonlinear Trajectory Tracking Controller for Underwater Vehicles with Shifted Center of Mass Model. *Appl. Sci.* **2024**, *14*, 5376. [CrossRef]