sensors

MDPI

Article

USV Trajectory Tracking Control Based on Receding Horizon Reinforcement Learning

Yinghan Wen [1], Yuepeng Chen [1] and Xuan Guo [2,*]

[1] School of Automation, Wuhan University of Technology, Wuhan 430070, China; 261688@whut.edu.cn (Y.W.); chenyuepengneu@163.com (Y.C.)
[2] School of Information Engineering, Wuhan University of Technology, Wuhan 430070, China
* Correspondence: guoxuan@whut.edu.cn

Abstract: We present a novel approach for achieving high-precision trajectory tracking control in an unmanned surface vehicle (USV) through utilization of receding horizon reinforcement learning (RHRL). The control architecture for the USV involves a composite of feedforward and feedback components. The feedforward control component is derived directly from the curvature of the reference path and the dynamic model. Feedback control is acquired through application of the RHRL algorithm, effectively addressing the problem of achieving optimal tracking control. The methodology introduced in this paper synergizes with the rolling time domain optimization mechanism, converting the perpetual time domain optimal control predicament into a succession of finite time domain control problems amenable to resolution. In contrast to Lyapunov model predictive control (LMPC) and sliding mode control (SMC), our proposed method employs the RHRL controller, which yields an explicit state feedback control law. This characteristic endows the controller with the dual capabilities of direct offline and online learning deployment. Within each prediction time domain, we employ a time-independent executive–evaluator network structure to glean insights into the optimal value function and control strategy. Furthermore, we substantiate the convergence of the RHRL algorithm in each prediction time domain through rigorous theoretical proof, with concurrent analysis to verify the stability of the closed-loop system. To conclude, USV trajectory control tests are carried out within a simulated environment.

Keywords: unmanned surface vehicle; receding horizon reinforcement learning; trajectory tracking; executive–evaluator

Citation: Wen, Y.; Chen, Y.; Guo, X. USV Trajectory Tracking Control Based on Receding Horizon Reinforcement Learning. *Sensors* **2024**, *24*, 2771. https://doi.org/10.3390/s24092771

Academic Editor: Sergio Toral Marín

Received: 12 March 2024
Revised: 22 April 2024
Accepted: 24 April 2024
Published: 26 April 2024

1. Introduction

A USV inherently constitutes a complex nonlinear system, being subject to disturbances and influences from the environment during navigation. Consequently, enhancing the path-tracking accuracy of unmanned ship motion control is a pressing concern.

At present, common methods for achieving such control include the PID [1,2], which is the most widely used, feedback control [3,4], fuzzy control [5,6], module predictive control (MPC) [7,8], and reinforcement learning (RL)-based control [9,10] methods. Of the aforementioned approaches, the PID control method stands out for its advantages. Notably, it eliminates the necessity for modeling the unmanned ship, rendering it a robust and easily implementable controller. However, a challenge lies in ensuring the optimality of specific performance indices. While the fuzzy controller exhibits the capability to deduce and generate expert behavior, its application is challenged by the intricacies of crafting fuzzy rules that primarily arise from the complexity inherent in the navigation environment.

The feedback controller, in its typical operation, computes heading and lateral deviations by analyzing the geometric relationship between the USV and the desired path. Based on this, it directly determines the steering wheel angle for precise steering control. The methods used for tracking, which involve deriving the correlation between the selected

path anchor point and the USV position, are the single-point tracking method, pre-sight distance method, and the Stanley method. Both the single-point tracking method [11] and pre-viewing distance method [12,13] offer the advantages of simplicity in algorithms and ease of implementation. However, a notable consideration lies in the fact that the selection of pre-viewing distance is contingent upon the experiential judgment of designers. The Stanley method, initially introduced by Stanford University for an unmanned vehicle fleet, is well suited for lower vehicle speeds. It necessitates a continuous curvature in the reference trajectory for optimal implementation.

A plethora of research findings have emerged concerning the application of MPC in vehicle motion control, as documented in the literature [14–17]. Of the achievements in these cited works, Falcone et al. [15] introduced an MPC motion controller grounded in the continuous linearization model, and their simulation results underscore the efficacy of the continuous linearization MPC design approach in minimizing computational costs. Carvalho et al. [17] studied an algorithm for local path planning using locally linearized MPC, carrying out linearization and convex approximation of nonlinear obstacle avoidance boundaries. Liniger et al. [18] proposed a lateral motion method of model predictive controlling control (MPCC). Using this method, the lateral deviation is calculated by estimating the position of the projection point, which reduces the computational complexity to a certain extent. Ostafew et al. [19] adopted Gaussian process regression to build a nonparametric model of a mobile robot. In the realm of unmanned surface vehicles, the trajectory tracking controller, employing the MPC method, typically necessitates real-time numerical calculations for solving an open-loop control sequence. The performance of this approach can be influenced by the precision of the model in addition to the unavoidable challenge of managing the complexity inherent in online calculations. Collectively, the current control strategies have various limitations characterized by suboptimal tracking accuracy and constrained computational efficiency.

In recent years, approximate dynamic programming (ADP) as well as reinforcement learning (RL) have experienced widespread adoption in the design of robot decision and control algorithms, thanks to their remarkable efficiency in solving optimization problems and adaptive learning capabilities [20,21]. Yang [22] developed a learning method which is based on PID control for the tracking control of vehicles. Aiming at optimizing the tracking deviation of robots, the DHP algorithm was employed for real-time adjustment of PID parameters, enhancing path-tracking accuracy. Gong et al. [23] designed a finite-time dynamic positioning controller for surface vessels. Shen et al. [24] introduced an innovative LMPC framework aiming to enhance trajectory tracking performance. Jiang et al. [25] also proposed sliding mode control to improve the tracking performance of USVs.

Recent advancements include noteworthy works employing deep learning and deep reinforcement learning to design controllers based on image or state information, facilitating trajectory control for USVs [26–28]. A key advantage of this approach lies in leveraging deep networks to enhance the feature representation capabilities of both reinforcement learning and supervised learning. Notably, the training process is entirely data driven, eliminating the need for dynamic model information. However, it has the following disadvantages:

(1) Due to the inherent complexity of deep networks, application of this method is limited to offline training control strategies for online deployment. Moreover, its control performance is susceptible to the influence of factors such as the quantity and distribution of training samples.
(2) In the context of deep network learning, the analysis of theoretical characteristics, such as convergence and robustness, remains a crucial and challenging issue for the academic community to address.

Motivated by the challenges outlined above, we propose a RHRL-based control method, aiming at achieving high-precision lateral control for USVs. The initial step involves constructing a dynamic deviation model for a USV. The steering control of such vehicles comprises two parts, which are feedforward and feedback. Feedforward control is derived directly from the curvature and deviation model for the reference path. In parallel,

the establishment of feedback control is achieved by addressing the problem of optimal tracking through application of the RHRL algorithm proposed in this paper. Diverging from conventional optimal control methods rooted in reinforcement learning, RHRL employs a rolling horizon optimization mechanism. This transformation converts infinite time domain optimal control problems into a sequence of finite time domain heuristic dynamic programming problems for resolution. In contrast to the MPC method for unwinding the loop control sequence, the strategy learned by this method is an explicit state feedback control law, which is amenable to offline direct deployment and online learning. Furthermore, in Section 3, the convergence and stability of the closed-loop associated with the proposed RHRL algorithm are theoretically analyzed within each prediction time domain. Finally, simulation and comparative experiments for USV trajectory control using the RHRL algorithm are conducted. Through simulation tests, the control performance is found to be comparable to that of LMPC, with notable advantages in terms of computational efficiency, lower sample complexity, and higher learning efficiency. To verify the algorithm's robustness and anti-interference capabilities, simulation incorporating disturbances are also conducted.

The remaining sections of this manuscript are arranged as follows. In Section 2, a dynamic model of a USV is built. Then, a USV trajectory control algorithm based on RHRL is proposed and shown to be stable. In Section 3, the simulation and comparison experiments are carried out, and disturbances are added. Section 4 contains the conclusions.

2. Materials and Methods

2.1. Modeling

In contemporary vehicle modeling, the utilization of three degrees of freedom (DOF) and six DOF predominates. However, considering the environment of the USV investigated in this study, which navigates on the sea surface, we opt for three degrees of freedom in the modeling process to avoid unnecessary complexity.

In the process of establishing dynamical equations, a crucial decision lies in selecting the coordinate system for their formulation. Direct application of Newton's laws of motion necessitates the expansion of equations in an inertial coordinate system. Nevertheless, various considerations compel us to derive the dynamic equations in a satellite coordinate system. One such reason is to establish dynamic equations that are direction independent. Additionally, employing the satellite coordinate system facilitates the direct assignment of forces and control moments. However, this would result in the current frame of reference not being an inertial frame of reference. Hence, to account for the non-inertial reference frame, Coriolis and centripetal forces are artificially introduced. This allows us to derive the remaining dynamics as if they were in an inertial reference frame.

The USV under investigation features a catamaran-like structure, incorporating two fixed propellers positioned at the extremities of each hull. In Figure 1, variables U_1 and U_2 denote the speeds of the two thrusters, while θ represents the heading angle.

Considering its actual working environment, trajectory tracking control of the USV on the horizontal plane will be the focus of our study.

There is a reference frame called the BF (body frame) that is securely fixed to the USV, with the point of origin deliberately chosen to coincide with the center of gravity. Global information is recorded by the IF (inertial frame). Thus, the USV's motion can be accurately described via the kinematic equation and dynamic equation of the coordinate transformation between these two frames.

The kinematic equation is

$$\dot{\boldsymbol{\xi}} = \mathbf{R}(\theta)\mathbf{v} \tag{1}$$

where $\boldsymbol{\xi} = [x, y, z]^T$ represents the USV's position and heading in the IF; $\mathbf{v} = [u, v, r]^T$ represents the USV's velocity in the BF; and the rotation matrix $\mathbf{R}(\theta)$ depends on θ, which is the heading angle.

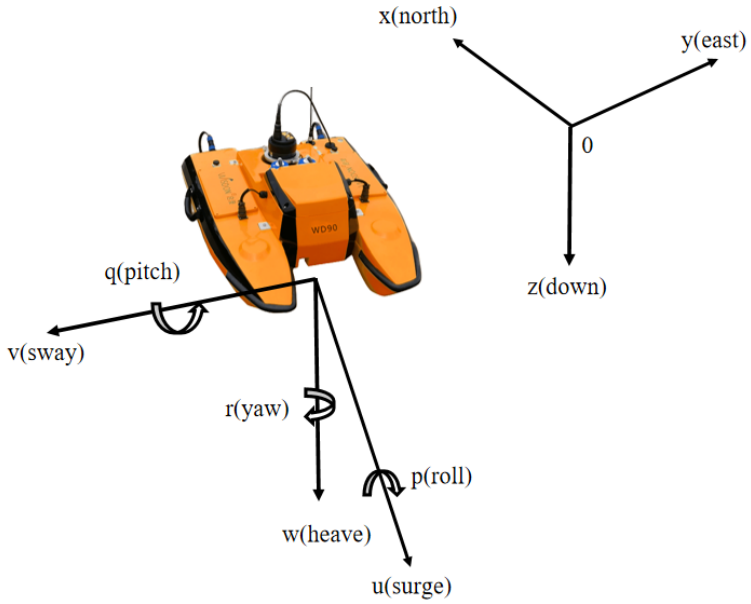

Figure 1. Diagram of the BF (**left**) and IF (**right**).

$\mathbf{R}(\theta)$ can be expressed by the follow equation:

$$\mathbf{R}(\theta) = \begin{bmatrix} cos\theta & -sin\theta & 0 \\ sin\theta & cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{2}$$

According to the Newton's law of motion, the dynamic equation can be established as follows:

$$\mathbf{M}\dot{\mathbf{v}} + \mathbf{C}(\mathbf{v})\mathbf{v} + \mathbf{D}(\mathbf{v})\mathbf{v} + \mathbf{g}(\boldsymbol{\xi}) = \boldsymbol{\kappa} \tag{3}$$

where $\boldsymbol{\kappa} = [F_u, F_v, F_r]^T$ represents the thrust force of each propeller. The matrix $\mathbf{M}$ takes the mass (which is added) into consideration; $\mathbf{C}(\mathbf{v})$ represents the Coriolis and centripetal matrix. Concrete forms of the above three matrices are shown as follows:

$$\mathbf{M} = \begin{bmatrix} M_{\dot{u}} & 0 & 0 \\ 0 & M_{\dot{v}} & 0 \\ 0 & 0 & M_{\dot{r}} \end{bmatrix} \tag{4a}$$

$$\mathbf{C}(\mathbf{v}) = \begin{bmatrix} 0 & 0 & -M_{\dot{v}}v \\ 0 & 0 & M_{\dot{u}}u \\ M_{\dot{v}}v & -M_{\dot{u}}u & 0 \end{bmatrix} \tag{4b}$$

$$\mathbf{D}(\mathbf{v}) = \begin{bmatrix} X_u + D_u|u| & 0 & 0 \\ 0 & Y_v + D_v|v| & 0 \\ 0 & 0 & Z_r + D_r|r| \end{bmatrix} \tag{4c}$$

where $\mathbf{D}(\mathbf{v})$ is the USV's damping matrix; $\mathbf{g}(\boldsymbol{\xi})$ denotes the specific restoring force.

The thrusters $\boldsymbol{\tau} = [\tau_1, \tau_2, \tau_3]^T$ generate thrust force κ, and the τ comes from $\boldsymbol{\kappa} = \mathbf{B}(\boldsymbol{\alpha})\boldsymbol{\tau}.\boldsymbol{\alpha}$ denoting the thrusters' azimuth vector in the BF. We can obtain the distribution of the thruster:

$$\boldsymbol{\kappa} = \mathbf{B}\boldsymbol{\tau}, \mathbf{B} = \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 0 \\ l_1 & 0 & l_2 \end{bmatrix} \tag{5}$$

where $\mathbf{B}$ denotes an input matrix that is constant. $\mathbf{B}$ is a 3×3 matrix that distributes power to the thrusters in three directions, and $\mathbf{B}$ satisfies the condition that $B^{\mathsf{T}}B$ is not singular. $l_1, l_2 \in (0, 1)$ are the thrusters' efficiency factors.

Therefore, we can derive the dynamic model of the USV for trajectory tracking by combining Equations (1), (3), and (5):

$$\dot{\mathbf{x}} = \begin{bmatrix} \mathbf{R}(\boldsymbol{\theta})\mathbf{v} \\ \mathbf{M}^{-1}\mathbf{B}\boldsymbol{\tau} - \mathbf{M}^{-1}\mathbf{C}\mathbf{v} - \mathbf{M}^{-1}\mathbf{D}\mathbf{v} - \mathbf{M}^{-1}\mathbf{g} \end{bmatrix} = \mathbf{f}(\mathbf{x}, \boldsymbol{\tau}) \tag{6}$$

where $\mathbf{x} = [x, y, \theta, u, v, r]^T$ is the defined state, and input control is expressed as $\boldsymbol{\tau} = [\tau_1, \tau_2, \tau_3]^T$. At the end of this section, we successfully derive the dynamic equation governing USV operation on the water surface.

2.2. The USV Trajectory Control Algorithm Based on RHRL

In this section, the USV trajectory control algorithm utilizing RHRL is elaborated. We initially formulate the performance index for the finite time domain trajectory control problem of the USV. Subsequently, we outline the core concepts of the associated reinforcement learning algorithm along with the design and implementation process of the controller. Also included is a detailed analysis of convergence based on this approach.

When conducting tracking control, it is necessary to describe the relative position between the USV and the desired path, as shown in Figure 2. The point P represents the closet point from the desired path, which is called the road projection point. $P(X_p, Y_p, \varphi_d, \kappa)$ is denoted as the path information at the projection point, where X_p, Y_p are the global coordinates of P. φ_d is the angle between the tangent line of P and the X-axis, also known as the direction of the path; κ is the curvature of the path at point P.

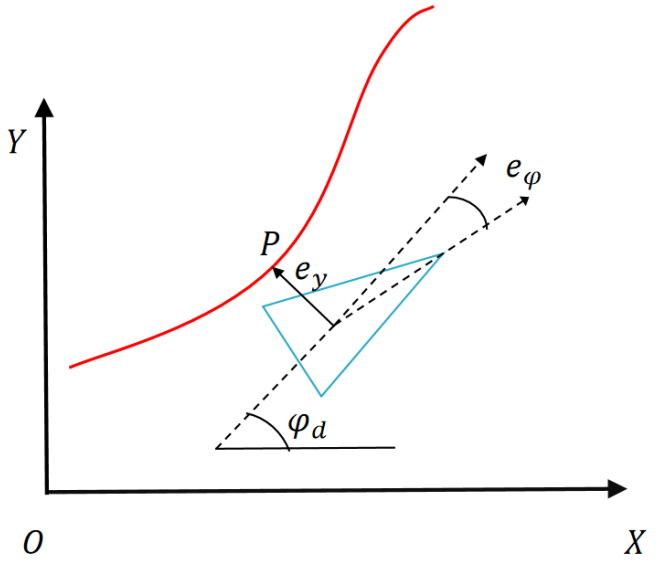

Figure 2. Lateral error model.

The distance between P and the USV centroid is called the lateral deviation e_y, and $e_y > 0$ is specified for when the USV is located on the left side of the path, and $e_y < 0$ when the USV is on the right side. Therefore, the lateral deviation can be expressed as

$$e_y = -(X - X_p)sin(\varphi_d) + (Y - Y_p)cos(\varphi_d) \tag{7}$$

The path deviation e_φ of the USV is defined as the difference between the path and the direction, which is $e_\varphi = \varphi - \varphi_d$. $\varphi = \frac{1}{2}(\dot{z} + r)tan\theta = rtan\theta$. The first derivative of e_y and e_φ are shown below:

$$\begin{cases} \dot{e_y} = v_y cos(e_\varphi) + v_x sin(e_\varphi) \\ \dot{e_\varphi} = \omega - \kappa[v_x cos(e_\varphi) - v_y sin(e_\varphi)] \end{cases} \tag{8}$$

where $\omega = \dot{\varphi}$. $v_x = \dot{x}cos\varphi + \dot{y}sin\varphi + \sqrt{u^2 + v^2}sin\varphi, v_y = -\dot{x}sin\varphi + \dot{y}cos\varphi + \sqrt{u^2 + v^2}cos\varphi$. It is assumed that v_x remains constant and there is no sidescale phenomenon in the moving process, and that the expected yaw velocity of the USV's desired path is constant; then, the lateral acceleration of the USV when it stably tracks the path is $a_y = {v_x}^2\kappa$.

Assuming that the course deviation e_φ is small, then according to the small angle theorem, $sin(e_\varphi) \approx e_\varphi, cos(e_\varphi) \approx 1$. Then, the second derivative of the lateral deviation with respect to time can be expressed as

$$\dot{e_y} = (\dot{v_y} + v_x\omega) - v_x^2\kappa \tag{9}$$

The first derivative can be approximated as

$$\dot{e_y} = v_y + v_x e_\varphi \tag{10}$$

Combining Equations (1), (3) and (4), also (8)–(10); the following equation can be derived as

$$\dot{e} = \mathbf{A}_c e + \mathbf{B}_{c1}u + \mathbf{B}_{c2}\omega_d \tag{11a}$$

$$\mathbf{A}_c = \begin{bmatrix} \mathbf{A}_w & \mathbf{0}_{6\times3} & \mathbf{0}_{6\times3} \\ \mathbf{0}_{3\times6} & \mathbf{0}_{3\times3} & \mathbf{I}_{3\times3} \\ \mathbf{0}_{3\times6} & \mathbf{0}_{3\times3} & -\mathbf{M}^{-1}D \end{bmatrix} \tag{11b}$$

$$\mathbf{A}_w = \begin{bmatrix} 0_{3\times3} \\ \overline{\mathbf{A}_w} \end{bmatrix}, \overline{\mathbf{A}_w} = \begin{bmatrix} \mathbf{M}_{\dot{u}} + \mathbf{M}_{\dot{v}}V & 0 & X_u + D_u|u| \\ 0 & \mathbf{M}_{\dot{u}}u & -\mathbf{M}_{\dot{r}} \\ \mathbf{M}_{\dot{r}} & Y_v + D_v|v| & Z_r + D_r|r| \end{bmatrix} \tag{11c}$$

$$\mathbf{B}_{c1} = \begin{bmatrix} \mathbf{E}_w & \mathbf{0}_{6\times3} \\ \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} \\ \mathbf{0}_{3\times3} & \mathbf{M}^{-1} \end{bmatrix}, \mathbf{B}_{c2} = \begin{bmatrix} \mathbf{0}_{6\times3} \\ \mathbf{0}_{3\times3} \\ \mathbf{M}^{-1} \end{bmatrix} \tag{11d}$$

where $\omega_d = \dot{\varphi_d}, e = [e_y, \dot{e_y}, e_\varphi, \dot{e_\varphi}]^T$, and the control quantity $u = \delta_f$.

Given a sampling period Δt, the discrete time model of Equation (11a) can be discretized as

$$e(k+1) = Ae(k) + B_1u(k) + B_2\omega_d(k) \tag{12}$$

where $A = I + \Delta tA_c, B_1 = \Delta tB_{c1}, B_2 = \Delta tB_{c2}$, and k is a discrete time point.

For the above model Equation (12), it is assumed that path information $(X_i, Y_i)_{i=1}^M$, and the purpose of this paper is to design a lateral control algorithm based on RHRL (as shown in Figure 3) such that during the control process, the above-mentioned lateral error state quantity gradually converges to 0, that is, $e \to 0$.

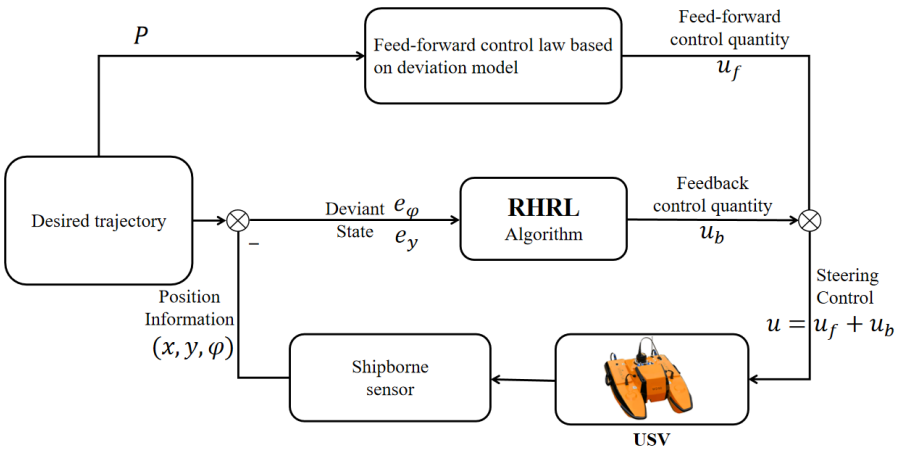

Figure 3. Trajectory tracking control block diagram of the USV.

2.2.1. Design of Performance Index for the Finite Time Domain Trajectory Control Problem

In this section, a detailed control algorithm based on RHRL is presented. We commence by designing the performance index for the USV finite time domain lateral control problem. Subsequently, we outline the core concept of the RHRL algorithm and delve into the design implementation and convergence analysis based on the actuator–evaluator. For the system deviation model of Equation (12), the control quantity can be decomposed into the form of a feedforward component u_f plus a feedback component u_b such that $u = u_f + u_b$, which is shown in Figure 3. The feedforward control quantity represents the expected control input during steady-state vehicle operation and is applicable when the vehicle is stably following the reference path.At the same time $e(k) = e(k+1) = 0$ holds, $u_b = 0$ as well. The feedforward control quantity u_f can be determined as follows:

$$\sum_{j=0}^{\infty} A^j B_1 u_f \approx -\sum_{j=0}^{\infty} A^j B_2 \omega_d \tag{13}$$

The value in the above formula can be obtained by $\omega_d = v_x k$. A, B_1, B_2 are discrete time coefficient matrices.

Since u_f can be easily solved at any current time value k, we assume that u_f remains constant throughout the prediction time domain $[k, k+N]$, then the feedback control quantity u_b to be solved needs to meet the following constraints:

$$u_b \in U_b = \left\{ u \in R | \underline{u} - u_f \leq u \leq \bar{u} - u_f \right\} \tag{14}$$

where $\bar{u}$ represents the maximum of u, $\underline{u}$ is the minimum of u. The RHRL algorithm, introduced in this paper, seeks to minimize the following performance indicator function by optimizing $u_b \in U_b$ in each prediction time domain:

$$V(e(k)) = \sum_{l=k}^{k+N-1} L(e(l), u_b(l)) + V_f(e(k+N)) \tag{15}$$

where the cost function $L(e(l), u_b(l)) = e^T(l) Q e(l) + P u_b(l)^2, Q \in R^{4\times 4}$ is a matrix which is positive definite, P is a preset positive real number, and the cost function of the predictive time domain terminal is

$$V_f(e(k+N)) = e^T(k+N) \bar{R} e(k+N) \tag{16}$$

where the penalty matrix $\bar{R} \in R^{4\times 4}$ is a positive definite matrix, which can be solved using the following Lyapunov equation:

$$F^T \bar{R} F - \bar{R} = -Q - K^T P K \tag{17}$$

where $F = A + B_1 K, K \in R^{1\times 4}$ is the feedback gain matrix satisfying the conditions indicating that F is Schur-stable. (The characteristic polynomial 'F' for discrete linear systems is such that the roots are located within the unit circle. This property results in the system being classified as Schur-stable).

2.2.2. Path Control Algorithm Based on RHRL

The implementation of the finite time domain reinforcement learning algorithm using the executive–evaluator involves the following main steps:

First of all, according to Equation (15), in any $l \in [k, k+N-1]$, we can express the value function as a differential form:

$$V(e(l)) = L(e(l), u_b(l)) + V(e(l+1)) \tag{18}$$

where $V(e(k+N)) = V_f(e(k+N))$. At the l-th prediction moment, $V^*(e(l))$ would be defined as the optimal value function, and we obtain the HJB equation of the above finite time domain optimization control problem as

$$V^*(e(l)) = \min_{u_b(l) \in U_b} L(e(l), u_b(l)) + V^*(e(l+1)) \tag{19}$$

and the optimal control strategy:

$$u^*(e(l)) = \underset{u_b(l) \in U_b}{\operatorname{argmin}} L(e(l), u_b(l)) + V^*(e(l+1)) \tag{20}$$

In fact, due to the control constraints, it is difficult to obtain analytical solutions for V^* and u^* using Equations (19) and (20). In principle, we can approximate the optimal solution of the value function and the control strategy through the method of value iteration. For any $l \in [k, k+N-1]$, at given initial values where $V^0(e(l)) = 0$, then iterate steps $i = 0, 1, 2 \cdots$. This needs to be repeated until $V^{i+1}(e(l)) - V^i(e(l)) \to 0$ to resolve the following two steps.

(1) Strategy update

$$u^i(e(l)) = \underset{u_b(l) \in U_b}{\operatorname{argmin}} L(e(l), u_b(l)) + V^i(e(l+1)) \tag{21a}$$

(2) Value update

$$V^{i+1}(e(l)) = L(e(l), u_b^i(e(l)) + V^i(e(l+1)) \tag{21b}$$

In conclusion, the task of trajectory tracking is accomplished through continuous updating of the strategy and feedback values.

2.2.3. Rolling Time Domain Executor–Evaluator Learning Implementation

We employ the executive–evaluator structure to implement the finite time domain value function iteration algorithm described above. In existing finite time domain reinforcement learning control algorithms [17], the value function in the prediction time domain is regarded as a time-dependent function.

Assumption 1. *If there has a control strategy $u_b(e) = \Phi(v(e))$ so that system (Equation (12)) is asymptotically stable under control strategy $u = u_b + u_f$, where $\Phi(v(e))$ is a continuous function satisfying $u_b(e) \in U_b, \forall v(e) \in R$.*

The aforementioned assumptions essentially represent another aspect of the stabilizability of the system Equation (12). Simultaneously, it is worth noting that the dynamic model Equation (12) presented in this paper is controllable, so there must be a continuous equation $u_b(e) \in U_b$ that renders Equation (16) asymptotically stable under the control strategy $u = u_b + u_f$. Therefore, the above assumptions are reasonable.

We define χ_f as a control invariant set under the control law $u_b = Ke \in U_b$, then we can state the following theorem.

Theorem 1. *(Time-independent value function) If the value of the prediction time domain N satisfies $t \in [k, k+N]$ in any prediction time domain, for any initial state $e(k) \in R^4$, the terminal state $e(k+N) \in \chi_f$ is driven by the control strategy $u(e(l)), l \in [k, k+N-1]$ of system Equation (9) such that there is such a control strategy $u_b(e) \in U_b$ that $V(e(l))$, and $l \in [k, k+N-1]$ is a function that is independent of time.*

Proof of Theorem 1. Firstly, consider the case of $e(k) \in \chi_f$. Based on the definition of χ_f, there is a control law $u_b = Ke = \Phi(v(e)) \in U_b$ that ensures the quantity of states at any time in the future satisfy $x(l) \in \chi_f$. From that, we can solve and obtain the following function:

$$V\big(e(l)\big) = \sum_{i=l}^{k+n-1} L\big(e(i), u_b(i)\big) + V_f\big(e(k+N)\big) = e(l)^T \bar{P} e(l) \tag{22}$$

For the case of $e(k) \notin \chi_f$, according to Assumption 1, there is such a control strategy $u_b = \Phi(v(e))$ and a finite prediction step N that $e(k+N) \in \chi_f$. In particular, let $v = Ke$, then

$$V(e(l)) = \sum_{i=l}^{k+n-1} L(e(i), u_b(i)) + V_f(e(k+N)) = \sum_{i=l}^{+\infty} L(e(i), u_b(i)) \tag{23}$$

where $u_b = \Phi(v(e))$.

Hence, a value function and a strategy independent of time exist. Drawing inspiration from this, we adopt a time-independent executive–evaluator structure to execute the finite time-domain value function iteration process described above. Initially, a network of evaluators is designed to approximate the value function:

$$\hat{V}(e) = \hat{W}_c^T \varphi(e) \tag{24}$$

where $\widehat{W_c} \in R^{N_c}$ represents the weight of the evaluator network, N_c denotes network node number; $\varphi(e)$ is the network's basis function. According to the definition of the evaluator network, the resulting errors E and the end error E_f can be expressed as

$$E(l) = \widehat{W_c}^T \varphi(l) - L(e(l), \widehat{u_b}(l)) - \widehat{W_c}^T \varphi(l+1) \tag{25}$$

$$E_f = \widehat{W_c}^T \varphi\Big(e_f\Big) - {e_f}^T \bar{P} e_f \tag{26}$$

Therein, $e_f = e(k+N)$, which can be randomly valued around 0. By minimizing $E_c(l) = E(l)^2 + {E_f}^2$, the equation for updating the weights of the evaluator network is derived as follows:

$$\widehat{W_c}(l+1) = \widehat{W_c}(l) + \mu_c\Big(\Delta\varphi(e(l+1))E(l) - \varphi\Big(e_f\Big)E_f\Big) \tag{27}$$

where $\mu_c > 0$ is the learning rate of the evaluator network.

Next, to deal with control constraints, we construct the network of actuators as follows:

$$\widehat{u_b}(l) = \overline{u_1} tanh\Big(\widehat{W_a}^T \sigma(e(l))\Big) + \overline{u_2} \tag{28}$$

where $\widehat{u_1} = 0.5\Big(\overline{u_b} - \underline{u_b}\Big)$, $\widehat{u_2} = 0.5\Big(\overline{u_b} + \underline{u_b}\Big)$, including $\widehat{W_a} \in R^{N_a}$ is the weight of actuator network; $\sigma(e)$ is the basis function vector of the network. N_a indicates the node number, which is on network. Given that the actuator network aims to approximate the optimal strategy of control, we define the control quantity deviation as follows:

$$E_a(l) = \widehat{W_a}^T \sigma(e(l)) + \frac{1}{2} R^{-1} {B_1}^T \nabla\varphi(e(l)) \widehat{W_c}(l) \tag{29}$$

By minimizing ${E_a}^2$, we can obtain the update rule of the network weight as

$$\widehat{W_a}(l+1) = \widehat{W_a}(l) - \mu_a \frac{\delta {E_a}^2(l)}{\delta \widehat{W_a}(l)} \tag{30}$$

where $\mu_a > 0$ represents the learning rate of the actuator network.

Algorithm 1 The main steps of implementing the above finite time domain reinforcement learning algorithm, which makes use of the executive–evaluator.

(I) Initialize the weights $\widehat{W_c}$, $\widehat{W_a}$, and obtain the initial state $Z(0)$.
(II) When the time $t = k\Delta t$, the projection point P is found according to the state $Z(t)$, and the deviation state $e(t)$ is calculated.
(III) $\forall l \in [k, k+N-1]$, repeat the following process 1–3:
  (1) According to Equations (17) and (28), $u_f(l)$ and $\widehat{u_b}(l)$ are respectively calculated.
  (2) Update $\widehat{W_c}$, $\widehat{W_a}$ according to Equations (27) and (30).
  (3) Calculate $u(l) = u_f(l) + \widehat{u_b}(l)$ according to Equations (13) and (28), and apply the prediction model for $e(l+1)$.
(IV) Calculate $u_f(k)$ and $\widehat{u_b}(e(k))$ according to Equations (12) and (27), respectively.
(V) In the time period $[k\Delta t, (k+1)\Delta t]$, apply quantity $u(t) = u(k\Delta t)$ directly to the USV, and update the system states $Z((k+1)\Delta t)$.
(VI) Set $k \leftarrow k+1$ and repeat operations II-V based on the receding time domain optimization strategy.

□

2.2.4. Convergence Analysis of the Weight of Finite Time Domain Actuator and Evaluator

Next, we present the convergence analysis of the above RHRL algorithm in each prediction domain $[k, k+N-1]$. First, the (local) optimal value function and control strategy can be represented as a network:

$$V^*(e) = {W_c}^T \varphi(e) + \kappa_c \tag{31}$$

$$u_b{}^* = \overline{u_1} \tan h\Big({W_a}^T \mu(e) + \kappa_a\Big) + \overline{u_2} \tag{32}$$

where both W_a and W_c are weight matrices, and κ_a and κ_c are the errors of reconstruction.

Assumption 2. *(Network reconstruction error)*

(1) $W_c \le W_{c,m}, \varphi \le \varphi_m, \nabla\varphi \le \bar{\varphi}_m, \kappa_c \le \kappa_{c,m}, \nabla\kappa_c \le \bar{\kappa}_{c,m}$
(2) $W_a \le W_{a,m}, \psi \le \psi_m, \kappa_a \le \kappa_{a,m}$

Assumption 3. *(Continuous excitation)*

There are positive real numbers $q_1, q_2, (q_1 < q_2)$ *such that*

$$q_1 \le \bar{\varphi}, \bar{\varphi}_f \le q_2 \tag{33}$$

where $\bar{\varphi} = \Delta\varphi^T \Delta\varphi, \bar{\varphi}_f = {\varphi_f}^T \varphi_f, \varphi_f = \varphi\Big(e_f\Big)$.

In order to more compactly describe the following theorem, define $\gamma_1 = 4 - 4\bar{\psi}\mu_a - (4 - 8\bar{\psi}\mu_a)(\beta_1 + \beta_3), \bar{\psi} = \psi^T\psi, \bar{\varphi} = \bar{\varphi}(l+1) + \bar{\varphi}_f, \alpha = \beta_0, \beta_1, \beta_2, \beta_3$ *are tunable positive real numbers.*

Theorem 2. *Under Assumptions 2 and 3, if the appropriate learning laws* μ_c *and* μ_a *and* $\{\beta_i\}^3 (i = 0)$ *are chosen so that* $\gamma_1 > 0$ *and* $\alpha - \gamma_2 > 0$, *then the network weights* $\hat{W}_c$ *and* $\hat{W}_a$ *of Equations (27) and (30) will asymptotically converge to the following region when using the above strategy:*

$$\bar{W}_c \le \frac{\sqrt{E_t}}{\sqrt{\gamma_1}} \tag{34a}$$

$$\varepsilon_a \le \frac{\sqrt{E_t}}{\sqrt{\alpha - \gamma_2}\lambda_{min(\bar{g})}} \tag{34b}$$

where $\bar{W}_c = W_c - \hat{W}_c, \bar{W}_a = W_a - \hat{W}_a, \xi_a = \bar{W}_a^T \psi$, *and* E_t *is the error.*

Furthermore, if $\kappa_{c,m}, \bar{\kappa}_{c,m}, \kappa_{a,m} \to 0$, *then* $\bar{W}_c$ *and* ξ_a *converge asymptotically to 0.*

Proof of Theorem 2. The Lyapunov function is defined as follows:

$$L(l) = L_c(l) + L_a(l)$$

where $L_c = tr(\bar{W}_c^T \eta_c{}^{-1} \bar{W}_c)$, and $L_a = tr\left(\bar{W}_a^T \eta_a{}^{-1} \bar{W}_a\right)$. They can be calculated based on Equation (26).

$$E(l) = \widehat{W}_c^T \varphi(l) - \widehat{W}_c^T \varphi(l+1) + \Delta V^*(l+1) = \overline{W}_c^T \Delta\varphi(l+1) + \Delta\kappa_c(l+1) \tag{35}$$

where $\Delta V^*(l+1) = V^*(l+1) - V^*(l)$, $\Delta\kappa_c(l+1) = \kappa_c(l+1) - \kappa_c(l)$.

$$E_f = \bar{W}_c^T \varphi_f - W_c{}^T \varphi_f - \kappa_{c,f} = -\bar{W}_c^T \varphi_f - \kappa_{c,f} \tag{36}$$

where $\kappa_{c_1} f = \kappa_c(k+N)$, then according to Equations (27), (35) and (36):

$$\begin{aligned} \Delta L_c(l+1) &= L_c(l+1) - L_c(l) \\ &= 2\bar{W}_c^T(-\bar{\varphi}\bar{W}_c + \bar{\kappa}_c) + \mu_c(-\bar{\varphi}\bar{W}_c + \bar{\kappa}_c)^T(-\bar{\varphi}\bar{W}_c + \bar{\kappa}_c) \\ &\le -\alpha\bar{W}_c^2 + E_c \end{aligned} \tag{37}$$

where $\bar{\kappa}_c = -\Delta\varphi(l+1)\Delta\kappa_c(l+1) - \varphi_f\kappa_{cf}$, $E_c = \left(2\mu_c + \beta_0{}^{-1}\right)\bar{\kappa}_c^2$.

Similarly, $\Delta L_a(l+1)$ can be expressed as

$$\Delta L_a(l+1) = tr\left[2\bar{W}_a^T(l)\frac{\partial E_a{}^2(l)}{\partial \hat{W}_a(l)} + \mu_a\left(\frac{\partial E_a{}^2(l)}{\partial \hat{W}_a(l)}\right)^T \frac{\partial E_a{}^2(l)}{\partial \hat{W}_a(l)}\right]$$

In consideration of $E_a = -\xi_a - g\bar{W}_c + \bar{\kappa}_a$, $g = \nabla\varphi\left(\frac{1}{2}R^{-1}B_1{}^T\right)$, $\bar{\kappa}_a = -\kappa_a - \nabla\kappa_c\left(\frac{1}{2}R^{-1}B_1{}^T\right)$, and $\frac{\partial E_a{}^2(l)}{\partial \hat{W}_a(l)} = 2\psi E_a$, then

$$\Delta L_a = -(4 - 4\bar{\psi}\mu_a)\|\xi_a\|^2 - 8\bar{\psi}\mu_a g\bar{W}_c\bar{\kappa}_a + 4\bar{\psi}\mu_a\|\bar{W}_c\|\bar{g}^2 + (4 - 8\bar{\psi}\mu_a)(\xi_a\bar{\kappa}_a - \xi_a^T g\bar{W}_c)$$

where $\bar{g} = g^T g$. According to Young's inequality theorem,

$$\Delta L_a(l+1) \le -\gamma_1\|\xi_a\|^2 + \gamma_2\|\overline{W}_c\|_{\bar{g}}^2 + E_a$$

where $E_a = (1/\beta_2 + 1/\beta_3)\bar{\kappa}_a^2$. Then, by defining $E_t = E_{c,m} + E_{a,m}$, we obtain

$$\Delta L = -\gamma_1\|\xi_a\|^2 - (\alpha - \gamma_2)\|\bar{W}_c\|_{\bar{g}}^2 + E_t \tag{38}$$

On this basis, if $\bar{\kappa}_{c,m}$, $\kappa_{c,m}$, $\kappa_{a,m} \to 0$, $E_t \to 0$ is obtained, then $\bar{W}_c$ and ξ_a asymptotically converge to 0.

Hence, at this juncture, we have successfully concluded the proof of Theorem 2. □

The conclusion of the above theorem indicates that we can make u converge to u_b^* with an arbitrarily small error by increasing the number of base function nodes in the actuator and the evaluator. Therefore, under the premise that Assumption 1 is true, if a sufficiently large N is chosen, the equation of system (12) satisfies the terminal state $e(k+N) \in x_f$ in the prediction time domain $[k, k+N-1]$ driven by strategy $u_b^*(k|k), \cdots, u_b^*(k+N-1|k)$. Thus, the next prediction time domain $[k+1, k+N]$, $u_b^*(k+1|k), \cdots, u_b^*(k+N-1|k)$, $Ke(k+N|k)$ is a feasible control strategy. We define the loss function produced by the feasible strategy for $Los^f(k+1|k)$, and referring to Rawling's [29], $Los^f(k+1|k) - Los^*(k|k) \le -L(e(k|k), u_b(k|k))$ is available. Due to $Ke(k+N|k)$ being suboptimal, we may safely derive

$$Los^*(k+1|k+1) - Los^*(k|k) \le Los^f(k+1|k) - Los^*(k|k) \le -L(e(k|k), u_b(k|k))$$

which can be obtain by using Lyapunov stability analysis of the stability of the system, which is a closed-loop system.

3. Simulation Analysis

To ensure a precise comparison of the control performance between RHRL, Lyapunov-based MPC (LMPC), and sliding mode control (SMC), the control variable method was adopted using experimental parameters from [24,25]. In the simulations, all of the hydrodynamic parameters in the equations are based on the Falcon model [30].

The simulation results are presented in this section in showcasing the advantages of the RHRL method. In addition, the operating environment is Matlab 2021b, and the core is R7-5800H.

3.1. Parameter Selection

Two distinct desired trajectories are employed. Refer to the article of Li [31], where one trajectory (Path I) is a typical sinusoidal path:

$$p(t) = \begin{cases} x_d = 0.4t \\ y_d = \sin(0.4t) \end{cases} \tag{39}$$

The other trajectory, Path II, is based on [32] and is an S-shaped path:

$$p(t) = \begin{cases} \chi_d = -\sin(0.4t) \\ y_d = \sin(0.24t) \end{cases} \tag{40}$$

For the RHRL controller, the following parameters are utilized: the prediction horizon is set such that $T = 5\delta$, where $\delta = 0.1$ [s] represents the time period; three matrices are set for weighting as $Q = diag(10^5, 10^5, 10^3, 10^2, 10^2, 10^2)$, $R = diag(10^{-4}, 10^{-4}, 10^{-4}, 10^{-4})$, and $P = diag(10^3, 10^3, 10^2, 10, 10)$. The gains of the control are $K_p = K_d = diag(1, 1, 1)$.And the $l_1 = l_2 = 0.8$.

In this section, the desired trajectory tracking simulation of a USV based on RHRL will be executed as described to emphasize the feasibility and efficiency of RHRL algorithm proposed earlier. The parameters for USV simulation are presented in Table 1.

Table 1. Parameters for USV simulation.

Parameters	Value
M/kg (mass of USV)	37
D/m (distance from motors and center of mass)	0.7
K (viscosity coefficient)	0.1
I (moment of inertia)	0.2
T_e (sampling period)	0.2
i (loop index)	1
$U_{cruise} = U_1 = U_2$	2

3.2. Tracking Performance

Both Figure 4a,c depict the tracking results for Path I. The USV trajectories are represented by the blue curve for the LMPC control method, the green curve for the SMC controller, and the red curve for the USV RHRL controller, and the black curve illustrates the sinusoidal trajectory, which is the desired trajectory. The results demonstrate that all controllers are successful in guiding the USV along the desired trajectory, affirming the stability of the closed loop. However, the RHRL method notably exhibits a considerably accelerated convergence compared to the LMPC and SMC methods. This acceleration in convergence is attributed to the selection of control gain matrices K_p and K_d, which are small. The simulation results show that the improvement of tracking accuracy is due to synchronous online incremental learning and deployment.

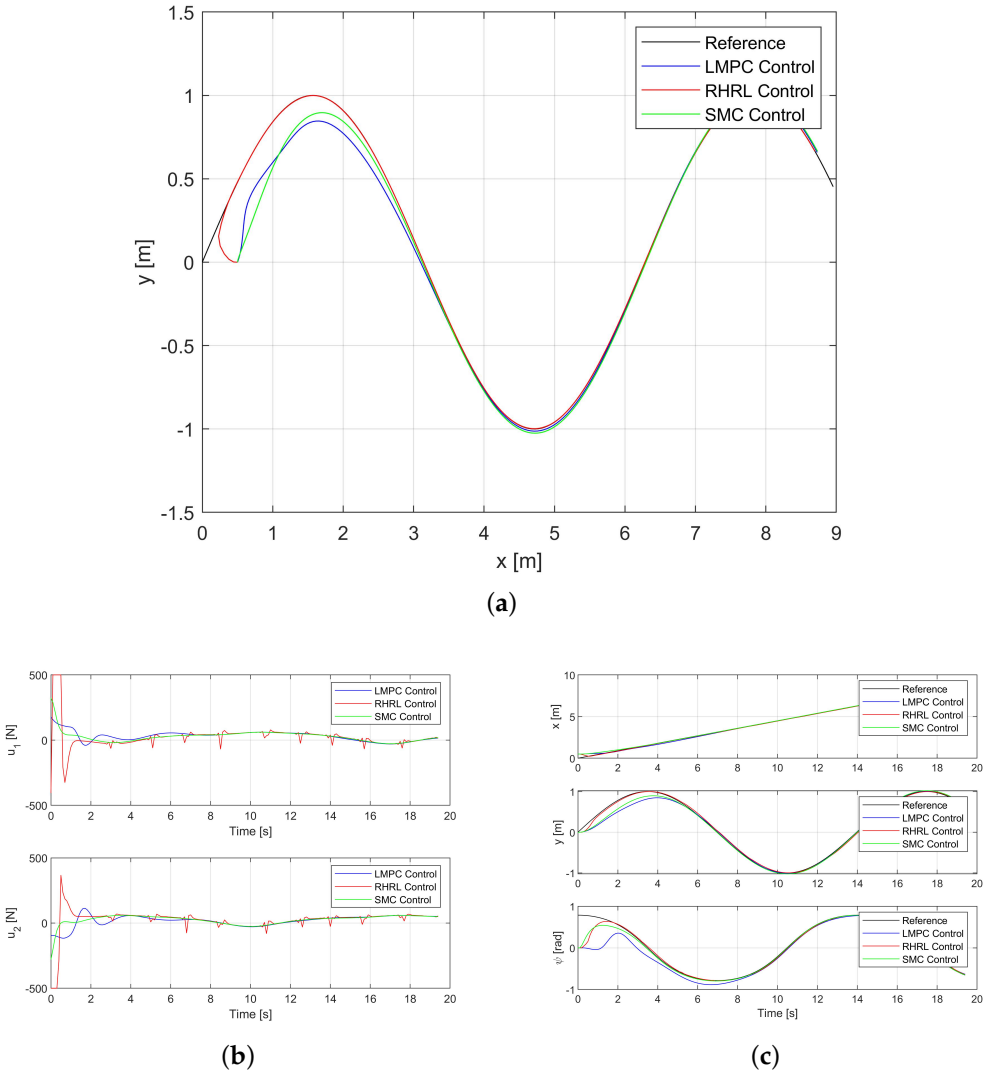

Figure 4. The USV trajectory tracking performance in Path I. (**a**) The USV trajectory for Path I. (**b**) The thrust outputs for Path I. (**c**) The state trajectories for Path I.

Figure 4b illustrates the thrust output of each propeller. It is evident that at the commencement of tracking, the RHRL controller maximally utilizes the onboard thrust capability to achieve convergence as swiftly as possible. In essence, the state remains within the prescribed boundary, aligning with expectations. It is also notable that RHRL demonstrates superior adjustment capability and undergoes more rapid adjustments.

The outcomes for Path II are presented in Figure 5. Similarities arise from the observations: The USV exhibits quicker convergence to the desired trajectory through RHRL.

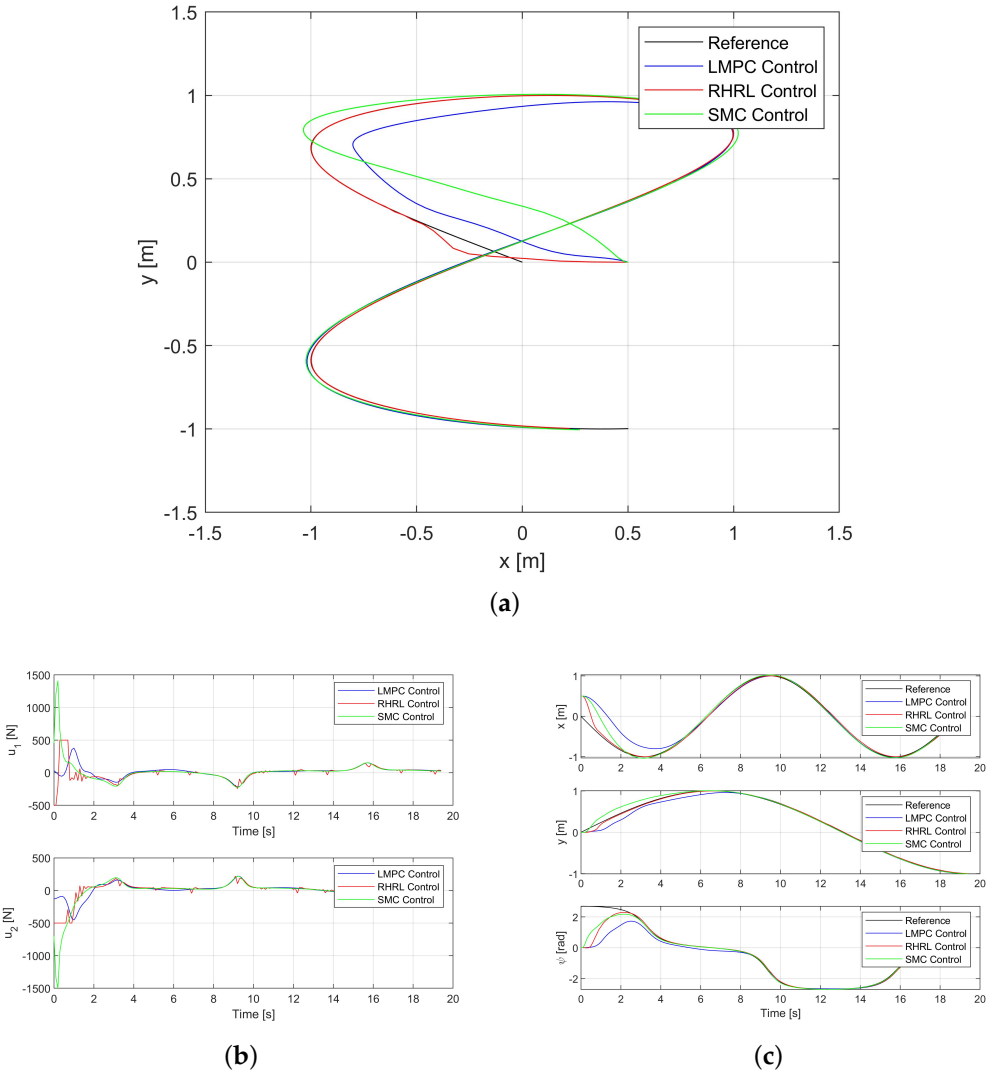

Figure 5. The USV trajectory tracking performance in Path II. (**a**) The USV trajectory for Path II. (**b**) The thrust outputs for Path II. (**c**) The state trajectories for Path II.

3.3. Robustness Experiment with Disturbance

The incorporation of the receding horizon implementation introduces feedback into the closed-loop system. One of the inherent advantages of the RHRL controller is its robustness toward disturbances and emergencies, making it particularly well-suited for control systems in marine and submarine environments.The RHRL's robustness is thoroughly examined and demonstrated through simulations. The definite simulated disturbance of magnitude $[100(N), 100(N), 0(Nm)]^T$ was added. To provide a clearer visualization of the deviation between the three algorithms, the reference trajectory, indicated by a black line, is also included in this experiment.

In analyzing the outcomes shown from Figure 5 to Figure 6, it is evident that RHRL tracking control consistently guides the USV to adequately converge toward the desired trajectory. In contrast, substantial tracking errors are exhibited when conducting tracking control using LMPC, the even greater errors are associated with SMC. Figures 6b and 7b illustrate that the RHRL controller consistently provides feedback for responding within a small time domain, ensuring minimal deviation.

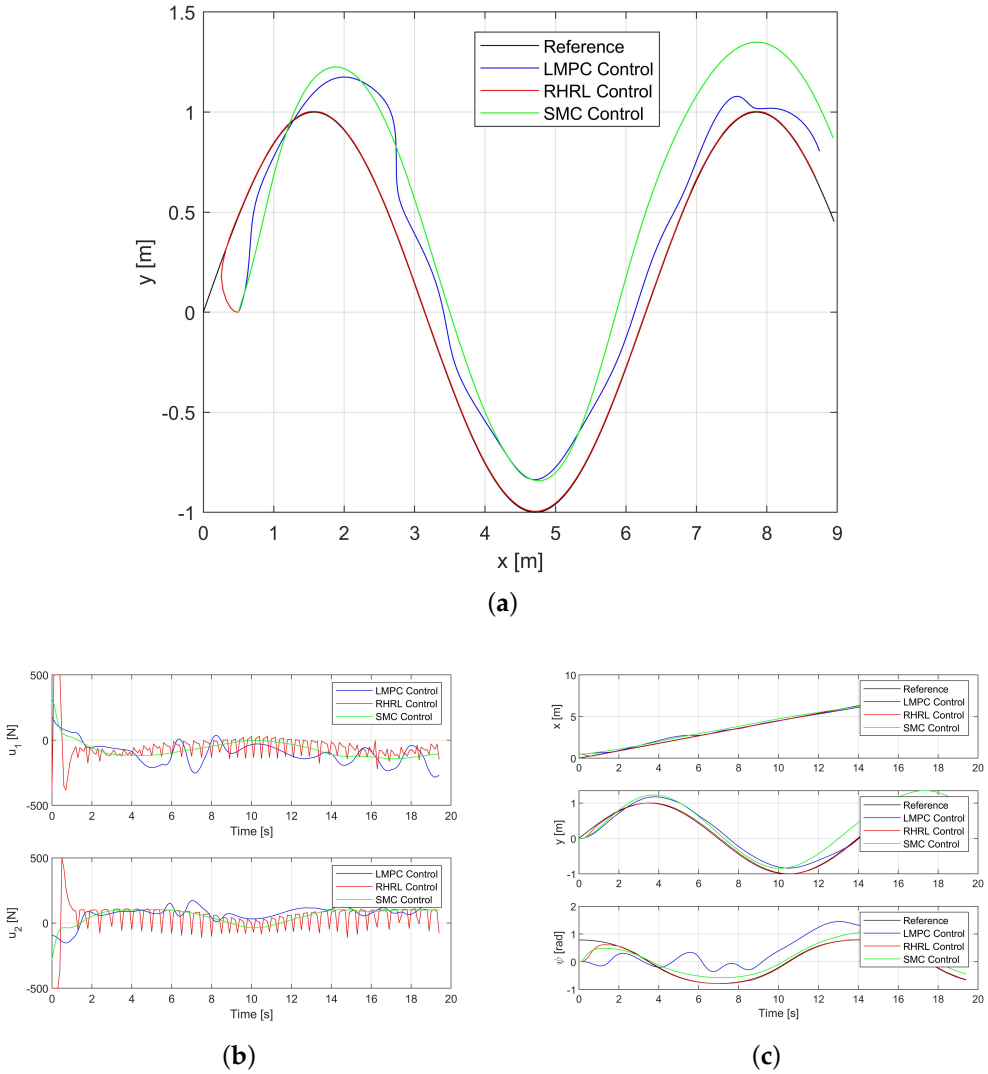

Figure 6. The USV trajectory tracking performance in Path I with disturbance. (**a**) The USV trajectory for Path I. (**b**) The thrust outputs for Path I. (**c**) The state trajectories for Path I.

The MSEs (mean square errors) for both paths are consolidated in Tables 2 and 3. Generally, the MSEs are approximately 10 times smaller for RHRL compared to LMPC and SMC, especially in the case of Path II. Indeed, it is widely acknowledged that smaller MSEs correspond to reduced tracking error, thereby resulting in higher tracking accuracy; thus, it is evident that the RHRL algorithm significantly enhances tracking accuracy.

Table 2. MSE for disturbances in Path I.

MSE	LMPC	RHRL	SMC	Improvement I	Improvement II
x/m^2	0.0518	0.0086	0.0732	83.3%	88.2%
y/m^2	0.0286	0.0031	0.0391	89.3%	92.1%
ψ/rad^2	0.3198	0.0358	0.4273	88.9%	91.6%

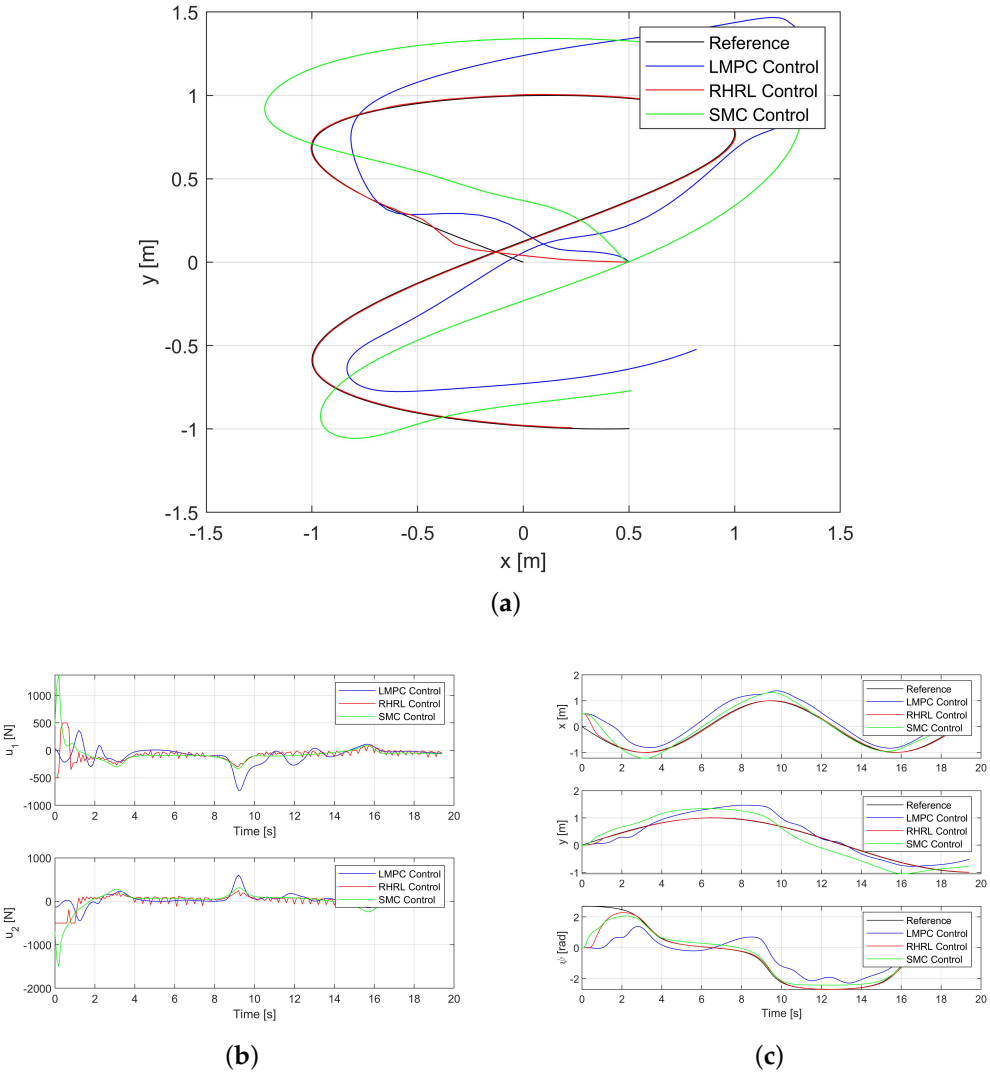

Figure 7. The USV trajectory tracking performance in Path II with disturbance. (**a**) The USV trajectory for Path II. (**b**) The thrust outputs for Path II. (**c**) The state trajectories for Path II.

Table 3. MSE for disturbances in Path II.

MSE	LMPC	RHRL	SMC	Improvement I	Improvement II
x/m^2	0.1386	0.0158	0.1450	88.6%	89.1%
y/m^2	0.0968	0.0079	0.1002	91.8%	92.1%
ψ/rad^2	0.8663	0.3561	0.9984	58.8%	64.3%

In order to more objectively demonstrate the excellent performance of the algorithm, we propose conducting quantitative analysis based on a new factor, namely thrust output. It is known that a smaller average value of thrust corresponds to lower energy consumption and enhanced cost-effectiveness. The specific data are shown in Tables 4 and 5. As can be seen from the tables, the energy consumption of RHRL compared with LMPC is reduced by 43.85% and 41.65% for Paths I and II, respectively. The data show that RHRL is much more economical than LMPC. However, due to the algorithm characteristics, RHRL does not have a significant advantage over SMC based on this analysis.

Table 4. The average thrust output with disturbances in Path I.

TO	LMPC	RHRL	SMC	Improvement I	Improvement II
$\|U_1\|/N$	152.9	86.3	86.8	43.6%	0.57%
$\|U_2\|/N$	154.2	86.5	87.6	43.9%	1.25%

Table 5. The average thrust output with disturbances in Path II.

TO	LMPC	RHRL	SMC	Improvement I	Improvement II
$\|U_1\|/N$	106.7	62.1	63.2	41.8%	1.74%
$\|U_2\|/N$	109.2	63.9	64.6	41.5%	0.11%

The observed disparity stems from RHRL's ability to learn and adapt online, utilizing online optimization to dynamically adjust control gains and effectively compensate for interference. Conversely, both LMPC and SMC lack this flexibility. Consequently, robustness is significantly enhanced by RHRL control.

4. Conclusions

In this paper, a trajectory control algorithm for USV based on RHRL is introduced in which reinforcement learning is seamlessly integrated with a rolling time domain optimization mechanism. Thus, infinite time self-learning optimization problems are effectively converted into a series of finite time optimization problems, which can then be solved using an executive–evaluator algorithm. The incorporation of the rolling time domain mechanism in this design approach significantly enhances the learning efficiency of the RL algorithm. Moreover, compared to LMPC and SMC, the optimization method utilizing both effector and evaluator contributes to enhanced computational efficiency. In diverging from the majority of existing finite time domain executive–evaluator learning algorithms, the proposed RHRL employs a time-independent single-network structure. This innovative approach serves to diminish the intricacy associated with network design and online computational complexity. Moreover, we analyzed the stability of the closed-loop system theoretically. Concerning scenarios involving significant errors in the learned approximation strategy, we plan to conduct in-depth analysis and substantiation in our forthcoming research. The results of simulations demonstrate that our algorithm is effective based on comparison with typical traditional algorithms in simulation scenarios. The simulation results show that RHRL control is superior to LMPC and SMC in terms of control performance and computational efficiency while also being more economical than LMPC. RHRL control also has lower sample complexity and higher learning efficiency.

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

Funding: This research was funded by Fundamental Research Funds for the Central Universities of Ministry of Education of China grant number 2023IVA092.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Data are contained within the article.

Acknowledgments: The authors would like to thank the anonymous reviewers for their helpful comments which improved the quality of the paper.

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

Abbreviations

The following abbreviations are used in this manuscript:

USV	unmanned surface vehicle
RHRL	receding horizon reinforcement learning
LMPC	Lyapunov model predictive control
PID	proportional integral derivative
MPCC	model predictive controlling control
ADP	approximate dynamic programming
KDHP	kernel-based DHP
BF	body frame
IF	inertial frame
MSE	mean square error

References

1. Alim, M.F.A.; Kadir, R.E.A.; Gamayanti, N.; Santoso, A.; Sahal, M. Autopilot system design on monohull USV- LSS01 using PID-based sliding mode control method. *IOP Conf. Ser. Earth Environ. Sci.* **2021**, *649*, 012058. [CrossRef]
2. Guo, X.-K. Particle swarm optimization for pid usv heading stability control. *Ship Sci. Technol.* **2019**, *41*, 52–54.
3. Ege, E.; Ankarali, M.M. Feedback motion planning of unmanned surface vehicles via random sequential composition. *Trans. Inst. Meas. Control* **2019**, *41*, 3321–3330 [CrossRef]
4. Huanyin, Z.; Xisheng, F.; Zhiqiang, H.U.; Wei, L.I. Dynamic Feedback Controller Based on Optimized Switching of Multiple Identification Models for Course Control of Unmanned Surface Vehicle. *Robot* **2013**, *35*, 552.
5. Yan, D.; Xiao, C.; Wen, Y. Pod Propulsion Small Surface USV Heading Control Research. In Proceedings of the 26th International Ocean and Polar Engineering Conference, Rhodes, Greece, 26 June–1 July 2016.
6. Deng, Y.; Zhang, X.; Im, N.; Zhang, G.; Zhang, Q. Adaptive fuzzy tracking control for underactuated surface vessels with unmodeled dynamics and input saturation. *ISA Trans.* **2020**, *103*, 52–62. [CrossRef] [PubMed]
7. Dong, Z.; Zhang, Z.; Qi, S.; Zhang, H.; Li, J.; Liu, Y. Autonomous cooperative formation control of underactuated USVs based on improved MPC in complex ocean environment. *Ocean Eng.* **2023**, *270*, 113633. [CrossRef]
8. Han, X.; Zhang, X. Tracking control of ship at sea based on MPC with virtual ship bunch under Frenet frame. *Ocean Eng.* **2022**, *247*, 110737. [CrossRef]
9. Johnson, A.M.; Lenell, C.; Severa, E.; Rudisch, D.M.; Morrison, R.A.; Shembel, A.C. Semi-Automated Training of Rat Ultrasonic Vocalizations. *Front. Behav. Neurosci.* **2022**, *16*, 826550. [CrossRef] [PubMed]
10. Zhao, Y.; Qi, X.; Ma, Y.; Li, Z.; Sotelo, M.A. Path Following Optimization for an Underactuated USV Using Smoothly-Convergent Deep Reinforcement Learning. *IEEE Trans. Intell. Transp. Syst.* **2020**, *22*, 6208–6220. [CrossRef]
11. Guo, J. Study on Lateral Fuzzy Control of Unmanned Vehicles Via Genetic Algorithms. *J. Mech. Eng.* **2012**, *48*, 76. [CrossRef]
12. Leonard, J.; How, J.; Teller, S.; Berger, M.; Williams, J. A Perception-Driven Autonomous Urban Vehicle. *J. Field Robot.* **2009**, *25*, 727–774. [CrossRef]
13. Rajamani, R.; Zhu, C.; Alexander, L. Lateral control of a backward driven front-steering vehicle. *Control Eng. Pract.* **2003**, *11*, 531–540. [CrossRef]
14. Taherian, S.; Halder, K.; Dixit, S.; Fallah, S. Autonomous Collision Avoidance Using MPC with LQR-Based Weight Transformation. *Sensors* **2021**, *21*, 4296. [CrossRef] [PubMed]
15. Falcone, P.; Borrelli, F.; Asgari, J.; Tseng, H.E.; Hrovat, D. Predictive Active Steering Control for Autonomous Vehicle Systems. *IEEE Trans. Control Syst. Technol.* **2007**, *15*, 566–580. [CrossRef]
16. Beal, C.E.; Gerdes, J.C. Model Predictive Control for Vehicle Stabilization at the Limits of Handling. *IEEE Trans. Control Syst. Technol.* **2013**, *21*, 1258–1269. [CrossRef]
17. Li, D.; Zhao, D.; Zhang, Q.; Chen, Y. Reinforcement Learning and Deep Learning Based Lateral Control for Autonomous Driving [Application Notes]. *IEEE Comput. Intell. Mag.* **2019**, *14*, 83–98. [CrossRef]
18. Domahidi, A.; Liniger, A.; Morari, M. Optimization-Based Autonomous Racing of 1:43 Scale RC Cars. *Optim. Control Appl. Methods* **2017**, *36*, 628–647.
19. Ostafew, C.J.; Schoellig, A.P.; Barfoot, T.D. Robust Constrained Learning-based NMPC enabling reliable mobile robot path tracking. *Int. J. Robot. Res.* **2016**, *35*, 1547–1563. [CrossRef]
20. Alighanbari, S.; Azad, N.L. Safe Adaptive Deep Reinforcement Learning for Autonomous Driving in Urban Environments. Additional Filter? How and Where? *IEEE Access* **2021**, *9*, 141347–141359 [CrossRef]
21. Chen, Y.; Hereid, A.; Peng, H.; Grizzle, J. Enhancing the Performance of a Safe Controller Via Supervised Learning for Truck Lateral Control. *J. Dyn. Syst. Meas. Control* **2019**, *141*, 101005. [CrossRef]
22. Zhou, X.; Wu, Y.; Huang, J. MPC-based path tracking control method for USV. In Proceedings of the 2020 Chinese Automation Congress (CAC), Shanghai, China, 6–8 November 2020.
23. Gong, C.; Su, Y.; Zhu, Q.; Zhang, D.; Hu, X. Finite-time dynamic positioning control design for surface vessels with external disturbances, input saturation and error constraints. *Ocean Eng.* **2023**, *276*, 114259. [CrossRef]

24. Shen, C.; Shi, Y.; Buckham, B. Trajectory Tracking Control of an Autonomous Underwater Vehicle Using Lyapunov-Based Model Predictive Control. *IEEE Trans. Ind. Electron.* **2017**, *65*, 5796–5805. [CrossRef]
25. Jiang, X.; Xia, G. Sliding mode formation control of leaderless unmanned surface vehicles with environmental disturbances. *Ocean Eng.* **2022**, *244*, 110301. [CrossRef]
26. Mayne, D.Q.; Kerrigan, E.C. Tube-Based Robust Nonlinear Model Predictive Control. *Int. J. Robust Nonlinear Control* **2009**, *21*, 1341–1353. [CrossRef]
27. Zhang, X.; Pan, W.; Scattolini, R.; Yu, S.; Xu, X. Robust Tube-based Model Predictive Control with Koopman Operators–Extended Version. *arXiv* **2021**, arXiv:2108.13011.
28. Haarnoja, T.; Zhou, A.; Abbeel, P.; Levine, S. Soft Actor-Critic: Off-Policy Maximum Entropy Deep Reinforcement Learning with a Stochastic Actor. In Proceedings of the 35th International Conference on Machine Learning, Stockholm, Sweden, 10–15 July 2018.
29. Rawlings, J.; Mayne, D.; Diehl, M. *Model Predictive Control: Theory, Computation, and Design*; Nob Hill Publishing, LLC: Madison, WI, USA, 2017.
30. Proctor, A.A. Semi-autonomous guidance and control of a Saab SeaEye Falcon ROV. Ph.D. Thesis, University of Victoria, Victoria, BC, Canada, 2014.
31. Li, B.; Zhang, H.; Niu, Y.; Ran, D.; Xiao, B. Finite-time disturbance observer-based trajectory tracking control for quadrotor unmanned aerial vehicle with obstacle avoidance. *Math. Methods Appl. Sci.* **2023**, *46*, 1096–1110. [CrossRef]
32. Hmeyda, F.; Bouani, F. Camera-based autonomous Mobile Robot Path Planning and Trajectory tracking using PSO algorithm and PID Controller. In Proceedings of the 2017 International Conference on Control, Automation and Diagnosis (ICCAD), Hammamet, Tunisia, 19–21 January 2017.