Policy Learning for Nonlinear Model Predictive Control With Application to USVs

Rizhong Wang, Huiping Li, Senior Member, IEEE, Bin Liang, Senior Member, IEEE, Yang Shi, Fellow, IEEE, and Demin Xu

Abstract—The unaffordable computation load of nonlinear model predictive control (NMPC) has prevented it from being used in robots with high sampling rates for decades. This article is concerned with the policy learning problem for nonlinear MPC with system constraints, where the nonlinear MPC policy is learned offline and deployed online to resolve the computational complexity issue. A deep neural networks (DNN)-based policy learning MPC (PL-MPC) method is proposed to avoid solving nonlinear optimal control problems online. The detailed policy learning method is developed and the PL-MPC algorithm is designed. The strategy to ensure the practical feasibility of policy implementation is proposed, and it is theoretically proved that the closed-loop system under the proposed method is asymptotically stable in probability. In addition, we apply the PL-MPC algorithm successfully to the motion control of unmanned surface vehicles (USVs). It is shown that the proposed algorithm can be implemented at a sampling rate up to 5 Hz with high-precision motion control.

Index Terms—Constraints, deep neural networks (DNN), model predictive control (MPC), policy learning.

I. INTRODUCTION

As one of the most advanced control technologies, model predictive control (MPC) has been widely used in process control [1]. Recently, MPC strategies have become popular to solve robot control problems because of its optimal control performance and the ability to handle complex system constraints [2], [3], [4]. However, MPC requires solving an optimal control problem (OCP) periodically online, which is time consuming and cannot guarantee the real-time implementation in robot control applications. This brings great challenges to the application of MPC in the field of robotics [5].

In order to improve the online computational efficiency of OCP, several improved MPC algorithms have been proposed. The work in [6] simplifies the nonlinear quadrupeled robot model to a linear model. In that framework, the original nonlinear model predictive control (NMPC) optimization problem was replaced by a quadratic program (QP) problem, which can greatly reduce computational burden. In [3], a penalty term is added to the cost function to relax the inequality constraints of OCP, which improves the MPC efficiency.

Another approach to accelerate solution efficiency of OCP is to perform the online MPC optimization process offline. Explicit MPC is a typical method in this regard [7]. The works in [8] and [9] obtain the explicit solution of the linear MPC optimal problem by solving the multiparameter QP problem or using the piecewise affine function, respectively. In [10], an approximate explicit MPC control policy is constructed using the barycentric interpolation method, and the system stability and feasibility of the algorithm are theoretically guaranteed. Note that all the above algorithms are only suitable for linear systems.

Recently, the learning MPC has become a promising approach based on machine learning [11]. Since the control accuracy of MPC heavily relies on the accuracy of system model, the works in [12] and [13] focused on learning the system model for MPC. Besides, machine learning methods can also assist the design of the MPC controller. For example, the works in [14] and [15] used a machine learning method to design the cost function. Alternatively, the work in [16] used the machine learning method to obtain a terminal region of MPC. Besides, machine learning algorithms can also learn the best system constraints of MPC [17]. On the other hand, the MPC can provide expert data for machine learning algorithms. The work in [18] used MPC to guide reinforcement learning (RL) training to improve the training efficiency of RL and guarantee the safety of the RL results.

Different from the traditional machine learning methods, deep neural networks (DNN) have strong capability to learn complex functions and policies. It can be implemented with a faster computational speed of forward propagation after appropriate training, without solving complex optimization problems online. Therefore, a promising method to solve the real-time implementation issue of MPC is to use DNNs to learn the control policy offline and then deploy it online [19]. In [20], the primal-dual DNN is used to learn the approximate policy of
MPC. To improve the speed of solving large-scale linear MPC, the work in [21] used DNNs to learn the optimal solution, and the system stability and feasibility of the optimization problem were guaranteed by primal active sets. The work in [22] utilized the DNN to approximate control policy of robust MPC, and the feasibility and stability of the closed-loop system are also analyzed. However, most of the above works are developed for linear systems, which are not suitable for nonlinear systems such as robotic and vehicle systems.

In this article, we propose a new policy learning MPC (PL-MPC) scheme for constrained nonlinear systems using DNN and implement it to the motion control of unmanned surface vehicles (USVs). The main contributions of this article include the following.

1) We propose a deep supervised learning method to learn the policy of the constrained nonlinear MPC offline to greatly reduce the computational load. The detailed DNN model and the PL-MPC algorithm are developed, which provides a feasible tool to deploy nonlinear MPC for robotic control and planning in real-time application.

2) We conduct rigorous analysis of the proposed PL-MPC algorithm to make the proposed method theoretically valid. The proximal operator-based optimization and the quadratic penalty method are developed to ensure the practical feasibility of the proposed algorithm. With the PL-MPC algorithm, the closed-loop system is proved to be asymptotically stable in probability under mild conditions.

3) We implement the developed PL-MPC algorithm successfully to the motion control of an underactuated USV via lake experiments. The experimental results show that the PL-MPC algorithm can run up to the sampling rate of 5 Hz, and the control performance is almost the same as the ideal NMPC in simulation. This verifies the effectiveness and advantage of the proposed policy learning MPC method for robotic.

The rest of this article is organized as follows. Section II presents the preliminaries and Section III discusses the detailed PL-MPC algorithm. In Section IV, the feasibility of the algorithm and the stability of the closed-loop system are presented. Then real-time experiments on USV motion control with the proposed method are conducted in Section V. Finally, Section VI concludes this article.

II. PRELIMINARIES

A. System Model

Consider a general discrete-time nonlinear dynamic system

\[ x(k+1) = f(x(k), u(k)). \]  

(1)

Here, \( x(k) \in \mathbb{R}^n \) and \( u(k) \in \mathbb{R}^m \) are the system state and the control input, respectively. The system state and control input are constrained by the compact sets \( \mathcal{X} \) and \( \mathcal{U} \) as follows:

\[ \mathcal{U} = \{ u(k) \in \mathbb{R}^m | u_{\text{min}} \leq u(k) \leq u_{\text{max}} \} \]

\[ \mathcal{X} = \{ x(k) \in \mathbb{R}^n | g(x(k)) \leq 0 \} . \]

(2)

We make the following standing assumptions [23].

**Assumption 1:** \( \mathcal{X} \) is a connected set, and \( (0, 0) \) is included within \( \mathcal{U} \times \mathcal{X} \).

**Assumption 2:** The function \( f(\cdot, \cdot) : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^n \) is second-order continuous differentiable, and \( 0 \in \mathbb{R}^n \) is an equilibrium point of the system, i.e., \( f(0,0) = 0 \).

B. Optimal Control Problem

Without loss of generality, we choose the equilibrium point as the target system state and control input. The cost function is designed as

\[ J(x(k), \bar{u}(\cdot; k)) = \sum_{i=0}^{N-1} F(\bar{x}(k+i; k), \bar{u}(k+i; k)) + V_f(\bar{x}(k+N; k)) \]

\[ = \sum_{i=0}^{N-1} (\|\bar{x}(k+i; k)\|_Q^2 + \|\bar{u}(k+i; k)\|_R^2) \]

\[ + \|\bar{x}(k+N; k)\|_P^2 , \tag{3} \]

where \( \bar{u}(k+i; k) \) and \( \bar{x}(k+i; k) \) are control input sequence and system state sequence emanating from \( x(k) \). \( N \) is the prediction horizon. The quadratic norm \( x^T Q x \) is denoted by \( \|x\|_Q^2 \), \( Q \geq 0, R \geq 0 \) and \( P \geq 0 \) represent the adjustable weighting matrices. Then, Problem P1 is formulated as follows:

\[ u^*(\cdot; k) = \arg\min_{\bar{u}(\cdot; k)} J(x(k), \bar{u}(\cdot; k)) \]

s.t. \( \bar{x}(k+i+1; k) = f(\bar{x}(k+i; k), \bar{u}(k+i; k)) \]

\[ \bar{u}(k+i; k) \in \mathcal{U} \]

\[ \bar{x}(k+i; k) \in \mathcal{X} \]

\[ \bar{x}(k+N; k) \in \mathcal{X}_f \]

\[ \bar{x}(k+N; k) \in \mathcal{X}_f \]

where \( i = 0, \ldots, N-1, \mathcal{X}_f = \{ x \in \mathbb{R}^n | x^T P x \leq \alpha \} \subseteq \mathcal{X} \) is the terminal region, and \( u^*(\cdot; k) = \{ u^*(k+1; k), u^*(k+1; k), \ldots, u^*(k+N-1; k) \} \) is the optimal control input sequence. The following standard assumption is made.

**Assumption 3:** For the system terminal state \( x \in \mathcal{X}_f \) and \( V_f = \|\bar{x}(k+N; k)\|_P^2 \), there exists a local stabilizing control policy \( \kappa(x) \in \mathcal{U} \) satisfying: \( V_f(f(x, \kappa(x))) - V_f(x) \leq -F(x, \kappa(x)) \), with \( F(x, \kappa(x)) = \|x\|_Q^2 + \|\kappa(x)\|_P^2 \).

Here, given the initial system state \( x(k) \) and a control sequence \( U(k) = \bar{u}(\cdot; k) \), the state sequence \( x(k+i; k) = \phi(i, U(k), x(k)) \) satisfies: \( \phi(i, U(k), x(k)) = f(\ldots f(x(k), \mu(0, U(k))), \ldots, \mu(i-1, U(k))) \), where \( i = 0, \ldots, N-1 \) and \( \mu(i, U(k)) = \bar{u}(k+i; k) \).

On this basis, we rewrite Problem P1 as the following compact form to facilitate discussions, Problem P2:

\[ U^*(k) = \arg\min_{U(k)} J(x(k), U(k)) \]

s.t. \( G(x(k), U(k)) \leq 0 \)

\[ \bar{x}(k+N; k) \in \mathcal{X}_f. \]
According to (2), $G(x(k), U(k))$ is defined as

$$G(x(k), U(k)) = \begin{pmatrix}
  g(\phi(0, U(k), x(k))) \\
  \vdots \\
  g(\phi(N-1, U(k), x(k))) \\
  \mu(0, U(k)) - u_{max} \\
  \vdots \\
  \mu(N-1, U(k)) - u_{max} \\
  u_{min} - \mu(0, U(k)) \\
  \vdots \\
  u_{min} - \mu(N-1, U(k))
\end{pmatrix}.$$ \quad (4)

C. Feedforward Neural Network

This article adopts feedforward neural networks (FNN), a type of widely used DNN model, to learn the control policy of NMPC. FNN provides an L-layer neural networks architecture, which forms a mapping from the input $z$ to the output $f_0(z)$. The detailed formulation is as follows: $f_0(z) = h_L(h_{L-1}(\ldots h_1(z)))$, with the function $h_i(z) = \delta(\omega^i z + b^i)$. Here, $\delta$ is the activation function of DNN. To fitting a function $f(z) \in \mathcal{F}$ with the given training data $\{z^{(i)}, f(z^{(i)})\}_{i=1}^{M}$, the DNN optimizes its parameter $\theta := \{\omega^i, b^i\}_{i=1}^{L}$ by minimizing a designed loss function $L(\cdot)$.

III. POLICY LEARNING MPC

A. Policy Learning MPC

In the framework of NMPC, Problem $P2$ is recursively solved while the new system state $x(k)$ is obtained, and only the first element of $U^*(k)$ is obtained. Therefore, we can establish a function map between the system state $x(k)$ and $U^*(k)$ as follows: $U^*(k) = \pi(x(k))$. Different from the traditional NMPC algorithms utilizing the online optimization method, we use the DNN to fit $\pi(x(k))$ by $\pi_\theta(x(k))$ and denote $U_{\theta}(k) = \pi_\theta(x(k))$, where $U_{\theta}(k) = \{u_0(k; k), u_0(k + 1; k), \ldots, u_0(k + N - 1; k)\}$. As introduced in Section II, the parameters $\theta$ of DNN are $\{\omega^i, b^i\}_{i=1}^{L}$.

The loss function of deep learning is designed as

$$L(U^*(k), U_{\theta}(k)) = J(x(k), U_{\theta}(k)) - J(x(k), U^*(k)).$$ \quad (5)

It is straightforward that the loss function $L(U^*(k), U_{\theta}(k)) \geq 0$ because of the optimality property of $U^*(k)$. By utilizing the training dataset $\mathcal{D} = \{(x(k), U^*(k))\}_{k=1}^{M}$, the loss function $L(U^*(k), U_{\theta}(k))$, the supervised learning problem of $U_{\theta}(k)$ is converted into the following optimization problem:

$$\theta^* = \arg \min_{\theta} \sum_{k=1}^{M} L(U^*(k), U_{\theta}(k)).$$

Considering the constraints of system defined in (2), we obtain the following optimization problem, Problem $P3$:

$$\theta^* = \arg \min_{\theta} \sum_{k=1}^{M} L(U^*(k), U_{\theta}(k))$$

s.t. $G(x(k), U_{\theta}(k)) \leq 0.$

B. Dual Optimization Learning Algorithm

Traditional learning algorithms only use the unconstrained optimization problems to update $\theta$, which may lead to the failure of the learned control policy in practical implementation because of violating constraints. To deal with this problem, we fulfill the system state and input constraints in $P3$ by means of Lagrange duality theory [24].

First, an augmented Lagrangian loss function is defined as follows: $\mathcal{L}(U^*(k), U_{\theta}(k), v) = L(U^*(k), U_{\theta}(k)) + v[G(x(k), U_{\theta}(k))]^2$, where $[x^2]^2$ denotes the function $\max(x, 0)$ and $v = [v_1, \ldots, v_N]$ is the dual variable associated with the constraint $G(x(k), U_{\theta}(k))$.

Then, the dual form of problem $P3$ can be converted to Problem $P4$

$$\theta^*(v) = \arg \min_{\theta} \sum_{k=1}^{M} \mathcal{L}(U^*(k), U_{\theta}(k), v)$$

$$v^* = \arg \max_{v} \min_{\theta} \sum_{k=1}^{M} L(U^*(k), U_{\theta}(k), v).$$

Finally, for Problem $P4$, we integrate the dual gradient descent algorithm and back propagation algorithm to solve it. The concrete steps are described in Algorithm I.

Note that $v$ and $\theta$ are designed as an alternative update process for Algorithm I. The procedure will be finished once the stopping criteria are satisfied, which means that $v$ may not converge to the optimal value at the end of the iteration. Therefore, although the existence of the dual variable $v$ encourages the parameter $\theta$ to converge to the approximate optimal value while satisfying the constraints in the process of updating, the nonoptimal value $v$ would lead to a slight violation of the constraints in probability. This issue will be further discussed and tackled in Section IV-A.

C. PL-MPC Algorithm

The approximate control policy $\pi_\theta(x(k))$ is learned through the training dataset. In order to have a more comprehensive learning control policy, we need training samples that can cover the whole feasible region of MPC.

We use the rejection random sampling algorithm to sample the initial state $x(0)$ of control trajectories in the feasible region. In order to reduce the similarity of the samplings and improve the coverage of samples in the feasible region, we determine whether to adopt the new sample by comparing the distance between the new sample $x^k(0)$ and the existing one.

Assuming that there are $h$ existing samples $\{x^1(0), x^2(0), \ldots, x^h(0)\}$, their center of gravity can be expressed as: $x^{(0)}(0) = 1/h \sum_{i=1}^{h} x^{(i)}(0)$. We can get the distance from the new sampling point $x^k(0)$ to the center of gravity $x^{(0)}(0)$ as: $\text{dis} = \|x^k(0) - x^{(0)}(0)\|_2$. If $\text{dis} > \tau_k$ is satisfied, $x^k(0)$ is taken as the new initial point, otherwise the initial point will be sampled again. The minimum distance $\tau_k$ decreases with the increase of sampling number $k$ as $\tau_k = \gamma^{1/k} \tau$. Here, $\tau > 0$ is a constant value used.
Algorithm 1: Dual Optimization Learning Algorithm.

Require: Initial training dataset \( \mathcal{D} = \{(x(k), U^*(k))\}_{k=1}^{\mathcal{D}} \), initial DNN parameter \( \theta \), initial optimal step size of DNN \( \varepsilon \), initial dual variable \( v \), initial dual updated step size \( \alpha = (\alpha_0, \alpha_1, \ldots) \).

for each \( j = 0, 1, 2, \ldots \) do
  while Loss function gradient greater than expected do
    Randomly sample \( m \) small batch samples \( \{(x(1), U^*(1)), \ldots, (x(m), U^*(m))\} \in \mathcal{D} \).
    Calculate the output of the DNN \( U_{\theta}(k) = \pi_{\theta}(x_k) \).
    Calculate \( J(x(k), U_{\theta}(k)) \) and \( J(x(k), U^*(k)) \) using (1) and (3).
    Compute the Lagrangian dual variable \( v_i^{j+1} \)
    end while
  end for
end for

Algorithm 2: PL-MPC Algorithm.

Require: Initialize training data buffer.

Offline:
  Sample the initial state \( x(0) \).
  Obtain \( \tau(x(0)) \) by NMPC algorithm and store it.
  Collect batch data from data buffer and train the DNN.
  if The empirical error is less than the expected error then
    Stop training.
  else
    Train the approximate control policy \( \pi_{\theta} \).
    end if

Online:
  for \( k = 0, 1, 2, \ldots \) do
    Obtain the state of the controlled object \( x(k) \).
    if \( x(k) \in X_0 \) then
      Set control input \( u(k) = \kappa(x(k)) \).
    else
      Set \( N \) control sequences \( U(k) = \bar{u}(\cdot; k) = \pi_{\theta}(x(k)) \).
      Set control input \( u(k) = \bar{u}(k; k) \).
    end if
  end for

IV. FEASIBILITY HANDLING AND STABILITY ANALYSIS

A. Feasibility Handling

This subsection deals with the constraint satisfaction issue of Problem P3 to ensure the feasibility of the proposed Algorithm 2. As mentioned at the end of Section III-B, although the dual optimization learning algorithm provides an effective method to handle the constraints of PL-MPC, there exists a small chance that the approximate solution may violate constraints. In this particular case, we use the proximal operator to guarantee the feasibility of the PL-MPC algorithm.

By adding a projection mapping layer on the basis of the original DNN, the proximal operator is able to translate the approximate solution \( U_{\theta}(k) \) generated by DNN into a high-quality feasible solution \( U_{\hat{\theta}}(k) \). The projection mapping layer is designed as follows, Problem P5:

\[
\arg \min_{U_{\hat{\theta}}(k)} \left\| U_{\theta}(k) - U_{\hat{\theta}}(k) \right\|_2 \\
\text{s.t.} \quad G(x(k), U_{\theta}(k)) \leq 0.
\]  

(6)

Generally, \( U_{\hat{\theta}}(k) \) belongs to the feasible region by using the dual optimization learning algorithm, i.e., \( U_{\hat{\theta}}(k) = U_{\hat{\theta}}(k) \). When the approximate solution \( U_{\hat{\theta}}(k) \) does not exist in the feasible region, we can obtain the corresponding feasible solution \( U_{\hat{\theta}}(k) \) by solving the optimization problem P5.

Here, the problem P5 is solved through exterior point penalty function methods. Define a penalty function as: \( Q(U_{\theta}(k), \mu) = \left\| U_{\theta}(k) - U_{\hat{\theta}}(k) \right\|_2 + \mu \left( G(x(k), U_{\theta}(k)) \right)^+ \), where \( \mu > 0 \) is the penalty parameter and \( \left[ x \right]^+ \) denotes the function \( \max(x, 0) \). Obviously, as the increase of \( \mu \), the penalty
function $Q(U_{\theta}(k), \mu)$ will be far away from the optimal value $Q(U_{\theta}^*(k), \mu)$ if $U_{\theta}(k)$ violates the constraint $G(x(k), U_{\theta}(k)) \leq 0$, which makes the unconstrained problem arg min$_{U_{\theta}(k)} Q(U_{\theta}(k), \mu)$ equivalent to problem $P^3$. Now, by integrating the dual optimization learning algorithm and the proximal operator, there always exists a feasible solution for the problem $P^3$.

Remark 1: Note that after the preprocessing of $U_p(k)$ by the dual optimization learning algorithm, $U_p(k)$ is not far away from $U_{\theta}(k)$ in the case of constraint violations. Therefore, a small number of iterations are required to find $U_p(k)$, which can still generate a fast solution without much computation resources.

B. Stability Analysis

In this subsection, we develop the stability result of the closed-loop system for the proposed learning algorithm.

We start with considering the loss function in (5) with the learned control policy $U_{\theta}$. To improve the control performance of the proposed algorithm, we require the loss function to be less than a positive real number $\varepsilon > 0$

$$\mathcal{L}(U_{\theta}, U^*) < \varepsilon.$$  \hspace{1cm} (7)

In the policy learning procedure, due to the randomness of learning data and samples, the loss function $\mathcal{L}(U_{\theta}, U^*)$ is a random variable. In what follows, we develop a result ensuring (7) holds in probability.

Theorem 1: Suppose the training data input $x(k)$ satisfies the uniform distribution and the training data samples are sampled independently from this distribution. For the learned control policy $U_{\theta}$ generated by Algorithm 2 and the optimal one $U^*$, given $\delta \in (0, 1)$ and $\varepsilon > 0$, there exists a large number of training samples $M$, such that

$$P\{\mathcal{L}(U_{\theta}, U^*) < \varepsilon\} \geq 1 - \frac{R_{\text{emp}}(U_{\theta}, U^*)}{(1 - \sqrt{\frac{e}{M\delta}})\varepsilon} \hspace{1cm} (8)$$

where $R_{\text{emp}}(U_{\theta}, U^*)$ is the empirical error of $M$ training data samples defined as follows: $R_{\text{emp}}(U_{\theta}, U^*) = \frac{1}{M} \sum_{k=1}^{M} (J(x(k), U_{\theta}(k)) - J(x(k), U^*(k)))$, $c$ is a constant value satisfying $c \geq 1$.

Proof 1: Considering the nonnegativity of $\mathcal{L}(U_{\theta}, U^*)$ and applying the Markov’s inequality to it for $\varepsilon > 0$, we have

$$P\{\mathcal{L}(U_{\theta}, U^*) \geq \varepsilon\} \leq \frac{E[\mathcal{L}(U_{\theta}, U^*)]}{\varepsilon} \hspace{1cm} (9)$$

where $E[\mathcal{L}(U_{\theta}, U^*)]$ is the generalization error of the learning algorithm defined as: $E[\mathcal{L}(U_{\theta}, U^*)] = R_{\text{gen}}(U_{\theta}, U^*) = \int J(x, U_{\theta}) - J(x, U^*) dF(J(x, U_{\theta})|x)$. Where $F$ is the probability density function of the distribution $x(k)$. Then, by utilizing the inequality in (9), the probability of (7) is bounded by

$$P\{\mathcal{L}(U_{\theta}, U^*) < \varepsilon\} \geq 1 - \frac{R_{\text{gen}}(U_{\theta}, U^*)}{\varepsilon}.$$  \hspace{1cm} (10)

However, the generalization error $R_{\text{gen}}(U_{\theta}, U^*)$ of the learning algorithm usually cannot be determined. To deal with this issue, we derive the upper bound of $R_{\text{gen}}(U_{\theta}, U^*)$ in probability in the following.

First, we show that

$$\frac{E[\mathcal{L}^2(U_{\theta}, U^*)]}{\mathcal{E}^2[\mathcal{L}(U_{\theta}, U^*)]} \leq c.$$  \hspace{1cm} (11)

Because $\mathcal{L}(u_{\theta}(x), u^*(x))$ can be formulated as

$$V[\mathcal{L}(U_{\theta}, U^*)] = E[\mathcal{L}^2(U_{\theta}, U^*)] - \mathcal{E}^2[\mathcal{L}(U_{\theta}, U^*)] \hspace{1cm} (12)$$

it can be obtained that $c \geq 1$. Since the variance $V[\mathcal{L}(U_{\theta}, U^*)]$ is bounded and greater than zero, there exists a positive integer $c \geq 1$ such that the inequality in (11) holds. Since $\mathcal{L}(U_{\theta}, U^*) \geq 0$ and $x(k)$ satisfy a uniform distribution, we have: $E[\mathcal{L}^2(U_{\theta}, U^*)] \geq \frac{1}{2} E^2[\mathcal{L}(U_{\theta}, U^*)]$, $c$ can take the value $c = 2$. Then, substituting (11) into (12), we obtain

$$V[\mathcal{L}(U_{\theta}, U^*)] \leq (c - 1) R^2_{\text{gen}}(U_{\theta}, U^*). \hspace{1cm} (13)$$

Since the training sampling data $x_k$ to be independently, by using the inequality in (13), the upper bound of the variance of $R_{\text{emp}}(U_{\theta}, U^*)$ can be calculated as

$$V[R_{\text{emp}}(U_{\theta}, U^*)] = \frac{1}{M} V[\mathcal{L}(U_{\theta}, U^*)]$$

$$\leq \frac{c - 1}{M} R^2_{\text{gen}}(U_{\theta}, U^*). \hspace{1cm} (14)$$

In addition, the expectation of $R_{\text{emp}}(U_{\theta}, U^*)$ can be calculated as

$$E[R_{\text{emp}}(U_{\theta}, U^*)] = \frac{1}{M} \sum_{k=1}^{M} R_{\text{gen}}(U_{\theta}, U^*) \hspace{1cm} (15)$$

(15)

By combining the inequality in (14) and (15), we can get

$$P\{|R_{\text{emp}}(U_{\theta}, U^*) - R_{\text{gen}}(U_{\theta}, U^*)| < \sigma\} \geq 1 - \frac{(c - 1) R^2_{\text{gen}}(U_{\theta}, U^*)}{\sigma^2 M} \hspace{1cm} (16)$$

where the Chebyshev inequality is utilized.

Setting $\sigma$ as $\sigma = \sqrt{\frac{c - 1}{M \delta}} R_{\text{gen}}(U_{\theta}, U^*)$ and substituting it into (16), we have: $P\{|R_{\text{emp}}(U_{\theta}, U^*) - R_{\text{gen}}(U_{\theta}, U^*)| < \sqrt{\frac{c - 1}{M \delta}} R_{\text{gen}}(U_{\theta}, U^*)\} \geq 1 - \delta$.

Finally, considering that the empirical error is less than the generalization error [25], the upper bound of $R_{\text{gen}}(U_{\theta}, U^*)$ satisfies the following:

$$P\{R_{\text{gen}}(U_{\theta}, U^*) \leq \left(1 - \sqrt{\frac{c - 1}{M \delta}}\right)^{-1} R_{\text{emp}}(U_{\theta}, U^*)\} \geq 1 - \delta. \hspace{1cm} (17)$$

When $\delta \to 0$ and the number of training samples $M$ is large enough, the probability of $R_{\text{gen}}(U_{\theta}, U^*) \leq \left(1 - \sqrt{\frac{c - 1}{M \delta}}\right)^{-1} R_{\text{emp}}(U_{\theta}, U^*)$ approaches to 1. Applying (17) into (10), the inequality in (8) is derived, which completes the proof. □
Next, the asymptotic stability in probability of the closed-loop system with the learned policy is presented.

**Theorem 2:** Suppose that Assumptions 1–3 hold. For any state region $X_0 \subseteq X_f$, the closed-loop system (1) under the suboptimal control policy $U_\theta$ obtained by Algorithm 2 is asymptotically stable in probability.

**Proof 2:** Consider that the system state is $x(k)$, and the feasible suboptimal solution obtained from DNN is $U_\theta(k) = [u_\theta(k); k, \ldots, u_\theta(k + N - 1; k)]$. Then we can get the state of the next time instant $x(k + 1) = f(x(k), u_\theta(k); k)$. For state $x(k + 1)$, there is a feasible control input sequence $U_\theta(k + 1) = [u_\theta(k + 1; k), \ldots, u_\theta(k + N - 1; k), v]$ due to Assumptions 3, where $v = \kappa(\dot{x}(k + N; k))$. Then, we can get the following inequality:

$$J(x(k + 1), U^*(k + 1)) \leq J(x(k + 1), U_\theta(k + 1))$$

Using $\mathcal{L}(U_\theta, U^*) < \varepsilon < \max_{x \in X_0} ||x||_Q^2$. Then we can get $J(x, U_\theta) < J(x, U^*) + \max_{x \in X_\theta} ||x||_Q^2$. Since $||u(k)||_R^2 \geq 0$, the inequality (19) can be transformed into:

$$J(x(k + 1), U^*(k + 1)) - J(x(k), U_\theta(k)) \leq - (||x(k)||_Q^2 + ||u(k)||_R^2).$$

Using Theorem 1, when the sampling number $N$ is large enough, then there is small enough $\varepsilon > 0$, such that $\mathcal{L}(U_\theta, U^*) < \varepsilon \leq \max_{x \in X_0} ||x||_Q^2$. Then we can get $J(x, U_\theta) < J(x, U^*) + \max_{x \in X_\theta} ||x||_Q^2$. Since $||u(k)||_R^2 \geq 0$, the inequality (19) can be transformed into:

$$J(x(k + 1), U^*(k + 1)) - J(x(k), U_\theta(k)) \leq - (||x(k)||_Q^2 + ||u(k)||_R^2).$$

Using (19), when the empirical error tend to zero. It follows that, $P(\mathcal{L}(U_\theta, U^*) < \max_{x \in X_\theta} ||x||_Q^2) \rightarrow 1$. As a result, the probability that the system is asymptotically stable will also be close to 1. The proof is completed.

$$\Delta x_k = \begin{bmatrix} \cos \psi_k & -\sin \psi_k & 0 \\ \sin \psi_k & \cos \psi_k & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} u_k \\ v_k \\ r_k \end{bmatrix}$$

$$\Delta u_k = \frac{(m_{22}v_k r_k - d_{11} u_k + F)/m_{11}}{m_{22}}$$

$$\Delta v_k = \frac{(-m_{11} u_k r_k - d_{22} v_k)/m_{22}}{m_{22}}$$

Here, the state $[x_k, y_k, \psi_k]^T$ represents the x-axis position, y-axis position, and orientation in the n-frame, respectively. The state $[u_k, v_k, r_k]^T$ is x-axis velocity, y-axis velocity, and angular velocity in the b-frame. The control input $[F, M]^T$ denotes the force and the torque generated by the thrusters. On this basis, according to the dynamic model (21), the velocity state of USVs is generated, and then the kinematic model (20) updates the position state.

By practical system identification of the USVs, the system control inputs (i.e., the torque and the force limits of the two thrusters) are constrained by $-19.6 N \leq F \leq 39.2 N$ and $-5 N m \leq M \leq 5 N m$, respectively. The system state is constrained by $|x| \leq 70 m, |y| \leq 70 m, -1 m/s \leq u \leq 2 m/s, |\psi| \leq 1 m/s$ and $|r| \leq 0.2 r/ds$. $m_{11} = 493.8, m_{22} = 455.8, m_{33} = 55.8$ are the diagonal elements of the inertia matrix. $d_{11} = 29.2, d_{22} = 2173.7, d_{33} = 17.7$ are the diagonal elements of the damping matrix.

**B. Learning NMPC**

The training dataset is generated by the traditional NMPC algorithm. The detailed parameters are as follows: $Q = \text{diag}(10, 10, 10, 0.1, 0.1, 0.1)$, $R = \text{diag}(0.01, 0.2)$ and $P = \text{diag}(10, 10, 20, 0.1, 0.1, 0.1, 0.1)$, respectively. The prediction horizon is $N = 15$. For such an underactuated USV system, the specific design of the terminal region and terminal control law $\kappa(x)$ can refer to [27]. The simulation nonlinear optimization problem of NMPC is solved by CasADi [28].

For DNN, the node of the input layer is set as 6, and the node of the output layer is 2. There are five layers in the hidden layer. Each layer has 150, 250, 250, 250, and 50 nodes, and the activation function is chosen as the ReLU function. We set $\delta = 0.01, \varepsilon = 0.05$ and let $D$ contain $10^6$ training data, i.e., $M = 10^6$.

To verify the learning effect, we apply the PL-MPC algorithm to the simulation model (20)–(21) and compare it with the NMPC algorithm. Fig. 1(a) and (b) compares the system state with the initial system state $x(0) = [-64, -64, 0, 0, 0, 0]^T$ curves of the NMPC algorithm and the proposed PL-MPC algorithm. It can be observed that the difference of the two algorithms is very small. Fig. 2(a) and (b) shows the control input $F$ of the NMPC algorithm and the PL-MPC algorithm, from which it can be seen that the two algorithms have almost the same control input.

Simulations of PL-MPC and NMPC with different prediction horizons are also carried out for comparison in the same environment (CPU i7-8550 U). By choosing different prediction

V. IMPLEMENTATION TO USVS

A. USV DYNAMICS

In this section, we implement the proposed PL-MPC algorithm for an underactuated USV.

To generate training data and validate performance of the proposed algorithm, we identify the model for the USV. The USV is driven by two rear thrusters, which are $T_r$ and $T_r$, respectively. The control input $F = T_r + T_r$, $M = (T_r - T_r) \cdot d_t$, where $d_t$ is axis distance of USV equals to 0.277 m. The state space of the USV is from the north-east-down frame (n-frame) to the body frame (b-frame). It has the nonlinear discrete-time system dynamics as follows:
horizons of MPC. Fig. 3 shows the average one-step computational time for NMPC and PL-MPC on a control trajectory (1500 step). It can be seen that, the PL-MPC algorithm greatly reduces the computation time compared with the NMPC algorithm, especially with the increase of the prediction horizon.

C. Hardware Implementation

In this section, we implement the designed PL-MPC algorithm on the USV platform and conduct the lake experiments. The experimental equipment is shown in Fig. 4. The USV platform used in the experiment has the same parameters as the simulation platform. In order to verify the effectiveness of the algorithm, we have conducted two experiments corresponding to two control missions: point stabilization and trajectory tracking.

The experimental parameters of point stabilization are the same as those of simulation experiments. By training the policy using the DNN offline with data generated by simulation and deploying it for the USV, we conduct the point stabilization experiments for USV in the lake. Fig. 5(a) is the trajectory of the USV point stabilization experiment on the map. Fig. 5(b) shows the USV position in the experiment. Under the control algorithm, the USV can move to the equilibrium point and remain stable. Fig. 6 shows the position error and velocity error. The root mean square (rms) values of position error are 0.151 and 0.442 m, and the rms value of angle error is 0.043 rd. The rms values of velocity error are 0.076 m/s, 0.019 m/s, and 0.036 rd/s, respectively. Note that there exist unavoidably disturbances from waves and winds in the lake when the experiments were conducted. Even though the unknown disturbances, the PL-MPC controller has achieved satisfactory control performance in terms of the relatively small errors in real time experiments.

For trajectory tracking experiments, we design a reference trajectory \( (x_d(k), u_d(k)) \) as a circle. The reference trajectory and USV share the same kinetic model in (20)–(21), it is denoted as \( x_d(k+1) = f(x_d(k), u_d(k)) \). Here, \( x_d(k) = [x_d(k), y_d(k), \psi(k), u_d(k), v(k), r(k)]^T \) and \( u_d(k) = [F_d(k), M_d(k)]^T \). In this experiment, the reference force and moment are set as: \( F_d = 17.5 \) N and \( M_d = 1 \) Nm, respectively.
function optimization variables. Naturally, the input of the nonlinear discrete-time system with state and control inputs control input and state constraints. The sufficient conditions for ensuring the closed-loop stability had also been developed. The hardware experiment for the motion control of USV had verified the feasibility and advantages of the proposed method.

VI. CONCLUSION

In this article, we had developed an PL-MPC method for nonlinear discrete-time system with state and control inputs constraints, and implemented it successfully in the motion control of USVs. The DNN was trained to approximate NMPC control policy to accelerate the computational speed. By this method, we had designed the dual optimization learning algorithm, making the approximation control policies satisfied the control input and state constraints. The sufficient conditions for ensuring the closed-loop stability had also been developed. The hardware experiment for the motion control of USV had verified the feasibility and advantages of the proposed method.

REFERENCES

[1] D. Mayne, J. Rawlings, C. Rao, and P. Scokaert, “Constrained model predictive control: Stability and optimality,” Automatica, vol. 36, no. 6, pp. 789–814, Jun. 2000.
[2] H. Liang, H. Li, and D. Xu, “Nonlinear model predictive trajectory tracking control of underactuated marine vehicles: Theory and experiment,” IEEE Trans. Ind. Electron., vol. 68, no. 5, pp. 4238–4248, May 2021.
[3] D. Wang, Q. Pan, Y. Shi, J. Hu, and C. Zhao, “Efficient nonlinear model predictive control for quadrotor trajectory tracking: Algorithms and experiment,” IEEE Trans. Cybern., vol. 51, no. 10, pp. 5057–5068, Oct. 2021.
[4] F. Wang, L. He, J. Kang, R. Kenna, and J. Rodríguez, “Adaptive model predictive current control for PMSM drive system,” IEEE Trans. Ind. Electron., vol. 70, no. 4, pp. 3493–3502, Apr. 2023.
[5] Y. Shi, C. Shen, H. Fang, and H. Li, “Advanced control in marine mechatronic systems: A survey,” IEEE/ASME Trans. Mechatron., vol. 22, no. 3, pp. 1121–1131, Jan. 2017.
[6] J. Di Carlo, P. M. Wensing, B. Katz, G. Blelt, and S. Kim, “Dynamic locomotion in the MIT cheetah 3 through convex model-predictive control,” in Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst., 2018, pp. 1–9.
[7] H. J. Ferreau, H. G. Bock, and M. Diehl, “An online active set strategy to overcome the limitations of explicit MPC,” Int. J. Robust Nonlinear Control, vol. 18, no. 8, pp. 816–830, 2008.
[8] P. Tondel, T. A. Johansen, and A. Bemporad, “An algorithm for multi-parametric quadratic programming and explicit MPC solutions,” Automatica, vol. 39, no. 3, pp. 489–497, Mar. 2003.
[9] C. Wen, X. Ma, and B. Erik Ydstie, “Analytical expression of explicit MPC solution via lattice piecewise-affine function,” Automatica, vol. 45, no. 4, pp. 910–917, Apr. 2009.
[10] E. C. Kerrigan, “Predictive control for linear and hybrid systems,” IEEE Control Syst. Mag., vol. 38, no. 2, pp. 94–96, Apr. 2018.
[11] L. Xi, X. Wang, L. Jiao, S. Lai, Z. Peng, and B. M. Chen, “GTO-MPC-based target chasing using a quadrotor in cluttered environments,” IEEE Trans. Ind. Electron., vol. 69, no. 6, pp. 6026–6035, Jun. 2022.
[12] A. Aswani, H. Gonzalez, S. S. Sastry, and C. Tomlin, “Provably safe and robust learning-based model predictive control,” Automatica, vol. 49, no. 5, pp. 1216–1226, May 2013.
[13] P. Bouffard, A. Aswani, and C. Tomlin, “Learning-based model predictive control on a quadrotor: Onboard implementation and experimental results,” in Proc. IEEE Int. Conf. Robot. Autom., 2012, pp. 279–284.
[14] D. Piga, M. Forgione, S. Formentin, and A. Bemporad, “Performance-oriented model learning for data-driven MPC design,” IEEE Contr. Syst. Lett., vol. 3, no. 3, pp. 577–582, Jul. 2019.
[15] S. Gros and M. Zanon, “Data-driven economic NMPC using reinforcement learning,” IEEE Trans. Autom. Control, vol. 65, no. 2, pp. 636–648, Feb. 2020.
[16] U. Rosolia and F. Borrelli, “Learning model predictive control for iterative tasks: a data-driven control framework,” IEEE Trans. Autom. Control, vol. 63, no. 7, pp. 1883–1896, Jul. 2018.
[17] M. Menner, P. Worsnop, and M. N. Zeilinger, “Constrained inverse optimal control with application to a human manipulation task,” IEEE Trans. Control Syst. Technol., vol. 29, no. 2, pp. 826–834, Mar. 2021.
[18] T. Zhang, G. Kahn, S. Levine, and P. Abbeel, “Learning deep control policies for autonomous aerial vehicles with MPC-guided policy search,” in Proc. IEEE Int. Conf. Robot. Autom., 2016, pp. 528–535.
[19] D. Wang et al., “Model predictive control using artificial neural network for power converters,” IEEE Trans. Ind. Electron., vol. 69, no. 4, pp. 3689–3699, Apr. 2022.
[20] X. Zhang, M. Bujarbaruah, and F. Borrelli, “Near-optimal rapid MPC using neural networks: A primal-dual policy learning framework,” IEEE Trans. Control Syst. Technol., vol. 29, no. 5, pp. 2102–2114, Sep. 2021.
[21] S. W. Chen, T. Wang, N. Atanasov, V. Kumar, and M. Morari, “Large scale model predictive control with neural networks and primal active sets,” Automatica, vol. 135, Jan. 2022, Art. no. 109947.

[22] M. Hertneck, J. Köhler, S. Trimpe, and F. Allgöwer, “Learning an approximate model predictive controller with guarantees,” IEEE Contr. Syst. Lett., vol. 2, no. 3, pp. 543–548, Jul. 2018.

[23] H. Chen and F. Allgöwer, “A quasi-infinite horizon nonlinear model predictive control scheme with guaranteed stability,” Automatica, vol. 34, no. 10, pp. 1205–1217, Oct. 1998.

[24] F. Fioretto, P. Van Hentenryck, T. W. K. Mak, C. Tran, F. Baldo, and M. Lombardi, “Lagrangian duality for constrained deep learning,” in Proc. Mach. Learn. Knowl. Discov. Databases Appl. Data Sci. Demo Track, 2021, pp. 118–135.

[25] R. M. Kil and I. Koo, “Generalization bounds for the regression of real-valued functions,” in Proc. 9th Int. Conf. Neural Inf. Process., 2002, pp. 1766–1770.

[26] T. A. Johansen, “Approximate explicit receding horizon control of constrained nonlinear systems,” Automatica, vol. 40, no. 2, pp. 293–300, Feb. 2004.

[27] H. Li and W. Yan, “Model predictive stabilization of constrained underactuated autonomous underwater vehicles with guaranteed feasibility and stability,” IEEE/ASME Trans. Mechatron., vol. 22, no. 3, pp. 1185–1194, Jun. 2017.

[28] J. A. E. Andersson, J. Gillis, G. Horn, J. B. Rawlings, and M. Diehl, “CasADI: A software framework for nonlinear optimization and optimal control,” Math. Program. Comput., vol. 11, pp. 1–36, Mar. 2019.

Rizhong Wang received the B.Sc. degree in automation and control in 2017 from Northwestern Polytechnical University, Xi’an, China, where he is currently working toward the Ph.D. degree in intelligent decision and control with the School of Marine Science and Technology. His research interests include multiagent reinforcement learning and model predictive control.

Bin Liang (Senior Member, IEEE) received the B.S. and M.S. degrees in control engineering from Northwestern Polytechnical University, Xi’an, China, in 1991 and 1994, respectively, and the Ph.D. degree in precision instruments and mechatronics from Tsinghua University, Beijing, China, in 1994.

He is currently a Professor with the Department of Automation, Tsinghua University. His research interests include space robotics, manipulators, and intelligent control.

Huiping Li (Senior Member, IEEE) received the B.Sc. and M.Sc. degrees in mechanical engineering and automatic control from Northwestern Polytechnical University, Xi’an, China, in 2006 and 2009, respectively, and the Ph.D. degree in mechanical engineering from the University of Victoria, BC, Canada, in 2013. From May to October 2013, he was a Postdoctoral Research Fellow with the Department of Mechanical Engineering, University of Victoria. Since November 2013, he has been with the School of Marine Science and Technology, Northwestern Polytechnical University, where he is currently a Full Professor. His current research interests include model predictive control, multiagent systems, and distributed and cooperative control of underwater vehicles.

Dr. Li was a Guest Technical Editor for a Special Section in IEEE/ASME TRANSACTIONS ON MECHATRONICS. He serves as the Associate Editor for IEEE TRANSACTIONS ON INDUSTRIAL ELECTRONICS, IEEE TRANSACTIONS ON INDUSTRIAL INFORMATICS, and ASME Journal on Dynamic Systems, Measurement and Control.

Yang Shi (Fellow, IEEE) received the Ph.D. degree in electrical and computer engineering from the University of Alberta, Edmonton, AB, Canada, in 2005.

From 2005 to 2009, he was an Assistant Professor and an Associate Professor with the Department of Mechanical Engineering, University of Saskatchewan, Saskatoon, SK, Canada. In 2009, he joined the University of Victoria, Victoria, BC, Canada, where he is currently a Professor with the Department of Mechanical Engineering. His current research interests include networked and distributed systems, model predictive control (MPC), cyber-physical systems (CPS), robotics and mechatronics, navigation and control of autonomous systems (AUV and UAV), and energy system applications.

Dr. Shi was the recipient of the University of Saskatchewan Student Union Teaching Excellence Award in 2007, the Faculty of Engineering Teaching Excellence Award in 2012 at the University of Victoria (UVic), the JSPS Invitation Fellowship (short-term) in 2013, the UVic Craigdarroch Silver Medal for Excellence in Research in 2015, the 2016 IEEE TRANSACTIONS ON FUZZY SYSTEMS Outstanding Paper Award, and the Humboldt Research Fellowship for Experienced Researchers in 2018. He is the VP for Conference Activities of IES (2022–2023) and was a member of the IES Fellow Evaluation Committee during 2017–2019; he is the Chair of IEEE IES Technical Committee on Industrial Cyber-Physical Systems. He is currently the Coeditor-in-Chief of IEEE TRANSACTIONS ON INDUSTRIAL ELECTRONICS; he is also an Associate Editor for Automatica, IEEE TRANSACTIONS ON AUTOMATIC CONTROL, etc. He is the General Chair of the 2019 International Symposium on Industrial Electronics (ISIE) and the 2021 International Conference on Industrial Cyber-Physical Systems (ICCFPS). He is a Fellow of ASME, Engineering Institute of Canada (ElC), and Canadian Society for Mechanical Engineering (CSME), and a Registered Professional Engineer in British Columbia, Canada.

Demin Xu received the M.Sc. degree in mechanical engineering and automatic control from Northwestern Polytechnical University, Xi’an, China, in 1964.

Since 1964, he has been with the School of Marine Science and Technology, Northwestern Polytechnical University, where he is currently a Full Professor. He is also an Academician of the Chinese Academy of Engineering, Division of Mechanical and Vehicle Engineering. His current research interests include advanced control theory and control of underwater vehicles.