2025 IEEE Conference on Control Technology and Applications (CCTA)
August 25 - 27, 2025. San Diego, CA, USA

Safety Factor Profile Control in EAST via Reinforcement-Learning-based Model Predictive Control*

Zibo Wang, Sai Tej Paruchuri, and Eugenio Schuster

Abstract— **A tokamak is a toroidal device that utilizes helical magnetic fields to confine a superheated plasma. The spatial distribution of the pitch of the magnetic field, referred to as the safety factor profile, is linked to the stability and performance of the confined plasma. Thus, the capability to control the safety factor profile is essential for realizing advanced operation scenarios in tokamaks. This work proposes a Reinforcement Learning-based Model Predictive Control (RLMPC) approach for the enhanced regulation of the safety factor profile in the Experimental Advanced Superconducting Tokamak (EAST). By estimating in real time the uncertain parameters of the plasma model, the RLMPC method aims to achieve more accurate and robust tracking of the target profile. Besides learning the parameterized control-oriented response model, which serves as a linear constraint in the MPC problem, the reinforcement learning-based algorithm also learns the weight associated with the cost function. Swift convergence of the learnable parameters within the RLMPC is facilitated through a second-order Least Squares Temporal Difference Q-learning algorithm. Simulations based on the Control Oriented Transport SIMulator (COTSIM) show that the proposed RLMPC performs better than conventional linear MPC schemes.**

I. INTRODUCTION

The tokamak, an axisymmetric device that uses strong magnetic fields to confine the plasma, is considered one of the most feasible approaches to achieving nuclear fusion. However, the viability of commercial fusion-energy generation may depend on the ability to operate tokamaks in Advanced Tokamak (AT) scenarios. These scenarios, characterized by superior confinement and enhanced stability, rely on meticulous control of plasma properties such as the safety factor (denoted simply as q) profile, which measures the spatial variation of the magnetic field pitch. Achieving optimal q profiles, and thereby AT scenarios, requires non-inductive heating and current drive systems such as neutral beam injection (NBI) and lower hybrid wave (LHW).

Model Predictive Control (MPC) algorithms for the regulation of the q profile [1], [2], [3], [4] have been studied and experimentally tested recently. The ability to incorporate state and input constraints makes MPC algorithms preferable over other control approaches. However, as with any model-based control method, the performance of the MPC solution is linked to the underlying response model, which should balance accuracy and complexity. A too-complex model could make both design and implementation intractable. A too-simple model, failing to capture the critical plasma dynamics, could make the solution irrelevant.

In response to these limitations and challenges, reinforcement learning (RL) has emerged as a promising alternative in recent years [5], [6], [7]. RL methods enable exploration and decision-making in unknown or evolving environments without explicit reliance on system models. However, offline-trained RL controllers are vulnerable to environmental shifts. Their performance hinges on the assumption that the environment they encounter will closely resemble the training environment. As an alternative, MPC can be integrated with a real-time reinforcement learning (RL) framework to enhance MPC's applicability to complex systems with evolving operating conditions and improve control performance. This approach is generally called RL-based MPC. This hybrid approach leverages RL's ability to learn from data and adjust in real time while retaining the advantages of MPC's optimal decision-making over a predictive horizon. For instance, the RL framework can assist MPC by learning the terminal cost of MPC [8], or adjusting constraint and model parameters [9], [10], [11]. Previous work like [12], [13] demonstrated the effectiveness of RL-based MPC in improving trajectory tracking for unmanned surface vehicles. RL-based MPC has also been used to estimate model uncertainties and effectively enhance controllers' robustness [14].

The main contribution of this work is the synthesis of an RL-based MPC to regulate the safety factor profile in the Experimental Advanced Superconducting Tokamak (EAST). The novel approach uses an RL algorithm (Least Squares Temporal Difference (LSTD) Q-learning algorithm) [15] to estimate and update the underlying uncertain parameters in the system model as well as the weight associated with the initial cost in the to-be-minimized cost function within the MPC scheme. The proposed control solution consists of two components: 1- system identification using first-order LSTD Q-learning; 2- solution of the RL-based MPC using second-order LSTD Q-learning [11]. Both components are described in detail in Section IV. The system-identification algorithm based on LSTD Q-learning updates the parameters of a linearized model by using input-output data generated by a fast nonlinear plasma simulator, named COTSIM [16]. The goal of this system-identification step is to re-linearize the model around the actual trajectory and provide an initial model constraint for the subsequent MPC step. During the solution of the RL-based MPC problem, the LSTD Q-learning algorithm updates the MPC parameters (linearized model constraint and initial cost modifier) by interacting directly with the plant. While this RL-MPC framework has

*This material is based upon work supported by the U.S. Department of Energy, Office of Science, Office of Fusion Energy Sciences (FES), under Award Number DE-SC0010537.

Z. Wang (zibo.wang@lehigh.edu), S. T. Paruchuri, and E. Schuster are with the Department of Mechanical Engineering and Mechanics, Lehigh University, Bethlehem, PA 18015, USA.

2025 IEEE Conference on Control Technology and Applications (CCTA) | 979-8-3315-3908-5/25/$31.00 ©2025 IEEE | DOI: 10.1109/CCTA53793.2025.11151443

the potential to be applied in real-time control of a tokamak, in this work, the plant is simulated using COTSIM, and the RL-based MPC learns from data generated by COTSIM simulations. This learning occurs at every time step after solving a finite horizon optimal control problem, which is characteristic of an MPC approach. Thus, any changes in the tokamak operating conditions are incorporated into the plasma model in real time. COTSIM-based simulations demonstrate that the proposed RL-based MPC can perform better than a conventional linear MPC.

This paper is organized as follows. Section II introduces the COTSIM models, which are used as the simulator in this work. Section III summarizes the formulation of traditional linear MPC. Section IV introduces a system identification algorithm and an RL-based MPC framework. The system identification algorithm initializes the model parameters for the RL-based MPC, aiding in faster convergence. Section V presents simulation results comparing RL-based MPC and regular linear MPC methods. Conclusions and future work are discussed in Section VI.

II. RL Environment Models

In reinforcement learning, an agent learns from interactions with an environment by taking actions and receiving rewards. Due to the inherent complexity, operational cost, and machine safety, tokamaks cannot be used as environments for RL. Instead, reliable transport simulators that capture the tokamak plasma dynamics are used. In this work, the Control-Oriented Transport Simulator (COTSIM) is used as the RL and simulation environment. This section briefly describes the COTSIM environment used in this work. Detailed modeling components such as the one-dimensional magnetic diffusion equation (MDE) and electron heat transport equation (EHTE) are not included here but are described in [17], which also presents validation results.

As discussed in [17], the spatial variation of the pitch of the helical magnetic field in a tokamak is characterized by the safety factor q. The safety factor depends on the gradient of the poloidal stream function ψ. Thus, it is more convenient to work with the evolution of the poloidal flux gradient $\theta \triangleq \partial\psi/\partial\hat{\rho}$ for q profile control. As a result, the evolution of θ is

$$\frac{\partial\theta}{\partial t} = \underbrace{\frac{\partial}{\partial\hat{\rho}}\left[\frac{\eta(T_e)}{\mu_0\rho_b^2\hat{F}^2}\frac{1}{\hat{\rho}}\frac{\partial}{\partial\hat{\rho}}\left(\hat{\rho}D_\psi\theta\right)+R_0\hat{H}\eta(T_e)\frac{\langle\bar{j}_{NI}\cdot\bar{B}\rangle}{B_{\phi,0}}\right]}_{f_\theta}, \quad (1)$$

subject to the boundary conditions $\theta|_{\hat{\rho}=0}=0$ and $\theta|_{\hat{\rho}=1}=k_bI_p$, where I_p is the plasma current, μ_0 is the permeability in vacuum, R_0 is the major radius, η is the plasma resistivity, $\hat{\rho}$ is the mean effective minor radius, ρ_b is the mean effective minor radius of the last-closed plasma boundary, T_e is the electron temperature, $\bar{j}_{NI}$ is non-inductive current drive and $\hat{F}$, $\hat{G}$ and $\hat{H}$ are geometric factors [18] capturing the topology of the magnetohydrodynamic (MHD) equilibrium. The terms $D_\psi(\hat{\rho}) \triangleq \hat{F}(\hat{\rho})\hat{H}(\hat{\rho})\hat{G}(\hat{\rho})$ and $k_b \triangleq -\frac{\mu_0}{2\pi}\frac{R_0}{\hat{G}(1)\hat{H}(1)}$.

III. Linear MPC Formulation

The model for θ shown in (1) can be discretized at N_g spatial nodes $\hat{\rho}_0,\cdots,\hat{\rho}_{N_g}$ using the finite-difference method, resulting in a nonlinear ordinary differential equation that governs the evolution of $x(t)=\left[\theta_1,\cdots,\theta_{N_g-1}\right]^T$, where $\theta_n(t)=\theta(\hat{\rho}_n,t)$ and $n\in\{1,\cdots,N_g-1\}$, when driven by the physical inputs such as plasma current, NBI, and LHW. Let t_j represent the time at the j^{th} time step, where t_0 is the initial time, and Δt is the time interval between consecutive steps. The time t_j is given by the equation $t_j=t_0+j\times\Delta t$, where j is the index of the time step, starting from 0 for the initial time. Note that $x_j=x(t_j)$ and $u_j=u(t_j)$. As presented in [4], [19], the combination of a first-order Taylor series approximation and zero-order hold temporal discretization results in a discrete-time linear model:

$$\Delta x_{j+1}=\tilde{f}_\theta(\Delta x_j,\Delta u_j)=A\Delta x_j+B\Delta u_j, \quad (2)$$

where $\Delta x_j=x_j-x^{ref}$, $\Delta u_j=u_j-u^{ref}$, x^{ref} and u^{ref} are the reference state and input vectors used for linearization, respectively. The state and input matrices, A and B, respectively, are time-invariant.

Following [4], the MPC strategy employs the system dynamics described by (2) to predict the future behavior of the system over a specified prediction horizon (N_p). This can be mathematically formulated as

$$\min_{\Delta U_j} J=\sum_{i=j}^{j+N_p-1}\underbrace{\left\|\left(\Delta x_i-\Delta x_i^{tar}\right)\right\|_Q^2+\|\Delta u_i\|_R^2}_{\text{stage cost: } l(\Delta x_i,\Delta u_i)} \quad (3a)$$

$$\text{s.t. } \Delta x_{i+1}=\tilde{f}_\theta(\Delta x_i,\Delta u_i) \quad \forall i\in[j,\cdots,j+N_p-1], \quad (3b)$$

$$h(\Delta u_i)\preceq\mathbf{0}, \quad (3c)$$

where $\Delta x_i^{tar}\triangleq x_i^{tar}-x^{ref}$, $(\cdot)^{tar}$ represents the target trajectory, Q and R are positive-definite weighting matrices, and the operator $||\cdot||_W^2\triangleq(\cdot)^TW(\cdot)$. The fast quadratic programming (QP) solver implemented in [4] can be used to find the solution $\Delta U_j{}^*$ of the optimization problem (3). It is worth noting that the control input u_j^* implemented in the plant is obtained by extracting Δu_j^* from the solution ΔU_j^* of the optimization problem discussed above and then using the formula $u_j^*=\Delta u_j^*+u^{ref}$.

IV. Reinforcement learning-based MPC Design

Due to the uncertainty introduced via unmodeled plasma physics and model approximations, the linear MPC formulated above might be unable to accurately track a given q profile target in all tokamak scenarios. Learning-based MPC leverages machine learning's ability to learn the system dynamics. Thus, it could be more beneficial for controlling complex systems such as the one considered in this work. In the following analysis, a learning-based MPC algorithm for q profile control is designed by leveraging Q-learning [15], a foundational algorithm in reinforcement learning. Through iterative interactions with the environment, RL-based MPC continuously updates the nominal model to approximate the true plant dynamics in a Tokamak. This iterative learning

process aims to enhance the accuracy of the model, thereby improving the performance of the model-based control algorithm through more reliable feedback predictions.

Let a Markov Decision Process (MDP) be characterized by a state space $\mathscr{S}$, an action space $\mathscr{A}$, state transition dynamics $\mathbb{P}[s_+|s,a]$, and stage cost $L(s,a)$, where $s,s_+ \in \mathscr{S}$ are the current and next states, and $a \in \mathscr{A}$ is the action. In this work, the state and action are defined as $s = \Delta x$ and $a = \Delta u$, respectively. The action-value function $Q_*(s,a)$ and value function $V_*(s)$ associated with the optimal policy $\pi_*(s)$ are defined by the Bellman equations:

$$Q_*(s,a) = L(s,a) + \gamma \mathbb{E}_{\pi_*}[V_*(s_+)|s,a], \tag{4}$$

where the discount factor γ is set to 0.95 in this work, $V_*(s) = Q_*(s,\pi_*(s))$, and $\mathbb{E}_{\pi_*}[V_*(s_+)|s,a]$ represents the expected value of the value function of the next state, conditioned on taking action a in state s and following the optimal policy thereafter. Q-learning aims to approximate the true action-value function Q_* by its estimate Q_Θ, where Θ is a vector of parameters that must be learned. In Least Squares Temporal Difference (LSTD) Q-learning [20], [21], the parameters are chosen such that $||Q_*(s,a) - Q_\Theta(s,a)||^2$ is minimized. This is done by using the least-squares method to solve the Bellman equation (4) for the action-value function, where $Q_*(s,a)$ is the true action-value function, and $Q_\Theta(s,a)$ is its approximation using a linear function representation. Specifically, the action-value function is approximated as: $Q_\Theta(s,a) = \Theta^T \phi(s,a)$, where Θ is the vector of parameters to be learned, and $\phi(s,a) = [\Delta x, \Delta u]$ is the feature vector corresponding to the state-action pair (s,a). This linear form is compatible with the structure of the parametric MPC, where the cost and dynamics are typically linear or quadratic, leading to a cost-to-go function that is naturally well-approximated by a linear Q-function. The following subsections discuss how LSTD Q-learning can be exploited to achieve RL-based model predictive control of the q profile.

A. System Identification

This section presents a system-identification strategy to estimate the evolution model of the system. The identified model is used as the initial value of the to-be-learned model in the RL-based MPC discussed in the next subsection. Suppose $\Delta x_j^{ob} = x_j^{ob} - x^{ref}$, where x_j^{ob} is the observed state of the simulation environment. Assume there exists a nonlinear function $\hat{f}_\theta$ that governs the evolution of Δx_j^{ob} by

$$\Delta x_{j+1}^{ob} = \hat{f}_\theta(\Delta x_j^{ob}, \Delta u_j). \tag{5}$$

Suppose that the function $\hat{f}_\theta$ can be linearized and parameterized with Θ_1 (i.e., $\Theta_1 = \{A_{\Theta_1}, B_{\Theta_1}, d_{\Theta_1}\}$) in the form

$$f_{\Theta_1}(\Delta x_j, \Delta u_j) = \Delta x_{j+1} = A_{\Theta_1}\Delta x_j + B_{\Theta_1}\Delta u_j + d_{\Theta_1}. \tag{6}$$

Thus, the objective of system identification is to minimize the difference between the observed state (Δx_{j+1}^{ob}) and the predicted state (Δx_{j+1}) given Δx_j and Δx_j^{ob}. In other words, the objective is to minimize the least squared state error e at time t_j between the parametric model and the observed state (i.e., $e_j = \left\| \Delta x_{j+1}^{ob} - \Delta x_{j+1} \right\|^2$).

To exploit the framework of LSTD Q-learning, define the parametric action-value function $Q_{\Theta_1}^{sys}(s,a)$ as:

$$Q_{\Theta_1}^{sys}(s_B, a_B) = \begin{bmatrix} f_{\Theta_1}(s_t, a_{t,1}) - \hat{f}_\theta(s_t, a_{t,1}) \\ f_{\Theta_1}(s_t, a_{t,2}) - \hat{f}_\theta(s_t, a_{t,2}) \\ \vdots \\ f_{\Theta_1}(s_t, a_{t,B}) - \hat{f}_\theta(s_t, a_{t,B}) \end{bmatrix}, \tag{7}$$

where the batched action vector $a_B = [a_{t,1}^T, \cdots, a_{t,N_B}^T]^T \in \mathbb{R}^{N_B \times 7}$, $a_{t,i} \in \mathscr{A}$, and N_B is the number of actions/batches. The batched state vector $s_B = [s_t^T, \cdots, s_t^T]^T \in \mathbb{R}^{N_B \times (N_g - 1)}$, and $s_t \in \mathscr{S}$. Note that in this case, Q_*^{sys}, the true action-value function, is equal to zero since it characterizes the "optimal" condition where f_{Θ_1} matches $\hat{f}_\theta$. The objective is to minimize the least squares of the Bellman residual error w.r.t Θ_1, i.e.,

$$\min_{\Theta_1} \mathbb{E}\left[\| \cancelto{0}{Q_*^{sys}(s_B, a_B)} - Q_{\Theta_1}^{sys}(s_B, a_B) \|^2 \right]. \tag{8}$$

To perform Q-learning, the first-order semi-gradient update law [15], which takes the form

$$\Theta_1 \leftarrow \Theta_1 + \alpha \delta^{sys} \nabla_{\Theta_1} Q_{\Theta_1}^{sys}(s_B, a_B) \tag{9}$$

is used. In the above update law, α is the learning rate, δ^{sys} is the temporal-difference error:

$$\delta^{sys} = Q_*^{sys}(s_B, a_B) - Q_{\Theta_1}^{sys}(s_B, a_B) = -Q_{\Theta_1}^{sys}(s_B, a_B). \tag{10}$$

To accelerate the convergence of the parameters, the initial guesses for $A_{\Theta_1} = A$ and $B_{\Theta_1} = B$ are selected based on the discrete-time linear model (2), while d_{Θ_1} is initialized as a zero vector.

Algorithm 1 System-identification Algorithm

```
n_B = 0, iter = 0
if t = 0 then
    A'_{Θ1} ← A, B'_{Θ1} ← B, d'_{Θ1} ← 0
    Randomly generate a_{t,B}  s.t.  h(0, a_{t,B}) ⪯ 0
end if
while n_B < N_B do
    n_B = n_B + 1
    while δ^{sys} > ε^{sys} or n_{iter} ≤ Max^{sys}_{iter} do
        n_{iter} = n_{iter} + 1
        A_{Θ1} ← A'_{Θ1}, B_{Θ1} ← B'_{Θ1}, d_{Θ1} ← d'_{Θ1};
        δ^{sys} ← f̂_θ(s_t, a_{t,n_B}) − f_{Θ1}(s_t, a_{t,n_B});
        Θ'_1 ← Θ_1 + α δ^{sys} (∇_{Θ1} δ^{sys});
    end while
end while
return A_{Θ1}, B_{Θ1}, d_{Θ1}
```

The pseudo-code for the corresponding iterative update mechanism is detailed in Algorithm 1. The algorithm initializes Θ_1 and randomly generates N_B actions that satisfy (3c). Then for each action, $\Theta_1 = \{A_{\Theta_1}, B_{\Theta_1}, d_{\Theta_1}\}$ is updated following (9) until tolerance ε^{sys} is satisfied or the maximum iteration number Max_{iter}^{sys} is exceeded.

B. Second-Order LSTD Q-Learning

This section introduces the LSTD Q-learning-based MPC algorithm, which primarily comprises two steps. The first step involves solving the finite-horizon optimal control problem associated with the MPC. The second step involves updating the MPC model based on observations using LSTD Q-learning. Unlike the system-identification algorithm presented earlier, the model update employs a second-order update law, which facilitates faster parameter convergence. It is important to note that the effectiveness of LSTD Q-learning depends on the accuracy of the initial parameter estimates. The system-identification method discussed earlier can be utilized to generate these initial parameter estimates, which can then serve as the starting point for the LSTD Q-learning-based MPC. The parametric version of the learning-based MPC can be written as

$$\min_{\Delta U_j} \; V^0_{\Theta_2} + J \tag{11a}$$

$$\text{s.t. } \Delta x_{i+1} = f_{\Theta_2}(\Delta x_i, \Delta u_i) \quad \forall i \in [j, \cdots, j+N_p-1], \tag{11b}$$

$$h(u_i) \preceq \mathbf{0}, \tag{11c}$$

where $f_{\Theta_2}(\Delta x_j, \Delta u_j) = A_{\Theta_2}\Delta x_j + B_{\Theta_2}\Delta u_j + d_{\Theta_2}$. In the above problem, the initial cost modifier $V^0_{\Theta_2}$ is a function of the initial state [9]. The term $V^0_{\Theta_2}(s)$ approximates the residual cost beyond the finite MPC horizon, thereby allowing the parametric Q-function to match the structure of the infinite-horizon return required by temporal-difference learning. Its inclusion is to ensure consistency with the Bellman equation and to avoid systematic underestimation of the value function. The parameters $\Theta_2 = \{V^0_{\Theta_2}, A_{\Theta_2}, B_{\Theta_2}, d_{\Theta_2}\}$ will be adjusted using LSTD Q-learning.

The parametric action-value function $Q^{mpc}_{\Theta_2}$ is defined as

$$Q^{mpc}_{\Theta_2}(s,a) = \min_{\Delta U_j} \; V^0_{\Theta_2} + J \tag{12a}$$

$$\text{s.t. } (11\text{b}) - (11\text{c}) \tag{12b}$$

$$\Delta u_j = a, \tag{12c}$$

which satisfies the fundamental equalities underlying the Bellman equation $V_{\Theta_2}(s) = \min_a Q^{mpc}_{\Theta_2}(s,a)$, where V_{Θ_2} is the value function estimate. The true action-value function Q^{mpc}_* takes the form

$$Q^{mpc}_*(s,a) \approx l(s,a) + \omega^T \max(0, h(a)) + \gamma V_{\Theta_2}(s_+), \tag{13}$$

where the weight ω penalizes constraint violation of (11c). The objective is to minimize the least squares of the Bellman residual error w.r.t Θ_2, i.e.,

$$\min_{\Theta_2} \mathbb{E}\left[\|Q^{mpc}_*(s,a) - Q^{mpc}_{\Theta_2}(s,a)\|^2\right]. \tag{14}$$

Instead of the classical semi-gradient Q-learning scheme given in (9), a second-order method, which yields a faster convergence, is implemented. The corresponding update law, that uses quasi-Newton steps, is given by

$$\Theta_2 \leftarrow \Theta_2 + \beta \delta^{mpc} H^{-1} \nabla_{\Theta_2} Q^{mpc}_{\Theta_2}(s,a), \tag{15}$$

where δ^{mpc} is the temporal-difference error

$$\delta^{mpc} = Q^{mpc}_*(s,a) - Q^{mpc}_{\Theta_2}(s,a), \tag{16}$$

$\beta > 0$ is the learning rate, and $H = \nabla^2_{\Theta_2}(Q^{mpc}_{\Theta_2})^2$ is the Hessian of the parametric action-value function. As proposed in [10], the Gauss-Newton Hessian approximation can be used to compute H:

$$H \approx (\nabla_{\Theta_2} Q^{mpc}_{\Theta_2}(s,a))^T \times \nabla_{\Theta_2} Q^{mpc}_{\Theta_2}(s,a). \tag{17}$$

The gradient and Hessian terms in (15) can be hard to compute. This computation can be simplified by computing the gradient of the Lagrangian $\mathscr{L}_{\Theta_2} = \Phi_{\Theta_2} + \lambda^T \mathrm{F}_{\Theta_2} + \nu^T \mathrm{H}_{\Theta_2}$, where Φ_{Θ_2} is the cost of the MPC optimization problem, λ is the Lagrange multiplier associated with the equality constraints F_{Θ_2} of (11b), ν is the Lagrange multiplier associated with the inequality constraints H_{Θ_2} of (11c). From [9],

$$\nabla_{\Theta_2} Q^{mpc}_{\Theta_2}(s,a) = \nabla_{\Theta_2} \mathscr{L}_{\Theta_2}(s, z^*), \tag{18}$$

where z^* is the primal-dual solution of (11). The Karush-Kuhn-Tucker (KKT) conditions [4] are used to find z^* and to ensure that this nonlinear optimization problem has a feasible solution. Thus, (17) can be rewritten as

$$H \approx (\nabla_{\Theta_2} \mathscr{L}_{\Theta_2}(s, z^*))^T \times \nabla_{\Theta_2} \mathscr{L}_{\Theta_2}(s, z^*). \tag{19}$$

The algorithm for this RL-based MPC strategy is outlined in Algorithm 2. At time t, the algorithm begins with an initialization of the iteration count n_{iter} and Θ_1. If $t = 0$, Algorithm1 is executed to initialize $A_{\Theta_1}, B_{\Theta_1}, d_{\Theta_1}$. Then, (11) is solved to find the optimal solution z^* corresponding to $\Theta_2 = \{V^0_{\Theta_2}, A_{\Theta_2}, B_{\Theta_2}, d_{\Theta_2}\}$. The TD error δ^{mpc} and Θ_2 are iteratively updated following (16) and (15), respectively, until tolerance ε^{mpc} is satisfied or the maximum iteration number Max^{mpc}_{iter} is exceeded.

Algorithm 2 RL-based MPC Algorithm

```
n_iter = 0
if t = 0 then
    Run Algorithm 1 and update A'_Θ1, B'_Θ1, d'_Θ1
end if
Initialize Θ'_2 : V^0'_Θ2, A'_Θ2 ← A_Θ1, B'_Θ2 ← B_Θ1, d'_Θ2 ← d_Θ1
while δ^mpc > ε^mpc or n_iter ≤ Max^mpc_iter do
    n_iter = n_iter + 1;
    V^0_Θ2 ← V^0'_Θ2, A_Θ2 ← A'_Θ2, B_Θ2 ← B'_Θ2, d_Θ2 ← d'_Θ2;
    Solve (11) for z*;
    Find δ^mpc by (16): δ^mpc ← Q^mpc_*(s,z*) − Q^mpc_Θ2(s,z*);
    Θ'_2 ← Θ_2 + β δ^mpc H^{-1} ∇_Θ2 L_Θ2(s,z*);
end while
return Θ_2, z*
```

V. SIMULATION RESULT AND DISCUSSION

To evaluate the performance of the proposed reinforcement learning-based MPC scheme, both linear MPC (3) and RL-based MPC are implemented and tested within the COTSIM environment. The total plasma discharge, i.e., the simulation duration, is 15 seconds. Initially, a feedforward simulation is

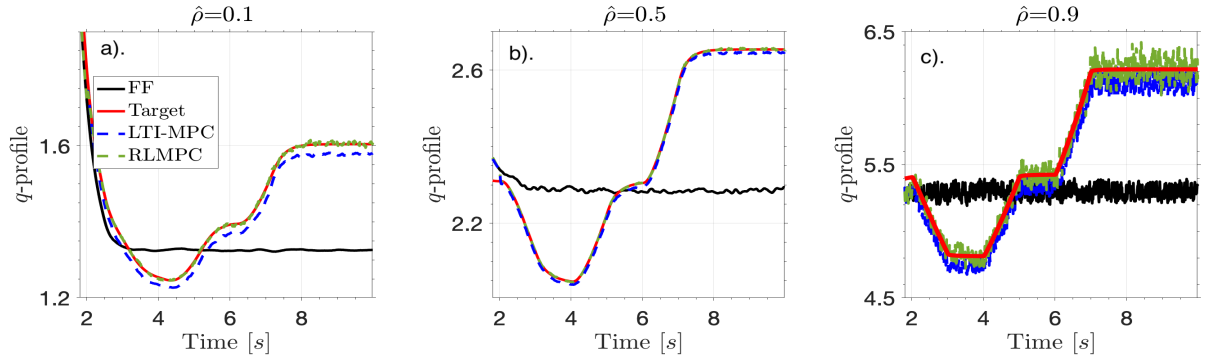

Fig. 1. Comparison of the q profile tracking results between linear MPC and RLMPC. Black solid lines: the feedforward trajectories generated with u_1; red solid lines: the target trajectories generated with u_2; blue dashed lines: the feedforward + feedback trajectories generated by the linear MPC algorithm; green dashed lines: the feedforward + feedback trajectories generated by the RL-based MPC algorithm.

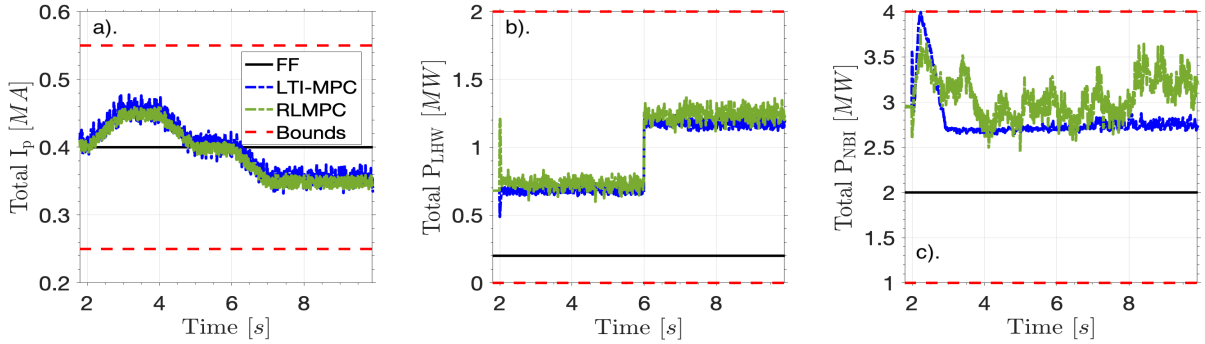

Fig. 2. Comparison of the actuation inputs prescribed by linear MPC and RLMPC. Black solid lines: the feedforward inputs corresponding to u_1; blue dashed lines: the inputs prescribed by the linear MPC algorithm; green dashed lines: the inputs prescribed by the RL-based MPC algorithm; red dotted lines: the upper and lower input constraints.

run with a specific input u_1 with Gaussian noise to generate a feedforward-only trajectory T_{ff}. A modified input u_2 without noise is then used in another feedforward simulation to create a feasible target trajectory T_{tar} for controller validation. For the feedforward + feedback simulation, the MPC controllers are active between 2-15 seconds. The feedback simulations are then carried out with the feedforward input u_1 to assess the ability of the controllers to steer the system towards the feasible target trajectory T_{tar}. Both the linear MPC and RL-based MPC configurations share the same set of parameters, including weight matrices Q and R, and horizon length $N_p = 10$, ensuring consistency between both control strategies.

Fig. 1(a)-(c) shows a comparative analysis of the q profile tracking results at different values of $\hat{\rho}$ between the two MPC algorithms. The feedforward-only trajectories are shown as solid black lines. The trajectories for the LTI MPC and the RL-based MPC are displayed by the blue and green dashed lines, respectively. The target trajectory is marked in the red solid line. As demonstrated in Fig. 1(a), the tracking performance of the linear MPC deteriorates over time. As evident, even after the time $q(\hat{\rho}=0.1)$ enters the steady state phase around 8 seconds, it exhibits a consistent tracking error attributed to model mismatch, which persists despite adjustments to MPC design parameters. In contrast, the RL-based MPC continuously accounts for the discrepancy between the nominal model and the environment, resulting in improved tracking performance compared to the linear MPC. The q values at $\hat{\rho} = 0.5$ and $\hat{\rho} = 0.9$ shown in the subsequent plots, Fig. 1(b) and (c), illustrate a similar behavior upon controller activation. Fig. 2(a)-(c) compares the actuator trajectories generated by the linear and RL-based MPC algorithms. The figure shows that the plasma current I_p and LHW power trajectories have a similar shape (but with an offset), and the NBI trajectories are different. As illustrated in Fig.2(a), the I_p computed by RL-based MPC is observed to be lower compared to that of linear MPC. This discrepancy in I_p computation contributes to the differences in the evolution of the safety factor $q(\hat{\rho}=0.9)$ between the proposed RL-based MPC and linear MPC, as shown in Fig.1(c). Specifically, the higher q profile observed at the plasma edge in Fig. 1(c) can be attributed to the lower I_p computed by RL-based MPC. This relationship is explained by the boundary condition of safety factor q. Additionally, as shown in Fig.2(b) and (c), the total power of LHW and NBI computed from the proposed RL-based MPC is higher than that of linear MPC, leading to higher q values at $\hat{\rho} = 0.1$ (Fig. 1(a)) and $\hat{\rho} = 0.5$ (Fig. 1(b)), and bringing them closer to the target trajectories.

During the simulation with RL-based MPC, the parameterized state-space system f_Θ is saved. The norms of the model error for both the linear MPC case (i.e., $e_{lmpc}(t_j) = \left\|\hat{f}_\theta(x_j,u_j) - \tilde{f}_\theta(x_j,u_j)\right\|^2$) and RL-based MPC case (6) (i.e.,

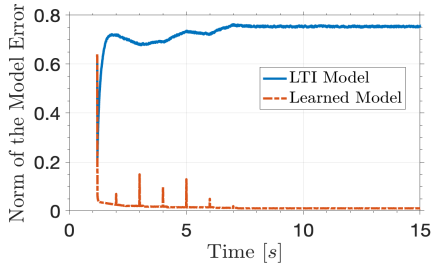

Fig. 3. Comparison of model error norms. Blue line: the norm of model error between the environment and the linear model used in LTI MPC; red dotted line: the norm of model error between the environment and the learned linear model.

$e_{rlmpc}(t_j) = \left\| \hat{f}_\theta(x_j,u_j) - f_\Theta(x_j,u_j) \right\|^2$) are plotted in Fig. 3. It indicates that the model mismatch for the linear model accumulates over time until it reaches a steady-state value, whereas the overall error of the learned model decreases.

While the RL-based MPC offers benefits in approximating the nonlinear evolution of the poloidal flux gradient as a linear function, its computational speed lags significantly behind that of the linear MPC. While there is room for execution-time optimization, the developed RL-based MPC requires approximately 30 times more computational effort than the linear MPC. This implies that real-time control of the q profile based on the RL-based MPC would require enhancement of present computational capabilities at EAST. This is indeed a general trend in machine learning, where the benefits of this type of approach are materialized thanks to recent progress in computing technologies.

VI. CONCLUSIONS AND FUTURE WORK

A reinforcement learning-based model predictive control framework has been developed in this work to tackle the q profile control problem in the EAST tokamak. The proposed algorithm integrates reinforcement learning by parameterizing the MPC problem. To address the discrepancies between the simplified linear model and the real nonlinear system, an online system-identification method is incorporated to learn the linearized equation of the environment through the first-order Temporal Difference learning method. The RL-based MPC scheme is applied to learn the system dynamics while accounting for modeling uncertainties and system noise. Uncertainty is measured by tracking the prediction error between the model's output and the actual output from the nonlinear system. The noise, assumed to follow a Gaussian distribution, is incorporated into the model to adjust parameters, ensuring the MPC remains robust against real-world disturbances. Through comparative analysis utilizing the COTSIM simulation environment, this study evaluates the performance of both the linear MPC and the RL-based MPC schemes. The RL-based MPC demonstrates enhanced closed-loop performance, effectively minimizing discrepancies between the environment and the model utilized, thereby outperforming the linear MPC approach. Future work includes the improvement of the computational efficiency of the RL-based MPC algorithm to facilitate real-time applicability in tokamaks.

REFERENCES

[1] H. Wang, W. P. Wehner, and E. Schuster, "Combined current profile and plasma energy control via model predictive control in the EAST tokamak," in *2018 26th Mediterranean Conference on Control and Automation (MED)*. IEEE, 2018, pp. 1–9.

[2] D. Moreau, S. Wang, E. Witrant, J. Qian, and Q. Yuan, "Model-predictive kinetic control experiments on EAST," in *Proc. 28th IAEA Fusion Energy Conf.* IAEA, 2021.

[3] S. T. Paruchuri, Z. Wang, T. Rafiq, and E. Schuster, "Model predictive current profile control in tokamaks by exploiting spatially moving electron cyclotron current drives," *Fusion Engineering and Design*, vol. 192, p. 113796, 2023.

[4] Z. Wang, H. Wang, E. Schuster, Z. Luo, Y. Huang, Q. Yuan, B. Xiao, D. Humphreys, and S. T. Paruchuri, "Implementation and initial testing of a model predictive controller for safety factor profile and energy regulation in the EAST tokamak," in *2023 American Control Conference (ACC)*. IEEE, 2023, pp. 3276–3281.

[5] J. Seo, Y.-S. Na, B. Kim, C. Lee, M. Park, S. Park, and Y. Lee, "Feedforward beta control in the KSTAR tokamak by deep reinforcement learning," *Nuclear Fusion*, vol. 61, no. 10, p. 106010, 2021.

[6] J. Degrave, F. Felici, J. Buchli, M. Neunert, B. Tracey, F. Carpanese, T. Ewalds, R. Hafner, A. Abdolmaleki, D. de Las Casas, *et al.*, "Magnetic control of tokamak plasmas through deep reinforcement learning," *Nature*, vol. 602, no. 7897, pp. 414–419, 2022.

[7] I. Char, J. Abbate, L. Bardóczi, M. Boyer, Y. Chung, R. Conlin, K. Erickson, V. Mehta, N. Richner, E. Kolemen, *et al.*, "Offline model-based reinforcement learning for tokamak control," in *Learning for Dynamics and Control Conference*. PMLR, 2023, pp. 1357–1372.

[8] M. Lin, Z. Sun, Y. Xia, and J. Zhang, "Reinforcement learning-based model predictive control for discrete-time systems," *IEEE Transactions on Neural Networks and Learning Systems*, 2023.

[9] S. Gros and M. Zanon, "Data-driven economic NMPC using reinforcement learning," *IEEE Transactions on Automatic Control*, vol. 65, no. 2, pp. 636–648, 2019.

[10] M. Zanon, S. Gros, and A. Bemporad, "Practical reinforcement learning of stabilizing economic MPC," in *2019 18th European Control Conference (ECC)*. IEEE, 2019, pp. 2258–2263.

[11] H. N. Esfahani, A. B. Kordabad, and S. Gros, "Approximate robust NMPC using reinforcement learning," in *2021 European Control Conference (ECC)*. IEEE, 2021, pp. 132–137.

[12] W. Cai, A. B. Kordabad, H. N. Esfahani, A. M. Lekkas, and S. Gros, "MPC-based reinforcement learning for a simplified freight mission of autonomous surface vehicles," in *2021 60th IEEE Conference on Decision and Control (CDC)*. IEEE, 2021, pp. 2990–2995.

[13] A. B. Martinsen, A. M. Lekkas, and S. Gros, "Reinforcement learning-based NMPC for tracking control of ASVs: Theory and experiments," *Control Engineering Practice*, vol. 120, p. 105024, 2022.

[14] J. F. Fisac, A. K. Akametalu, M. N. Zeilinger, S. Kaynama, J. Gillula, and C. J. Tomlin, "A general safety framework for learning-based control in uncertain robotic systems," *IEEE Transactions on Automatic Control*, vol. 64, no. 7, pp. 2737–2752, 2018.

[15] R. S. Sutton, "Reinforcement learning: An introduction," *A Bradford Book*, 2018.

[16] S. Morosohk, "Enhanced plasma profile estimation and control in tokamaks via machine learning," Ph.D. dissertation, Lehigh University, 2024.

[17] Z. Wang, E. Schuster, T. Rafiq, Y. Huang, Z. Luo, Q. Yuan, and J. Barr, "Enabling model-based scenario control in EAST by fast surrogate modeling within COTSIM," *Fusion Engineering and Design*, vol. 215, p. 114969, 2025.

[18] Y. Ou, T. Luce, E. Schuster, J. Ferron, M. Walker, C. Xu, and D. Humphreys, "Towards model-based current profile control at DIII-D," *Fusion Engineering and Design*, vol. 82, no. 5-14, pp. 1153–1160, Oct 2007.

[19] Z. Wang, H. Wang, E. Schuster, Z. Luo, Y. Huang, Q. Yuan, B. Xiao, and D. Humphreys, "Optimal shaping of the safety factor profile in the EAST tokamak," in *2021 IEEE Conference on Control Technology and Applications (CCTA)*. IEEE, 2021, pp. 63–68.

[20] S. J. Bradtke and A. G. Barto, "Linear least-squares algorithms for temporal difference learning," *Machine learning*, vol. 22, no. 1, pp. 33–57, 1996.

[21] J. A. Boyan, "Technical update: Least-squares temporal difference learning," *Machine learning*, vol. 49, pp. 233–246, 2002.