Constrained Optimization Formulation of Bellman Optimality Equation for Online Reinforcement Learning

Hyochan Lee[1 [0000−0002−7445−7412]] and Kyunghwan Choi[1* [0000−0002−4832−1597]]

[1]CCS Graduate School of Mobility, Korea Advanced Institute of Science and Technology, Daejeon, 34051, Republic of Korea
{hyochanlee,kh.choi}@kaist.ac.kr

Abstract. This paper proposes an online reinforcement learning algorithm that directly solves the Bellman optimality equation by casting it as a constrained optimization problem. Unlike policy or value iteration, which incrementally approximate the Bellman (optimality) equation, the method treats the value function and control policy as joint decision variables and solves them simultaneously. The formulation also permits systematic incorporation of additional constraints, such as input or safety limits. Direct solution of the Bellman optimality equation enables coordinated value–policy updates that stabilize online adaptation. Explicit constraint handling ensures per-step feasibility in online settings. The problem is addressed using a Lagrangian-based primal–dual approach, resulting in online update laws that drive the Bellman error toward zero while satisfying all constraints. The effectiveness of the method is demonstrated on a constrained nonlinear optimal control task.

Keywords: Online Reinforcement Learning, Optimal Control, Constrained Optimization, Neural Network, Nonlinear Systems.

1 Introduction

Optimal control of nonlinear systems remains a central challenge in modern control theory, with broad applications in robotics, mobility, aerospace, and process control [1–3]. In practice, progress is impeded by two well-known obstacles: obtaining an accurate nonlinear model and the general intractability of analytically solving the Hamilton–Jacobi–Bellman (HJB) equation or Bellman optimality equation for most nonlinear systems [4, 5].

Reinforcement Learning (RL) has emerged as a compelling alternative. From a control-theoretic perspective, RL is often framed as adaptive optimal control, and the control community has extensively studied Adaptive Dynamic Programming (ADP) as a principal vehicle for online RL in continuous control [6, 7]. ADP learns optimal policies through system interaction and function approximation, typically by minimizing the residual of the HJB/Bellman equation or by actor–critic schemes grounded in Bellman error minimization [8–11].

Despite its success, many ADP formulations used in control are implemented as on-policy, incremental parameter adaptation schemes. This design can limit data efficiency and the persistence of learned information across operating regimes, which may hinder generalization beyond the currently excited trajectories. In addition, while there has been progress on safe and constrained RL, constraint handling is not uniformly integrated in ADP-based controllers: policies are often derived from unconstrained Bellman updates, which may lead to actuator-limit or safety violations when deployed [8–11].

To address these issues, we propose an online RL framework that solves the Bellman optimality equation within a constrained optimization problem. In contrast to policy iteration (PI) and value iteration (VI)—which approach optimality via incremental Bellman backups (alternating policy evaluation/improvement or repeated application of the Bellman operator)—the proposed method treats the value function parameters and the control policy as joint decision variables and solves the Bellman optimality equation directly at each time step. This direct, joint solution removes the decoupled evaluation–improvement loop and, together with the constrained formulation, explicitly enforces input and safety limits throughout learning, which is crucial in online operation. A Lagrangian-based primal–dual scheme then yields online update laws for the actor, critic, and Lagrange multipliers, driving the Bellman error toward zero under the imposed constraints. The approach is validated on a constrained nonlinear optimal control task, demonstrating stable, constraint-satisfying learning in an online setting.

2 Preliminaries

2.1 System Dynamics

This paper considers a class of discrete-time, control-affine nonlinear systems, for which the dynamics are described by the state-space model:

$$x_{k+1} = f(x_k) + g(x_k)u_k \tag{1}$$

where $k \in \mathbb{Z}_{\geq 0}$ is the discrete time index, $x_k \in \mathbb{R}^n$ is the state vector, and $u_k \in \mathbb{R}^m$ is the control input. The function $f : \mathbb{R}^n \to \mathbb{R}^n$ represents the unknown drift dynamics of the system, and $g : \mathbb{R}^n \to \mathbb{R}^{n\times m}$ represents the control input effectiveness. The objective is to design an online learning controller for this class of systems, as detailed in the subsequent sections.

2.2 Problem Formulation

For the discrete-time system (1), the objective of optimal control problem is to find a state-feedback policy $u_k = \pi(x_k)$ that minimizes the infinite-horizon cost function, which is the sum of all future costs:

$$V(x_k) = \sum_{i=k}^{\infty} r(x_i, u_i) \tag{2}$$

where $r(x_i, u_i)$ is the reward function.

According to Bellman's principle of optimality, the solution to this problem is characterized by the optimal value function, $V^*(x)$, which represents the minimum possible cost-to-go from a given state x. This function must satisfy the following Bellman optimality equation:

$$V^*(x_k) = \min_{u_k} \{r(x_k, u_k) + V^*(x_{k+1})\} \tag{3}$$

The optimal control policy, $\pi^*(x_k)$, is the policy that achieves this minimum at every state. It is defined as:

$$\pi^*(x_k) = \arg\min_{u_k} \{r(x_k, u_k) + V^*(x_{k+1})\} \tag{4}$$

For a quadratic cost reward function $r(x_k, u_k) = x_k^T Q x_k + u_k^T R u_k$, the optimal control input can be derived analytically as a function of the value function's gradient:

$$u_k^* = -\frac{1}{2} R^{-1} g(x_k)^T \nabla V^*(x_{k+1}) \tag{5}$$

A critical limitation of this standard formulation is that the minimization in (5) is unconstrained, which precludes the direct handling of input constraints. As a result, the computed u_k^* may violate actuator bounds or safety requirements. In parallel, control-oriented ADP implementations of PI/VI typically rely on bootstrapped, sample-based Bellman updates (e.g., TD, Q-learning) which—under function approximation—may lose contraction, leading to oscillatory or even divergent value estimates and sensitivity to stepsizes and distribution shift.

To address feasibility and stability within a single framework, we replace incremental backups with a constrained optimization that is equivalent to the Bellman optimality equation and treats the value-function parameters and the policy as joint decision variables. Input and safety limits are encoded as explicit constraints in the same program, so feasibility is handled by the optimization rather than by ad hoc projections or penalties. We solve the problem online via a Lagrangian-based primal–dual scheme, whose updates seek a Karush-Kuhn-Tucker (KKT) point that simultaneously satisfies Bellman optimality and the imposed constraints, thereby providing a principled mechanism for per-step feasibility and driving the Bellman error toward zero under standard conditions.

3 Proposed method

3.1 Neural Network Approximation

The optimal value function $V^*(x)$ in the Bellman equation (3) is generally unknown and cannot be solved analytically for nonlinear systems. Therefore, a function approximator is required to represent $V^*(x)$ over a compact set $\Omega \subset \mathbb{R}^n$. This approach utilizes a neural network (NN) structure for this purpose.

According to the Weierstrass Higher-Order Approximation Theorem [12], there exists a dense basis set that allows the optimal value function to be represented by a NN as:

$$V^*(x_k) = \theta^{*T}\phi(x_k) + \varepsilon(x_k) \tag{6}$$

where $\phi(x_k) \in \mathbb{R}^p$ is a vector of p basis functions and $\varepsilon(x_k)$ is the function approximation error. The ideal weight vector $\theta^* \in \mathbb{R}^p$ is defined as the weight that minimizes the supremum of the approximation error over the compact set Ω:

$$\theta^* = \arg\min_{\theta \in \mathbb{R}^p} \left\{ \sup_{x_k \in \Omega} \|V^*(x_k) - \theta^T \phi(x_k)\| \right\} \tag{7}$$

The approximation theorem states that the error $\varepsilon(x_k)$ can be made arbitrarily small by selecting a sufficiently large number of basis functions p [13].

In the proposed learning framework, the objective is to find an online estimate of the ideal weights, denoted as θ_k. This leads to the definition of the approximated value function, $\hat{V}(x_k, \theta_k)$, which is implemented by the critic network:

$$\hat{V}(x_k, \theta_k) = \theta_k^T \phi(x_k) \tag{8}$$

This approximation is then utilized to formulate the constrained optimization problem, as detailed in the subsequent section.

3.2 Constrained Optimization Formulation

Building on the Bellman optimality equation (3) and the neural-network approximation (8), we cast the online update at time step k as a constrained optimization problem. The key idea is to eliminate the minimization operator in the equation by introducing the control input u_k as an additional decision variable and enforcing the equation as an equality constraint, while minimizing the right-hand side of the equation. This yields a unified program that jointly optimizes the value-function parameters and the policy under explicit input/safety constraints; the resulting formulation at time step k is given below:

$$\begin{aligned} \min_{u_k, \theta_k} \quad & r(x_k, u_k) + \hat{V}(x_{k+1}, \theta_k) \\ \text{s.t.} \quad & e(u_k, \theta_k) = 0 \\ & c_i(u_k, x_k) \leq 0, \quad i = 1, 2, \cdots \end{aligned} \tag{9}$$

where the equality constraint $e(u_k, \theta_k)$ is the Bellman error:

$$e(u_k, \theta_k) \triangleq r(x_k, u_k) + \hat{V}(x_{k+1}, \theta_k) - \hat{V}(x_k, \theta_k) \tag{10}$$

and inequality constraints $c_i(u_k, x_k)$ define the system constraints.

The Lagrangian function L for this problem is constructed as:

$$\begin{aligned} L(u_k, \theta_k, \lambda_{1k}, \dots) = r(x_k, u_k) + \hat{V}(x_{k+1}, \theta_k) \\ + \lambda_{1,k} e(u_k, \theta_k) \ + \sum_i \lambda_{i+1,k} c_i(u_k, x_k) \end{aligned} \tag{11}$$

where λ_{ik} are the Lagrange multipliers. The solution is found by satisfying the KKT conditions [14].

3.3 Online Update Laws

Online update laws are developed to iteratively find a solution satisfying the KKT conditions.

Critic Update. The critic weights are updated by gradient descent on the Lagrangian:

$$\theta_{k+1} = \theta_k - \alpha_\theta \nabla_{\theta_k} L \tag{12}$$

where $\nabla_{\theta_k} L = \phi(x_{k+1}) + \lambda_{1k}(\phi(x_{k+1}) - \phi(x_k))$.

Dual Updates. The Lagrange multipliers are updated by gradient ascent to enforce the constraints. The multiplier for the Bellman error is updated to drive $e \to 0$:

$$\lambda_{1,k+1} = \lambda_{1,k} + \alpha_{\lambda 1} e(u_k, \theta_k) \tag{13}$$

The multipliers for input constraints are updated using projected gradient ascent:

$$\lambda_{i+1,k+1} = \max(0, \lambda_{i+1,k} + \alpha_{\lambda_{i+1}} c_i(u_k, x_k)) \tag{14}$$

Actor Policy. The control policy is derived by solving the stationary condition $\nabla_{u_k} L = 0$:

$$u_k = -\frac{1}{2R}\left(\theta_k^T \nabla\phi(x_{k+1}) g(x_k) + \frac{\sum_i \lambda_{i+1,k} \nabla_u c_i(u_k, x_k)}{1 + \lambda_{1,k}}\right) \tag{15}$$

4 Simulation Validation

4.1 Simulation Setup

To validate the performance of the proposed method, a nonlinear benchmark system presented in [15] was investigated. This benchmark was chosen because [15] provides an analytical optimal solution, enabling direct and quantitative comparison against ground truth performance.

The system was implemented in its continuous-time form, while the controller operated in discrete time, making it a sampled-data system. The continuous-time dynamics are given by:

$$\dot{x}_1 = -x_1 + x_2 \tag{16}$$

$$\dot{x}_2 = -0.5(x_1 + x_2) + 0.5x_2 \sin^2(x_1) + \sin(x_1)u \tag{17}$$

The simulation was conducted in the MATLAB/Simulink 2024b environment. The proposed method was implemented as a discrete-time algorithm

with a sampling time of $T_s = 1$ ms. For the critic network, a polynomial basis function vector was chosen as $\phi(x_k) = [x_1^2, x_1x_2, x_2^2]^T$. The reward function is defined by $r(x_k, u_k) = x_k^T Q x_k + u_k^T R u_k$ with weighting matrices $Q = \mathbf{I}_{2\times 2}$ and $R = 1$. The learning rates for the online update laws were selected as $\alpha_\theta = 1 \times 10^{-5}$, $\alpha_{\lambda 1} = 1 \times 10^{-4}$, and $\alpha_{\lambda 2} = \alpha_{\lambda 3} = 0.6$. The system was initialized at $x_0 = [-5, 5]^T$, with initial critic weights $\theta_0 = [1.5, 1.5, 1.5]^T$.

4.2 Simulation Results

To demonstrate the efficacy of the proposed control algorithm, two distinct scenarios were simulated: an unconstrained case to establish a performance baseline, and a constrained case to validate the algorithm's performance under system constraint limitations.

Unconstrained Control First, the system was simulated without any constraints on the control input u to establish an ideal performance baseline. For a comparative analysis, the proposed control method was compared with two alternative approaches: an analytical solution presented in [15] and a controller based on conventional Temporal Difference (TD) learning [16]. TD learning was chosen as a baseline because it typifies the bootstrapped, sample-based Bellman updates used in control-oriented ADP/PI for online RL, providing a widely adopted yet sensitivity-prone comparator under function approximation. The TD learning algorithm was implemented using the same basis functions $\phi(x_k) = [x_1^2, x_1x_2, x_2^2]^T$ and initial critic weights $\theta_0 = [1.5, 1.5, 1.5]^T$ as the proposed method. The learning rate for the TD critic update was set to $\alpha_\theta = 1 \times 10^{-6}$.

The simulation results for the three methods are presented in Figs. 1-4. Figure 1 shows that all three methods successfully stabilize the system states. However, the performance differences become evident in the control input and value function trajectories, as shown in Figs. 2 and 3. The proposed method's control input and value function closely track of the optimal solution. In contrast, the TD learning approach exhibits significant deviation.

This difference in performance can be attributed to the learning dynamics of the critic network, as illustrated in Fig. 4. The critic weights of the proposed method are learned to approximate the behavior of the optimal value function, ensuring stable and near-optimal convergence. In contrast, the TD learning method fails to achieve this during the online learning phase, as its weights follow a different convergence pattern.

Constrained Control To validate the proposed method under a constrained scenario, state-dependent input constraints were added, defined as $c_1(u_k, x_k) = 0.1x_1^2 + u - 3 \leq 0$ and $c_2(u_k, x_k) = 0.3x_2 - u - 2 \leq 0$.

The simulation results are summarized in Figs. 5–7. Figure 5 presents the overall closed-loop performance. The state trajectories (Fig. 5(a)) show that the controller stabilizes the system, driving the states to the origin. The control input (Fig. 5(b)) adapts to satisfy the time-varying, state-dependent constraints

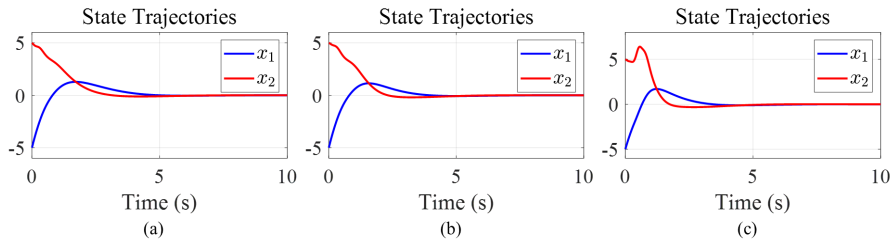

Fig. 1. Comparison of state trajectories in the unconstrained case: (a) analytical solution, (b) proposed method, and (c) TD learning-based solution.

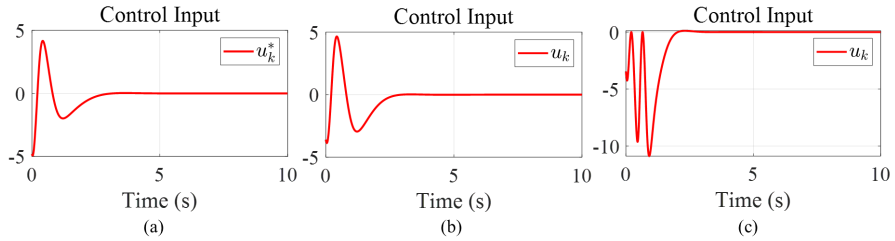

Fig. 2. Comparison of control input trajectories in the unconstrained case: (a) analytical solution, (b) proposed method, and (c) TD learning.

induced by x_1 and x_2. The approximated value function (Fig. 5(c)) converges smoothly, indicating stable learning.

The behavior of the Lagrange multipliers in Fig. 6 further supports the proposed constraint-handling mechanism. The multipliers λ_2 and λ_3 become active only when their respective constraints are binding, consistent with the KKT conditions. Meanwhile, the convergence of the multiplier associated with the Bellman error, λ_1, indicates that this constraint is enforced throughout the simulation.

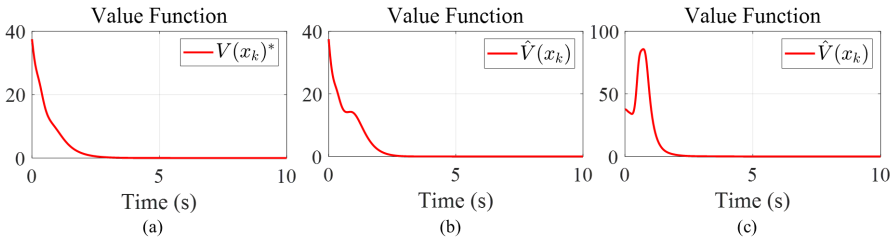

Fig. 3. Comparison of value function trajectories in the unconstrained case: (a) analytical solution, (b) proposed method, and (c) TD learning.

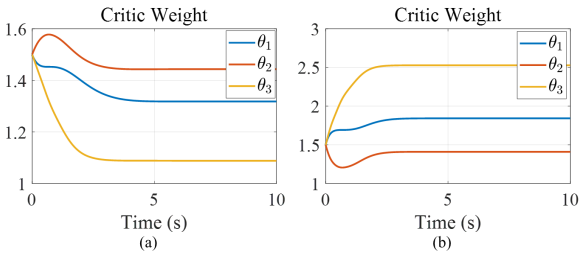

Fig. 4. Convergence of the critic network weights in the unconstrained case: (a) proposed method and (b) TD learning.

Finally, Fig. 7 illustrates the learning dynamics. The critic weights (Fig. 7(a)) converge to steady values, and the Bellman error e (Fig. 7(b)) is driven toward zero. Taken together, these results demonstrate that the proposed framework learns an optimal policy online while adhering to state–input constraints.

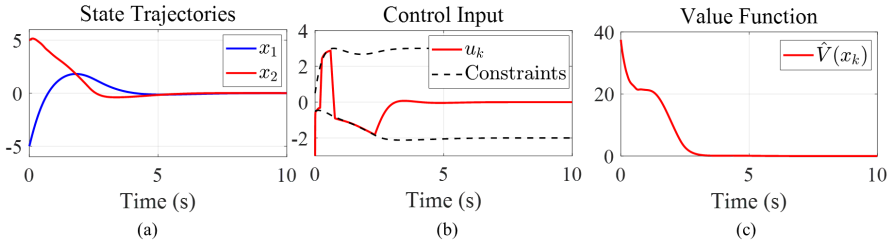

Fig. 5. Trajectories under the proposed method in the constrained case: (a) state, (b) input, and (c) value function.

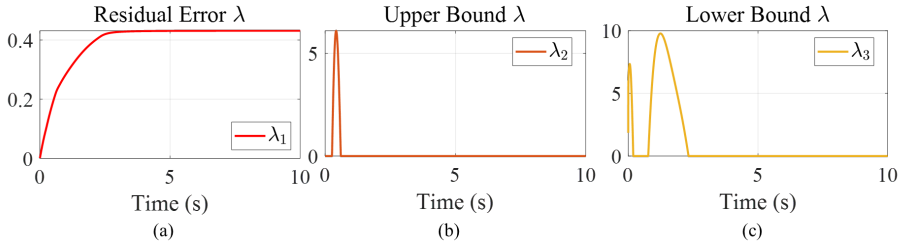

Fig. 6. Trajectories under the proposed method in the constrained case: (a) Bellman error multiplier (λ_1), (b) constraint c_1 multiplier (λ_2) , and (c) constraint c_2 multiplier (λ_3).

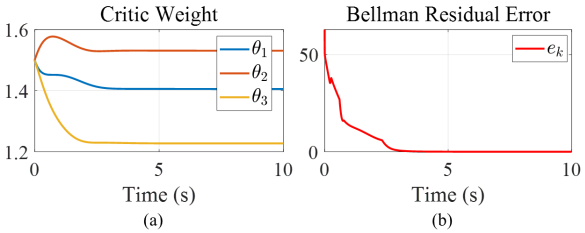

Fig. 7. Learning dynamics of the proposed method in the constrained case: (a) convergence of critic weights and (b) Bellman error.

5 Conclusion

This paper introduced a novel online RL framework that directly solves the Bellman optimality equation within a constrained optimization problem. By jointly optimizing the value function and policy while explicitly encoding input/safety constraints, the method replaces incremental PI/VI-style Bellman backups with a unified, constraint-aware online update suited to streaming control settings.

A Lagrangian primal–dual scheme provides online update laws for the actor, critic, and Lagrange multipliers, seeking KKT feasibility at each step. Simulations on a constrained nonlinear task show per-step constraint satisfaction, closed-loop stabilization, smooth critic convergence, and a decreasing Bellman error, yielding performance consistent with an analytical optimum.

Future work will: (i) extend the formulation to continuous-time tracking with rigorous stability analysis; (ii) study robustness on stochastic control problems; and (iii) improve data efficiency by integrating batch/replay–based online RL within the same constrained optimization paradigm while preserving feasibility guarantees.

Acknowledgment

This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) (RS-2025-00554087).

References

1. F. L. Lewis, D. L. Vrabie, and V. L. Syrmos, *Optimal Control*, 3rd ed. Wiley, 2012.
2. S. Bhasin, R. Kamalapurkar, M. Johnson, K. Vamvoudakis, F. Lewis, and W. Dixon, "A novel actor–critic–identifier architecture for approximate optimal control of uncertain nonlinear systems," *Automatica*, vol. 49, no. 1, pp. 82–92, Jan. 2013.

3. Y. Li, Y. Liu, and S. Tong, "Observer-Based Neuro-Adaptive Optimized Control of Strict-Feedback Nonlinear Systems With State Constraints," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 33, no. 7, pp. 3131–3145, Jul. 2022.
4. S. He, H. Fang, M. Zhang, F. Liu, and Z. Ding, "Adaptive Optimal Control for a Class of Nonlinear Systems: The Online Policy Iteration Approach," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 31, no. 2, pp. 549–558, Feb. 2020.
5. L. Zhu, Q. Wei, and P. Guo, "Online Off-Policy Reinforcement Learning for Optimal Control of Unknown Nonlinear Systems Using Neural Networks," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 54, no. 8, pp. 5112–5122, Aug. 2024.
6. D. Liu and Q. Wei, "Policy Iteration Adaptive Dynamic Programming Algorithm for Discrete-Time Nonlinear Systems," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 25, no. 3, pp. 621–634, Mar. 2014.
7. D. Liu, S. Xue, B. Zhao, B. Luo, and Q. Wei, "Adaptive Dynamic Programming for Control: A Survey and Recent Advances," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 51, no. 1, pp. 142–160, Jan. 2021.
8. D. Wang, N. Gao, D. Liu, J. Li, and F. L. Lewis, "Recent Progress in Reinforcement Learning and Adaptive Dynamic Programming for Advanced Control Applications," *IEEE/CAA Journal of Automatica Sinica*, vol. 11, no. 1, pp. 18–36, Jan. 2024.
9. R. Kamalapurkar, P. Walters, and W. E. Dixon, "Model-based reinforcement learning for approximate optimal regulation," *Automatica*, vol. 64, pp. 94–104, Feb. 2016.
10. P. Deptula, J. A. Rosenfeld, R. Kamalapurkar, and W. E. Dixon, "Approximate Dynamic Programming: Combining Regional and Local State Following Approximations," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 29, no. 6, pp. 2154–2166, Jun. 2018.
11. W. A. Makumi, O. S. Patil, and W. E. Dixon, "Lyapunov-based adaptive deep system identification for approximate dynamic programming," *Automatica*, vol. 180, p. 112462, Oct. 2025.
12. D. L. Vrabie, K. G. Vamvoudakis, and F. L. Lewis, *Optimal adaptive control and differential games by reinforcement learning principles*, ser. IET control engineering series. London: Institution of engineering and technology, 2013.
13. P. Tabuada and B. Gharesifard, "Universal approximation power of deep residual neural networks through the lens of control," *IEEE Transactions on Automatic Control*, vol. 68, no. 5, pp. 2715–2728, may 2023.
14. J. Nocedal and S. J. Wright, *Numerical optimization*, ser. Springer series in operations research. New York: Springer, 1999.
15. D. Vrabie and F. Lewis, "Neural network approach to continuous-time direct adaptive optimal control for partially unknown nonlinear systems," *Neural Networks*, vol. 22, no. 3, pp. 237–246, Apr. 2009.
16. R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*, 2nd ed. The MIT Press, 2018.