

Constrained Optimization-Based Neuro-Adaptive Control (CONAC) for Unknown Systems Under Multiple Convex Input Constraints

Abstract—This study presents a constrained optimization-based neuro-adaptive control (CONAC) for a class of unknown multi-input-multi-output (MIMO) systems subject to multiple convex input constraints. A deep neural network (DNN) is employed to approximate the ideal control law while handling multiple convex input constraints within a unified constrained optimization framework. The adaptive variables, including the DNN weights and Lagrange multipliers, are updated through adaptation law derived from the formulated constrained optimization problem, yielding Karush-Kuhn-Tucker (KKT) optimality conditions at equilibrium. The controller's stability is rigorously analyzed using Lyapunov theory, establishing the uniform ultimate boundedness (UUB) of the tracking errors and adaptive variables. The proposed controller is compared with existing methods through real-time experiments on a 2-degree-of-freedom (DOF) robotic manipulator, highlighting its superior capability to handle input constraints and its feasibility in real-time implementation.

Index Terms—Neuro-adaptive control, constrained optimization, deep neural network, multiple convex input constraints.

I. INTRODUCTION

A. Background

MANY control systems are constrained by various safety requirements, actuator limitations, and physical restrictions [1]. These constraints often manifest as convex input constraints, which can be in the form of nonlinear constraints rather than the element-wise (symmetric or asymmetric) constraints that limit each input independently, i.e., $\underline{u}_i \leq u_i \leq \bar{u}_i$, $\forall i \in \{1, \dots, n\}$, where $\underline{u}_i, \bar{u}_i \in \mathbb{R}$ denote the lower and upper bounds of the i th input u_i , respectively. For example, under road friction limits, the admissible global force and moment of vehicles can be characterized by a weighted norm bound, which yields a convex ellipsoidal constraint [2]. In motor drive systems, the feasible voltage region is often represented by circular [3] or hexagonal sets [4]. Furthermore, admissible input sets may be determined by multiple constraints that must hold simultaneously, such as element-wise constraints combined with norm constraints, which can be represented as the intersection of convex sets. Therefore, addressing these multiple convex input constraints is crucial for ensuring the safety and reliability of control systems in real-world applications.

In parallel, control systems often involve uncertain or even unknown dynamics due to modeling errors, unmodeled dynamics, and external disturbances. Although adaptive control provides a systematic mechanism for online adaptation of controllers [5], conventional formulations often rely on structural assumptions about the uncertain or unknown dynamics, which may not be available in practice. Neuro-adaptive control (NAC) alleviates this assumption by using neural networks (NNs) to approximate the dynamics in a structured and

parameterized representation [6], [7]. Like adaptive control, NAC methods ensure closed-loop stability using Lyapunov theory, and the NN weights are adapted online without prior knowledge of the system. The NAC methods have shown promising results in various real-world applications, including vehicle control [8] and flexible robotic manipulators [9].

Because multiple convex input constraints and uncertain or unknown dynamics often coexist in real-world applications, incorporating multiple input constraint handling into NAC is essential. This motivates a review of existing input constraint handling methods in NAC and related constrained control frameworks.

B. Literature Review

In Table I, we summarize input constraint handling methods reported in the literature, which can be broadly categorized into structural methods and optimization-based methods. In subsequent subsections, these methods are reviewed, highlighting their advantages and limitations. Additionally, the input constraints also referred to as input saturation to broadly refer to limitations imposed on the control input.

1) *Structural Methods for Input Constraint Handling in NAC*: Since the NAC framework is based on Lyapunov theory, input constraints have been primarily handled using structural methods in the NAC literature. In [1], Ji *et al.* categorized the literature on the structural methods into two strategies: anti-saturation design and direct design. We review these strategies in the context of NAC.

In anti-saturation design, a virtual auxiliary system is mainly constructed to compensate for the effects of input saturation [9]–[23]. Owing to their high compatibility, auxiliary systems have been adopted in various control designs. For example, auxiliary systems have been incorporated into backstepping-based NAC [15] and sliding mode control (SMC)-based NAC [10] to mitigate the effects of input saturation. Furthermore, the auxiliary systems have been integrated into reinforcement learning (RL)-based NAC designs [9] and barrier Lyapunov function (BLF)-based NAC designs [11]. To achieve finite-time or fixed-time convergence, novel finite-time [12] or fixed-time [13], [14] auxiliary systems have been proposed.

Generally, when the control input reaches the saturation limit, the auxiliary system is excited, and the auxiliary state is fed back into the control law as additional compensation terms. However, from a learning perspective, if the auxiliary state is not incorporated into the feedback error used in the adaptation law, the NN weights may generate excessive control input to achieve the desired tracking performance, since the saturation effect is not considered in the adaptation process. Therefore, the auxiliary state is often considered in the adaptation law

TABLE I: Comparison of input-constraint handling approaches.

	Structural Methods	Optimization-Based Methods	Proposed Approach
Representative literature	Auxiliary system design [9]–[23], Smooth approximation of saturation [24]–[26]	MPC [30]–[36], CBF-QP filter [37]–[39]	Preliminary works of this study [3], [40]
Stability analysis	Lyapunov-based stability guarantees	Conditions on feasibility and related assumptions required	Lyapunov-based stability guarantees
Input constraints handling capability	Primarily element-wise (symmetric or asymmetric) input constraints	General input constraints (convex/non-convex constraints)	Multiple convex input constraints
Implementation	Continuous adaptation (typically Lyapunov-based)	Optimization at each time step	Continuous adaptation (constrained optimization-based)
Computational burden	Typically low	Problem-dependent (may require iterative solvers)	Typically low

[13]–[19]. This ensures that the NN weights adapt not only for approximating the ideal control law but also for compensating for the effect of input saturation.

On the other hand, in direct design, the saturation effects are directly incorporated into the controller design and stability analysis. Typically, the saturation function is approximated by a smooth function, such as $\tanh(\cdot)$, and the mean value theorem is used to transform it into affine form [24]–[26]. However, as reported in [27], the smooth approximation of the saturation function may introduce conservative behavior and performance degradation, especially when the control input is close to the saturation limit, compared to the auxiliary system-based anti-saturation design.

Another approach can be found in reinforcement learning (RL)-based NAC [28], [29]. The input saturation is handled by introducing a nonquadratic input cost into the infinite-horizon optimal control problem, which is solved using adaptive dynamic programming (ADP). Within the resulting ADP framework, the optimal cost function is approximated by a critic neural network (CNN), and the optimal control law is derived from the Hamilton-Jacobi-Isaacs (HJI) equation or the Hamilton-Jacobi-Bellman (HJB) equation. Owing to the nonquadratic input cost, the resulting optimal control law is based on $\tanh(\cdot)$ and satisfies asymmetric or symmetric input bounds.

Despite their effectiveness, the aforementioned structural methods are limited to element-wise input constraints, such as symmetric or asymmetric bounds on each input. Thus, extending them to admissible input sets described by multiple convex constraints is not straightforward.

2) *Optimization-Based Methods for Input Constraint Handling*: Besides structural methods, optimization-based methods have been studied for constraint handling across a broad range of control design frameworks. These methods compute the control input by solving constrained optimization problems at each time step, as in model predictive control (MPC) [30]–[36] and control barrier function (CBF)-based quadratic programming (QP) filters [37]–[39]. In these methods, the constrained control problem is explicitly formulated as a constrained optimization problem. Therefore, complex nonlinear (even non-convex) input constraints can be systematically incorporated into the control design, which is a significant advantage over the structural methods.

Even though they solve constrained optimization problems

at each time step, the computational burden may not necessarily be significant for small-scale QPs or MPC problems with short horizons and simple constraints. However, it can increase substantially when multiple constraints must be handled simultaneously, nonlinear constraints are introduced, or the resulting problem is non-convex. In this regard, some studies [34]–[36] trained the optimal constrained control policy offline using deep NNs (DNNs) to reduce online computation.

Moreover, closed-loop stability guarantees generally rely on additional analytical conditions specific to the selected optimization framework. For MPC, these conditions commonly include recursive feasibility, appropriate terminal ingredients, sufficiently long prediction horizons, and computational tractability [30]. In addition to these conditions, model accuracy is critical, thereby motivating robust [31], data-driven [32], and learning-based MPC [33], [34] approaches. Model accuracy is also crucial for CBF-QP filters, where the validity of the safety guarantee depends on the accuracy of the system dynamics used to formulate the barrier constraint. Accordingly, neuro-adaptive system identifiers have been introduced to construct the CBF-QP filters using approximated system dynamics [39].

Although these approaches can provide stability or safety guarantees under their stated assumptions, the guarantees typically depend on feasibility, prediction, or safety-critical filters whose validity is affected by model accuracy, uncertainty descriptions, data quality, and offline training coverage.

C. Motivation and Proposed Approach

Both structural and optimization-based methods are effective for input constraint handling, but they have complementary limitations, as summarized in Table I. Structural methods offer Lyapunov-based stability guarantees and solver-free computation, but they are limited in terms of the classes of constraints they can accommodate. Optimization-based methods can handle general constraints, but their closed-loop stability guarantees are often tied to additional conditions and iterative solvers may be required.

These observations motivate a unified NAC framework in which more complex input constraints are incorporated directly into the online adaptation law, as presented in the previous work [3]. In the proposed CONAC framework, the NAC problem is formulated as a constrained optimization

problem in which multiple convex input constraints are imposed as inequality constraints. The constrained optimization formulation is not used to solve a control allocation problem at each time step; instead, it is used to derive continuous-time adaptation law for the adaptive variables. Therefore, the proposed framework combines the Lyapunov-based stability analysis of structural methods with the broader input-constraint handling capability of optimization-based methods, particularly for multiple convex input constraints.

D. Contributions

The main contributions of this study are:

- Any combination of convex input constraints can be handled, thereby significantly extending the applicability of the NAC to real-world systems with complex input constraints;
- The adaptation law for adaptive variables, including the DNN weights and the Lagrange multipliers, is systematically derived from the formulated constrained optimization problem, and the Karush-Kuhn-Tucker (KKT) optimality conditions are satisfied at an equilibrium of the adaptation dynamics [41, Chap. 12, Thm. 12.1]; and
- The closed-loop stability under the proposed Constrained Optimization-Based Neuro-Adaptive Control (CONAC) is rigorously analyzed using Lyapunov theory, ensuring the uniform ultimate boundedness (UUB) of the tracking error and adaptive variables.

This study extends the preliminary work in [3], [40], which introduced inequality constraints on the weight and input norms within a constrained optimization framework for NAC using single-hidden-layer NNs. In the present work, the same framework is generalized to accommodate multiple convex input constraints and extended to DNNs, while still ensuring Lyapunov-based stability guarantees. Furthermore, unlike the previous study, which was limited to numerical simulations, the proposed CONAC is experimentally validated through real-time implementation on a 2-degree-of-freedom (DOF) robotic manipulator.

Notation: $\otimes$ denotes the Kronecker product [42, Chap. 7, Def. 7.1.2]. A vector and a matrix are denoted by $\mathbf{x} = [x_i]_{i \in \{1, \dots, n\}} \in \mathbb{R}^n$ and $\mathbf{A} := [a_{ij}]_{i \in \{1, \dots, n\}, j \in \{1, \dots, m\}} \in \mathbb{R}^{n \times m}$, respectively. The i th column of the matrix $\mathbf{A} \in \mathbb{R}^{n \times m}$ is denoted by $\text{Col}_i(\mathbf{A})$. For the matrix $\mathbf{A} \in \mathbb{R}^{n \times m}$, $\text{vec}(\mathbf{A}) := (\text{Col}_1(\mathbf{A})^\top, \dots, \text{Col}_m(\mathbf{A})^\top)^\top \in \mathbb{R}^{nm}$ denotes the vectorization of $\mathbf{A}$. Finally, $\mathbf{I}_n$ denotes the $n \times n$ identity matrix, and $\mathbf{0}_{n \times m}$ denotes the $n \times m$ zero matrix. The operator $\text{Proj}_{h(\mathbf{z}) \leq 0}(\boldsymbol{\nu})$ denotes the projection of the direction $\boldsymbol{\nu} \in \mathbb{R}^n$ according to a convex set $\mathcal{C}(\mathbf{z}) := \{\mathbf{z} \in \mathbb{R}^n : h(\mathbf{z}) \leq 0\}$, where $h : \mathbb{R}^n \rightarrow \mathbb{R}$ is a convex function, following the projection rule in [43, Appendix E, Eq. (E.4)].

E. Organization

The remainder of this paper is organized as follows. Section II presents the target system and control objective. Section III introduces the proposed CONAC, including the adaptation law for adaptive variables. Section IV analyzes the closed-loop stability. Section V presents a real-time implementation

of CONAC on a 2-DOF robotic manipulator, compares it with existing methods, and discusses its advantages and limitations. Finally, Section VI concludes the paper and discusses potential future works.

II. PROBLEM FORMULATION

We consider an unknown multi-input multi-output (MIMO) system modeled as

$$\frac{d}{dt}\mathbf{x} = \mathbf{f}(\mathbf{x}) + \mathbf{B}(\mathbf{x})\text{sat}(\mathbf{u}), \quad (1)$$

where $\mathbf{x}, \mathbf{u} \in \mathbb{R}^n$ denote the state and control input vectors, respectively. The unknown system function and control effectiveness matrix are denoted by $\mathbf{f} : \mathbb{R}^n \rightarrow \mathbb{R}^n$ and $\mathbf{B} : \mathbb{R}^n \rightarrow \mathbb{R}^{n \times n}$, respectively, which are assumed to be locally Lipschitz continuous. The saturation function is denoted by $\text{sat} : \mathbb{R}^n \rightarrow \Omega_u$, where $\Omega_u \subset \mathbb{R}^n$ is the convex admissible input set.

The system (1) is subject to the following assumptions.

Assumption 1. *The control effectiveness matrix $\mathbf{B}(\mathbf{x})$ is nonsingular and bounded as $\underline{b}\mathbf{I}_n \preceq \mathbf{B}(\mathbf{x}) \preceq \bar{b}\mathbf{I}_n$, where $\underline{b}, \bar{b} \in \mathbb{R}_{>0}$ are unknown positive constants.*

Assumption 2 ([44, Assump. 1(b)]). *There exists an unknown positive function $\rho_\beta(\mathbf{x}) : \mathbb{R}^n \rightarrow \mathbb{R}_{>0}$ such that $\frac{1}{2}\mathbf{v}^\top \left(\frac{d}{dt}\mathbf{B}(\mathbf{x})^{-1}\right)\mathbf{v} \leq \rho_\beta(\mathbf{x})\|\mathbf{v}\|^2$, $\forall \mathbf{v} \in \mathbb{R}^n$, and $\forall \mathbf{x} \in \mathbb{R}^n$.*

Assumption 3. *The convex admissible input set Ω_u and saturation function satisfy the following conditions:*

- 1) *The admissible input set Ω_u is defined by smooth convex constraint functions $c_j^u : \mathbb{R}^n \rightarrow \mathbb{R}$, $\forall j \in \mathcal{I}$, as*

$$\Omega_u = \bigcap_{j \in \mathcal{I}} \{\mathbf{u} \in \mathbb{R}^n : c_j^u(\mathbf{u}) \leq 0\},$$

where $\mathcal{I}$ denotes the input constraint index set.

- 2) *The origin lies in the interior of Ω_u , i.e.,*

$$c_j^u(\mathbf{0}) \leq -\underline{c}_j^u < 0, \quad \forall j \in \mathcal{I},$$

where $\underline{c}_j^u \in \mathbb{R}_{>0}$, $\forall j \in \mathcal{I}$, is known.

- 3) *There exists a known constant $\underline{u} \in \mathbb{R}_{>0}$ such that*

$$\inf_{\mathbf{u} \notin \Omega_u} (\|\text{sat}(\mathbf{u})\|) = \underline{u}.$$

By Assumption 3, the saturation function $\text{sat}(\mathbf{u})$ can represent any combination of multiple convex input constraints, rather than simple element-wise constraints which are conventionally considered in the NAC literature, as illustrated in Fig. 1.

Additionally, the desired trajectory $\mathbf{x}_d(t) : \mathbb{R}_{\geq 0} \rightarrow \mathbb{R}^n$ is assumed to be given and satisfies the following assumption.

Assumption 4. *The desired trajectory $\mathbf{x}_d := \mathbf{x}_d(t)$ is sufficiently smooth, and $\mathbf{x}_d$ and $\frac{d}{dt}\mathbf{x}_d$ are bounded.*

The control objective is to design a NAC for the unknown system (1) such that $\mathbf{x}$ tracks the desired trajectory $\mathbf{x}_d$, the multiple convex input constraints defined by $\text{sat}(\cdot)$ are satisfied, and closed-loop stability is guaranteed.

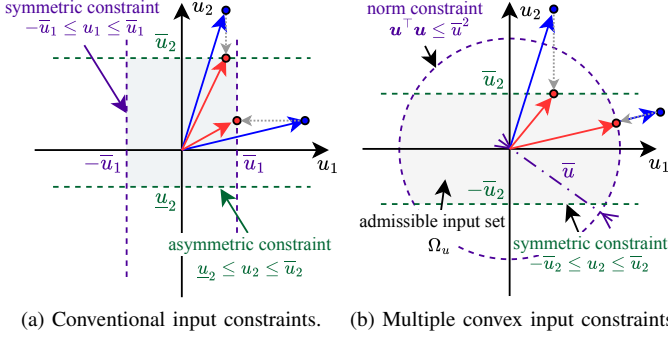


Fig. 1: Comparison of conventionally considered input constraints in the NAC literature and the proposed multiple convex input constraints.

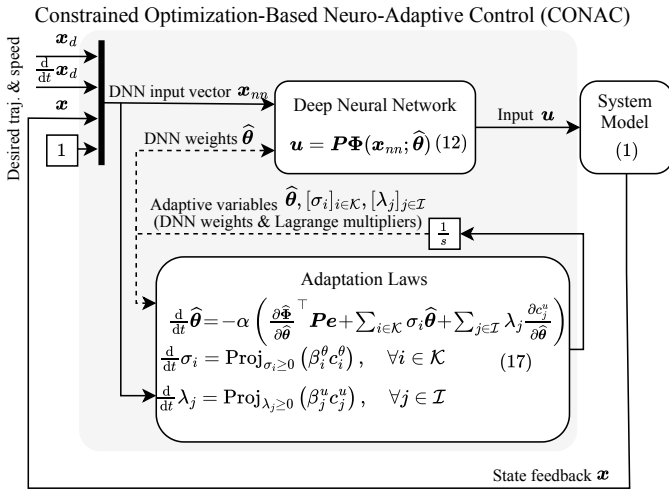


Fig. 2: Overall architecture of the proposed CONAC.

III. PROPOSED CONAC FRAMEWORK

The architecture of the proposed CONAC consists of a DNN that functions as a NAC and an adaptation process for the adaptive variables such as the DNN weights and Lagrange multipliers, as illustrated in Fig. 2. Before presenting CONAC, we first review some preliminaries in Section III-A. Then, the controller is presented in Section III-B along with the adaptation law in Section III-C.

A. Preliminaries

The following proposition will be used in the subsequent sections:

Proposition 1 ([42, Prop. 7.1.9]). *For a matrix $A \in \mathbb{R}^{n \times m}$ and a vector $b \in \mathbb{R}^n$, the following property holds:*

$$A^T b = \text{vec}(A^T b) = \text{vec}(b^T A) = (I_m \otimes b^T) \text{vec}(A). \quad (2)$$

We review the DNN structure and its approximation capability, which are essential for the subsequent controller design and stability analysis. According to the universal approximation property [7], the DNN $\Phi : \mathbb{R}^{l_0+1} \times \mathbb{R}^\Xi \rightarrow \mathbb{R}^{l_{k+1}}$ can be

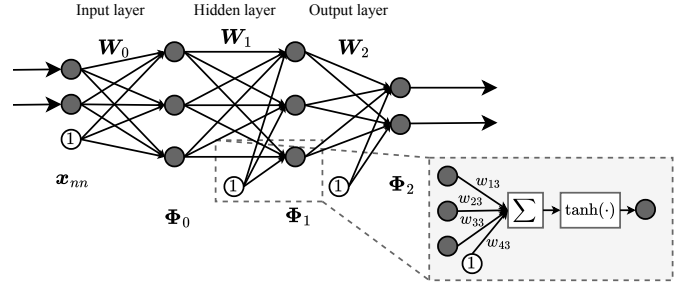


Fig. 3: Example of the DNN $\Phi(x_{nn}; \theta)$ with $k = 2, l_0 = 2, l_1 = l_2 = 3$, and $l_3 = 2$.

used to approximate any smooth unknown nonlinear function $g : \mathbb{R}^{l_0+1} \rightarrow \mathbb{R}^{l_{k+1}}$. Specifically, there exists a DNN such that

$$\sup_{x_{nn} \in \Omega_{x_{nn}}} \|g(x_{nn}) - \Phi(x_{nn}; \theta^*)\| = \sup_{x_{nn} \in \Omega_{x_{nn}}} \|\epsilon\| = \bar{\epsilon}, \quad (3)$$

where $x_{nn} \in \mathbb{R}^{l_0+1}$ is the DNN input vector and $\theta^* \in \mathbb{R}^\Xi$ denotes the ideal DNN weights that minimize the approximation error $\epsilon \in \mathbb{R}^{l_{k+1}}$. The DNN input vector x_{nn} should be designed to include all arguments of the unknown function g . A compact set $\Omega_{x_{nn}} \subset \mathbb{R}^{l_0+1}$ denotes the approximation region of the DNN, which ensures the boundedness of θ^* and ϵ , such that $\|\theta^*\| \leq \bar{\theta}^* \in \mathbb{R}_{>0}$ and $\|\epsilon\| \leq \bar{\epsilon} \in \mathbb{R}_{>0}$. The DNN $\Phi(x_{nn}; \theta) = \Phi_k$ can be recursively defined as

$$\Phi_i := \begin{cases} W_i^T \phi_i(\Phi_{i-1}), & i \in \{1, \dots, k\}, \\ W_0^T x_{nn}, & i = 0, \end{cases} \quad (4)$$

where $W_i \in \mathbb{R}^{(l_i+1) \times l_{i+1}}$ and $\phi_i : \mathbb{R}^{l_i} \rightarrow \mathbb{R}^{l_{i+1}}$ represent the weight matrix and the activation function of the i th layer for all $i \in \mathcal{K}$, respectively, and $\mathcal{K} := \{0, 1, \dots, k\}$ denotes the index set of layers. The DNN $\Phi(x_{nn}; \theta)$ has $k+1$ layers, where the input and output layers are indexed by $i = 0$ and $i = k$, respectively, as illustrated by the example in Fig. 3. The activation function is defined as $\phi_i(z) = (\tanh(z_1), \dots, \tanh(z_{l_i}), 1)^T$, for some $z \in \mathbb{R}^{l_i}$. We simplify the notations by vectorizing all weight matrices as

$$\theta_i := \text{vec}(W_i) \in \mathbb{R}^{(l_i+1)l_{i+1}}, \quad \forall i \in \mathcal{K},$$

and, finally, stacking them as $\theta := [\theta_i]_{i \in \mathcal{K}} \in \mathbb{R}^\Xi$, where $\Xi := \sum_{i \in \mathcal{K}} (l_i+1)l_{i+1}$. Additionally, like $\bar{\theta}$, there exists a maximum norm bound for the weights of each layer $\bar{\theta}_i^* \in \mathbb{R}_{>0}$, such that $\|\theta_i^*\| \leq \bar{\theta}_i^*, \forall i \in \mathcal{K}$.

The gradient of $\Phi(x_{nn}; \theta)$ with respect to θ can be obtained using Proposition 1 and the chain rule as follows:

$$\frac{\partial \Phi}{\partial \theta} = \begin{bmatrix} \frac{\partial \Phi}{\partial \theta_k} & \frac{\partial \Phi}{\partial \theta_{k-1}} & \dots & \frac{\partial \Phi}{\partial \theta_0} \end{bmatrix} \in \mathbb{R}^{l_{k+1} \times \Xi}, \quad (5)$$

where

$$\frac{\partial \Phi}{\partial \theta_i} = \begin{cases} I_{l_{k+1}} \otimes \phi_k^T, & i = k, \\ W_k^T \phi'_k (I_{l_k} \otimes \phi_{k-1}^T), & i = k-1, \\ \vdots & \vdots \\ W_k^T \phi'_k \dots W_1^T \phi'_1 (I_{l_1} \otimes x_{nn}^T), & i = 0, \end{cases} \quad (6)$$

and $\phi_i := \phi_i(\Phi_{i-1})$ and $\phi'_i := \frac{d\phi_i}{d\Phi_{i-1}}$, for all $i \in \mathcal{K} \setminus \{0\}$.

B. Controller Design

We define the tracking error as $e = x - x_d$, and derive the error dynamics from (1) as

$$\frac{d}{dt}e = f(x) + B(x) \text{sat}(u) - \frac{d}{dt}x_d. \quad (7)$$

Let us consider the Lyapunov function $V_c := \frac{1}{2}e^\top B^{-1}(x)e$. Invoking Assumptions 2, the time derivative of V_c can be bounded as follows:

$$\begin{aligned} \frac{d}{dt}V_c &= \frac{1}{2}e^\top \left(\frac{d}{dt}B^{-1} \right) e + e^\top B^{-1} \left(f + B \text{sat}(u) - \frac{d}{dt}x_d \right) \\ &\leq \rho_\beta(x) \|e\|^2 + e^\top u + e^\top B^{-1} \left(f - \frac{d}{dt}x_d \right) + e^\top \Delta u, \end{aligned} \quad (8)$$

where the arguments of $f(x)$ and $B(x)$ are omitted for brevity and $\Delta u := \text{sat}(u) - u$.

A desired control law u_d such that the tracking error e is stable in the absence of input saturation is selected as

$$u_d := -Pe - \rho_\beta(x)e + B^{-1} \left(-f + \frac{d}{dt}x_d \right), \quad (9)$$

where $pI_n \preceq P = P^\top \preceq \bar{p}I_n$ for known positive constants $p, \bar{p} \in \mathbb{R}_{>0}$.

By substituting u_d in (9) into (8), we have

$$\begin{aligned} \frac{d}{dt}V_c &\leq -e^\top Pe + e^\top (u - u_d) + e^\top \Delta u \\ &\leq -p\|e\|^2 + e^\top (u - u_d) + e^\top \Delta u. \end{aligned} \quad (10)$$

Notably, in the ideal case, i.e., $u = u_d$ and $\Delta u = 0$, the tracking error e asymptotically converges to zero with decay rate $\underline{p}$, since $\frac{1}{b} \frac{d}{dt}(\|e\|^2) \leq \frac{d}{dt}V_c \leq -\underline{p}\|e\|^2$.

Since the ideal control law (9) contains unknown functions $\rho_\beta(x)$, f and B , we employ the DNN $\Phi(x_{nn}; \theta)$ in (4) to approximate the ideal control law as follows:

$$\Phi(x_{nn}; \theta^*) + \epsilon = -e - P^{-1}(\rho_\beta(x)e + B^{-1}(f - \frac{d}{dt}x_d)).$$

The DNN input vector x_{nn} can be defined in various ways, as long as it includes the arguments of the unknown functions. In this formulation, we define the DNN input vector as $x_{nn} = (x_d^\top, \frac{d}{dt}x_d^\top, x^\top, 1)^\top \in \mathbb{R}^{3n+1}$. Then, we have

$$u_d = P(\Phi(x_{nn}; \theta^*) + \epsilon). \quad (11)$$

The actual control law is designed as

$$u := P\Phi(x_{nn}; \hat{\theta}), \quad (12)$$

where $\hat{\theta} \in \mathbb{R}^\Xi$ denotes the estimated DNN weights.

By substituting (11) and (12) into (10), we have

$$\frac{d}{dt}V_c \leq -\underline{p}\|e\|^2 + e^\top P(\hat{\Phi} - \Phi^* - \epsilon) + e^\top \Delta u, \quad (13)$$

where $\hat{\Phi} := \Phi(x_{nn}; \hat{\theta})$ and $\Phi^* := \Phi(x_{nn}; \theta^*)$ are abbreviated.

C. Adaptation Law Derivation

The control objective defined in Section II can be formulated as a constrained optimization problem, as follows:

$$\min_{\hat{\theta}} J(e; \hat{\theta}) = \frac{1}{2}e^\top e \quad (14a)$$

$$\text{subject to } c_i^\theta(\hat{\theta}_i) = \frac{1}{2} \left(\|\hat{\theta}_i\|^2 - \bar{\theta}_i^2 \right) \leq 0, \quad \forall i \in \mathcal{K}, \quad (14b)$$

$$c_j^u(\hat{\theta}) \leq 0, \quad \forall j \in \mathcal{I}, \quad (14c)$$

where $J(e; \hat{\theta})$ is the objective function. In the optimization problem, inequality constraints of weight boundedness $c_i^\theta(\hat{\theta}_i)$, $\forall i \in \mathcal{K}$, and input constraints $c_j^u(\hat{\theta})$, $\forall j \in \mathcal{I}$, are imposed.

Specifically, the input constraints used in (14c) are the same convex constraint functions that characterize the admissible input set in Assumption 3. With a slight abuse of notation, the composed input constraints are denoted by $c_j^u(\hat{\theta}) := c_j^u(u(\hat{\theta})) = c_j^u(P\Phi(x_{nn}; \hat{\theta}))$, $\forall j \in \mathcal{I}$, to explicitly represent the dependence of the input constraints on the DNN weights $\hat{\theta}$. Additionally, all active constraints are assumed to satisfy the Linear Independence Constraint Qualification (LICQ) to avoid degeneracy [41, pp. 465–467].

The Lagrangian function $L := L(e, \hat{\theta}, [\sigma_i]_{i \in \mathcal{K}}, [\lambda_j]_{j \in \mathcal{I}})$ is defined as

$$L = J(e; \hat{\theta}) + \sum_{i \in \mathcal{K}} \sigma_i c_i^\theta(\hat{\theta}_i) + \sum_{j \in \mathcal{I}} \lambda_j c_j^u(\hat{\theta}), \quad (15)$$

where $[\sigma_i]_{i \in \mathcal{K}}$ and $[\lambda_j]_{j \in \mathcal{I}}$ denote the Lagrange multipliers for the weight constraints (14b) and input constraints (14c), respectively. Then, the primal-dual problem can be formulated as follows:

$$\min_{\hat{\theta}} \max_{[\sigma_i]_{i \in \mathcal{K}}, [\lambda_j]_{j \in \mathcal{I}}} L(e, \hat{\theta}, [\sigma_i]_{i \in \mathcal{K}}, [\lambda_j]_{j \in \mathcal{I}}). \quad (16)$$

The adaptation law for the adaptive variables, $\hat{\theta}$, $[\sigma_i]_{i \in \mathcal{K}}$, and $[\lambda_j]_{j \in \mathcal{I}}$, is derived to solve (16) by using a primal-dual gradient update law, i.e., $\frac{d}{dt}\hat{\theta} = -\alpha \frac{\partial L}{\partial \hat{\theta}}$, $\frac{d}{dt}\sigma_i = \text{Proj}_{\sigma_i \geq 0} \left(\beta_i^\theta \frac{\partial L}{\partial \sigma_i} \right)$, and $\frac{d}{dt}\lambda_j = \text{Proj}_{\lambda_j \geq 0} \left(\beta_j^u \frac{\partial L}{\partial \lambda_j} \right)$, as follows:

$$\frac{d}{dt}\hat{\theta} = -\alpha \left(\frac{\partial J}{\partial \hat{\theta}} + \sum_{i \in \mathcal{K}} \sigma_i \hat{\theta}_i + \sum_{j \in \mathcal{I}} \lambda_j \frac{\partial c_j^u}{\partial \hat{\theta}} \right), \quad (17a)$$

$$\frac{d}{dt}\sigma_i = \text{Proj}_{\sigma_i \geq 0} \left(\beta_i^\theta c_i^\theta \right), \quad \forall i \in \mathcal{K}, \quad (17b)$$

$$\frac{d}{dt}\lambda_j = \text{Proj}_{\lambda_j \geq 0} \left(\beta_j^u c_j^u \right), \quad \forall j \in \mathcal{I}, \quad (17c)$$

where $\alpha \in \mathbb{R}_{>0}$ denotes the adaptation gain (learning rate) and $\beta_i^\theta, \beta_j^u \in \mathbb{R}_{>0}$, $\forall i \in \mathcal{K}, j \in \mathcal{I}$, denote the update rates of the Lagrange multipliers. The Lagrange multipliers associated with inequality constraints are kept non-negative, i.e., $\sigma_i, \lambda_j \geq 0$, due to $\text{Proj}_{(\cdot) \geq 0}(\cdot)$ in (17b) and (17c).

When a constraint becomes violated, i.e., positive constraint violation $c_i^\theta, c_j^u > 0$, the corresponding Lagrange multiplier increases according to (17b) and (17c), to address the violation by updating weight $\hat{\theta}$ through (17a). Ultimately, the Lagrange multiplier exhibits two distinct behaviors depending on the constraint status: it either decreases to zero (e.g., via a projection operator) when the constraint becomes inactive, $c_i^\theta, c_j^u < 0$, or converges to a positive value when the constraint remains binding, $c_i^\theta, c_j^u = 0$, to enforce the limit.

Remark 1 (Comparison with Existing Methods for Weight Boundedness). Unlike projection-based methods that directly restrict the adaptive weights to a prescribed set, or σ -modification and ϵ -modification that continuously bias them toward the origin [40], CONAC handles weight boundedness through inequality constraints within the primal-dual adaptation framework like the input constraints. The corresponding

Algorithm 1 Proposed CONAC

```

1: Initialize:  $\hat{\theta}^{(0)}, \sigma_i^{(0)}, \forall i \in \mathcal{K}, \lambda_j^{(0)}, \forall j \in \mathcal{I}$ 
2: for  $r = 0$  to  $\infty$  do
3:   Obtain  $\mathbf{x}^{(r)}$  and  $\mathbf{x}_d^{(r)}$ 
4:    $\mathbf{e}^{(r)} \leftarrow \mathbf{x}^{(r)} - \mathbf{x}_d^{(r)}$ 
5:   // Neuro-adaptive control law (Section III-B)
6:   Define  $\mathbf{x}_{nn}^{(r)}$  from all arguments of  $\mathbf{u}_d^{(r)}$ 
7:    $\mathbf{u}^{(r)} \leftarrow \mathbf{P}\Phi(\mathbf{x}_{nn}^{(r)}; \hat{\theta}^{(r)})$  based on (12)
8:   Apply  $\mathbf{u}^{(r)}$  to the system in (1)
9:   // Update Adaptive Variables (Section III-C)
10:  Compute  $\frac{\partial \hat{\Phi}^{(r)}}{\partial \hat{\theta}^{(r)}}$  as defined in (5)
11:  Compute  $c_i^{\theta(r)}, c_j^u(r)$  and  $\frac{\partial c_j^u(r)}{\partial \hat{\theta}^{(r)}}$  for all  $i \in \mathcal{K}, j \in \mathcal{I}$ 
12:  Update  $\hat{\theta}^{(r+1)}, \sigma_i^{(r+1)}, \forall i \in \mathcal{K}$ , and  $\lambda_j^{(r+1)}, \forall j \in \mathcal{I}$ ,
    using (18), (17b), and (17c), respectively.
13: end for

```

constraint term is only activated when the prescribed bound is violated, thereby softly driving the weights toward the admissible set without direct clipping or persistent origin-directed bias.

Meanwhile, the adaptation law of $\hat{\theta}$ defined in (17a) involves the partial derivative of the error with respect to the control input $\frac{\partial \mathbf{e}}{\partial \hat{\theta}}$, i.e., $\frac{\partial J}{\partial \hat{\theta}} = ((\frac{\partial \mathbf{e}}{\partial \hat{\theta}})(\frac{\partial \mathbf{u}}{\partial \hat{\theta}}))^\top \mathbf{e} = ((\frac{\partial \mathbf{e}}{\partial \hat{\theta}})\mathbf{P}(\frac{\partial \hat{\Phi}}{\partial \hat{\theta}}))^\top \mathbf{e}$. Since the objective function depends on the tracking error $\mathbf{e}$ of a dynamic system, obtaining $\frac{\partial \mathbf{e}}{\partial \hat{\theta}}$ is not straightforward. Exact computation methods, such as the forward sensitivity method [40], [45], can be used in principle, but they require exact knowledge of the system dynamics which is not available in this study. On the other hand, several approximation methods have been proposed [46], [47]. They approximate $\frac{\partial \mathbf{e}}{\partial \hat{\theta}}$ by taking the sign of each entry of the control effectiveness matrix, i.e., $\frac{\partial \mathbf{e}}{\partial \hat{\theta}} \approx [\text{sign}(b_{i,j})]_{i,j \in \{1, \dots, n\}}$, which corresponds to the MIT rule in the context of adaptive control. However, since the entry of the control effectiveness matrix $\mathbf{B}(\mathbf{x})$ is unknown in this study, we instead approximate $\mathbf{B}(\mathbf{x})$ by its sign, i.e., $\frac{\partial \mathbf{e}}{\partial \hat{\theta}} \approx \text{sign}(\mathbf{B}(\mathbf{x})) = \mathbf{I}_n$, under Assumption 1. Consequently, the adaptation law in (17a) becomes:

$$\frac{d}{dt} \hat{\theta} \approx -\alpha \left(\frac{\partial \hat{\Phi}}{\partial \hat{\theta}}^\top \mathbf{P} \mathbf{e} + \sum_{i \in \mathcal{K}} \sigma_i \hat{\theta}_i + \sum_{j \in \mathcal{I}} \lambda_j \frac{\partial c_j^u}{\partial \hat{\theta}} \right). \quad (18)$$

Additionally, the adaptation law (17) provides an optimal-ity interpretation at equilibrium. At an equilibrium of (17), where $\frac{d}{dt} \hat{\theta} = \mathbf{0}$, $\frac{d}{dt} \sigma_i = 0$, and $\frac{d}{dt} \lambda_j = 0$, the resulting equilibrium conditions correspond to KKT conditions, i.e., $\frac{\partial L}{\partial \hat{\theta}} = 0$, $c_i^\theta, c_j^u \leq 0$, $\sigma_i, \lambda_j \geq 0$, and $\sigma_i c_i^\theta = 0$, and $\lambda_j c_j^u = 0$, $\forall i \in \mathcal{K}, j \in \mathcal{I}$. Thus, the equilibria characterize candidates for locally optimal solutions of the formulated constrained optimization problem.

The proposed CONAC algorithm is presented in Algorithm 1, implemented in a discrete-time controller, where $(\cdot)^{(r)}$ denotes the value at the r th time step.

IV. STABILITY ANALYSIS

In this section, the stability of the closed-loop system under the proposed CONAC is analyzed. To facilitate the stability analysis, we introduce the following lemmas.

Lemma 1 (Convexity of input constraints with respect to $\hat{\theta}_k$). *The input constraint function $c_j^u(\hat{\theta}_k)$, $\forall j \in \mathcal{I}$, is convex with respect to the output layer weights $\hat{\theta}_k$, resulting in the inequality*

$$c_j^u(\hat{\theta}_k) - \frac{\partial c_j^u}{\partial \hat{\theta}_k}^\top \hat{\theta}_k \leq -c_j^u < 0.$$

Proof. According to Proposition 1, the output of the DNN $\hat{\Phi}$ depends linearly on the output layer weights $\hat{\theta}_k$ as follows:

$$\hat{\Phi} = \widehat{\mathbf{W}}_k^\top \hat{\phi}_k = (\mathbf{I}_{l_{k+1}} \otimes \hat{\phi}_k^\top) \text{vec}(\widehat{\mathbf{W}}_k) = (\mathbf{I}_{l_{k+1}} \otimes \hat{\phi}_k^\top) \hat{\theta}_k. \quad (19)$$

This implies that the control input $\mathbf{u} = \mathbf{P}\hat{\Phi}$ depends linearly on $\hat{\theta}_k$. Therefore, the convexity of $c_j^u(\hat{\theta})$, $\forall j \in \mathcal{I}$, with respect to $\mathbf{u}$ is preserved with respect to $\hat{\theta}_k$, due to the linear dependence of $\mathbf{u}$ on $\hat{\theta}_k$. In addition, the convexity of $c_j^u(\hat{\theta})$

with respect to $\hat{\theta}_k$ implies $c_j^u(\hat{\theta}_k) - \frac{\partial c_j^u}{\partial \hat{\theta}_k}^\top \hat{\theta}_k \leq c_j^u(\hat{\theta})|_{\hat{\theta}_k=\mathbf{0}}$.

Additionally, since $\hat{\theta}_k = \mathbf{0}$ implies $\mathbf{u} = \mathbf{P}\hat{\Phi} = \mathbf{0}$, (i.e., output layer weight is zero), we have $c_j^u(\hat{\theta})|_{\hat{\theta}_k=\mathbf{0}} = -c_j^u < 0$ by Assumption 3, which concludes the proof. $\square$

Lemma 2 (Boundedness of $\Delta \mathbf{u}$ with respect to $\hat{\theta}_k$). *The saturation of control input $\Delta \mathbf{u}$ is bounded by $\hat{\theta}_k$ as*

$$\|\Delta \mathbf{u}\| \leq \bar{p} \sqrt{l_k + 1} \|\hat{\theta}_k\|.$$

Proof. Since $\phi_k(\cdot)$ employs the bounded activation function $|\tanh(\cdot)| < 1$, we have

$$\|\phi_k(\cdot)\| \leq \sqrt{l_k + 1}, \quad (20)$$

for any output of the preceding layers and $\mathbf{x}_{nn}$. Hence, invoking the relation in (19), the control input $\mathbf{u} = \mathbf{P}\hat{\Phi}$ is bounded by $\hat{\theta}_k$ as follows:

$$\|\mathbf{u}\| \leq \bar{p} \|\hat{\Phi}\| \leq \bar{p} \sqrt{l_k + 1} \|\hat{\theta}_k\|. \quad (21)$$

In addition, since there exists $\underline{u} > 0$ such that $\inf_{\mathbf{u} \notin \Omega_u} (\|\text{sat}(\mathbf{u})\|) = \underline{u}$ by Assumption 3, we have $\|\Delta \mathbf{u}\| \leq \|\mathbf{u}\| - \underline{u}$, while $\Delta \mathbf{u} = \mathbf{0}$ when $\mathbf{u} \in \Omega_u$. Thus, we have

$$\|\Delta \mathbf{u}\| \leq \max \left\{ \bar{p} \sqrt{l_k + 1} \|\hat{\theta}_k\| - \underline{u}, 0 \right\} \leq \bar{p} \sqrt{l_k + 1} \|\hat{\theta}_k\|, \quad (22)$$

which concludes the proof. $\square$

Lemma 3. *For any $a, b \in \mathbb{R}_{\geq 0}$ and $c \in \mathbb{R}_{> 0}$, the following inequality holds:*

$$(a - b) \text{Proj}_{a \geq 0}(c) \leq (a - b)c. \quad (23)$$

Proof. The inequality can be verified by considering the two cases of $a > 0$ and $a = 0$, as follows:

$$(a - b) \text{Proj}_{a \geq 0}(c) = \begin{cases} (a - b)c, & \text{if } a > 0, \\ 0, & \text{if } a = 0. \end{cases}$$

This implies that $(a - b) \text{Proj}_{a \geq 0}(c) \leq (a - b)c$ always holds and concludes the proof. $\square$

The following theorem establishes the UUB of the tracking error e , the DNN weights $\hat{\theta}$, and the Lagrange multipliers σ_i , $\forall i \in \mathcal{K}$, and λ_j , $\forall j \in \mathcal{I}$.

Theorem 1. Consider the closed-loop system consisting of (1), the control law (12), and the adaptation law (17). Suppose that Assumptions 1–4 hold and the initial condition belongs to the compact invariant set $\prod_{\ell=0}^k \mathcal{S}_\ell(\bar{V}_\ell)$ constructed in (29) and (34). Then, the tracking error e , the DNN weights $\hat{\theta}$, and the Lagrange multipliers σ_i , $\forall i \in \mathcal{K}$, and λ_j , $\forall j \in \mathcal{I}$, are uniformly ultimately bounded.

Proof. The proof is divided into two steps. We first analyze the UBB of the tracking error e , the output-layer adaptive variables, $\hat{\theta}_k$, and σ_k , and the Lagrange multipliers associated with the input constraints λ_j , $\forall j \in \mathcal{I}$. This step provides the key boundedness result for the tracking error and the controller output. In the following step, we analyze the remaining inner-layer adaptive variables, $\hat{\theta}_i$ and σ_i , $\forall i \in \mathcal{K} \setminus \{k\}$.

Step 1: Boundedness of the tracking error, output-layer and input-constraint adaptive variables.

Consider the following Lyapunov function

$$V_k(z_k) := V_c + \frac{1}{2\alpha} \hat{\theta}_k^\top \hat{\theta}_k + \frac{1}{\beta_k^\theta} (\sigma_k - \sigma_k^o)^2 + \sum_{j \in \mathcal{I}} \frac{1}{2\beta_j^u} \lambda_j^2, \quad (24)$$

where $z_k := \left(e^\top, \hat{\theta}_k^\top, \sigma_k, [\lambda_j]_{j \in \mathcal{I}}^\top \right)^\top$ and $\sigma_k^o \in \mathbb{R}_{>0}$ will be determined later. The Lyapunov function $V_k := V_k(z_k)$ is lower bounded and radially unbounded with respect to z_k . In the following analysis, we consider the case where the output layer weight constraint is violated, i.e., $c_k^\theta(\hat{\theta}_k) > 0$. This is because the boundedness of $\hat{\theta}_k$ directly follows when the output layer weight constraint is not violated, i.e., $c_k^\theta(\hat{\theta}_k) \leq 0$.

The time derivative of V_k can be analyzed as follows:

$$\begin{aligned} \frac{d}{dt} V_k &\leq -\underline{p} \|e\|^2 + e^\top P \left(\hat{\Phi} - \Phi^* - \epsilon \right) + e^\top \Delta u \\ &\quad - \underbrace{\hat{\theta}_k^\top \frac{\partial \hat{\Phi}}{\partial \hat{\theta}_k} P e}_{=\hat{\Phi}^\top P e, \text{ see, (5), (19)}} - \sigma_k \hat{\theta}_k^\top \hat{\theta}_k - \sum_{j \in \mathcal{I}} \lambda_j \hat{\theta}_k^\top \frac{\partial c_j^u}{\partial \hat{\theta}_k} \\ &\quad + 2(\sigma_k - \sigma_k^o) c_k^\theta(\hat{\theta}_k) + \sum_{j \in \mathcal{I}} \lambda_j c_j^u(\hat{\theta}) \quad (25) \\ &\leq -\underline{p} \|e\|^2 + e^\top P (-\Phi^* - \epsilon) + e^\top \Delta u - \sigma_k \left\| \hat{\theta}_k \right\|^2 \\ &\quad + (\sigma_k - \sigma_k^o) \left(\left\| \hat{\theta}_k \right\|^2 - \bar{\theta}_k^2 \right) - \underbrace{\sum_{j \in \mathcal{I}} \lambda_j c_j^u}_{\text{see, Lemma 1}} \\ &= -\underline{p} \|e\|^2 + e^\top P (-\Phi^* - \epsilon) + e^\top \Delta u \\ &\quad - \sigma_k^o \left\| \hat{\theta}_k \right\|^2 - \sigma_k \bar{\theta}_k^2 + \sigma_k^o \bar{\theta}_k^2 - \sum_{j \in \mathcal{I}} \lambda_j c_j^u. \end{aligned}$$

The projection operators in (17b) and (17c) are omitted according to Lemma 3.

For the boundedness of θ_k^* and ϵ , we assume that x_{nn} remains in the approximation region $\Omega_{x_{nn}}$, which will be ensured later in the proof. Hence, θ_k^* and ϵ are bounded by $\bar{\theta}_k^*$ and $\bar{\epsilon}$, respectively. Similarly to the bound of $\hat{\Phi}$, Φ^* is bounded by $\bar{\theta}_k^* \sqrt{l_k + 1}$.

Using Lemma 2, we can further analyze $\frac{d}{dt} V_k$ as follows:

$$\frac{d}{dt} V_k \leq -\underline{p} \|e\|^2 - \sigma_k^o \left\| \hat{\theta}_k \right\|^2 \quad (26)$$

$$\begin{aligned} &+ \bar{p} \|e\| \left(\bar{\theta}_k^* \sqrt{l_k + 1} + \bar{\epsilon} + \sqrt{l_k + 1} \left\| \hat{\theta}_k \right\| \right) \\ &- \sigma_k \bar{\theta}_k^2 + \sigma_k^o \bar{\theta}_k^2 - \sum_{j \in \mathcal{I}} \lambda_j c_j^u. \end{aligned}$$

For further analysis, we use Young's inequality which states $ab \leq \eta a^2 + \frac{1}{4\eta} b^2$, $\forall a, b \in \mathbb{R}, \eta \in \mathbb{R}_{>0}$. By applying Young's inequality to the third term in the right-hand side of (26), we have

$$\begin{aligned} \frac{d}{dt} V_k &\leq -(\underline{p} - \eta_1 - \eta_2) \|e\|^2 - \left(\sigma_k^o - \frac{1}{4\eta_2} \bar{p}^2 (l_k + 1) \right) \left\| \hat{\theta}_k \right\|^2 \\ &\quad - \sigma_k \bar{\theta}_k^2 - \sum_{j \in \mathcal{I}} \lambda_j c_j^u \quad (27) \\ &\quad + \underbrace{\frac{1}{4\eta_1} \left[\bar{p} \left(\bar{\theta}_k^* \sqrt{l_k + 1} + \bar{\epsilon} \right) \right]^2}_{=: \delta_k \in \mathbb{R}_{>0}} + \sigma_k^o \bar{\theta}_k^2, \end{aligned}$$

where $\eta_1, \eta_2 \in \mathbb{R}_{>0}$ and $\sigma_k^o \in \mathbb{R}_{>0}$ are arbitrarily chosen such that $\underline{p} - \eta_1 - \eta_2 > 0$ for given $\underline{p}$, and $\sigma_k^o - \frac{1}{4\eta_1} \bar{p}^2 (l_k + 1) > 0$, respectively.

This result implies that $\frac{d}{dt} V_k$ is negative outside the compact set defined by

$$\Omega_k = \left\{ z_k : a \left\| \left(e^\top, \hat{\theta}_k^\top \right)^\top \right\|^2 + b \left\| \left(\sigma_k, [\lambda_j]_{j \in \mathcal{I}}^\top \right)^\top \right\|_1 \leq \delta_k \right\}, \quad (28)$$

where $a := \min \left\{ \underline{p} - \eta_1 - \eta_2, \sigma_k^o - \frac{1}{4\eta_2} \bar{p}^2 (l_k + 1) \right\} \in \mathbb{R}_{>0}$ and $b := \min \left\{ \bar{\theta}_k^2, \min_{j \in \mathcal{I}} \{ c_j^u \} \right\} \in \mathbb{R}_{>0}$. Moreover, since V_k is lower bounded and radially unbounded, there exists a compact sublevel set

$$\mathcal{S}_k(\bar{V}_k) := \{ z_k : V_k(z_k) \leq \bar{V}_k \}, \quad (29)$$

where $\bar{V}_k \in \mathbb{R}_{>0}$ is arbitrarily chosen such that $\Omega_k \subset \mathcal{S}_k(\bar{V}_k)$. Because $\frac{d}{dt} V_k < 0$ for all $z_k \notin \Omega_k$, z_k initialized in $\mathcal{S}_k(\bar{V}_k)$ will remain in $\mathcal{S}_k(\bar{V}_k)$ for all time and eventually enter Ω_k . Therefore, UBB of e , $\hat{\theta}_k$, σ_k , and λ_j , $\forall j \in \mathcal{I}$, is ensured in $\mathcal{S}_k(\bar{V}_k)$.

In addition, the boundedness of z_k implies that x is bounded, since $e = x - x_d$ and x_d is bounded by Assumption 4. Therefore, since z_k remains bounded in $\mathcal{S}_k(\bar{V}_k)$, all components of x_{nn} remain bounded, implying there exists a compact approximation region $\Omega_{x_{nn}}$. This justifies the boundedness of θ_k^* and ϵ used in this analysis.

Step 2: Boundedness of the inner-layer adaptive variables.

The remaining weights $[\hat{\theta}_i]_{i \in \mathcal{K} \setminus \{k\}}$ and the Lagrange multipliers $[\sigma_i]_{i \in \mathcal{K} \setminus \{k\}}$ are analyzed recursively from the $(k-1)$ th layer to the input layer ($i=0$). Specifically, for an arbitrary layer index $i \in \{0, \dots, k-1\}$, we assume that the subsequent adaptive variables, $\hat{\theta}_{i+1}, \sigma_{i+1}, \dots, \hat{\theta}_k, \sigma_k$, have already been shown to be bounded, and then prove the boundedness of $\hat{\theta}_i$ and σ_i .

Consider the Lyapunov function of the i th layer defined as

$$V_i(z_i) = \frac{1}{2\alpha} \hat{\theta}_i^\top \hat{\theta}_i + \frac{1}{\beta_i^\theta} (\sigma_i - \sigma_i^o)^2, \quad (30)$$

where $z_i := \left(\hat{\theta}_i^\top, \sigma_i \right)^\top$, and $\sigma_i^o \in \mathbb{R}_{>0}$ will be determined in the subsequent analysis. As in the analysis of V_k , the weight constraint $c_i^\theta(\hat{\theta}_i)$ is assumed to be violated, and the projection

operator in (17b) is omitted according to Lemma 3. Taking the time derivative of V_i yields

$$\begin{aligned} \frac{d}{dt} V_i &\leq -\sigma_i \left\| \hat{\theta}_i \right\|^2 - \hat{\theta}_i^\top \frac{\partial \hat{\Phi}}{\partial \hat{\theta}_i}^\top P e \\ &\quad - \sum_{j \in \mathcal{I}} \lambda_j \hat{\theta}_i^\top \frac{\partial \hat{\Phi}}{\partial \hat{\theta}_i}^\top P \frac{\partial c_j^u}{\partial \mathbf{u}} + 2(\sigma_i - \sigma_i^o) c_i^\theta(\hat{\theta}_i) \\ &\leq -\sigma_i \left\| \hat{\theta}_i \right\|^2 + (\sigma_i - \sigma_i^o) \left(\left\| \hat{\theta}_i \right\|^2 - \bar{\theta}_i^2 \right) \\ &\quad - \hat{\theta}_i^\top \underbrace{\left(\frac{\partial \hat{\Phi}}{\partial \hat{\theta}_i}^\top P \left(e + \sum_{j \in \mathcal{I}} \lambda_j \frac{\partial c_j^u}{\partial \mathbf{u}} \right) \right)}_{=: \gamma_i \in \mathbb{R}^{\Xi_i}} \\ &\leq -\sigma_i^o \left\| \hat{\theta}_i \right\|^2 - \sigma_i \bar{\theta}_i^2 + \left\| \hat{\theta}_i \right\| \|\gamma_i\| + \sigma_i^o \bar{\theta}_i^2. \end{aligned} \quad (31)$$

To establish the boundedness of γ_i , we note that: (i) $\frac{\partial \hat{\Phi}}{\partial \hat{\theta}_i}$ is bounded because the subsequent weights $\hat{\theta}_{i+1}, \dots, \hat{\theta}_k$ are bounded under the recursive assumption; (ii) e , $\hat{\theta}_k$, and $[\lambda_j]_{j \in \mathcal{I}}$ are bounded by Step 1; and (iii) $\frac{\partial c_j^u}{\partial \mathbf{u}}$ is bounded over the compact domain containing $\mathbf{u}$, owing to the smoothness of $c_j^u(\cdot)$ in Assumption 3 and the boundedness of $\mathbf{u}$ established in (21). Therefore, there exists an unknown constant $\bar{\gamma}_i \in \mathbb{R}_{>0}$ such that $\|\gamma_i\| \leq \bar{\gamma}_i$.

Then, by applying Young's inequality to the coupled term, i.e., $\left\| \hat{\theta}_i \right\| \|\gamma_i\| \leq \left\| \hat{\theta}_i \right\|^2 + \frac{1}{4} \bar{\gamma}_i^2$, in (31), we have

$$\frac{d}{dt} V_i \leq -(\sigma_i^o - 1) \left\| \hat{\theta}_i \right\|^2 - \sigma_i \bar{\theta}_i^2 + \delta_i, \quad (32)$$

where $\delta_i := \frac{1}{4} \bar{\gamma}_i^2 + \sigma_i^o \bar{\theta}_i^2$ is a positive constant, and $\sigma_i^o \in \mathbb{R}_{>0}$ is arbitrarily chosen such that $\sigma_i^o - 1 > 0$. As in Step 1, $\frac{d}{dt} V_i$ is negative outside the compact set defined by

$$\Omega_i = \left\{ \mathbf{z}_i : (\sigma_i^o - 1) \left\| \hat{\theta}_i \right\|^2 + \bar{\theta}_i^2 |\sigma_i| \leq \delta_i \right\}, \quad (33)$$

which implies the boundedness of $\hat{\theta}_i$ and σ_i . Moreover, a compact sublevel set $\mathcal{S}_i(\bar{V}_i)$ of V_i can be constructed such that

$$\mathcal{S}_i(\bar{V}_i) := \{ \mathbf{z}_i : V_i(\mathbf{z}_i) \leq \bar{V}_i \}, \quad (34)$$

where $\bar{V}_i \in \mathbb{R}_{>0}$ is arbitrarily chosen such that $\Omega_i \subset \mathcal{S}_i(\bar{V}_i)$. Since $\frac{d}{dt} V_i < 0$ for all $\mathbf{z}_i \notin \Omega_i$, $\mathcal{S}_i(\bar{V}_i)$ is positively invariant. Together with the previously established invariance of $\mathcal{S}_{i+1}(\bar{V}_{i+1}), \dots, \mathcal{S}_k(\bar{V}_k)$, this implies that any trajectory initialized in $\prod_{\ell=i}^k \mathcal{S}_\ell(\bar{V}_\ell) = \mathcal{S}_k(\bar{V}_k) \times \dots \times \mathcal{S}_i(\bar{V}_i)$ remains in this set for all time.

By recursively applying the above analysis from $i = k - 1$ to $i = 0$, the boundedness of the control system, including e , $\hat{\theta}$, and σ_i , $\forall i \in \mathcal{K}$, and λ_j , $\forall j \in \mathcal{I}$, is established, which concludes the proof. $\square$

Additionally, we introduce the following remarks regarding the initialization and the parameter tuning of CONAC.

Remark 2 (Initialization of CONAC). *The invariant sublevel set $\prod_{\ell=0}^k \mathcal{S}_\ell(\bar{V}_\ell)$ used in the stability analysis depends on unknown constants. Therefore, the exact admissible initial domain cannot be computed explicitly. In practice, we recommend initializing the system state $\mathbf{x}(0)$ close to the initial*

reference state $\mathbf{x}_d(0)$ to reduce the initial tracking error $e(0)$; DNN weights with small random values; and zero for the Lagrange multipliers, $[\sigma_i(0)]_{i \in \mathcal{K}} = \mathbf{0}$ and $[\lambda_j(0)]_{j \in \mathcal{I}} = \mathbf{0}$, without constraint violations, to help ensure that the system starts within $\prod_{\ell=0}^k \mathcal{S}_\ell(\bar{V}_\ell)$.

Remark 3 (Initial Desired Trajectory Design). *The initialization in Remark 2 helps prevent initial unstable behavior, which generally results from poor initial performance of NAC due to the randomly initialized DNN weights. With small random weights, the initial control input is expected to be small, thereby avoiding abrupt control actions. However, if the system starts near an unstable or highly nonlinear operating region, small random weights may result in unstable behavior or slow adaptation. Hence, it is preferable to design the desired trajectory such that the system starts in a sufficiently stable operating region (e.g., near the stable equilibrium point), and use a conservative desired trajectory during an initial warm-up phase, as introduced in [3].*

Remark 4 (Tuning of CONAC Parameters). *The adaptation gain α and the multiplier update rates β_i^o and β_j^u affect the transient behavior of CONAC. The gain α determines the overall update speed of the DNN weights. A large α may improve the initial learning response but can also induce overshooting or oscillatory behavior, resulting in failure to learn the ideal weights. Conversely, an excessively small α may lead to slow adaptation and poor transient tracking performance. The update rates β_i^o and β_j^u determine how rapidly the Lagrange multipliers respond to weight and input constraint violations, respectively. Larger values accelerate the enforcement of the corresponding constraints by increasing the multiplier response when a violation occurs. However, excessively large update rates may introduce abrupt changes in the adaptation law, leading to chattering or high-frequency oscillations in the control input on the constraint boundary.*

Therefore, α , β_i^o , and β_j^u should be selected to balance learning speed, prompt constraint enforcement, and smooth closed-loop behavior. The effect of the input constraint multiplier update rate β_j^u is evaluated in the parametric study in Section V-B.

V. VALIDATIONS

A. Validation Setup

To validate the proposed CONAC, a real-time experiment was conducted on a two-link robotic manipulator, as shown in Fig. 4. The OpenCR1.0 board [48] with a 32-bit ARM Cortex processor and a 216 MHz clock frequency was used as the microcontroller unit (MCU). The two-link robotic manipulator was actuated by two Cubemars AK10-90 servo motors. The servos were controlled using the MCU, which transmits the torque reference to the corresponding servo via controller area network (CAN) communication at a sampling rate of 500 Hz using a PCAN-USB interface. The system parameters and the schematic diagram are shown in Table II and Fig. 5a, respectively.

The two-link robotic manipulator dynamics are described by the following Euler-Lagrange equation:

$$\mathbf{M}(\mathbf{q}) \frac{d^2}{dt^2} \mathbf{q} + \mathbf{C}(\mathbf{q}, \frac{d}{dt} \mathbf{q}) \frac{d}{dt} \mathbf{q} + \mathbf{G}(\mathbf{q}) = \text{sat}(\boldsymbol{\tau}), \quad (35)$$

where $\mathbf{q} := (q_1, q_2)^\top \in \mathbb{R}^2$ is the joint angle vector, $\boldsymbol{\tau} := (\tau_1, \tau_2)^\top \in \mathbb{R}^2$ is the joint torque vector, $\mathbf{M}(\mathbf{q}) \in \mathbb{R}^{2 \times 2}$ is the inertia matrix, $\mathbf{C}(\mathbf{q}, \frac{d}{dt} \mathbf{q}) \in \mathbb{R}^{2 \times 2}$ is the Coriolis matrix, and $\mathbf{G}(\mathbf{q}) \in \mathbb{R}^2$ is the gravity vector. The inertia matrix $\mathbf{M}(\mathbf{q})$ whose inverse is the control effectiveness matrix, is a symmetric positive-definite matrix according to the physical properties of Euler-Lagrange equation, [1, Lemma 3.1].

To apply CONAC, the system was reformulated into the first-order dynamics form in (1) by defining the state as $\mathbf{x} := \frac{d}{dt} \mathbf{q} + \Lambda \mathbf{q}$, and the control input as $\mathbf{u} := \boldsymbol{\tau}$, where $\Lambda \in \mathbb{R}^{2 \times 2}$ is a positive definite matrix, and $\mathbf{f} := \mathbf{M}^{-1}(\mathbf{q}) (-\mathbf{C}(\mathbf{q}, \frac{d}{dt} \mathbf{q}) \frac{d}{dt} \mathbf{q} - \mathbf{G}(\mathbf{q})) + \Lambda \frac{d}{dt} \mathbf{q}$, and $\mathbf{B} := \mathbf{M}^{-1}(\mathbf{q})$. Moreover, the DNN input vector was defined as

$\mathbf{x}_{nn} := \left((\mathbf{q} - \mathbf{q}_d)^\top, \frac{d}{dt} (\mathbf{q} - \mathbf{q}_d)^\top, \frac{d^2}{dt^2} \mathbf{q}_d^\top, 1 \right)^\top$, since $\mathbf{f}$, $\mathbf{B}$ and $\frac{d}{dt} \mathbf{x}_d$ in (12) are functions of $\mathbf{q}$, $\mathbf{q}_d$, $\frac{d}{dt} \mathbf{q}$, $\frac{d}{dt} \mathbf{q}_d$ and $\frac{d^2}{dt^2} \mathbf{q}_d$.

The input saturation $\text{sat}(\boldsymbol{\tau})$ was defined as follows:

$$\text{sat}(\boldsymbol{\tau}) = \begin{cases} \begin{pmatrix} \text{sign}(\tau_1) \bar{\tau}_1 \\ \text{sign}(\tau_2) \bar{\tau}_2 \end{pmatrix}, & \text{if } |\tau_2| > \bar{\tau}_2 \text{ and } |\tau_1| > \bar{\tau}_1 \\ (\tau_1, \text{sign}(\tau_2) \bar{\tau}_2)^\top, & \text{if } |\tau_2| > \bar{\tau}_2 \text{ and } |\tau_1| \leq \bar{\tau}_1 \\ \frac{\bar{\tau}}{\|\boldsymbol{\tau}\|} \boldsymbol{\tau}, & \text{if } |\tau_2| \leq \bar{\tau}_2 \text{ and } \|\boldsymbol{\tau}\| > \bar{\tau} \\ \boldsymbol{\tau}, & \text{otherwise,} \end{cases} \quad (36)$$

where $\bar{\tau}_2 = 3.8 \text{ N m}$, $\bar{\tau}_1 = 10.32 \text{ N m}$ and $\bar{\tau} = 11 \text{ N m}$, as illustrated in Fig. 5b. The saturation function can be decomposed into three convex input constraints, which can be physically interpreted as the upper and lower torque limit of the second joint and the power source limit, respectively, and are denoted by

$$c_{\tau_2}^u(\boldsymbol{\tau}) := \tau_2 - \bar{\tau}_2, \quad c_{\tau_2}^l(\boldsymbol{\tau}) := -\tau_2 - \bar{\tau}_2, \quad c_{\tau}^p(\boldsymbol{\tau}) := \|\boldsymbol{\tau}\|^2 - \bar{\tau}^2. \quad (37)$$

Accordingly, the input constraint index set is defined as $\mathcal{I} = \{\bar{\tau}_2, \underline{\tau}_2, \bar{\tau}\}$.

The satisfaction of the LICQ of the considered constraints in (37) is analyzed. First, the gradient of the upper and lower torque limit constraints $c_{\tau_2}^u$ and $c_{\tau_2}^l$ are derived as follows:

$$\frac{\partial c_{\tau_2}^u}{\partial \boldsymbol{\theta}} = -\frac{\partial c_{\tau_2}^l}{\partial \boldsymbol{\theta}} = \frac{\partial (\mathbf{P} \hat{\boldsymbol{\Phi}})^\top}{\partial \boldsymbol{\theta}} \begin{pmatrix} 0 \\ 1 \end{pmatrix} = \frac{\partial \hat{\boldsymbol{\Phi}}^\top}{\partial \boldsymbol{\theta}} \mathbf{P}^\top \begin{pmatrix} 0 \\ 1 \end{pmatrix}. \quad (38)$$

Next, the gradient of the power source limit constraint c_{τ}^p is derived as follows:

$$\frac{\partial c_{\tau}^p}{\partial \boldsymbol{\theta}} = \frac{\partial \boldsymbol{\tau}^\top}{\partial \boldsymbol{\theta}} \boldsymbol{\tau} = \frac{\partial (\mathbf{P} \hat{\boldsymbol{\Phi}})^\top}{\partial \boldsymbol{\theta}} \mathbf{P} \hat{\boldsymbol{\Phi}} = \frac{\partial \hat{\boldsymbol{\Phi}}^\top}{\partial \boldsymbol{\theta}} \mathbf{P}^\top \mathbf{P} \hat{\boldsymbol{\Phi}}. \quad (39)$$

Since the upper and lower torque limit constraints are not active simultaneously, the gradients of the active constraints are linearly independent, satisfying the LICQ condition.

The desired trajectory comprised 4 segments that move the system from $\mathbf{q}_d^{(0)} := (-60^\circ, 60^\circ)^\top$ to $\mathbf{q}_d^{(1)} := (45^\circ, -90^\circ)^\top$, then to $\mathbf{q}_d^{(2)} := (-45^\circ, 45^\circ)^\top$, back to $\mathbf{q}_d^{(3)} := \mathbf{q}_d^{(1)}$, and finally to the initial position $\mathbf{q}_d^{(0)}$, as illustrated in Fig. 6. Each

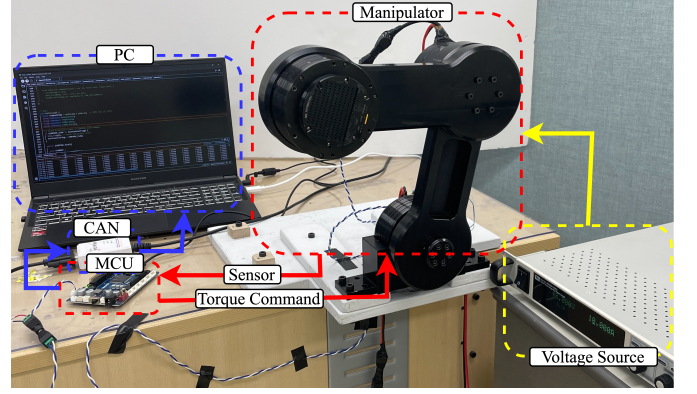


Fig. 4: Experimental setup of the two-link robotic manipulator used for real-time validation.

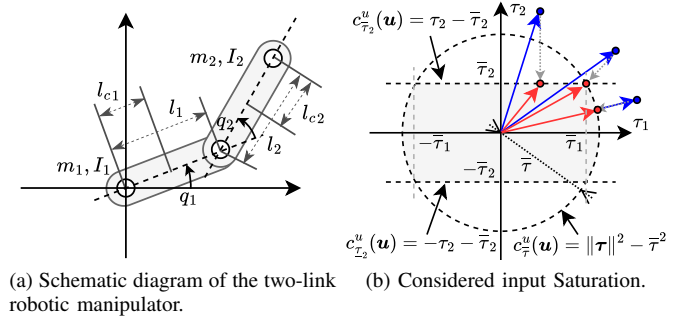


Fig. 5: The two-link robotic manipulator and the considered input saturation.

segment was designed using a quintic (5th-order) polynomial trajectory to ensure the smoothness of the desired trajectory, with zero initial and final velocities. The time duration of the segments was defined as 2 s. To demonstrate the learning capability of the DNN, the desired trajectory was applied twice, referred to as Episode 1 and Episode 2, respectively.

As discussed in Remark 3, a warm-up phase was introduced prior to applying the desired trajectory to avoid the instability caused by the initial poor performance of the DNN. In the warm-up phase, the manipulator was initially positioned at a stable equilibrium point, denoted as $\mathbf{q}_d^{\text{origin}} := (-90^\circ, 0^\circ)^\top$, and remained at this position for 3 s. Subsequently, a conservative desired trajectory was applied to gently excite the system, which was designed as a quintic polynomial trajectory that moves the system from $\mathbf{q}_d^{\text{origin}}$ to $\mathbf{q}_d^{(0)}$ in 3 s, followed by a constant reference that holds the system at $\mathbf{q}_d^{(0)}$ for the next 8 s.

For a comparative analysis, four controllers were implemented, with their properties summarized in Table III. In the comparative study, optimization-based methods, such as MPC and CBF-QP, were not included because the focus is on NAC methods that handle input constraints. This choice isolates the proposed contribution, namely the integration of multiple convex input constraints into the online adaptation law of NAC while retaining a Lyapunov-based stability analysis. In

TABLE II: Two-link robotic manipulator's parameters.

Symbol	Description	Value
m_1, m_2	Mass of the links	2.465 kg
l_1, l_2	Length of the links	0.2 m
l_{c1}, l_{c2}	Length to the center of mass	0.139 m
I_1, I_2	Inertia of the links	0.069 kg m ²

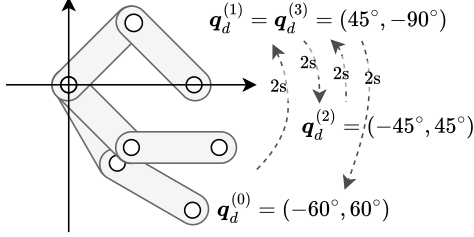


Fig. 6: The desired trajectory in Episode 1 and Episode 2.

addition, among the structural methods, the auxiliary system-based NAC was included as a comparative controller, since it has better input constraint handling capability than smooth approximation methods [27].

Controller 1 (C₁)—Proposed CONAC: To handle all imposed input saturation, the proposed CONAC was implemented with the torque limit constraints $c_{\tau_2}^u$ and $c_{\tau_2}^l$ for the second link, and the power source limit constraint c_{τ}^u (defined in (37)), allowing the controller to fully utilize the authority of the actuator. As described in (12), the input (torque) corresponds to the output of the DNN, and the adaptation law is defined in (17). All parameters of (C₁) are listed in Table III.

Controller 2 (C₂)—Proposed CONAC with small update rate β_j^u , $\forall \lambda_j, \forall j \in \mathcal{I}$: For the parametric study on β_j^u mentioned in Remark 4, the proposed CONAC was implemented with the same input constraints as in (C₁), but with smaller update rates for the input constraints, as listed in Table III. All other properties and parameters were kept identical to those of (C₁).

Controller 3 (C₃)—NAC with auxiliary system: As a baseline for comparison in handling input constraints in NAC, the auxiliary system used in [14], [18], [19] was adopted. It should be noted that the auxiliary system is limited to handling only the element-wise input bound constraints. The auxiliary system is defined by

$$\frac{d}{dt}\zeta = A_\zeta \zeta + B_\zeta \Delta_\zeta \tau, \quad \zeta(0) = \mathbf{0}_{2 \times 1}, \quad (40)$$

where $\zeta \in \mathbb{R}^2$ denotes the auxiliary state which is initialized to zero, $A_\zeta \in \mathbb{R}^{2 \times 2}$ is a stable matrix, $B_\zeta \in \mathbb{R}^{2 \times 2}$ is a gain matrix, and $\Delta_\zeta \tau_i := \tau_i - \max\{\min\{\tau_i, \bar{\tau}_i\}, -\bar{\tau}_i\}$, $\forall i \in \{1, 2\}$. It is notable that $\Delta_\zeta \tau$ is defined as the element-wise input overflow, which is different from the actual input saturation Δu in (8). The auxiliary state was used in the adaptation law (18), by substituting the tracking error e with the auxiliary error $e_\zeta := e + \zeta$. The parameters of the auxiliary system were chosen to balance the responsiveness to input overflow with the convergence speed of the auxiliary dynamics, as listed

TABLE III: Controllers and parameters used for comparative analysis.

	Description and parameters	Input constraint
(C ₁) [—]	Proposed CONAC $\beta_{\tau_2}^u = \beta_{\tau_2}^l = 1000, \beta_{\tau}^u = 100,$ $\bar{\theta}_i = 15, \beta_i^q = 1, \forall i \in \{0, 1, 2\},$	$-\bar{\tau}_2 \leq \tau_2 \leq \bar{\tau}_2,$ $\ \tau\ \leq \bar{\tau},$ ($\bar{\tau}_2 = 3.8, \bar{\tau} = 11$)
(C ₂) [—]	Proposed CONAC (smaller β_j) $\beta_{\tau_2}^u = \beta_{\tau_2}^l = 50, \beta_{\tau}^u = 10,$ Others: same as (C ₁)	same as (C ₁)
(C ₃) [—]	NAC with auxiliary system [14], [18], [19] $A_\zeta = \text{diag}(-120, -100),$ $B_\zeta = \text{diag}(5 \times 10^3, 5 \times 10^3),$ $\bar{\theta}_i = 15, \beta_i^q = 1, \forall i \in \{0, 1, 2\}$	$-\bar{\tau}_1 \leq \tau_1 \leq \bar{\tau}_1,$ $-\bar{\tau}_2 \leq \tau_2 \leq \bar{\tau}_2,$ ($\bar{\tau}_1 = 10.32, \bar{\tau}_2 = 3.8$)
(C ₄) [—]	NAC without input constraints [7] $\bar{\theta} = 15$	not considered

DNN and other parameters for all controllers: $\alpha = 0.2, \mathbf{P} = \mathbf{I}_2,$
 $\mathbf{x}_{nn} = ((\mathbf{q} - \mathbf{q}_d)^\top, \frac{d}{dt}(\mathbf{q} - \mathbf{q}_d)^\top, \frac{d^2}{dt^2} \mathbf{q}_d^\top, 1)^\top, \mathbf{A} = \text{diag}(6, 13),$
 $l_0 = 6, l_1 = 4, l_2 = 4, l_3 = 2, \hat{\boldsymbol{\theta}}(0) \sim \mathcal{U}([-0.5, 0.5]^\Xi)$

in Table III. All other properties and parameters were kept the same as in (C₁) and (C₂).

Controller 4 (C₄)—NAC without input constraints: The fourth controller, adopted from [7], does not handle input constraints. The controller structure is identical to (C₁)–(C₃), except that the input constraints were removed and weight boundedness was handled using the projection operator, as follows:

$$\frac{d}{dt}\hat{\boldsymbol{\theta}} = \text{Proj}_{\|\hat{\boldsymbol{\theta}}\| \leq \bar{\theta}} \left(-\alpha \frac{\partial \Phi}{\partial \boldsymbol{\theta}}^\top \mathbf{P} \mathbf{e} \right). \quad (41)$$

The weight norm bound was set to $\bar{\theta} = 15$.

For all controllers, the remaining parameters, including the DNN architecture, were kept identical to ensure a fair comparison, as listed in Table III. Specifically, the architecture of the DNNs are empirically chosen considering the approximation capability and computational efficiency. The DNN weights were initialized with small random values according to $\hat{\boldsymbol{\theta}}(0) \sim \mathcal{U}([-0.5, 0.5]^\Xi)$, where $\mathcal{U}(\cdot)$ denotes a uniform random distribution, invoking Remark 2.

B. Validation Results

1) Tracking and Adaptation Performance: The real-time implementation results are illustrated in Fig. 7. As shown in Fig. 7a and Fig. 7b, all controllers, except (C₄), tracked the desired trajectory with their tracking performance improving from Episode 1 to Episode 2. It should be noted that, due to the input saturation effect, the tracking performance of all controllers is limited. A quantitative comparison is summarized in Fig. 8, using the root mean square error (RMSE) of the tracking error for each joint angle during 2 episodes. As shown in Fig. 8, all controllers, except (C₄), improved their tracking performance from Episode 1 to Episode 2 by around 25% and 70% for the first and second joint angles, respectively, demonstrating the adaptation capability of the DNN.

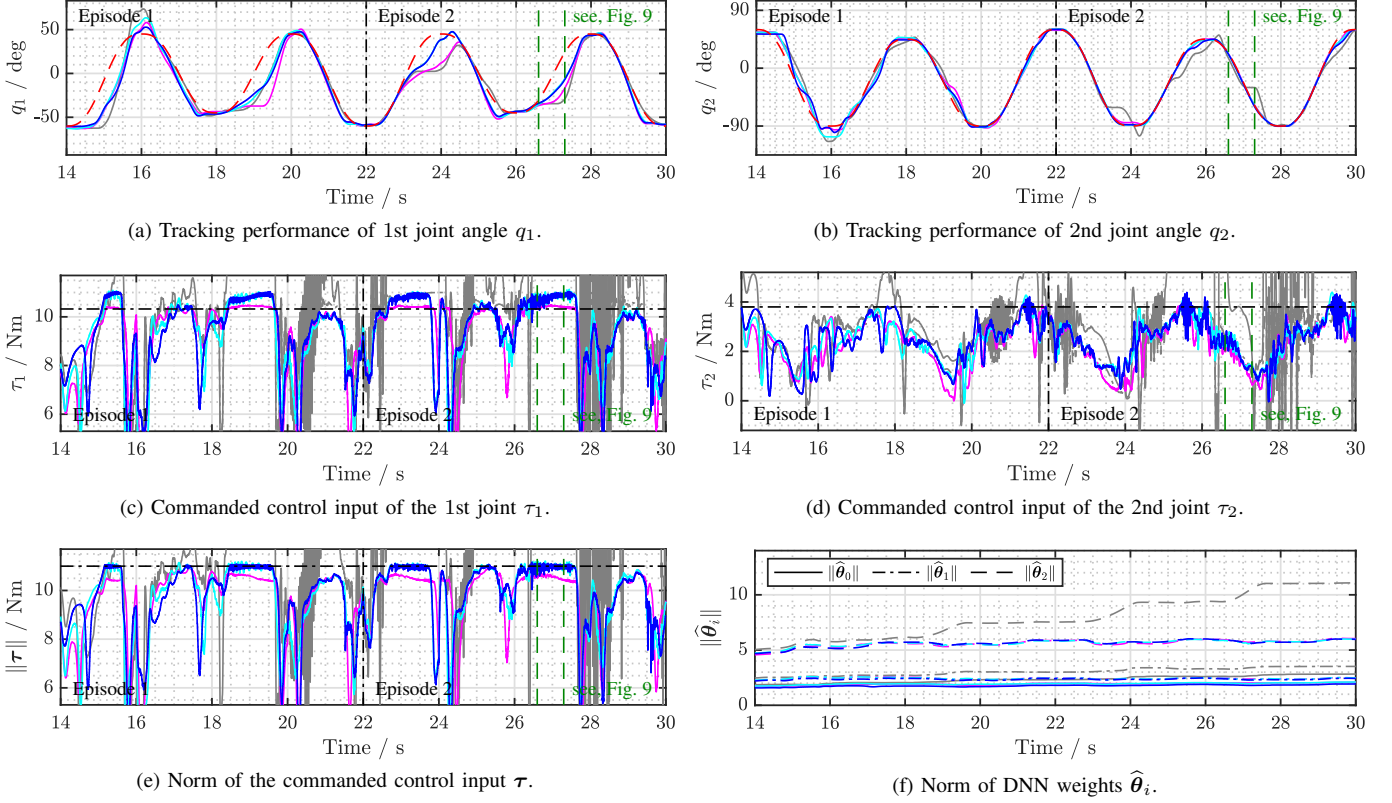


Fig. 7: Real-time implementation results of (C₁) [—], (C₂) [—], (C₃) [—], (C₄) [—], and desired trajectory q_d [---].

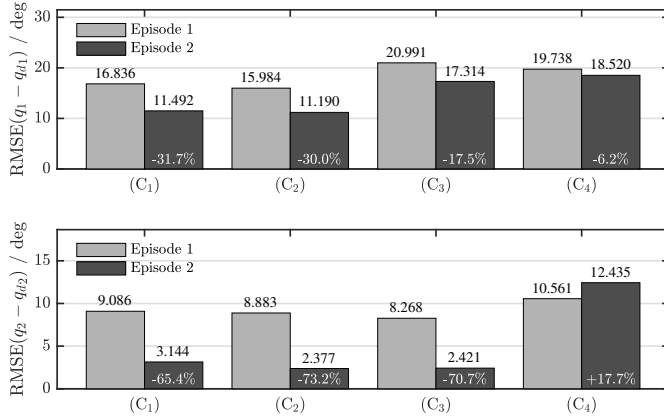


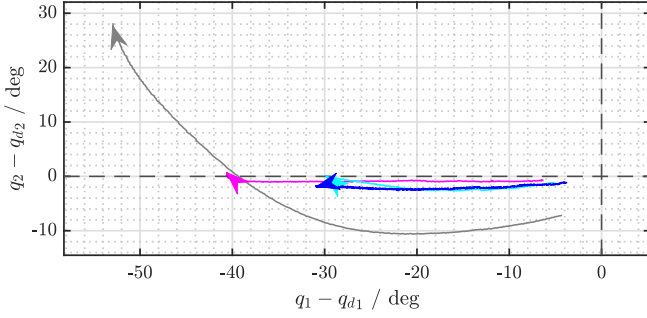
Fig. 8: Quantitative comparison of tracking performance.

Remarkably, the proposed CONAC (C₁) and (C₂) demonstrated superior tracking performance compared to the other controllers. This is because (C₃) could not fully utilize the actuator authority due to the auxiliary system design, which will be discussed in detail in Section V-B2. For (C₄), since it does not consider the input saturation in the adaptation process, the DNN weights were continuously increased in an attempt to reduce the tracking error, even though the corresponding control input could not be realized by the saturated actuators. Consequently, the DNN weights exhibited persistent growth rather than convergence as shown in Fig. 7f, which may eventually lead to parameter drift or unbounded

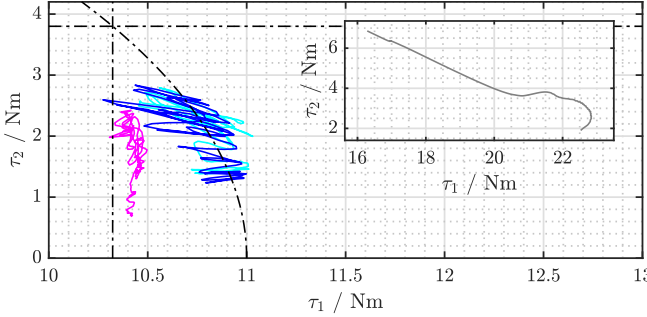
weight growth and is therefore undesirable.

2) *Constraint Handling Performance*: In this section, the constraint-handling performance of the controllers is analyzed. For a detailed analysis, we focus on the time interval [26.6, 27.3 s] of Episode 2 which is highlighted by the green dashed lines in Fig. 7, where the difference in the considered input constraints among the controllers is most pronounced. As shown in Fig. 9a, the tracking errors of all controllers were observed to be increased due to the input saturation effect, but (C₁) and (C₂) exhibited smaller tracking errors compared to (C₃) and (C₄). In Fig. 9b, the commanded control inputs of (C₁) and (C₂) were observed to properly handle the input constraints, using the full authority of the actuator. In contrast, the commanded control input of (C₃) was saturated at the box constraints, which is attributed to the design of the auxiliary system that only constrains the element-wise input overflow, rather than the actual input saturation, as illustrated in Fig. 10. Finally, the commanded control input of (C₄) was observed to exceed the input limits, which is expected since (C₄) does not consider input constraints in its adaptation process.

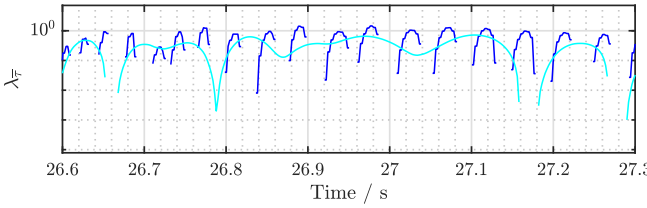
The parametric study of the Lagrange multiplier update rates is conducted by comparing (C₁) and (C₂), focusing on the multiplier associated with the power source limit constraint $\lambda_{\bar{\tau}}$. In Fig. 9c, $\lambda_{\bar{\tau}}$ of (C₁) and (C₂) was observed to be non-zero, indicating that the power source limit constraint was active and enforced during this time interval. Specifically, the Lagrange multiplier $\lambda_{\bar{\tau}}$ of (C₁), which had a larger update rate, repeatedly grew and decayed, while that of (C₂) remained rela-



(a) Trajectory of the tracking error of each joint angle $\mathbf{q}$ during the time interval [26.6, 27.3 s].



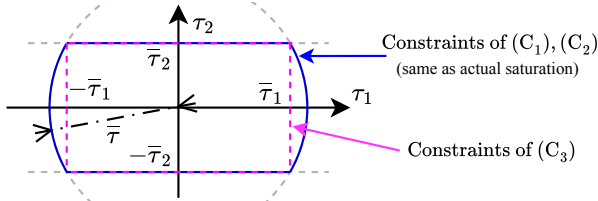
(b) Trajectory of the commanded control input τ during the time interval [26.6, 27.3 s].



(c) Lagrange multiplier associated with the power source limit constraint for (C1) and (C2) during the time interval [26.6, 27.3 s].

Fig. 9: Zoomed-in view of the tracking error, commanded control input, and Lagrange multiplier λ_{τ} during the time interval [26.6, 27.3 s] of (C1) [—], (C2) [—], (C3) [—], and (C4) [—].

Fig. 10: Considered input saturation in (C1), (C2), and (C3).



tively constant. Consequently, a larger update rate β_{τ}^u provides a faster response to the input constraint but induces greater oscillations in the Lagrange multiplier, whereas a smaller update rate yields smoother but less responsive constraint enforcement. This result confirms the trade-off discussed in Remark 4.

Additionally, the nearly constant value of λ_{τ} in Fig. 9c and the small oscillations of c_{τ}^u in Fig. 9b of (C2) suggest that the adaptation dynamics remain near an equilibrium characterized

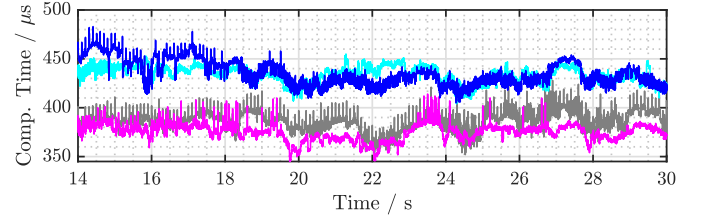


Fig. 11: Computation time of (C1) [—], (C2) [—], (C3) [—], and (C4) [—] on the OpenCR1.0 board.

by a nonzero Lagrange multiplier. This behavior is qualitatively consistent with the KKT-like optimality conditions discussed in Section III-C, according to which, at an equilibrium, the positive Lagrange multiplier λ_{τ} is associated with active constraint satisfying $c_{\tau}^u = 0$ through the complementary slackness relation $\lambda_{\tau} c_{\tau}^u = 0$. Although the adaptation dynamics do not converge exactly to a stationary KKT equilibrium because of the practical limitations in the real-time implementation, the observed behavior can be interpreted as a near-equilibrium behavior.

3) *Feasibility of CONAC in Real-Time Implementation:* The feasibility of the real-time implementation is analyzed in terms of the computation time. As shown in Fig. 11, the computation time of all controllers is consistently below the sampling time of 500 μ s, confirming the feasibility of real-time implementation at a sampling frequency of 500 Hz (2 ms maximum sampling time). This is attributed to the fact that the proposed CONAC does not require solving an optimization problem at each time step, but rather continuously adapts the adaptive variables to solve the optimization problem in an online manner.

C. Discussion on the Advantages and Limitations of CONAC

In the validations, we demonstrated that CONAC can handle multiple convex input constraints simultaneously, while the other NAC methods failed to do so. However, it is important to note that the proposed CONAC is only able to handle the constraints in soft manner, meaning that the constraints may be violated during the adaptation process. One may impose a conservative constraint considering the potential violation of the constraints, but this may lead to a suboptimal solution. In safety perspective, some constraints must be satisfied without violation in operation time. Consequently, extension of the proposed CONAC to incorporate hard constraints remains an important direction for future work.

D. Video Demonstration

A supplementary video demonstrating the experimental results is provided with this paper.

VI. CONCLUSION

This paper proposed CONAC for unknown MIMO systems, addressing multiple convex input constraints within a unified constrained optimization framework. The adaptation law was derived from the constrained optimization problem, which

was formulated to minimize the tracking error while satisfying the multiple convex input constraints. The stability of CONAC was analyzed using Lyapunov theory, demonstrating the boundedness of the tracking errors and adaptive variables, including the DNN weights and Lagrange multipliers. By deriving adaptation law from the constrained optimization problem, CONAC ensured weight convergence in accordance with the KKT optimality conditions, thereby ensuring both stability and optimality.

Real-time implementation demonstrated the capability of CONAC in handling multiple convex input constraints simultaneously, while achieving superior tracking performance compared to other NAC methods. Moreover, the parametric study revealed the effect of the update rates of the Lagrange multipliers on the enforcement of input constraints. In addition, the experiments validated the feasibility of CONAC in real-time implementation.

Future work may extend this approach to handle hard constraints, such as state constraints, in addition to input constraints. Moreover, by incorporating reinforcement learning into the optimization problem formulation and deriving adaptation law that explicitly accounts for disturbances, the proposed CONAC framework could be extended to infinite-horizon optimal control problems while further enhancing its robustness.

REFERENCES

- [1] R. Ji, S. S. Ge, and D. Li, *Constrained nonlinear control: Performance and robustness*. Cham, Switzerland: Springer Nature, May 2025.
- [2] G. Chen, J. Yao, H. Hu, Z. Gao, and M. Hua, "Hierarchical coupling motion control of distributed vehicle considering vehicle posture," *IEEE Transactions on Vehicular Technology*, vol. 73, no. 8, pp. 10 899–10 915, 2024.
- [3] M. Ryu, N. Monzen, P. Seitter, K. Choi, and C. M. Hackl, "Constrained optimization-based neuro-adaptive control (CONAC) for synchronous machine drives under voltage constraints," in *IECON 2025 – 51st Annual Conference of the IEEE Industrial Electronics Society*, 2025, pp. 1–7.
- [4] N. Monzen and C. M. Hackl, "Optimal reference voltage saturation for nonlinear current control of synchronous machine drives," *IEEE Transactions on Industry Applications*, vol. 62, no. 1, pp. 1029–1038, 2026.
- [5] P. Ioannou and B. Fidan, *Adaptive Control Tutorial*. Philadelphia, PA: Society for Industrial and Applied Mathematics, 2006.
- [6] J. A. Farrell and M. M. Polycarpou, *Adaptive Approximation Based Control: Unifying Neural, Fuzzy and Traditional Adaptive Approximation Approaches (Adaptive and Learning Systems for Signal Processing, Communications and Control Series)*. USA: Wiley-Interscience, 2006.
- [7] O. S. Patil, D. M. Le, M. L. Greene, and W. E. Dixon, "Lyapunov-derived control and adaptive update laws for inner and outer layer weights of a deep neural network," *IEEE Control Systems Letters*, vol. 6, pp. 1855–1860, 2022.
- [8] S. You, G. Kim, S. Lee, D. Shin, and W. Kim, "Neural approximation-based adaptive control using reinforced gain for steering wheel torque tracking of electric power steering system," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 53, no. 7, pp. 4216–4225, 2023.
- [9] H. Gao, J. Liu, Z. Yu, and C. Sun, "Reinforcement learning-based adaptive vibration control of flexible two-link manipulator systems with input saturation," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, pp. 1–13, 2026.
- [10] S. Ma, K. Li, and Y. Li, "Neuroadaptive consensus control for heterogeneous port uncrewed container transporter platoon via sliding mode estimators," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 55, no. 11, pp. 8261–8274, 2025.
- [11] C. Yang, D. Huang, W. He, and L. Cheng, "Neural control of robot manipulators with trajectory tracking constraints and input saturation," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 32, no. 9, pp. 4231–4242, 2021.
- [12] S. Jia and J. Shan, "Finite-time trajectory tracking control of space manipulator under actuator saturation," *IEEE Transactions on Industrial Electronics*, vol. 67, no. 3, pp. 2086–2096, 2020.
- [13] Z. Liu, C. Wu, X. Shen, W. Yao, J. Liu, and L. Wu, "Adaptive interval type-2 fuzzy neural network-based novel fixed-time backstepping control for uncertain euler-lagrange systems," *IEEE Transactions on Fuzzy Systems*, vol. 32, no. 5, pp. 2966–2975, 2024.
- [14] Z. Liu, O. Zhang, Y. Zhao, Q. Zhu, and J. Liu, "Adaptive neural network-based fixed-time control for robots with input saturation and prescribed performance," *Nonlinear Dynamics*, vol. 113, no. 14, pp. 18 229–18 241, 2025.
- [15] Y.-F. Gao, X.-M. Sun, C. Wen, and W. Wang, "Observer-based adaptive nn control for a class of uncertain nonlinear systems with nonsymmetric input saturation," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 28, no. 7, pp. 1520–1530, 2017.
- [16] K. Esfandiari, F. Abdollahi, and H. Talebi, "A stable nonlinear in parameter neural network controller for a class of saturated nonlinear systems," *IFAC Proceedings Volumes*, vol. 47, no. 3, pp. 2533–2538, 2014, 19th IFAC World Congress.
- [17] K. Esfandiari, F. Abdollahi, and H. A. Talebi, "Adaptive control of uncertain nonaffine nonlinear systems with input saturation using neural networks," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 26, no. 10, pp. 2311–2322, 2015.
- [18] D. Yu, J. Long, C. L. P. Chen, and Z. Wang, "Adaptive swarm control within saturated input based on nonlinear coupling degree," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 52, no. 8, pp. 4900–4911, 2022.
- [19] M. H. Rezaei, M. Ghaseminezhad, and M. Kabiri, "Control of uncertain non-affine nonlinear systems using neural networks subject to input saturation with unknown control direction," *JVC/Journal of Vibration and Control*, vol. 31, no. 17-18, p. 3647 – 3660, 2025, cited by: 4.
- [20] W. He, Y. Dong, and C. Sun, "Adaptive neural impedance control of a robotic manipulator with input saturation," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 46, no. 3, pp. 334–344, 2016.
- [21] W. He, Z. Li, Y. Dong, and T. Zhao, "Design and adaptive control for an upper limb robotic exoskeleton in presence of input saturation," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 30, no. 1, pp. 97–108, 2019.
- [22] F. Zhao, S. S. Ge, F. Tu, Y. Qin, and M. Dong, "Adaptive neural network control for active suspension system with actuator saturation," *IET Control Theory & Applications*, vol. 10, no. 14, pp. 1696–1705, 2016.
- [23] G. Peng, C. Yang, W. He, and C. L. P. Chen, "Force sensorless admittance control with neural learning for robots with actuator saturation," *IEEE Transactions on Industrial Electronics*, vol. 67, no. 4, pp. 3138–3148, 2020.
- [24] H. Wang, B. Chen, X. Liu, K. Liu, and C. Lin, "Robust adaptive fuzzy tracking control for pure-feedback stochastic nonlinear systems with input constraints," *IEEE Transactions on Cybernetics*, vol. 43, no. 6, pp. 2093–2104, 2013.
- [25] G. Cui, T. Jiao, Y. Wei, G. Song, and Y. Chu, "Adaptive neural control of stochastic nonlinear systems with multiple time-varying delays and input saturation," *Neural Computing and Applications*, vol. 25, no. 3-4, p. 779 – 791, 2014, cited by: 32.
- [26] J. Li, R. Ji, X. Liang, H. Yan, and S. S. Ge, "Finite-time adaptive fuzzy control of nonlinear systems with actuator faults and input saturation," *Neural Comput. Appl.*, vol. 36, no. 7, pp. 3569–3581, Mar. 2024.
- [27] H. Min, S. Xu, and Z. Zhang, "Adaptive finite-time stabilization of stochastic nonlinear systems subject to full-state constraints and input saturation," *IEEE Transactions on Automatic Control*, vol. 66, no. 3, pp. 1306–1313, 2021.
- [28] D. Zhang, Y. Wang, L. Meng, J. Yan, and C. Qin, "Adaptive critic design for safety-optimal ftc of unknown nonlinear systems with asymmetric constrained-input," *ISA Transactions*, vol. 155, pp. 309–318, 2024.
- [29] C. Qin, M. Pang, Z. Wang, S. Hou, and D. Zhang, "Observer based fault tolerant control design for saturated nonlinear systems with full state constraints via a novel event-triggered mechanism," *Engineering Applications of Artificial Intelligence*, vol. 161, p. 112221, 2025.
- [30] L. Greune, L. Grune, and J. Pannek, *Nonlinear model predictive control*, ser. Communications and Control Engineering Series. London, England: Springer, Jan. 2011.
- [31] A. Bemporad and M. Morari, "Robust model predictive control: A survey," in *Robustness in identification and control*, A. Garulli and A. Tesi, Eds. London: Springer London, 1999, pp. 207–226.
- [32] J. Berberich and F. Allgöwer, "An overview of systems-theoretic guarantees in data-driven model predictive control," *Annual Review of Control*,

- Robotics, and Autonomous Systems, vol. 8, no. Volume 8, 2025, pp. 77–100, 2025.
- [33] A. Draeger, S. Engell, and H. Ranke, “Model predictive control using neural networks,” *IEEE Control Systems Magazine*, vol. 15, no. 5, pp. 61–66, 1995.
- [34] A. Vahidi-Moghaddam, K. Chen, K. Zhang, Z. Li, Y. Wang, and K. Wu, “Safe data-driven predictive control,” *IEEE Transactions on Automation Science and Engineering*, vol. 22, pp. 14 422–14 435, 2025.
- [35] S. Mallick, A. Dabiri, and B. De Schutter, “Learning-based model predictive control for piecewise affine systems with feasibility guarantees,” in *2025 European Control Conference (ECC)*, 2025, pp. 345–350.
- [36] A. Khodaverdian, D. Gohil, and P. D. Christofides, “Neural network implementation of model predictive control with stability guarantees,” *Digital Chemical Engineering*, vol. 16, p. 100262, 2025.
- [37] A. D. Ames, S. Coogan, M. Egerstedt, G. Notomista, K. Sreenath, and P. Tabuada, “Control barrier functions: Theory and applications,” in *2019 18th European Control Conference (ECC)*, 2019, pp. 3420–3431.
- [38] D. Ko and W. K. Chung, “A backup control barrier function approach for safety-critical control of mechanical systems under multiple constraints,” *IEEE/ASME Transactions on Mechatronics*, vol. 30, no. 6, pp. 4460–4471, 2025.
- [39] H. M. Sweatland, O. S. Patil, and W. E. Dixon, “Adaptive deep neural network-based control barrier functions,” 2025, arXiv:2406.14430.
- [40] M. Ryu, J. Kim, and K. Choi, “Imposing a weight norm constraint for neuro-adaptive control,” in *2025 European Control Conference (ECC)*, 2025, pp. 380–385.
- [41] J. Nocedal and S. Wright, *Numerical optimization*, 2nd ed., ser. Springer series in operations research and financial engineering. New York, NY: Springer, 2006.
- [42] D. S. Bernstein and D. S. Bernstein, *Matrix Mathematics: Theory, Facts, and Formulas (Second Edition)*. Princeton: Princeton University Press, 2009.
- [43] M. Krstic, P. V. Kokotovic, and I. Kanellakopoulos, *Nonlinear and Adaptive Control Design*, 1st ed. USA: John Wiley & Sons, Inc., 1995.
- [44] A. Boulkroune, M. Tadjine, M. M’Saad, and M. Farza, “Fuzzy adaptive controller for mimo nonlinear systems with known and unknown control direction,” *Fuzzy Sets and Systems*, vol. 161, no. 6, pp. 797–820, 2010, theme: “Fuzzy Control”.
- [45] H. K. Khalil, *Nonlinear systems; 3rd ed.* Upper Saddle River, NJ: Prentice-Hall, 2002.
- [46] I. Douratsos and J. B. Gomm, “Neural network based model reference adaptive control for processes with time delay,” 2007.
- [47] M. Saerens and A. Soquet, “Neural controller based on back-propagation algorithm,” *IEE Proceedings F (Radar and Signal Processing)*, vol. 138, pp. 55–62, 1991.
- [48] ROBOTIS. (2016) OpenCR1.0 [Online]. Available: <https://www.robotis.us/opencr1-0/>. Accessed: Apr. 10, 2026.