International Journal of Robust and Nonlinear Control

WILEY

RESEARCH ARTICLE OPEN ACCESS

Validation of Neural Network Controllers for Uncertain Systems Using the Keep-Close Approach: Robustness Analysis and Safety Verification

Abdelhafid Zenati | Nabil Aouf

Department of Electrical and Electronic Engineering, School of Mathematics, Computer Science and Engineering, City, University of London, London, UK

Correspondence: Abdelhafid Zenati (abdelhafid.zenati@city.ac.uk)

Received: 26 March 2025 | **Revised:** 10 July 2026 | **Accepted:** 27 July 2026

Keywords: IQCs | Lyapunov | NN controller | robustness analysis | safety and verification

ABSTRACT

The safety verification of neural network (NN) controllers operating in uncertain environments characterized by unmodeled dynamics, nonlinearities, and time delays remains a fundamental challenge in robust control analysis. This article introduces a novel method, termed *Keep-Close*, for analyzing the performance and robustness of uncertain feedback systems equipped with NN controllers. The proposed framework formulates the problem as an analysis of the dynamical deviation between an NN-controlled uncertain system and a robust reference model. First, the behavior of the NN controller is characterized using the Differential Mean Value Theorem (DMV) and linear approximation techniques. A new dynamical system is then constructed to describe this deviation, enabling worst-case analysis of the Relative Integral Square Error (RISE) and the Supreme Square Error (SSE) through the combined use of Integral Quadratic Constraints (IQCs) and Lyapunov theory. The effectiveness of the proposed approach is demonstrated through two case studies, namely the Single-Link Robot Arm and the Apollo Lander, highlighting its versatility and capability in assessing the robustness and performance of NN controllers in complex and uncertain environments.

1 | Introduction

Neural networks (NNs) have demonstrated remarkable performance across a broad range of artificial intelligence (AI) applications, and their integration into feedback control systems has consequently attracted significant research interest [1, 2]. Despite these advances, NN-based controllers still present major challenges when deployed in safety-critical environments. These challenges arise from their opaque internal structure, sensitivity to disturbances and uncertainty-induced perturbations, and the presence of nonlinear activation functions that complicate analytical treatment [2–4]. As a result, the direct application of classical stability and robustness analysis tools, such as Lyapunov-based methods, remains non-trivial. This limitation continues to hinder the widespread adoption of NN controllers in domains such as aerospace, robotics, and autonomous systems, where formal certification of safety and robustness is a fundamental requirement [5–8]. Recent developments in neuro-control and adaptive dynamic programming (ADP) have further highlighted both the potential and the inherent challenges of data-driven control frameworks. For instance, self-triggered approximate optimal neuro-control strategies [9] and adaptive critic-based safety-optimal control architectures [10] have demonstrated promising performance in nonlinear and uncertain environments. However, these approaches typically rely on strong assumptions regarding system structure, availability of accurate models, or restrictive constraint formulations, which may limit their applicability in scenarios where rigorous

Abbreviations: ABC, a black cat; DEF, doesn't ever fret; GHI, goes home immediately.

worst-case robustness guarantees and safety certification are required.

Over the past decades, Integral Quadratic Constraints (IQCs) have emerged as a rigorous and powerful framework for the analysis of uncertain dynamical systems [11, 12]. IQCs provide a unified mathematical representation for a wide class of uncertainties, including slope-restricted nonlinearities, unmodeled dynamics, time delays, and structured perturbations [13, 14]. Owing to their flexibility and generality, IQCs have become a central methodology for certifying stability and performance through convex optimization formulations [15]. This framework has enabled significant advances in regional stability analysis via sum-of-squares (SOS) relaxations [16], multiplier-dependent extensions of the Kalman-Yakubovich-Popov (KYP) lemma [17, 18], and local gain certification for interconnected nonlinear systems [19].

More recently, IQC-based techniques have been extended to the analysis and verification of NN-controlled systems. These include reachability-driven verification methods [20–22] as well as input-output characterizations that capture NN-induced nonlinearities within a tractable analytical framework [4]. While these approaches provide valuable guarantees on stability or reachable behavior, they typically focus on certifying invariance or boundedness properties rather than quantifying explicit worst-case performance deviations relative to a prescribed reference closed-loop behavior. In parallel, advances in safety-critical learning and adaptive control have emphasized embedding safety constraints directly into the controller synthesis process. For example, event-triggered H_∞ control for uncertain nonlinear systems with constrained inputs [23], barrier-critic adaptive robust control in differential game settings [24], and state-constrained reinforcement learning-based control architectures [25] demonstrate how neural approximators can enforce safety during online learning and control. However, these approaches fundamentally address *controller synthesis* rather than *controller validation*. They typically require modifying the controller architecture, incorporating barrier or critic structures, or solving Hamilton–Jacobi–Bellman or Isaacs equations, which can be computationally intensive and impractical in scenarios where a neural network controller has already been trained and must be certified without altering its structure.

In this article, we address the problem of certifying safety and providing robustness guarantees for neural network (NN) controllers in the presence of structured system uncertainties, including unmodeled dynamics, slope-restricted nonlinearities, and time delays. Inspired by the comparison principle [26, 27], the core idea is to quantify and certify the deviation between the output of an uncertain system governed by a trained NN controller and that of an ideal reference closed-loop model when subjected to bounded inputs and disturbances. This perspective enables safety and robustness verification to be formulated as a deviation analysis problem relative to a trusted reference behavior. To this end, we propose a novel framework, termed the *keep-close* approach, for analyzing and certifying the dynamical deviation between an NN-controlled feedback system and a reference closed-loop model. Since direct analysis of the uncertain interconnected system with an NN controller is generally intractable due to nonlinearities and uncertainty interactions, the problem is reformulated in terms of the dynamical tracking error between the two closed-loop systems. A less conservative characterization of the neural controller error is obtained using the Differential Mean Value (DMV) theorem and linear approximation techniques combined with IQCs, which relaxes the nonlinear NN mapping while preserving rigorous bounds. This reformulation eliminates the explicit nonlinear dependence on the NN controller and enables tractable analysis of the deviation dynamics. Building on this representation, an extended error dynamical system is constructed to capture both system uncertainties and NN-induced nonlinearities within a unified analytical framework. Using dissipation-based arguments, we derive conditions that bound the worst-case $\mathcal{L}_2 \to \mathcal{L}_2$ and $\mathcal{L}_2 \to \mathcal{L}_\infty$ induced gains of the deviation system. These bounds correspond to the Relative Integral Square Error (RISE) and the Supreme Square Error (SSE), respectively, and provide explicit worst-case performance guarantees. By integrating Lyapunov theory with IQC-based analysis, the proposed framework yields certified upper bounds on the deviation between the reference closed-loop model and the NN-controlled uncertain system, thereby enabling rigorous safety and robustness verification.

Therefore, analyzing the deviation between the reference closed-loop model and the NN-controlled system offers several important advantages. First, the use of the DMV theorem enables elimination of the explicit NN nonlinearity, resulting in a more tractable analytical formulation. Second, unlike approaches that rely on restrictive assumptions such as trivial initial conditions [28], the keep-close framework naturally accommodates arbitrary initial conditions. This is achieved through the deviation-based formulation, which ensures that the analysis remains valid under realistic operating conditions. Third, in contrast to IQC-based stability verification methods that focus on estimating regions of attraction [2], the keep-close approach provides explicit worst-case performance bounds for general tracking behavior. From a safety verification perspective, this enables quantification of the maximum deviation from desired behavior during transient operation, rather than only certifying asymptotic stability.

The effectiveness of the proposed keep-close framework is demonstrated through two complementary validation scenarios of increasing complexity.

1. The first case study considers a Single-Link Robot Arm control problem with a single output and slope-restricted nonlinearities. This benchmark case provides a controlled setting in which the methodology can be clearly illustrated and the theoretical guarantees validated against well-understood system dynamics.
2. The second case study addresses the Deep Guidance and Control of an Apollo Lander during atmospheric descent [29], a multi-output system characterized by strong nonlinearities, coupled dynamics, and significant environmental uncertainties arising from aerodynamic effects. In this case, aerodynamic forces are derived from high-fidelity atmospheric data obtained from the Mars Climate Database, providing a realistic and demanding validation environment.

Together, these case studies demonstrate the applicability and generality of the keep-close framework across a wide range of systems of varying complexity. They highlight its ability to provide meaningful and computable robustness and safety certificates for NN-controlled systems operating under uncertainty, including both benchmark control problems and aerospace guidance and control scenarios.

The remainder of this article is organized as follows. Section 2 presents the notation and preliminaries. Section 3 formulates the problem and introduces the key idea of the keep-close approach. Section 4 analyses the neural controller approximation error. Section 5 develops the robustness analysis and safety verification framework. Section 6 provides validation and numerical simulations, and Section 7 concludes the article.

2 | Notation and Preliminaries

2.1 | Mathematical Notation

This subsection summarizes the notation and basic definitions used throughout the article. All signals and matrices are assumed to have compatible dimensions unless stated otherwise.

- $\mathbb{R}$, $\mathbb{R}_+$, and $\mathbb{C}$ refer to the set of real, non-negative real, and complex numbers, respectively. Also, in the complex plane, the notation $\mathbb{RH}$ denotes all analytic functions within the closed exterior of the unit disk. M^T and M^* are the transpose and complex conjugate transpose of the matrix M, respectively.
- Consider the LPV system Σ where its matrices of state space depend on a time-varying parameter vector $s(t) : \mathbb{R}_+ \to \mathcal{S}$ and $\mathcal{S} \subset \mathbb{R}^n$. $s(t)$ is a continuously differentiable function of time, where the parameter rates of variation $\dot{s}(t) : \mathbb{R}_+ \to \dot{\mathcal{S}}$. $\dot{\mathcal{S}}$ is a hyperrectangle given by:

$$\dot{\mathcal{S}} = \left\{ \rho \in \mathbb{R}^n \mid \underline{\rho}_i \le \rho_i \le \overline{\rho}_i,\ i = 1 \cdots n \right\} \tag{1}$$

  Let a parameter-dependent matrix P be a continuously differentiable function of the parameter s, given by: $P : \mathcal{S} \to \mathbb{S}^n$, where $\mathbb{S}^n$ refers to the set of $n \times n$ symmetric matrices. The differential operator $\partial P : \mathcal{S} \times \dot{\mathcal{S}} \to \mathbb{S}^n$ is given by:

$$\partial P(s, \dot{s}) = \sum_{k=1}^{n} \frac{\partial P(s)}{\partial s_k} \dot{s}_k \tag{2}$$

- The space $\mathcal{L}_2([0, \infty))$ denotes the set of functions $\varphi : \mathbb{R}_+ \to \mathbb{R}^n$ satisfying

$$||\varphi||_2 = \left[\int_0^{+\infty} \varphi(\tau)^T \varphi(\tau)\, d\tau \right]^{\frac{1}{2}} < \infty \tag{3}$$

- The space $\mathcal{L}_\infty$ refers to the set of functions $\varphi : \mathbb{R}_+ \to \mathbb{R}^n$ satisfying

$$||\varphi||_\infty = \sup_{t \in [0,\infty)} \left[\varphi(t)^T \varphi(t) \right]^{\frac{1}{2}} < \infty \tag{4}$$

- We recall here the Differential Mean Value (DMV) Theorem:

Lemma 1 (DMV Theorem [30, 31]). *Consider a function $g : \Omega \to \mathbb{R}^n$ that is differentiable, where Ω is an open subset of $\mathbb{R}^n$. Given two points a and b in Ω, such that the open line segment $]a, b[$ lies entirely within Ω, there exists at least one point c on this segment for which*

$$g(b) - g(a) = \langle \nabla_x g(c), b - a \rangle, \tag{5}$$

where c lies on the curve $\varrho(\nu) = a + \nu(b - a)$ for some $\nu \in [0, 1]$ called the convex domain $\mathbb{C} \rtimes (a, b)$. Here, $\nabla_x g(c)$ denotes the gradient of g at c, defined as

$$\nabla_x g(c) = \left(\frac{\partial g}{\partial x_1}(c), \frac{\partial g}{\partial x_2}(c), \ldots, \frac{\partial g}{\partial x_n}(c) \right), \tag{6}$$

representing the vector of partial derivatives of g with respect to each component of the input vector at the point c.

Lemma 2 (Cauchy–Schwarz Inequality). *Let $\phi_1(t)$ and $\phi_2(t)$ be two square-integrable functions defined on the domain $[0, \infty)$. Then, the following inequality holds:*

$$\left| \int_0^\infty \phi_1(t)\phi_2(t)\, dt \right| \le \sqrt{\int_0^\infty |\phi_1(t)|^2\, dt} \cdot \sqrt{\int_0^\infty |\phi_2(t)|^2\, dt}.$$

- As the aim in this work is to estimate the Integral Square Error (ISE) and worst expected Supreme Square Error (SSE), it is worth defining these metrics as follows:

Definition 1 (RISE and SSE). Let us consider that y and $\hat{y}$ are the outputs of the closed-loop interconnected-based NN system and the closed-loop reference model, respectively. Then the Relative Integral Square Error (RISE) and the Supreme Square Error (SSE) are given by:

$$RISE \triangleq \frac{||y - \hat{y}||_2}{\sqrt{||d||_2^2 + ||\hat{x}||_2^2}}, \qquad SSE \triangleq \frac{||y - \hat{y}||_\infty}{\sqrt{||d||_2^2 + ||\hat{x}||_2^2}} \tag{7}$$

where the worst case of the Relative Integral Square Error γ and the worst case of the Supreme Square Error σ satisfy:

$$\frac{||y - \hat{y}||_2}{\sqrt{||d||_2^2 + ||\hat{x}||_2^2}} < \gamma, \qquad \frac{||y - \hat{y}||_\infty}{\sqrt{||d||_2^2 + ||\hat{x}||_2^2}} < \sigma \tag{8}$$

2.2 | Integral Quadratic Constraints (IQCs) Concept

The key concept of IQCs is that, instead of investigating the dynamical behavior of the system that contains uncertainty Δ, we analyze the system where Δ is removed and the signals $(p(t),\ q(t))$ are utilized to enforce the following constraints:

$$\int_{-\infty}^{+\infty} \begin{bmatrix} \hat{p}(j\omega) \\ \hat{q}(j\omega) \end{bmatrix}^* \Pi(j\omega) \begin{bmatrix} \hat{p}(j\omega) \\ \hat{q}(j\omega) \end{bmatrix} d\tau \ge 0 \tag{9}$$

where $\Pi(j\omega)$ is named an "IQC multiplier" or can simply called a "multiplier", $\hat{p}(j\omega)$ and $\hat{q}(j\omega)$ are Fourier transforms

of $p(t)$ and $q(t)$, respectively. Since (9) is valid for all admissible choices of Δ, all properties that can be proved for the constrained system also hold for the original system. Therefore, the mathematical relationship between input-output quantities of system components can be described via IQCs. The IQC in Equation (9) can be described in the time domain based on Parseval's theorem [32]. Assuming that Π is a function that is uniformly bounded and rational. Then, $\Pi(j\omega)$ can be factorized as:

$$\Pi(j\omega) = \Psi(j\omega)^* \mathcal{M} \Psi(j\omega), \tag{10}$$

where $\Psi(j\omega) \in \mathbb{RH}$ and $\mathcal{M}$ represents a constant matrix [14]. $\Psi(j\omega)$ is expressed as:

$$\Psi(j\omega) := C_\Psi[j\omega I - A_\Psi][B_{\Psi_1}\ B_{\Psi_2}] + [D_{\Psi_1}\ D_{\Psi_2}] \tag{11}$$

Then, the IQC of (9) can be written as:

$$\int_0^{+\infty} r(\tau)^T \mathcal{M} r(\tau)\, d\tau \geq 0 \tag{12}$$

where $r(t)$ is the output of the following linear system:

$$\Psi := \begin{cases} \dot{\psi}(t) = A_\Psi \psi(t) + B_{\Psi_1} p(t) + B_{\Psi_2} q(t) \\ r(t) = C_\Psi \psi(t) + D_{\Psi_1} p(t) + D_{\Psi_2} q(t) \\ \psi(0) = 0 \end{cases} \tag{13}$$

The time-domain constraint (12) is generally satisfied over infinite time intervals. A hard IQC satisfies the following more restrictive conditions: If Δ is any causal and bounded operator that fulfills (9), then:

$$\int_0^{T} r(\tau)^T \mathcal{M} r(\tau)\, d\tau \geq 0 \tag{14}$$

holds for all $T \geq 0$. Conversely, the soft IQCs in the time-domain constraint do not require holding overall finite time intervals. This property is critical because the major technical and pedagogical problems in the IQCs framework occur when we utilize soft IQCs. Unfortunately, there is an ambiguity that surrounds both the terms soft IQC and hard IQC. In particular, the uniqueness of factorization of $\Pi(j\omega)$ as $\Psi^*(j\omega)\mathcal{M}\Psi(j\omega)$, is not unique. For clarity, the following definition will be adopted in the rest of the article.

Definition 2 (Hard IQC [28]). A uniformly bounded and rational function $\Pi : j\mathbb{R} \to \mathbb{C}$ admits a hard IQC factorization if there exists $\Psi(j\omega) \in \mathbb{RH}$ and matrix $\mathcal{M}$ with $\mathcal{M}_{ij} \in \mathbb{C}$, such that $\Pi(j\omega) = \Psi^*(j\omega)\mathcal{M}\Psi(j\omega)$, and any bounded, causal operator Δ which satisfies the IQC defined by $\Pi(j\omega)$ also satisfies:

$$\int_0^{T} r(\tau)^T \mathcal{M} r(\tau)\, d\tau \geq 0 \tag{15}$$

for all $T \geq 0$ and for all $p(t) \in \mathcal{L}_2([0,\infty))([0\ \infty))$, $q = \Delta(p)$. $(\Psi(j\omega), \mathcal{M})$ is a hard IQCs factorization of Π.

2.2.1 | Example: IQC Representation of Sector-Bounded Nonlinearities

Consider a memoryless time-varying nonlinearity Δ defined by

$$q(t) = \Delta(t, p(t)),$$

where $p(t) \in \mathbb{R}$. For clarity, the scalar case is presented, noting that the extension to the multi-input multi-output case is straightforward. The nonlinearity Δ is said to be sector bounded in $[\alpha, \beta]$ (see Figure 1), with $\alpha < \beta$, if

$$\begin{aligned} &\alpha p(t)^2 \leq \Delta(t, p(t))\ p(t) \leq \beta p(t)^2 \\ &\qquad \Leftrightarrow [\Delta(t, p(t)) - \alpha p(t)][\beta p(t) - \Delta(t, p(t))] \geq 0,\ \forall\ t \geq 0. \end{aligned} \tag{16}$$

The sector condition (16) can be equivalently expressed as the quadratic constraint

$$\begin{bmatrix} p(t) \\ q(t) \end{bmatrix}^\top \underbrace{\begin{bmatrix} -\alpha\beta & \frac{\alpha+\beta}{2} \\ \frac{\alpha+\beta}{2} & -1 \end{bmatrix}}_{\mathcal{M}} \begin{bmatrix} p(t) \\ q(t) \end{bmatrix} \geq 0, \quad \mathcal{M} = \begin{bmatrix} -\alpha\beta & \frac{\alpha+\beta}{2} \\ \frac{\alpha+\beta}{2} & -1 \end{bmatrix}. \tag{17}$$

where $\mathcal{M}$ is the multiplier. Since $p \in \mathcal{L}_2([0,\infty))$, integration over time yields the following hard IQC:

$$\int_0^{T} r(\tau)^\top \mathcal{M} r(\tau)\, d\tau \geq 0, \quad \forall\ T \geq 0. \tag{18}$$

This IQC compactly captures the sector-bounded behavior of Δ and is suitable for robustness and performance analysis.

In this work, the IQC multiplier in Equation (9) is introduced in a general uniformly bounded and rational form, as the proposed keep-close framework is not restricted to a specific uncertainty class. Instead, the formulation accommodates a broad spectrum of uncertainty structures, including sector- and slope-restricted nonlinearities, Lipschitz mappings, time-varying parametric perturbations, and dynamic uncertainties such as rate-bounded delays. The use of rational dynamic multipliers without poles on the extended imaginary axis enables a unified treatment of these cases within a single convex analysis framework. In practice,

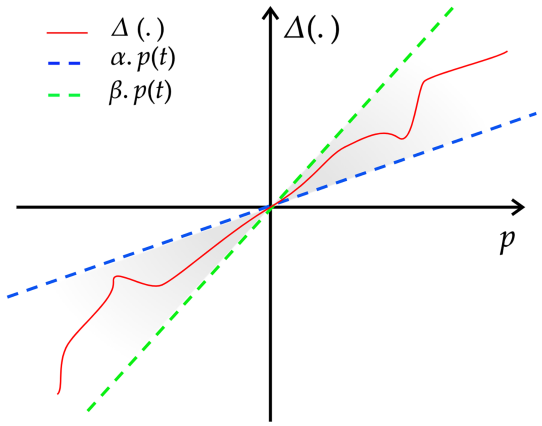

FIGURE 1 | A sector bounded nonlinearity.

the multiplier is selected according to the structural properties of the uncertainty under consideration. For example, sector- and slope-restricted nonlinearities admit Zames–Falb-type multipliers and their rational parameterizations, while time-delay and frequency-dependent uncertainties can be characterized through rational dynamic multipliers defined over finite frequency ranges. This flexibility allows the proposed approach to reduce conservatism compared to static multipliers while preserving computational tractability through LMI-based conditions, see [33–36].

3 | Problem Formulation

3.1 | Keep-Close Approach

In this subsection, the *keep-close* approach is introduced for analyzing the robustness of feedback systems employing neural network (NN) controllers in the presence of uncertainty. The main objective is to quantify how the output of a closed-loop system governed by a trained NN controller deviates from that of an ideal and trusted reference closed-loop model when subjected to bounded inputs and uncertainties. Rather than directly analyzing the uncertain NN-controlled system, the proposed approach focuses on characterizing and bounding the deviation between the outputs of the actual system and the reference model. This deviation serves as a measure of robustness and safety, allowing the estimation of the worst-case discrepancy between the two behaviors. By establishing explicit bounds on this deviation, the keep-close framework provides a rigorous mechanism for verifying whether the NN-controlled system remains sufficiently close to the desired reference behavior despite the presence of uncertainties.

The key idea is illustrated in Figure 2. Two closed-loop systems are considered. The first is a reference model, denoted by $\Sigma_P^{\pi^*}$, which is linear, controllable, and regulated by an ideal controller π^*. This reference system represents the desired closed-loop behavior and is assumed to be fully known and trusted. The second system, denoted by $\Sigma_{F(P,\Delta_\delta)}^{\pi}$, corresponds to the actual plant subject to uncertainties Δ_δ and controlled by a trained NN controller π. This system represents the real implementation for which robustness analysis and safety verification are required.

The central objective of the keep-close approach is to characterize and bound the deviation $z(t)$ between the outputs of these two closed-loop systems, and to establish an explicit relationship between this deviation and known variables of the reference model. By deriving guaranteed bounds on $z(t)$ under admissible uncertainties and bounded inputs, the proposed framework enables a quantitative assessment of whether the NN-controlled system behaves sufficiently close to the reference model to satisfy prescribed safety and performance requirements.

For the closed-loop system $\Sigma_{F(P,\Delta_\delta)}^{\pi}$, interconnected with an NN-based control system and subject to uncertainty Δ_δ as illustrated in Figure 2. This system, specifically the uncertain plant $F(P,\Delta_\delta)$ in conjunction with the NN controller π, embodies the perturbed plant $F(P,\Delta_\delta)$. It represents an interconnection

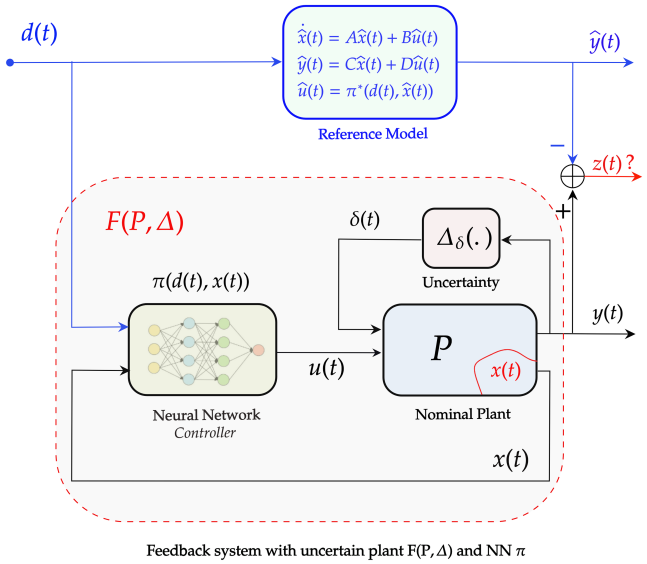

FIGURE 2 | Keep-Close Approach Illustration.

between a nominal plant P and an uncertainty Δ_δ, which can be mathematically described by the following dynamical equations:

$$\Sigma_{F(P,\Delta_\delta)}^{\pi} := \begin{cases} \dot{x}(t) = Ax(t) + Bu(t) + \tilde{B}\delta(t) \\ y(t) = Cx(t) + Du(t) + \tilde{D}\delta(t) \\ \delta(t) = \Delta_\delta(y(t)) \\ u(t) = \pi(d(t), x(t)) \\ x(0) = x_0 \end{cases} \tag{19}$$

where $x \in \mathbb{R}^n$ is the system's state vector, $d \in \mathbb{R}^d$ is the input reference signal control, inputs of the neural controller $u \in \mathbb{R}^m$ and $(A, B, C, D, \tilde{B}, \tilde{D})$ are known with adequate dimensions, and the perturbation operator Δ_δ is a bounded, causal. Such a system formulation is commonly employed in the literature and has been extensively studied; see, for instance, [2, 14, 28, 37–39].

As stated above, the objective of this study is to establish bounds on the deviation $z(t)$ as a function of the reference system variables. This deviation serves as a key indicator of how closely the closed-loop system equipped with an NN controller, denoted by $\Sigma_{F(P,\Delta_\delta)}^{\pi}$, follows a given reference model $\Sigma_P^{\pi^*}$. The latter represents a closed-loop system regulated by an ideal controller designed using classical control techniques and for which stability and performance properties are well established. The reference model is described as follows:

$$\Sigma_P^{\pi^*} := \begin{cases} \dot{\hat{x}}(t) = A\hat{x}(t) + B\hat{u}(t) \\ \hat{y}(t) = C\hat{x}(t) + D\hat{u}(t) \\ \hat{u}(t) = \pi^*(d(t), \hat{x}(t)) \\ \hat{x}(0) = x_0 \end{cases} \tag{20}$$

with the reference system's state vector $\hat{x} \in \mathbb{R}^n$, control inputs $\hat{u} \in \mathbb{R}^m$, and the system matrices (A, B, C, D) being the same as those mentioned in Equation (19). After designing the controller

$\hat{u}(t) = \pi^*(d(t), \hat{x}(t))$ for the reference system $\Sigma_P^{\pi^*}$, the system is transformed as follows:

$$\Sigma_P^{\pi^*} := \begin{cases} \dot{\hat{x}}(t) = A_r\hat{x}(t) + B_r d(t) \\ \hat{y}(t) = C_r\hat{x}(t) + D_r d(t) \\ \hat{x}(0) = x_0 \end{cases} \tag{21}$$

To accomplish our aim stated above, we present an analysis of the dynamical error system between the two closed-loop systems (i.e., the reference model and the system closed with an NN controller) in the following section. We start by mathematically describing the dynamical error system.

Using (19) and (20), let us assume that the state of the dynamic error system is defined as $\zeta(t) = x(t) - \hat{x}(t)$ and $z(t) = y(t) - \hat{y}(t)$. Consequently, the time derivative of the error state is given by $\dot{\zeta}(t) = \dot{x}(t) - \dot{\hat{x}}(t)$, which leads to:

$$\dot{\zeta}(t) = A\zeta(t) + B\mu(t) + \tilde{B}\delta(t), \tag{22a}$$

$$z(t) = C\zeta(t) + D\mu(t) + \tilde{D}\delta(t), \tag{22b}$$

$$\mu(t) = \pi(d(t), x(t)) - \pi^*(d(t), \hat{x}(t)), \tag{22c}$$

$$\zeta(0) = 0. \tag{22d}$$

where $\mu(t)$ represents the controller error (between the NN controller and the ideal controller adopted in the closed-loop reference model). Furthermore, the initial condition of the dynamical error state $\zeta(t)$ is $\zeta(0) = 0$ because $x(0) = \hat{x}(0) = x_0$.

In this work, the initial condition of the system is assumed to be known. This assumption is standard in robust control and safety verification, where the system state at the initial time is typically measured, estimated, or specified as part of the operating conditions. By initializing the reference closed-loop model with the same initial state, the deviation between the two systems is zero at the initial time, thereby eliminating artificial transients caused solely by mismatched initialization. This allows the analysis to focus exclusively on the effects of system uncertainties and neural-network-induced nonlinearities.

On the other hand, to enable the robustness analysis later within the IQC framework, we impose the following standard assumption on the uncertainty operator.

Assumption 1 (IQC Admissibility). Let $y(t) \in \mathcal{L}_2([0, \infty))$ be a signal and let the uncertainty operator $\Delta_\delta : \mathcal{L}_2 \to \mathcal{L}_2$ be bounded and causal. Assume that $\Delta_\delta(\cdot)$ satisfies the Integral Quadratic Constraint (IQC) defined by the multiplier $\Pi_\delta(j\omega)$.

Then, the input-output behavior of the uncertainty $\Delta_\delta(\cdot)$ satisfies the following time-domain IQC:

$$\int_0^T r_\delta(\tau)^T \, \mathcal{M}_\delta \, r_\delta(\tau) \, d\tau \geq 0, \quad \forall \, T \geq 0, \tag{23}$$

where $r_\delta(t)$ is the output of the IQC filter associated with the pair $(\Psi_\delta, \mathcal{M}_\delta)$, and $(\Psi_\delta(j\omega), \mathcal{M}_\delta)$ represents a hard IQC factorization of the multiplier $\Pi_\delta(j\omega)$.

4 | Neural Controller Approximation Error Analysis

4.1 | Neural Controller Approximation Error Expression

This subsection analyses the controller error $\mu(t)$ in Equation (22c) (i.e., the difference between the NN controller and the ideal controller adopted in the reference closed-loop model). The objective of this analysis is to identify a less conservative and more accurately representative expression for $\mu(t)$, from which we can extract and dissociate certain and uncertain quantities. Achieving this will allow us to directly use this refined expression in the dynamic error system as referenced in Equation (22). To reach the aforementioned objective, we start by introducing the following lemma:

Lemma 3 (NN Approximation Error Expression). *Consider the dynamics of the error as defined in Equation* (22) *and Let c the trajectory lies on the curve $\vartheta(\nu) = x + \nu(\hat{x} - x)$ for some $\nu \in [0, 1]$ called the convex domain $\mathbb{C}\bowtie(x, \hat{x})$. The controllers error $\mu(t)$, as given in Equation (22c), can be expressed as*:

$$\mu(t) = \Lambda(s)\zeta(t) + \frac{\partial\alpha}{\partial\hat{x}}(0,0)\hat{x}(t) + \frac{\partial\alpha}{\partial d}(0,0)d(t) + \epsilon(t), \tag{24}$$

where the trajecory $s(t) = [d(t), c(t)]^T$, the LPV matrix $\Lambda(s(t)) = \nabla_x\pi(d(t), c(t))$ is the Jacobian matrix of the neural network, $\epsilon(t) = \mathcal{O}(d(t), \hat{x}(t))$ and the training error $\alpha(d(t), \hat{x}(t)) = \pi(d(t), \hat{x}(t)) - \pi^*(d(t), \hat{x}(t))$.

Proof. From Equation (22c), the controller error $\mu(t)$ is given by:

$$\mu(t) = \pi(d(t), x(t)) - \pi^*(d(t), \hat{x}(t)) \tag{25}$$

Adding and subtracting $\pi(d(t), \hat{x}(t))$, the error approximation $\mu(t)$ can be expressed by:

$$\mu(t) = \underbrace{\pi(d(t), \hat{x}(t)) - \pi^*(d(t), \hat{x}(t))}_{=\alpha(d(t),\hat{x}(t))} + \underbrace{\pi(d(t), x(t)) - \pi(d(t), \hat{x}(t))}_{=\beta(t)} = \alpha(d(t), \hat{x}(t)) + \beta(t) \tag{26}$$

It is clear that the function $\alpha(d(t), \hat{x}(t))$ in Equation (26) is directly related to the neural network controller training error. Indeed, the primary objective of the training procedure is to minimize this quantity so that the neural controller accurately reproduces the behavior of the ideal control law for input $\hat{x}(t)$. The function $\alpha(d(t), \hat{x}(t))$ is defined as follows:

$$\alpha(d(t), \hat{x}(t)) = \pi(d(t), \hat{x}(t)) - \pi^*(d(t), \hat{x}(t)) \tag{27}$$

Knowing that $d(t), \hat{x}(t) \in \mathcal{L}_2([0, \infty))$ because it is the state of the reference model controlled by the ideal controller $\pi^*(d(t), \hat{x}(t))$. Therefore, by using Taylor approximation, we can write the approximation of $\alpha(d(t), \hat{x}(t))$ as follows

$$\alpha(d(t), \hat{x}(t)) = \alpha(0,0) + \frac{\partial\alpha}{\partial\hat{x}}(0,0)\hat{x}(t) + \frac{\partial\alpha}{\partial r}(0,0)d(t) + \underbrace{\mathcal{O}(d(t), \hat{x}(t))}_{=\epsilon(t)} \tag{28}$$

Clearly, if $\hat{x}(t) = 0$ and $d(t) = 0$ there is no controller action, that is, $\alpha(0, 0) = 0$, therefore

$$\alpha(d(t), \hat{x}(t)) = \frac{\partial \alpha}{\partial \hat{x}}(0,0)\hat{x}(t) + \frac{\partial \alpha}{\partial r}(0,0)d(t) + \epsilon(t) \tag{29}$$

On the other hand, using the Differential Mean Value Theorem in Lemma 1 and the fact that the function π is continuous and differentiable on convex hull of the set $\mathbb{C}\rtimes(x(t), \hat{x}(t))$, then, given two points $x(t)$ and $\hat{x}(t)$ in Ω at time t, such that the open line segment between $x, \hat{x}$ lies entirely within Ω, there exists at least one point $c(t)$ on this segment for which

$$\begin{aligned}\beta(t) &= \pi(d(t), x(t)) - \pi(d(t), \hat{x}(t)) \\ &= \langle \nabla_x \pi(d(t), c(t)), x(t) - \hat{x}(t) \rangle = \langle \nabla_x \pi(d(t), c(t)), \zeta(t) \rangle, \end{aligned} \tag{30}$$

where c lies on the curve $\varrho(\nu) = x + \nu(\hat{x} - x)$ for some $\nu \in [0, 1]$ called the convex domain $\mathbb{C}\rtimes(x, \hat{x})$. and

$$\begin{aligned}&\nabla_x \pi(d(t), c(t)) \\ &= \left(\frac{\partial \pi}{\partial x_1}(d(t), c(t)), \frac{\partial \pi}{\partial x_2}(d(t), c(t)), \ldots, \frac{\partial \pi}{\partial x_n}(d(t), c(t)) \right), \end{aligned} \tag{31}$$

The equations (29) and (30) allow us to conclude that:

$$\begin{aligned}\mu(t) = &\underbrace{\langle \nabla_x \pi(d(t), c(t)), \zeta(t) \rangle}_{\Lambda(s)\zeta(t)} \\ &+ \frac{\partial \alpha}{\partial \hat{x}}(0,0)\hat{x}(t) + \frac{\partial \alpha}{\partial r}(0,0)d(t) + \epsilon(t), \end{aligned} \tag{32}$$

where the trajectory $s(t) = [d(t), c(t)]^T$, the LPV matrix $\Lambda(s(t)) = \nabla_x \pi(d(t), c(t))$ is the Jacobian matrix of the neural network, $\epsilon(t) = \mathcal{O}(d(t), \hat{x}(t))$ and the training error $\alpha(d(t), \hat{x}(t)) = \pi(d(t), \hat{x}(t)) - \pi^*(d(t), \hat{x}(t))$. □

The result in Equation (32) provides a less conservative and more representative expression for $\mu(t)$. The specific quantity $\Lambda(s(t))\zeta(t)$ is influenced by the Jacobian of the neural controller $\nabla_x \pi(d(t), c(t))$, which incorporates the weights of the neural controller, and it is also dependent on the state of the dynamic error system $\zeta(t)$. For uncertain components, the term $\epsilon(t)$ is determined by the reference state $\hat{x}(t)$ and the reference signal $d(t)$, both of which belong to the $\mathcal{L}_2([0, \infty))$ space. In the next subsection, we will prove that $\epsilon(t)$ also belongs to $\mathcal{L}_2([0, \infty))$ and it is a bounded, causal operator, leveraging this property.

4.2 | NN Approximation Error Admissibility

The perturbations acting on the feedback system can encompass various types of uncertainties, including unmodeled dynamics, saturation effects, time delays, and slope-restricted nonlinearities. In the next lemma, we will demonstrate that if $d(t)$ and $\hat{x}(t)$ belong to $\mathcal{L}_2([0, \infty))$, then $\epsilon(t) = \mathcal{O}(d(t), \hat{x}(t)) = \Delta_\epsilon(\eta(t))$ where $\eta(t) = [d(t), \hat{x}(t)]^T$ also belongs to $\mathcal{L}_2([0, \infty))$ and it is a bounded and causal operator. Establishing this result will enable us to apply the IQCs technique to address the uncertainty $\epsilon(t)$ in the subsequent analysis.

Lemma 4 (NN Approximation Error Admissibility). *Let $\eta(t) = [d(t), \hat{x}(t)]^T \in \mathcal{L}_2([0, \infty))$ and consider the operator $\Delta_\epsilon : \mathcal{L}_2 \to \mathcal{L}_2$ defined by*

$$\epsilon(t) = \Delta_\epsilon(\eta(t)) = \mathcal{O}(d(t), \hat{x}(t)).$$

Then the following properties hold:

1. *The signal $\epsilon(t)$ belongs to $\mathcal{L}_2([0, \infty))$, and the operator $\Delta_\epsilon(\cdot)$ is bounded and causal.*
2. *The operator $\Delta_\epsilon(\cdot)$ admits an Integral Quadratic Constraint (IQC) characterization defined by the multiplier $\Pi_\epsilon(j\omega)$. In particular, there exists a hard IQC factorization $(\Psi_\epsilon, \mathcal{M}_\epsilon)$ such that the signal*

$$r_\epsilon(t) = \Psi_\epsilon \begin{bmatrix} \eta(t) \\ \epsilon(t) \end{bmatrix}$$

satisfies the time-domain IQC

$$\int_0^T r_\epsilon(\tau)^T \mathcal{M}_\epsilon \, r_\epsilon(\tau) \, d\tau \geq 0, \quad \forall T \geq 0. \tag{33}$$

Proof. By assumption, $d(t)$ and $\hat{x}(t)$ belong to the space $\mathcal{L}_2([0, \infty))$. Additionally, we have $\epsilon(t) = \mathcal{O}(d(t), \hat{x}(t))$, which implies that there exist positive constants $\kappa_1 \in]0, 1[$ and $\kappa_2 \in]0, 1[$ such that:

$$|\epsilon(t)| = |\mathcal{O}(d(t), \hat{x}(t))| \leq \kappa_1 |d(t)| + \kappa_2 |\hat{x}(t)|. \tag{34}$$

This implies:

$$\int_0^{+\infty} |\epsilon(t)|^2 \, \mathrm{d}t \leq \int_0^{+\infty} \left(\kappa_1 |d(t)| + \kappa_2 |\hat{x}(t)| \right)^2 \mathrm{d}t. \tag{35}$$

Expanding the squared term and applying the triangle inequality under the integral gives:

$$\int_0^{+\infty} |\epsilon(t)|^2 \, \mathrm{d}t \leq \int_0^{+\infty} \left(\kappa_1^2 |d(t)|^2 + \kappa_2^2 |\hat{x}(t)|^2 + 2\kappa_1 \kappa_2 |d(t)||\hat{x}(t)| \right) \mathrm{d}t. \tag{36}$$

Since $d(t), \hat{x}(t) \in \mathcal{L}_2([0, \infty))$, their respective norms are finite:

$$\|d(t)\|_{\mathcal{L}_2}^2 = \int_0^{+\infty} |d(t)|^2 \, \mathrm{d}t < \infty, \quad \|\hat{x}(t)\|_{\mathcal{L}_2}^2 = \int_0^{+\infty} |\hat{x}(t)|^2 \, \mathrm{d}t < \infty. \tag{37}$$

The cross-term $2\kappa_1 \kappa_2 |d(t)||\hat{x}(t)|$ is also integrable because of the Cauchy–Schwarz inequality:

$$\int_0^{+\infty} |d(t)||\hat{x}(t)| \, \mathrm{d}t \leq \|d(t)\|_{\mathcal{L}_2} \|\hat{x}(t)\|_{\mathcal{L}_2}. \tag{38}$$

Thus, $\epsilon(t) \in \mathcal{L}_2([0,\infty))$ and it is a bounded operator. Also, it is clear that the output of $\epsilon(t)$ at any time t depends only on the input $(d(t), \hat{x}(t))$ up to that time t, that is, $\epsilon(t)$ is a causal operator. This completes the proof, and then the IQCs technique can be applied to address the uncertainty $\epsilon(t) = \Delta_\epsilon(\eta(t))$.

On the other hand, The equation (34) implies the existence of a constant $\kappa > 0$ such that

$$|\epsilon(t)| \le \kappa \|\eta(t)\|. \tag{39}$$

Squaring both sides yields

$$\epsilon^T(t)\epsilon(t) \le \kappa^2 \eta^T(t)\eta(t). \tag{40}$$

Rearranging gives

$$\begin{bmatrix}\eta(t)\\ \epsilon(t)\end{bmatrix}^T \begin{bmatrix}\kappa^2 I & 0\\ 0 & -I\end{bmatrix} \begin{bmatrix}\eta(t)\\ \epsilon(t)\end{bmatrix} \ge 0. \tag{41}$$

Integrating over $[0, T]$ yields

$$\int_0^T \begin{bmatrix}\eta(\tau)\\ \epsilon(\tau)\end{bmatrix}^T \mathcal{M}_\epsilon \begin{bmatrix}\eta(\tau)\\ \epsilon(\tau)\end{bmatrix} d\tau \ge 0, \tag{42}$$

where

$$\mathcal{M}_\epsilon = \begin{bmatrix}\kappa^2 I & 0\\ 0 & -I\end{bmatrix}.$$

This proves that the operator $\Delta_\epsilon(\cdot)$ satisfies IQC. □

5 | Robustness Analysis and Safety Verification

5.1 | IQC Filter Realization

The following corollary establishes a state-space realization of the IQC multipliers associated with the uncertainty operators Δ_δ and Δ_ϵ through virtual LTI filters. This formulation allows the IQC constraints to be represented dynamically and incorporated into an augmented system description. In the next subsection, this result will be used to construct the extended dynamical system, which forms the basis for deriving Lyapunov-based robustness and safety certificates within the keep-close framework.

Corollary 1 (IQC Filter Realization). *Under Assumption 1 and Lemma 4, the signals $r_\delta(t)$ and $r_\epsilon(t)$ satisfying the IQC conditions (23) and (33), respectively, can be generated as the outputs of linear time-invariant (LTI) systems corresponding to the hard IQC factorizations$(\Psi_\delta, \mathcal{M}_\delta)$ and $(\Psi_\epsilon, \mathcal{M}_\epsilon)$, respectively.*

Specifically, the virtual filter associated with Δ_δ admits the state-space realization

$$\begin{cases}\dot{\psi}_\delta(t) = A_\delta \psi_\delta(t) + B_{\delta 1}\delta(t) + B_{\delta 2} y(t),\\ r_\delta(t) = C_\delta \psi_\delta(t) + D_{\delta 1}\delta(t) + D_{\delta 2} y(t),\\ \psi_\delta(0) = 0,\end{cases} \tag{43}$$

and similarly, the virtual filter associated with Δ_ϵ is given by

$$\begin{cases}\dot{\psi}_\epsilon(t) = A_\epsilon \psi_\epsilon(t) + B_{\epsilon 1}\epsilon(t) + B_{\epsilon 2}\eta(t),\\ r_\epsilon(t) = C_\epsilon \psi_\epsilon(t) + D_{\epsilon 1}\epsilon(t) + D_{\epsilon 2}\eta(t),\\ \psi_\epsilon(0) = 0.\end{cases} \tag{44}$$

The operators Ψ_δ and Ψ_ϵ represent the virtual IQC filters associated with the uncertainty operators Δ_δ and Δ_ϵ, respectively. These filters process the inputs and outputs of the uncertainty operators and generate signals that satisfy the corresponding quadratic constraints.

For compact representation, the individual IQC filters (43) and (44) can be combined into a single extended filter of the form

$$\Psi := \begin{cases}\dot{\xi}(t) = A_\xi \xi(t) + B_{\xi 1} q(t) + B_{\xi 2} p(t),\\ r(t) = C_\xi \xi(t) + D_{\xi 1} q(t) + D_{\xi 2} p(t),\\ \xi(0) = 0,\end{cases} \tag{45}$$

where the augmented state, input, and output vectors are defined as

$$\xi(t) = \begin{bmatrix}\psi_\delta(t)\\ \psi_\epsilon(t)\end{bmatrix}, \quad p(t) = \begin{bmatrix}y(t)\\ \eta(t)\end{bmatrix}, \quad q(t) = \begin{bmatrix}\delta(t)\\ \epsilon(t)\end{bmatrix}.$$

The corresponding state-space matrices of the extended filter are given by the block-diagonal structure

$$A_\xi = \begin{bmatrix}A_\delta & 0\\ 0 & A_\epsilon\end{bmatrix}, \quad B_{\xi 1} = \begin{bmatrix}B_{\delta 1} & 0\\ 0 & B_{\epsilon 1}\end{bmatrix}, \quad B_{\xi 2} = \begin{bmatrix}B_{\delta 2} & 0\\ 0 & B_{\epsilon 2}\end{bmatrix},$$

$$C_\xi = \begin{bmatrix}C_\delta & 0\\ 0 & C_\epsilon\end{bmatrix}, \quad D_{\xi 1} = \begin{bmatrix}D_{\delta 1} & 0\\ 0 & D_{\epsilon 1}\end{bmatrix}, \quad D_{\xi 2} = \begin{bmatrix}D_{\delta 2} & 0\\ 0 & D_{\epsilon 2}\end{bmatrix}.$$

5.2 | Extended System Construction

Considering the error dynamics system in Equation (22), the conservative expression of μ presented in Equation (24) and Corollary 1, let us define $\chi(t)$ as the extended state vector, given by

$$\chi(t) = \begin{bmatrix}\zeta(t)\\ \xi(t)\end{bmatrix},$$

where $\zeta(t)$ represents the system state error and $\xi(t)$ encapsulates additional dynamics given in Equation (45). The dynamics of $\chi(t)$, omitting the dependency on s for brevity, are governed by the following equations:

$$\Sigma_{ex} \begin{cases}\dot{\chi}(t) &= \mathcal{A}\chi(t) + \mathcal{B}_1 q(t) + \mathcal{B}_2 \eta(t)\\ r(t) &= \mathcal{C}_1 \chi(t) + \mathcal{D}_{11} q(t) + \mathcal{D}_{12}\eta(t)\\ z(t) &= \mathcal{C}_2 \chi(t) + \mathcal{D}_{21} q(t) + \mathcal{D}_{22}\eta(t)\\ \eta(t) &= \left[d(t)\ \ \hat{x}(t)\right]^T,\ \chi(0) = 0\end{cases} \tag{46}$$

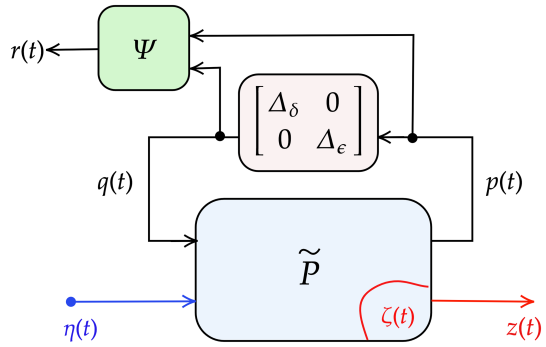

FIGURE 3 | Graphical Interpretation of the IQCs with the Equivalent System of (46).

where the matrices of the Linear Parameter Varying (LPV) extended system (46) have the appropriate dimensions.

$$\mathcal{A}(s) = \begin{bmatrix} A + B\Lambda(s) & 0 & 0 \\ B_{\delta 2}(C + D\Lambda(s)) & A_\delta & 0 \\ 0 & 0 & A_\epsilon \end{bmatrix}$$

$$\mathcal{B}_1(s) = \begin{bmatrix} \tilde{B} & B \\ B_{\delta 2}\tilde{D} & B_{\delta 2}D \\ D_{\delta 2}\tilde{D} & D_{\delta 2}D \end{bmatrix},$$

$$\mathcal{B}_2 = \begin{bmatrix} \left[B\frac{\partial \alpha}{\partial d}(0,0) \;\; B\frac{\partial \alpha}{\partial \hat{x}}(0,0)\right] \\ \left[B_{\delta 2}D\frac{\partial \alpha}{\partial d}(0,0) + B_{\delta 2}D_r \;\; B_{\delta 2}D\frac{\partial \alpha}{\partial \hat{x}}(0,0) + B_{\delta 2}C_r\right] \\ B_{\epsilon 2} \end{bmatrix}$$

$$\mathcal{C}_1(s) = \begin{bmatrix} D_{\delta 2}(C + D\Lambda(s)) \\ C_\delta \\ C_\epsilon \end{bmatrix}, \; \mathcal{C}_2(s) = \begin{bmatrix} C + D\Lambda(s) \\ 0 \\ 0 \end{bmatrix}$$

$$\mathcal{D}_{11} = \begin{bmatrix} D_{\delta 2}\tilde{D} + D_{\delta 1} & D_{\delta 2}D \\ 0 & D_{\epsilon 1} \end{bmatrix},$$

$$\mathcal{D}_{12} = \begin{bmatrix} \left[D_{\delta 2}D\frac{\partial \alpha}{\partial d}(0,0) + D_{\delta 2}D_r \;\; D_{\delta 2}D\frac{\partial \alpha}{\partial \hat{x}}(0,0) + D_{\delta 2}C_r\right] \\ D_{\epsilon 2} \end{bmatrix}$$

$$\mathcal{D}_{21} = \begin{bmatrix} \tilde{D} & D \end{bmatrix}, \; \mathcal{D}_{22}(s) = \begin{bmatrix} D\frac{\partial \alpha}{\partial d}(0,0) & D\frac{\partial \alpha}{\partial \hat{x}}(0,0) \end{bmatrix} \tag{47}$$

This extended system is constructed by combining the dynamics of the deviation system, derived in Appendix A, with the neural network approximation error representation $\mu(t)$ given in Equation (24), and the extended virtual IQC filter defined in Equation (45). Following the formulation in Reference [28], the resulting augmented system, described in Equation (46), is illustrated in Figure 3. This representation captures the interconnected structure of the deviation dynamics, the uncertainty operators Δ_δ and Δ_ϵ, and the associated IQC filters. In particular, the extended system takes the signal $\eta(t)$, which characterizes the reference model behavior, as its input and produces the deviation signal $z(t)$ as its output, thereby explicitly describing how uncertainties and neural network approximation errors influence the deviation from the reference closed-loop model.

Based on this extended representation, a dissipation inequality can be established to characterize and upper bound the worst-case induced gain of the interconnected operator $\tilde{F}(\tilde{P}, \Delta_\delta, \Delta_\epsilon)$, as depicted in Figure 3. This is achieved by combining the state-space representation of the extended system in Equation (46) with the time-domain IQCs defined in Equations (23) and (33).

5.3 | Relative Integral Square Error (RISE)

The following theorem delineates the conditions necessary to evaluate the worst-case performance of the Relative Integral of the Square of the Error (RISE) metric. In control theory, the RISE metric is instrumental in assessing the efficacy of a system's response [40]. Specifically, RISE offers critical insights into the comparative behavior of an uncertain closed-loop system, equipped with a neural network (NN) controller, against a predefined closed-loop reference model. This comparison is pivotal for understanding the resilience and robustness of the NN-controlled system in the face of uncertainties.

Theorem 1 (Worst-case RISE Performance). *Let Δ_δ and Δ_ϵ satisfy the Integral Quadratic Constraint (IQC) given by $IQC(\Xi, M)$, and assume that the extended system Σ_{ex} as defined in Equation (46) is well-posed. Suppose there exists a scalar $\lambda > 0$ and a continuously differentiable matrix $P = P^T$ with appropriate dimensions such that $P \geq 0$, and for all $(s, \dot{s}) \in \Omega \times \dot{S}$ (omitting s and $\dot{s}$ for brevity)*:

$$\begin{bmatrix} P\mathcal{A} + \mathcal{A}^T P + \partial P & P\mathcal{B}_1 & P\mathcal{B}_2 \\ \mathcal{B}_1^T P & 0 & 0 \\ \mathcal{B}_2^T P & 0 & -I \end{bmatrix} + \frac{1}{\gamma^2}\begin{bmatrix} \mathcal{C}_2^T \\ \mathcal{D}_{21}^T \\ \mathcal{D}_{22}^T \end{bmatrix}$$

$$\begin{bmatrix} \mathcal{C}_2 & \mathcal{D}_{21} & \mathcal{D}_{22} \end{bmatrix} + \lambda \begin{bmatrix} \mathcal{C}_1^T \\ \mathcal{D}_{11}^T \\ \mathcal{D}_{12}^T \end{bmatrix} M \begin{bmatrix} \mathcal{C}_1 & \mathcal{D}_{11} & \mathcal{D}_{12} \end{bmatrix} < 0 \tag{48}$$

Then, the Relative Integral Square Error (RISE) satisfies:

$$\sup_{s \in \Omega, 0 \neq \eta \in \mathcal{L}_2, \zeta(0)=0} \left[\frac{||y - \hat{y}||_2}{\sqrt{||d||_2^2 + ||\hat{x}||_2^2}} \right] < \gamma, \tag{49}$$

Proof. Consider the various signals (χ, z, p, η, r) of the error dynamics system, with the input $\eta \in \mathcal{L}_2([0, \infty))$ and the solution trajectory χ subject to zero initial conditions $\chi(0) = 0$. Assuming the extended system, as referred to in Equation (46), is well-posed implies that all signals within this system are well-defined. By assumption, the uncertainties Δ_δ and Δ_ϵ are constrained by the Integral Quadratic Constraint (IQC) defined by (Ξ, M), and consequently, the quantity r must satisfy the time-domain IQC as given in Equation (23) for any $T > 0$. Furthermore, utilizing the state-space representation of the virtual filter detailed in Equation (13), the expression for $r^T M r$ is formulated as follows:

$$rMr^T = \begin{bmatrix} \xi \\ p \\ q \end{bmatrix}^T \begin{bmatrix} C_\xi^T M C_\xi & C_\xi^T M D_\xi \\ D_\xi^T M C_\xi & D_\xi^T M D_\xi \end{bmatrix} \begin{bmatrix} \xi \\ p \\ q \end{bmatrix} \tag{50}$$

The foundation of the following proof lies in defining a parameter-dependent storage function, also known as a Lyapunov function, $V : \mathbb{R}^{\chi} \times \Omega \to \mathbb{R}_+$, given by $V(\chi, s) = \chi^T P(s)\chi$. The strict inequality presented in Equation (48) suggests that there exists a very small quantity $\varepsilon > 0$ such that the following perturbed matrix inequality is satisfied for all $s \in \Omega$:

$$\begin{bmatrix} P\mathcal{A}^T + \mathcal{A}P^T + \partial P & P\mathcal{B}_1 & P\mathcal{B}_2 \\ \mathcal{B}_1^T P & 0 & 0 \\ \mathcal{B}_2^T P & 0 & -(1-\varepsilon)I \end{bmatrix} + \frac{1}{\gamma^2}\begin{bmatrix} \mathcal{C}_2^T \\ \mathcal{D}_{21}^T \\ \mathcal{D}_{22}^T \end{bmatrix}\begin{bmatrix} \mathcal{C}_2 & \mathcal{D}_{21} & \mathcal{D}_{22} \end{bmatrix} + \lambda \begin{bmatrix} \mathcal{C}_1^T \\ \mathcal{D}_{11}^T \\ \mathcal{D}_{12}^T \end{bmatrix} M \begin{bmatrix} \mathcal{C}_1 & \mathcal{D}_{11} & \mathcal{D}_{12} \end{bmatrix} \leq 0 \tag{51}$$

Taking into account the structural condition given in Equation (50) and multiplying both sides of equation (51) from the left and right by $[\chi^T, q^T, \eta^T]$ and its transpose $[\chi^T, q^T, \eta^T]^T$ respectively (see Appendix B for instance), we observe that the Lyapunov function V satisfies the following dissipation inequality:

$$\nabla_{\chi} V \dot{\chi} + \nabla_s V \dot{s} \leq -\frac{1}{\gamma^2} z^T z + (1-\varepsilon)\eta^T \eta - \lambda r M r^T \tag{52}$$

This analysis demonstrates that the dissipation inequality presented in Equation (52) corresponds directly to the perturbed linear matrix inequality delineated in Equation (48). By integrating the dissipation inequality obtained from Equation (52) over the interval from the lower bound time $t = 0$ to the upper bound time $t = T$, and considering the initial condition $\chi(0) = 0$, that is, $V(0) = 0$, we obtain:

$$\underbrace{V(\chi(T), s(T)) - V(0,0)}_{=V(\chi(T),s(T))\geq 0 \;\because\; V(0,0)=0} + \lambda \int_0^T r(\tau)^T \mathcal{M} r(\tau)\, d\tau \leq (1-\varepsilon) \times \int_0^T \eta(\tau)\eta(\tau)^T\, d\tau - \frac{1}{\gamma^2}\int_0^T z(\tau) z(\tau)^T\, d\tau \tag{53}$$

It follows from the positivity condition of IQCs as in Equation (23), $\lambda \geq 0$, and also due to the non-negativity of the storage function V, which leads to:

$$\frac{1}{\gamma^2}\int_0^T z(\tau) z(\tau)^T\, d\tau \leq (1-\varepsilon)\int_0^T \eta(\tau)\eta(\tau)^T\, d\tau \tag{54}$$

As we assume that $\varepsilon > 0$, thus

$$\lim_{T\to+\infty}\left[\frac{1}{\gamma^2}\int_0^T z(\tau) z(\tau)^T\, d\tau < \int_0^T \eta(\tau)\eta(\tau)^T\, d\tau\right] \tag{55}$$

Knowing that

$$||z(t)||_2 = ||y(t) - \hat{y}(t)||_2 \quad ||\eta(t)||_2 = \sqrt{||d(t)||_2^2 + ||\hat{x}(t)||_2^2} \tag{56}$$

this allows us to deduce that:

$$\sup_{s\in\Omega.\ 0\neq y,\hat{y}\in\mathcal{L}_2,\zeta(0)=0} \frac{||y-\hat{y}||_2}{\sqrt{||d||_2^2 + ||\hat{x}||_2^2}} < \gamma \tag{57}$$

□

The value γ specified in Theorem 1 represents the maximal $\mathcal{L}_2$ induced gain of the deviation system over all admissible uncertainty trajectories. In particular, it provides an explicit upper bound on the worst-case Relative Integral of the Square of the Error (RISE) between the closed-loop reference model and the NN-controlled system. This worst-case bound accounts for the neural network parameters Λ and α, as well as the associated approximation error ϵ, and incorporates the effects of system uncertainties Δ_δ and Δ_ϵ within the IQC framework. This result is significant because it enables prediction of the behavior of the uncertain NN-controlled system using only the known reference signals $d(t)$ and $\hat{x}(t)$. Specifically, the bound γ characterizes the region within which the system output is guaranteed to remain relative to the reference closed-loop model, despite the presence of uncertainties and neural network training errors. Consequently, the theorem provides a rigorous robustness and safety certificate by quantifying the maximum deviation from the desired reference behavior.

From a practical perspective, this worst-case bound serves as a quantitative performance and safety indicator. A smaller value of γ implies that the NN-controlled system more closely reproduces the desired reference behavior, indicating higher robustness and reliability. Conversely, larger values of γ reveal increased sensitivity to uncertainties and neural network approximation errors. Therefore, the proposed framework not only certifies safety and robustness but also provides a systematic tool for evaluating and improving neural network controllers by identifying and minimizing the worst-case deviation from the ideal closed-loop performance. Moreover, this predictive capability is particularly important in safety-critical applications such as aerospace guidance and autonomous systems, where guaranteeing bounded deviation from a trusted reference model is essential. By explicitly quantifying worst-case performance, Theorem 1 enables rigorous verification of neural network controllers prior to deployment, thereby supporting their safe and reliable integration into uncertain dynamical systems.

5.4 | Supreme Square Error (SSE)

The Linear Parameter Varying (LPV) model approximation, employed in the robust tracking problem elaboration, invites further scrutiny under more conservative conditions. Specifically, the robust performance analysis of the LPV model for system (46) can be broadened to encompass performance metrics less conservative than the Relative Integral Square Error (RISE). The forthcoming theorem aims to establish the conditions for the worst-case Supreme Square Error (SSE), denoted by σ, under conditions that are less conservative, such as $||y - \hat{y}||_2 \geq ||y - \hat{y}||_\infty$. More granular than RISE, SSE offers profound insights into the dynamical behavior of the interconnected system featuring a neural network (NN) controller in juxtaposition with the closed-loop reference model. As delineated in Definition 1, SSE serves as a robust estimator for the maximal deviation between the outputs of the two systems. Identifying the bounds of SSE enhances the confidence level for users considering the adoption of these control schemes.

Theorem 2 (Worst-case SSE). *Let Δ_δ and Δ_ϵ satisfy the Integral Quadratic Constraint $IQC(\Xi, M)$ and assume that the*

extended system Σ_{ex} as defined in Equation (46) is well-posed. Suppose there exists a scalar $\lambda > 0$ and a continuously differentiable matrix $P = P^T$ with appropriate dimensions such that $P \geq 0$, and for all $(s, \dot{s}) \in \Omega \times \dot{\mathcal{S}}$ (omitting s and $\dot{s}$ for brevity):

$$\begin{bmatrix} P\mathcal{A} + \mathcal{A}^T P + \partial P & P\mathcal{B}_1 & P\mathcal{B}_2 \\ \mathcal{B}_1^T P & 0 & 0 \\ \mathcal{B}_2^T P & 0 & -I \end{bmatrix} + \lambda \begin{bmatrix} \mathcal{C}_1^T \\ \mathcal{D}_{11}^T \\ \mathcal{D}_{12}^T \end{bmatrix} M \begin{bmatrix} \mathcal{C}_1 & \mathcal{D}_{11} & \mathcal{D}_{12} \end{bmatrix} < 0, \tag{58}$$

and

$$\begin{bmatrix} P & \mathcal{C}_2^T \\ \mathcal{C}_2 & \sigma^2 I \end{bmatrix} > 0. \tag{59}$$

Then, the Supreme Square Error (SSE) satisfies:

$$\sup_{s\in\Omega, 0\neq\eta\in\mathcal{L}_2, \zeta(0)=0} \left[\frac{||y - \hat{y}||_\infty}{\sqrt{||d||_2^2 + ||\hat{x}||_2^2}} \right] < \sigma. \tag{60}$$

Proof. Considering the signals (χ, z, η, q, r) generated by the extended system as outlined in Equation (46), where the input η belongs to $\mathcal{L}_2([0, \infty))$ and the parameter trajectory s resides within Ω, all under the assumption of zero initial conditions. The well-posedness of the extended Linear Parameter Varying (LPV) system ensures that all signals (χ, z, η, q, r) are well-defined. Furthermore, it is assumed that the uncertainties Δ_δ and ϵ adhere to the Integral Quadratic Constraints (IQCs) specified by (Ξ, M), necessitating that the signal r satisfies the time-domain expression of the IQCs in Equation (15) for any $T > 0$.

A storage function $V : \mathbb{R}^{n_\chi} \times \Omega \to \mathbb{R}_+$ is defined by $V(\chi, s) = \chi^T P(s)\chi$. Additionally, the state-space representation of the virtual filter described in Equation (13) leads to (50).

Evaluating the perturbed Linear Matrix Inequality (LMI) in Equation (58) at $(s(t), \dot{s}(t))$ and performing multiplication from the left and right by $[\chi^T, q^T, \eta^T]$ and its transpose $[\chi^T, q^T, \eta^T]^T$, respectively, results in the following dissipation inequality (see Appendix B for more details):

$$\nabla_{x_\chi} V \dot{\chi} + \nabla_{x_s} V \dot{s} \leq (1 - \varepsilon)\eta\eta^T - \lambda r M r^T \tag{61}$$

This implies that the dissipation inequality presented in Equation (B14) is directly equivalent to the perturbed linear matrix inequality (LMI) delineated in Equation (58). Proceeding with this understanding, integrating the dissipation inequality obtained in Equation (B14) along the state/parameter trajectory from the initial time $t = 0$ to an arbitrary upper bound $t = T$ and taking into account that the initial condition for χ is $\chi(0) = 0$, we arrive at the following conclusion:

$$V(\chi(T), s(T)) \leq (1 - \varepsilon) \int_0^T \eta(\tau)\eta(\tau)^T \, d\tau - \lambda \int_0^T r(\tau) M r(\tau)^T \, d\tau \tag{62}$$

It follows from the IQCs condition (15), that

$$V(\chi(T), s(T)) \leq (1 - \varepsilon) \int_0^T \eta(\tau)\eta(\tau)^T \, d\tau \tag{63}$$

Now, applying Schur complement on (59) and multiplying left and right by χ and its transpose χ^T, respectively, results:

$$\frac{1}{\sigma^2} z(t) z^T(t) \leq \chi^T(t) P(s) \chi(t) \quad \forall t > 0 \tag{64}$$

Evaluating (64) at $t = T$, and applying (63) yields:

$$\frac{1}{\sigma^2} z(T) z^T(T) \leq (1 - \varepsilon) \int_0^T \eta(\tau)\eta(\tau)^T \, d\tau \tag{65}$$

Knowing the $\varepsilon > 0$, therefore

$$\frac{1}{\sigma^2} z(T) z^T(T) < \int_0^T \eta(\tau)\eta(\tau)^T \, d\tau \tag{66}$$

Considering the supremum over T to demonstrate that $||z||_\infty \leq \sigma ||\eta||_2$. Knowing that this result holds for any given input $\eta \in \mathcal{L}_2([0, \infty))$, admissible trajectory solution $s \in \Omega$, and uncertainty $\Delta_\delta, \epsilon \in IQC(\Xi, M)$. Thus,

$$\sup_{s\in\Omega.\ 0\neq\eta\in\mathcal{L}_2, \zeta(0)=0} \frac{||z(t)||_\infty}{||\eta(t)||_2} < \sigma \tag{67}$$

Having $||z(t)||_\infty = ||y(t) - \hat{y}(t)||_\infty$ and $||\eta(t)||_2 = \sqrt{||d(t)||_2^2 + ||\hat{x}(t)||_2^2}$, this implies

$$\frac{||z(t)||_\infty}{||\eta(t)||_2} = \frac{||y - \hat{y}||_\infty}{\sqrt{||d||_2^2 + ||\hat{x}||_2^2}} \tag{68}$$

□

Theorem 2 establishes an explicit upper bound on the worst-case Supreme Square Error (SSE), which quantifies the maximum instantaneous deviation between the NN-controlled system and the reference closed-loop model over all admissible uncertainty trajectories. This result provides a rigorous robustness certificate by guaranteeing that the deviation remains uniformly bounded despite the presence of system uncertainties and neural network approximation errors. In contrast to integral performance measures such as the RISE bound in Theorem 1, the SSE bound characterizes the peak deviation of the system output. This provides critical information regarding transient behavior and ensures that the system output remains within a certified safety envelope relative to the reference model at all times. As a result, Theorem 2 enables direct assessment of the worst-case instantaneous performance degradation induced by uncertainties and neural network nonlinearities. From a practical standpoint, this worst-case bound serves as a quantitative safety and reliability indicator as well. A smaller SSE bound implies tighter adherence to the reference closed-loop behavior and therefore higher robustness of the NN controller. Conversely, larger bounds indicate increased sensitivity to uncertainties and approximation errors. Consequently, the theorem provides a predictive and verifiable guarantee on system performance, supporting the safe deployment of neural network controllers in safety-critical and uncertain dynamical environments.

The proposed robustness guarantee is realized through a verification procedure applied after the neural network controller

has been trained, as illustrated in the case studies presented in Section 6. A trusted reference closed-loop model is first selected to represent the desired system behavior, while the uncertainties affecting the real system are characterized through appropriate IQC descriptions. Using the trained NN controller, the reference model, and the uncertainty characterization, the dynamical deviation between the two closed-loop systems is constructed, leading to the extended error system. The corresponding IQC optimization problem is then formulated and solved using an appropriate IQC analysis toolbox, yielding the RISE and SSE values, which constitute certified upper bounds on the worst-case deviation between the reference model and the NN-controlled uncertain system. These bounds can subsequently be combined with the reference model to construct a certified envelope around the reference closed-loop trajectory. This envelope represents the worst-case region within which the behavior of the NN-controlled system is guaranteed to evolve under all admissible uncertainties considered in the analysis. Consequently, the designer can assess, prior to deployment, whether the predicted system behavior satisfies the required robustness and safety specifications.

6 | Validation and Numerical Simulation

To validate the theoretical results established in the previous section, we conduct numerical simulations on two distinct case studies of increasing complexity. The first case study is the *Single-Link Robot Arm Control Problem*, a single-output system in which the neural network is trained to emulate a linear closed-loop behavior. This makes the linear model a natural and readily available reference for applying the keep-close framework. The simplicity of this scenario allows us to clearly demonstrate the fundamental effectiveness of the proposed method in a controlled setting. The second case study examines a more challenging multi-output problem involving the *Deep Guidance and Control of the Apollo Lander*. In this scenario, the neural network is trained using data generated from nonlinear dynamics, and an appropriate linear reference model must be derived for use within the keep-close approach. This setting represents a realistic guidance and landing problem, where precise control is required during descent in the presence of disturbances and nonlinear aerodynamic effects.

In summary, the first case study uses a single-output system with an explicitly available reference model, whereas the second addresses a multi-output system in which the reference model is not provided and must be constructed. Together, these two examples enable a comprehensive assessment of the robustness and adaptability of the keep-close method across both simple and complex control scenarios. Moreover, we utilize the IQClab Toolbox,[1] a MATLAB-based extension of the Robust Control Toolbox, designed for robustness analysis and control design in uncertain and linear parameter-varying (LPV) systems. As described by [37] and compared with another toolbox such as IQC-β [41], IQClab provides a versatile framework incorporating integral quadratic constraint (IQC) methodologies, enabling efficient model reduction, control switching schemes, and performance weighting function generation. The modular architecture of the toolbox facilitates seamless integration with various solvers and parsers, making it a powerful tool for developing and implementing new linear matrix inequality (LMI) and IQC-based algorithms. Given its adaptability and ease of use, IQClab plays a crucial role in our analysis, offering a robust computational environment for evaluating system stability and performance.

6.1 | Case Study I: Single-Link Robot Arm Control Problem

In this scenario, we consider a Single-Link Robot Arm Control Problem to illustrate the effectiveness of the keep-close approach. The neural network controller is trained to regulate the nonlinear dynamics of the arm so as to track a desired trajectory, while replicating the behavior of an associated linear model. The training process follows the model reference neural control procedure.[2] The linear model is subsequently employed as the reference closed-loop system. It serves as a benchmark against which the trained NN is evaluated using the keep-close framework, as detailed in the remainder of this subsection.

6.1.1 | Step 1: Problem Setting

Consider the nonlinear single-link robot arm example, shown in Figure 4, with parameters $m = 0.15kg$, length $l = 0.5m$, and friction coefficient $\mu = 0.5Nms/rad$. The state space representation of the arm's motion is given by:

$$\begin{cases} \dot{\omega}(t) &= -10\sin(\theta(t)) - 2\omega(t) + \tau(t) \\ \dot{\theta}(t) &= \omega(t) \end{cases} \tag{69}$$

Let assume the static nonlinearity $\delta(t) = \Delta_\delta(\theta(t)) = \theta(t) - \sin(\theta(t))$ which is slope-restricted and sector bounded. We rearrange (69) then into the form:

$$\begin{cases} \dot{\omega}(t) &= -10\ \theta(t) - 2\omega(t) + \tau(t) + 10\delta(t) \\ \dot{\theta}(t) &= \omega(t), \\ y(t) &= \theta(t), \\ \Delta_\delta(\theta(t)) &= \delta(t) = \theta(t) - \sin(\theta(t)) \end{cases} \tag{70}$$

which is the same form as (19) with

$$A = \begin{bmatrix} -2 & -10 \\ 1 & 0 \end{bmatrix},\ B = \begin{bmatrix} 1 \\ 0 \end{bmatrix},\ \tilde{B} = \begin{bmatrix} 10 \\ 0 \end{bmatrix},\ C = \begin{bmatrix} 0 & 1 \end{bmatrix},\ D = 0, \tilde{D} = 0. \tag{71}$$

6.1.2 | Step 2: Linear Reference Model

To apply the keep-close technique, a reference model is required. In this case, the neural controller has been trained explicitly with the objective of ensuring that the nonlinear arm, when governed by this controller, reproduces the behavior of the following closed-loop linear model, which serves as the reference dynamics:

$$\begin{cases} \dot{\omega}_r(t) &= -9\theta_r(t) - 6\omega_r(t) + 9r(t) \\ \dot{\theta}_r(t) &= \omega_r(t) \end{cases} \tag{72}$$

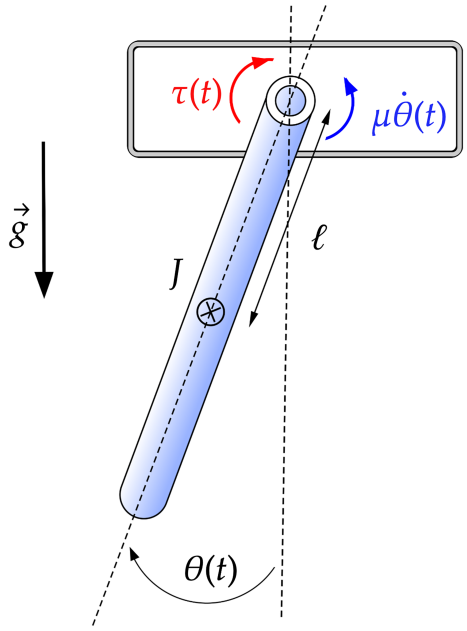

FIGURE 4 | Single-link robot arm.

Therefore, the system (72) is an excellent reference model to compare the uncertain system (70) with is given by

$$\begin{cases} \dot{\omega}_r(t) & = -10\ \theta_r(t) - 2\ \omega_r(t) + \tau_r(t) \\ \dot{\theta}_r(t) & = \omega_r(t) \\ \pi^*(r(t), \hat{x}(t)) & = \tau_r(t) = \theta_r(t) - 4\ \omega_r(t) + 9r(t). \end{cases} \tag{73}$$

This implies that the dynamic behavior of the system (70) in closed-loop, after training the neural controller, approximates the desired closed-loop dynamics (73). Ideally, the system governed by the neural network controller would perfectly replicate the reference model for all t. In practice, however, exact alignment is unattainable due to nonlinear effects and modeling inaccuracies.

The purpose of this case study is therefore to quantify the deviation between the reference model (73) and the uncertain nonlinear system (70) controlled by the neural network. The keep-close framework enables the calculation of the worst-case relative squared deviation, denoted by γ in Theorem 1, between the closed-loop trajectories of the robot arm and the ideal reference model. This bound acts as a certified estimate of the region within which the output of the uncertain system remains for all $t > 0$, thus providing a rigorous safety guarantee.

$$\sup_{s\in\Omega.\ 0\neq[\tau(t)\ \theta_r(t)]^T\in\mathcal{L}_2,\zeta(0)=0} \sup \frac{\left|\left|\theta(t) - \theta_r(t)\right|\right|_2}{\left|\left|[r(t)\ \hat{x}(t)]^T\right|\right|_2} < \gamma \tag{74}$$

6.1.3 | Step 3: Error Dynamical System

Now, we generate the dynamical error system using the closed-loop reference model (73) and the new mathematical form of the motion of the arm (70). Let $\delta\omega = \omega - \omega_r$, $\delta\theta = \theta - \theta_r$ and $\delta\tau = \tau - \tau_r$ then

$$\frac{d}{dt}\zeta(t) = \begin{bmatrix} -2 & -10 \\ 1 & 0 \end{bmatrix}\zeta(t) + \begin{bmatrix} 10 \\ 0 \end{bmatrix} q(t) + \begin{bmatrix} 1 \\ 0 \end{bmatrix}\delta\tau(t), \quad \zeta(t) = \begin{bmatrix} \delta\omega \\ \delta\theta \end{bmatrix} \tag{75}$$

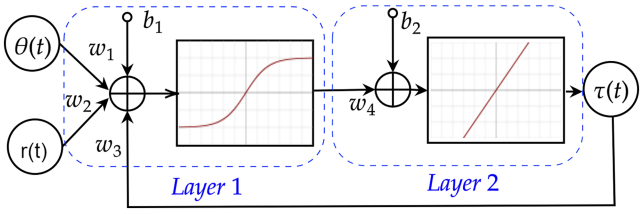

FIGURE 5 | Neural Network controller π Architecture.

The neural controller architecture adopted in this example is given in Figure 5. The neural network controller is designed to map the reference signal $r(t)$ and the measured plant variable $\theta(t)$ to the control action $\tau(t)$. It consists of a single hidden layer with 13 neurons, enabling the approximation of nonlinear control laws. The hidden layer applies a hyperbolic tangent activation function, and the output layer is linear. The controller therefore adapts its parameters online to minimize the tracking error with respect to the reference model. Let $\omega_1,\ \omega_2,\ \omega_3 \in \mathbb{R}^{13\times1}$ and $\omega_4 \in \mathbb{R}^{1\times13}$ denote the input-to-hidden and hidden-to-output weight matrices, respectively. The bias vectors are $b_1 \in \mathbb{R}^{13}$ for the hidden layer and $b_2 \in \mathbb{R}$ for the output layer.

The explicit form of the controller is

$$\pi(r(t), \theta(t)) = w_4 \tanh\big(w_1\theta(t) + w_2 r(t) + w_3\pi(r(t), \theta(t)) + b_1\big) + b_2 \tag{76}$$

where $\tau(t) = \pi(r(t), \theta(t))$. Componentwise, this can be written as

$$\tau(t) = \sum_{i=1}^{13} w_{4,i} \tanh\big(w_{1,i}\theta(t) + w_{2,i} r(t) + w_{3,i}\pi(r(t), \theta(t)) + b_{1,i}\big) + b_2$$

Using the Differential Mean Value Theorem [30] introduced in Lemma 1, we can convert the dynamical error system to an LPV system without nonlinearities of the neural controller. According to Lemma 3, the difference between the two controller $\mu(t) = \delta\tau(t)$ given in Equation (22) can be expressed as:

$$\delta\tau(t) = \mu(t) = \Lambda(s)\zeta(t) + \frac{\partial\alpha}{\partial\hat{x}}(0,0)\hat{x}(t) + \frac{\partial\alpha}{\partial d}(0,0)d(t) + \epsilon(t), \tag{77}$$

where the trajecory $s(t) = [d(t), c(t)]^T$, the LPV matrix $\Lambda(s(t)) = \nabla_x\pi(d(t), c(t))$ is the Jacobian matrix of the neural network, $\epsilon(t) = \mathcal{O}(d(t), \hat{x}(t))$ and the training error $\alpha(d(t), \hat{x}(t)) = \pi(d(t), \hat{x}(t)) - \pi^*(d(t), \hat{x}(t))$. In this case, we have $C = \begin{bmatrix} 0 & 1 \end{bmatrix}$, $\zeta(t) = \begin{bmatrix} \delta\omega(t) & \delta\theta(t) \end{bmatrix}^T$, $\hat{y}(t) = \theta_r$ and $y(t) = \theta(t)$. Therefore,

$$\delta\tau(t) = \Lambda(s)\zeta(t) + \frac{\partial\alpha}{\partial\hat{x}}(0,0)\hat{x}(t) + \frac{\partial\alpha}{\partial r}(0,0)r(t) + \epsilon(t) \tag{78}$$

where $\pi^*(.,.)$ and $\pi(.,.)$ are given in Equations (73) and (76), respectively, and

$$\alpha(r(t), \hat{x}) = \pi(r(t), \theta_r(t)) - \pi^*(r(t), \theta_r(t)) \tag{79}$$

Now, to calculate $\Lambda(s)$ let consider the function of he neural controller (76)

$$\pi(r(t), \theta(t)) = w_4 \tanh\big(w_1\theta(t) + w_2 r(t) + w_3\pi(r(t), \theta(t)) + b_1\big) + b_2. \tag{80}$$

Differentiating both sides with respect to ω and θ and applying implicit differentiation, we obtain:

$$\frac{\partial \pi}{\partial \omega}(r(t),\theta(t)) = 0,$$
$$\frac{\partial \pi}{\partial \theta}(r(t),\theta(t)) = w_4 \operatorname{sech}^2(\varrho(r(t),\theta(t)))\Big(w_1 + w_3 \frac{\partial \pi}{\partial \theta}(r(t),\theta(t))\Big), \tag{81}$$

where

$$\varrho(r(t),\theta(t)) = w_1\theta(t) + w_2 r(t) + w_3\pi(r(t),\theta(t)) + b_1. \tag{82}$$

Rearranging and solving for $\frac{\partial \pi}{\partial \theta}$, we obtain:

$$\Lambda(s(t)) = \left[\frac{\partial \pi}{\partial \omega}(s(t)) \;\; \frac{\partial \pi}{\partial \theta}(s(t))\right] = \left[0 \;\; \frac{w_4 w_1 \operatorname{sech}^2(\varrho(s(t)))}{1 - w_4 w_3 \operatorname{sech}^2(\varrho(s(t)))}\right], \tag{83}$$

where the trajectory $s(t) = [r(t)\ c(t)]$ and $c \in \mathbb{C} \rtimes (\theta, \theta_r)$ lies on the curve $\vartheta(\nu) = \theta + \nu(\theta_r - \theta)$ for some $\nu \in [0,1]$. Also, it is clear that

$$\frac{\partial \alpha}{\partial \omega_r}(0,0) = 4,\ \frac{\partial \alpha}{\partial \theta_r}(0,0) = \frac{w_4 w_1 (1 - \tanh^2(b_1))}{1 - w_4 w_3 (1 - \tanh^2(b_1))} - 1,$$
$$\frac{\partial \alpha}{\partial r}(0,0) = \frac{w_4 w_2 (1 - \tanh^2(b_1))}{1 - w_4 w_3 (1 - \tanh^2(b_1))} - 9 \tag{84}$$

Therefore, the dynamical error system became:

$$\begin{cases} \frac{d}{dt}\zeta(t) = \begin{bmatrix} -2 & -10 \\ 1 & 0 \end{bmatrix}\zeta(t) + \begin{bmatrix} 10 \\ 0 \end{bmatrix} q(t) + \begin{bmatrix} 1 \\ 0 \end{bmatrix}\delta\tau(t), \\ \delta\tau(t) = \Lambda(s)\zeta(t) + \frac{\partial \alpha}{\partial \hat{x}}(0,0)\hat{x} + \frac{\partial \alpha}{\partial r}(0,0) r(t) + \epsilon(t), \end{cases} \tag{85}$$

where $\Lambda(s)$, $\frac{\partial \alpha}{\partial \hat{x}}(0,0)$ and $\frac{\partial \alpha}{\partial \theta_r}(0,0)$ are given in Equations (83) and (84) and depend on parameters of the neural network controller w_1, w_1, w_2, w_3, w_4, b_1 and b_2. The exact numerical values of these parameters can be extracted directly from the trained controller object in MATLAB. The Jacobian of the neural controller becomes

$$\Lambda(s(t)) = \left[0 \;\; \frac{2.16\ \operatorname{sech}^2(\varrho(s(t)))}{1 - 0.90\ \operatorname{sech}^2(\varrho(s(t)))}\right].$$

Moreover, the partial derivatives of α at $(0,0)$ are

$$\frac{\partial \alpha}{\partial \omega_r}(0,0) = 4, \quad \frac{\partial \alpha}{\partial \theta_r}(0,0) = 10.21, \quad \frac{\partial \alpha}{\partial r}(0,0) = -16.00.$$

6.1.4 | Step 4: Characterization of the Nonlinearities Δ_δ and Δ_ϵ Using IQCs

To choose the type of IQCs of the uncertainties considered for this simulation example, let us start with the static nonlinearity $\Delta_\delta(\theta) = \theta - \sin(\theta)$. It is clear from Figure 6 that the uncertainty $\Delta_\delta(\theta)$ is slope-restricted and sector-bounded. Figure 6 provides the IQCs characterization of this nonlinearity and plots the graph $\Delta_\delta = f(\theta)$ where $\Delta_\delta(\theta(t)) = \varphi(\theta,t)$ and $\tilde{\alpha}\theta(t) \le \varphi(\theta,t) \le \tilde{\beta}\theta(t)$ for all $\theta \in [-\frac{\pi}{2}, \frac{\pi}{2}]$. Simple calculation of the slope gives $\tilde{\alpha} = 0$ and $\tilde{\beta} = 0.364$ representing the bounds of the uncertainty Δ_δ.

A similar argument is applied to the nonlinearity, we have the nonlinearity $\epsilon(t) = \Delta_\epsilon(r(t),\hat{x}(t)) = \mathcal{O}(r(t),\hat{x}(t))$ which is reminder of the first-order approximation of $\alpha(r(t),\hat{x}) = \pi(r(t),\hat{x}(t)) - \pi^*(r(t),\hat{x}(t))$, where π^* and π are given in Equations (73) and (76), respectively. According to (34) we have

$$\epsilon(t) = \Delta_\epsilon(r(t),\hat{x}(t)), \quad |\epsilon(t)| \le \kappa_1 |r(t)| + \kappa_2 |\theta_r(t)|, \quad \forall\ r, \theta_r \in \left[-\frac{\pi}{2}, \frac{\pi}{2}\right], \tag{86}$$

Therefore, it follows that $\epsilon(t)$ is confined within the sector defined by the bounds $\tilde{\alpha}_r = -0.2$ and $\tilde{\beta}_r = 0.2$, and thus admits an appropriate IQC characterization suitable for integration into the keep-close analysis.

6.1.5 | Step 5: IQC-Based Analysis Results

Now, we would like to compute an upper bound on the $\mathcal{L}_2$-gain from the input vector $[r,\ \theta_r]^T$ to the output error $\theta - \theta_r$. We use the IQClab toolbox to express and formulate the $\mathcal{L}_2$ gain estimation problem. We only need the two linear systems, the closed-loop reference model and the dynamical error system,

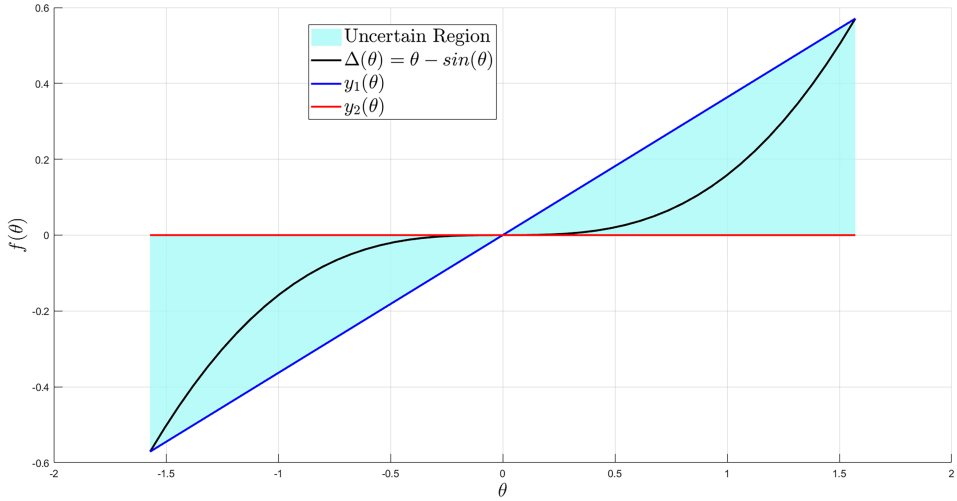

FIGURE 6 | IQCs characterization of the nonlinearity Δ_δ.

since the output of the closed-loop system with the NN controller is the sum of the outputs of those two systems. The bounds of the uncertainty Δ_δ and the nonlinearity Δ_ϵ derived in the previous subsection are also provided as inputs to the IQClab Toolbox.

According to the results given by IQClab for the model in Equation (85) corresponding to the original Single-Link Robot Arm Control Problem:

$$\sup_{s\in\Omega.\ 0\neq[r(t)\ \theta_r(t)]^T\in\mathcal{L}_2,\zeta(0)=0} \sup \frac{\left|\left|\theta(t)-\theta_r(t)\right|\right|_2}{\left|\left|[r(t)\ \theta_r(t)]^T\right|\right|_2} < \gamma = 0.04009 \tag{87}$$

According to [40], the time Integral of the Square of the Error (ISE) is commonly used in control systems as a measure of system performance. Thus, γ in Equation (87) represents the worst-case mean integrated squared error between the closed-loop system with the trained neural controller and the closed-loop reference system. In other words, the value of γ quantifies the worst-case relative error between the closed-loop reference model, which serves as the basis for training the neural network controller, and the uncertain system operating with the designed neural controller. To illustrate further the results of our controlled Single-Link Robot Arm with our designed neural controller, we propose two case studies: A and B. The reference of the first case study is periodically applied to the uncertain system with a neural controller and the reference closed-loop model. Using the equation given in Equation (87), we can plot the expected region by utilizing the information about the worst-case of the error and the known linear reference model as follows:

$$\left\|\theta(t)-\theta_r(t)\right\|_2 < \gamma\left\|[r(t)\ \theta_r(t)]^T\right\|_2 = 0.04009\sqrt{\|r(t)\|_2^2+\left\|\theta_r(t)\right\|_2^2} \tag{88}$$

Knowing that,

$$\theta_r(t)-\left\|\theta(t)-\theta_r(t)\right\|_2 \le \theta(t) \le \theta_r(t)+\left\|\theta(t)-\theta_r(t)\right\|_2 \tag{89}$$

Using (88), we conclude that

$$\begin{aligned}\theta_r(t)-0.04009\sqrt{\|r(t)\|_2^2+\left\|\theta_r(t)\right\|_2^2} \le \theta(t)\\ \le \theta_r(t)+0.04009\sqrt{\|r(t)\|_2^2+\left\|\theta_r(t)\right\|_2^2}\end{aligned} \tag{90}$$

Using this equation is of significant importance, as it implies that, by relying solely on the known quantities $r(t)$ and $\theta_r(t)$, we can predict the behavior of the uncertain system controlled by the neural network. More specifically, it allows us to determine the expected regions and boundaries within which the system's output will remain, despite uncertainties and nonlinearities.

This insight is particularly crucial in control applications, as it enables the assessment of the robustness and reliability of the neural network controller to maintain stability and performance under varying conditions. By characterizing these boundaries, we gain a deeper understanding of the extent to which the trained neural controller approximates the desired closed-loop reference model. Moreover, this formulation provides a foundation for further optimization, enabling the refinement of the neural controller to enhance its adaptability and accuracy in real-world applications. This worst-case result can be used to improve the robustness of the NN controller by making the dynamical error smaller, which in turn will guarantee better performance of the NN controller when implemented. This worst-case measure is also an indication of the likelihood of any undesirable behavior in the closed-loop system when adopting the designed NN controller.

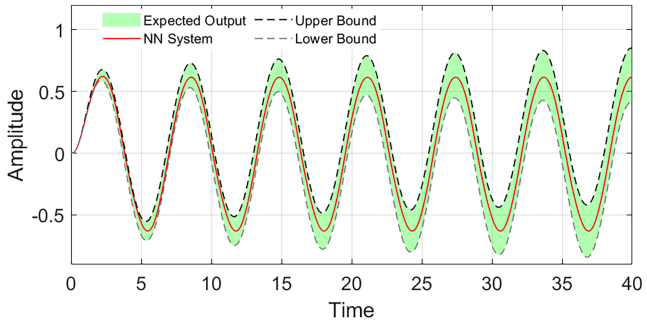

FIGURE 7 | **Scenario A**: Plot for the closed-loop system with the NN within the region of expectation.

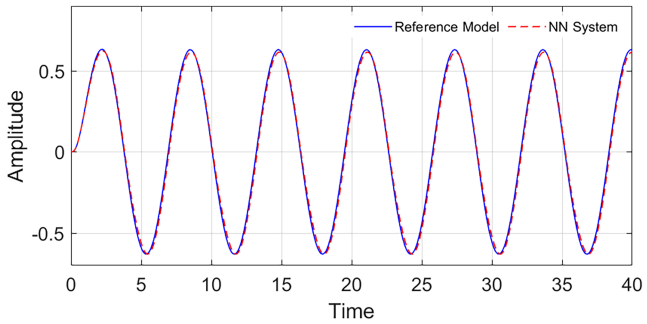

FIGURE 8 | **Scenario A**: The controllers π and π^* outputs.

Figure 7 illustrates the response of the system with the trained neural network (NN) controller, where it is evident that the output remains within the expected bounded region. In Figure 8, the responses of both the closed-loop reference model and the uncertain plant with the NN controller are presented for this first scenario. From this figure, it can be observed that the uncertain plant with the NN controller **closely follows** the response of the closed-loop reference model.

Additionally, the output of the neural controller remains highly similar to that of the reference controller used in the closed-loop reference model, as depicted in Figure 9. The observations made for Figures 7–9 are equally applicable to the non-periodic scenario B, illustrated in Figures 10–12, where the system's response with the NN controller consistently remains within the expected performance region. These results further validate our findings, demonstrating the effectiveness of the neural network controller in maintaining stability and ensuring system performance.

6.2 | Case Study II: Deep Guidance and Control of Apollo Lander

In the classical Guidance and Control architecture for the Apollo Lander, as described in Reference [42], the system generates

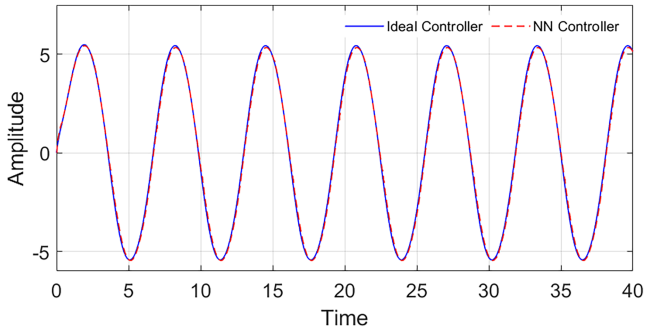

FIGURE 9 | **Scenario A**: Plot of the output of the reference model response θ_r indicated in blue and the one of the closed-loop uncertain robotic arm system with neural controller θ indicated in red.

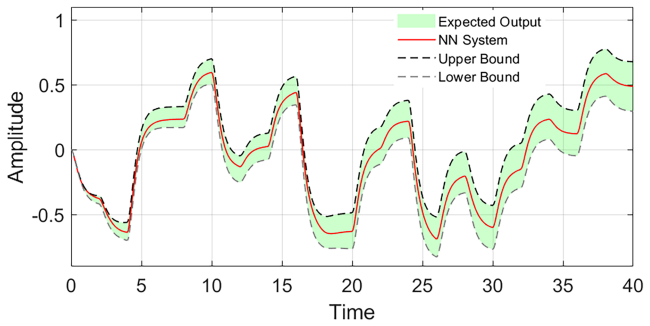

FIGURE 10 | **Scenario B**: Plot for the closed-loop system with the NN within the region of expectation.

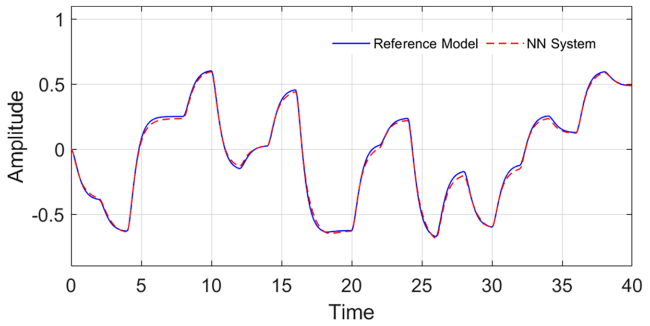

FIGURE 11 | **Scenario B**: The controllers π and π^* outputs.

the required trajectory and thrust commands to manoeuvre the lander from a given initial position and velocity toward a desired landing target. The control subsystem compensates for the Martian atmospheric (see [29] for instance) effects and performs the necessary transformations into inertial coordinates before trajectory tracking is executed [43].

In this case study, we remove the classical controller and replace it with a neural network-based controller. This substitution naturally raises a key concern: Can the neural network be trusted to guarantee safe landing performance in the presence of external disturbances and model uncertainties? To address this, we employ the keep-close framework to quantify the worst-case deviation between the ideal reference model and the NN-controlled lander dynamics under atmospheric disturbances and parametric uncertainty. This enables a rigorous safety validation by

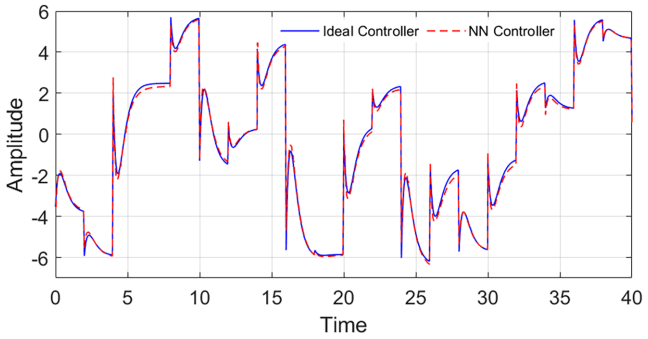

FIGURE 12 | **Scenario B**: Plot of the output of the reference model response θ_r indicated in blue and the one of the closed-loop uncertain robotic arm system with neural controller θ indicated in red.

TABLE 1 | Simulation Parameters.

Parameter	value
Initial Position p_0 $[m]$	$[-5632.2, 709.25, 6190.5]^T$
Initial Velocity v_0 $[m/s]$	$[206.48, -26.006, -103.51]^T$
Initial Acceleration a_0 $[m/s^2]$	$[-2.1124, 0.24947, -2.6404]^T$
Initial Mass m_0 $[Kg]$	600.3
The capsule reference area S	5.137426149499100
Mars Gravity g $[m/s^2]$	$[0, 0, 3.725258]^T$
Engine Maximum Thrust T $[N]$	3,600
Maximum Throttle Level T_{max}	100%
Minimum Throttle Level T_{min}	30%

certifying whether the neural network maintains the lander sufficiently close to the reference behavior throughout the descent.

6.2.1 | Step 1: Problem Setting

This problem represents a realistic and demanding benchmark, making it one of the most suitable case studies for validating the effectiveness of the keep-close approach. We evaluate the trained neural network in closed loop with the architecture and parameters listed in Table 1, with the objective of certifying its performance and robustness through the keep-close framework. The descent dynamics explicitly account for aerodynamic forces and torques acting as external disturbances, characterized by a drag coefficient $C_D = 2.0$, a surface-relative head wind of up to 20 m/s, and a constant air density $\rho = 0.023$ kg/m^3. Under these conditions, the worst-case aerodynamic resistance during the descent phase remains below 2% of the vehicle's maximum thrust. Owing to the closed-loop control architecture, these aerodynamic forces and torques are automatically compensated, allowing the keep-close analysis to rigorously assess whether the neural network controller maintains safe and bounded behavior under realistic environmental disturbances.

6.2.1.1 | **Atmosphere.** As part of the Mars environment module, the horizontal wind is included in the atmospheric

model. Based on lookup tables, an atmospheric model specifies the properties of the Martian atmosphere at a specific location. The atmospheric properties are obtained directly from the Mars Climate Database (MCD) v.5.2 at the mission's specified landing coordinates. MCD is derived from numerical simulations of the Martian atmosphere using General Circulation Models (GCM) and validated by observations of the Martian atmosphere. For more details see [29].

6.2.1.2 | Lander Dynamics. In closed-loop simulations, Newton's second law is used to model a point mass's robotic lander motion, where the lander motion is expressed in the LENU frame as follows:

$$\frac{d[r]_{LENU}}{dt} = [v]_{LENU}, \tag{91}$$

where $([v]_{LENU})$ refers to the relative velocity of the robotic lander. In the LENU frame and the lander mass (m), the dynamics equation is given based on the resultant force $([F]_{LENU})$ expressed as:

$$\frac{d[v]_{LENU}}{dt} = [a]_{LENU} = \frac{[F]_{LENU}}{m}. \tag{92}$$

6.2.1.3 | Aerodynamics. In the simulations, a lookup table is used as the aerodynamics model of the Apollo capsule. In this module, only aerodynamic drag forces are considered following equation:

$$F_D = \frac{1}{2}\rho v^2 C_D(\mathrm{Ma})S, \tag{93}$$

where ρ represents the atmosphere density, C_D denotes the capsule drag coefficient as a function of the Mach number (Ma), the capsule reference area represented by the S, and v is the capsule velocity relative to the fluid velocity expressed in the landing site East-North-UP frame.

6.2.1.4 | Actuators. In the actuator model, four thruster engines are used to generate forces concerning the center of mass of the vehicle. The main engine responsible for the force F_{Thr} is located along the vehicle z-axis, while the remaining are the sources of saturated forces F_{Trq}, are placed at a distance l from the mass center to generate pure torques along the positive and negative axes. To calculate the fuel consumption, the mass flow rate is integrated:

$$\dot{m}_{fuel} = V_e^{-1}\left(F_{Thr} + \sum F_{Trq}\right), \tag{94}$$

where $V_e = g_0 I_{sp}$.

6.2.1.5 | Apollo Guidance. The algorithm is based on the idea of following the reference trajectory backwards in time from the target point to the present moment (Figure 13). The time-to-go (t_{go}) represents the time left until the desired landing is reached, so it tends to zero as time gets closer to the goal. In Reference [44], the reference trajectory satisfies a two-boundary problem with five degrees of freedom. Accordingly, the reference trajectory must be a polynomial function of fourth or higher order. Therefore, the reference position (r_d) given at t_{go} as a function of the target state as:

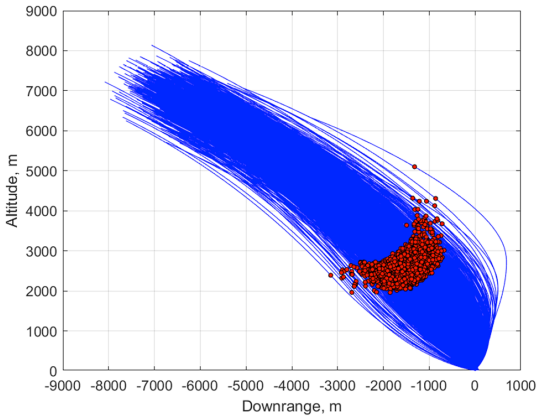

FIGURE 13 | Plot of the 1537 trajectories. The marked dots represent the trigger locations. An optimal triggering altitude is in the range of 2 to 3 km, and for some trajectories, overshooting the target is the optimal behavior. Reproduced from [42].

$$r_d = r_t + v_t t_{go} + a_t \frac{t_{go}^2}{2} + j_t \frac{t_{go}^3}{6} + s_t \frac{t_{go}^4}{24}, \tag{95}$$

where r_t refers to the target position, v_t represents the target velocity, and a_t, j_t, s_t denote the target acceleration, jerk, and snap, respectively. According to (95), the work of [45] describes the acceleration command (a_{cmd}) as a quadratic polynomial, that is,

$$a_{\mathrm{cmd}} = C_0 + C_1 t_{go} + C_2 t_{go}^2. \tag{96}$$

The coefficients of the polynomial are derived by solving a system of three equations comprising (96). The two other equations related to the target velocity and position equations are obtained by integrating the target acceleration equation:

$$\begin{cases} a_t = C_0 + C_1 t_{go} + C_2 t_{go}^2 \\ v_t = C_0 t_{go} + \frac{1}{2} C_1 t_{go}^2 + \frac{1}{3} C_2 t_{go}^3 + v \\ r_t = v t_{go} + \frac{1}{2} C_0 t_{go}^2 + \frac{1}{6} C_1 t_{go}^3 + \frac{1}{12} C_2 t_{go}^4 + r \end{cases} \tag{97}$$

where a_t represents the reference acceleration vector, r and v denote the vehicle current position and velocity, respectively. The following values for the polynomial coefficients C_0, C_1 and C_2 are obtained by solving the following matrix system:

$$\begin{bmatrix} a_t \\ v_t \\ r_t \end{bmatrix} = \begin{bmatrix} 1 & t_{go} & t_{go}^2 \\ t_{go} & \frac{1}{2} t_{go}^2 & \frac{1}{3} t_{go}^3 \\ \frac{1}{2} t_{go}^2 & \frac{1}{6} t_{go}^3 & \frac{1}{12} t_{go}^4 \end{bmatrix} \begin{bmatrix} C_0 \\ C_1 \\ C_2 \end{bmatrix} + \begin{bmatrix} 0 \\ v \\ v_{t_{go}} \end{bmatrix}. \tag{98}$$

Therefore,

$$\begin{cases} C_0 = a_t - 6\left(\frac{v_t + v}{t_{go}}\right) + 12\left(\frac{r_t - r}{t_{go}^2}\right) \\ C_1 = -6\left(\frac{a_t}{t_{go}}\right) + 6\left(\frac{5v_t + 3v}{t_{go}^2}\right) - 48\left(\frac{r_t - r}{t_{go}^3}\right) \\ C_2 = 6\left(\frac{a_t}{t_{go}^2}\right) - 12\left(\frac{2v_t + v}{t_{go}^3}\right) + 36\left(\frac{r_t - r}{t_{go}^4}\right) \end{cases} \tag{99}$$

By defining the vertical component of the acceleration profile as a linear function of t_{go}. Hence, solving (96) for t_{go} two solutions are obtained:

$$\begin{cases} \textbf{If } \left|\left(a_t\right)_z\right| > 0 \\ t_{go} = \frac{2\left(v_t\right)_z + \left(v\right)_z}{\left(a_z\right)_t} + \sqrt{\left[\frac{2\left(v_t\right)_z + \left(v\right)_z}{\left(a_t\right)_z}\right]^2 + \frac{6\left[\left(\mathbf{r}\right)_z - \left(r_t\right)_z\right]}{\left(a_t\right)_z}}, \\ \textbf{else if} \left(a_t\right)_z = 0 \\ t_{go} = \frac{3\left[\left(r_t\right)_z - \left(r\right)_z\right]}{\left(v\right)_z + 2\left(v_t\right)_z}, \end{cases} \tag{100}$$

where $(.)_z$ denotes the vertical component of the variable $(.)$.

A robustness analysis of feedback systems for the Apollo lander based on NN controllers against potential uncertainties is presented here. To achieve this, we need to follow the pattern below, where we aim at keeping the output of the closed-loop system with an NN controller close to the output of an ideal and trusted closed-loop reference model when its input changes within a bounded set.

6.2.2 | Step 2: Linear Reference Model

In contrast to the previous case study, the Apollo lander system exhibits a strongly nonlinear structure, particularly in the presence of external disturbances. In this study, the objective is to train a neural controller such that the lander system equipped with this controller closely tracks the classical Apollo controller presented in Reference [42]. Owing to the inherent nonlinearity of this baseline Apollo system, it is not suitable to be directly employed as a reference model. Consequently, an appropriate linear approximation is required. To this end, we adopt the Five Tau rule and a regression-based linearization approach. The Five Tau rule states that transient dynamics decay after approximately five time constants, implying that transitions between steady states are completed within this duration. Meanwhile, the regression technique is utilized to obtain the best linear approximation of the nonlinear aerodynamic dynamics described in Equation (93). Accordingly, we consider the following form of the reference model:

$$\begin{bmatrix} \frac{d\hat{r}}{dt} \\ \frac{d\hat{v}}{dt} \end{bmatrix} = \begin{bmatrix} \hat{v}(t) \\ q\hat{v}(t) + \hat{u}(t) \end{bmatrix} \tag{101}$$

where

$$q = \begin{pmatrix} q_x & 0 & 0 \\ 0 & q_y & 0 \\ 0 & 0 & q_z \end{pmatrix}$$

MATLAB is used to find the matrix a using a linear regression technique:

$$q = \arg\min_{v} \left(\frac{1}{2} \frac{\rho}{m} v^2 C_D(\mathrm{Ma}) S \; - qv(t) \right) \tag{102}$$

As a result, we obtain the matrix a as follows:

$$a = \begin{pmatrix} -0.0087 & 0 & 0 \\ 0 & -0.0075 & 0 \\ 0 & 0 & -0.0077 \end{pmatrix} \tag{103}$$

The linear reference model is now complete, and we can move on to finding the best controller that has dynamical behavior too close to the nonlinear classical one. MATLAB is used to implement the pole placement technique (state feedback control) along with the Five Tau rule. Figure 14 illustrates the comparison between the output of the nonlinear Apollo Lander system and

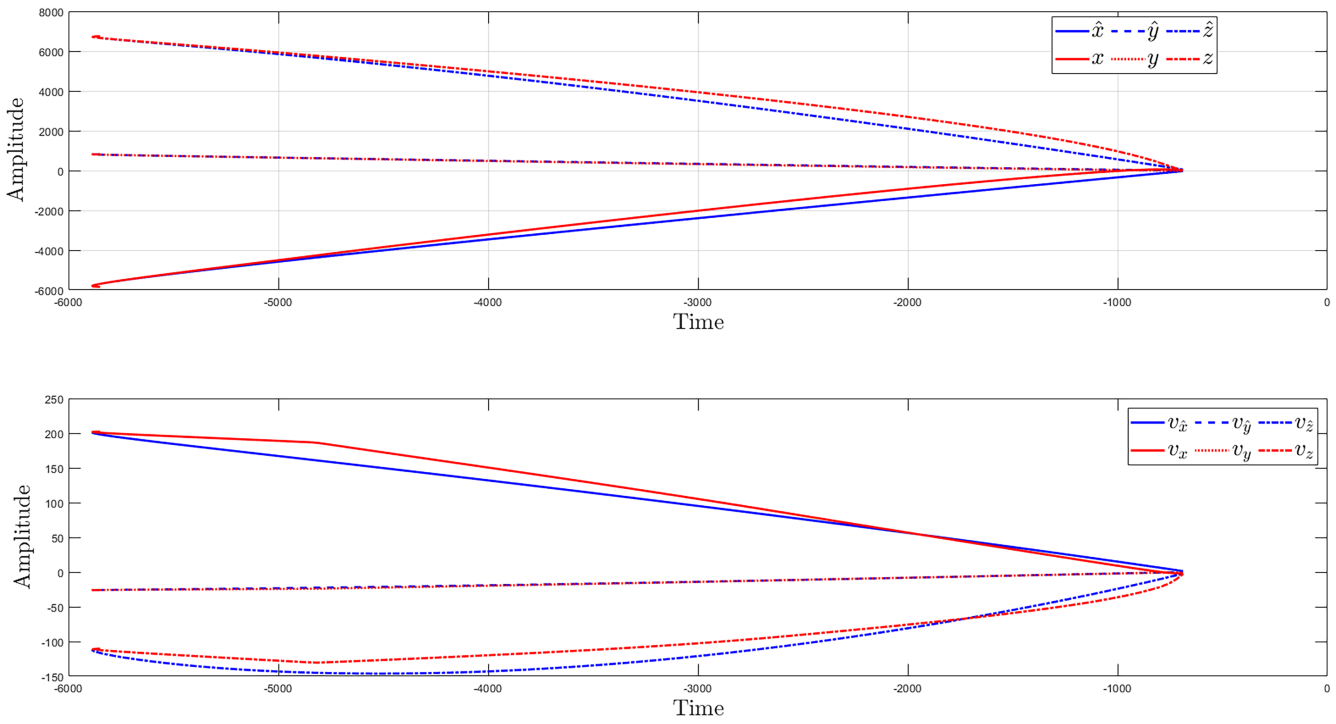

FIGURE 14 | Comparison between the linear reference model (indicated in blue) constructed using the five-τ rule and the nonlinear Apollo Lander system employed for neural network training (indicated in red).

TABLE 2 | Neural Controller Architecture.

Layer	No. neurons/activation function
Input layer	6/Linear $[r^T(t), v^T(t)]^T$
1^{st} layer	40/Tanh
2^{nd} layer	40/Sigmoid
3^{rd} layer	40/Tanh
Output layer	3/Linear

the corresponding linear reference model for one of the scenarios. As can be observed, despite the strong nonlinearities present in the lander dynamics, the trajectories of both systems remain sufficiently close over the considered operating range. This demonstrates that the linear reference model provides an accurate representation of the desired closed-loop behavior and captures the dominant system dynamics relevant for control and verification purposes. This observation justifies the use of the linear reference model within the proposed keep-close framework.

The neural controller in Table 2 was trained with the aim that the lander dynamics with this neural controller track the closed-loop reference Apollo model. This means that the dynamic behavior of the system in the closed-loop system after the training of the neural controller is close to the linear reference closed-loop dynamics. In an ideal world, the closed-loop Apollo model would be matched by the system with an NN controller at all times t. In reality, this is almost impossible. In this example, we are calculating the worst relative squared difference, γ, between the classical Apollo system provided in Reference [42] with a neural control system closing the loop, while subject to its nonlinear functions, and the ideal closed-loop reference model. For this purpose, let assume the static nonlinearity $\Delta_\delta(v(t)) = \frac{1}{2}\frac{\rho}{m}v^2(t)C_D(\text{Ma})S - qv(t)$ which is slope-restricted, and sector bounded. The Lander dynamics are then rearranged into the following form:

$$\begin{cases} \begin{pmatrix} \frac{dr}{dt} \\ \frac{dv}{dt} \end{pmatrix} = \begin{bmatrix} v \\ qv(t) + u(t) + \Delta_\delta(v(t)) \end{bmatrix} \\ \Delta_\delta(v(t)) = \frac{1}{2}\frac{\rho}{m}v^2(t)C_D(\text{Ma})S - av(t) \\ y(t) = \begin{pmatrix} r(t) \\ v(t) \end{pmatrix} \end{cases} \tag{104}$$

The neural controller defines the nonlinear mapping

$$\begin{aligned} u(t) &= \pi(r(t), v(t)) \\ &= W_4\phi_3\left(W_3\phi_2\left(W_2\phi_1\left(W_1\begin{bmatrix} r(t) \\ v(t) \end{bmatrix} + b_1\right) + b_2\right) + b_3\right) + b_4, \end{aligned} \tag{105}$$

where $\phi_1 = \tanh$, $\phi_2 = \text{sigmoid}$, and $\phi_3 = \tanh$.

6.2.3 | Step 3: Error Dynamical System

By using the closed-loop reference model and the new mathematical form of the Lander dynamical motion, we can now construct the error dynamical system. Let $\delta r = r - \hat{r}$, $\delta v = v - \hat{v}$, and $\delta u = u - \hat{u}$, then

$$\begin{cases} \begin{pmatrix} \frac{d\delta r}{dt} \\ \frac{d\delta v}{dt} \end{pmatrix} = \begin{bmatrix} \delta v \\ q\delta v(t) + \delta u(t) + \Delta_\delta(v(t)) \end{bmatrix} \\ \Delta_\delta(v(t)) = \frac{1}{2}\frac{\rho}{m}v^2(t)C_D(\text{Ma})S - qv(t) \\ z(t) = \begin{pmatrix} \delta r(t) \\ \delta v(t) \end{pmatrix} \end{cases} \tag{106}$$

According to Lemma 3, the approximation error $\delta u = u - \hat{u}$ given in the equation above can be expressed as:

$$\delta u(t) = \Lambda(s)\zeta(t) + \frac{\partial\alpha}{\partial\hat{\eta}}(0,0)\hat{\eta}(t) + \epsilon(t), \tag{107}$$

where $\hat{\eta} = [\hat{r}^T \ \hat{v}^T]^T$, Also, we notice that in this case we don't have a reference signal, that is, $d(t) = 0$, the LPV matrix $\Lambda(s(t)) = \nabla_s u(s)$ is the Jacobian matrix of the neural network (105) at point $s \in \mathbb{C} \rtimes (\hat{\eta}, \eta, \ \epsilon(t) = \mathcal{O}(\hat{\eta}(t))$ and the training error $\alpha(\hat{\eta}(t), d(t) = 0) = u(\hat{\eta}(t)) - \hat{u}\ (\hat{\eta}(t))$. Then, the dynamical error became:

$$\begin{cases} \begin{bmatrix} \frac{d\delta r}{dt}(t) \\ \frac{d\delta v}{dt}(t) \end{bmatrix} = \begin{bmatrix} \delta v(t) \\ q\delta v(t) + \Delta_\delta(v(t)) + \Delta_\epsilon(\hat{v}(t)) + \frac{\partial\alpha}{\partial\hat{\eta}}(0,0)\hat{\eta}(t) \end{bmatrix} + B\Lambda(s)\begin{bmatrix} \delta r(t) \\ \delta v(t) \end{bmatrix}, \\ \Delta_\delta(v(t)) = \frac{1}{2}\frac{\rho}{m}v^2(t)C_D(\text{Ma})S - qv(t), \ \epsilon((t)) = \mathcal{O}(\hat{\eta}(t)) \\ z(t) = C\begin{pmatrix} \delta r(t) \\ \delta v(t) \end{pmatrix}, \quad B = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \end{cases} \tag{108}$$

6.2.4 | Step 4: IQCs Characterization of the Nonlinearities Δ_δ and Δ_ϵ

To choose the type of IQCs of the uncertainties considered for this example, let start with the static nonlinearity $\Delta_\delta(v(t)) = \frac{1}{2}\frac{\rho}{m}v^2(t)C_D(\text{Ma})S - qv(t)$. It is clear from the drawn Figure 15 that the uncertainty $\Delta_\delta(v(t))$ is slope-restricted and sector-bounded. Figure 15 provides the IQCs characterization of this nonlinearity and plots the graph $\Delta_\delta(v(t))$ where $\Delta_\delta(v(t)) = \varphi(v,t)$ and $\tilde{\alpha}v(t) \le \varphi(v,t) \le \tilde{\beta}v(t)$ for all v. Simple calculation of the slope gives $\tilde{\alpha} = \textbf{diag}(-0.003, 0.013, 0.006)$ and $\beta = \textbf{diag}(0.008, -0.004, -0.004)$ representing the bounds of the uncertainty Δ_δ.

Similarly, we have the nonlinearity $\epsilon(\hat{v}(t)) = u(\hat{v}(t)) - \hat{u}\ (\hat{v}(t))$. The graph of $\epsilon = \hat{f}(\hat{v}(t))$ is shown in Figure 16 where $\hat{\alpha}\hat{v}(t) \le \hat{f}(\hat{v}(t)) \le \hat{\beta}\hat{v}(t)$. The IQCs characterization of this nonlinearity is thus conducted to in $\hat{\beta} = \textbf{diag}(-0.052, 0, 0.0053)$ and $\hat{\alpha} = \textbf{diag}(0.022, -0.024, -0.0043)$.

6.2.5 | Step 5: IQCs-Based Robustness Analysis Results

According to the results given by IQClab Toolbox for the model corresponding to the original Apollo Robustness Control Problem:

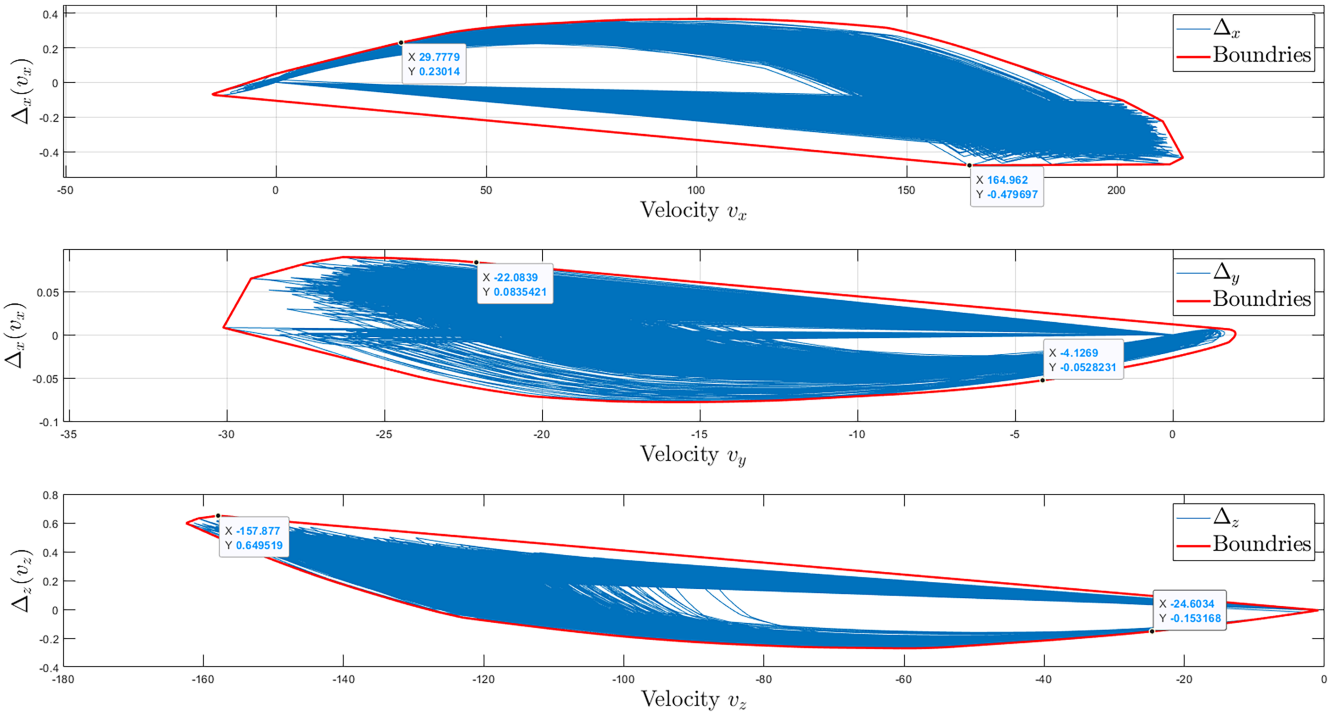

FIGURE 15 | IQCs characterization of the nonlinearity Δ_δ.

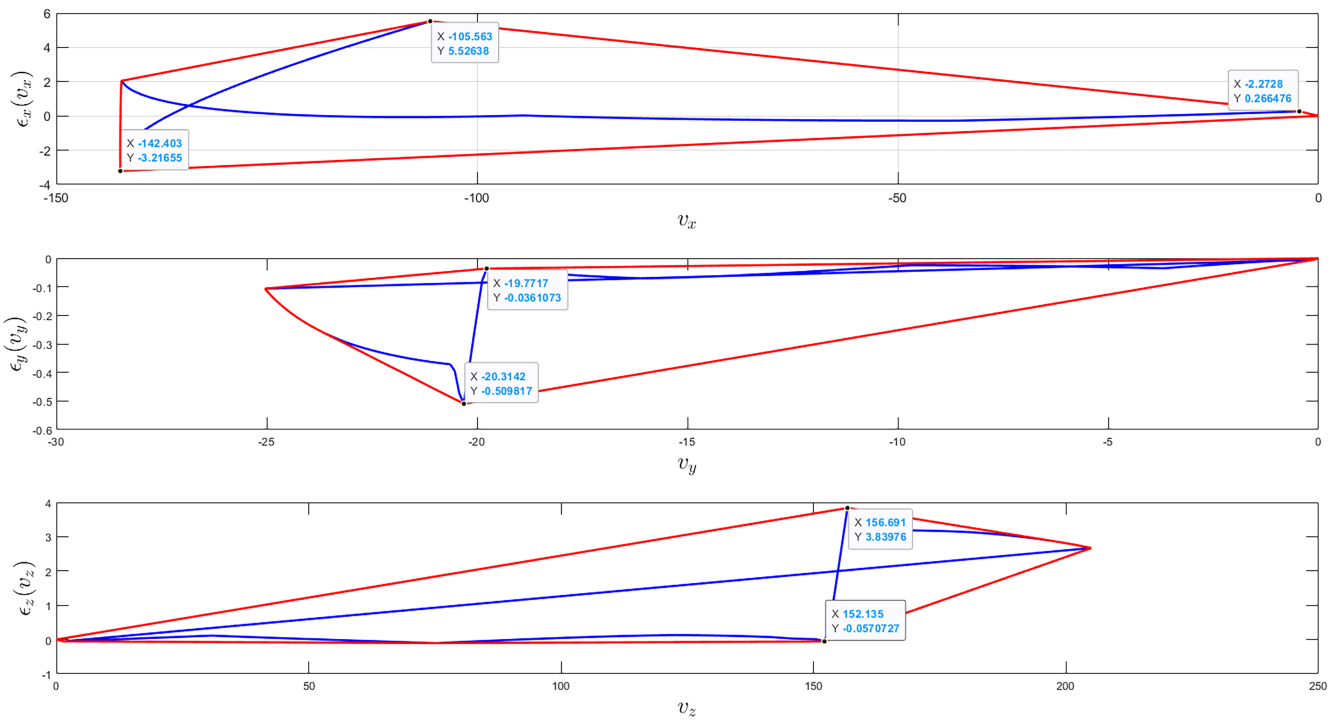

FIGURE 16 | IQCs characterization of the nonlinearity Δ_ϵ.

$$\begin{bmatrix} \gamma_x & \gamma_y & \gamma_z & \gamma_{v_x} & \gamma_{v_y} & \gamma_{v_z} \end{bmatrix}^T = \begin{bmatrix} 0.0988 & 0.1992 & 0.1581 & 0.0119 & 0.0244 & 0.0158 \end{bmatrix}^T$$

The worst relative error between the closed-loop system with the trained neural controller and the closed-loop reference system is less than $\begin{bmatrix} 9.88\% & 19.92\% & 15.81\% & 1.19\% & 2.44\% & 1.58\% \end{bmatrix}^T$ of the reference model outputs. To further illustrate the results of our controlled Lander with our designed neural controller, we perform some simulation results. Figures 17 and 18 show the response of the system with the trained NN controller for two different scenarios with different initial positions and velocities. Clearly, the output remains inside the expected bound region.

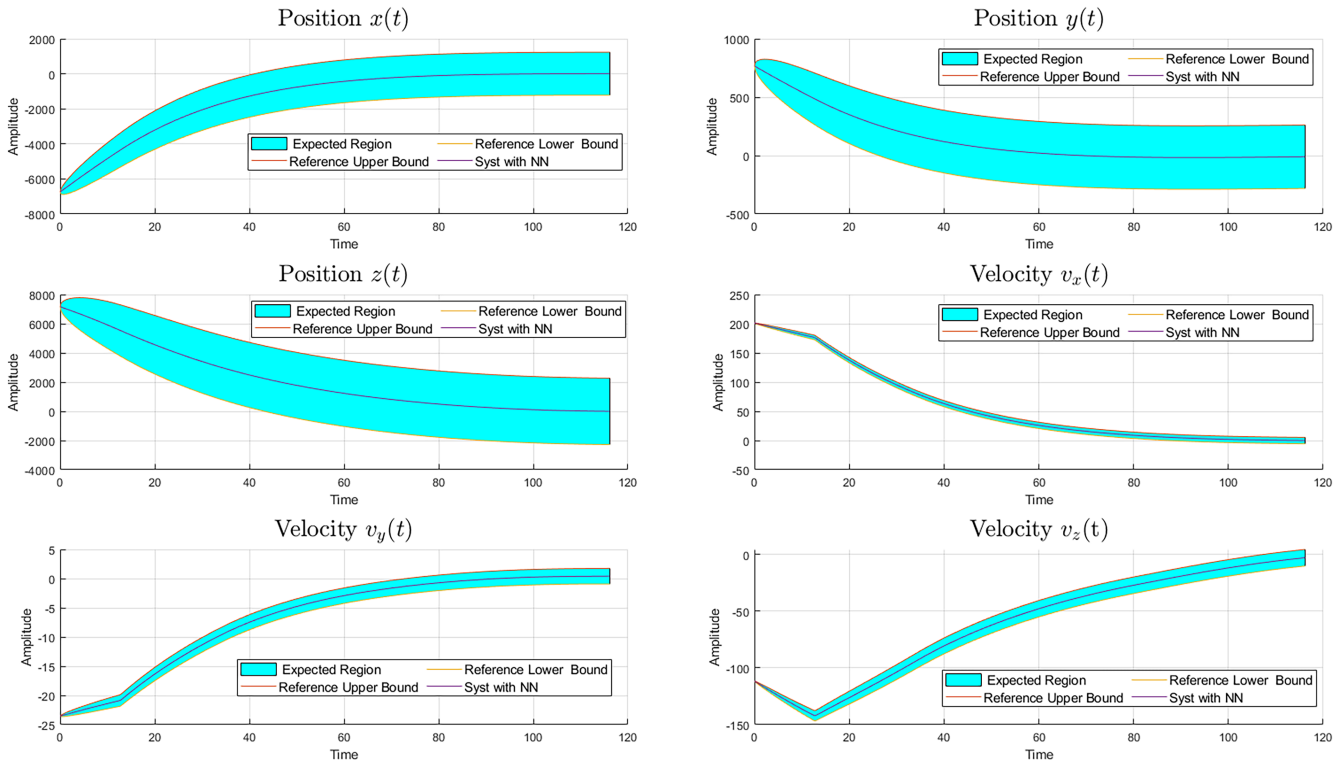

FIGURE 17 | Simulation Results, Successful Landing Scenario.

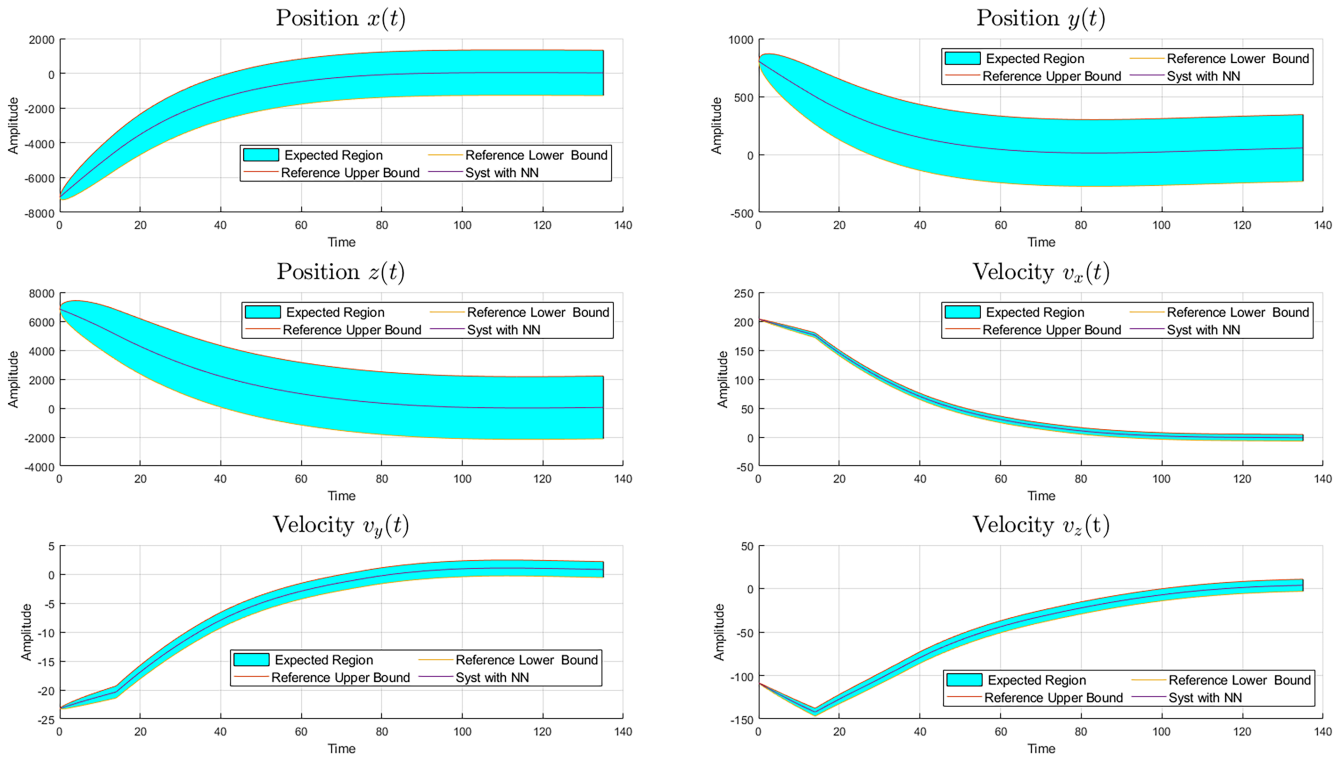

FIGURE 18 | Simulation Results, Failed Landing Scenario.

For further validation of the proposed keep-close framework, we performed a Monte Carlo study in Figure 19 consisting of 100 independent landing scenarios. In these simulations, the initial horizontal position in the x-direction was selected beyond 8 km from the landing target, while the initial y-position was approximately 4 km. The worst-case gain bounds obtained from the theoretical analysis were $\gamma_x = 0.0988$ and $\gamma_y = 0.1992$. Since $\gamma_x \ll \gamma_y$, the deviation accumulation in the y-direction is expected to

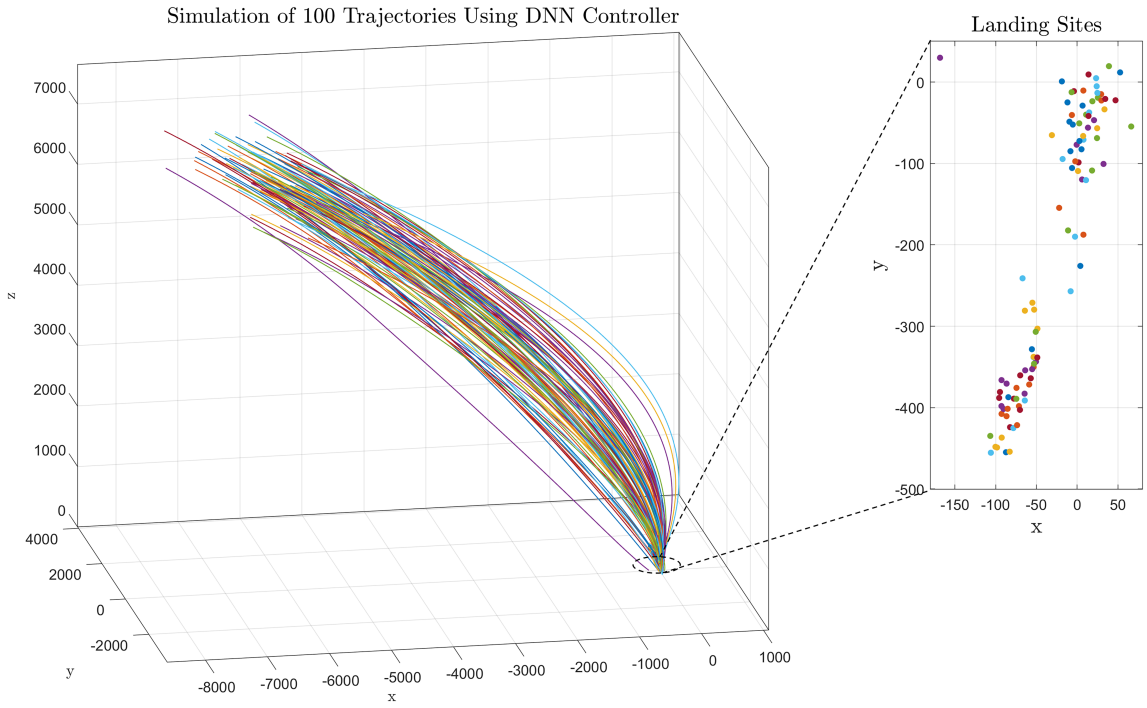

FIGURE 19 | IQCs characterization of the nonlinearity Δ_ϵ.

grow faster than in the x-direction under comparable uncertainty levels.

The simulation results confirm this behavior. Despite large initial offsets, the final landing site errors remain bounded within

$$-170 \text{ m} < x_{\text{error}} < 70 \text{ m}$$

for initial conditions exceeding 8 km in x, and

$$-460 \text{ m} < y_{\text{error}} < 90 \text{ m}$$

for initial conditions around 4 km in y, across all 100 Monte Carlo scenarios. These results are consistent with the theoretical gain bounds and demonstrate that the keep-close framework provides reliable and predictive robustness certificates for the NN-controlled landing system under substantial initial deviations and uncertainties. Clearly, the estimated worst-case measure serves as a crucial indicator of the potential occurrence of undesirable behavior within the closed-loop system when employing the designed NN controller.

7 | Conclusion

A novel technique, termed "keep-close," was introduced to ensure the safety and robustness of feedback systems equipped with neural network (NN) controllers amidst various types of uncertainties. The theoretical findings provided for worst-case analysis offer users preliminary insights into the maximum expected error between the reference model and the system's state when controlled by an NN amidst perturbations. The central concept behind this new methodology is to preserve the closed-loop system's output under an NN controller close to that of a robustly performing reference model, even when its input varies within a predefined bounded set. The "keep-close" strategy furnishes users with advanced knowledge regarding the worst-case rise and steady-state error (SSE) between the reference closed-loop model and the uncertain system managed by the neural controller. This insight is crucial for validating AI-based machine learning (ML) techniques through robust control theory, thereby enhancing user trust in these technologies. The efficacy and adaptability of this innovative approach have been empirically validated through simulation implementation and testing on two distinct challenges: The Single-Link Robot-Arm Control Problem and the Deep Guidance and Control of the Apollo Lander. These real-world problems have demonstrated the proposed method's effectiveness and flexibility across different application areas, offering empirical proof of its utility and adaptability.

Although the proposed framework establishes rigorous guarantees for a broad class of continuous-time NN-controlled systems, several limitations remain. First, the current analysis is developed within a continuous-time setting and relies on continuous-time Lyapunov theory and IQC-based dissipation inequalities. Consequently, the present results do not explicitly account for implementation effects arising in digital and sampled-data control architectures, such as sampling, quantization, and computational delays. Extending the keep-close methodology to discrete-time and sampled-data systems, together with the development of corresponding discrete-time IQC formulations and performance certificates, constitutes an important direction for future research. Second, the proposed analysis relies on the Differential Mean Value theorem to characterize the neural controller error and therefore assumes that the NN mapping is continuously differentiable over the operating region of interest. While this assumption is satisfied by many commonly used activation functions, such as sigmoid and hyperbolic tangent

functions, it does not directly cover non-smooth activation functions such as ReLU and Leaky-ReLU, which are widely adopted in modern deep learning architectures. Extending the keep-close framework to accommodate such non-smooth NN controllers through hybrid-system or switched-system formulations, together with appropriate IQC- and Lyapunov-based verification tools, represents another promising avenue for future investigation.

Acknowledgments

The authors express their gratitude to Spin-Works for their support and to Tiago Amaral for providing the Apollo software used in data generation. Additionally, we extend our appreciation to Joost Veenman from SENER Aeroespacial, Tres Cantos, Spain, for granting access to the IQClab Toolbox, which was instrumental in this research. Furthermore, we sincerely thank Samir Bennani and David Sanchez de la Llana from the European Space Research and Technology Centre (ESA), 2201 AZ Noordwijk, The Netherlands, for their valuable feedback and support.

Funding

This work was supported by the European Space Agency (ESA), AITIVE-GNC Project, ESA Contract No. 4000133980/21/NL/CRS.

Conflicts of Interest

The authors declare no conflicts of interest.

Data Availability Statement

Data sharing is not applicable to this article as no datasets were generated or analyzed during the current study.

Endnotes

[1]https://www.iqclab.eu.

[2]https://uk.mathworks.com/help/deeplearning/ug/design-model-reference-neural-controller-in-simulink.html.

References

1. J. Schulman, S. Levine, P. Abbeel, M. Jordan, and P. Moritz, "Trust Region Policy Optimization," in *International Conference on Machine Learning* (PMLR, 2015).

2. M. Fazlyab, M. Morari, and G. J. Pappas, "Safety Verification and Robustness Analysis of Neural Networks via Quadratic Constraints and Semidefinite Programming," *IEEE Transactions on Automatic Control* 67 (2020): 1–15.

3. S. A. Deka, D. M. Stipanović, and C. J. Tomlin, "Dynamically Computing Adversarial Perturbations for Recurrent Neural Networks," *IEEE Transactions on Control Systems Technology* 30 (2022): 2615–2629.

4. H. Yin, P. Seiler, and M. Arcak, "Stability Analysis Using Quadratic Constraints for Systems With Neural Network Controllers," *IEEE Transactions on Automatic Control* 67 (2021): 1980–1987.

5. C. Szegedy, W. Zaremba, I. Sutskever, et al., "Intriguing Properties of Neural Networks," arXiv preprint, arXiv:13126199 (2013).

6. S. Zheng, Y. Song, T. Leung, and I. Goodfellow, "Improving the Robustness of Deep Neural Networks via Stability Training," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (IEEE, 2016), 4480–4488.

7. H. K. Khalil, *Nonlinear Systems*, 3rd ed. (Prentice-Hall, 2002).

8. W. Xiang, P. Musau, A. A. Wild, et al., "Verification for Machine Learning, Autonomy, and Neural Networks Survey," arXiv preprint, arXiv:181001989 (2018).

9. B. Zhao, S. Zhang, and D. Liu, "Self-Triggered Approximate Optimal Neuro-Control for Nonlinear Systems Through Adaptive Dynamic Programming," *IEEE Transactions on Neural Networks and Learning Systems* 36, no. 3 (2024): 4713–4723.

10. F. Zhang and G. H. Yang, "Adaptive Safety-Certified Reinforcement Learning for Constrained Optimal Control of Autonomous Robots With Uncertainties," *IEEE Internet of Things Journal* 12 (2025): 23154–23168.

11. A. Megretski and A. Rantzer, "System Analysis via Integral Quadratic Constraints," *IEEE Transactions on Automatic Control* 42 (1997): 819–830.

12. J. Veenman and C. W. Scherer, "Stability Analysis With Integral Quadratic Constraints: A Dissipativity Based Proof," in *52nd IEEE Conference on Decision and Control IEEE* (IEEE, 2013).

13. L. Lessard, B. Recht, and A. Packard, "Analysis and Design of Optimization Algorithms via Integral Quadratic Constraints," *SIAM Journal on Optimization* 26, no. 1 (2016): 57–95.

14. J. Veenman, C. W. Scherer, and H. Koroglu, "Robust Stability and Performance Analysis Based on Integral Quadratic Constraints," *European Journal of Control* 31 (2016): 1–32.

15. R. Wang and I. R. Manchester, "Robust Contraction Analysis of Nonlinear Systems via Differential IQC," in *2019 IEEE 58th Conference on Decision and Control (CDC)* (IEEE, 2019).

16. A. Iannelli, P. Seiler, and A. Marcos, "Region of Attraction Analysis With Integral Quadratic Constraints," *Automatica* 109 (2019): 108543.

17. M. Fetzer, C. W. Scherer, and J. Veenman, "Invariance With Dynamic Multipliers," *IEEE Transactions on Automatic Control* 63, no. 7 (2017): 1929–1942.

18. A. Rantzer, "On the Kalman-Yakubovich-Popov Lemma," *Systems & Control Letters* 28 (1996): 7–10.

19. E. Summers and A. Packard, "$\mathcal{L}_2$ Gain Verification for Interconnections of Locally Stable Systems Using Integral Quadratic Constraints," in *49th IEEE Conference on Decision and Control (CDC)* (IEEE, 2010).

20. W. Xiang, H. D. Tran, J. A. Rosenfeld, and T. T. Johnson, "Reachable Set Estimation and Safety Verification for Piecewise Linear Systems With Neural Network Controllers," in *2018 Annual American Control Conference (ACC)* (IEEE, 2018).

21. R. Ivanov, T. J. Carpenter, J. Weimer, R. Alur, G. J. Pappas, and I. Lee, "Verifying the Safety of Autonomous Systems With Neural Network Controllers," *ACM Transactions on Embedded Computing Systems* 20, no. 1 (2020): 1–26.

22. R. Ivanov, T. Carpenter, J. Weimer, R. Alur, G. Pappas, and I. Lee, "Verisig 2.0: Verification of Neural Network Controllers Using Taylor Model Preconditioning," in *International Conference on Computer Aided Verification* (Springer, 2021).

23. C. Qin, K. Jiang, Y. Wang, T. Zhu, Y. Wu, and D. Zhang, "Event-Triggered H_∞ Control for Unknown Constrained Nonlinear Systems With Application to Robot Arm," *Applied Mathematical Modelling* 144 (2025): 116089.

24. C. Qin, X. Qiao, J. Wang, D. Zhang, Y. Hou, and S. Hu, "Barrier-Critic Adaptive Robust Control of Nonzero-Sum Differential Games for Uncertain Nonlinear Systems With State Constraints," *IEEE Transactions on Systems, Man, and Cybernetics: Systems* 54, no. 1 (2023): 50–63.

25. C. Qin, T. Zhu, K. Jiang, and Y. Wu, "Integral Reinforcement Learning-Based Dynamic Event-Triggered Safety Control for

Multiplayer Stackelberg–Nash Games With Time-Varying State Constraints," *Engineering Applications of Artificial Intelligence* 133 (2024): 108317.

26. P. T. Nam, T. N. Nguyen, and H. Trinh, "Comparison Principle for Positive Time-Delay Systems: An Extension and Its Application," *Journal of the Franklin Institute* 358, no. 13 (2021): 6818–6834.

27. A. Zenati, T. M. Laleg-Kirati, M. Tadjine, and N. Aouf, "On the Admissibility and Stability of Multi-Agent Nonlinear Interconnected Positive Systems With Heterogeneous Delays," *IEEE Transactions on Automatic Control* 68 (2022): 5775–5782.

28. H. Pfifer and P. Seiler, "Robustness Analysis of Linear Parameter Varying Systems Using Integral Quadratic Constraints," *International Journal of Robust and Nonlinear Control* 25, no. 15 (2015): 2843–2864.

29. E. Millour, F. Forget, A. Spiga, et al., "The Mars Climate Database (MCD Version 5.2)," in *European Planetary Science Congress*, vol. 10 (Copernicus Publications, 2015), 2015–2438.

30. A. Zemouche, M. Boutayeb, and G. I. Bara, "Observer Design for Nonlinear Systems: An Approach Based on the Differential Mean Value Theorem," in *Proceedings of the 44th IEEE Conference on Decision and Control IEEE* (IEEE, 2005).

31. V. Sharma, V. Agrawal, B. B. Sharma, and R. Nath, "Unknown Input Nonlinear Observer Design for Continuous and Discrete Time Systems With Input Recovery Scheme," *Nonlinear Dynamics* 85 (2016): 645–658.

32. S. S. Kelkar, L. L. Grigsby, and J. Langsner, "An Extension of Parseval's Theorem and Its Use in Calculating Transient Energy in the Frequency Domain," *IEEE Transactions on Industrial Electronics* 1 (1983): 42–45.

33. M. Fetzer and C. W. Scherer, "Zames–Falb Multipliers for Invariance," *IEEE Control Systems Letters* 1, no. 2 (2017): 412–417.

34. J. Veenman and C. W. Scherer, "IQC-Synthesis With General Dynamic Multipliers," *International Journal of Robust and Nonlinear Control* 24, no. 17 (2014): 3027–3056.

35. M. Fu, S. Dasgupta, and Y. C. Soh, "Integral Quadratic Constraint Approach vs. Multiplier Approach," *Automatica* 41, no. 2 (2005): 281–287.

36. M. Jun and M. G. Safonov, "Rational Multiplier IQCs for Uncertain Time-Delays and LMI Stability Conditions," *IEEE Transactions on Automatic Control* 47, no. 11 (2002): 1871–1875.

37. J. Veenman, C. W. Scherer, C. Ardura, S. Bennani, V. Preda, and B. Girouart, "IQClab: A New IQC Based Toolbox for Robustness Analysis and Control Design," *IFAC-PapersOnLine* 54, no. 8 (2021): 69–74.

38. H. Fang, Z. Lin, and M. Rotea, "On IQCs Approach to the Analysis and Design of Linear Systems Subject to Actuator Saturation," *Systems & Control Letters* 57, no. 8 (2008): 611–619.

39. C. Y. Kao, A. Megretski, and U. Jönsson, "Specialized Fast Algorithms for IQC Feasibility and Optimization Problems," *Automatica* 40, no. 2 (2004): 239–252.

40. A. G. Atkins, T. Atkins, and M. Escudier, *A Dictionary of Mechanical Engineering* (Oxford University Press, 2013).

41. A. Megretski, C. Kao, U. Jonsson, and A. Rantzer, "A Guide to iqc-β: A MATLAB Toolbox for Robust Stability and Performance Analysis," Technical Report, MIT (2004).

42. T. Amaral, Guidance Optimization for Mars Pinpoint Landing With Optimal Trigger and Re-Optimization 2019.

43. E. Millour, F. Forget, A. Spiga, et al., The Mars Climate Database (MCD Version 5.2) 2015 European Planetary Science Congress.

44. A. R. Klumpp, "Apollo Lunar Descent Guidance," *Automatica* 10, no. 2 (1974): 133–146.

45. X. Huang, C. Xu, J. Hu, et al., "Powered-Descent Landing GNC System Design and Flight Results for Tianwen-1 Mission," *Astrodynamics* 6, no. 1 (2022): 3–16.

Appendix A: Extended System Derivation

Proof. First, let us recall the definition of variables:

$$\eta(t) = \begin{bmatrix} d(t) \\ \hat{x}(t) \end{bmatrix}, \quad q(t) = \begin{bmatrix} \delta(t) \\ \epsilon(t) \end{bmatrix}.$$

Now, we start with the error dynamic equation in Equation (22a):

$$\dot{\zeta}(t) = A\zeta(t) + B\mu(t) + \tilde{B}\delta(t), \tag{A1}$$

where $\zeta(0) = 0$. Also, the controller error $\mu(t)$ from Equation (24) is given by:

$$\mu(t) = \Lambda(s)\zeta(t) + \frac{\partial \alpha}{\partial \hat{x}}(0,0)\hat{x}(t) + \frac{\partial \alpha}{\partial d}(0,0)d(t) + \epsilon(t). \tag{A2}$$

Substitute $\mu(t)$ into $\dot{\zeta}(t)$ Substituting $\mu(t)$ from Equation (A2) into the first equation of (A1), we have:

$$\dot{\zeta}(t) = A\zeta(t) + B\left[\Lambda(s)\zeta(t) + \frac{\partial \alpha}{\partial \hat{x}}(0,0)\hat{x}(t) + \frac{\partial \alpha}{\partial d}(0,0)d(t) + \epsilon(t)\right] + \tilde{B}\delta(t), \tag{A3}$$

$$= (A + B\Lambda(s))\zeta(t) + B\begin{bmatrix} \frac{\partial \alpha}{\partial d}(0,0) & \frac{\partial \alpha}{\partial \hat{x}}(0,0) \end{bmatrix}\begin{bmatrix} d(t) \\ \hat{x}(t) \end{bmatrix} + B\epsilon(t) + \tilde{B}\delta(t), \tag{A4}$$

$$= (A + B\Lambda(s))\zeta(t) + \begin{bmatrix} B\frac{\partial \alpha}{\partial d}(0,0) & B\frac{\partial \alpha}{\partial \hat{x}}(0,0) \end{bmatrix}\begin{bmatrix} d(t) \\ \hat{x}(t) \end{bmatrix} + \begin{bmatrix} \tilde{B} & B \end{bmatrix}\begin{bmatrix} \delta(t) \\ \epsilon(t) \end{bmatrix}, \tag{A5}$$

$$= (A + B\Lambda(s))\zeta(t) + \begin{bmatrix} B\frac{\partial \alpha}{\partial d}(0,0) & B\frac{\partial \alpha}{\partial \hat{x}}(0,0) \end{bmatrix}\eta(t) + \begin{bmatrix} \tilde{B} & B \end{bmatrix}q(t). \tag{A6}$$

Now, we move to the output in Equation (22b), which is given by:

$$z(t) = C\zeta(t) + D\mu(t) + \tilde{D}\delta(t), \tag{A7}$$

Similarly, substituting $\mu(t)$ from Equation (A2) into $z(t)$ in Equation (A7), we have:

$$z(t) = C\zeta(t) + D\left[\Lambda(s)\zeta(t) + \frac{\partial \alpha}{\partial \hat{x}}(0,0)\hat{x}(t) + \frac{\partial \alpha}{\partial d}(0,0)d(t) + \epsilon(t)\right] + \tilde{D}\delta(t), \tag{A8}$$

$$= (C + D\Lambda(s))\zeta(t) + D\begin{bmatrix} \frac{\partial \alpha}{\partial d}(0,0) & \frac{\partial \alpha}{\partial \hat{x}}(0,0) \end{bmatrix}\begin{bmatrix} d(t) \\ \hat{x}(t) \end{bmatrix} + D\epsilon(t) + \tilde{D}\delta(t), \tag{A9}$$

$$= (C + D\Lambda(s))\zeta(t) + \begin{bmatrix} D\frac{\partial \alpha}{\partial d}(0,0) & D\frac{\partial \alpha}{\partial \hat{x}}(0,0) \end{bmatrix}\begin{bmatrix} d(t) \\ \hat{x}(t) \end{bmatrix} + \begin{bmatrix} \tilde{D} & D \end{bmatrix}\begin{bmatrix} \delta(t) \\ \epsilon(t) \end{bmatrix}, \tag{A10}$$

$$= (C + D\Lambda(s))\zeta(t) + \begin{bmatrix} D\frac{\partial \alpha}{\partial d}(0,0) & D\frac{\partial \alpha}{\partial \hat{x}}(0,0) \end{bmatrix}\eta(t) + \begin{bmatrix} \tilde{D} & D \end{bmatrix}q(t). \tag{A11}$$

Using (A6) and (A11), we can rewrite the system (22) and the new system becomes:

$$\Sigma^{\eta,q}_{\tilde{F}(\tilde{P},\Delta_\delta)} := \begin{cases} \dot{\zeta}(t) = (A + B\Lambda(s))\zeta(t) + \begin{bmatrix} B\frac{\partial \alpha}{\partial d}(0,0) & B\frac{\partial \alpha}{\partial \hat{x}}(0,0) \end{bmatrix}\eta(t) + \begin{bmatrix} \tilde{B} & B \end{bmatrix}q(t), \\ z(t) = (C + D\Lambda(s))\zeta(t) + \begin{bmatrix} D\frac{\partial \alpha}{\partial d}(0,0) & D\frac{\partial \alpha}{\partial \hat{x}}(0,0) \end{bmatrix}\eta(t) + \begin{bmatrix} \tilde{D} & D \end{bmatrix}q(t), \\ \zeta(0) = 0. \end{cases} \tag{A12}$$

Now will try to eliminate the input of the uncertainties $\Delta_\delta(.)$ and $\Delta_\epsilon(.)$ from 'virtual' filters in Equations (43) and (44), respectively. From Equation (43), the 'virtual' filter equation is given by

$$\begin{cases} \dot{\psi}_\delta(t) = A_\delta \psi_\delta(t) + B_{\delta 1}\delta(t) + B_{\delta 2} y(t), \\ r_\delta(t) = C_\delta \psi_\delta(t) + D_{\delta 1}\delta(t) + D_{\delta 2} y(t), \\ \psi_\delta(0) = 0, \end{cases} \tag{A13}$$

Knowing that $y(t) = z(t) + \hat{y}(t)$, therefore, substituting $y(t)$ into the system, we get:

$$\dot{\psi}_\delta(t) = A_\delta \psi_\delta(t) + B_{\delta 1}\delta(t) + B_{\delta 2}(z(t) + \hat{y}(t)), \tag{A14}$$

$$r_\delta(t) = C_\delta \psi_\delta(t) + D_{\delta 1}\delta(t) + D_{\delta 2}(z(t) + \hat{y}(t)). \tag{A15}$$

From the earlier results in Equation (A12), $z(t)$ is given by:

$$z(t) = (C + D\Lambda(s))\zeta(t) + \begin{bmatrix} D\frac{\partial \alpha}{\partial d}(0,0) & D\frac{\partial \alpha}{\partial \dot{x}}(0,0) \end{bmatrix}\eta(t) + \begin{bmatrix} \tilde{D} & D \end{bmatrix} q(t), \tag{A16}$$

and from the reference system $\Sigma_P^{\pi^*}$ in Equation (21), $\hat{y}(t)$ is:

$$\hat{y}(t) = C_r \hat{x}(t) + D_r d(t). \tag{A17}$$

Substitute $z(t)$ and $\hat{y}(t)$ into $\dot{\psi}_\delta(t)$:

$$\begin{aligned} \dot{\psi}_\delta(t) = {} & A_\delta \psi_\delta(t) + B_{\delta 1}\delta(t) + B_{\delta 2}(C + D\Lambda(s))\zeta(t) \\ & + \begin{bmatrix} D\frac{\partial \alpha}{\partial d}(0,0) + B_{\delta 2} D_r & D\frac{\partial \alpha}{\partial \dot{x}}(0,0) + B_{\delta 2} C_r \end{bmatrix}\eta(t) \\ & + \begin{bmatrix} B_{\delta 2}\tilde{D} & B_{\delta 2} D \end{bmatrix} q(t). \end{aligned} \tag{A18}$$

Similarly, substitute $z(t)$ and $\hat{y}(t)$ into $r_\delta(t)$:

$$\begin{aligned} r_\delta(t) = {} & C_\delta \psi_\delta(t) + D_{\delta 1}\delta(t) + D_{\delta 2}(C + D\Lambda(s))\zeta(t) \\ & + \begin{bmatrix} D\frac{\partial \alpha}{\partial d}(0,0) + D_{\delta 2} D_r & D\frac{\partial \alpha}{\partial \dot{x}}(0,0) + D_{\delta 2} C_r \end{bmatrix}\eta(t) \\ & + \begin{bmatrix} D_{\delta 2}\tilde{D} & D_{\delta 2} D \end{bmatrix} q(t). \end{aligned} \tag{A19}$$

Thus, the virtual filter for the uncertainty Δ_δ is:

$$\begin{cases} \dot{\psi}_\delta(t) = A_\delta \psi_\delta(t) + B_{\delta 2}(C + D\Lambda(s))\zeta(t) \\ \quad + \begin{bmatrix} B_{\delta 2} D\frac{\partial \alpha}{\partial d}(0,0) + B_{\delta 2} D_r & B_{\delta 2} D\frac{\partial \alpha}{\partial \dot{x}}(0,0) + B_{\delta 2} C_r \end{bmatrix}\eta(t) \\ \quad + \begin{bmatrix} B_{\delta 2}\tilde{D} + B_{\delta 1} & B_{\delta 2} D \end{bmatrix} q(t), \\ r_\delta(t) = C_\delta \psi_\delta(t) + D_{\delta 2}(C + D\Lambda(s))\zeta(t) \\ \quad + \begin{bmatrix} D_{\delta 2} D\frac{\partial \alpha}{\partial d}(0,0) + D_{\delta 2} D_r & D_{\delta 2} D\frac{\partial \alpha}{\partial \dot{x}}(0,0) + D_{\delta 2} C_r \end{bmatrix}\eta(t) \\ \quad + \begin{bmatrix} D_{\delta 2}\tilde{D} + D_{\delta 1} & D_{\delta 2} D \end{bmatrix} q(t), \\ \psi_\delta(0) = 0. \end{cases} \tag{A20}$$

For the virtual filter for the uncertainty Δ_ϵ in Equation (44) is given by:

$$\begin{cases} \dot{\psi}_\epsilon(t) = A_\epsilon \psi_\epsilon(t) + B_{\epsilon 1}\epsilon(t) + B_{\epsilon 2}\eta(t), \\ r_\epsilon(t) = C_\epsilon \psi_\epsilon(t) + D_{\epsilon 1}\epsilon(t) + D_{\epsilon 2}\eta(t), \\ \psi_\epsilon(0) = 0, \end{cases} \tag{A21}$$

Using an extended state $\chi(t) = [\zeta(t)\ \psi_\delta\ \psi_\epsilon]^T$ and the equations (A12), (A20), and (A21) we get the extended system in Equation (46). □

Appendix B: Theorems 1 to 2 Complement

Derivation of (52)

Let the stacked signal vector $[\chi(t)^T\ q(t)^T\ \eta(t)^T]$. Since (51) holds for all $s \in \Omega$, pre- and post-multiplying (51) by $\begin{bmatrix} \chi^T & q^T & \eta^T \end{bmatrix}$ and its transpose yields the scalar inequality

$$\begin{aligned} 0 \geq {} & \begin{bmatrix} \chi \\ q \\ \eta \end{bmatrix}^\top \begin{bmatrix} P\mathcal{A}^T + \mathcal{A}P + \partial P & P\mathcal{B}_1 & P\mathcal{B}_2 \\ \mathcal{B}_1^T P & 0 & 0 \\ \mathcal{B}_2^T P & 0 & -(1-\varepsilon)I \end{bmatrix} \begin{bmatrix} \chi \\ q \\ \eta \end{bmatrix} \\ & + \frac{1}{\gamma^2} \begin{bmatrix} \chi \\ q \\ \eta \end{bmatrix}^\top \begin{bmatrix} \mathcal{C}_2^T \\ \mathcal{D}_{21}^T \\ \mathcal{D}_{22}^T \end{bmatrix} \begin{bmatrix} \mathcal{C}_2 & \mathcal{D}_{21} & \mathcal{D}_{22} \end{bmatrix} \begin{bmatrix} \chi \\ q \\ \eta \end{bmatrix} \\ & + \lambda \begin{bmatrix} \chi \\ q \\ \eta \end{bmatrix}^\top \begin{bmatrix} \mathcal{C}_1^T \\ \mathcal{D}_{11}^T \\ \mathcal{D}_{12}^T \end{bmatrix} M \begin{bmatrix} \mathcal{C}_1 & \mathcal{D}_{11} & \mathcal{D}_{12} \end{bmatrix} \begin{bmatrix} \chi \\ q \\ \eta \end{bmatrix}. \end{aligned} \tag{B1}$$

We now identify each term in Equation (B1). Using the error dynamics given in Equation (46) and the storage function $V(\chi, s) = \chi^T P(s)\chi$, the derivative of V along trajectories is

$$\begin{aligned} \dot{V}(\chi, s) &= \nabla_\chi V\ \dot{\chi} + \nabla_s V\ \dot{s} \\ &= \chi^T \left(P\mathcal{A}^T + \mathcal{A}P + \partial P\right)\chi + 2\chi^T P\mathcal{B}_1 q + 2\chi^T P\mathcal{B}_2 \eta. \end{aligned} \tag{B2}$$

Expanding the first quadratic form in Equation (B1) gives

$$\begin{aligned} & \begin{bmatrix} \chi \\ q \\ \eta \end{bmatrix}^\top \begin{bmatrix} P\mathcal{A}^T + \mathcal{A}P + \partial P & P\mathcal{B}_1 & P\mathcal{B}_2 \\ \mathcal{B}_1^T P & 0 & 0 \\ \mathcal{B}_2^T P & 0 & -(1-\varepsilon)I \end{bmatrix} \begin{bmatrix} \chi \\ q \\ \eta \end{bmatrix} \\ & = \chi^T \left(P\mathcal{A}^T + \mathcal{A}P + \partial P\right)\chi + 2\chi^T P\mathcal{B}_1 q + 2\chi^T P\mathcal{B}_2 \eta \\ & \quad - (1-\varepsilon)\eta^T\eta = \dot{V}(\chi, s) - (1-\varepsilon)\eta^T\eta. \end{aligned} \tag{B3}$$

By definition of the performance output of the extended system,

$$z = \mathcal{C}_2\chi + \mathcal{D}_{21} q + \mathcal{D}_{22}\eta, \tag{B4}$$

we have

$$z^T z = \begin{bmatrix} \chi \\ q \\ \eta \end{bmatrix}^\top \begin{bmatrix} \mathcal{C}_2^T \\ \mathcal{D}_{21}^T \\ \mathcal{D}_{22}^T \end{bmatrix} \begin{bmatrix} \mathcal{C}_2 & \mathcal{D}_{21} & \mathcal{D}_{22} \end{bmatrix} \begin{bmatrix} \chi \\ q \\ \eta \end{bmatrix}. \tag{B5}$$

Similarly, using the IQC filter output

$$r = \mathcal{C}_1\chi + \mathcal{D}_{11} q + \mathcal{D}_{12}\eta, \tag{B6}$$

it follows that

$$rMr^T = \begin{bmatrix} \chi \\ q \\ \eta \end{bmatrix}^\top \begin{bmatrix} \mathcal{C}_1^T \\ \mathcal{D}_{11}^T \\ \mathcal{D}_{12}^T \end{bmatrix} M \begin{bmatrix} \mathcal{C}_1 & \mathcal{D}_{11} & \mathcal{D}_{12} \end{bmatrix} \begin{bmatrix} \chi \\ q \\ \eta \end{bmatrix}, \tag{B7}$$

which is consistent with the structural condition in Equation (50).

Substituting (B3) to (B7) into (B1) yields

$$0 \geq \dot{V}(\chi, s) - (1-\varepsilon)\eta^T\eta + \frac{1}{\gamma^2} z^T z + \lambda rMr^T.$$

Rearranging terms gives the dissipation inequality

$$\nabla_\chi V\ \dot{\chi} + \nabla_s V\ \dot{s} = \dot{V}(\chi, s) \leq -\frac{1}{\gamma^2} z^T z + (1-\varepsilon)\eta^T\eta - \lambda rMr^T, \tag{B8}$$

which coincides with (52).

Derivation of (B14)

Since the matrix inequality in Equation (58) is *strict*, there exists a sufficiently small scalar $\varepsilon \in (0,1)$ such that the following *perturbed* inequality also holds for all $(s,\dot{s}) \in \Omega \times \dot{\mathcal{S}}$:

$$\begin{bmatrix} P\mathcal{A}+\mathcal{A}^TP+\partial P & P\mathcal{B}_1 & P\mathcal{B}_2 \\ \mathcal{B}_1^TP & 0 & 0 \\ \mathcal{B}_2^TP & 0 & -(1-\varepsilon)I \end{bmatrix} + \lambda \begin{bmatrix} \mathcal{C}_1^T \\ \mathcal{D}_{11}^T \\ \mathcal{D}_{12}^T \end{bmatrix} M \begin{bmatrix} \mathcal{C}_1 & \mathcal{D}_{11} & \mathcal{D}_{12} \end{bmatrix} \le 0. \tag{B9}$$

Let the signals (χ, z, η, q, r) be generated by the extended system Σ_{ex} in Equation (46), with $\eta \in \mathcal{L}_2([0,\infty))$, $s(t) \in \Omega$, and zero initial conditions. Define the storage function

$$V(\chi, s) = \chi^T P(s)\chi,$$

where $P(s) = P(s)^T \succeq 0$ is continuously differentiable. Along trajectories,

$$\begin{aligned} \dot{V}(\chi,s) &= \nabla_\chi V\ \dot{\chi} + \nabla_s V\ \dot{s} \\ &= \chi^T\left(P\mathcal{A}+\mathcal{A}^TP+\partial P\right)\chi + 2\chi^TP\mathcal{B}_1q + 2\chi^TP\mathcal{B}_2\eta, \end{aligned} \tag{B10}$$

where $\dot{\chi} = \mathcal{A}\chi + \mathcal{B}_1 q + \mathcal{B}_2\eta$.

Evaluating (B9) at $(s(t), \dot{s}(t))$ and pre- and post-multiplying by $\begin{bmatrix}\chi^T & q^T & \eta^T\end{bmatrix}$ and $\begin{bmatrix}\chi^T & q^T & \eta^T\end{bmatrix}^T$, respectively, gives

$$\begin{aligned} 0 \ge &\begin{bmatrix} \chi \\ q \\ \eta \end{bmatrix}^{\mathsf{T}} \begin{bmatrix} P\mathcal{A}+\mathcal{A}^TP+\partial P & P\mathcal{B}_1 & P\mathcal{B}_2 \\ \mathcal{B}_1^TP & 0 & 0 \\ \mathcal{B}_2^TP & 0 & -(1-\varepsilon)I \end{bmatrix} \begin{bmatrix} \chi \\ q \\ \eta \end{bmatrix} \\ &+ \lambda \begin{bmatrix} \chi \\ q \\ \eta \end{bmatrix}^{\mathsf{T}} \begin{bmatrix} \mathcal{C}_1^T \\ \mathcal{D}_{11}^T \\ \mathcal{D}_{12}^T \end{bmatrix} M \begin{bmatrix} \mathcal{C}_1 & \mathcal{D}_{11} & \mathcal{D}_{12} \end{bmatrix} \begin{bmatrix} \chi \\ q \\ \eta \end{bmatrix}. \end{aligned} \tag{B11}$$

Expanding the first quadratic form in Equation (B11) yields

$$\begin{aligned} &\begin{bmatrix} \chi \\ q \\ \eta \end{bmatrix}^{\mathsf{T}} \begin{bmatrix} P\mathcal{A}+\mathcal{A}^TP+\partial P & P\mathcal{B}_1 & P\mathcal{B}_2 \\ \mathcal{B}_1^TP & 0 & 0 \\ \mathcal{B}_2^TP & 0 & -(1-\varepsilon)I \end{bmatrix} \begin{bmatrix} \chi \\ q \\ \eta \end{bmatrix} \\ &= \chi^T\left(P\mathcal{A}+\mathcal{A}^TP+\partial P\right)\chi + 2\chi^TP\mathcal{B}_1q + 2\chi^TP\mathcal{B}_2\eta \\ &\quad - (1-\varepsilon)\eta^T\eta = \dot{V}(\chi,s) - (1-\varepsilon)\eta^T\eta, \end{aligned} \tag{B12}$$

where the last equality follows from Equation (B10).

Moreover, by the IQC filter relation (cf. (13)) one has

$$r = \mathcal{C}_1\chi + \mathcal{D}_{11}q + \mathcal{D}_{12}\eta,$$

and therefore

$$rMr^T = \begin{bmatrix} \chi \\ q \\ \eta \end{bmatrix}^{\mathsf{T}} \begin{bmatrix} \mathcal{C}_1^T \\ \mathcal{D}_{11}^T \\ \mathcal{D}_{12}^T \end{bmatrix} M \begin{bmatrix} \mathcal{C}_1 & \mathcal{D}_{11} & \mathcal{D}_{12} \end{bmatrix} \begin{bmatrix} \chi \\ q \\ \eta \end{bmatrix}, \tag{B13}$$

which is consistent with (50).

Substituting (B12) and (B13) into (B11) gives

$$0 \ge \dot{V}(\chi,s) - (1-\varepsilon)\eta^T\eta + \lambda rMr^T.$$

Rearranging terms yields

$$\nabla_\chi V\ \dot{\chi} + \nabla_s V\ \dot{s} = \dot{V}(\chi,s) \le (1-\varepsilon)\eta^T\eta - \lambda rMr^T, \tag{B14}$$

which is the desired dissipation inequality.