ADAPTIVE CONTROL WITH LSTM AUGMENTATION: THEORY AND HUMAN-IN-THE-LOOP VALIDATION

A THESIS SUBMITTED TO

THE GRADUATE SCHOOL OF ENGINEERING AND SCIENCE

OF BILKENT UNIVERSITY

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR

THE DEGREE OF

MASTER OF SCIENCE

IN

MECHANICAL ENGINEERING

By

Emirhan İnanç

August 2024

Adaptive Control with LSTM Augmentation: Theory and Human-in-the-Loop Validation
By Emirhan İnanç
August 2024

We certify that we have read this thesis and that in our opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.

Yıldıray Yıldız(Advisor)

Ömer Morgül

Mustafa Mert Ankaralı

Approved for the Graduate School of Engineering and Science:

Orhan Arıkan
Director of the Graduate School

ABSTRACT

ADAPTIVE CONTROL WITH LSTM AUGMENTATION: THEORY AND HUMAN-IN-THE-LOOP VALIDATION

Emirhan İnanç
M.S. in Mechanical Engineering
Advisor: Yıldıray Yıldız
August 2024

This thesis presents a novel adaptive control architecture that provides dramatically better transient response performance compared to conventional adaptive control methods. This is accomplished by the synergistic employment of a traditional Adaptive Neural Network (ANN) controller and a Long Short-Term Memory (LSTM) network. LSTM structures can take advantage of the dependencies in an input sequence, which helps predict uncertainty. We introduce a training approach through which the LSTM network learns to compensate for the deficiencies of the ANN controller in a closed-loop setting. This improves the system's transient response and allows the controller to react to unexpected events quickly. This study also investigates the human-in-the-loop performance of the proposed control framework. Although the LSTM-augmented control method drastically improves the transient response, especially in the presence of significant and rapid uncertainty changes, its interactions with a human operator must be analyzed to ensure safe operation. First, a human pilot model is used to investigate the overall system's behavior and explore the controller's performance for a reference tracking task. Then, human-in-the-loop experiments are conducted to analyze how the system responds in the presence of an actual human operator in the loop. Through careful simulation studies, we demonstrate that this architecture improves the estimation accuracy on a diverse set of uncertainties. The overall system's stability is analyzed via a rigorous Lyapunov analysis, and the proposed method is shown to be highly effective, as demonstrated through numerical simulations and human-in-the-loop experiments.

Keywords: Adaptive control, long short-term memory, human in-the-loop.

ÖZET

LSTM GÜÇLENDİRMESİ İLE ADAPTİF KONTROL: TEORİ VE DÖNGÜ İÇİNDE İNSAN DOĞRULAMASI

Emirhan İnanç
Makine Mühendisliği, Yüksek Lisans
Tez Danışmanı: Yıldıray Yıldız
Ağustos 2024

Bu tez, geleneksel uyarlanabilir kontrol yöntemlerine kıyasla önemli ölçüde daha iyi geçici tepki performansı sağlayan yeni bir uyarlanabilir kontrol mimarisi sunmaktadır. Bu, geleneksel bir Uyarlanabilir Sinir Ağı (YSA) kontrolörünün ve bir Uzun Kısa Süreli Bellek (LSTM) ağının sinerjik kullanımı ile gerçekleştirilir. LSTM yapıları, belirsizliği tahmin etmeye yardımcı olan bir giriş dizisindeki bağlılıklardan yararlanabilir. LSTM ağının kapalı döngü içinde YSA kontrolörünün eksikliklerini telafi etmeyi öğrendiği bir eğitim yöntemi sunuyoruz. Bu sayede sistemin geçici tepkisi iyileştirilmekte ve kontrolörün beklenmedik durumlarda hızla tepki vermesi sağlanmaktadır. Bu çalışma aynı zamanda önerilen kontrol çerçevesinin döngü içinde insan performansını da araştırmaktadır. LSTM ile güçlendirilmiş kontrol yöntemi, özellikle belirsizliklerde önemli ve hızlı değişikliklerin varlığında geçici tepkiyi büyük ölçüde iyileştirse de, güvenli çalışmayı sağlamak için bir insan operatörle etkileşimlerinin analiz edilmesi gerekir. İlk olarak, sistemin genel davranışını araştırmak ve referans izleme görevi için kontrolörün performansını incelemek için bir insan pilot modeli kullanılır. Ardından, sistemin döngüde gerçek bir insan operatörün varlığında nasıl tepki verdiğini analiz etmek için insan deneyleri yapılır. Özenle gerçekleştirilen simülasyon çalışmaları sayesinde, bu mimarinin çeşitli belirsizlikler kümesinde tahmin doğruluğunu artırdığını gösteriyoruz. Genel sistemin kararlılığı titiz bir Lyapunov analizi ile analiz edilmiş ve önerilen yöntemin performansı sayısal simülasyonlar ve döngü içinde insan deneyleri ile gösterilmiştir.

Anahtar sözcükler: Uyarlamalı kontrol, uzun kısa süreli bellek , döngüde insan.

Acknowledgement

I would like to express my deep gratitude to Assoc. Prof. Yıldıray Yıldız, whose insights and guidance have significantly contributed to this thesis from beginning to end. I am deeply grateful to Prof. Gökhan Kiper, whose encouragement inspired me to pursue an academic career. His continuous support has been a constant source of motivation throughout my journey. I would like to express my sincere gratitude to Prof. Ömer Morgül and Assist. Prof. Mert Ankaralı for their feedback as members of my thesis committee. Their insights and suggestions have significantly enriched the quality of this work.

I owe everything to my parents, Ümit and Figen İnanç, who have supported me through every stage of my academic journey. I could not have reached this point without their constant encouragement and belief in me.

During my master's studies, Abdullah Habboush played a pivotal role in my development, and I am especially grateful for both his contributions and his friendship. I would like to extend my thanks to M. Yusuf Uzun, who worked alongside me throughout this journey, offering invaluable support when it was most needed.

I would like to acknowledge my dear friends Aykut Bakan, Özgür Can Gümüş, Yunus Altıntop, Burak Sarı, Sena Aykut, Güneş Kibar, Doğa Dağ, Enise Kartal, Uzay Tefek, Batuhan Emre Kaynak, Sayedus Salehin, and Vahit Çorumlu for filling my work environment with joy and making my master's experience truly memorable.

Finally, my heartfelt thanks go to my long-time friend Melih Görgülü, who has always been a pillar of support since our high school days, and to Doğuhan Barlas Sevindik, whose friendship during my master's journey has been a gift I deeply appreciate.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Neural Networks (NN) have long been a topic of interest [1]. Still, affordable and fast parallel computing has unlocked its full potential, leading to its prominence in artificial intelligence and machine learning [2]. Fields such as computer vision and natural language processing have greatly benefited from these advancements [3, 4]. Deep learning research has produced robust structures like long short-term memory (LSTM) [5] and gated recurrent units (GRU) [6], which excel at processing sequential data [7]. Additionally, reinforcement learning (RL) has made significant progress through the application of deep learning techniques [8]. In RL, the system's state is updated based on actions an agent selects to maximize cumulative rewards. RL shares similarities with control systems, where system states, control inputs and feedback loops correspond to the states, actions, and observations in RL. As a result, RL-based controllers have been developed and explored in the literature [9, 10].

The extensive and well-established literature on NN-based adaptive control highlights the effectiveness of online-tuned feed-forward NN controllers in managing uncertain systems [11, 12, 13, 14, 15]. These controllers, equipped with adaptive update laws, have proven to be stable and capable of compensating for uncertainties [14]. However, they may exhibit oscillatory transient responses when faced with sudden changes in uncertainty. This issue is addressed by augmenting a conventional adaptive neural network (ANN) with a Long Short-Term Memory (LSTM) network. The LSTM is trained offline to compensate for the deficiencies of the ANN controller by leveraging its ability to capture long and short-term dependencies in the input sequence, thereby improving transient response and performance.

This hybrid control architecture, combining an ANN controller and an LSTM network, addresses the transient response issues inherent in traditional adaptive control methods. Inspired by reinforcement learning methodologies [16, 17], the LSTM is trained in a closed-loop setting to predict and compensate for undesirable transient error dynamics. Simulation results have demonstrated that the LSTM effectively mitigates high-frequency errors, complementing the ANN controller's management of low-frequency dynamics.

While this control method has shown dramatic improvements in transient response for predefined reference inputs, its interaction with human operators still needs to be investigated. Adaptive controllers can interact unpredictably with pilots, potentially leading to unfavorable outcomes [18, 19]. Given that pilot error and adverse pilot-flight control system interactions account for a significant portion of aviation accidents [20], it is crucial to evaluate how well the LSTM-ANN controller aligns with the human pilot's inputs. To this end, we conducted human-in-the-loop experiments to test the controller's performance.

Additionally, we examined the overall closed-loop performance by enhancing the ANN controller with the LSTM network through simulations involving a human pilot model following a continuously changing pitch reference. These simulations evaluated the controller's performance under varying uncertainties and reference signals. The results indicated that while the ANN controller performs adequately under certain conditions, it struggles without LSTM enhancement when faced with larger uncertainties. The LSTM enhancement significantly improved the transient response in human-in-the-loop experiments, underscoring its effectiveness in practical scenarios.

In summary, this thesis presents a novel adaptive control framework that enhances transient performance by integrating an LSTM network with a traditional ANN controller, validated through both simulations and human-in-the-loop experiments.

Chapter 2

Neural Network Adaptive Control with Long Short-Term Memory

2.1 Introduction

The methodology for approximating nonlinear functions with *neural networks* (NNs) in adaptive control and identification was established by Narendra and Parthasarathy in 1990 [21]. Well-established *adaptive neural network* (ANN) approaches are the result of several subsequent studies. In the very first step, only the outer NN weights are updated online by a tractable Lyapunov analysis made possible by the use of linearly parameterized NNs, such as radial basis function NNs [22, 23, 24, 25, 26]. These techniques involve the prior selection of appropriate basis functions or preliminary offline hidden-layer updating since hidden-layer weights are kept fixed. These issues are addressed in [14] by proposing an ANN for a general serial robot manipulator that has tunable hidden-layer weights. The problem gets substantially more complex when the hidden layer is adjusted, as demonstrated by the NN being nonlinear in its tunable weights. This problem is revisited in [27] for a single hidden layer ANN and [28] for a deep ANN.

While the previously mentioned studies make use of feed-forward NNs, several recent initiatives have focused on using more advanced NN structures. Using a technique inspired by *neural turing machines* (NTMs) [29], a method is presented in [30] to improve the performance of an ANN controller. However, instead of using a *recurrent neural network* (RNN), as is typically done in NTMs, a feed-forward network is employed. Because of their internal memory, RNNs are well-known for their capacity to process sequential data, and they have been successfully used in a variety of domains [31, 32, 33]. *Long short-term memory* (LSTM) networks are a special type of RNN. LSTM networks, in contrast to conventional RNNs, have the capacity to learn and remember data over long periods of time [34]. They show promise in many kinds of tasks and can resolve a number of issues that are still unsolvable with any other RNN architecture [35].

In this chapter, we present an adaptive control architecture that significantly improves controller performance by using an LSTM network. Specifically, we create a synergistic strategy in which the acting adaptive controller's shortcomings are mitigated by leveraging the long-term learning capabilities of LSTM networks. The ANN in [14] is adopted as the acting adaptive controller even though the suggested approach is indifferent to different adaptive control methods. The selected ANN controller allows for a general uncertainty structure and has produced successful implementation results [36, 37, 38]. An overall improvement in performance is achieved by training the LSTM in a closed-loop configuration to predict and compensate for the error made by the ANN in approximating the nonlinear uncertainty. Recursive simulations on a limited set of specified uncertainties are used to train the LSTM, and even with this small set, the LSTM can generalize well with unseen uncertainties. The suggested system's stability is studied using a Lyapunov analysis. Lastly, extensive simulation studies are employed to illustrate the performance of the proposed design.

The study has been developed for a general *multi-input multi-output* (MIMO) dynamical system, for which the ANN controller in [14] is modified in order to preserve the same stability characteristics. Consequently, a supplementary contribution of this work is an extended robustifying term, which generalizes the findings for MIMO dynamical systems in [14].

Other applications of LSTM networks for adaptive control have been reported in the literature. A single-axis hydraulic shaking table is controlled by an LSTM network in [39], where some LSTM weights are updated online, and the rest are tuned offline. For Euler-Lagrange systems, an online update of an LSTM controller's weights is suggested in [40]. In contrast to [39], the suggested architecture—which takes the form of (1)—is created for general MIMO dynamics as opposed to a particular use case. We also present a systematic approach to train the offline LSTM network to improve the active adaptive controller's transients. Despite being trained on a limited set of uncertainty, the suggested offline training strategy allows the LSTM network to generalize with unseen uncertainties effectively. This eliminates the need for updating the LSTM weights online as in [40], thus maintaining simplicity. Additionally, it is noted that the suggested approach is designed to enhance the performance of an active adaptive controller. Depending on the plant uncertainty and the acceptable degree of complexity in the controller structure, the proposed method can be implemented in one of three ways:1) a simple linearly parametrized form [41],2) a feedforward neural network form [42, 27], or 3) an LSTM network form [40].

To summarize, this chapter presents a novel control architecture where an LSTM network is employed to increase the transient performance of the active adaptive control element. We present a systematic method, in which the LSTM is trained by recursive simulations to complement the adaptive controller, resulting in enhanced performance. This work offers the whole theoretical development with a careful stability study and comprehensive simulation scenarios. A preliminary version of this work is published in our conference paper [43], and a detailed version is reported in [44].

The structure of this chapter is as follows: We outline the ANN controller's formulation in Section 2.2. The training procedure and the suggested LSTM augmentation are described in Section 2.3. Section 2.4 contains the simulation results, and Section 2.5 has a summary.

2.2 Problem Formulation

Consider the following plant dynamics

$$\dot{x}_p(t) = A_p x_p(t) + B_p(u(t) + f(x_p(t))), \tag{2.1a}$$

$$y_p(t) = C_p^T x_p(t), \tag{2.1b}$$

where $x_p(t) \in \mathbb{R}^{n_p}$ is the measurable state vector, $u(t) \in \mathbb{R}^m$ is the plant control input, $A_p \in \mathbb{R}^{n_p \times n_p}$ is a known system matrix, $f(x_p(t)) : \mathbb{R}^{n_p} \to \mathbb{R}^m$ is a state-dependent, possibly nonlinear, matched uncertainty, which is continuous on a known compact set, $B_p \in \mathbb{R}^{n_p \times m}$ is a known control input matrix, $C_p \in \mathbb{R}^{n_p \times s}$ is a known output matrix, and $y_p(t) \in \mathbb{R}^s$ is the plant output. Furthermore, it is assumed that the pair (A_p, B_p) is controllable.

Remark 1. For an unknown system matrix, the suggested approach is also applicable: Assume the dynamics of the plant are given by

$$\dot{x}_p(t) = A_u x_p(t) + B_p(u(t) + f_u(x_p(t))), \tag{2.2}$$

where $f_u(x_p(t))$ represents a state-dependent, potentially nonlinear, matched uncertainty and $A_u \in \mathbb{R}^{n_p \times n_p}$ is an unknown system matrix. Given a known matrix A_p, the dynamics (2.2) can be reformulated as (2.1) by defining $f(x_p(t)) \triangleq K_u x_p(t) + f_u(x_p(t))$, provided the matching condition $A_p = A_u - B_p K_u$ holds for some feedback gain $K_u \in \mathbb{R}^{m \times n_p}$. This enables us to assume A_p is known, without any loss of generality.

To accomplish command following of a reference input $r(t) \in \mathbb{R}^s$, we define a state $x_e(t) \in \mathbb{R}^s$ as

$$\dot{x}_e(t) = r(t) - y_p(t), \tag{2.3}$$

which, when combined with (2.1), results in the augmented dynamics as

$$\dot{x}(t) = Ax(t) + B_m r(t) + B(u(t) + f(x_p(t))), \tag{2.4a}$$

$$y(t) = C^T x(t), \tag{2.4b}$$

where $x(t) \triangleq [x_p(t)^T, x_e(t)^T]^T \in \mathbb{R}^n$ is the augmented state with dimension $n \triangleq$

$n_p + s$. The system matrices A, B_m, B and C are defined as

$$A \triangleq \begin{bmatrix} A_p & 0_{n_p \times s} \\ -C_p^T & 0_{s \times s} \end{bmatrix} \in \mathbb{R}^{n \times n}, \tag{2.5a}$$

$$B_m \triangleq \begin{bmatrix} 0_{n_p \times s} \\ I_{s \times s} \end{bmatrix} \in \mathbb{R}^{n \times s}, \quad B \triangleq \begin{bmatrix} B_p \\ 0_{s \times m} \end{bmatrix} \in \mathbb{R}^{n \times m}, \tag{2.5b}$$

$$C \triangleq \begin{bmatrix} C_p^T & 0_{s \times s} \end{bmatrix}^T \in \mathbb{R}^{n \times s}. \tag{2.5c}$$

A standard state feedback controller can be expressed as

$$u_{bl}(t) = -Kx(t), \tag{2.6}$$

where $K \in \mathbb{R}^{m \times n}$ is chosen so that $A_m \triangleq A - BK$ is Hurwitz. Subsequently, we define a reference model by

$$\dot{x}_m(t) = A_m x_m(t) + B_m r(t), \tag{2.7}$$

with $x_m(t) \in \mathbb{R}^n$ representing the state vector of the reference model.

By defining the state tracking error as

$$e(t) \triangleq x(t) - x_m(t), \tag{2.8}$$

the goal of adaptive control is to ensure that $e(t)$ converges to zero while maintaining all system signals within bounded limits and minimizing transient behavior. To accomplish this, we employ the control input

$$u(t) = u_{bl}(t) + u_{ad}(t) + u_{lstm}(t), \tag{2.9}$$

where $u_{bl}(t)$ is the baseline controller (2.6) designed to match the reference model (2.7) in the absence of uncertainties. The term $u_{ad}(t)$ represents an adaptive neural network (ANN) controller that compensates for the matched uncertainty $f(x_p(t))$, and $u_{lstm}(t)$ is generated by an LSTM network.

The ANN controller's transient performance is improved through LSTM training. This can be accomplished by generating a suitable $u_{ad}(t)$ to cancel out the matched uncertainty $f(x_p(t))$. Depending on the prior knowledge regarding this uncertainty $f(x_p(t))$, there are various methods available in the literature [41, 15, 45, 46].

2.2.1 Adaptive Neural Network Controller

The primary objective of the ANN control method is to estimate the unknown function $f(x_p(t))$ [14, 42, 47, 36, 37, 38]. Here, we give the basics that are essential to grasp the next section, which the main contribution of this chapter. We also discuss a modification to the original work here [14].

Assumption 1: The unknown matched uncertainty $f(x_p(t)) : \mathbb{R}^{n_p} \to \mathbb{R}^m$, in (2.1), is continuous within a known compact set $S_p \triangleq \{x_p(t) : \|x_p(t)\| \leq b_x\} \subset \mathbb{R}^{n_p}$, where b_x is a known positive constant.

Assumption 1 indicates that $f(x_p(t))$ can be approximated by a neural network (NN) [14] as

$$f(x_p(t)) = W^T \bar{\sigma}(V^T \bar{x}_p(t)) + \varepsilon(x_p(t)), \tag{2.10}$$

where W and V are the weight matrices of the NN, including the bias vectors. Additionally,

$$\|\varepsilon(x_p(t))\| \leq \varepsilon_N, \qquad \forall x_p(t) \in S_p, \tag{2.11}$$

ensuring that the approximation error $\varepsilon(x_p(t))$ is bounded within S_p.

Remark 2. Assumption 1 does not presuppose that the states x_p of the system in (1) always remain within S_p. It merely states that *when* x_p is within S_p, then the assumption's premise is valid. This will be elaborated further in the subsequent stability analysis.

The NN input is defined as $\bar{x}_p(t) \triangleq \begin{bmatrix} x_p(t)^T & 1 \end{bmatrix}^T \in \mathbb{R}^{n_p+1}$, and the hidden layer output is $\bar{\sigma}(V^T \bar{x}_p(t)) \triangleq \begin{bmatrix} \sigma(V^T \bar{x}_p(t)) & 1 \end{bmatrix}^T \in \mathbb{R}^{n_h+1}$. The nonlinear activation function $\sigma(.) : \mathbb{R}^{n_h} \to \mathbb{R}^{n_h}$ can be either sigmoid or tanh, where n_h represents the number of hidden neurons.

The overbar is dropped, and (2.10) is written as

$$f(x_p(t)) = W^T \sigma(V^T x_p(t)) + \varepsilon(x_p(t)) \tag{2.12}$$

for convenience, where the dimensions of the weight matrices are loosely defined as $W \in \mathbb{R}^{n_h \times m}$ and $V \in \mathbb{R}^{n_p \times n_h}$.

Assumption 2: There exist known positive constants that bound W and V such that $\|W\|_F \leq W_M$ and $\|V\|_F \leq V_M$.

The ANN controller $u_{ad}(t)$ in (2.9) is designed as

$$u_{ad}(t) = -\hat{f}(x_p(t)) + v(t), \tag{2.13}$$

to compensate for $f(x_p(t))$, where $\hat{f}(x_p(t))$ is an estimate of $f(x_p(t))$, and $v(t)$ represents a robustifying term, which is detailed in the following subsection. We define $\hat{f}(x_p(t))$ using (2.12) as

$$\hat{f}(x_p(t)) = \hat{W}^T(t)\sigma(\hat{V}^T(t)x_p(t)), \tag{2.14}$$

where $\hat{V}(t) \in \mathbb{R}^{n_p \times n_h}$ and $\hat{W}(t) \in \mathbb{R}^{n_h \times m}$ are estimates of the unknown ideal weights V and W, respectively. The adaptive laws used to update the weights are [14]

$$\dot{\hat{W}} = F(\hat{\sigma} - \hat{\sigma}'\hat{V}^T x_p)e^T PB - F\kappa\|e\|\hat{W}, \tag{2.15a}$$

$$\dot{\hat{V}} = Gx_p e^T PB\hat{W}^T\hat{\sigma}' - G\kappa\|e\|\hat{V}, \tag{2.15b}$$

where $F \in \mathbb{R}^{n_h \times n_h}$ and $G \in \mathbb{R}^{n_p \times n_p}$ are symmetric positive definite matrices, acting as learning rates, $\kappa > 0$ is a scalar gain, and $\hat{\sigma}'$ and $\hat{\sigma}$ are defined as

$$\hat{\sigma}' \triangleq \left.\frac{d\sigma(z)}{dz}\right|_{z=\hat{V}^T(t)x_p(t)}, \quad \hat{\sigma} \triangleq \sigma(\hat{V}^T(t)x_p(t)). \tag{2.16}$$

Furthermore, for some symmetric positive definite matrix $Q \in \mathbb{R}^{n \times n}$, $P \in \mathbb{R}^{n \times n}$ is the symmetric positive definite solution of the Lyapunov equation

$$A_m^T P + PA_m = -Q. \tag{2.17}$$

2.2.2 Robustifying Term

It is demonstrated in the work of Lewis et al. [14] that the adaptive control problem can be solved for nonlinear robot arm dynamics by using the baseline controller (2.6) and the ANN controller (2.13). However, the control input matrix

B presents a challenge for the generic multi-input multi-output (MIMO) dynamical system (2.1). A subsidiary contribution of this chapter, which addresses this issue, is a modified robustifying term $v(t)$. This term, in the form of (2.1), achieves the same stability results as those reported in Lewis et al.[14], but for general MIMO systems.

To provide robustness against disturbances resulting from the high-order Taylor series terms, the robustifying term $v(t)$ is employed. We propose a modified version of the robustifying term for the MIMO plant dynamics (2.1) as

$$v(t) = \begin{cases} 0, & \text{if } \|B^T Pe\| = 0 \\ -\frac{B^T Pe}{\|B^T Pe\|} k_z \|e\| (\|\hat{Z}\|_F + Z_M), & \text{otherwise} \end{cases}, \tag{2.18}$$

with

$$k_z > C_2, \tag{2.19}$$

where $e(t)$ is the state-tracking error given in (2.8), $\hat{Z}(t)$ is defined in (A.19) in the Appendix, $Z_M \triangleq \sqrt{W_M^2 + V_M^2}$ (see Assumption 2), B is the input matrix in (2.1), and P is the solution of (2.17). Furthermore, $C_2 \triangleq 2C_{\sigma'} Z_M$, where $C_{\sigma'}$ is a known upper bound for the absolute value of the sigmoid or tanh activation function derivatives, i.e. $||\sigma'(.)|| \leq C_{\sigma'}$, where $\sigma'(z) = d\sigma(z)/dz$.

The main stability finding of the overall architecture given in the sequel (see Theorem 1) depends critically on the robustifying term, as the Appendix illustrates. The following section introduces the main focus of this chapter.

2.3 LSTM Network Design

In this section, we clearly outline the distinct roles of the LSTM and ANN within the closed-loop system. The overall control architecture is depicted in Figure 3.1.

Figure 2.2 shows an LSTM block, where the input e^n_{norm} represents the normalized version of the sequence e^n. This sequence is obtained from $e(t)$ defined in (2.8) as $e^n = e(nT)$, where n represents the current time step and $T > 0$ is

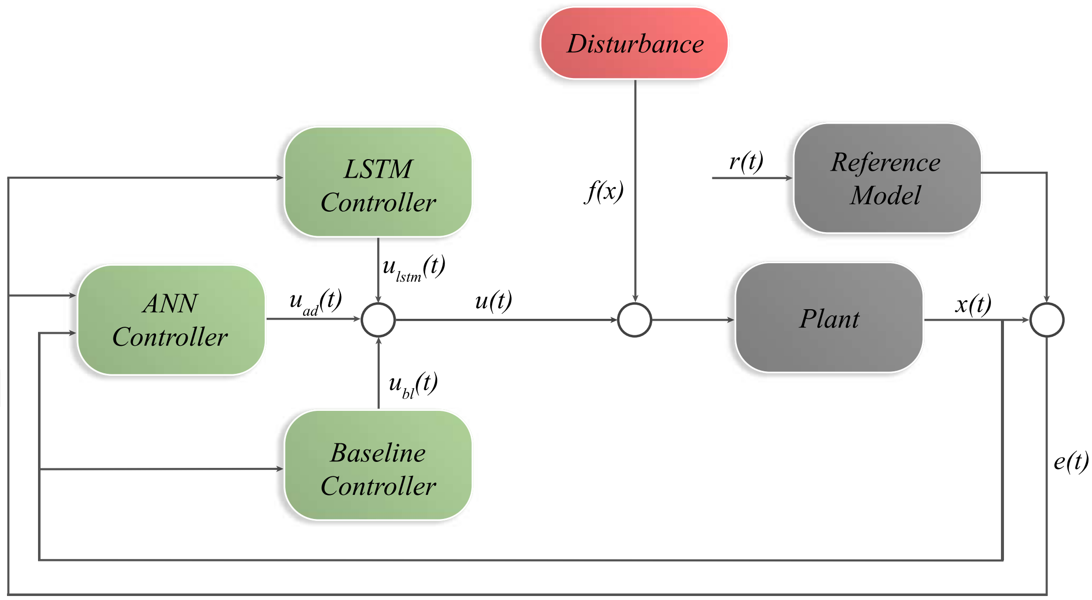

Figure 2.1: Block diagram of the proposed control architecture.

the LSTM sampling time. The previous time step is indicated by the superscript $n-1$. Moreover, the cell state is denoted by $c^n \in \mathbb{R}^{N_h}$, and the hidden state is denoted by $h^n \in \mathbb{R}^{N_h}$, where N_h is the total number of hidden units in the LSTM network.

LSTM uses its previous hidden state h^{n-1} in addition to the input sequence e^n_{norm} to 1) update the cell state utilizing the outputs of a *forget gate* out^n_f, a *input gate* out^n_i, and a *cell candidate* $\bar{c}^n$ as

$$c^n = out^n_f \odot c^{n-1} + out^n_i \odot \bar{c}^n, \tag{2.20}$$

and 2) update the hidden state using the updated cell state and the output of an *output gate* out^n_o as

$$h^n = out^n_o \odot \sigma_c(c^n). \tag{2.21}$$

These gate operations are applied using the activation functions $\sigma_g(.)$ and $\sigma_c(.)$, which denote sigmoid and tanh, respectively, and suitable weights $\{R_f, W_f, b_f\}$, $\{R_c, W_c, b_c\}$, $\{R_i, W_i, b_i\}$, and $\{R_o, W_o, b_o\}$, as shown below.

a) Forget Gate: The forget gate, as the first term in (2.20) indicates, decides which data the cell state c^n should keep from its previous time step. The output is computed as

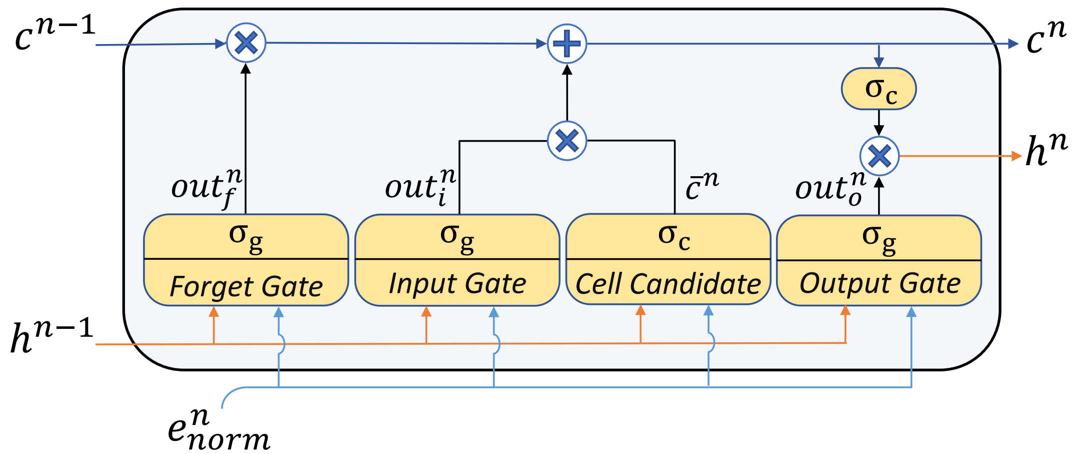

Figure 2.2: Detailed sketch of an LSTM block.

$$out_f^n = \sigma_g(W_f e_{norm}^n + R_f h^{n-1} + b_f). \tag{2.22}$$

b) Input Gate and Cell Candidate: A cell candidate $\bar{c}^n$ provides the information flow into the cell state (note the second term in (2.20)) as

$$\bar{c}^n = \sigma_c(W_c e_{norm}^n + R_c h^{n-1} + b_c), \tag{2.23}$$

and is adjusted by the output of the input gate

$$out_i^n = \sigma_g(W_i e_{norm}^n + R_i h^{n-1} + b_i). \tag{2.24}$$

c) Output Gate: How much of the cell state will be used as the hidden state for the next step is determined by the output gate (see (2.21)), which is defined as

$$out_o^n = \sigma_g(W_o e_{norm}^n + R_o h^{n-1} + b_o). \tag{2.25}$$

Finally, to formulate the output sequence of the LSTM network, the hidden state is fed to a fully connected layer yielding an $m-$dimensional output that matches the dimension of the control input as

$$u_{lstm}^n = W_{fc} h^n + b_{fc}, \tag{2.26}$$

where W_{fc} and b_{fc} are the parameters of the fully connected layer. The control input $u_{lstm}(t)$ is constructed as

$$u_{lstm}(t) = u_{lstm}^{n}, \quad \text{for } nT \leq t < (n+1)T, \quad n = 0, 1, 2, \dots \,. \tag{2.27}$$

2.3.1 Training Method

The LSTM network is designed to predict and correct the estimation error of the ANN controller. This error is defined by the difference between the ANN controller input, $\hat{f}(x_p(t))$ (refer to (2.13)), and the actual uncertainty, $f(x_p(t))$, as

$$\tilde{f}(x_p(t)) \triangleq f(x_p(t)) - \hat{f}(x_p(t)). \tag{2.28}$$

The "truth" for the LSTM, or the target signal that the LSTM aims to generate, is defined as

$$y(t) = -\tilde{f}(x_p(t)). \tag{2.29}$$

Therefore, LSTM provides a signal to the system that is an estimate of this target signal as

$$u_{lstm}(t) = \hat{y}(t). \tag{2.30}$$

During the training process, an uncertainty $f_{train}(x_p(t))$ is introduced into the closed-loop system. The LSTM is then expected to learn $\tilde{f}_{train}(x_p(t)) = f_{train}(x_p(t)) - \hat{f}(x_p(t))$, with $-\hat{f}(x_p(t))$ being the ANN's estimate used to mitigate the uncertainty (refer to (2.13)). Alongside $\tilde{f}_{train}(x_p(t))$, the LSTM takes the state-tracking error $e(t)$, defined in (2.8), as part of its input (see Fig. 2.1). Each iteration of training modifies the sequence the LSTM is trained on, and this dynamic nature of training aids the LSTM in generalizing to a range of functions not present in the training set.

Figure 2.3 illustrates the LSTM network's training process. Initially, the LSTM weights are initialized, and training data $(e(t), y(t))$ are generated through numerical simulations. After updating the LSTM weights, the training data are regenerated from simulations using these updated weights. This step is crucial

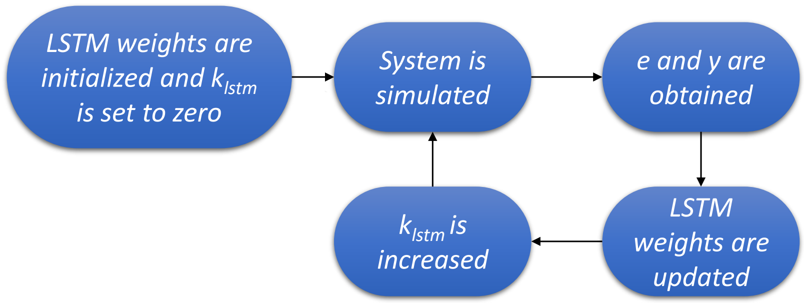

Figure 2.3: Flow chart of the training process

because, in a closed-loop setting, changes in the LSTM weights impact its output, which subsequently influences the evolution of its input. To prevent the untrained LSTM network's output from adversely affecting the system's response at the beginning of the training, the LSTM output u_{lstm} is scaled by a gain "k_{lstm}". This gain starts at 0 and gradually increases to 1 as training progresses, maintaining a value of 1 for the remainder of the training process.

2.3.2 Normalization

The input e^n for the LSTM network (see Fig. 2.2) is normalized before training. The simulation is run with f_{train} (see explanations after (2.30)) for z number of times in order to gather the normalization parameters only. For each simulation run, a gain is uniformly sampled from $[0, 1]$ in order to scale f_{train}. The minimum and maximum values of each error vector component, e_i^{min} and e_i^{max}, within the gathered data are used to derive the normalized input parameters as

$$e^n_{norm_i} = (e_i^n - e_i^{min})/(e_i^{max} - e_i^{min}), \tag{2.31}$$

where "i" is used to refer to the vector's i^{th} component.

2.3.3 Stability Analysis

The following theorem outlines the stability properties of the overall architecture, the proof of which is postponed to the Appendix.

Theorem 1. *Consider the uncertain plant dynamics described in (2.1), subject to Assumptions 1 and 2, along with the reference model (2.7). Define the control input according to (2.9), which includes the baseline controller (2.6), the ANN controller specified by (2.13), (2.14), and (2.15), the robustifying term provided in (2.18) and (2.19), and the LSTM controller detailed in (2.22)-(2.30). Additionally, select the Lyapunov matrix Q in (2.17) such that*

$$\lambda_{min}(Q) > \frac{\kappa q_1^2 + 4q_0}{2(b_x - C_m)}, \tag{2.32}$$

with $b_x > C_m$, where $q_0 \triangleq \|B^T P\|F(C_0 + \bar{u}_{lstm})$, $q_1 \triangleq C_1\|B^T P\|F/\kappa + Z_M$, κ is the scalar gain used in (2.15), b_x is defined in Assumption 1, and C_0, C_1, C_m, and $\bar{u}_{lstm}$ are defined in the Appendix. Given that $x_p(0) \in S_p$ (see Assumption 1), the solution $(e(t), \tilde{W}(t), \tilde{V}(t))$ is uniformly ultimately bounded (UUB) [48] and converges to a predefined compact set, where $\tilde{V}(t) \triangleq V - \hat{V}(t)$ and $\tilde{W}(t) \triangleq W - \hat{W}(t)$.

Proof. See Appendix A.2. □

Note that increasing Q alone does not ensure that the condition in (2.32) is satisfied. This is because raising Q results in an increased P, which implies higher values q_0 and q_1, according to the Lyapunov equation (2.17). For any scalar $\gamma > 0$, one way to solve this is to use the pair (γP_0,$\gamma Q_0 = \gamma I$), which satisfies (2.17). Then, one can locate an appropriate γ to satisfy (2.32): Substituting this pair to (2.32), yields

$$b\gamma^2 + (c - a)\gamma + d < 0, \tag{2.33}$$

where $b \triangleq C_1^2\|B^T P_0\|_F^2/\kappa$, $c \triangleq 2C_1\|B^T P_0\|_F Z_M + 4\|B^T P_0\|_F(C_0 + \bar{u}_{lstm})$, $a \triangleq 2(b_x - C_m)$, and $d \triangleq \kappa Z_M^2$. The condition (2.33) is in a quadratic form, for which there exists a positive real $\gamma > 0$ if and only if $a \geq \sqrt{4bd} + c$.

Remark 3. Without the need for a neural network, the suggested LSTM augmentation method can also be used with a conventional adaptive controller in which it is assumed that the matched uncertainty $f(x_p(t))$ in (3.3) is linearly parametrized. It is also possible to present stability proofs and simulation results that highlight the benefits of the chosen strategy in this particular scenario.

2.4 Simulations

The effectiveness of the suggested framework is demonstrated in this section for the control of the short-period longitudinal flight dynamics, which are provided for a B-747 aircraft flying at a speed of 274 m/s and an altitude of 6000 m as [30]

$$\begin{bmatrix} \dot{\alpha} \\ \dot{q} \end{bmatrix} = \begin{bmatrix} -0.32 & 0.86 \\ -0.93 & -0.43 \end{bmatrix} \begin{bmatrix} \alpha \\ q \end{bmatrix} + \begin{bmatrix} -0.02 \\ -1.16 \end{bmatrix} (u + f(x_p)), \tag{2.34}$$

where α is the angle of attack (rad), q is the pitch rate (rad/s), u is the elevator deflection (rad), $x_p \triangleq [\alpha,\ q]^T$, and $f(x_p)$ is a state dependent matched uncertainty. Pitch rate tracking is the goal of the control strategy. In addition, the saturation limits for elevator magnitude and rate are set at $+17/-23$ (deg) and $+37/-37$ (deg/s) [49]. The augmented state is defined as $x(t) \triangleq [x_1(t),\ x_2(t),\ x_3(t)]^T$, where $x_1 = \alpha$ (rad), $x_2 = q$ (rad/s), and x_3 is the error-integral state (2.3). This sets the dynamics in the form of (2.4).

The input of the LSTM network is $in^n_{lstm} = \begin{bmatrix} e^n_{norm_1} & e^n_{norm_2} & e^n_{norm_3} \end{bmatrix}^T$, where the components of the normalized version of the state tracking error vector (2.31) are $e^n_{norm_1}$, $e^n_{norm_2}$, and $e^n_{norm_3}$. The baseline controller (2.6) is an LQR controller with cost matrices $Q_{LQR} = I$ and $R_{LQR} = 1$. The LSTM network is trained alongside an ANN controller (refer to Section 2.2.1), which includes a hidden layer containing $n_h = 4$ neurons. The learning rates in (2.15a) and (2.15b) are both set to $F = G = 10$. The Lyapunov matrix Q in (2.17) is chosen as $Q = I$. The outer weights $\hat{W}$ and the bias $\hat{b}_w$ are initialized to zero. Random initialization is used to set the inner weights $\hat{V}$ and biases $\hat{b}_v$ between 0 and 1. The LSTM network contains one hidden layer with $N_h = 128$ neurons. Due to the number of state-tracking error components, the input layer has three neurons. Xavier Initialization method is used to initialize the LSTM weights [50]. Stochastic gradient descent is used to train the network, and the Adam optimizer with gradient clipping and a threshold value of 1 is used. The minibatch size is assumed to be 1, and the simulation step time is set to 0.01 s. In Section 2.3.2, the system is simulated 1000 times ($z = 1000$, see Section 2.3.2), yielding the normalization constants provided in (2.31). The loss function is chosen as $L(y, \hat{y}) = \frac{1}{N}\sum_{i=0}^{N}(y - \hat{y})^2$, where y and $\hat{y}$ are defined in (3.15) and (2.30), and N is the number of data in the data set.

Below, we investigate two scenarios, presenting simulation results for each: 1) a generic nonlinear uncertainty $f(x_p(t))$ that satisfies the conditions given in Assumption 1, and 2) an uncertainty representation structured as specified in (2.12).

2.4.1 Scenerio 1

The LSTM network (2.26) is trained on the nonlinear uncertainty f_{train} (as explained following (2.30)), while the test uncertainty f_{test}, used to evaluate the proposed control framework, is defined respectively as

$$
f_{train}(x_p) = \begin{cases} 0.1x_1x_2 & \text{if } 0 \leq t < 5 \\ \exp(x_2) & \text{if } 5 \leq t < 10 \\ 2x_1x_2 & \text{if } 10 \leq t < 15 \\ -0.1\cos(x_1) & \text{if } 15 \leq t < 20 \\ 0.5(x_1 - x_2) & \text{if } 20 \leq t < 25 \\ 0.1x_1x_2 & \text{if } 25 \leq t < 30 \\ -x_1x_2(\sin(5x_1x_2) + 5\sin(x_2)) & \text{if } 30 \leq t < 35 \\ x_1x_2(3\sin(2x_1x_2) + 2x_1) & \text{if } 35 \leq t < 45 \\ -x_1x_2(2(\tan(2x_1x_2) + x_2^2) & \text{if } 45 \leq t < 55 \\ x_1x_2(x_1 + x_2) & \text{if } 55 \leq t \leq 60 \end{cases}, \tag{2.35}
$$

$$
f_{test}(x_p) = \begin{cases} 0 & \text{if } 0 \leq t < 2 \\ -0.1\exp(x_1x_2) & \text{if } 2 \leq t < 8 \\ 0.5x_2^2 & \text{if } 8 \leq t < 12 \\ 0.05\exp(x_1 + 2x_2) & \text{if } 12 \leq t < 20 \\ -0.1\sin(0.05x_1x_2) & \text{if } 20 \leq t < 28 \\ 0.1(x_1^2 + x_2^3) & \text{if } 28 \leq t < 34 \\ -0.1\sqrt{cos(x_2)^2} & \text{if } 34 \leq t < 40 \\ -0.2\sin(x_1x_2) & \text{if } 40 \leq t < 49 \\ 0.1(x_1x_2)^2 & \text{if } 49 \leq t < 53 \\ 0.5x_2 & \text{if } 53 \leq t \leq 60 \end{cases}, \tag{2.36}
$$

where (2.36) is designed to include different types of sub-functions with various time intervals compared to (2.35). To address both low and high-value uncertainties, f_{train} is scaled by a parameter alternating between 0.2 and 2 during training. Furthermore, the robustifying term $v(t)$ in (2.13) is set to zero. There are two main reasons for this choice. First, satisfactory control performance is achieved without this term, simplifying the implementation. Second, implementing the term requires calculating Z_M in (2.18), which involves determining the upper

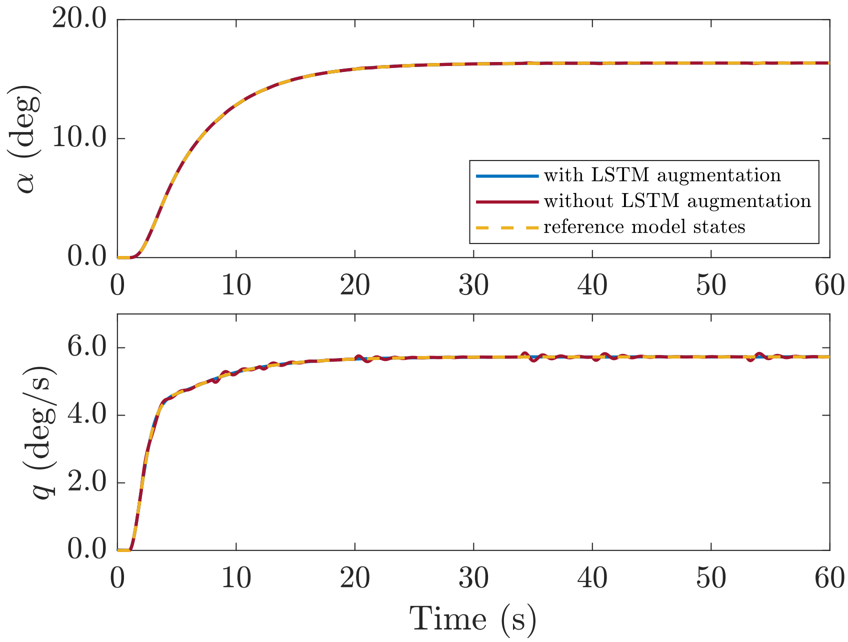

Figure 2.4: Tracking performances with and without LSTM augmentation, in the presence of small uncertainty.

bounds on the ideal NN weights. Although this can be accomplished, it adds complexity to the implementation. One approach is to train an additional NN to approximate potential uncertainties and use upper bounds that limit the trained NN weights' upper limits.

2.4.1.1 Controller performance with small uncertainty

In this section, the proposed LSTM augmentation performance is examined using f_{test} with a scaling factor of 0.1.

Figures 2.4 and 2.5 illustrate the state-tracking and tracking-error curves, respectively. From the error plots, it is evident that the LSTM augmentation notably diminishes both error magnitudes and transient oscillations. Despite this improvement, the error values remain minimal and negligible in relation to the absolute state values, suggesting that performance in both the LSTM-augmented

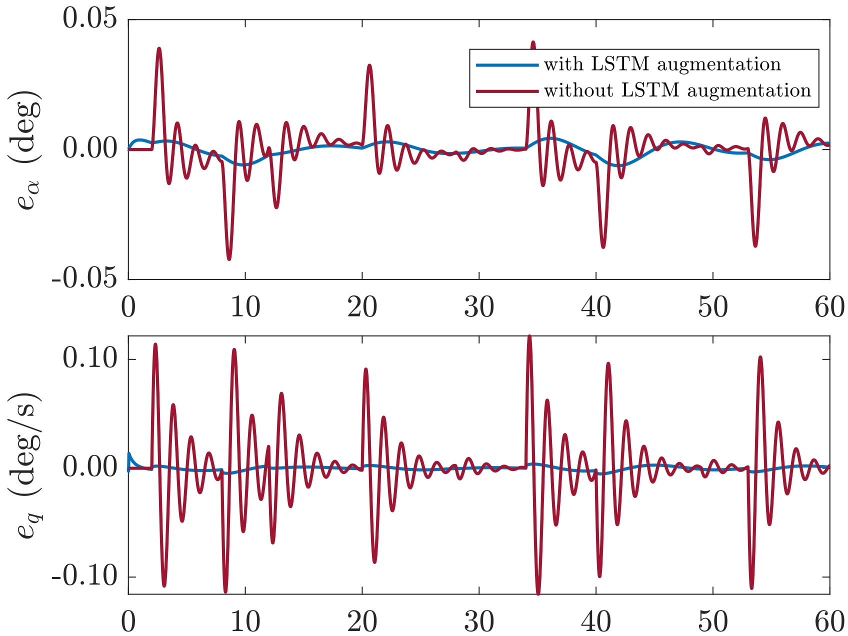

Figure 2.5: Tracking errors with and without LSTM augmentation, in the presence of small uncertainty.

and non-augmented scenarios is comparable. Figure 2.6 depicts the control inputs, indicating that although the LSTM augmentation does not greatly affect the overall control input magnitude, it effectively reduces oscillations by providing precise, agile compensation for uncertainties.

2.4.1.2 Controller performance with large uncertainty

In this section, the proposed LSTM augmentation performance is examined using f_{test} in (2.36), without a scaling factor.

Figures 2.7 and 2.8 show that LSTM augmentation significantly enhances the system's transient response, particularly in tracking the pitch rate, which is the primary output of interest. It is important to note that the same ANN controller and LSTM network are utilized as in the small uncertainty case. Figure 2.9 displays the individual control inputs. In this scenario, unlike the low-uncertainty case, the LSTM augmentation makes the total control signal noticeably more

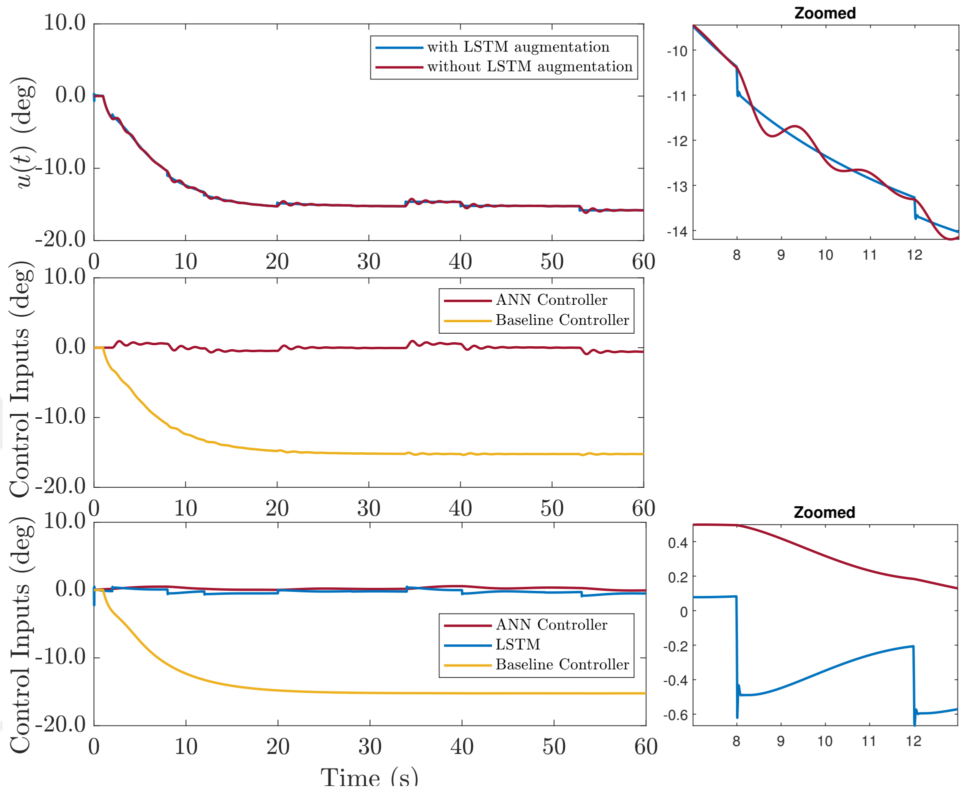

Figure 2.6: Control inputs in the presence of small uncertainty. Top: Total control input. Middle: The contributions of individual control inputs without the LSTM. Bottom: The contributions of individual control inputs with the LSTM.

responsive, effectively preventing excessive oscillations.

2.4.1.3 Addressing Saturation

The zoomed-in section of the top sub-figure in Fig. 9 reveals that the total control input experiences rate saturation at certain times. While this did not result in any severe adverse effects, addressing rate saturation is recommended to ensure safe and reliable control systems. To tackle this issue, we modified LSTM training by: a) incorporating the saturation limits into the plant dynamics during training, and b) informing the LSTM network whenever the control signal rate saturates. The latter is accomplished by providing additional input to the LSTM network as

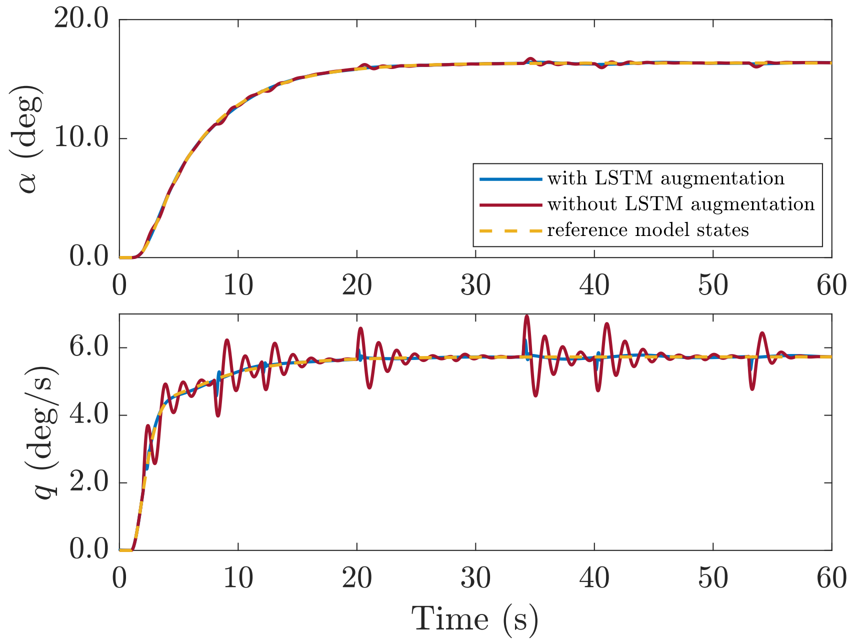

Figure 2.7: Tracking performances with and without LSTM augmentation, in the presence of large uncertainty.

$$u_r(t) = \begin{cases} 0.1, & \text{if rate saturation is positive,} \\ -0.1, & \text{if rate saturation is negative,} \\ 0, & \text{otherwise.} \end{cases} \tag{2.37}$$

The number of neurons in the input layer of the LSTM network has been increased to 4, as the modified input now includes additional information given by

$$in_{lstm}(t) = [e_{norm_1}(t)\ e_{norm_2}(t)\ e_{norm_3}(t)\ u_r(t)]^T. \tag{2.38}$$

The contribution of the LSTM network and the overall control inputs are shown in Figure 2.10, with and without rate-limit information during training. The LSTM network uses the rate-limit data to deliver smoother compensation. After the LSTM network is trained using the rate-limit information, the tracking

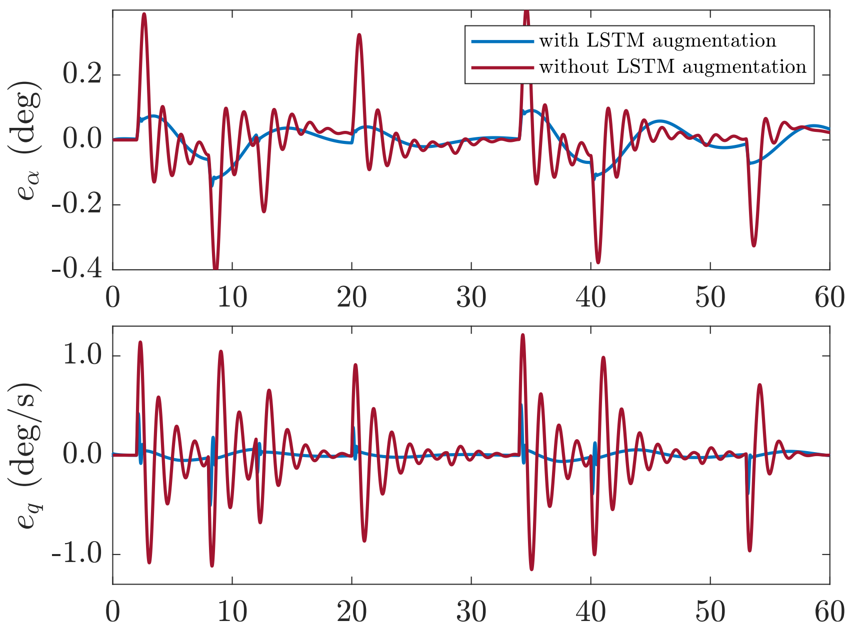

Figure 2.8: Tracking errors with and without LSTM augmentation, in the presence of large uncertainty.

error and tracking curves are shown in Figures 2.11 and 2.12, respectively. Although there are small variations in the pitch rate during uncertainty shifts, the suggested controller performs similarly even if the LSTM is not trained using this data.

2.4.2 Scenerio 2

In this section, the uncertainty $f(x_p(t))$ in (2.1) is represented in the form of (2.12). For both training and testing functions, the weight matrices W and V are randomly generated within the range of $[-0.05, 0.05]$ as described in (2.35) and (2.36). The NN reconstruction error vector $\epsilon(x_p(t))$ is then defined separately for the training and testing functions, respectively, as

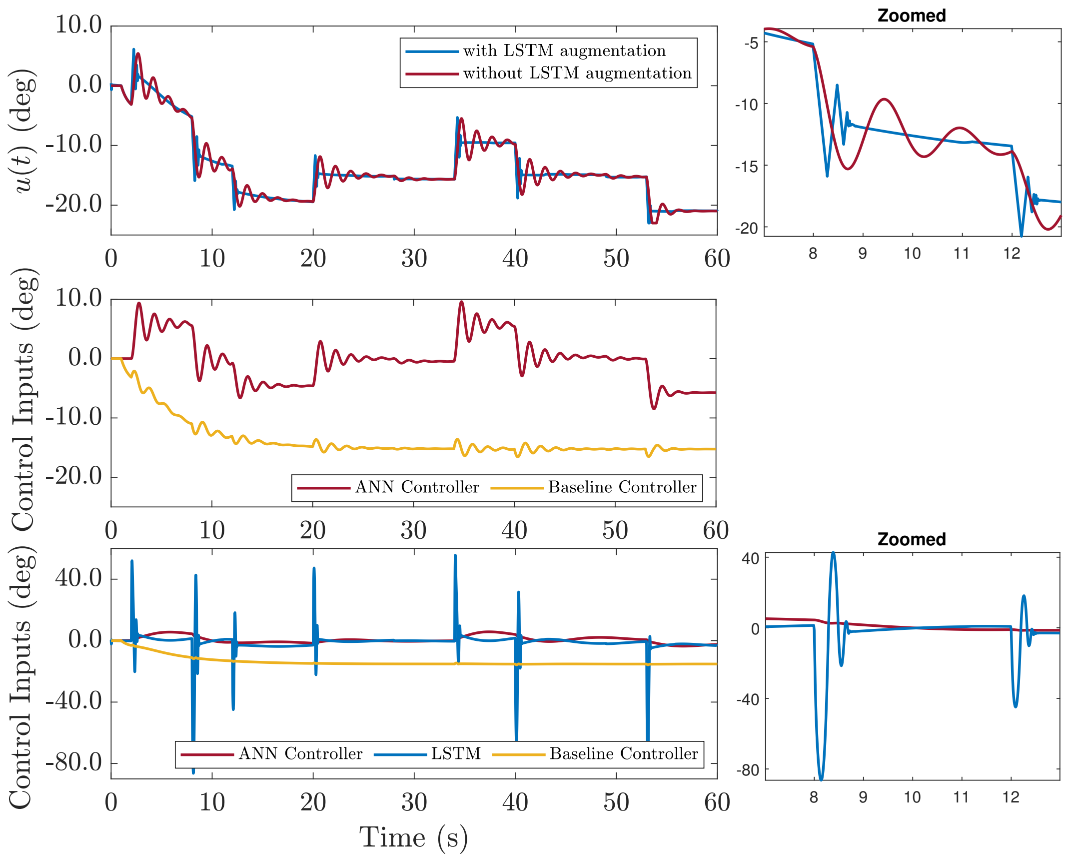

Figure 2.9: Control inputs in the presence of high uncertainty. Top: Total control input. Middle: The contributions of individual control inputs without the LSTM. Bottom: The contributions of individual control inputs with the LSTM.

$$\epsilon(x_p(t))_{train} = \begin{cases} 0.1cos(\|x_p\|)^2 & \text{if } 0 \leq t < 5 \\ -0.1cos(\|x_p\|) & \text{if } 5 \leq t < 10 \\ 0.1sin(\|x_p\|) & \text{if } 10 \leq t < 15 \\ 0.1sin(\|x_p\|)^2 & \text{if } 15 \leq t < 20 \\ 0.1sin(\|x_p\|) & \text{if } 20 \leq t < 25 \\ 0.1cos(\|x_p\|) & \text{if } 25 \leq t < 30 \\ 0.1sin(\|x_p\|)^2 & \text{if } 30 \leq t < 35 \\ -0.1cos(\|x_p\|) & \text{if } 35 \leq t < 45 \\ 0.1sin(\|x_p\|) & \text{if } 45 \leq t < 55 \\ -0.1sin(\|x_p\|)^3 & \text{if } 55 \leq t \leq 60 \end{cases}, \tag{2.39}$$

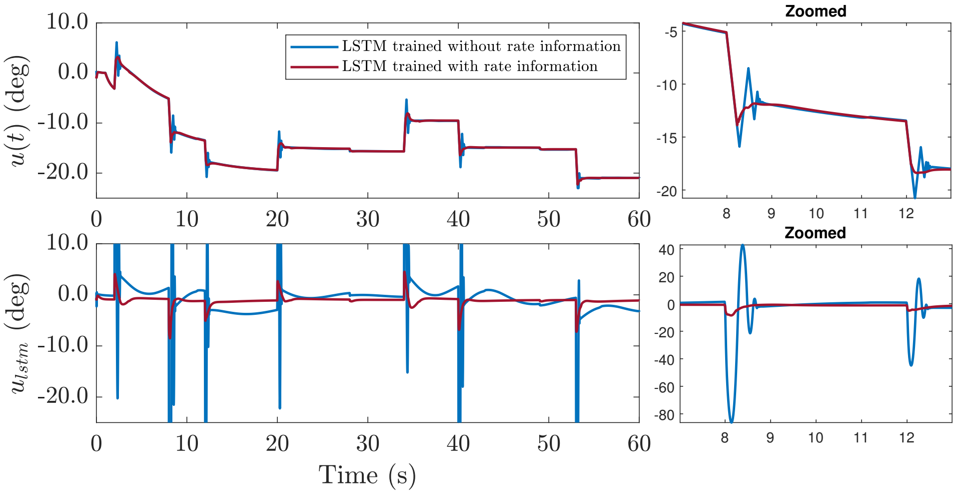

Figure 2.10: Control signals with an without rate-limit-informed LSTM. Top: Total control input. Bottom: LSTM network contribution to the total control input.

$$\epsilon(x_p(t))_{test} = \begin{cases} 0.1sin(\|x_p\|)^2 & \text{if } 0 \leq t < 2 \\ 0.1cos(\|x_p\|) & \text{if } 2 \leq t < 8 \\ 0.1cos(\|x_p\|)^2 & \text{if } 8 \leq t < 12 \\ 0.1sin(\|x_p\|)^2 & \text{if } 12 \leq t < 20 \\ 0.1sin(\|x_p\|) & \text{if } 20 \leq t < 28 \\ -0.1sin(\|x_p\|)^2 & \text{if } 28 \leq t < 34 \\ 0.1cos(\|x_p\|) & \text{if } 34 \leq t < 40 \\ -0.1cos(\|x_p\|)^3 & \text{if } 40 \leq t < 49 \\ 0.1sin(\|x_p\|) & \text{if } 49 \leq t < 53 \\ 0.1cos(\|x_p\|) & \text{if } 53 \leq t \leq 60 \end{cases}. \tag{2.40}$$

The second term of the uncertainty representation in (2.12) is the reconstruction error, which is represented by (2.39) and (2.40). In comparison to the first term in (2.12), this term is substantially smaller. Thus, choosing distinct weight matrices V and W, which are produced by uniformly random sampling within a

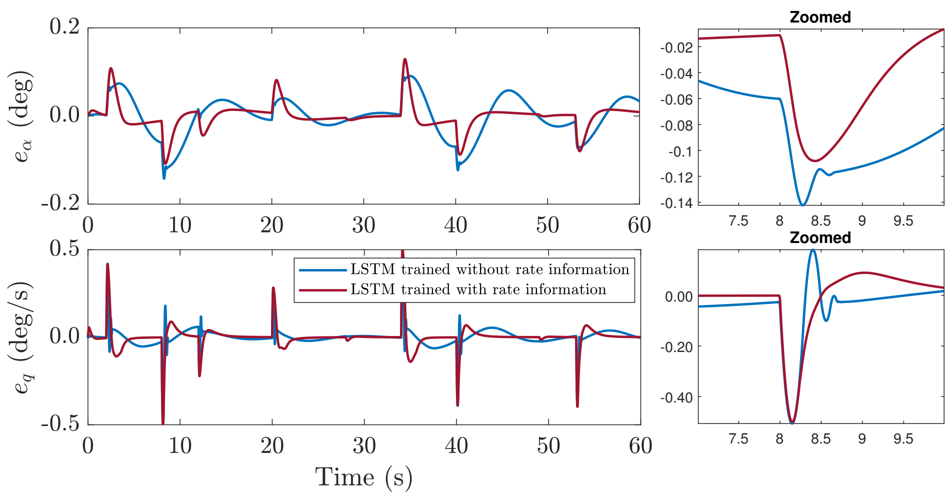

Figure 2.11: Tracking error with LSTM trained without rate information and rate-limit-informed LSTM

given range, is the primary method to ensure that the test and training uncertainties differ. For this reason, it is adequate to allocate distinct time intervals to the corresponding terms in (2.39) and (2.40).

The procedure is made simpler by eliminating the robustifying term's implementation in (2.13), as was covered in the previous subsection. However, when dealing with an explicit uncertainty representation (2.12), the robustifying term can be implemented easily because no more NN training is needed. We employ weight-clipping at the range $[\pm 0.4]$ to derive $\bar{u}_{lstm}$ for (2.33), and select $C_m = 0.4$ (refer to the Appendix). Then, $\gamma = 1$, $\kappa = 1$, and $b_x = 84$ satisfy (2.33).

2.4.2.1 Controller performance with small uncertainty

The performance of the proposed LSTM augmentation is studied in this section while small-valued uncertainties are present. These uncertainties are produced by scaling the test uncertainty by 0.1.

Figures 2.13 and 2.14 show the tracking and tracking-error curves, respectively. The results demonstrate that the LSTM-augmented system reduces the tracking error and minimizes oscillations. Although there is an improvement, it is not

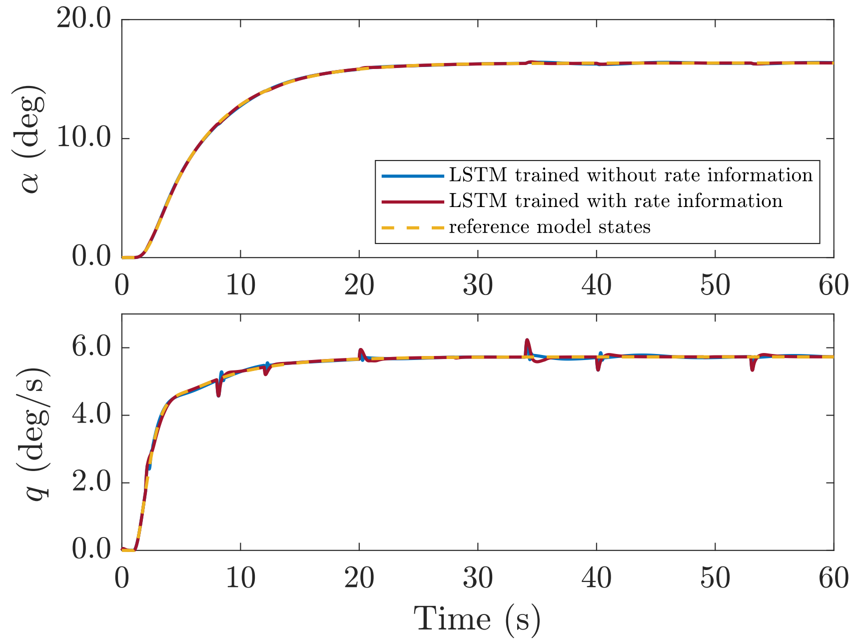

Figure 2.12: Tracking performances with LSTM trained without rate information and rate-limit-informed LSTM

substantial. Figure 2.13 shows that the ANN controller with the robustifying term provides satisfactory performance. As in the previous scenario, the control input of the LSTM network in Figure 2.15 generates more agile control signals, making the system more responsive to sudden changes.

2.4.2.2 Controller performance with large uncertainty

The objective of this section is to examine the effects of the suggested LSTM augmentation when high-valued uncertainties are present. These uncertainties are introduced by employing the test uncertainty directly, without a scaling factor.

The results presented in Figs. 2.16 and 2.17 demonstrate a significant improvement in the system's transient behavior compared to the scenario with smaller uncertainties. A detailed examination of the individual control inputs in Fig. 2.18 highlights that the LSTM network's contribution enables the system to respond quickly to rapidly changing uncertainties. Moreover, a closer inspection of

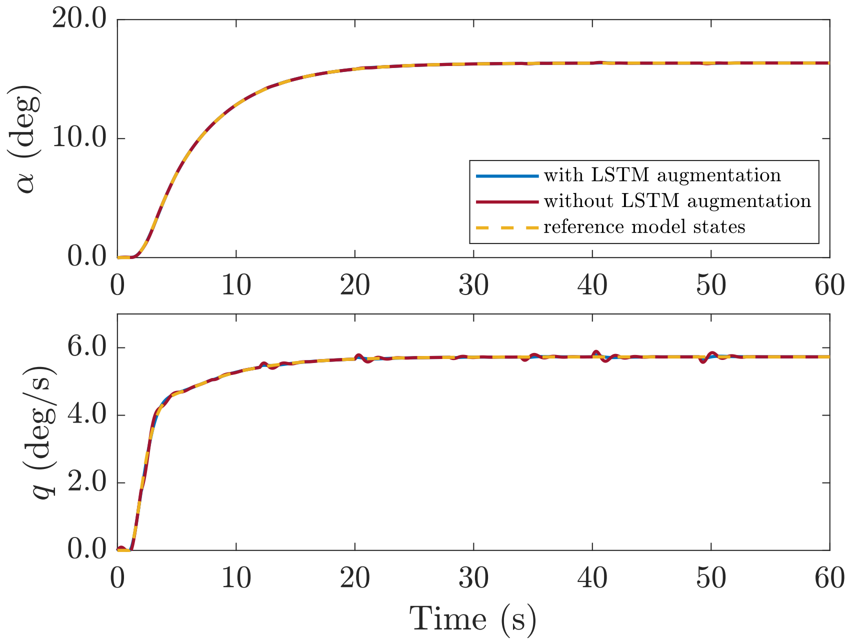

Figure 2.13: Tracking performances with and without LSTM augmentation, in the presence of small uncertainty.

Fig. 2.18 shows that the LSTM augmentation effectively reduces the oscillatory behavior of both the robustifying term and the ANN controller.

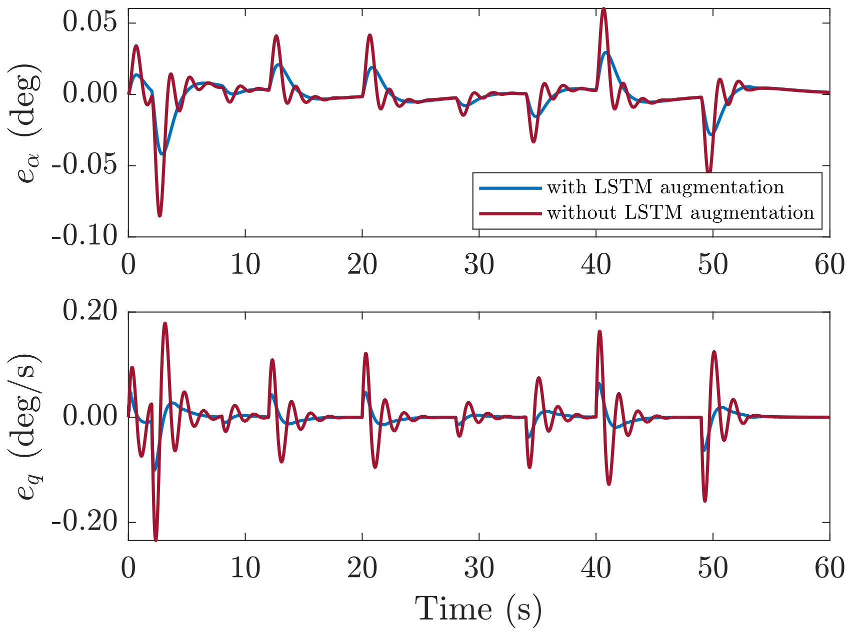

Figure 2.14: Tracking error with and without LSTM augmentation, in the presence of small uncertainty.

2.5 Summary

This study introduces an adaptive neural network (ANN) control structure enhanced with Long Short-Term Memory (LSTM) to improve the transient response of adaptive closed-loop control systems. We show that the LSTM, with its ability to predict time-series data, enables the ANN controller to handle uncertainties more effectively, leading to significantly improved tracking performance.

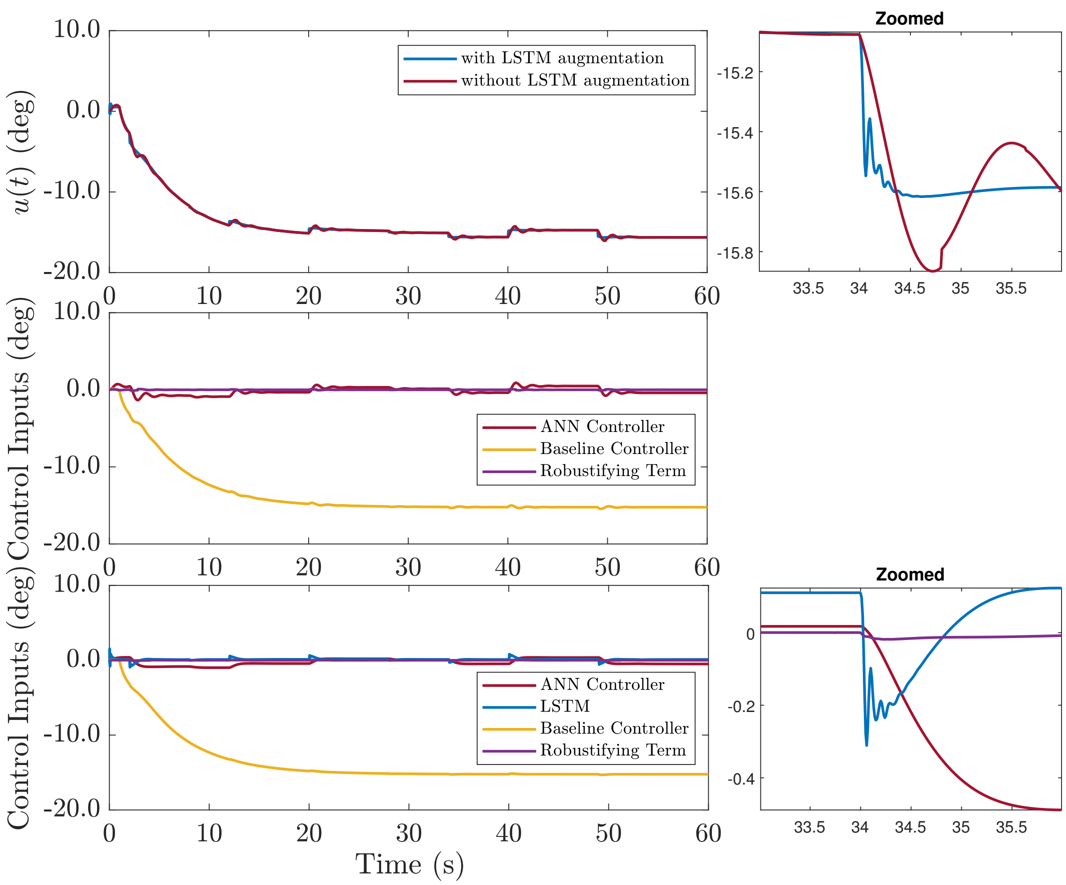

Figure 2.15: Control inputs in the presence of small uncertainty. Top: Total control input. Middle: The contributions of individual control inputs without the LSTM. Bottom: The contributions of individual control inputs with the LSTM.

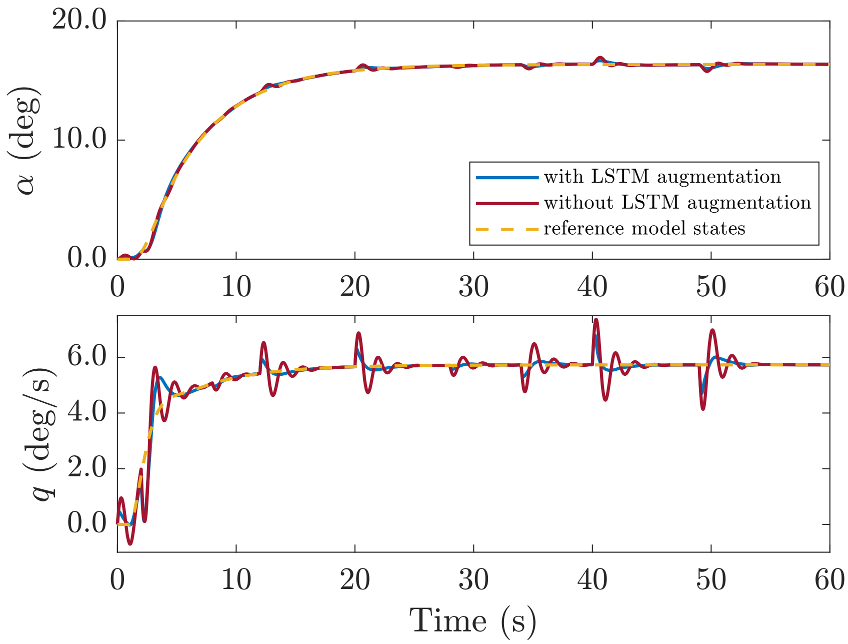

Figure 2.16: Tracking performances with and without LSTM augmentation, in the presence of high uncertainty.

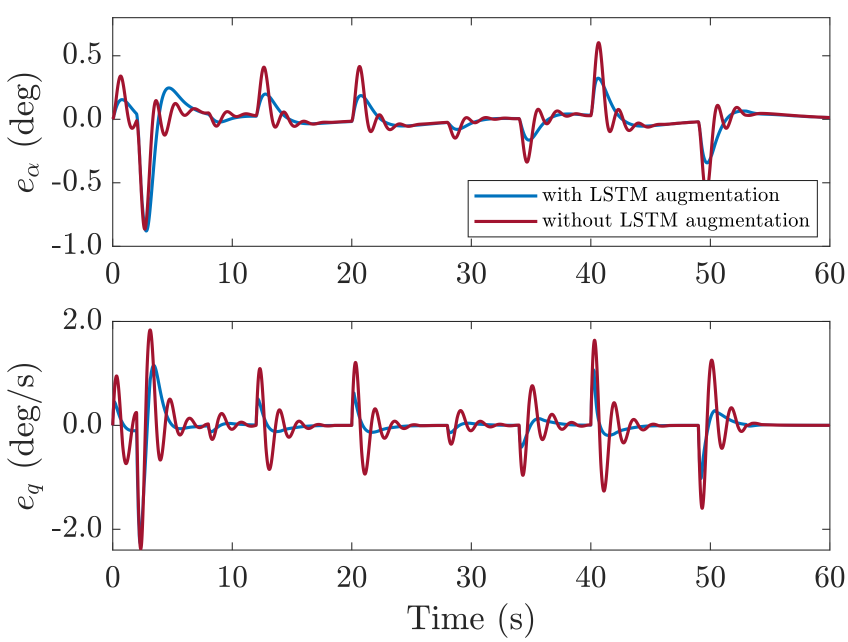

Figure 2.17: Tracking error with and without LSTM augmentation, in the presence of high uncertainty.

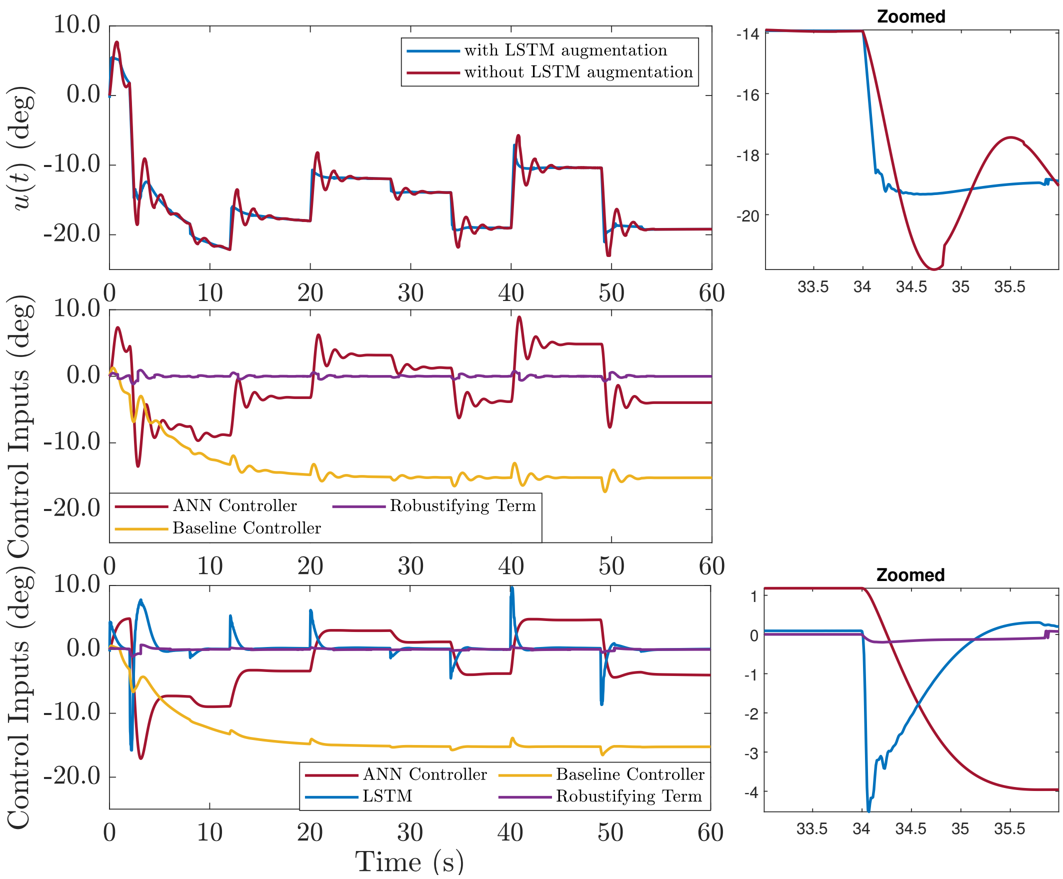

Figure 2.18: Control inputs in the presence of high uncertainty. Top: Total control input. Middle: The contributions of individual control inputs without the LSTM. Bottom: The contributions of individual control inputs with the LSTM.

Chapter 3

Human-in-the-Loop Performance Evaluation of an Adaptive Control Framework with Long Short-Term Memory Augmentation

3.1 Introduction

The pitch rate control task with predefined reference inputs shows a significant improvement in the transient response with the LSTM augmentation as explained in Chapter 2. However, its interactions with a human operator require further examination. It is well-known that adaptive controllers and pilots can sometimes interact in undesired ways [18], [19]. In fact, statistics show that pilot error and adverse interactions between pilots and flight control systems account for nearly two-thirds of all aviation accidents [20]. Given this, it is crucial to thoroughly assess how effectively the adaptive controller and LSTM network align with the pilot's intended inputs. To address this, we have performed a series of human-in-the-loop experiments.

Another goal of this study is to investigate the effect of enhancing the ANN controller with the LSTM network on the overall closed-loop performance. To understand this effect, simulations are performed using a human pilot model following a continuously changing pitch reference. With these simulations, the controller's performance is evaluated under different uncertainties and reference signals. It is shown that although the ANN controller performs reasonably well under certain conditions, it fails to provide adequate performance without the LSTM network enhancement when the uncertainties get larger. We have also demonstrated that LSTM enhancement dramatically improves the transient response in human-in-the-loop experiments.

This study also aims to explore the impact of integrating an LSTM network into the ANN controller on the overall closed-loop performance. To assess this effect, simulations are conducted using a human pilot model following a dynamically changing pitch reference. These simulations evaluate the controller's performance under varying uncertainties and reference signals. The results indicate that while the ANN controller performs adequately in some scenarios, it struggles to maintain satisfactory performance without LSTM network when the magnitude of the uncertainties increase. Additionally, it has been demonstrated that incorporating the LSTM significantly improves the transient response in human-in-the-loop experiments. This work is published in our conference paper [51].

The structure of this chapter is as follows: Section II provides an explanation of the human pilot model, and the training method for the LSTM network. Section III presents the simulation and experimental results. Finally, Section IV offers a summary of the findings.

3.2 Problem Formulation

3.2.1 Pilot Model

We use a generic model to represent the pilot as [52]

$$\dot{\xi}(t) = A_{\mathrm{h}}\xi(t) + B_{\mathrm{h}}e_r(t-\tau), \quad \xi(0) = \xi_0, \tag{3.1a}$$

$$y_h(t) = C_{\mathrm{h}}\xi(t) + D_{\mathrm{h}}e_r(t-\tau). \tag{3.1b}$$

Here, $\xi(t) \in \mathbb{R}^{n_\xi}$ represents the internal human state vector, $\tau \in \mathbb{R}_+$ denotes the internal human time delay, $A_{\mathrm{h}} \in \mathbb{R}^{n_\xi \times n_\xi}, B_{\mathrm{h}} \in \mathbb{R}^{n_\xi \times n_{\mathrm{r}}}, C_{\mathrm{h}} \in \mathbb{R}^{n_{\mathrm{c}} \times n_\xi}, D_{\mathrm{h}} \in \mathbb{R}^{n_{\mathrm{c}} \times n_{\mathrm{r}}}$ and $y_h(t) \in \mathbb{R}^{n_{\mathrm{c}}}$ represents the human output, which serves as the commanded input to the inner loop architecture.

The input to the human dynamics is defined as

$$e_r(t) \triangleq r(t) - E_{\mathrm{h}}x(t), \tag{3.2}$$

where $e_r(t) \in \mathbb{R}^{n_{\mathrm{r}}}$ represents the output tracking error, $r(t) \in \mathbb{R}^{n_{\mathrm{r}}}$ is the system reference (assumed to be bounded), $x(t) \in \mathbb{R}^n$ denotes the state vector (detailed below), and $E_{\mathrm{h}} \in \mathbb{R}^{n_{\mathrm{r}} \times n}$ is the state selection matrix.

3.2.2 Plant Dynamics with a Baseline Controller

Consider the plant dynamics

$$\dot{x}(t) = Ax(t) + B(u(t) + f(x(t))), \tag{3.3a}$$

$$y(t) = C^T x(t), \tag{3.3b}$$

where $x(t) \in \mathbb{R}^n$ is the state vector, $u(t) \in \mathbb{R}^m$ is the control input, $f(x(t)) : \mathbb{R}^n \to \mathbb{R}^m$ represents a state-dependent matched uncertainty, $A \in \mathbb{R}^{n\times n}$, $B \in \mathbb{R}^{n\times m}$, and $C \in \mathbb{R}^{n\times s}$ are known, with A being the system matrix, B the control input matrix, and C the output matrix. The output of interest is $y(t) \in \mathbb{R}^s$. It is assumed that the pair (A, B) is controllable.

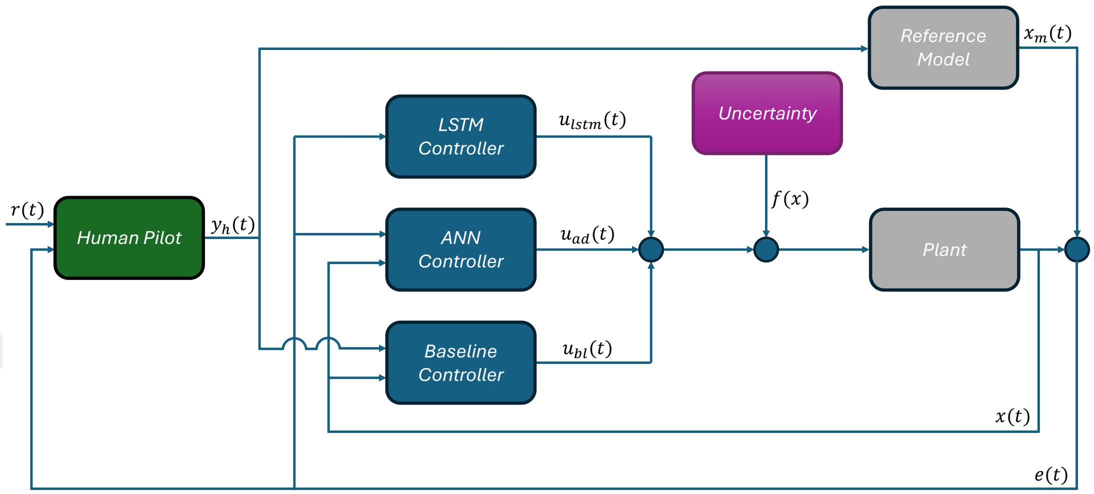

Figure 3.1: Block diagram of the overall control architecture.

Assumption 1: The uncertainty $f(x(t))$ in (3.3) is continuous over a known compact set $S \triangleq \{x(t) : \|x(t)\| \leq b_x\} \subset \mathbb{R}^n$.

Based on Assumption 1, a multi-layer neural network (NN) of the following form can be used to approximate $f(x(t))$ as

$$f(x(t)) = W^T \bar{\sigma}(V^T \bar{x}(t)) + \varepsilon(x(t)), \tag{3.4}$$

with

$$\|\varepsilon(x(t))\| \leq \varepsilon_N \quad , \quad \forall x \in S, \tag{3.5}$$

where the NN error vector $\varepsilon(x(t))$ in $S \subset \mathbb{R}^n$ is bounded by a known positive constant ε_N. The NN input and the hidden layer output are

$$\bar{x}(t) \triangleq \begin{bmatrix} x(t)^T & 1 \end{bmatrix}^T \in \mathbb{R}^{n+1}, \tag{3.6a}$$

$$\bar{\sigma}(V^T \bar{x}(t)) \triangleq \begin{bmatrix} \sigma(V^T \bar{x}(t)) & 1 \end{bmatrix}^T \in \mathbb{R}^{n_h+1}, \tag{3.6b}$$

respectively. The nonlinear activation function $\sigma(.) : \mathbb{R}^{n_h} \to \mathbb{R}^{n_h}$ is a bounded function, typically a sigmoid or tanh. The weight matrices V and W incorporate bias terms, with unity elements added to account for these biases. For simplicity, we omit the overbar notation and rewrite (3.4) as

$$f(x(t)) = W^T \sigma(V^T x(t)) + \varepsilon(x(t)), \tag{3.7}$$

where the weight matrices' dimensions are loosely defined as $W \in \mathbb{R}^{n_h \times m}$ and $V \in \mathbb{R}^{n \times n_h}$.

Assumption 2: The unknown ideal weights W and V are bounded by known positive constants such that $\|V\|_F \leq V_M$ and $\|W\|_F \leq W_M$ [14].

The baseline control input, $u_{bl}(t) \in \mathbb{R}^m$, is defined as

$$u_{bl}(t) = -L_x x(t) + L_r y_h(t), \tag{3.8}$$

where $y_h(t) \in \mathbb{R}^s$ represents the reference signal for the inner loop provided by the human pilot. It is assumed that $L_x \in \mathbb{R}^{m \times n}$ can be chosen such that the matrix $A_m \triangleq A - BL_x$ is stable. Additionally, the reference input $y_h(t)$ is bounded due to manipulator's limits. By defining $B_m \triangleq BL_r$, where $L_r \in \mathbb{R}^{m \times s}$, a reference model can be established as

$$\dot{x}_m(t) = A_m x_m(t) + B_m y_h(t). \tag{3.9}$$

For a constant y_h, at steady state,

$$A_m x_m(\infty) + B_m y_h = 0, \tag{3.10a}$$

$$x_m(\infty) = -A_m^{-1} B L_r y_h. \tag{3.10b}$$

Therefore, the plant output $y(t)$ takes the form $y(\infty) = -C^T A_m^{-1} B L_r y_h$ once $\lim_{t\to\infty} x(t) = x_m(t)$. Selecting

$$L_r = -\left(C^T A_m^{-1} B\right)^{-1}, \tag{3.11}$$

results in $\lim_{t\to\infty} y(t) = y_h$.

The state tracking error is defined as

$$e(t) = x(t) - x_m(t). \tag{3.12}$$

The control signal is given as

$$u(t) = u_{bl}(t) + u_{ad}(t) + u_{lstm}(t) + v(t), \tag{3.13}$$

where $u_{bl}(t)$ is the baseline controller, $u_{ad}(t)$ is an adaptive neural network (ANN) controller is matched uncertainty $f(x(t))$, $v(t)$ is a robustifying term (see Chapter 2), and $u_{lstm}(t)$ is the output of the LSTM network.

3.2.3 LSTM Network Training

The closed-loop system shown in Fig. 3.1 restricts the use of pre-existing data for training the LSTM network. Due to the system's dynamic nature, the training data changes after each iteration. Thus, once the LSTM network weights are initialized, we place it in a closed-loop environment that includes the plant, the baseline controller, and the ANN controller. We then run a simulation to gather a set of data, use it to train the network, and apply the updated weights in the subsequent simulation.

As in Chapter 2, rather than directly estimating the uncertainty, we train the LSTM network to predict and correct the estimation error of the ANN, thereby maintaining the stability of the system.

$$\tilde{f}(x(t)) \triangleq f(x(t)) - \hat{f}(x(t)), \tag{3.14}$$

where $f(x(t))$ and $\hat{f}(x(t))$ are given in (3.7) and (2.14). Then, the *truth* for the LSTM is selected as

$$y(t) = -\tilde{f}(x(t)). \tag{3.15}$$

Thus, the system receives a signal from the LSTM network that is an estimate of the truth and is identified as

$$u_{lstm}(t) = \hat{y}(t). \tag{3.16}$$

During training, $f_{train}(x(t))$ is introduced into the system. A combination of sinusoidal waves is used as a reference signal, and we expect the LSTM network to learn $\tilde{f}_{train}$ as defined in (3.14). The LSTM network is trained using the tracking error, defined in (3.12), as its input. Training the network in this dynamic setting allows the LSTM to generalize effectively to previously unseen uncertainty functions.

3.2.4 Stability Guarantee

Theorem 1: Given the uncertain plant dynamics described by (3.3) and assuming that Assumptions 1 and 2 hold, along with the reference model (3.9) and the control input (3.13), which includes the baseline controller (3.8), the ANN controller (2.13), (2.14), and (2.15), the robustifying term (2.18) and (2.19), as well as the LSTM network output (2.26), then if $y_h(t)$ remains bounded and $x(0) \in S$ (see Assumption 1), the solution $(e(t), \tilde{W}(t), \tilde{V}(t))$, where $\tilde{W}(t) \triangleq W - \hat{W}(t)$ and $\tilde{V}(t) \triangleq V - \hat{V}(t)$, is *uniformly ultimately bounded* and converges to a compact set.

Proof. See Appendix B.1. □

Theorem 2: If A_h in (3.1a) is Hurwitz, and the reference $r(t)$ in (3.2) is bounded, all system signals are bounded.

Proof. See Appendix B.2. □

3.3 Simulations and Experiments

In this section, we focus on a flight control task where the pilot is responsible for guiding the aircraft to follow a specified pitch angle reference. The task is simulated using both a pilot model and real human participants. Two distinct pitch angle references are employed: one with a rapid change and another with a slower transition. These references are

$$r(t) = \Sigma_1^7 A_i \sin(f_i t), \tag{3.17a}$$

$$A = 360/\pi(2\,\mathrm{rand}(7,1) - 1), \tag{3.17b}$$

$$f = \mathrm{rand}(7,1), \tag{3.17c}$$

and

$$r(t) = \Sigma_1^7 A_i \sin(f_i t), \tag{3.18a}$$
$$A = 360/\pi(2\,\mathrm{rand}(7,1) - 1), \tag{3.18b}$$
$$f = 2\,\mathrm{rand}(7,1), \tag{3.18c}$$

which are denoted as "slow" and "fast", respectively, where "rand$(7,1)$" function generates uniformly distributed random numbers (between 0 and 1) of the size 7-by-1. Therefore, $A \in \mathbb{R}^{7\times1}$, $A_i \in [-360/\pi \quad 360/\pi]$, and $f \in \mathbb{R}^{7\times1}$ for both cases. For the slow reference case $f_i \in [0 \quad 1]$, whereas for the fast reference case $f_i \in [0 \quad 2]$.

In the outer loop architecture, the plant is assumed to be controlled by a pilot model described in [53] as

$$k_p \frac{T_p s + 1}{T_z s + 1} e^{-\tau s}, \tag{3.19}$$

where k_p, T_p and T_z are positive constants, and τ represents the time delay in the pilot's reaction. For the simulations, the human pilot model parameters are set to $T_p = 1$, $T_z = 2$, and $\tau = 0.3$. The pilot gain k_p is chosen as 0.5 for the slow reference case in (3.17), and 0.9 for the fast reference case in (3.18).

The plant dynamics are based on the longitudinal flight dynamics of a Boeing 747, cruising in level flight at an altitude of 40kft and a velocity of 774ft/sec. The corresponding state and control input matrices for this plant are given as [54]

$$A_n = \begin{bmatrix} -0.003 & 0.039 & 0 & -0.322 \\ -0.065 & -0.319 & 7.740 & 0 \\ 0.020 & -0.101 & -0.429 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}, \tag{3.20a}$$
$$B = \begin{bmatrix} 0.010 & -0.180 & -1.160 & 0 \end{bmatrix}^T. \tag{3.20b}$$

The state vector is defines as

$$x(t) = \begin{bmatrix} x_1(t) & x_2(t) & x_3(t) & x_4(t) \end{bmatrix}^T, \tag{3.21}$$

where $x_1(t)$ and $x_2(t)$ represent the aircraft's velocity components along the x and z axes (in ft/sec), $x_3(t) = q(t)$ denotes the pitch rate (in rad/sec), and $x_4(t) = \theta(t)$

corresponds to the pitch angle of the aircraft (in radians). The control input $u(t)$ refers to the elevator deflection (in radians). The elevator deflection is constrained by saturation limits of $+17/-23$ (deg) and $+37/-37$ (deg/s) [49].

The feedback gain L_x in (3.8) is determined using an LQR controller, where the cost matrices are chosen as $Q_{LQR} = \text{diag}\left([0.001, 0.01, 10, 0.1]\right)$ and $R_{LQR} = 10$.

The ANN controller is configured with a single hidden layer containing eight neurons. The learning rates for the controller are both set to $F = G = 300$, and the Lyapunov matrix Q is set as $Q = 10^{-3}Q_{LQR}$. The weights $(\hat{W})$ and biases $(\hat{b}_w)$ of the output layer are initialized to zero, while the initial inner weights $(\hat{V})$ and biases $(\hat{b}_v)$ are randomly generated within the range of 0 to 1. The gain parameters k_z in (2.18) and κ in (2.15) are both set to 0.

The LSTM network used in this study has a single hidden layer with 128 neurons. Given that the plant has four states (as described in (3.21)), the network takes four state-tracking error components as input. The weights are initialized using Xavier initialization [50]. The network is trained using stochastic gradient descent with the Adam optimizer [55], incorporating an L2 regularization value of 0.0001, a gradient decay rate of 0.9, and a squared gradient decay rate of 0.999. The simulation time step is set to 0.01 seconds, and the minibatch size is 1.

During training, the nonlinear uncertainty

$$f_{train}(x) = \begin{cases} 0.1x_3 + 0.2x_1 & \text{if } 0 \leq t < 5 \\ 0.05\exp(x_3) + 0.5x_2 & \text{if } 5 \leq t < 10 \\ -0.1(x_4 + x_3)^2 & \text{if } 10 \leq t < 15 \\ -0.1\cos(x_4) & \text{if } 15 \leq t < 20 \\ 0.2(x_4 - x_1) & \text{if } 20 \leq t < 25 \\ x_4/(2 + |x_3|) & \text{if } 25 \leq t < 30 \\ -\sin(2x_2x_3)/10 & \text{if } 30 \leq t < 35 \\ 2(\sin(2x_3x_2) + x_1) & \text{if } 35 \leq t < 45 \\ \|x\| & \text{if } 45 \leq t < 55 \\ (x_3 + x_4)/3 & \text{if } 55 \leq t \leq 60 \end{cases} \tag{3.22}$$

is used. To enhance the diversity of the training data, we trained the network using both f_{train} and a scaled-down version, $0.1f_{train}$. The nonlinear uncertainty applied during the tests is defined as

$$f_{test}(x) = \begin{cases} 0 & \text{if } 0 \leq t < 7 \\ \exp(\sin(x_1))/5 & \text{if } 7 \leq t < 12 \\ -0.1\exp(x_1) & \text{if } 12 \leq t < 19 \\ 0.1\sin(\|x\|) & \text{if } 19 \leq t < 28 \\ 0.05(x_1x_2) & \text{if } 28 \leq t < 34 \\ -\|x\|^2/1.5 & \text{if } 34 \leq t \leq 41 \\ \sin(x_1x_3) & \text{if } 41 \leq t \leq 49 \\ 0.1(x_4)^2 & \text{if } 49 \leq t \leq 53 \\ 2\cos(x_2)x_3 & \text{if } 53 \leq t \leq 60 \end{cases}, \tag{3.23}$$

where the sub-function types and durations differ from those of the uncertainty used in the training.

During training, the loss function for the LSTM network is selected as

$$L(y, \hat{y}) = \frac{1}{N} \sum_{i=0}^{N} (y - \hat{y})^2, \tag{3.24}$$

where y is the truth of the LSTM network (see (3.15)), $\hat{y}$ is the network output (see (3.16)), and N is the number of data in the data set.

3.3.1 Simulations with a pilot model

This section discusses the impact of the LSTM network augmentation on the system when subjected to a fast reference input, as in (3.18).

Figures 3.2 and 3.3 shows the state tracking and control input curves under low uncertainty, specifically $0.1 f_{test}$ (3.23). Although the LSTM-augmented system demonstrates a reduction in oscillations within the error dynamics, the results indicate that, in the presence of small uncertainties, the tracking performance of the system is similar regardless of LSTM augmentation.

Compared to the scenario with a small uncertainty, Figure 3.4 shows that the closed-loop system achieves an improved performance with LSTM augmentation when the uncertainty is significant (f_{test}). The ANN-only scenario exhibits large oscillations; tracking curves deviate from the reference pitch rate and angle after $40s$, leading to excessive oscillations. On the other hand, these oscillations are eliminated by the LSTM augmented system, ensuring steady performance. Additional analysis of this result is shown in Fig. 3.5, which indicates that the LSTM network augmentation results in smoother control signals. In contrast to the ANN-only scenario, the controller cancels the uncertainties in the presence of the LSTM augmentation while remaining within the actuator saturation range ($+17/-23$ deg).

3.3.2 Human-in-the-loop experiments

he experiments, participants use a monitor and a Logitech Extreme 3D Pro joystick as the control interface for performing pitch reference tracking tasks. These experiments involved five individuals from Bilkent University's Mechanical Engineering Department. Fig. 3.6 shows all five participants' performance results. The tracking performance is evaluated by calculating the root mean square errors (RMSE) between the reference pitch angles and the actual pitch angles as RMSE $= \sqrt{\sum_{i=1}^{N} (\theta_i - \theta_r)^2 / N}$,

where subscript i denotes the i^{th} participant. Table 3.1 presents the RMSE values for four different scenarios, with each column representing the results of individual participants. The cells labeled "LOC" indicate instances of loss of control. The abbreviations "L," "H," "S," and "F" correspond to "low," "high," "slow," and "fast," respectively, while "R" and "U" stand for "reference" and "uncertainty."

As shown in Figure 3.6 and Table 3.1, the LSTM does not significantly enhance performance when uncertainty is low and the reference signal is slow. However, as uncertainty increases or the reference signal speeds up, the tracking performance of the ANN alone deteriorates considerably. In the ANN-only system, several participants lost control when using the fast reference, with most loss of control events occurring under high uncertainty. One participant also experienced a loss of control even under low uncertainty. When the LSTM augmentation is applied, no loss of control events are observed.

3.4 Summary

This research explores the tracking capabilities of an adaptive neural network control structure enhanced with Long Short-Term Memory (LSTM) through simulations and human-in-the-loop experiments. The findings demonstrate that LSTM augmentation leads to a substantial improvement in tracking performance. Future work will involve increasing the number of participants to conduct a thorough statistical analysis of the performance gains achieved with LSTM augmentation.

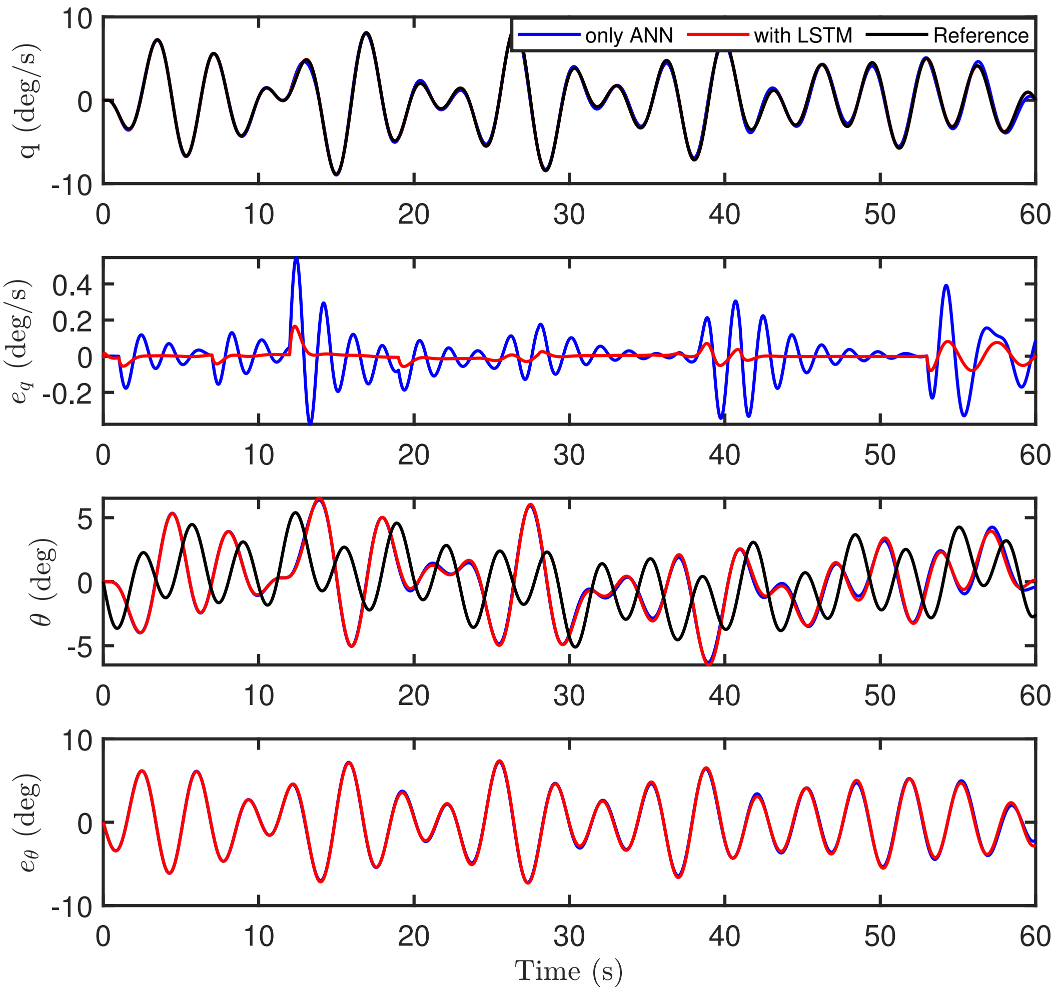

Figure 3.2: Tracking performance and errors of pitch rate q and pitch angle θ with and without LSTM augmentation, in the presence of low uncertainty.

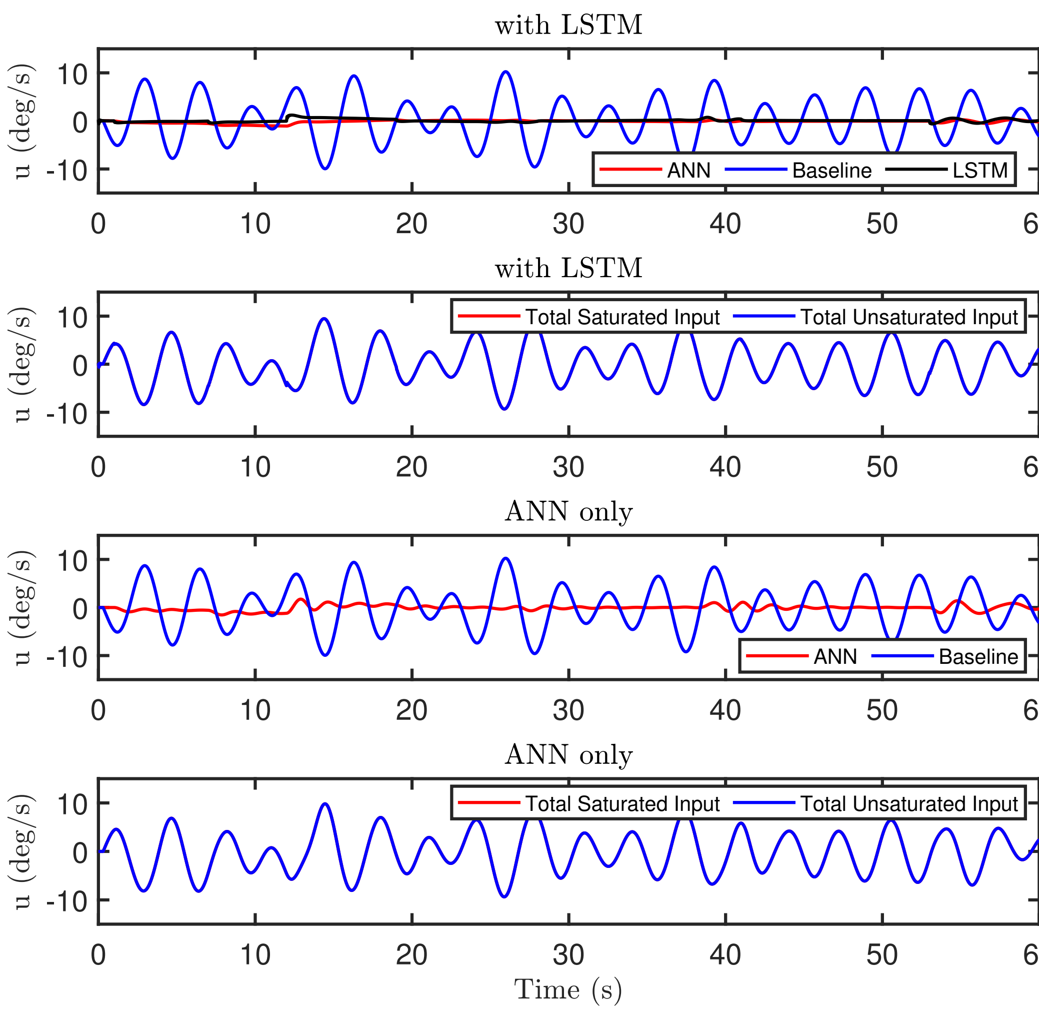

Figure 3.3: Control inputs for low uncertainty. LSTM-augmented system: individual inputs (top), total input before/after saturation (2nd). ANN-only system: individual inputs (2nd from bottom), total inputs before/after saturation (bottom).

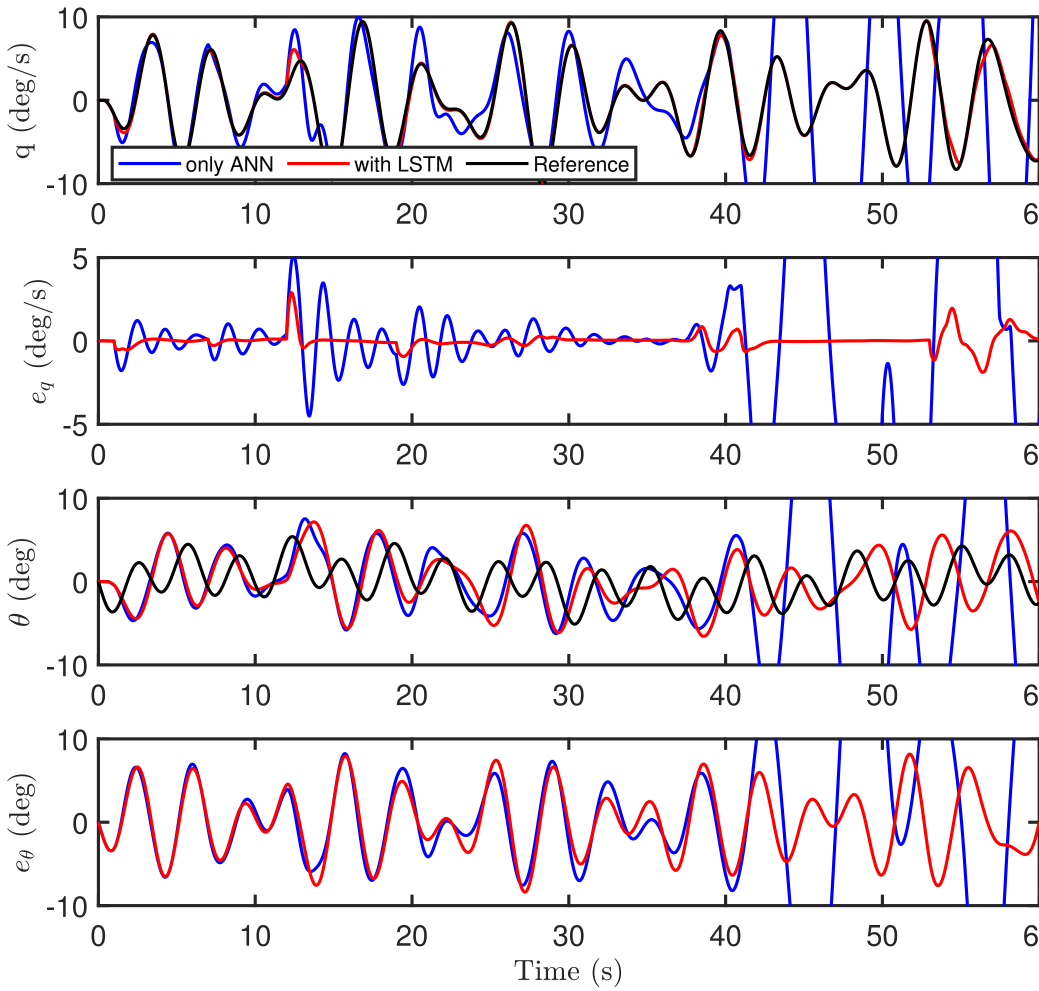

Figure 3.4: Tracking performance and errors of pitch rate q and pitch angle θ with and without LSTM augmentation, in the presence of high uncertainty.

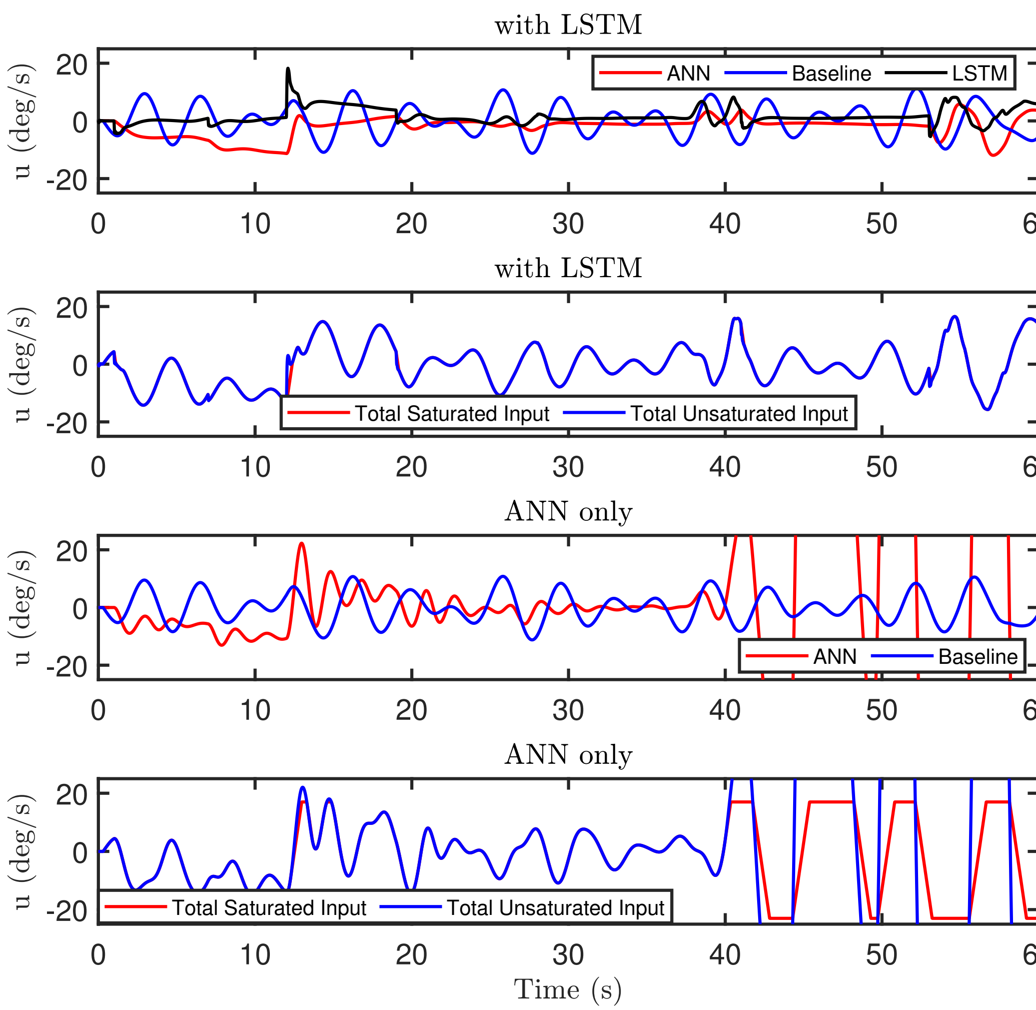

Figure 3.5: Control inputs for high uncertainty. LSTM-augmented system: individual inputs (top), total input before/after saturation (2nd). ANN-only system: individual inputs (2nd from bottom), total inputs before/after saturation (bottom).

Table 3.1: RMSE of pitch angle (θ) tracking for low uncertainty (LU), high uncertainty (HU), slow reference (SR), and fast reference (FR). LOC represents a loss-of-control event.

Conditions		P1	P2	P3	P4	P5	Average
LU-SR	ANN	0.64	1.40	1.26	0.58	0.48	0.69
	LSTM	0.64	0.97	1.16	0.59	0.51	0.61
LU-FR	ANN	2.75	3.37	3.81	3.03	LOC	3.24
	LSTM	1.91	3.93	4.24	2.57	2.51	2.54
HU-SR	ANN	0.98	1.44	1.69	0.93	0.84	0.92
	LSTM	0.64	0.75	1.40	0.68	0.57	0.63
HU-FR	ANN	LOC	LOC	3.44	LOC	LOC	3.44
	LSTM	2.19	3.09	3.58	2.74	2.82	2.53

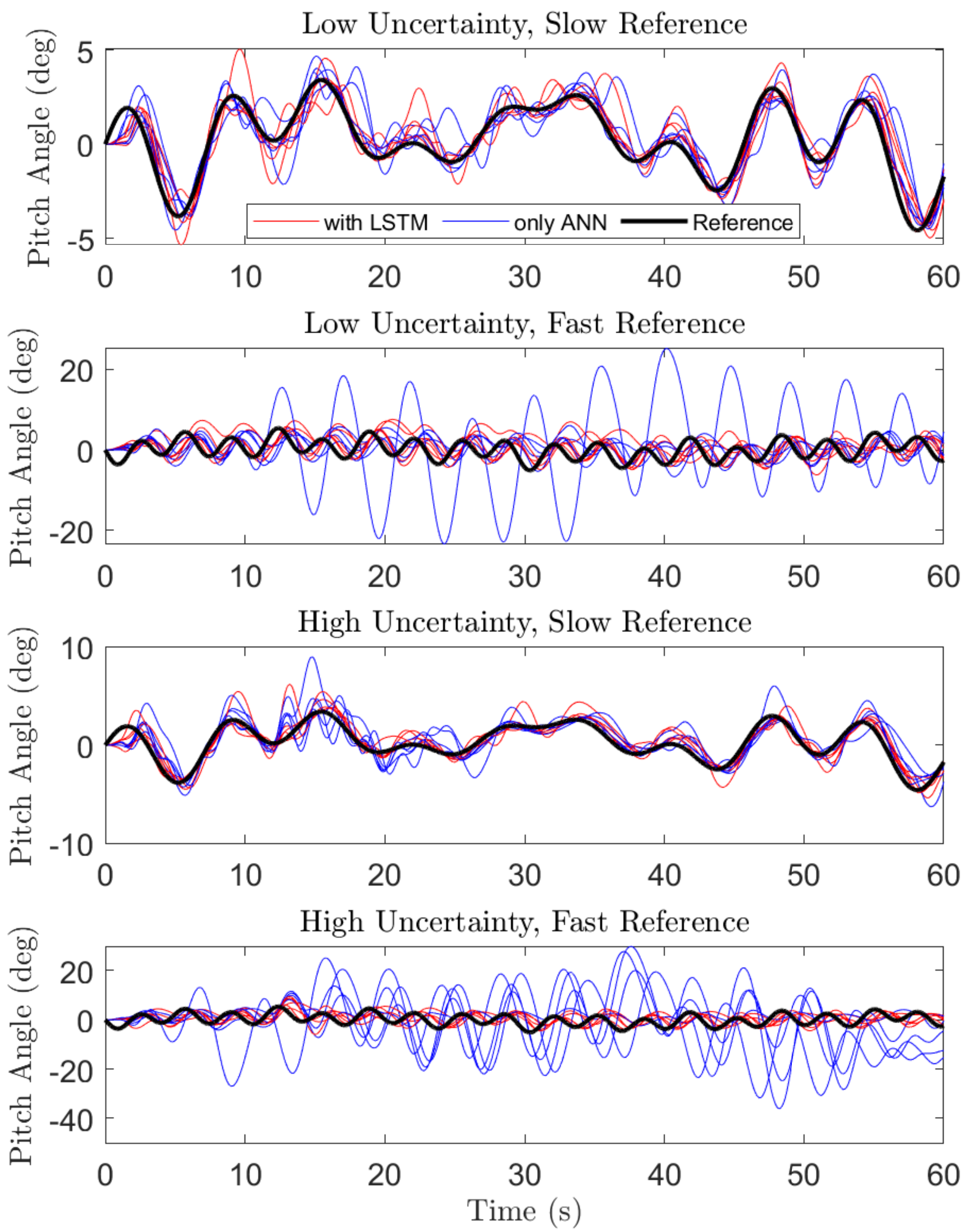

Figure 3.6: Individual participants' tracking performance of pitch angle θ with and without LSTM augmentation, for different scenarios.

Chapter 4

Conclusion

In conclusion, this thesis introduces a novel adaptive control framework that integrates an LSTM network with a traditional ANN controller, significantly improving transient response and overall closed-loop performance. Through a combination of simulations and human-in-the-loop experiments, it is demonstrated that the LSTM enhancement effectively addresses the shortcomings of the ANN controller, particularly in scenarios involving large uncertainties and dynamically changing reference signals. This thesis also emphasizes the importance of assessing how adaptive controllers interact with human operators, particularly in critical tasks like pitch rate control in aviation. Given the historical challenges associated with pilot-controller interactions and the significant role these interactions play in aviation safety, the findings underscore the value of enhancing adaptive control systems with advanced neural network structures such as LSTM. By improving the alignment between the controller and pilot inputs, this approach not only enhances system stability and performance but also contributes to reducing the risks of adverse pilot-controller interactions. The insights gained from this work pave the way for future exploration into the integration of neural networks with adaptive control systems, with a focus on optimizing human-machine interactions in safety-critical environments.

Bibliography

[1] B. Macukow, "Neural networks–state of art, brief history, basic models and architecture," in *IFIP international conference on computer information systems and industrial management*, pp. 3–14, Springer, 2016.

[2] C. C. Aggarwal *et al.*, "Neural networks and deep learning," *Springer*, vol. 10, pp. 978–3, 2018.

[3] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, *et al.*, "Imagenet large scale visual recognition challenge," *International journal of computer vision*, vol. 115, no. 3, pp. 211–252, 2015.

[4] D. W. Otter, J. R. Medina, and J. K. Kalita, "A survey of the usages of deep learning for natural language processing," *IEEE transactions on neural networks and learning systems*, vol. 32, no. 2, pp. 604–624, 2020.

[5] G. Van Houdt, C. Mosquera, and G. Nápoles, "A review on the long short-term memory model," *Artificial Intelligence Review*, vol. 53, no. 8, pp. 5929–5955, 2020.

[6] K. Cho, B. Van Merriënboer, D. Bahdanau, and Y. Bengio, "On the properties of neural machine translation: Encoder-decoder approaches," *arXiv preprint arXiv:1409.1259*, 2014.

[7] Z. C. Lipton, J. Berkowitz, and C. Elkan, "A critical review of recurrent neural networks for sequence learning," *arXiv preprint arXiv:1506.00019*, 2015.

[8] K. Arulkumaran, M. P. Deisenroth, M. Brundage, and A. A. Bharath, "A brief survey of deep reinforcement learning," *arXiv preprint arXiv:1708.05866*, 2017.

[9] F. L. Lewis, D. Vrabie, and K. G. Vamvoudakis, "Reinforcement learning and feedback control: Using natural decision methods to design optimal adaptive controllers," *IEEE Control Systems Magazine*, vol. 32, no. 6, pp. 76–105, 2012.

[10] L. Buşoniu, T. de Bruin, D. Tolić, J. Kober, and I. Palunko, "Reinforcement learning for control: Performance, stability, and deep approximators," *Annual Reviews in Control*, vol. 46, pp. 8–28, 2018.

[11] A. J. Calise, N. Hovakimyan, and M. Idan, "Adaptive output feedback control of nonlinear systems using neural networks," *Automatica*, vol. 37, no. 8, pp. 1201–1211, 2001.

[12] L. Chen and K. S. Narendra, "Nonlinear adaptive control using neural networks and multiple models," *Automatica*, vol. 37, no. 8, pp. 1245–1255, 2001.

[13] C. Kwan and F. L. Lewis, "Robust backstepping control of nonlinear systems using neural networks," *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans*, vol. 30, no. 6, pp. 753–766, 2000.

[14] F. L. Lewis, A. Yesildirek, and K. Liu, "Multilayer neural-net robot controller with guaranteed tracking performance," *IEEE Transactions on neural networks*, vol. 7, no. 2, pp. 388–399, 1996.

[15] A. M. Annaswamy and A. L. Fradkov, "A historical perspective of adaptive control and learning," *Annual Reviews in Control*, vol. 52, pp. 18–41, 2021.

[16] T. G. Thuruthel, E. Falotico, F. Renda, and C. Laschi, "Model-based reinforcement learning for closed-loop dynamic control of soft robotic manipulators," *IEEE Transactions on Robotics*, vol. 35, no. 1, pp. 124–134, 2018.

[17] D. Kalashnikov, A. Irpan, P. Pastor, J. Ibarz, A. Herzog, E. Jang, D. Quillen, E. Holly, M. Kalakrishnan, V. Vanhoucke, *et al.*, "Scalable deep reinforcement learning for vision-based robotic manipulation," in *Conference on Robot Learning*, pp. 651–673, PMLR, 2018.

[18] D. Klyde, C.-Y. Liang, D. Alvarez, N. Richards, R. Adams, and B. Cogan, "Mitigating unfavorable pilot interactions with adaptive controllers in the presence of failures/damage," in *AIAA Atmospheric Flight Mechanics Conference*, p. 6538, 2011.

[19] A. C. Trujillo, I. M. Gregory, and L. E. Hempley, "Adaptive controller effects on pilot behavior," in *Proceedings of the IEEE 2014 International Conference on Systems, Man, and Cybernetics*, NASA Rept. No. NL1676L-18398, San Diego, CA, Oct. 2014, p. 6.

[20] D. T. McRuer, "Aviation safety and pilot control: Understanding and preventing unfavorable pilot-vehicle interactions," 1997.

[21] S. N. Kumpati and P. Kannan, "Identification and control of dynamical systems using neural networks," *IEEE Transactions on neural networks*, vol. 1, no. 1, pp. 4–27, 1990.

[22] R. M. Sanner and J.-J. E. Slotine, "Gaussian networks for direct adaptive control," in *1991 American control conference*, pp. 2153–2159, IEEE, 1991.

[23] G. A. Rovithakis and M. A. Christodoulou, "Adaptive control of unknown plants using dynamical neural networks," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 24, no. 3, pp. 400–412, 1994.

[24] M. M. Polycarpou, "Stable adaptive neural control scheme for nonlinear systems," *IEEE Transactions on Automatic control*, vol. 41, no. 3, pp. 447–451, 1996.

[25] B. Ren, S. S. Ge, K. P. Tee, and T. H. Lee, "Adaptive neural control for output feedback nonlinear systems using a barrier lyapunov function," *IEEE Transactions on Neural Networks*, vol. 21, no. 8, pp. 1339–1345, 2010.

[26] R. Sun, M. L. Greene, D. M. Le, Z. I. Bell, G. Chowdhary, and W. E. Dixon, "Lyapunov-based real-time and iterative adjustment of deep neural networks," *IEEE Control Systems Letters*, vol. 6, pp. 193–198, 2021.

[27] A. Patkar and A. M. Annaswamy, "An adaptive controller for a class of nonlinear plants based on neural networks and convex parameterization," in *2020 59th IEEE Conference on Decision and Control (CDC)*, pp. 126–131, IEEE, 2020.

[28] O. S. Patil, D. M. Le, M. L. Greene, and W. E. Dixon, "Lyapunov-derived control and adaptive update laws for inner and outer layer weights of a deep neural network," *IEEE Control Systems Letters*, vol. 6, pp. 1855–1860, 2021.

[29] A. Graves, G. Wayne, and I. Danihelka, "Neural turing machines," *arXiv preprint arXiv:1410.5401*, 2014.

[30] D. Muthirayan and P. Khargonekar, "Memory augmented neural network adaptive controllers: Performance and stability," *IEEE Transactions on Automatic Control*, 2022.

[31] F. Visin, M. Ciccone, A. Romero, K. Kastner, K. Cho, Y. Bengio, M. Matteucci, and A. Courville, "Reseg: A recurrent neural network-based model for semantic segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pp. 41–48, 2016.

[32] G. Mesnil, Y. Dauphin, K. Yao, Y. Bengio, L. Deng, D. Hakkani-Tur, X. He, L. Heck, G. Tur, D. Yu, *et al.*, "Using recurrent neural networks for slot filling in spoken language understanding," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 23, no. 3, pp. 530–539, 2014.

[33] N. Mohajerin and S. L. Waslander, "Multistep prediction of dynamic systems with recurrent neural networks," *IEEE transactions on neural networks and learning systems*, vol. 30, no. 11, pp. 3370–3383, 2019.

[34] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[35] A. Graves, "Long short-term memory," *Supervised sequence labelling with recurrent neural networks*, pp. 37–45, 2012.

[36] W. Żylski and P. Gierlak, "Verification of multilayer neural-net controller in manipulator tracking control," in *Solid State Phenomena*, vol. 164, pp. 99–104, Trans Tech Publ, 2010.

[37] Z. Yao, J. Yao, and W. Sun, "Adaptive rise control of hydraulic systems with multilayer neural-networks," *IEEE Transactions on Industrial Electronics*, vol. 66, no. 11, pp. 8638–8647, 2018.

[38] G. L. Burrola, M. Á. L. Leal, and J. S. Mijares, "Adaptive neural control for a two wheeled inverted pendulum system," in *2022 XXIV Robotics Mexican Congress (COMRob)*, pp. 48–53, IEEE, 2022.

[39] J. Wen, C. Zhao, and Z. Shi, "Lstm-based adaptive robust nonlinear controller design of a single-axis hydraulic shaking table," *IET Control Theory & Applications*, 2022.

[40] E. J. Griffis, O. S. Patil, Z. I. Bell, and W. E. Dixon, "Lyapunov-based long short-term memory (lb-lstm) neural network-based control," *IEEE Control Systems Letters*, 2023.

[41] E. Lavretsky and K. Wise, *Robust and Adaptive Control: With Aerospace Applications*. London, UK: Springer, 2013.

[42] F. Lewis, S. Jagannathan, and A. Yesildirak, *Neural network control of robot manipulators and non-linear systems*. CRC press, 1998.

[43] E. Inanc, Y. Gurses, A. Habboush, and Y. Yildiz, "Neural network adaptive control with long short-term memory augmentation," *American control conference*, 2023.

[44] E. Inanc, Y. Gurses, A. Habboush, Y. Yildiz, and A. M. Annaswamy, "Neural network adaptive control with long short-term memory," *arXiv:2301.02316*, 2023.

[45] T. Yucelen, "Model reference adaptive control," *Wiley Encyclopedia of Electrical and Electronics Engineering*, pp. 1–13, 2019.

[46] B. C. Gruenwald, T. Yucelen, G. De La Torre, and J. A. Muse, "Adaptive control for uncertain dynamical systems with nonlinear reference systems," *International Journal of Systems Science*, vol. 51, no. 4, pp. 687–703, 2020.

[47] S. S. Ge, C. C. Hang, T. H. Lee, and T. Zhang, *Stable adaptive neural network control*, vol. 13. Springer Science & Business Media, 2013.

[48] H. K. Khalil, *Nonlinear Systems*. Upper Saddle River, NJ: Prentice Hall, 3 ed., 2002.

[49] L. Sun, C. C. de Visser, Q. P. Chu, and W. Falkena, "Hybrid sensor-based backstepping control approach with its application to fault-tolerant flight control," *Journal of Guidance, Control, and Dynamics*, vol. 37, no. 1, pp. 59–71, 2014.

[50] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pp. 249–256, JMLR Workshop and Conference Proceedings, 2010.

[51] M. Y. Uzun, E. Inanc, A. Habboush, and Y. Yildiz, "Human-in-the-loop performance evaluation of an adaptive control framework with long short-term memory augmentation," in *European Control Conference*, 2023.

[52] T. Yucelen, Y. Yildiz, R. Sipahi, E. Yousefi, and N. Nguyen, "Stability limit of human-in-the-loop model reference adaptive control architectures," *Int. J. Control*, vol. 91, no. 10, pp. 2314–2331, 2018.

[53] B. J. Bacon and D. K. Schmidt, "An optimal control approach to pilot/vehicle analysis and the neal-smith criteria," *Journal of Guidance, Control, and Dynamics*, vol. 6, no. 5, pp. 339–347, 1983.

[54] A. E. Bryson, *Control of spacecraft and aircraft*. Princeton, NJ, USA: Princeton Univ. Press, 1994.

[55] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[56] K. S. Narendra and A. M. Annaswamy, *Stable adaptive systems.* Courier Corporation, 2012.

Appendix A

Proofs of Chapter 2

In this section, we provide the proof of Theorem 1. According to Assumption 1, the NN approximation property (2.12) holds when $x_p(t) \in S_p$. The proof provided in this Appendix shows that if $x_p(0) \in S_p$, then $x_p(t)$ evolves within S_p for all $t > 0$, preventing a circular argument. Towards that end, we start, in Appendix A, by deriving error dynamics that hold when $x_p(t) \in S_p$. Useful inequalities are then listed in several remarks, which are employed in the proof that is given in Appendix B, showing that $x_p(t)$ does not leave S_p, once it is initialized in S_p. Therefore, the states of the system given in (3.3) are not assumed to be in S_p for all times, a priory, but it is proven. This is important to prevent a circular argument.

A.1 Error Dynamics and Useful Remarks for $\mathbf{x_p(t) \in S_p}$

We start by substituting the control input (3.13) and the baseline controller (2.6) into (2.4), which yields

$$\dot{x}(t) = A_m x(t) + B_m r(t) + B(u_{ad}(t) + u_{lstm}(t) + f(x_p(t))). \tag{A.1}$$

Using (3.12), (B.2) and (3.9), the state-tracking error dynamics can be written as

$$\dot{e}(t) = A_m e(t) + B(u_{ad}(t) + u_{lstm}(t) + f(x_p(t))), \tag{A.2}$$

which, by using the NN approximation property (2.12), can be written as

$$\dot{e}(t) = A_m e(t) + B(u_{ad}(t) + W^T \sigma(V^T x_p(t)) + u_{lstm}(t) + \varepsilon(x_p(t))). \tag{A.3}$$

It is noted that (A.3) and all subsequent remarks hold when $x_p(t) \in S_p$, while in this paper, we only assume that $x_p(0) \in S_p$. The following remarks are used later in the proof in the next section, where it is shown that once x_p is initialized in S_p, it stays in S_p.

The ANN controller $u_{ad}(t)$ is chosen to be the output of a multi-layer NN, as defined in (2.13) and (2.14). Substituting (2.13) and (2.14) into (A.3), yields

$$\begin{aligned} \dot{e}(t) =& A_m e(t) + B(-\hat{W}^T(t)\sigma(\hat{V}^T(t) x_p(t)) + W^T \sigma(V^T x_p(t)) \\ &+ u_{lstm}(t) + v(t) + \varepsilon(x_p(t))), \end{aligned} \tag{A.4}$$

where $\hat{V}(t) \in \mathbb{R}^{n_h \times n}$ and $\hat{W}(t) \in \mathbb{R}^{m \times n_h}$ are adaptive NN weights serving as estimates for the unknown ideal weights V and W, respectively. The weight estimation errors are defined as

$$\tilde{W}(t) = W - \hat{W}(t), \tag{A.5a}$$

$$\tilde{V}(t) = V - \hat{V}(t). \tag{A.5b}$$

Throughout the remainder of this section, we drop the time dependency notation for simplicity. Further, we denote $\sigma \triangleq \sigma(V^T x_p(t))$ and $\hat{\sigma} \triangleq \sigma(\hat{V}^T x_p(t))$. Then, (A.4) can be written as

$$\dot{e} = A_m e + B(-\hat{W}^T \hat{\sigma} + W^T \sigma + u_{lstm} + v + \varepsilon(x_p)). \tag{A.6}$$

Using (A.5a) in (A.6) yields

$$\dot{e} = A_m e + B(\tilde{W}^T \hat{\sigma} + W^T(\sigma - \hat{\sigma}) + u_{lstm} + v + \varepsilon(x_p)). \tag{A.7}$$

Remark 4. Since $r(t)$ is bounded and A_m is Hurwitz, it follows from (3.9) that

$$\|x_m(t)\| \leq C_m, \tag{A.8}$$

where $C_m > 0$ is a known constant, which depends on the user-defined reference input $r(t)$. Therefore, it follows from (3.12), and the fact that $x(t) \triangleq [x_p(t)^T, x_e(t)^T]^T \in \mathbb{R}^n$, that

$$\|x_p(t)\| \leq \|x(t)\| \leq C_m + \|e(t)\|, \tag{A.9}$$

Remark 5. From the Taylor series expansion

$$\sigma(V^T x_p) = \sigma(\hat{V}^T x_p) + \hat{\sigma}' \tilde{V}^T x_p + O(\tilde{V}^T x_p)^2, \tag{A.10}$$

one can write

$$\sigma(V^T x_p) - \sigma(\hat{V}^T x_p) = \hat{\sigma}' \tilde{V}^T x_p + O(\tilde{V}^T x_p)^2, \tag{A.11}$$

where $\hat{\sigma}'$ is defined in (2.16) and $O(\tilde{V}^T x_p)^2$ denote the higher order terms in the series. Furthermore, since

$$O(\tilde{V}^T x_p)^2 = \sigma(V^T x_p) - \sigma(\hat{V}^T x_p) - \hat{\sigma}' \tilde{V}^T x_p, \tag{A.12}$$

then, for sigmoid and tanh activation functions, $\|\sigma(.)\| \leq C_\sigma$ and $\|\sigma'(.)\| \leq C_{\sigma'}$ for known constants $C_\sigma > 0$ and $C_{\sigma'} > 0$. Hence, using (A.9), the higher-order terms in the Taylor series are bounded by

$$\|O(\tilde{V}^T x_p)^2\| \leq 2C_\sigma + C_{\sigma'} \|\tilde{V}\|_F C_m + C_{\sigma'} \|\tilde{V}\|_F \|e\|. \tag{A.13}$$

Substituting (A.11) for $(\sigma - \hat{\sigma})$ in (A.7), yields

$$\begin{aligned} \dot{e} =& A_m e + B(\tilde{W}^T \hat{\sigma} + W^T \hat{\sigma}' \tilde{V}^T x_p \\ &+ W^T O(\tilde{V}^T x_p)^2 + u_{lstm} + v + \varepsilon(x_p)), \end{aligned} \tag{A.14}$$

which, by using (A.5a) and (A.5b), can be written as

$$\dot{e} = A_m e + B(\tilde{W}^T(\hat{\sigma} - \hat{\sigma}' \hat{V}^T x_p) + \hat{W}^T \hat{\sigma}' \tilde{V}^T x_p + w + u_{lstm} + v), \tag{A.15}$$

where

$$w \triangleq \tilde{W}^T \hat{\sigma}' V^T x_p + W^T O(\tilde{V}^T x_p)^2 + \varepsilon(x_p). \tag{A.16}$$

Remark 6. Let

$$Z \triangleq \begin{bmatrix} W & 0_{n_h \times n_h} \\ 0_{n_p \times m} & V \end{bmatrix} \in \mathbb{R}^{(n_h+n_p)\times(m+n_h)}, \tag{A.17}$$

then, using (A.17) and Assumption 2, one can write

$$\|Z\|_F^2 = \|W\|_F^2 + \|V\|_F^2 \leq W_M^2 + V_M^2, \tag{A.18}$$

and hence, $\|Z\|_F \leq Z_M$, where $Z_M \triangleq \sqrt{W_M^2 + V_M^2}$. Furthermore, it follows from (A.18) that $\|W\|_F^2 \leq Z_M$ and $\|V\|_F^2 \leq Z_M$. Similarly, by defining

$$\hat{Z}(t) \triangleq \begin{bmatrix} \hat{W}(t) & 0_{n_h \times n_h} \\ 0_{n_p \times m} & \hat{V}(t) \end{bmatrix} \in \mathbb{R}^{(n_h+n_p)\times(m+n_h)}, \tag{A.19}$$

one can show that $\|\tilde{W}(t)\|_F \leq \|\tilde{Z}(t)\|_F$ and $\|\tilde{V}(t)\|_F \leq \|\tilde{Z}(t)\|_F$, where $\tilde{Z}(t) \triangleq Z - \hat{Z}(t)$.

Remark 7. Using (3.5), (A.13) and Remark 6, one can write

$$\|w\| \leq \|\tilde{Z}\|_F C_{\sigma'} Z_M (C_m + \|e\|) + Z_M (2C_\sigma + C_{\sigma'} \|\tilde{Z}\|_F C_m + C_{\sigma'} \|\tilde{Z}\|_F \|e\|) + \varepsilon_N, \tag{A.20}$$

or

$$\|w\| \leq C_0 + C_1 \|\tilde{Z}\|_F + C_2 \|\tilde{Z}\|_F \|e\|, \tag{A.21}$$

where

$$C_0 \triangleq 2C_\sigma Z_M + \varepsilon_N, \tag{A.22a}$$

$$C_1 \triangleq 2C_{\sigma'} C_m Z_M, \tag{A.22b}$$

$$C_2 \triangleq 2C_{\sigma'} Z_M. \tag{A.22c}$$

Remark 8. Sigmoid and tanh activation functions help define a bound on the LSTM output: We can write (2.21) as

$$h^n = \begin{pmatrix} \sigma_{g_1}(\cdots)\sigma_{c1}(\cdots) \\ \vdots \\ \sigma_{g_{N_h}}(\cdots)\sigma_{cN_h}(\cdots) \end{pmatrix} \in \mathbb{R}^{N_h}, \tag{A.23}$$

where N_h is the number of last hidden layer neurons. Then,

$$\|h^n\|^2 = \sum_{i=1}^{N_h} \sigma_{g_i}^2(\cdots)\sigma_{c\,i}^2(\cdots) \leq N_h. \tag{A.24}$$

Hence, $\|h^n\| \leq \sqrt{N_h}$ and the bound on the LSTM output can then be defined as

$$\|u^n_{lstm}\| \leq \|W_{fc}\|_F \sqrt{N_h} + \|b_{fc}\| \triangleq \bar{u}_{lstm}, \tag{A.25}$$

which by using (2.27) implies that $\|u_{lstm}(t)\| \leq \bar{u}_{lstm}$.

A.2 Proof of Theorem 1:

Let the approximation property (2.12) hold on a known compact set $S_p \triangleq \{x_p : \|x_p\| \leq b_x\}$ as stated in Assumption 1, for some $b_x > C_m$. Defining $S \triangleq \{x : \|x\| \leq b_x\}$, and using the fact that $\|x_p\| \leq \|x\|$, $x \in S$ implies that $x_p \in S_p$. Consider the compact set $S_e \triangleq \{e : \|e\| \leq b_x - C_m\}$, which imply that once $e \in S_e$, then $x_p \in S_p$. Let $e(0) \in S_e$. Then, $x_p(0) \in S_p$ and (2.12) holds. The proof proceeds by showing that $e(t) \in S_e$, $\forall t \geq 0$.

Consider the Lyapunov function

$$V = e^T P e + \text{tr}\{\tilde{W}^T F^{-1} \tilde{W}\} + \text{tr}\{\tilde{V}^T G^{-1} \tilde{V}\}. \tag{A.26}$$

Differentiating (B.4) along the trajectories (2.15) and (A.15), and by using (2.17), $\text{tr}\{A_1 A_2\} = \text{tr}\{A_2 A_1\}$, and $\dot{\tilde{W}} = -\dot{\hat{W}}$, it can be shown that

$$\dot{V} = -e^T Q e + 2e^T PB(w + u_{lstm} + v) + 2\text{tr}\{\tilde{W}^T \kappa \|e\| \hat{W}\} + 2\text{tr}\{\tilde{V}^T \kappa \|e\| \hat{V}\}. \tag{A.27}$$

Using (A.5), (A.17), and (A.19), one can write (A.27) as

$$\begin{aligned} \dot{V} =& -e^T Q e + 2e^T PB(w + u_{lstm} + v) + 2\kappa \|e\| \text{tr}\{\tilde{W}^T (W - \tilde{W})\} \\ &+ 2\kappa \|e\| \text{tr}\{\tilde{V}^T (V - \tilde{V})\} \\ =& -e^T Q e + 2e^T PB(w + u_{lstm} + v) + 2\kappa \|e\| \text{tr}\{\tilde{Z}^T (Z - \tilde{Z})\}. \end{aligned} \tag{A.28}$$

Since $\text{tr}\{\tilde{Z}^T (Z - \tilde{Z})\} = \text{tr}\{\tilde{Z}^T Z\} - \|\tilde{Z}\|_F^2 \leq \|\tilde{Z}\|_F (Z_M - \|\tilde{Z}\|_F)$, it follows from (A.28) that

$$\begin{aligned} \dot{V} \leq & -\lambda_{min}(Q) \|e\|^2 + 2\|B^T P e\| (\|w\| + \|u_{lstm}\|) \\ &+ 2e^T PBv + 2\kappa \|e\| \|\tilde{Z}\|_F (Z_M - \|\tilde{Z}\|_F), \end{aligned} \tag{A.29}$$

which, by using (A.21) and Remark 8, can be written as

$$\begin{aligned}\dot{V} \leq &-\lambda_{min}(Q)\|e\|^2 + 2\|B^T Pe\|(C_0 + \bar{u}_{lstm}) + 2C_1\|B^T Pe\|\|\tilde{Z}\|_F \\ &+ 2C_2\|B^T Pe\|\|e\|\|\tilde{Z}\|_F + 2e^T PBv + 2\kappa\|e\|\|\tilde{Z}\|_F(Z_M - \|\tilde{Z}\|_F).\end{aligned} \tag{A.30}$$

The proposed robustifying term (2.18) is designed to cancel the 4th term of (A.30). Two cases follow from using (2.18) in (A.30): *a)* $\|B^T Pe\| = 0$ and *b)* $\|B^T Pe\| \neq 0$.

Case a) $\|B^T Pe\| = 0$: Since $v = 0$, (A.30) reduces to

$$\begin{aligned}\dot{V} \leq &-\lambda_{min}(Q)\|e\|^2 + 2\kappa\|e\|\|\tilde{Z}\|_F(Z_M - \|\tilde{Z}\|_F) \\ = &-\|e\|(\lambda_{min}(Q)\|e\| + 2\kappa\|\tilde{Z}\|_F(\|\tilde{Z}\|_F - Z_M)) \\ = &-\|e\|\Big(\lambda_{min}(Q)\|e\| + 2\kappa(\|\tilde{Z}\|_F - \frac{Z_M}{2})^2 - \kappa\frac{Z_M^2}{2}\Big),\end{aligned} \tag{A.31}$$

which implies that $\dot{V} < 0$ if either

$$\|e\| > \frac{\kappa Z_M^2}{2\lambda_{min}(Q)}, \tag{A.32}$$

or

$$\|\tilde{Z}\|_F > Z_M. \tag{A.33}$$

Case b) $\|B^T Pe\| \neq 0$: Then, by using $\|\tilde{Z}\|_F \leq \|\hat{Z}\|_F + Z_M$ in the 4th term of (A.30), and substituting (2.18), it follows that

$$\begin{aligned}\dot{V} \leq &-\lambda_{min}(Q)\|e\|^2 + 2\|B^T Pe\|(C_0 + \bar{u}_{lstm}) + 2C_1\|B^T Pe\|\|\tilde{Z}\|_F \\ &+ 2C_2\|B^T Pe\|\|e\|(\|\hat{Z}\|_F + Z_M) - 2k_z\|B^T Pe\|\|e\|(\|\hat{Z}\|_F + Z_M) \\ &+ 2\kappa\|e\|\|\tilde{Z}\|_F(Z_M - \|\tilde{Z}\|_F).\end{aligned} \tag{A.34}$$

Since $k_z > C_2$, (A.34) reduces to

$$\begin{aligned}\dot{V} \leq &-\lambda_{min}(Q)\|e\|^2 + 2\|B^T Pe\|(C_0 + \bar{u}_{lstm}) + 2C_1\|B^T Pe\|\|\tilde{Z}\|_F + 2\kappa\|e\|\|\tilde{Z}\|_F(Z_M - \|\tilde{Z}\|_F) \\ \leq &-\lambda_{min}(Q)\|e\|^2 + 2\|B^T P\|_F\|e\|(C_0 + \bar{u}_{lstm}) + 2C_1\|B^T P\|_F\|e\|\|\tilde{Z}\|_F \\ &+ 2\kappa\|e\|\|\tilde{Z}\|_F(Z_M - \|\tilde{Z}\|_F) \\ = &-\|e\|\Big(\lambda_{min}(Q)\|e\| - 2\|B^T P\|_F(C_0 + \bar{u}_{lstm}) - 2C_1\|B^T P\|_F\|\tilde{Z}\|_F - 2\kappa\|\tilde{Z}\|_F(Z_M - \|\tilde{Z}\|_F)\Big) \\ = &-\|e\|\Big(\lambda_{min}(Q)\|e\| + 2\kappa(\|\tilde{Z}\|_F - \frac{q_1}{2})^2 - \kappa\frac{q_1^2}{2} - 2q_0\Big),\end{aligned} \tag{A.35}$$

where $q_0 \triangleq \|B^T P\|_F(C_0 + \bar{u}_{lstm})$ and $q_1 \triangleq C_1\|B^T P\|_F/\kappa + Z_M$. Therefore, it follows that $\dot{V} < 0$ if either

$$\|e\| > \frac{\kappa q_1^2 + 4q_0}{2\lambda_{min}(Q)}, \tag{A.36}$$

or

$$\|\tilde{Z}\|_F > \frac{q_1}{2} + \sqrt{\frac{q_1^2}{4} + \frac{q_0}{\kappa}}. \tag{A.37}$$

Hence, using the fact that the bounds given in (A.36) and (A.37) are bigger than the bounds in (A.32) and (A.33), and by selecting the Lyapunov matrix Q such that

$$\lambda_{min}(Q) > \frac{\kappa q_1^2 + 4q_0}{2(b_x - C_m)}, \tag{A.38}$$

it can be shown that the solution $(e(t), \tilde{Z}(t))$ is *uniformly ultimately bounded*,[48] and converges to the compact set

$$E \triangleq \left\{(e, \tilde{Z}) : \|e\| \leq \frac{\kappa q_1^2 + 4q_0}{2\lambda_{min}(Q)} < b_x - C_m \quad \text{and} \quad \|\tilde{Z}\|_F \leq \frac{q_1}{2} + \sqrt{\frac{q_1^2}{4} + \frac{q_0}{\kappa}}\right\}. \tag{A.39}$$

This implies that, since $e(0) \in S_e$ and $\dot{V} < 0$ on the boundary of S_e, $e(t)$ stays within S_e for all $t \geq 0$. Therefore $x_p(t) \in S_p$ for all $t \geq 0$.

Since the reference input $r(t)$ is bounded, $x_m(t)$ is bounded. Therefore, the boundedness of $e(t) = x(t) - x_m(t)$ implies that $x(t)$, and therefore all system signals, are bounded. ■

Appendix B

Proofs of Chapter 3

B.1 Proof of Theorem 1

Proof. Combining (3.2) with (3.1) gives

$$\dot{\xi}(t) = A_\mathrm{h}\xi(t) + B_\mathrm{h}[r(t-\tau) - E_h x(t-\tau)], \tag{B.1a}$$

$$y_h(t) = C_\mathrm{h}\xi(t) + D_\mathrm{h}[r(t-\tau) - E_h x(t-\tau)]. \tag{B.1b}$$

Given that the pilot output y_h is passed to the inner loop through an interface with magnitude saturation, y_h remains bounded. Consequently, according to Theorem 1, $x(t)$ is also bounded. Assuming that $r(t)$ is bounded, (B.1a) implies that for a stable A_h, ξ, and therefore y_h, are also bounded, thus completing the proof. □

B.2 Proof of Theorem 2

Proof. Consider the dynamics obtained by substituting the control input (3.13) and the baseline controller (3.8) into (3.3a), which is given as

$$\begin{aligned} \dot{x}(t) = &A_m x(t) + B_m y_h(t) \\ &+ B(u_{ad}(t) + u_{lstm}(t) + v(t) + f(x(t))). \end{aligned} \tag{B.2}$$

Substituting (B.2) and (3.9) into (3.12), the tracking error dynamics can be written as

$$\dot{e}(t) = A_m e(t) + B(u_{ad}(t) + u_{lstm}(t) + v(t) + f(x(t))). \tag{B.3}$$

Note that, the LSTM network is composed of layers with activation functions of tanh and sigmoid. Since these functions are bounded, the output of the LSTM network is bounded and this bound can be computed. Consider Lyapunov function,

$$L = e^T P e + \mathrm{tr}\{\tilde{W}^T F^{-1} \tilde{W}\} + \mathrm{tr}\{\tilde{V}^T G^{-1} \tilde{V}\}, \tag{B.4}$$

where $\tilde{V}(t) \in \mathbb{R}^{n_{h_h} \times n}$ and $\tilde{W}(t) \in \mathbb{R}^{m \times n_k}$ are the estimation errors of adaptive NN weights V and W. F and G are learning rate matrices in (2.15), and P is the solution of the Lyapunov equation (2.17). By differentiating (B.4) along the trajectory (B.3), and with a large enough selection of $\lambda_{min}(Q)$, where λ_{min} refers to the minimum eigenvalue, it can be shown that $\dot{L} < 0$ outside a compact set E, and therefore, the solution (e, V, W) is UUB [56, Theorem 2.24], and converges to E, where $(e, V, W) \in E$ implies that $x \in S$. This implies that, since $x(0) \in S$ and $\dot{L} < 0$ on the boundary of $S, x(t) \in S, \forall t \geq t(0)$ and hence the approximation property (3.4) always holds. □