Thesis for Master's Degree

Integral Error-Based Adaptive Neural Identifier for a Class of Uncertain Nonlinear Systems

Donghwa Hong

College of Engineering
Department of Mechanical and Robotics Engineering

Gwangju Institute of Science and Technology

2026

Integral Error-Based Adaptive Neural Identifier for a Class of Uncertain Nonlinear Systems

불확실한 비선형 시스템을 위한 적분 오차 기반 적응 신경망 식별기

Integral Error-Based Adaptive Neural Identifier for a Class of Uncertain Nonlinear Systems

Advisor: Pilwon Hur

by

Donghwa Hong

College of Engineering

Department of Mechanical and Robotics Engineering

Gwangju Institute of Science and Technology

A thesis submitted to the faculty of the Gwangju Institute of Science and Technology in partial fulfillment of the requirements for the degree of Master of Science in the Department of Mechanical and Robotics Engineering

Gwangju, Republic of Korea

November 29, 2025

Approved by

Professor Pilwon Hur

Committee Chair

Integral Error-Based Adaptive Neural Identifier for a Class of Uncertain Nonlinear Systems

Donghwa Hong

Accepted in partial fulfillment of the requirements for

the degree of Master of Science

November 29, 2025

Committee Chair _____

Prof. Pilwon Hur.

Committee Member _____

Prof. Pyojin Kim.

Committee Member _____

Prof. Kyunghwan Choi.

MS/ME 20244042 Donghwa Hong(홍동화). Integral Error-Based Adaptive Neural Identifier for a Class of Uncertain Nonlinear Systems (불확실한 비선형 시스템을 위한 적분 오차 기반 적응 신경망 식별기). Department of Mechanical and Robotics Engineering. 2026. 54p. Advisor: Prof. Pilwon Hur.

Abstract

This thesis studies online identification of unknown nonlinear dynamics in estimation problems, focusing on how past information is reflected in parameter updates. Conventional adaptive neural identifiers rely on instantaneous estimation errors, which are effective for real-time compensation but often show limited consistency when the adapted parameters are reused. In contrast, buffer-based online learning approaches improve consistency by explicitly storing past data, at the cost of increased computational and memory complexity. Motivated by this trade-off, an integral error–based adaptive neural identifier is proposed, in which past estimation errors are incorporated through a dynamic integral state rather than an explicit buffer. The proposed method preserves the continuous-time adaptive structure of conventional identifiers while enabling accumulated error information to influence parameter updates in a structured manner. A Lyapunov-based analysis establishes uniform ultimate boundedness of the estimation error, and simulations on a robot manipulator with static friction and a vehicle lateral dynamics model demonstrate improved consistency with comparable instantaneous estimation accuracy.

Contents

List of Figures

Chapter 1

Introduction

1.1 Motivation

Practical control and estimation problems are inevitably confronted with uncertainty. In engineered dynamical systems—including robotic manipulators, mobile robots, and ground vehicles—a nominal model is typically derived from first-principles analysis combined with simplifying assumptions [1–3]. However, real systems exhibit additional nonlinear effects that are difficult to model accurately, including friction, unmodeled couplings, load variations, and tire–road interactions. As a result, discrepancies between nominal models and actual plant behavior persist during operation and often dominate the achievable performance.

A common way to mitigate such discrepancies is to treat them as lumped disturbances and reject them through observer/filter-based compensation [4, 5]. This viewpoint has been widely adopted in control and estimation practice and is effective at improving instantaneous tracking and robustness when a reasonable nominal model is available.

Nevertheless, reducing tracking errors or estimation residuals does not necessarily imply that a *reusable model* of the uncertainty has been acquired. In many applications, a learned model is expected to support prediction and reuse—for example, in model predictive control, fault diagnosis, and digital-twin-oriented workflows—where model validity and consistency become as important as instantaneous error reduction [6–8].

This observation motivates an identification viewpoint: rather than treating the uncertainty merely as a disturbance to be attenuated, it can be regarded as an unknown component of the system dynamics that should be identified. Accordingly, many prac-

tical systems can be interpreted as admitting a decomposition

$$\dot{\mathbf{x}}(t) = \boldsymbol{f}(\mathbf{x}, \mathbf{u}) + \boldsymbol{g}(\mathbf{x}, \mathbf{u}), \tag{1.1}$$

where $\boldsymbol{f}(\mathbf{x}, \mathbf{u})$ denotes the nominal dynamics and $\boldsymbol{g}(\mathbf{x}, \mathbf{u})$ denotes an unknown nonlinear residual mapping [9–11].

In classical system identification, such a model is most commonly obtained offline from sufficiently rich input–output datasets, enabling systematic validation under controlled conditions [12]. However, offline identification is often misaligned with practical constraints: large datasets may be expensive or unsafe to collect, and the residual dynamics may change with operating conditions (e.g., temperature, wear, payload, road friction), so a model identified under one condition may not remain valid under another. Moreover, models learned for one system instance may not transfer reliably to another instance even within the same nominal class [13].

These limitations motivate online identification, in which the uncertainty model is updated during operation using streaming data. Online identification can be approached in several ways, ranging from recursive estimation schemes to incremental optimization based on streaming data, as well as adaptive formulations that embed parameter updates into the model structure [11, 12, 14–18].

In control and state-estimation deployments, such online identification schemes are typically integrated into estimator or controller pipelines, where the learned model is expected to remain meaningful when reused across time and operating conditions. In such settings, it is advantageous to employ update structures that remain interpretable from a control perspective and admit bounded-signal analysis [14, 19–26].

Neural-network-based adaptive identifiers or observers provide a representative control-oriented framework, in which parameter updates are formulated as a continuous-time adaptive law and are commonly accompanied by Lyapunov-based boundedness arguments [20, 21].

However, conventional adaptive neural identifiers are typically driven by instan-

taneous errors, and the resulting parameters may exhibit drift or local fitting, which limits the consistency and reusability of the learned model when adaptation is paused and the model is reused [27, 28].

A practical direction to improve model consistency is to incorporate past information more explicitly, for example by retaining recent samples and performing periodic optimization (e.g., mini-batch updates) or by optimizing a finite-horizon cost [25, 29–31]. These approaches can mitigate purely local fitting by revisiting past data, but they also introduce memory requirements and non-uniform computation, which can cause computational bursts and undermine deterministic timing in real-time control and estimation loops [18].

This thesis is motivated by the need to improve model consistency while preserving predictable real-time computation. The central idea is to incorporate past estimation errors into the adaptive law *without* relying on an explicit data buffer or batch optimization. To this end, an integral error–based adaptive neural identifier is developed, in which accumulated error information is embedded into a dynamic integral state that influences parameter updates continuously in time. The proposed formulation preserves the control-oriented adaptive structure of conventional adaptive neural identifiers while enabling principled retention of past information, aiming to improve model consistency and reusability with predictable computational timing. The resulting research objectives and contributions derived from this motivation are summarized in Section 1.2.

1.2 Research Objectives and Contributions

The discussion in Section 1.1 reveals a gap between real-time adaptive performance and consistent model reconstruction in online identification. Although adaptive neural schemes can effectively reduce instantaneous estimation errors, their reliance on local error information leaves unresolved the question of how accumulated past information should influence parameter updates.

The primary objective of this thesis is to clarify the distinction between identifica-

tion, learning, and adaptation in continuous-time online settings. In this work, identification is treated as the problem of reconstructing an unknown nonlinear mapping that represents system dynamics. Learning concerns how this mapping is refined over time using data in a manner that supports consistency and reuse. Adaptation, in contrast, is viewed as a mechanism for maintaining real-time performance without necessarily producing a reusable model.

A second objective is to investigate online parameter update structures that allow past estimation errors to influence the identification process implicitly, without explicit storage of historical data, while remaining compatible with continuous-time adaptive laws and real-time implementation constraints.

A third objective is to establish evaluation criteria for online identification that go beyond instantaneous error metrics and assess whether the identified parameters retain their functional meaning when adaptation is stopped.

The main contributions of this thesis are summarized as follows:

- An online neural identification framework in which past estimation errors influence parameter updates without explicit data storage.
- An evaluation perspective that assesses identified models based on their consistency when parameters are reused or frozen. To this end, a novel validation procedure termed the Reproducibility Test is introduced to quantitatively distinguish between transient overfitting and true structural learning.
- Demonstration of the proposed concepts through representative nonlinear systems in robotics and vehicle dynamics.

1.3 Outline of the Thesis

Chapter 2 introduces the class of nonlinear systems considered in this thesis and formulates the online identification problem from a model-level perspective. Representative examples from robot manipulator dynamics and vehicle lateral dynamics are

presented to illustrate common structural features.

Chapter 3 develops the online neural identification framework studied in this thesis and presents the associated parameter update laws together with a Lyapunov-based stability analysis.

Chapter 4 evaluates the proposed framework through simulation studies, describing the experimental setup, comparison methods, and validation procedures used to assess both instantaneous estimation performance and model consistency.

Chapter 5 concludes the thesis by summarizing the main findings and discussing their implications for online system identification.

Chapter 2

Preliminaries

This chapter establishes the mathematical framework and problem formulation adopted in this thesis. The primary objective is to define the class of uncertain nonlinear systems under consideration and to specify the structural assumptions required for valid online identification. The chapter is organized as follows:

- **Section 2.1** presents the general state-space formulation, decomposing the system into known nominal dynamics and an unknown nonlinear residual.
- **Section 2.2** illustrates this formulation through representative physical examples, specifically robot manipulator dynamics and vehicle lateral dynamics.
- **Section 2.3** outlines the scope and mathematical assumptions, such as boundedness and state availability, which serve as the foundation for the subsequent stability analysis.

2.1 Problem Formulation for Nonlinear Systems

Many practical control systems admit a decomposition into a nominal component derived from first-principles modeling and an additional nonlinear component that is not explicitly modeled. The nominal component captures the dominant dynamics based on physical insight and simplifying assumptions, while the remaining nonlinear effects appear as residual terms that depend on the system states and inputs.

Accordingly, consider a nonlinear system of the form

$$\dot{\mathbf{x}}(t) = \boldsymbol{f}(\mathbf{x}, \mathbf{u}) + \boldsymbol{g}(\mathbf{x}, \mathbf{u}), \tag{2.1}$$

where $\mathbf{x} \in \mathbb{R}^n$ denotes the system state vector and $\mathbf{u} \in \mathbb{R}^m$ denotes the control input. The function $\boldsymbol{f}(\mathbf{x}, \mathbf{u})$ represents the known nominal dynamics, while $\boldsymbol{g}(\mathbf{x}, \mathbf{u})$ represents an unknown nonlinear residual.

In this thesis, the residual term $\boldsymbol{g}(\mathbf{x}, \mathbf{u})$ is assumed to be bounded and time-invariant over a compact operating domain of interest. This assumption reflects the interpretation that $\boldsymbol{g}(\mathbf{x}, \mathbf{u})$ corresponds to a repeatable nonlinear mapping, rather than a transient or disturbance. Under this interpretation, $\boldsymbol{g}(\mathbf{x}, \mathbf{u})$ is treated as the object to be identified.

The problem of online identification considered in this work therefore concerns the estimation of $\boldsymbol{g}(\mathbf{x}, \mathbf{u})$ from available state and input measurements, while preserving the known structure of the nominal dynamics $\boldsymbol{f}(\mathbf{x}, \mathbf{u})$. This formulation provides a common framework for a variety of physical systems, as illustrated by the representative examples presented in the following subsections.

2.2 Representative Instances of the System Class

To illustrate the generality of the formulation in Section 2.1, two representative physical systems are briefly discussed. The purpose of these examples is not to provide detailed modeling or analysis, but to demonstrate how distinct systems can be cast into the common decomposition (2.1) at the model level.

2.2.1 Manipulator Dynamics with Joint Friction

Robot manipulator dynamics constitute a canonical example of the nonlinear system structure considered in this thesis. For an n-degree-of-freedom manipulator, the equations of motion derived from rigid-body dynamics can be expressed as

$$\boldsymbol{M}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + \boldsymbol{G}(\boldsymbol{q}) + \boldsymbol{\tau}_d(\boldsymbol{q}, \dot{\boldsymbol{q}}) = \mathbf{u}, \tag{2.2}$$

where $\boldsymbol{q} \in \mathbb{R}^n$ denotes the joint positions, $\boldsymbol{M}(\boldsymbol{q})$ is the inertia matrix, $\boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})$ represents Coriolis and centrifugal effects, and $\boldsymbol{G}(\boldsymbol{q})$ denotes the gravity vector.

The terms $\boldsymbol{M}(\boldsymbol{q})$, $\boldsymbol{C}(\boldsymbol{q},\dot{\boldsymbol{q}})$, and $\boldsymbol{G}(\boldsymbol{q})$ constitute the nominal dynamics derived from first-principles modeling. The remaining term $\boldsymbol{\tau}_d(\boldsymbol{q},\dot{\boldsymbol{q}})$ collects joint-level friction effects and unmodeled nonlinearities, and is commonly treated as an unknown but repeatable function over a given operating domain.

Define the state vector as

$$\mathbf{x} = \begin{bmatrix} \boldsymbol{q} \\ \boldsymbol{p} \end{bmatrix}, \qquad \boldsymbol{p} = \boldsymbol{M}(\boldsymbol{q})\dot{\boldsymbol{q}},$$

where $\boldsymbol{p}$ denotes the generalized momentum. Taking the time derivative yields

$$\dot{\mathbf{x}} = \underbrace{\begin{bmatrix} \boldsymbol{M}^{-1}(\boldsymbol{q})\boldsymbol{p} \\ \boldsymbol{C}^{\top}(\boldsymbol{q},\dot{\boldsymbol{q}})\boldsymbol{M}^{-1}(\boldsymbol{q})\boldsymbol{p} - \boldsymbol{G}(\boldsymbol{q}) + \mathbf{u} \end{bmatrix}}_{\boldsymbol{f}(\mathbf{x},\mathbf{u})} + \underbrace{\begin{bmatrix} \mathbf{0} \\ -\boldsymbol{\tau}_d(\boldsymbol{q},\dot{\boldsymbol{q}}) \end{bmatrix}}_{\boldsymbol{g}(\mathbf{x},\mathbf{u})}, \tag{2.3}$$

which admits the canonical decomposition of (2.1).

Equation (2.3) shows explicitly that joint friction enters the dynamics as a nonlinear residual term that depends on the system states and inputs, and can therefore be interpreted as a time-invariant state–input mapping suitable for online identification within the proposed framework.

2.2.2 Vehicle Lateral Dynamics with Nonlinear Tire Effects

Vehicle lateral dynamics provide a second representative instance of the system class considered in this thesis. As in the manipulator case, the formulation is first introduced at the force level before being recast into a state-space representation.

Consider the lateral and yaw dynamics of a ground vehicle described by the force balance equations

$$m(\dot{v}_y + v_x r) = F_{y\mathrm{f}} + F_{y\mathrm{r}}, \tag{2.4}$$

$$I_z \dot{r} = l_{\mathrm{f}} F_{y\mathrm{f}} - l_{\mathrm{r}} F_{y\mathrm{r}}, \tag{2.5}$$

where v_y denotes the lateral velocity, r is the yaw rate, v_x is the longitudinal velocity, and $F_{y\mathrm{f}}$ and $F_{y\mathrm{r}}$ denote the front and rear lateral tire forces, respectively.

Under nominal operating conditions, the lateral tire forces are commonly modeled using a linear approximation based on small slip-angle assumptions. Specifically, the tire forces are expressed as

$$F_{yi} = C_{\alpha i}\left(\delta_i - \frac{v_y \pm l_i r}{v_x}\right) + \Delta_i(v_x, v_y, r, \delta_i), \quad i \in \{\mathrm{f}, \mathrm{r}\}, \tag{2.6}$$

where $C_{\alpha i}$ denotes the cornering stiffness, δ_i is the steering input, and $\Delta_i(v_x, v_y, r, \delta_i)$ represents unknown nonlinear deviations from the linear tire model due to saturation effects, road conditions, and load transfer.

The first term in (2.6) corresponds to the nominal linear tire model derived under small slip-angle assumptions and bicycle model approximations [32], while the second term captures nonlinear tire effects that are not explicitly modeled.

Defining the state vector as

$$\mathbf{x} = \begin{bmatrix} v_y \\ r \end{bmatrix}, \qquad \mathbf{u} = \begin{bmatrix} \delta_{\mathrm{f}} \\ \delta_{\mathrm{r}} \end{bmatrix},$$

and substituting (2.6) into (2.4)–(2.5), the vehicle lateral dynamics can be written in the state-space form

$$\dot{\mathbf{x}} = \underbrace{\mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{u}}_{\boldsymbol{f}(\mathbf{x},\mathbf{u})} + \underbrace{\mathbf{K}\,\Delta(\mathbf{x}, \mathbf{u})}_{\boldsymbol{g}(\mathbf{x},\mathbf{u})}, \tag{2.7}$$

where $\Delta = [\Delta_{\mathrm{f}},\ \Delta_{\mathrm{r}}]^\top$ collects the nonlinear tire force components, and the matrices $\mathbf{A}$, $\mathbf{B}$, and $\mathbf{K}$ are determined by known vehicle parameters.

Equation (2.7) is a special case of the general nonlinear system formulation in (2.1), with the nominal component given by $\boldsymbol{f}(\mathbf{x}, \mathbf{u}) = \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{u}$ and the residual term representing repeatable nonlinear tire effects. Although the physical origin of these nonlinearities differs from joint friction in robotic manipulators, they play the same structural role as state–input mappings to be identified within the proposed framework.

Transition to assumptions The above examples also clarify the intended operating conditions under which the residual term $\boldsymbol{g}(\mathbf{x}, \mathbf{u})$ is to be identified online. These conditions are formalized in Assumptions 2.1–2.3, which define the system class considered throughout this thesis.

2.3 Scope and Assumptions for Online Identification

The problem formulation presented in this chapter relies on the following structural assumptions. These assumptions delimit the class of systems for which the proposed identification framework is valid.

Assumption 2.1 (Boundedness of Residual Dynamics)**.** The unknown nonlinear residual mapping $\boldsymbol{g}(\mathbf{x}, \mathbf{u})$ is continuous and bounded on a compact operating domain $\mathcal{D} \subset \mathbb{R}^n \times \mathbb{R}^m$. That is, there exists a constant $g_{max} > 0$ such that $\|\boldsymbol{g}(\mathbf{x}, \mathbf{u})\| \leq g_{max}$ for all $(\mathbf{x}, \mathbf{u}) \in \mathcal{D}$. Furthermore, the mapping is assumed to be quasi-static relative to the adaptation dynamics.

Assumption 2.2 (State Availability)**.** The full state vector $\mathbf{x}(t)$ is available for measurement. Alternatively, if a state observer is employed, the observation error is assumed to be bounded and converging, such that it does not destabilize the identification loop.

Assumption 2.3 (Nominal Model Validity)**.** The nominal dynamics $\boldsymbol{f}(\mathbf{x}, \mathbf{u})$ are known and Lipschitz continuous. The residual term $\boldsymbol{g}(\mathbf{x}, \mathbf{u})$ captures all structural discrepancies, including unmodeled dynamics and parametric uncertainties, such that the system trajectory remains unique and bounded for any bounded input $\mathbf{u}(t)$.

These assumptions explicitly define the class of nonlinear systems considered in this thesis.

Chapter 3

Proposed Adaptive Neural Identifier

This chapter develops the core contribution of this thesis: an integral error-based adaptive neural identifier. The focus is on deriving a stable adaptive law that incorporates past error information to improve model consistency. The development proceeds through the following steps:

- **Section 3.1** introduces the observer-like identifier structure designed to ensure stable state estimation.
- **Section 3.2** derives the backpropagation-based adaptive laws. It contrasts the conventional instantaneous error approach with the proposed integral error-based formulation and provides a rigorous Lyapunov-based stability analysis.
- **Section 3.3** offers practical guidelines for selecting key parameters, such as the learning rate and forgetting factor, to balance convergence speed and stability.

3.1 Adaptive Neural Identifier Structure

To guarantee stable online identification where the true values of the nonlinear dynamics are unmeasurable, a Luenberger observer-like framework is employed. The core strategy utilizes the state estimation error as the driving force for weight adaptation. By enforcing the error dynamics to be governed by a user-defined Hurwitz matrix $\mathbf{A}_d$, any persistent estimation error is driven solely by the inaccuracy of the neural network weights.

Based on the dynamics (2.1), the adaptive neural identifier structure is given by :

$$\dot{\hat{\mathbf{x}}} = \underbrace{\boldsymbol{f}(\mathbf{x}, \mathbf{u})}_{\text{known model}} + \underbrace{\hat{\boldsymbol{g}}(\hat{\mathbf{x}}, \mathbf{u})}_{\text{NN approximation}} + \underbrace{\mathbf{A}_d(\mathbf{x} - \hat{\mathbf{x}})}_{\text{stable error dynamics}} \tag{3.1}$$

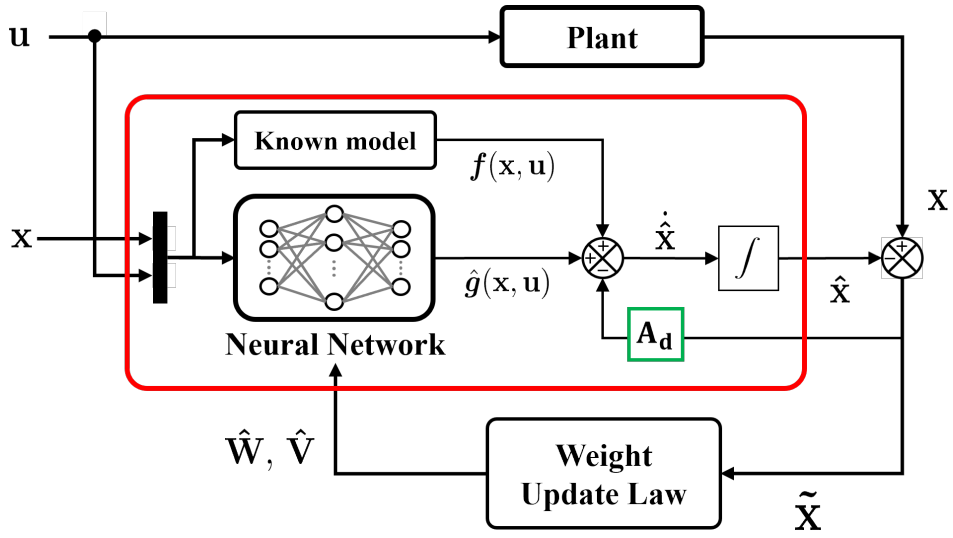


Figure 3.1. Schematic of the neural identifier Structure.

This structure consists of three key components:

Known Model : The bracketed term exploits the known nominal model evaluated using available state and input information, thereby separating the unknown residual nonlinearity to be approximated by the neural network.

NN Approximation : The term $\hat{\boldsymbol{g}}$ represents the neural network designed to approximate the residual nonlinearity. It is defined as $\hat{\boldsymbol{g}} = \hat{\mathbf{W}}\sigma(\hat{\mathbf{V}}\hat{\bar{\mathbf{x}}})$, where $\hat{\mathbf{W}}$ and $\hat{\mathbf{V}}$ are the estimated weight matrices, and $\sigma(\cdot)$ is the element-wise activation function, $\tanh(\cdot)$. The input vector $\hat{\bar{\mathbf{x}}} = [\hat{\mathbf{x}}^\top, \mathbf{u}^\top]^\top$ is constructed from the identifier states and control inputs.

Stable Error Dynamics : The term $\mathbf{A}_d(\mathbf{x} - \hat{\mathbf{x}})$ imposes strictly stable linear dynamics on the estimator. The Hurwitz matrix $\mathbf{A}_d$ is designed to govern the convergence rate of the estimation error, independent of the system's physical parameters. Defining the errors as $\tilde{\mathbf{x}} = \mathbf{x} - \hat{\mathbf{x}}$, $\tilde{\mathbf{W}} = \mathbf{W} - \hat{\mathbf{W}}$, and $\tilde{\mathbf{V}} = \mathbf{V} - \hat{\mathbf{V}}$, the error dynamics can be derived by subtracting the identifier dynamics (3.1) from the real system dynamics

(2.1):

$$\dot{\tilde{\mathbf{x}}} = \mathbf{A}_d\tilde{\mathbf{x}} + \tilde{\mathbf{W}}\sigma(\hat{\mathbf{V}}\hat{\tilde{\mathbf{x}}}) + w(t) \tag{3.2}$$

where $w(t) = \mathbf{W}\left(\sigma(\mathbf{V}\bar{\mathbf{x}}) - \sigma(\hat{\mathbf{V}}\hat{\tilde{\mathbf{x}}})\right) + \epsilon(\mathbf{x})$ represents the combined effect of the hidden layer weight error and the approximation error of the neural network. As shown in (3.2), $\mathbf{A}_d$ makes the estimation error $\tilde{\mathbf{x}}$ converge to a neighborhood around zero, with the size of this neighborhood determined by the neural network's approximation accuracy and the weight estimation errors.

3.2 Backpropagation-Based Adaptive Law with Static Approximation

To update the neural network weights, a backpropagation (BP) is one of the most popular algorithms []. However, the main drawback of the BP is the lack of a mathemical proof of stability in continuous-time systems. To address this issue, Abdollahi et al. proposed a modified BP algorithm [20], which enables tractable adaptive laws while preserving the continuous-time structure of the identifier.

While they defined the cost function based on the instantaneous error, this thesis presents a more generalized formulation and compares two types of error-based adaptive laws: the conventional instantaneous error-based approach and the proposed integral error-based approach.

3.2.1 Modified BP Algorithm for Neural Identifier

Consider the neural identifier dynamics given in (3.1). The weights of the neural network are updated using a gradient descent method based on a cost functional J, with leakage terms to prevent parameter drift:

$$\dot{\hat{\mathbf{W}}} = -\eta_1\frac{\partial J}{\partial \hat{\mathbf{W}}} - \rho_1\|\tilde{\mathbf{x}}\|\hat{\mathbf{W}}, \tag{3.3}$$

$$\dot{\hat{\mathbf{V}}} = -\eta_2\frac{\partial J}{\partial \hat{\mathbf{V}}} - \rho_2\|\tilde{\mathbf{x}}\|\hat{\mathbf{V}}, \tag{3.4}$$

where $\eta_1 > 0$ and $\eta_2 > 0$ are learning rates, and $\rho_1 > 0$ and $\rho_2 > 0$ are leakage gains.

Let

$$\text{net}_{\hat{\mathbf{V}}} = \hat{\mathbf{V}}\hat{\tilde{\mathbf{x}}}, \qquad \text{net}_{\hat{\mathbf{W}}} = \hat{\mathbf{W}}\sigma(\hat{\mathbf{V}}\hat{\tilde{\mathbf{x}}}),$$

where $\sigma(\cdot)$ denotes the element-wise activation function, $\tanh(\cdot)$. Using the chain rule, the gradients of the cost functional with respect to the weights can be expressed as

$$\begin{aligned}
\frac{\partial J}{\partial \hat{\mathbf{W}}} &= \frac{\partial J}{\partial \tilde{\mathbf{x}}} \frac{\partial \tilde{\mathbf{x}}}{\partial \hat{\mathbf{x}}} \frac{\partial \hat{\mathbf{x}}}{\partial \text{net}_{\hat{\mathbf{W}}}} \frac{\partial \text{net}_{\hat{\mathbf{W}}}}{\partial \hat{\mathbf{W}}} \\
&= \frac{\partial J}{\partial \tilde{\mathbf{x}}}(-I_{n\times n}) \frac{\partial \hat{\mathbf{x}}}{\partial \text{net}_{\hat{\mathbf{W}}}} \sigma(\hat{\mathbf{V}}\hat{\tilde{\mathbf{x}}}) \\
\frac{\partial J}{\partial \hat{\mathbf{V}}} &= \frac{\partial J}{\partial \tilde{\mathbf{x}}} \frac{\partial \tilde{\mathbf{x}}}{\partial \hat{\mathbf{x}}} \frac{\partial \hat{\mathbf{x}}}{\partial \text{net}_{\hat{\mathbf{V}}}} \frac{\partial \text{net}_{\hat{\mathbf{V}}}}{\partial \hat{\mathbf{V}}} \\
&= \frac{\partial J}{\partial \tilde{\mathbf{x}}}(-I_{n\times n}) \frac{\partial \hat{\mathbf{x}}}{\partial \text{net}_{\hat{\mathbf{V}}}} \hat{\tilde{\mathbf{x}}},
\end{aligned}$$

Since $\tilde{\mathbf{x}} = \mathbf{x} - \hat{\mathbf{x}}$, it follows that $\partial\tilde{\mathbf{x}}/\partial\hat{\mathbf{x}} = -I_{n\times n}$.

The chain-rule expressions involve the Jacobian terms $\partial\hat{\mathbf{x}}/\partial\text{net}_{\hat{\mathbf{W}}}$ and $\partial\hat{\mathbf{x}}/\partial\text{net}_{\hat{\mathbf{V}}}$, which describe how variations in the network outputs propagate to the identifier state. A direct evaluation of these terms requires differentiating the identifier dynamics and solving an additional set of auxiliary differential equations.

To justify a approximation, the principle of time-scale separation is invoked. The user-defined matrix $\mathbf{A}_d$ is designed to induce error dynamics that are significantly faster than the adaptation dynamics of the neural network weights. Under this assumption, the transient response of the identifier state $\hat{\mathbf{x}}$ to changes in the network output is considered instantaneous relative to the weight evolution. Consequently, the Jacobian terms can be approximated by their quasi-steady-state counterparts derived from the equilibrium condition of the error dynamics:

$$\frac{\partial \hat{\mathbf{x}}}{\partial \mathrm{net}_{\hat{\mathbf{W}}}} \approx -\mathbf{A}_d^{-1},$$
$$\frac{\partial \hat{\mathbf{x}}}{\partial \mathrm{net}_{\hat{\mathbf{V}}}} \approx -\mathbf{A}_d^{-1}\hat{\mathbf{W}}\left(\mathbf{I} - \mathbf{\Lambda}(\hat{\mathbf{V}}\hat{\bar{\mathbf{x}}})\right), \tag{3.5}$$

where $\mathbf{\Lambda}(\hat{\mathbf{V}}\hat{\bar{\mathbf{x}}}) = \mathrm{diag}\{\sigma_i^2(\hat{\mathbf{V}}_i\hat{\bar{\mathbf{x}}})\}$. This static approximation enables a closed-form expression of the adaptive laws while capturing the dominant direction of the gradient through the inverse of the error system matrix $\mathbf{A}_d$.

Substituting (3.5) into the chain-rule expressions yields the following unified update structure for the neural identifier:

$$\dot{\hat{\mathbf{W}}} = -\eta_1 \left(\frac{\partial J}{\partial \tilde{\mathbf{x}}}\mathbf{A}_d^{-1}\right)^\top \sigma(\hat{\mathbf{V}}\hat{\bar{\mathbf{x}}})^\top - \rho_1\|\tilde{\mathbf{x}}\|\hat{\mathbf{W}}, \tag{3.6}$$

$$\dot{\hat{\mathbf{V}}} = -\eta_2 \left(\frac{\partial J}{\partial \tilde{\mathbf{x}}}\mathbf{A}_d^{-1}\hat{\mathbf{W}}\left(\mathbf{I}_{n\times n} - \mathbf{\Lambda}(\hat{\mathbf{V}}\hat{\bar{\mathbf{x}}})\right)\right)^\top \hat{\bar{\mathbf{x}}}^\top - \rho_2\|\tilde{\mathbf{x}}\|\hat{\mathbf{V}}. \tag{3.7}$$

Within this framework, different adaptive laws are obtained by specifying the form of the error-related term $\partial J/\partial \tilde{\mathbf{x}}$.

3.2.2 Conventional Instantaneous Error-Based Adaptive Law

In the conventional approach [20], the cost functional is defined based on the instantaneous estimation error:

$$J_{\mathrm{inst}}(t) = \frac{1}{2}\,\tilde{\mathbf{x}}^\top(t)\,\tilde{\mathbf{x}}(t).$$

Accordingly, the gradient of the cost functional with respect to the estimation error is

$$\frac{\partial J_{\mathrm{inst}}}{\partial \tilde{\mathbf{x}}} = \tilde{\mathbf{x}}^\top. \tag{3.8}$$

Substituting this expression into the unified update structure (3.6)–(3.7) yields the conventional instantaneous error-based adaptive law. This approach updates the neural

network weights solely based on the current estimation error and serves as a baseline within the present framework Fig. 3.2.

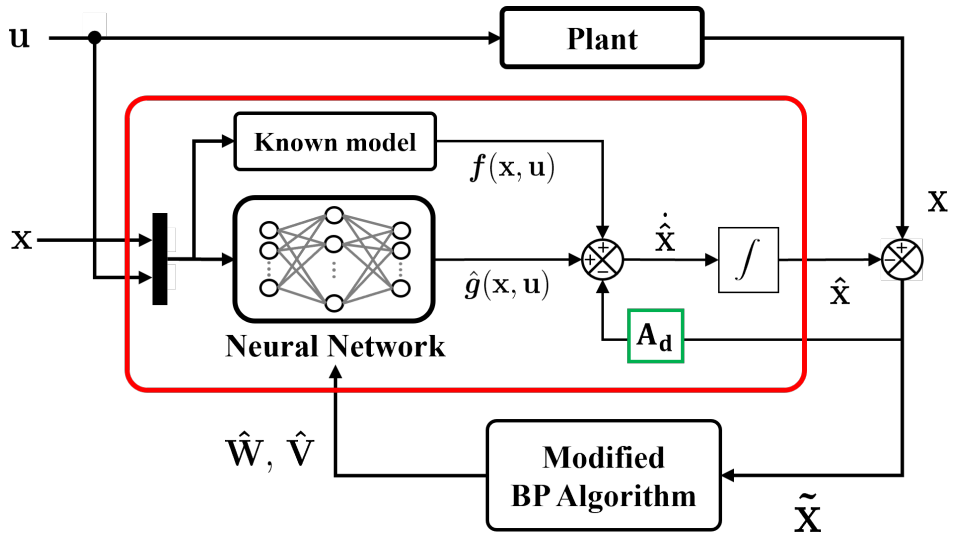


Figure 3.2. Schematic of the conventional neural identifier.

3.2.3 Proposed Integral Error-Based Adaptive Law

In contrast to the conventional approach, which minimizes the instantaneous error, the proposed method aims to minimize the accumulated estimation error over time. Accordingly, the cost functional is defined as the exponentially weighted integral of the squared estimation error:

$$J_{\text{int}}(t) = \frac{1}{2}\int_0^t e^{-\lambda(t-\tau)}\tilde{\mathbf{x}}^\top(\tau)\tilde{\mathbf{x}}(\tau)\,\mathrm{d}\tau,$$

where $\lambda > 0$ is the forgetting factor. The gradient of this cost functional with respect to the estimation error is given by:

$$\frac{\partial J_{\text{int}}}{\partial \tilde{\mathbf{x}}} = \mathbf{z}^\top, \tag{3.9}$$

where $\mathbf{z}(t)$ is the integral error state defined as:

$$\mathbf{z}(t) = \int_0^t e^{-\lambda(t-\tau)}\tilde{\mathbf{x}}(\tau)\,\mathrm{d}\tau.$$

By substituting the gradient expression (3.9) into the unified update structure (3.6)–(3.7), the proposed integral error-based adaptive law is obtained. Compared with the instantaneous approach, the proposed method allows past estimation errors to influence the parameter updates through the dynamic filter $\mathbf{z}(t)$, while maintaining the same adaptive structure and static approximation. It is worth noting that the integral state $\mathbf{z}(t)$ is implemented as a stable filter, ensuring boundedness without the risk of integral wind-up, as detailed in Remark 3.1. And, the overall structure of the proposed neural identifier is illustrated in Fig. 3.3.

Remark 3.1 (Boundedness of the Integral State)**.** The integral state $\mathbf{z}(t)$ remains strictly bounded despite the accumulation over $[0, t]$, owing to the exponential forgetting factor λ. The definition (3.2.3) is structurally equivalent to the stable linear filter dynamics:

$$\dot{\mathbf{z}}(t) = -\lambda\mathbf{z}(t) + \tilde{\mathbf{x}}(t), \quad \mathbf{z}(0) = \mathbf{0}.$$

Since $\lambda > 0$, this system is Bounded-Input Bounded-Output (BIBO) stable. Therefore, if the estimation error $\tilde{\mathbf{x}}(t)$ is bounded, the boundedness of $\mathbf{z}(t)$ is guaranteed. This required property ($\tilde{\mathbf{x}} \in \mathcal{L}_\infty$) is established in the subsequent stability analysis in Section 3.2.4, thereby ensuring that the adaptive law operates without the risk of integral wind-up.

3.2.4 Stability Analysis

To analyze the stability of the system described by (3.2) with the update laws (3.6)-(3.7), Lyapunov's direct method will be utilized. The goal is to demonstrate that the errors $\tilde{\mathbf{x}}$, $\tilde{\mathbf{W}}$, and $\tilde{\mathbf{V}}$ are uniformly ultimately bounded.

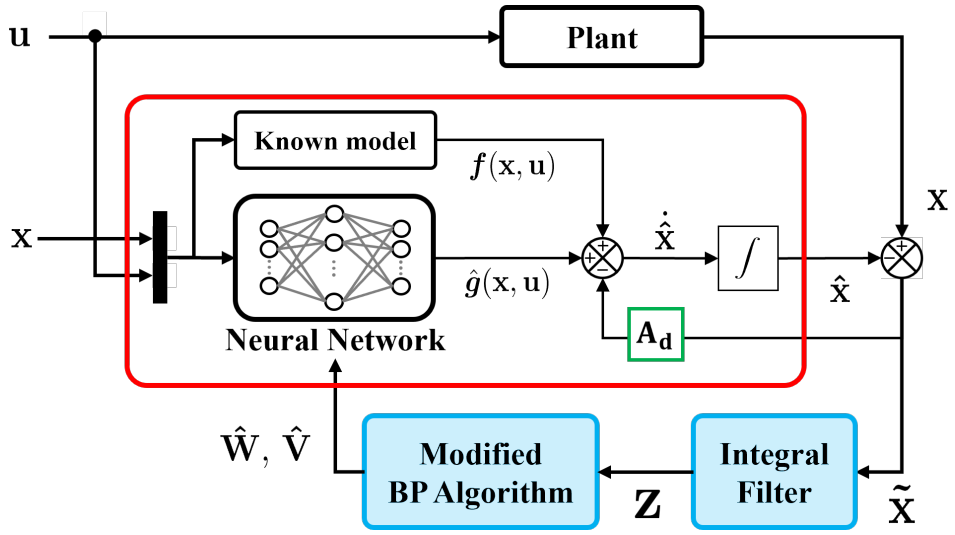


Figure 3.3. Schematic of the proposed neural identifier.

Theorem 3.1. For the system given by (3.2) with the update laws (3.6)-(3.7), all signals in the system $(\tilde{\mathbf{x}}, \tilde{\mathbf{W}}, \tilde{\mathbf{V}})$ are uniformly ultimately bounded.

Proof. To prove the theorem, the two subsystems are analyzed separately:

- Subsystem 1: the estimation error dynamics (3.2) and the output layer weight error dynamics.
- Subsystem 2: the hidden layer weight error dynamics.

The boundedness of $\tilde{\mathbf{V}}$ does not affect the stability of the first subsystem, as the activation function $\sigma(\hat{\mathbf{V}}\hat{\tilde{\mathbf{x}}})$ remains bounded irrespective of $\tilde{\mathbf{V}}$. However, the second subsystem is affected by the first subsystem. Thus, the stability of the first subsystem is established first, followed by the analysis of the second subsystem.

Consider the Lyapunov function candidate for the first subsystem:

$$L = \frac{1}{2}\tilde{\mathbf{x}}^\top \mathbf{P}_1 \tilde{\mathbf{x}} + \frac{1}{2}\operatorname{tr}(\tilde{\mathbf{W}}^\top \rho_1^{-1} \tilde{\mathbf{W}}) + \frac{1}{2}\int_0^t e^{-\lambda(t-\tau)} \tilde{\mathbf{x}}(\tau)^\top \mathbf{P}_2 \tilde{\mathbf{x}}(\tau) \mathrm{d}\tau$$

where $\mathbf{P}_1 > 0$ and $\mathbf{P}_2 > 0$ are positive definite matrices. The first two terms represent the energy of the error dynamics, while the last term represents the accumulated error

over time. After substituting the error dynamics, its time derivative is:

$$\dot{L} = \quad -\frac{1}{2}\tilde{\mathbf{x}}^\top(\mathbf{Q}_1 - \mathbf{P}_2)\tilde{\mathbf{x}} \quad + \tilde{\mathbf{x}}^\top\mathbf{P}_1(\tilde{\mathbf{W}}^\top\sigma_v + w) + \operatorname{tr}(\dot{\tilde{\mathbf{W}}}^\top\rho_1^{-1}\tilde{\mathbf{W}}) \quad - \lambda L_{\text{int}}$$

where $\sigma_v = \sigma(\hat{\mathbf{V}}^\top\hat{\tilde{\mathbf{x}}})$, $\mathbf{Q} = \mathbf{Q}_1 - \mathbf{P}_2 > 0$, and $\boldsymbol{L}_{\text{int}} = \frac{1}{2}\int_0^\top e^{-\lambda(t-\tau)}\tilde{\mathbf{x}}(\tau)^\top\mathbf{P}_2\tilde{\mathbf{x}}(\tau)\mathrm{d}\tau$. Substituting the update law (3.6) using $\dot{\tilde{\mathbf{W}}} = -\dot{\hat{\mathbf{W}}}$ yields

$$\operatorname{tr}(\dot{\tilde{\mathbf{W}}}^\top\rho_1^{-1}\tilde{\mathbf{W}}) = \eta_W \operatorname{tr}(\sigma_v\mathbf{z}^\top l_1\tilde{\mathbf{W}}) + \|\tilde{\mathbf{x}}\| \operatorname{tr}(\hat{\mathbf{W}}^\top\tilde{\mathbf{W}}),$$

where, for notational convenience, define $l_1 = \mathbf{A}^{-1}\rho_1^{-1}$, since $\mathbf{A}$ and ρ_1 are design parameters. The leakage term can be expanded by substituting $\hat{\mathbf{W}} = \mathbf{W} - \tilde{\mathbf{W}}$:

$$\|\tilde{\mathbf{x}}\| \operatorname{tr}(\hat{\mathbf{W}}^\top\tilde{\mathbf{W}}) = \|\tilde{\mathbf{x}}\| \operatorname{tr}((\mathbf{W} - \tilde{\mathbf{W}})^\top\tilde{\mathbf{W}}) = -\|\tilde{\mathbf{x}}\|\|\tilde{\mathbf{W}}\|^{\mathbf{2}} + \|\tilde{\mathbf{x}}\|\|\mathbf{W}\|\|\tilde{\mathbf{W}}\|$$

Substituting this into the $\dot{L}$ expression gives:

$$\begin{aligned}\dot{L} \leq & -\frac{1}{2}\tilde{\mathbf{x}}^\top\mathbf{Q}\tilde{\mathbf{x}} - \|\tilde{\mathbf{x}}\|\|\tilde{\mathbf{W}}\|^2 - \lambda\boldsymbol{L}_{\text{int}} + \tilde{\mathbf{x}}^\top\mathbf{P}_1(\tilde{\mathbf{W}}^\top\sigma_v + w) \\ & + \eta_W \operatorname{tr}(\sigma_v\mathbf{z}^\top l_1\tilde{\mathbf{W}}) + \|\tilde{\mathbf{x}}\|\|\mathbf{W}\|\|\tilde{\mathbf{W}}\|\end{aligned}$$

Additionally, the following inequalities hold:

$$\begin{aligned}|\tilde{\mathbf{x}}^\top\mathbf{P}_1\tilde{\mathbf{W}}^\top\sigma_v| &\leq \|\tilde{\mathbf{x}}\|\|\mathbf{P}_1\|(\|\tilde{\mathbf{W}}\|\sigma_M + \bar{w}), \\ \|\tilde{\mathbf{x}}\|\|\mathbf{W}\|\|\hat{\mathbf{W}}\| &\leq \|\tilde{\mathbf{x}}\|\|\tilde{\mathbf{W}}\|W_M, \\ |\eta_W \operatorname{tr}(\sigma_v\mathbf{z}^\top l_1\tilde{\mathbf{W}})| &\leq \eta_W\|\sigma_v\|\|\mathbf{z}\|\|l_1\|\|\tilde{\mathbf{W}}\| \leq \eta_W\sigma_M\frac{1}{\lambda}\|\tilde{\mathbf{x}}\|\|l_1\|\|\tilde{\mathbf{W}}\|.\end{aligned}$$

where $\|W\| \leq W_M$, $\|\sigma(\hat{\tilde{\mathbf{x}}})\| \leq \sigma_M$, and because $\mathbf{z}(t)$ is the state of the first-order filter (3.2.3) driven by $\tilde{\mathbf{x}}(t)$, its 2-norm satisfies

$$\|\mathbf{z}(t)\| \leq \int_0^t e^{-\lambda(t-\tau)}\|\tilde{\mathbf{x}}(\tau)\|\mathrm{d}\tau \leq \|\tilde{\mathbf{x}}\|_\infty\int_0^t e^{-\lambda(t-\tau)}\mathrm{d}\tau = \frac{1-e^{-\lambda t}}{\lambda}\|\tilde{\mathbf{x}}\|_\infty \leq \frac{1}{\lambda}\|\tilde{\mathbf{x}}\|_\infty$$

with n denoting the state dimension. Then, the inequality becomes:

$$\begin{aligned}\dot{L} \leq & -\frac{1}{2}\lambda_{\min}(\mathbf{Q})|\tilde{\mathbf{x}}\|^2 - \|\tilde{\mathbf{x}}\|\|\tilde{\mathbf{W}}\|^2 - \lambda \boldsymbol{L}_{\text{int}} + \|\tilde{\mathbf{x}}\|\|\mathbf{P}_1\|(\|\tilde{\mathbf{W}}\|\sigma_M + \bar{w}) \\ & + \|\tilde{\mathbf{x}}\|W_M\|\tilde{\mathbf{W}}\| + \frac{\eta_W \sigma_M}{\lambda}\|\tilde{\mathbf{x}}\|\|l_1\|\|\tilde{\mathbf{W}}\|\end{aligned}$$

By completing the squares for the terms involving $\|\hat{W}\|$, a sufficient condition on $\|x\|$ can be derived that is independent of the neural network weights error and ensures the time derivative of the Lyapunov candidate is negative.

$$\dot{L} \leq -\|\tilde{\mathbf{x}}\|\|\tilde{\mathbf{W}}\|^2 + k_b\|\tilde{\mathbf{x}}\|\|\tilde{\mathbf{W}}\| - \frac{1}{2}\lambda_{\min}(\mathbf{Q})\|\tilde{\mathbf{x}}\|^2 + \|\tilde{\mathbf{x}}\|\|\mathbf{P}_1\|w_M - \lambda \boldsymbol{L}_{\text{int}}$$

where $k_b = \|\mathbf{P}_1\|\sigma_M + W_M + \frac{\eta_W \sigma_M}{\lambda}\|l_1\|$. The terms involving $\tilde{\mathbf{W}}$ are of the form $-(\|\tilde{\mathbf{x}}\|)\|\tilde{\mathbf{W}}\|^2 + (k_b\|\tilde{\mathbf{x}}\|)\|\tilde{\mathbf{W}}\|$. By completing the square, this is bounded above by $\frac{(k_b\|\tilde{\mathbf{x}}\|)^2}{4|\tilde{\mathbf{x}}\|} = \frac{k_b^2}{4}\|\tilde{\mathbf{x}}\|$. The final inequality for $\dot{L}$ is:

$$\dot{L} \leq -\frac{1}{2}\lambda_{min}(\mathbf{Q})\|\tilde{\mathbf{x}}\|^2 - \lambda \boldsymbol{L}_{\text{int}} + \left(\|\mathbf{P}_1\|\bar{w} + \frac{k_b^2}{4}\right)\|\tilde{\mathbf{x}}\|$$

To find a sufficient condition that guarantees $\dot{L} \leq 0$ and subsequently derive the ultimate bound, a simpler upper bound can be analyzed. Since the term $-\lambda L_{int}$ is always non-positive, it can be omitted from the right-hand side while the inequality still holds. The analysis thus proceeds with the remaining terms:

$$\|\tilde{\mathbf{x}}\| \geq \frac{2\left(\|\mathbf{P}_1\|\bar{w} + k_b^2\right)}{\lambda_{min}(\mathbf{Q})} = b \tag{3.10}$$

Thus, the condition on $\|\tilde{\mathbf{x}}\|$ ensures the negative semi-definiteness of $\dot{L}$, leading to the ultimate boundedness of $\tilde{\mathbf{x}}$ and the output layer weight error $\tilde{\mathbf{W}}$. In fact, $\dot{L}$ is negative definite outside the ball with radius b.

To show the boundedness of the second subsystem, consider (3.7) which can be

rewritten as:

$$\begin{aligned}\dot{\tilde{\mathbf{V}}} &= \eta_2 \left(\mathbf{z}^\top \mathbf{A}_d^{-1} \hat{\mathbf{W}} (\mathbf{I} - \mathbf{\Lambda}(\hat{\mathbf{V}}\hat{\tilde{\mathbf{x}}}))\right)^\top \hat{\tilde{\mathbf{x}}}^\top + \rho_2 \|\tilde{\mathbf{x}}\| \hat{\mathbf{V}} \\ &= -\alpha \tilde{\mathbf{V}} + \alpha \mathbf{V} + f_2(\mathbf{z}, \hat{\mathbf{W}}, \hat{\mathbf{V}}, \hat{\tilde{\mathbf{x}}})\end{aligned}$$

where $f_2(\cdot)$ is a function of the system states and parameters, and $\alpha = \rho_2 \|\tilde{\mathbf{x}}\|$ is a positive constant. It follows that the term $f_2(\cdot)$ is bounded based on the following arguments: First, since the open-loop system is stable and the ideal wieghts are also constant, it follows that $\hat{\mathbf{W}} \in L_\infty$ and $\hat{\mathbf{x}} \in L_\infty$. Second, the function $\Lambda(\cdot)$ is bounded due to the boundedness of hyperbolic tangent function. Third, the filtered error $\mathbf{z}(t)$ is also bounded since it is the integral of the state error $\tilde{\mathbf{x}}(t)$, which is ultimately bounded by (3.10). Therefore, the boundedness of $\tilde{\mathbf{V}}$ is also ensured.

Thus, the second subsystem is uniformly ultimately bounded as well. The overall conclusion is that all signals in the system are uniformly ultimately bounded, which completes the proof.

□

3.3 Parameter Selection Guidelines

This section provides practical guidelines for selecting key parameters in the proposed integral error-based adaptive neural identifier. The focus is on three main parameters: the Hurwitz matrix $\mathbf{A}_d$, the learning rates η_1 and η_2, and the forgetting factor λ. Proper tuning of these parameters is crucial to balance convergence speed and stability.

Hurwitz Matrix ($\mathbf{A}_d$)

Placing the poles further in the left-half plane (i.e., choosing larger negative eigenvalues) increases the stability and convergence speed of the state estimator. However, since $\mathbf{A}_d^{-1}$ appears in the weight update law, a more stable $\mathbf{A}_d$ can slow down the convergence of the weights. Therefore, it is recommended to set $\mathbf{A}_d$ to be sufficiently stable and adjust

the learning rates accordingly.

Learning Rates (η_1, η_2)

For a single-layer NN, η_1 mainly affects the output layer weights, while η_2 affects the hidden layer weights. If the system to be identified is not highly nonlinear, it is advisable to fix η_1 at a small value and tune η_2 for faster adaptation of the hidden layer.

Forgetting Factor (λ)

This parameter determines how quickly past errors are forgotten. A smaller λ increases the influence of past errors, which is useful for slowly time-varying systems. Typically, λ is chosen between 1 and 100. If λ is too large, the effect of past errors is negligible, making the algorithm focus only on instantaneous errors and reducing overall approximation capability. Conversely, if λ is too small, the influence of past errors dominates, which can slow down the response to current errors and reduce the effectiveness of the Lyapunov function decrease.

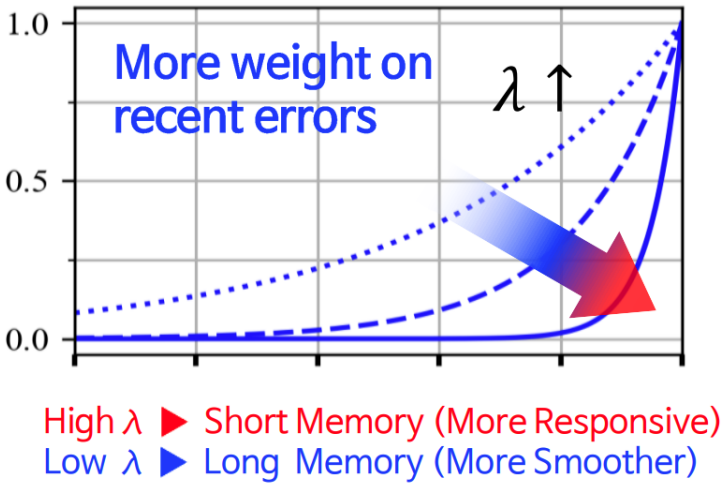


Figure 3.4. Conceptual illustration of the forgetting factor λ. The exponential term assigns higher weights to recent estimation errors, while the influence of past data is exponentially decayed over time.

Summary

Set $\mathbf{A}_d$ to be sufficiently stable to ensure stable state estimation, fix λ (e.g., to 10) to balance the influence of past errors, and then tune η_1 and η_2 to adjust the convergence speed of the weights.

Chapter 4

Validation in Simulations

This chapter presents a comprehensive validation of the proposed framework through numerical simulations. The primary objective is to demonstrate that incorporating past estimation errors via an integral state enhances the structural consistency of the identified model without compromising real-time adaptation performance. The evaluation is structured as follows:

- **Section 4.1** establishes the evaluation criteria, introducing the *Reproducibility Test* to quantitatively assess the consistency of the learned model.
- **Section 4.2** describes the comparison baselines, including the conventional instantaneous error-based method and a buffer-based online learning scheme.
- **Section 4.3** and **Section 4.4** report the simulation results on 2 cases : a 2-DOF robot manipulator and a vehicle lateral dynamics model, respectively. Each case includes detailed analysis in terms of *online estimation* and *model consistency*.
- **Section 4.5** summarizes the findings and discusses their implications for online identification.

4.1 Evaluation Methodology

To evaluate the performance of the proposed framework in identifying the uncertain models discussed in Chapter 2, this section introduces a novel validation procedure termed the *Reproducibility Test*. While conventional online identification schemes are often evaluated by their ability to minimize instantaneous estimation errors [], analyzing performance solely from a real-time perspective is insufficient for true model *identification*. From an identification standpoint, it is essential to verify whether the

identifier has captured the underlying structural characteristics of the system. By assessing the output consistency of the frozen model, this test determines if the learned mapping remains valid across the entire operating history rather than merely overfitting to transient residuals.

The procedure consists of three stages illustrated in Fig. 4.1:

1. **Online Estimation** : The identifier adapts the neural network online continuously up to a specific time instance T_{f}.

2. **Weight Freezing** : At $t = T_{\mathrm{f}}$, the online estimation is deactivated, and the weights are fixed at their instantaneous values $\hat{\mathbf{W}}_{t=T_{\mathrm{f}}}, \hat{\mathbf{V}}_{t=T_{\mathrm{f}}}$.

3. **Reproducibility Test** : The historical data is re-evaluated using these frozen weights. The network output $\hat{\mathbf{W}}_{t=T_{\mathrm{f}}}\boldsymbol{\sigma}(\hat{\mathbf{V}}_{t=T_{\mathrm{f}}}\bar{\mathbf{x}})$ is compared against the original outputs generated during the active adaptation phase.

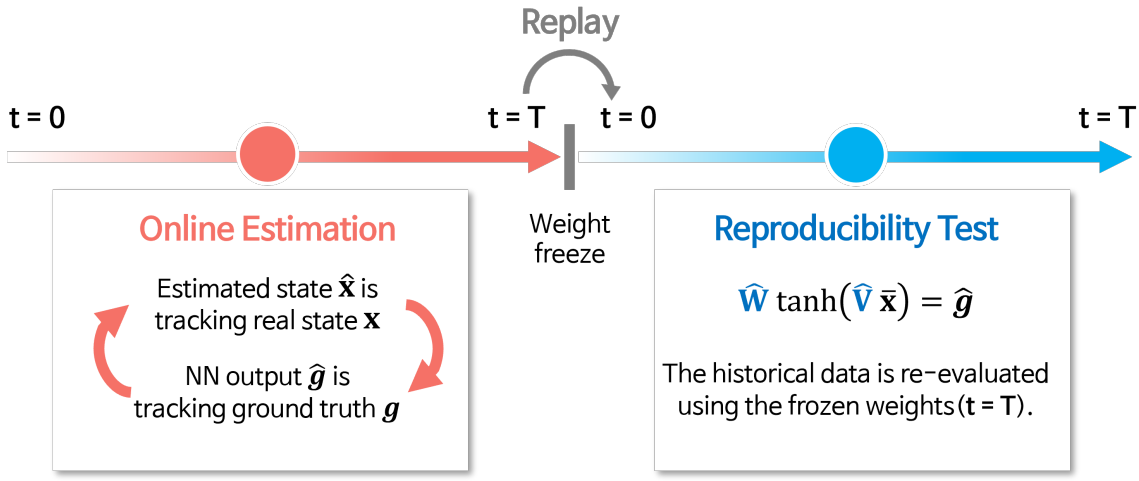


Figure 4.1. Schematic of the Reproducibility Test procedure for validating adapted neural network models.

The underlying intuition is that if the network has meaningfully adapted the dynamics within its operating region, the output generated by the frozen weights must identically reproduce the historical estimation results for $t < T_{\mathrm{f}}$. Any discrepancy implies that the weights were merely overfitting to instantaneous residuals rather than capturing the consistent physical structure.

4.2 Comparison Methods

This section defines the comparison methods used in the simulation validation and clarifies the rationale behind their selection. Rather than introducing unrelated algorithms, all methods considered here share a common neural identifier structure and differ only in *how past information is reflected in the parameter update law*. A summary is given in Table 4.1, which serves as the primary reference for the discussion below.

Update-Structure Viewpoint To avoid ambiguity regarding "how frequently" the model is updated, it is useful to distinguish between two update structures. In Baseline 1 and the proposed method, the neural weights are treated as *internal adaptive states* of the identifier and are updated according to continuous-time adaptive laws. In digital implementation, these laws appear as per-step updates because the state equations are numerically integrated on the sampling grid, i.e., $W_{k+1} = W_k + \Delta t\, \dot{W}(\cdot)$ and $V_{k+1} = V_k + \Delta t\, \dot{V}(\cdot)$. In contrast, Baseline 2 treats the weights as *parameters of an optimization routine* that is executed on a finite recent-memory dataset, resulting in a periodic (sample-and-hold) update structure. This distinction is important for interpreting both the algorithmic behavior and the computation-time characteristics reported in Section 4.5.2.

Baseline 1: Instantaneous Error-Based Adaptive Neural Identifier

This method updates the neural network parameters using only the instantaneous state estimation error, with the pointwise quadratic objective

$$J(t) = \tfrac{1}{2}\|\tilde{\mathbf{x}}(t)\|^2.$$

The weight update laws are derived using the modified backpropagation adaptive laws (3.6) and (3.7), and are executed in continuous time as $\dot{\hat{\mathbf{W}}}$ and $\dot{\hat{\mathbf{V}}}$.

In digital implementation, the continuous-time adaptation is applied on the sam-

pling grid via numerical integration, yielding the per-step recursion

$$\hat{\mathbf{W}}_{k+1} = \hat{\mathbf{W}}_k + \Delta t\, \dot{\hat{\mathbf{W}}}_k, \qquad \hat{\mathbf{V}}_{k+1} = \hat{\mathbf{V}}_k + \Delta t\, \dot{\hat{\mathbf{V}}}_k, \tag{4.1}$$

where Δt denotes the sampling period. As summarized in Table 4.1, no past information is retained or aggregated in the update signal. This method provides uniform per-step updates with minimal computational cost and is therefore widely used as a real-time adaptive baseline.

Baseline 2: Buffer-Based Mini-Batch Online Learning

As a representative data-driven online learning baseline, a buffer-based mini-batch online learning scheme with a sliding-window data buffer is adopted as Baseline 2. Although specific network architectures and optimization details vary across existing studies [31, 33–38], the practice of explicitly storing recent transitions and performing batch-wise parameter updates constitutes a widely adopted update structure in data-driven online learning approaches to system identification and dynamical model learning.

In this baseline, weights update is performed using stochastic gradient descent (SGD) based on prediction errors evaluated over randomly sampled mini-batches drawn from the sliding window. By decoupling the weight update from instantaneous measurements, this approach emphasizes data reuse and model-level consistency, at the cost of increased memory usage and computational overhead. Accordingly, Baseline 2 serves as a reference buffer-based method, enabling a direct comparison between explicit memory utilization and the proposed integral-error-based mechanism that incorporates past information without explicit data storage.

Given the discrete-time predictor formulation:

$$\hat{\mathbf{x}}_{k+1} = \mathbf{x}_k + \Delta t\big(f(\mathbf{x}_k, \mathbf{u}_k) + \hat{g}(\mathbf{x}_k, \mathbf{u}_k)\big), \tag{4.2}$$

the learning objective is defined as a mini-batch cost:

$$J_{\mathrm{mb}} = \frac{1}{2M} \sum_{k \in \mathcal{B}_m} \left\| \hat{\mathbf{x}}_{k+1} - \mathbf{x}_{k+1} \right\|^2, \tag{4.3}$$

where $\mathcal{B}_m$ denotes a randomly sampled mini-batch of size M drawn from a finite sliding-window buffer storing recent transition tuples $\{\bar{\mathbf{x}}_k, \mathbf{x}_k, \mathbf{x}_{k+1}, f_k\}$.

Weights update is obtained via SGD by minimizing the mini-batch prediction error cost J_{mb}:

$$\hat{\mathbf{W}} \leftarrow \hat{\mathbf{W}} - \eta_1 \nabla_{\hat{\mathbf{W}}} J_{\mathrm{mb}}, \qquad \hat{\mathbf{V}} \leftarrow \hat{\mathbf{V}} - \eta_2 \nabla_{\hat{\mathbf{V}}} J_{\mathrm{mb}}. \tag{4.4}$$

These updates are executed at discrete intervals (every P_{upd} steps) to accommodate the computational overhead associated with buffer management and gradient computation.

Proposed: Integral Error-Based Adaptive Neural Identifier

This method modifies only the error signal driving the adaptive law. Instead of the instantaneous error, an integral error state

$$\dot{\mathbf{z}} = -\lambda \mathbf{z} + \tilde{\mathbf{x}}$$

is introduced, corresponding to the exponentially weighted cost

$$J(t) = \tfrac{1}{2} \int_0^t e^{-\lambda(t-\tau)} \|\tilde{\mathbf{x}}(\tau)\|^2 \, d\tau.$$

Past information is incorporated implicitly through this dynamic state rather than through explicit data storage.

The weight update structure remains continuous-time and deterministic, and is otherwise identical to Baseline 1, except that the backpropagated error signal is replaced by $\mathbf{z}$. Accordingly, in digital implementation the adaptive laws are applied on

the sampling grid via numerical integration as

$$\hat{\mathbf{W}}_{k+1} = \hat{\mathbf{W}}_k + \Delta t\, \dot{\hat{\mathbf{W}}}_k, \qquad \hat{\mathbf{V}}_{k+1} = \hat{\mathbf{V}}_k + \Delta t\, \dot{\hat{\mathbf{V}}}_k, \tag{4.5}$$

while $\mathbf{z}$ is updated in the same manner:

$$\mathbf{z}_{k+1} = \mathbf{z}_k + \Delta t\big(-\lambda \mathbf{z}_k + \tilde{\mathbf{x}}_k \big). \tag{4.6}$$

Item	**Baseline 1**	**Baseline 2**	**Proposed**
Error signal	$\tilde{\mathbf{x}} = \mathbf{x} - \hat{\mathbf{x}}$	$\tilde{\mathbf{x}}^{\text{pred}}_{k+1} = \mathbf{x}_{k+1} - \hat{\mathbf{x}}^{\text{pred}}_{k+1}$	$\tilde{\mathbf{x}} = \mathbf{x} - \hat{\mathbf{x}}$
State update model	$\dot{\hat{\mathbf{x}}} = f + \hat{g} + A_d(\mathbf{x} - \hat{\mathbf{x}})$	$\hat{\mathbf{x}}^{\text{pred}}_{k+1} = \mathbf{x}_k + \Delta t\,(f_k + \hat{g})$	$\dot{\hat{\mathbf{x}}} = f + \hat{g} + A_d(\mathbf{x} - \hat{\mathbf{x}})$
Cost function	$J(t) = \frac{1}{2}\|\tilde{\mathbf{x}}(t)\|^2$	$J_{\text{mb}} = \frac{1}{2M}\sum_{k\in\mathcal{B}_m} \|\tilde{\mathbf{x}}^{\text{pred}}_{k+1}\|^2$	$J(t) = \frac{1}{2}\int_0^t e^{-\lambda(t-\tau)}\|\tilde{\mathbf{x}}(\tau)\|^2 d\tau$
Past information usage	None	Sliding-window buffer storing recent transition tuples $\{x_k, u_k, X_k, X_{k+1}, f_k\}$	$\dot{\mathbf{z}} = -\lambda\mathbf{z} + \tilde{\mathbf{x}}$ $\mathbf{z} \leftarrow \mathbf{z} + \dot{\mathbf{z}}\,\Delta t$
Weight update	Per-step (integrated)	Every P_{upd} steps	Per-step (integrated)
Update Mechanism	Instantaneous error-based Modified BP adaptive law (Chap. 3)	One-step prediction-error-based BP learning rule	Integral error-based Modified BP adaptive law (Chap. 3)

Table 4.1. Summary of comparison methods: objective formulation, information utilization, and learning structure.

4.3 Case 1 : 2-DOF Robot Manipulator

4.3.1 Setup

Environment The simulation is conducted on a 2 DOF robot manipulator, as described by Eq. (2.2), with physical parameters listed in Table 4.2. To clearly highlight the effectiveness of the proposed method, a simplified simulation environment is adopted. Specifically, the simulation assumes ideal conditions with no sensor noise, unmodeled dynamics, or external disturbances. This controlled setup allows for a focused

analysis of the algorithm's core performance. However, the proposed framework can be extended to higher-DOF manipulators, and more realistic scenarios.

Target Model The ground-truth friction dynamics consist of both viscous and Coulomb friction effects and are modeled as

$$\boldsymbol{\tau}_{d,i} = f_{c,i} \tanh(\frac{\dot{q}_i}{\sigma_i}) + b_i \dot{q}_i, \qquad i \in \{1, 2\}, \tag{4.7}$$

where $f_{c,i}$ and b_i denote the Coulomb and viscous friction coefficients, respectively, and σ_i is a smoothing constant. In the identification framework, this friction term is treated as an unknown nonlinear residual acting on top of the nominal rigid-body dynamics and serves as the target function to be approximated by the neural identifier.

Table 4.2. Physical parameters of the system model used in Case 1.

-	Description	Value	Unit
m	Link mass	2.465	kg
l	Link length	0.2	m
l_c	Link CoM position	0.13888	m
I	Link inertia	0.06911	kg·m^2
I_m	Motor inertia	0.008118	kg·m^2
b	Viscous friction	0.5	N·m·s
f_c	Coulomb friction	0.1	N·m
σ	Smoothing parameter	6.67×10^{-4}	N·m·s

Control and Reference Trajectory The reference trajectory for each joint is generated using a fifth-order polynomial (quintic) trajectory, which ensures smooth position, velocity, and acceleration profiles. The initial joint positions and velocities are set to zero. Specifically, the trajectory starts at $\boldsymbol{q}_d(0) = [\frac{\pi}{4},\ \frac{\pi}{2}]^\top$ and ends at $\boldsymbol{q}_d(T) = [\frac{3\pi}{4},\ -\frac{\pi}{2}]^\top$, where T is the duration of one cycle. After reaching the endpoint, the trajectory reverses and repeats between these two waypoints, resulting in a periodic motion.

A feedback linearization controller is used to track the reference trajectory:

$$\mathbf{u} = \boldsymbol{M}(\boldsymbol{q})\boldsymbol{v} + \boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + \boldsymbol{G}(\boldsymbol{q}) \tag{4.8}$$

$$\boldsymbol{v} = \ddot{\boldsymbol{q}}_d + K_d(\dot{\boldsymbol{q}}_d - \dot{\boldsymbol{q}}) + K_p(\boldsymbol{q}_d - \boldsymbol{q}) \tag{4.9}$$

where $\boldsymbol{M}, \boldsymbol{C}, \boldsymbol{G}$ are the inertia matrix, Coriolis/centrifugal matrix, and gravity vector respectively, $\boldsymbol{v}$ is the new control input designed for linearized error dynamics, $\boldsymbol{q}_d$, $\dot{\boldsymbol{q}}_d$, and $\ddot{\boldsymbol{q}}_d$ are the desired joint position, velocity, and acceleration, respectively , and K_p, K_d are positive definite gain matrices.

Configuration of Comparison Methods The robot manipulator exhibits relatively slow dynamics, and friction effects evolve over several seconds. Accordingly, the buffer size in Baseline 2 was set to 2000 samples, which corresponds to approximately 4 s of data (0.002×2000 s), allowing explicit utilization of recent motion history. The forgetting factor $\lambda = 2$ in the proposed method was chosen to induce a comparable effective memory length through exponential weighting. To limit computational load while preserving recent information, mini-batch updates in Baseline 2 were executed every 10 steps. In detail, the configuration parameters for all methods are summarized in Table 4.3.

Item	**Baseline 1**	**Baseline 2**	**Proposed**
Initialization	$(W_0, V_0) = (10^{-2}, 10^{-2})$ with random initialization		
Network architecture	$(n_i, n_h, n_o) = (6, 20, 4)$		
Observer gain A_d	$-40\, I_{2\times2}$	(None)	$-40\, I_{2\times2}$
Learning rates (η_1, η_2)	$(10^2,\ 8 \times 10^3)$	$(10^3,\ 5 \times 10^4)$	$(4 \times 10^1,\ 6 \times 10^3)$
Leakage rates (ρ_1, ρ_2)	$(10^{-2},\ 10^{-2})$	(None)	$(10^{-2},\ 10^{-2})$
Update period P_{upd}	every step	every 10 steps	every step
Buffer size $\mathcal{B}$	(None)	2000 samples	(None)
Mini-batch size M	(None)	64 samples	(None)
Forgetting factor λ	(None)	(None)	2

Table 4.3. Configuration summary of comparison methods.

4.3.2 Online Estimation Results

The online estimation performance is presented in Fig. 4.2. All three methods eventually reduce the estimation error, but their transient behaviors differ significantly, particularly at the zero-velocity crossings where static friction acts as a discontinuity.

Baseline 1 demonstrates the fastest adaptation, effectively suppressing the error spikes caused by the sudden onset of static friction. This is because its update law reacts instantaneously to the current error, allowing the weights to change rapidly to compensate for the discontinuity. In contrast, the Proposed Method exhibits larger error peaks (overshoots) at these friction reversals. This behavior is expected because the integral error state $\mathbf{z}(t)$ inherently smooths out the rapid changes in the error signal, resulting in a delayed response to instantaneous discontinuities. Baseline 2 shows a similar trend to the Proposed Method, as the mini-batch update over a sliding window also averages out instantaneous fluctuations.

4.3.3 Model Consistency Results

While Baseline 1 excels in instantaneous error suppression, Fig. 4.3 reveals the cost of such aggressive adaptation. The friction models reconstructed from weights frozen at different times ($t = 180\,\mathrm{s}$ and $t = 220\,\mathrm{s}$) show that Baseline 1 fails to maintain a consistent function shape. The weights fluctuate significantly to fit the local static friction spikes, leading to an inconsistent representation of the global friction dynamics.

Conversely, Baseline 2 and the Proposed Method demonstrate superior model consistency. The Proposed Method successfully identifies a time-invariant friction model that closely matches the ground truth. Although minor overshoots remain near the zero-crossings due to the inherent difficulty of approximating semi-discontinuous functions (steep slopes near zero), the overall shape is preserved consistently across different time instants. This confirms that the integral action prevents the parameters from overfitting to transient discontinuities, thereby capturing the underlying physical structure more effectively than the instantaneous approach.

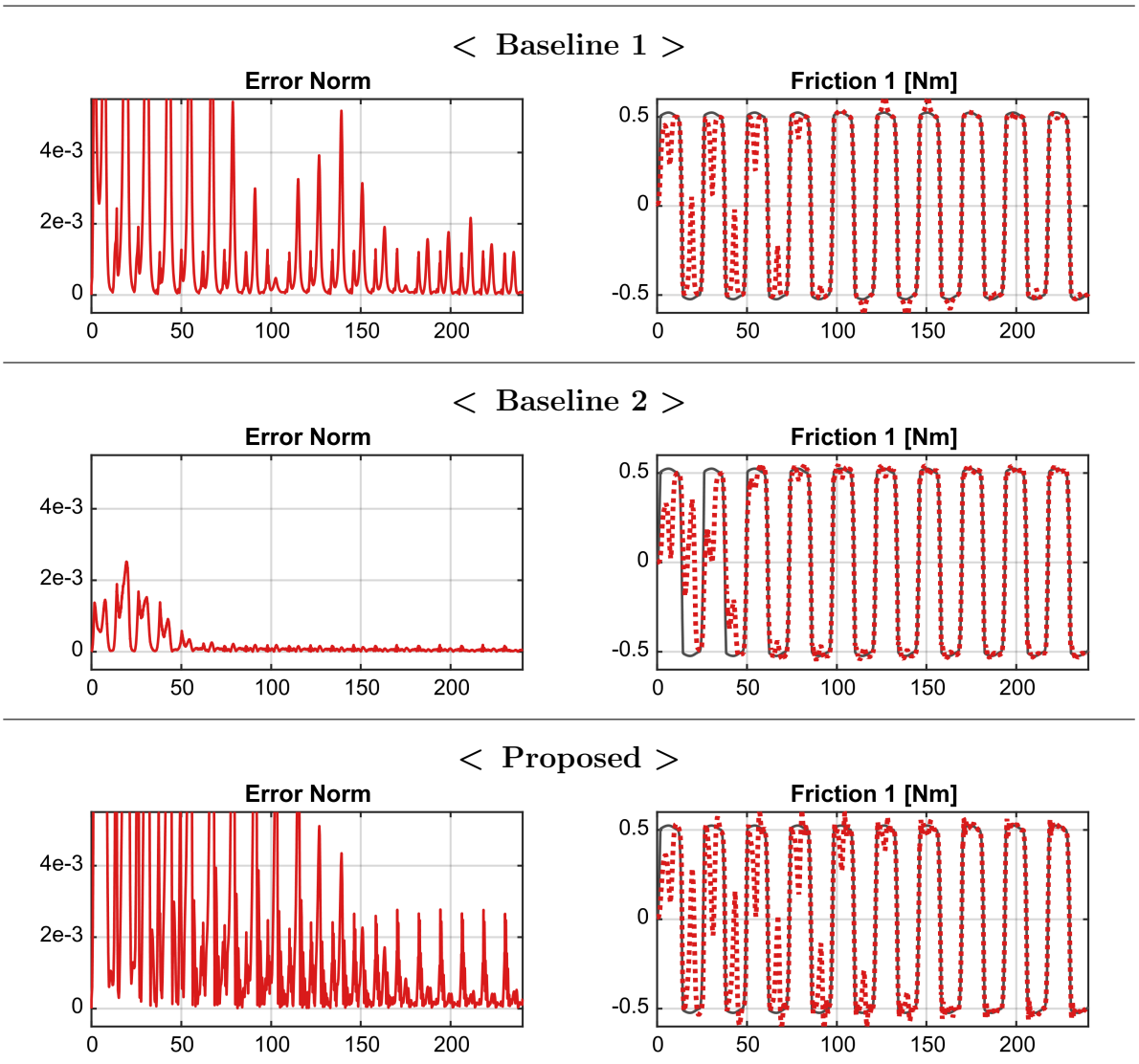
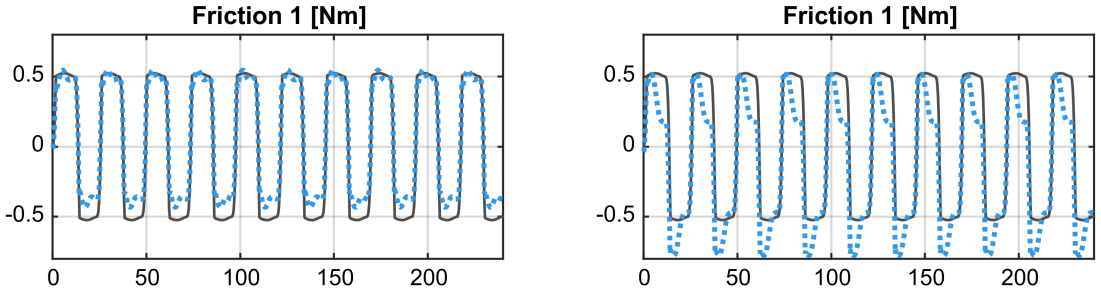


Figure 4.2. Comparison of online estimation performance of the three methods in Case 1. The left column shows the norm of the state estimation error $\|\tilde{\mathbf{x}}\|$, while the right column displays the estimated friction $\hat{\boldsymbol{g}}$ for joint 1 over time.

Model Consistency Results - Case 1

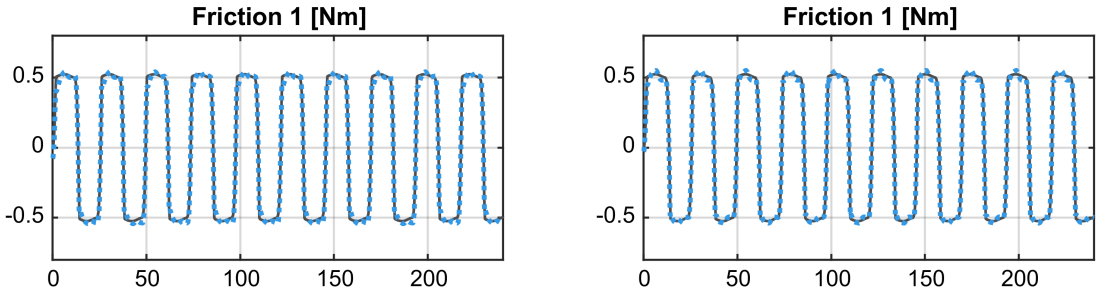
(Target : Static Joint Friction)

< Baseline 1 >



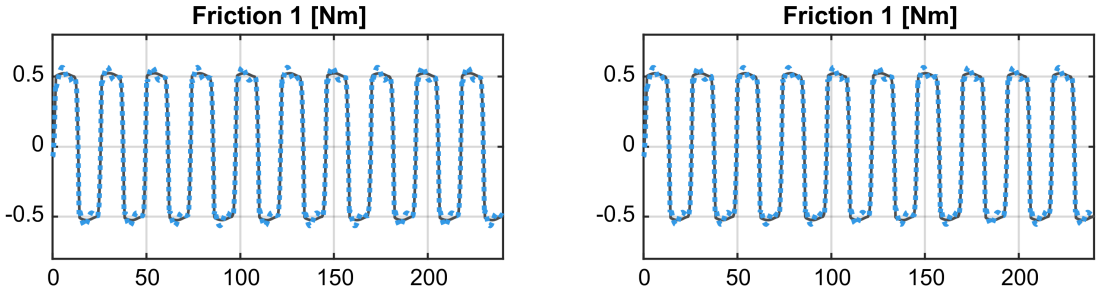
< Frozen Weight at $t = 180\,\mathrm{s}$ > < Frozen Weight at $t = 220\,\mathrm{s}$ >

< Baseline 2 >



< Frozen Weight at $t = 220\,\mathrm{s}$ > < Frozen Weight at $t = 220\,\mathrm{s}$ >

< Proposed >



< Frozen Weight at $t = 180\,\mathrm{s}$ > < Frozen Weight at $t = 220\,\mathrm{s}$ >

Figure 4.3. Comparison of model consistency performance of the three methods in Case 1. The left column shows the $\hat{\boldsymbol{g}} = \hat{\mathbf{W}}_{t=180}\,\boldsymbol{\sigma}(\hat{\mathbf{V}}_{t=180}\bar{\mathbf{x}})$, while the right column shows the $\hat{\boldsymbol{g}} = \hat{\mathbf{W}}_{t=220}\,\boldsymbol{\sigma}(\hat{\mathbf{V}}_{t=220}\bar{\mathbf{x}})$.

4.4 Case 2 : Vehicle Lateral Dynamics

4.4.1 Setup

Environment The validation is conducted within the IPG CarMaker environment, an industry-standard high-fidelity vehicle dynamics simulation software. The target vehicle is modeled as a representative commercial mid-size sedan. The vehicle configuration incorporates full multi-body dynamics, including suspension kinematics, compliance, and load transfer effects.

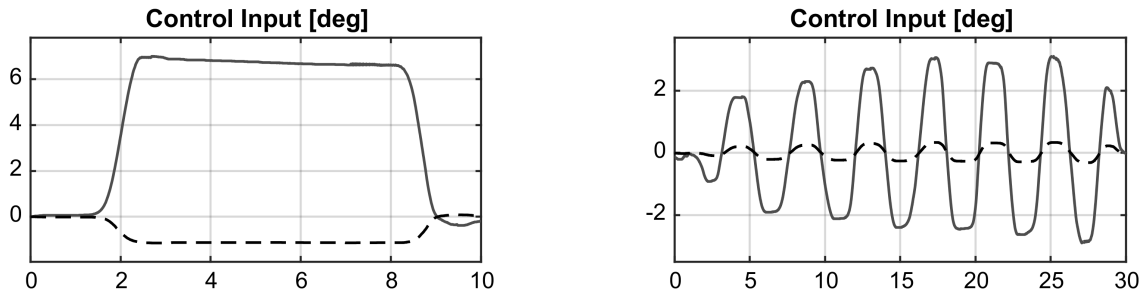
Target model The reference lateral tire forces are generated using the Pacejka Magic Formula 5.2 model implemented in IPG CarMaker. In the identification framework, the total lateral force is decomposed as

$$F_y = F_y^{\text{lin}} + F_y^{\text{res}},$$

where F_y^{lin} denotes the nominal linear tire-force component assumed to be known, and F_y^{res} represents the remaining nonlinear residual. Only the residual component F_y^{res} is treated as an unknown term to be approximated.

Control and Reference Trajectories To evaluate the identification performance under different dynamic conditions, two steering maneuvers are considered.

- **Step-Steer maneuver:** A step input in steering angle is applied to induce a rapid transient buildup of lateral tire forces followed by a quasi-steady-state response. This maneuver is intended to examine the transient adaptation behavior as well as estimation stability under near-constant operating conditions.
- **Slalom maneuver:** A sinusoidal steering input is applied to generate continuously time-varying lateral dynamics. This maneuver provides persistent excitation and is intended to assess the ability of each method to maintain structural consistency of the learned model under sustained dynamic variation.



(a) Step-steer maneuver (b) Slalom maneuver

Figure 4.4. Steering inputs for the evaluated maneuvers. Solid line: front steering angle δ_f [deg]; dashed line: rear steering angle δ_r [deg].

Configuration of Comparison Methods Vehicle lateral dynamics evolve on a faster time scale during driving, requiring shorter memory horizons. Therefore, the buffer size in Baseline 2 was reduced to 200 samples, corresponding to approximately 2 s of data (0.01×200 s). Consistently, a larger forgetting factor $\lambda = 10$ was selected for the proposed method to emphasize more recent estimation errors. Mini-batch updates were performed every 5 steps to balance responsiveness and computational cost. In detail, the configuration parameters for all methods are summarized in Table 4.4.

Item	**Baseline 1**	**Baseline 2**	**Proposed**
Initialization	$(W_0, V_0) = (10^1, 10^{-2})$ with random initialization		
Network architecture	$(n_i, n_h, n_o) = (6, 20, 4)$		
User-defined A_d	$-40\, I_{2\times2}$	(None)	$-40\, I_{2\times2}$
Learning rates (η_1, η_2)	$(5 \times 10^2,\ 1 \times 10^0)$	$(5 \times 10^2,\ 2 \times 10^{-1})$	$(5 \times 10^2,\ 1 \times 10^{-1})$
Leakage rates (ρ_1, ρ_2)	$(10^{-5},\ 10^{-5})$	(None)	$(10^{-5},\ 10^{-5})$
Update period P_{upd}	every step	every 5 steps	every step
Buffer size $\mathcal{B}$	(None)	200 samples	(None)
Mini-batch size M	(None)	32 samples	(None)
Forgetting factor λ	(None)	(None)	10

Table 4.4. Configuration summary of comparison methods.

4.4.2 Online Estimation Results (1) : Step-Steer Maneuver

Figure 4.5 compares the estimation performance during a step-steer maneuver. A notable observation is that both Baseline 2 and the **Proposed Method** exhibit relatively large estimation errors during the transient phases (step rise and fall), while Baseline 1 suppresses these errors almost immediately.

This discrepancy arises from the limitation of the input state space. The neural network inputs are restricted to the state $\mathbf{x} = [v_y, r]^\top$ and input $\mathbf{u}$. However, during rapid transient maneuvers, the actual tire forces are influenced by unmodeled dynamics and unmeasurable states (ie. roll angle), which are not explicitly provided to the network. Baseline 1 compensates for these unmodeled effects by rapidly adjusting the weights to minimize the instantaneous error, effectively "overfitting" the parameters to the transient dynamics. On the other hand, Baseline 2 and the Proposed Method, which incorporate past information (via buffer or integral state), are less sensitive to these transient unmodeled effects.

4.4.3 Model Consistency Results (1) : Step-Steer Maneuver

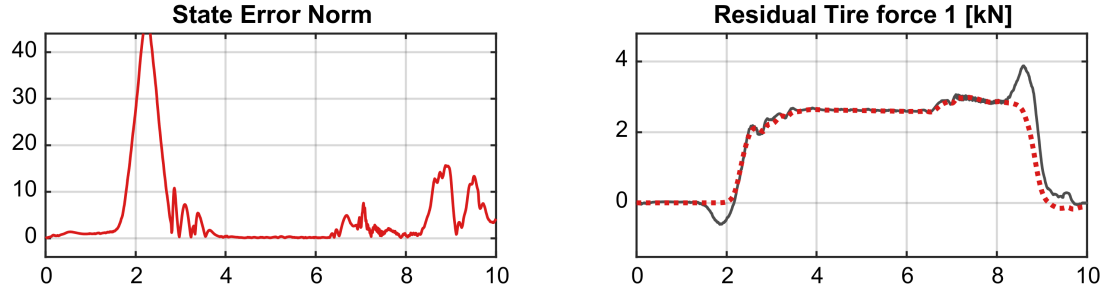
The consequence of the "overfitting" observed in the online phase is evident in the consistency test shown in Fig. 4.6. The target residual force represents the nonlinear saturation characteristic of the tires.

Baseline 1 fails to capture this saturation curve. The frozen model output fluctuates unpredictably and does not resemble the physical tire curve, confirming that the weights were manipulated to fit the transient unmodeled dynamics rather than learning the state-dependent tire mapping. In contrast, the **Proposed Method** generates a consistent saturation curve that aligns well with the ground truth, comparable to the buffer-based Baseline 2. This result validates that by filtering out the transient inconsistencies via the integral error, the proposed method successfully identifies the repeatable nonlinear structure (the tire force curve) that depends on the provided states, satisfying the assumption of model consistency.

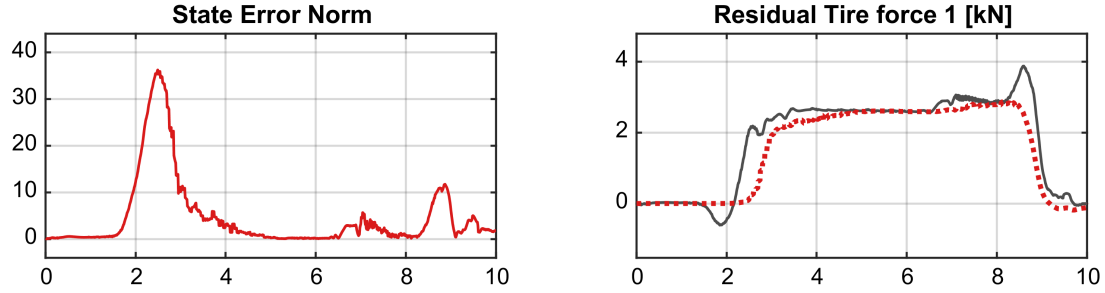
Online Estimation Results (Case 2-1)

Target - Residual Lateral Tire Force

< Baseline 1 >



< Baseline 2 >



< Proposed >

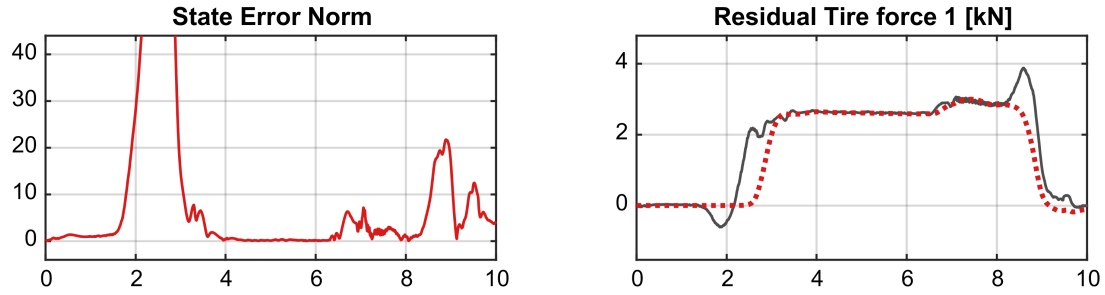
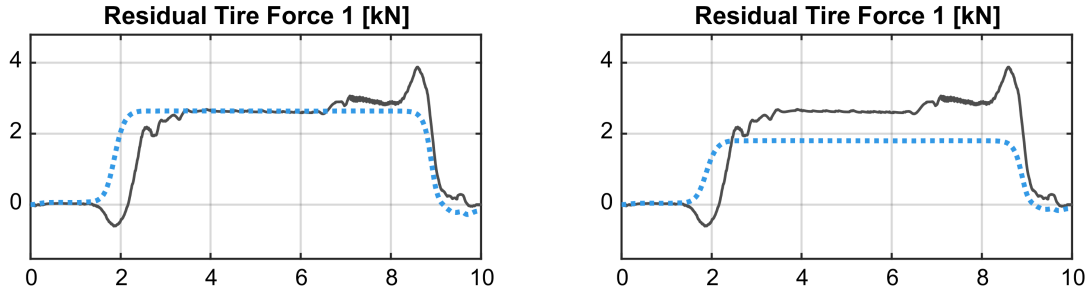


Figure 4.5. Comparison of online estimation performance of the three methods in Case 2-1. The left column shows the norm of the state estimation error $\|\tilde{\mathbf{x}}\|$, while the right column displays the estimated residual tire force $\hat{\boldsymbol{g}}$ for the front axle over time.

Model Consistency Results (Case 2-1)

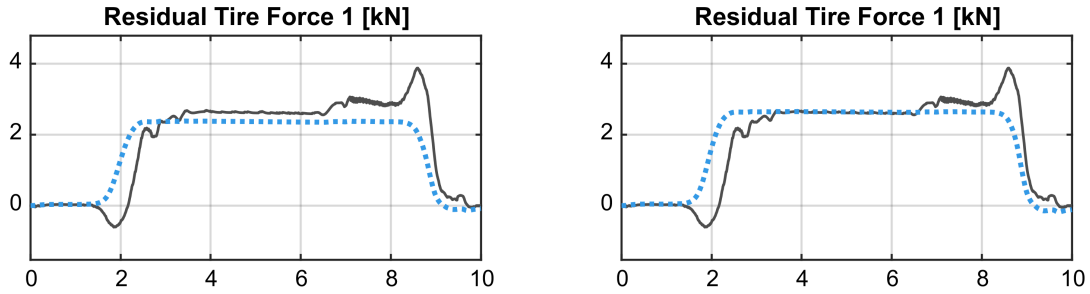
Target - Residual Lateral Tire Force

< Baseline 1 >



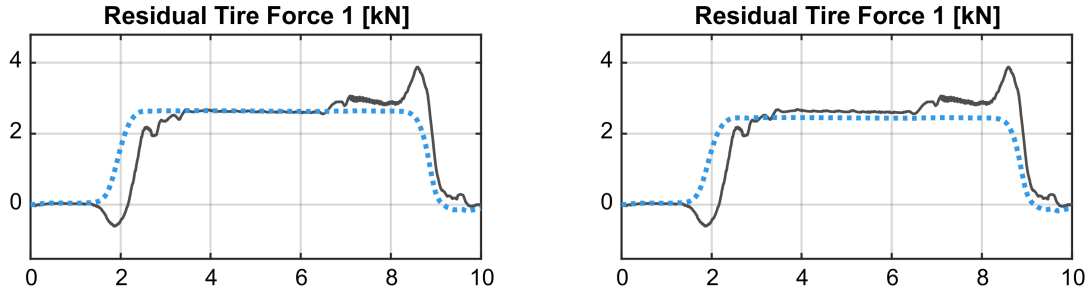
< Frozen Weight at $t = 4\,\mathrm{s}$ > < Frozen Weight at $t = 10\,\mathrm{s}$ >

< Baseline 2 >



< Frozen Weight at $t = 4\,\mathrm{s}$ > < Frozen Weight at $t = 10\,\mathrm{s}$ >

< Proposed >



< Frozen Weight at $t = 4\,\mathrm{s}$ > < Frozen Weight at $t = 10\,\mathrm{s}$ >

Figure 4.6. Comparison of model consistency performance of the three methods in Case 2-1. The left column shows the $\hat{\boldsymbol{g}} = \hat{\mathbf{W}}_{t=4}\,\boldsymbol{\sigma}(\hat{\mathbf{V}}_{t=4}\bar{\mathbf{x}})$, while the right column shows the $\hat{\boldsymbol{g}} = \hat{\mathbf{W}}_{t=10}\,\boldsymbol{\sigma}(\hat{\mathbf{V}}_{t=10}\bar{\mathbf{x}})$.

4.4.4 Online Estimation Results (2) : Slalom Maneuver

The slalom maneuver provides a condition of persistent excitation, where the vehicle states continuously vary across the operating range. As shown in Fig. 4.7, all three methods demonstrate excellent online tracking performance. The state estimation error norms for all methods remain low and bounded throughout the maneuver. Unlike the step-steer case, the continuous variation of the input signal prevents the integral state of the Proposed Method from accumulating a large static error, resulting in a convergence speed similar to that of the baselines. This confirms that under sufficiently excited conditions, the proposed method maintains competitive real-time estimation performance.

4.4.5 Model Consistency Results (2) : Slalom Maneuver

Figure 4.8 compares the consistency of the models learned during the slalom maneuver. Since the input provides persistent excitation, Baseline 1 performs better than in the step-steer case; however, slight deviations are still visible at the peaks of the tire force, suggesting a tendency to forget previously learned dynamics as the operating point shifts. Baseline 2 achieves near-perfect reconstruction, as the buffer ensures that data from different operating points are used simultaneously for optimization. The **Proposed Method** demonstrates a reconstruction capability almost identical to that of Baseline 2. The integral error term effectively aggregates the information from the time-varying residuals, allowing the network to learn a consistent global mapping of the nonlinear tire dynamics. This result is significant as it validates that the proposed method can achieve the robustness of memory-based learning without the computational and memory overhead associated with maintaining and sampling a data buffer.

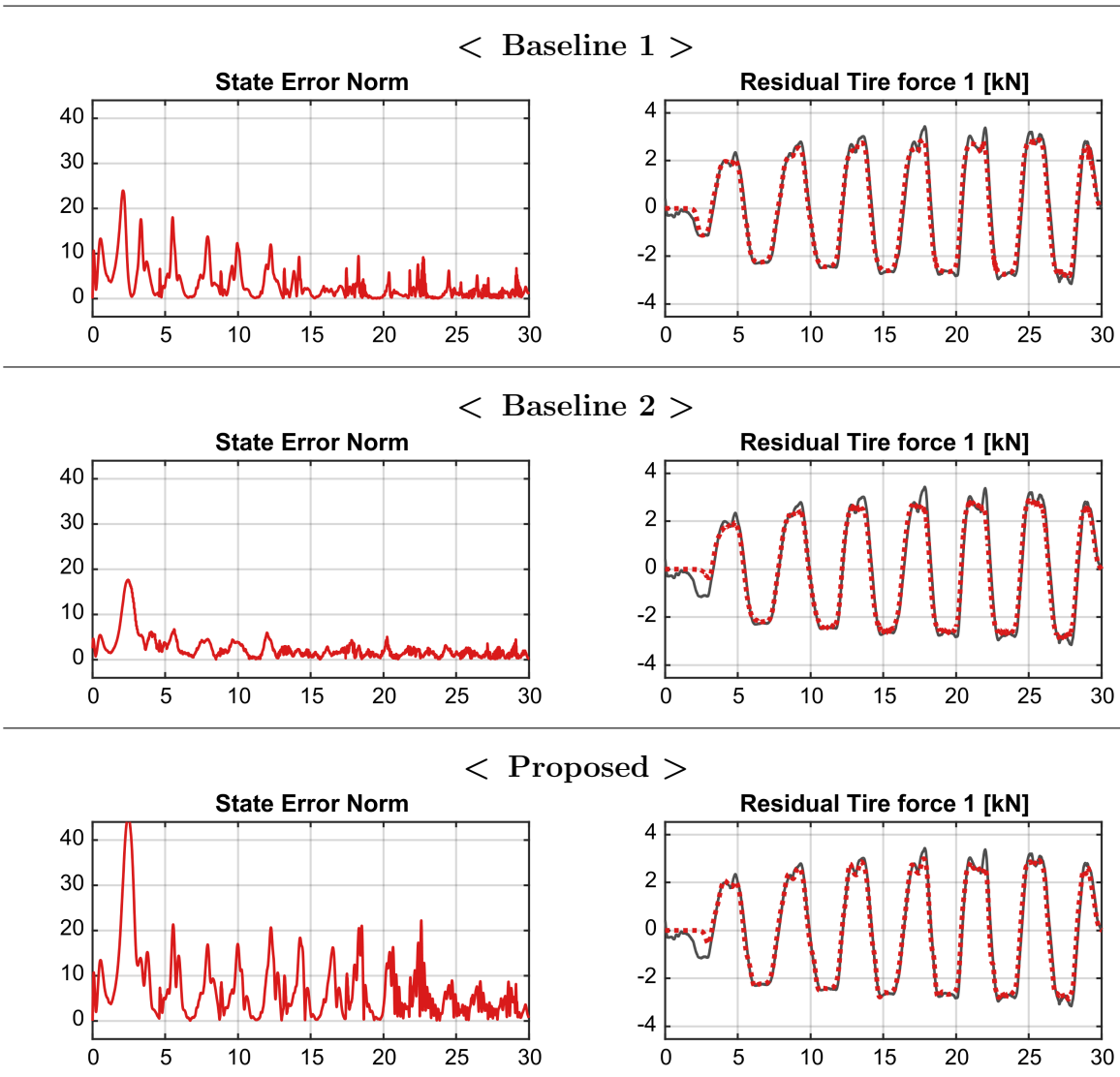
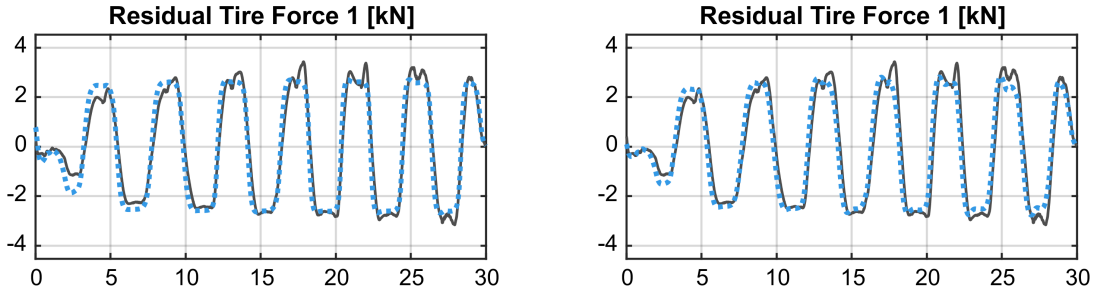


Figure 4.7. Comparison of online estimation performance of the three methods in Case 2-2. The left column shows the norm of the state estimation error $\|\tilde{\mathbf{x}}\|$, while the right column displays the estimated residual tire force $\hat{\boldsymbol{g}}$ for the front axle over time.

Model Consistency Results (Case 2-2)

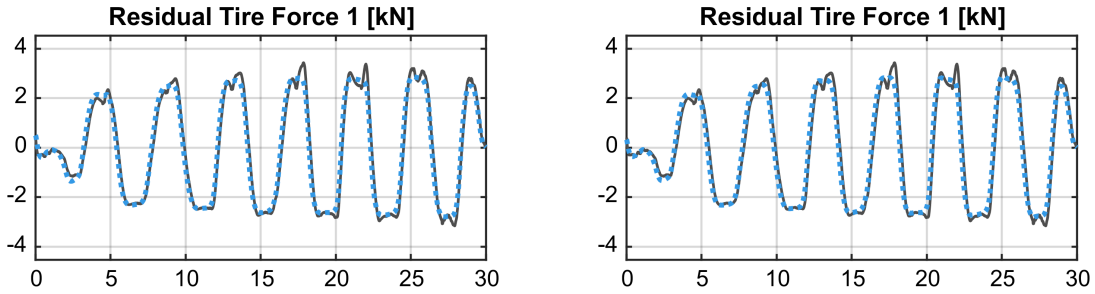
Target - Residual Lateral Tire Force

< Baseline 1 >



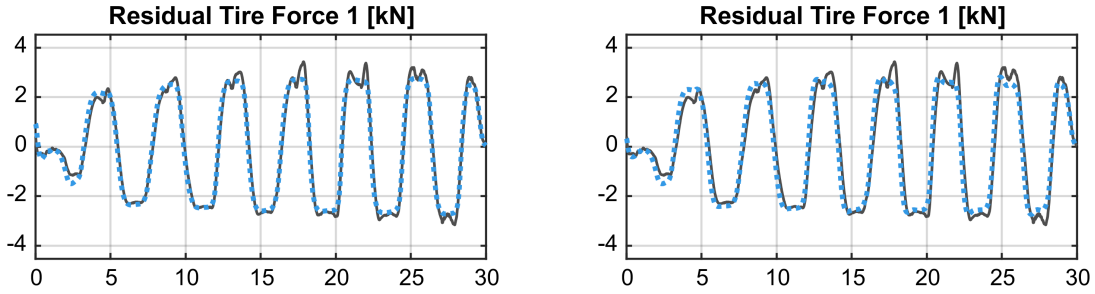
< Frozen Weight at $t = 15\,\mathrm{s}$ > < Frozen Weight at $t = 25\,\mathrm{s}$ >

< Baseline 2 >



< Frozen Weight at $t = 15\,\mathrm{s}$ > < Frozen Weight at $t = 25\,\mathrm{s}$ >

< Proposed >



< Frozen Weight at $t = 15\,\mathrm{s}$ > < Frozen Weight at $t = 25\,\mathrm{s}$ >

Figure 4.8. Comparison of model consistency performance of the three methods in Case 2-2. The left column shows the $\hat{\boldsymbol{g}} = \hat{\mathbf{W}}_{t=15}\,\boldsymbol{\sigma}(\hat{\mathbf{V}}_{t=15}\bar{\mathbf{x}})$, while the right column shows the $\hat{\boldsymbol{g}} = \hat{\mathbf{W}}_{t=25}\,\boldsymbol{\sigma}(\hat{\mathbf{V}}_{t=25}\bar{\mathbf{x}})$.

4.5 Discussion

This section consolidates the results presented in 4.3 – 4.4 and discusses their implications from the perspective of *online identification*. Rather than reiterating case-specific behaviors, the focus is on interpreting the observations in terms of fundamental trade-offs among (1) estimation accuracy, (2) model consistency, and (3) computational cost/real-time suitability.

4.5.1 Estimation Accuracy Versus Model Consistency

Table 4.5 summarizes online estimation accuracy and model consistency across all validation cases. Two distinct aspects are reported: **online estimation accuracy** during adaptation or learning phases, and **model consistency**, which reflects the consistency of the learned model when adaptation is disabled.

Baseline 1 often exhibits competitive online estimation accuracy because its update law directly reacts to the current estimation error. However, its model consistency degrades across cases, indicating that the adapted weights do not reliably preserve previously acquired information. In other words, Baseline 1 is better interpreted as an online compensation mechanism whose weights are optimized for instantaneous regulation, rather than as a persistent identification model.

Baseline 2 tends to achieve strong frozen-weight reproducibility because it explicitly reuses past data and therefore enforces consistency in the learned mapping. Nevertheless, this advantage is obtained through explicit buffer and batch-wise optimization, which introduces algorithmic overhead and may create intermittent computation spikes.

Proposed method may exhibit similar or slightly degraded online accuracy compared to Baseline 1 in some cases. However, its model consistency remains consistently closer to Baseline 2 than to Baseline 1. This supports the central claim of this thesis that past estimation errors, when incorporated into the adaptive law via an integral error state, can improve the consistency of the learned model without resorting to

explicit sample storage or sliding-window optimization.

Case	Metric (RMSE)	Baseline 1	Baseline 2	Proposed
Case 1	Estimation Accuracy	0.1238(⋆)	0.2241	0.1793
	Model Consistency (180 s)	0.1344	0.0427(⋆)	0.1109
	Model Consistency (220 s)	0.2891	0.0317(⋆)	0.0470(⋆)
Case 2-1	Estimation Accuracy	0.5777(⋆)	0.9248	0.9203
	Model Consistency (015 s)	0.6955	0.6619	0.6314(⋆)
	Model Consistency (025 s)	1.0534	0.7358(⋆)	0.9019
Case 2-2	Estimation Accuracy	0.8371(⋆)	0.8427	0.8236(⋆)
	Model Consistency (004 s)	0.8829	0.7944	0.6031(⋆)
	Model Consistency (010 s)	0.8950	0.8496(⋆)	0.8645(⋆)

Table 4.5. Summary of estimation accuracy and model consistency across all cases.

4.5.2 Computational Cost and Real-Time Suitability

For online learning in real-time control and estimation systems, the primary requirement is strict deadline compliance [39–41]. In particular, learning methods that explicitly store past data and perform batch-type updates raise immediate concerns regarding computational load and worst-case execution behavior [42, 43]. This is especially relevant for Baseline 2, which relies on a sliding-window buffer and periodic mini-batch updates.

Evaluation Setup To evaluate real-time suitability, the simulation for each method was executed independently over 100 runs under identical conditions. For each run, per-step computation time was recorded, and summary statistics were extracted to characterize both average efficiency and worst-case behavior. The resulting metrics are reported in Table 4.6.

Results The mean computation time indicates that all methods are lightweight on average, including Baseline 2. However, metrics related to worst-case behavior reveal a clear distinction. Baseline 2 exhibits a substantially larger mean run-wise maximum

and standard deviation, as well as an extreme global peak and a non-zero deadline miss rate. This behavior is consistent with its periodic mini-batch update mechanism, which concentrates gradient computation at discrete update instants and induces intermittent computation bursts. In contrast, Baseline 1 and the proposed method maintain small run-wise maxima, low variability, and zero deadline misses across all runs. Importantly, this burst behavior is not merely an implementation artifact but a structural consequence of the periodic learning paradigm in Baseline 2: weights are updated only when the optimization routine is invoked and are held constant between invocations (sample-and-hold), causing computation to be temporally clustered. In contrast, Baseline 1 and the proposed method maintain small run-wise maxima, low variability, and zero deadline misses across all runs, since their adaptive laws treat weights as internal states that are updated continuously in time and implemented on the sampling grid via uniform per-step integration.

These results demonstrate that, although Baseline 2 is efficient in terms of average computation time, its buffer-based mini-batch learning structure poses a fundamental limitation for real-time deployment. By contrast, the proposed method achieves improved model consistency while maintaining bounded per-step computation and reliable real-time suitability.

Metric	**Baseline 1**	**Baseline 2**	**Proposed**
Mean (computation time) [ms]	0.0047	0.0118	0.0050
Mean (run-wise max) [ms]	0.2172	1.0565	0.2273
Std. (run-wise max) [ms]	0.1977	2.7081 (×)	0.1807
Global peak computation time [ms]	1.4914	21.806 (×)	1.6880
Deadline miss rate [%]	0.0	8.0 (×)	0.0

Table 4.6. Timing summary : deadline = 2.0 ms (500 Hz).

Chapter 5

Conclusion and Future Work

5.1 Conclusion

This thesis proposed an integral error-based adaptive neural identifier designed to bridge the gap between real-time adaptation and consistent system identification for uncertain nonlinear systems without explicit data storage. The adaptive law was developed where past estimation information is implicitly incorporated through a dynamic integral error state with an exponential forgetting factor. This formulation preserves the benefits of continuous-time adaptation while enabling the parameter update process to reflect accumulated error information without explicit data storage. The effectiveness of the proposed framework was validated through numerical simulations on a 2-DOF robot manipulator and a high-fidelity vehicle lateral dynamics model. The results demonstrated that the proposed method identifies a consistent and reusable functional representation of unknown nonlinearities matching the consistency of buffer-based methods while maintaining the low computational cost of conventional identifiers. Furthermore, timing analysis confirmed that the proposed approach ensures stable timing, making it highly suitable for real-time control and estimation applications. In conclusion, the integral error-based identifier provides a robust and efficient solution for online system identification where both real-time responsiveness and model-level validity are essential.

5.2 Future Work

While the proposed integral error-based adaptive neural identifier has shown promising results, several avenues for future research remain.

- First, extending the framework to accommodate time-varying or non-stationary

systems could enhance its applicability in dynamic environments. Investigating adaptive forgetting factors that adjust based on system behavior may further improve identification accuracy.

- Second, integrating the identifier with advanced control strategies, such as model predictive control or reinforcement learning, could lead to more robust and adaptive control systems capable of handling complex tasks.

- Third, exploring the use of alternative function approximators, such as convolutional neural networks or recurrent neural networks, may provide improved performance for systems with spatial or temporal dependencies.

- Finally, conducting extensive experimental validations on real-world systems, such as autonomous vehicles or robotic platforms, would provide valuable insights into the practical implementation challenges and performance of the proposed identifier.

Addressing these areas in future work will further enhance the capabilities and applicability of the integral error-based adaptive neural identifier in various domains.

Summary

Integral Error-Based Adaptive Neural Identifier

for a Class of Uncertain Nonlinear Systems

This thesis investigated online identification of unknown nonlinear dynamics, focusing on how past information is reflected in weight updates. Rather than emphasizing instantaneous estimation accuracy alone, the study addressed the consistency of the learned model over time under real-time constraints.

Instantaneous error-based adaptive neural identifiers were considered as a baseline, providing fast and computationally efficient updates. However, since no historical information is retained, the adapted parameters tend to reflect short-term behaviors, resulting in limited model consistency when the learned model is reused.

In contrast, buffer-based mini-batch online learning methods improve weight consistency by explicitly accumulating past data through batch-wise updates. Nevertheless, their memory usage, irregular update timing, and computational overhead restrict their applicability to real-time embedded systems.

To address this trade-off, this thesis proposed an integral error–based adaptive neural identifier, which incorporates past estimation errors through a dynamic integral state without explicit data storage. Simulation results demonstrate that the proposed method improves model consistency compared to instantaneous adaptation, while preserving real-time feasibility that buffer-based approaches fail to maintain.

References

1. B. Armstrong-Hélouvry, P. Dupont, and C. C. De Wit, "A survey of models, analysis tools and compensation methods for the control of machines with friction," *Automatica*, vol. 30, no. 7, pp. 1083–1138, 1994.

2. C. Canudas de Wit, H. Olsson, K. Astrom, and P. Lischinsky, "A new model for control of systems with friction," *IEEE Transactions on Automatic Control*, vol. 40, no. 3, pp. 419–425, 1995.

3. H. Pacejka, *Tire and vehicle dynamics*. Elsevier, 2005.

4. W.-H. Chen, J. Yang, L. Guo, and S. Li, "Disturbance-observer-based control and related methods—an overview," *IEEE Transactions on Industrial Electronics*, vol. 63, no. 2, pp. 1083–1095, 2016.

5. J. Han, "From pid to active disturbance rejection control," *IEEE Transactions on Industrial Electronics*, vol. 56, no. 3, pp. 900–906, 2009.

6. E. F. Camacho and C. Bordons, "Constrained model predictive control," in *Model predictive control*, pp. 177–216, Springer, 2007.

7. R. Isermann, *Fault-diagnosis systems: an introduction from fault detection to fault tolerance*. Springer Science & Business Media, 2005.

8. F. Tao, H. Zhang, A. Liu, and A. Y. Nee, "Digital twin in industry: State-of-the-art," *IEEE Transactions on industrial informatics*, vol. 15, no. 4, pp. 2405–2415, 2018.

9. F. Lewis, S. Jagannathan, and A. Yesildirak, *Neural network control of robot manipulators and non-linear systems*. CRC press, 2020.

10. H. K. Khalil and J. W. Grizzle, *Nonlinear systems*, vol. 3. Prentice hall Upper Saddle River, NJ, 2002.

11. P. A. Ioannou and J. Sun, *Robust adaptive control*, vol. 1. PTR Prentice-Hall Upper Saddle River, NJ, 1996.

12. L. Ljung, "System identification," in *Signal analysis and prediction*, pp. 163–173, Springer, 1998.

13. L. Hewing, K. P. Wabersich, M. Menner, and M. N. Zeilinger, "Learning-based model predictive control: Toward safe learning in control," *Annual Review of Control, Robotics, and Autonomous Systems*, vol. 3, no. 1, pp. 269–296, 2020.

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

15. L. Zhang, S. Lu, and Z.-H. Zhou, "Adaptive online learning in dynamic environments," *Advances in neural information processing systems*, vol. 31, 2018.

16. R. Hadsell, D. Rao, A. A. Rusu, and R. Pascanu, "Embracing change: Continual learning in deep neural networks," *Trends in cognitive sciences*, vol. 24, no. 12, pp. 1028–1040, 2020.

17. J. Jia, W. Zhang, K. Guo, J. Wang, X. Yu, Y. Shi, and L. Guo, "Evolver: Online

learning and prediction of disturbances for robot control," *IEEE Transactions on Robotics*, vol. 40, pp. 382–402, 2024.

18. L. Wang, X. Zhang, H. Su, and J. Zhu, "A comprehensive survey of continual learning: Theory, method and application," *IEEE transactions on pattern analysis and machine intelligence*, vol. 46, no. 8, pp. 5362–5383, 2024.

19. O. Ogunmolu, X. Gu, S. Jiang, and N. Gans, "Nonlinear systems identification using deep dynamic neural networks," *arXiv preprint arXiv:1610.01439*, 2016.

20. F. Abdollahi, H. A. Talebi, and R. V. Patel, "A stable neural network-based observer with application to flexible-joint manipulators," *IEEE Transactions on Neural Networks*, vol. 17, no. 1, pp. 118–129, 2006.

21. M. M. Polycarpou, "Stable adaptive neural control scheme for nonlinear systems," *IEEE Transactions on Automatic control*, vol. 41, no. 3, pp. 447–451, 2002.

22. R. G. Hart, E. J. Griffis, O. S. Patil, and W. E. Dixon, "Lyapunov-based physics-informed long short-term memory (LSTM) neural network-based adaptive control," *IEEE Control Systems Letters*, 2023.

23. E. J. Griffis, O. S. Patil, R. G. Hart, and W. E. Dixon, "Lyapunov-based long short-term memory (lb-lstm) neural network-based adaptive observer," *IEEE Control Systems Letters*, vol. 8, pp. 97–102, 2024.

24. W. A. Makumi, O. S. Patil, and W. E. Dixon, "Lyapunov-based adaptive deep system identification for approximate dynamic programming," *Automatica*, vol. 180, p. 112462, 2025.

25. A. Parikh, R. Kamalapurkar, and W. E. Dixon, "Integral concurrent learning: Adaptive control with parameter convergence using finite excitation," *International Journal of Adaptive Control and Signal Processing*, vol. 33, no. 12, pp. 1775–1787, 2019.

26. 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.

27. P. A. Ioannou and P. V. Kokotovic, "Instability analysis and improvement of robustness of adaptive control," *Automatica*, vol. 20, no. 5, pp. 583–594, 1984.

28. G. Chowdhary, H. A. Kingravi, J. P. How, and P. A. Vela, "Bayesian nonparametric adaptive control using gaussian processes," *IEEE transactions on neural networks and learning systems*, vol. 26, no. 3, pp. 537–550, 2014.

29. G. Chowdhary and E. Johnson, "Concurrent learning for convergence in adaptive control without persistency of excitation," in *49th IEEE Conference on Decision and Control (CDC)*, pp. 3674–3679, IEEE, 2010.

30. R. G. Hart, O. S. Patil, Z. I. Bell, and W. E. Dixon, "System identification and control using lyapunov-based deep neural networks without persistent excitation: A concurrent learning approach," 2025.

31. K. Chua, R. Calandra, R. McAllister, and S. Levine, "Deep reinforcement learning in a handful of trials using probabilistic dynamics models," *Advances in neural information processing systems*, vol. 31, 2018.

32. R. Rajamani, *Tire-Road Friction Measurement on Highway Vehicles*, pp. 433–465. Boston, MA: Springer US, 2006.

33. Y. Song, A. Mavalankar, W. Sun, and S. Gao, "Provably efficient model-based policy adaptation," *arXiv preprint arXiv:2006.08051*, 2020.

34. T. Chaffre, G. LE CHENADEC, E. CHAUVEAU, B. CLEMENT, K. Sammut, *et al.*, "Learning stochastic adaptive control using a bio-inspired experience replay," *Authorea Preprints*, 2022.

35. A. Nagabandi, G. Kahn, R. S. Fearing, and S. Levine, "Neural network dynamics for model-based deep reinforcement learning with model-free fine-tuning," in *2018 IEEE international conference on robotics and automation (ICRA)*, pp. 7559–7566, IEEE, 2018.

36. P. Jain, S. S. Kowshik, D. Nagaraj, and P. Netrapalli, "Streaming linear system identification with reverse experience replay," *arXiv preprint arXiv:2103.05896*, 2021.

37. M. Champneys, G. I. Beintema, R. Tóth, M. Schoukens, and T. J. Rogers, "Baseline results for selected nonlinear system identification benchmarks," *IFAC-PapersOnLine*, vol. 58, no. 15, pp. 474–479, 2024.

38. M.-B. Radac and D.-P. Chirla, "Near real-time online reinforcement learning with synchronous or asynchronous updates," *Scientific Reports*, vol. 15, no. 1, p. 17158, 2025.

39. G. Bernat, A. Burns, and A. Liamosi, "Weakly hard real-time systems," *IEEE Transactions on Computers*, vol. 50, no. 4, pp. 308–321, 2001.

40. G. C. Buttazzo, *Hard real-time computing systems: predictable scheduling algorithms and applications*. Springer, 1997.

41. R. I. Davis and A. Burns, "A survey of hard real-time scheduling for multiprocessor systems," *ACM computing surveys (CSUR)*, vol. 43, no. 4, pp. 1–44, 2011.

42. N. Vreman, A. Cervin, and M. Maggio, "Stability and performance analysis of control systems subject to bursts of deadline misses," in *33rd Euromicro Conference on Real-Time Systems (ECRTS 2021)*, Schloss Dagstuhl-Leibniz-Zentrum für Informatik, 2021.

43. S. A. Edwards and E. A. Lee, "The case for the precision timed (pret) machine," in *Proceedings of the 44th annual Design Automation Conference*, pp. 264–265, 2007.

Acknowledgements

I would like to express my sincere gratitude to my advisor, Prof. Kyunghwan Choi, for his generous guidance and constructive feedback throughout this research. His insights and mentorship greatly contributed to the direction and quality of this thesis. I am deeply grateful to my family for their unconditional support throughout my studies; their trust and encouragement have been a constant source of strength. I also extend my sincere thanks to the one who has been by my side throughout this journey for her unwavering support and encouragement. Her patience and understanding helped me persevere during challenging times. Above all, I give thanks to God for His grace and faithful guidance throughout this journey.