NAZARBAYEV UNIVERSITY

Neural-Augmented Modeling and Control of Hybrid Series-Parallel Elastic Actuators for Humanoid Robotics

by

Timur Umurzakov

Submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy in Robotics Engineering

Date of Completion
April, 2026

Neural-Augmented Modeling and Control of Hybrid Series-Parallel Elastic Actuators for Humanoid Robotics

by
Timur Umurzakov

Submitted in partial fulfillment of the requirements for the degree of
Doctor of Philosophy in Robotics Engineering

Nazarbayev University

April, 2026

Supervised by
Prof. Michele Folgheraiter (lead supervisor)
Prof. Matteo Rubagotti (co-supervisor)
Prof. Giuseppina Gini (external supervisor)

Declaration

I, Timur Umurzakov, declare that the research contained in this thesis, unless otherwise formally indicated within the text, is the author's original work. The thesis has not been previously submitted to this or any other university for a degree and does not incorporate any material already submitted for a degree.

Signature:

Date:

BLANK

Abstract

Humanoid robots require actuators that are adaptive, efficient, and cost-effective, yet conventional solutions such as harmonic drives often struggle to balance compliance, precision, and affordability. This thesis introduces a neural-augmented modeling and control framework built around a novel hybrid Series-Parallel Elastic Actuator (SEA-PEA). The actuator integrates a BLDC motor, a planetary gearbox, and a rotary pneumatic element, leveraging controlled gearbox backlash to switch between series and parallel elastic behaviors. Within the backlash region, it functions as a Series Elastic Actuator (SEA), providing tunable stiffness through pneumatic pressure, enhancing compliance and safety, and enabling the storage and release of mechanical energy. Outside the region, it transitions to a Parallel Elastic Actuator (PEA), delivering stiff, precise, and energy-efficient torque transmission. To capture these nonlinear dynamics, a hybrid modeling strategy is proposed: an Echo State Network (ESN) for the SEA regime and a Wiener model for the PEA regime. Experimental validation shows that this architecture reduces modeling error compared to conventional ANN approaches. Beyond modeling, the artificial neural network framework was applied to enhance control of the hybrid SEA–PEA actuator itself, showing improved tracking performance and robustness compared to traditional control approaches. Building on this foundation, the same neural modeling and control principles were extended to a 12-DOF humanoid robot, where ESN-based Computed Torque Control (CTC) enabled reliable sim-to-real transfer validated both in simulation (PyBullet) and on a physical robot prototype. Compared to PID and torque control, neural-augmented CTC achieved lower mean square error (MSE) and reduced energy consumption, with further improvements under external load when combined with online Recursive Least Squares (RLS) adaptation.

BLANK

Acknowledgments

I would like to express my sincere gratitude to all the individuals who have supported and guided me throughout the course of this PhD thesis work in the Robotics Department of Nazarbayev University.

First and foremost, I extend my deepest appreciation to my lead supervisor, Professor Michele Folgheraiter, for his invaluable expertise, insightful feedback, and unwavering encouragement. His guidance has been instrumental in shaping this work, and I am truly grateful for his mentorship and the time he dedicated to my development.

I would also like to thank my external supervisor, Professor Giuseppina Gini, for her valuable insights and contributions to this research.

In addition, I am thankful to my internal co-supervisor, Professor Matteo Rubagotti, for his time, thoughtful feedback, and ongoing support throughout the development of this thesis. The perspectives and expertise of my supervisors have been essential in refining my work and broadening my understanding of the field of robotics.

I am also grateful to the academic managers and project administrators for their assistance and organizational support during my studies. I extend special thanks to my co-workers and colleagues in the Humanoid Robotics Laboratory for their collaboration, constructive discussions, and technical support.

Last but not least, I am deeply grateful to my family and friends for their unwavering emotional, mental, and financial support throughout this journey. In particular, I wish to express my heartfelt gratitude to my wife. During the course of my PhD studies, she was diagnosed with breast cancer, and bravely underwent surgery and chemotherapy. Despite the physical and emotional challenges, she remained a pillar of strength and continued to support me through every stage of this academic journey. Her resilience and courage were a constant source of inspiration, and I am profoundly thankful for her recovery and for her presence in my life.

During my PhD studies at Nazarbayev University, I deepened my knowledge of robotics control systems, artificial neural networks (ANNs), and humanoid robot motion. Additionally, my research work in the Humanoid Robotics Laboratory significantly enhanced my practical skills in 3D printing, PCB soldering, and the design and assembly of various robotic components.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

Humanoid robots demand actuators that balance compliance, precision, and efficiency. Conventional harmonic drives offer high stiffness but poor adaptability, while SEAs improve safety at the cost of speed and control complexity. Recent advances in neural modeling and adaptive control suggest new pathways toward dynamic, robust humanoid motion. Despite progress in compliant actuation, there remains no unified framework that combines the advantages of series and parallel elasticity while maintaining accurate, adaptive control across regimes. Similarly, transferring neural control from simulation to real humanoids remains a challenge due to model mismatch and nonlinearities. The principal objective of this research is to develop a unified neural-augmented modeling and control framework applicable both to hybrid actuators and to humanoid robotic systems. The study begins with the conception and realization of a novel hybrid Series–Parallel Elastic Actuator (SEA–PEA), which integrates a BLDC motor, a planetary gearbox, and a rotary pneumatic element. The first goal is to design and characterize this actuator such that it can exploit controlled gearbox backlash to transition between series and parallel elastic modes, thereby combining compliance and precision within a single mechanism. The second objective is to establish a hybrid modeling approach that accurately captures the actuator's nonlinear dynamics, employing an Echo State Network (ESN) for the SEA regime and a Wiener model for the PEA regime. A third objective focuses on developing and validating a neural-based control strategy capable of improving torque tracking and robustness beyond conventional feedback schemes. Finally, the methodology is extended to a twelve-degree-of-freedom humanoid robot, demonstrating the feasibility of the proposed framework for sim-to-real transfer and adaptive motion control in complex, multijoint systems. The research methodology is structured in two main parts. The first part concentrates on actuator-level investigation, encompassing the mechanical design, mathematical modeling, and experimental validation of the hybrid SEA–PEA. This phase integrates neural modeling and control algorithms tested under various operational conditions to assess compliance, precision, and energy efficiency. The second part applies the developed principles to humanoid locomotion control. Using a detailed URDF model and the PyBullet simulation environment, an ESN-based Computed Torque Control (CTC) scheme is trained and later transferred to the physical prototype. Through iterative testing and online adaptation using Recursive Least Squares (RLS), the humanoid robot achieves improved tracking accuracy and dynamic stability under

real-world disturbances. Together, these stages establish a coherent progression from component-level development to full-system validation. This thesis makes several key contributions to the field of compliant and intelligent actuation for humanoid robots. The hybrid SEA–PEA actuator combines the benefits of series and parallel elasticity within a single, compact design, offering a new approach to tunable compliance and energy-efficient torque transmission. The proposed hybrid modeling framework integrates data-driven and analytical techniques, achieving higher accuracy in representing complex nonlinear behaviors than conventional neural models. The neural-augmented control architecture further enhances actuator and system-level performance, enabling adaptive behavior without explicit dynamic modeling. Extending these principles to a 12-DOF humanoid robot, the research demonstrates effective sim-to-real transfer using ESN-based CTC, reducing energy consumption and improving robustness to external loads. Collectively, these results advance both the methodology and practical realization of neural-driven compliant actuation and adaptive control in humanoid robotics.

1.1 Thesis Outline

The content of this thesis work is structured as follows, where both the SEA-PEA Elastic Actuator and the humanoid robot prototype are characterized in each chapter:

- Chapter 2 provides a comprehensive background study encompassing three major topics relevant to this work: elastic actuator design principles, artificial neural network (ANN) modeling approaches, and adaptive control strategies applied to humanoid robotics.
- Chapter 3 describes the experimental setups in detail. The first part focuses on the mechanical, electrical, and pneumatic design of the SEA–PEA actuator, while the second part presents the electro-mechanical design of the humanoid prototype, including its sensor and actuation subsystems.
- Chapter 4 outlines the methodological framework used in the experiments. It details the mathematical formulations, modeling procedures, selection of ANN architectures and control strategies applied to both the SEA–PEA actuator and the humanoid prototype.
- Chapter 5 specifies the experimental procedures for both setups. This includes data acquisition protocols, test conditions, and control scenarios employed to evaluate system performance.
- Chapter 6 presents the experimental results. The outcomes of the SEA–PEA actuator experiments are first discussed, followed by those of the humanoid

prototype, highlighting their respective performance metrics and dynamic behaviors.

- Chapter 7 provides an in-depth discussion and interpretation of the obtained results. The first section focuses on the SEA–PEA actuator, emphasizing its elastic properties and control performance, while the second section discusses the humanoid prototype results in the context of adaptive neural control and real-world implementation.

- Chapter 8 concludes the thesis by summarizing the key findings and demonstrating how the two experimental parts collectively form an integrated framework for modeling and control of adaptive robotic systems. The chapter also discusses future research directions and potential extensions of this work.

1.2 Related publications

The following papers are the original materials, which are combined and paraphrased in this thesis work.

[1] T. Umurzakov, S. Yessirkepov and M. Folgheraiter, "Neural-Augmented Hybrid Dynamic Modeling of a Series-Parallel Elastic Actuator," in IEEE Access, vol. 13, pp. 104630-104650, 2025, doi: 10.1109/ACCESS.2025.3580222.

[2] T. Umurzakov, S. Yessirkepov and M. Folgheraiter, "Fast Prototyping and Testing of a New Full Scale Bipedal Robot," 2022 7th International Conference on Robotics and Automation Engineering (ICRAE), Singapore, 2022, pp. 215-221, doi: 10.1109/ICRAE56463.2022.10056190.

1.3 Other related publications

In addition to the aforementioned papers, the next list of publications provide the additional information (prerequisite). In this case, I took some ideas and information from those sources as a second and third author.

[3] Folgheraiter, M.; Yessirkepov, S.; Umurzakov, T. NU-Biped-4.5: A Lightweight and Low-Prototyping-Cost Full-Size Bipedal Robot. Robotics 2024, 13, 9. https://doi.org/10.3390/robotics13010009

[4] M. Folgheraiter, T. Umurzakov and S. Yessirkepov, "A Servomotor with Adjustable Stiffness for Humanoid Robotics Application," ACTUATOR 2022; International Conference and Exhibition on New Actuator Systems and Applications, Mannheim, Germany, 2022, pp. 1-4.

[5] S. Yessirkepov, T. Umurzakov and M. Folgheraiter, "Design and Analysis of a Parallel Elastic Shoulder Joint for Humanoid Robotics Application," in IEEE Access, vol. 13, pp. 8761-8778, 2025, doi: 10.1109/ACCESS.2025.3527873.

[6] Yessirkepov, S.; Folgheraiter, M.; Abakov, A.; Umurzakov, T. Development of a Reduced-Degree-of-Freedom (DOF) Bipedal Robot with Elastic Ankles. Robotics 2024, 13, 172. https://doi.org/10.3390/robotics13120172

[7] S. Yessirkepov, T. Umurzakov, R. Shaimerdenov and M. Folgheraiter, "An Elastic Shoulder Joint for Humanoid Robotics Application," 2023 9th International Conference on Automation, Robotics and Applications (ICARA), Abu Dhabi, United Arab Emirates, 2023, pp. 117-122, doi: 10.1109/ICARA56516.2023.10125796.

[8] M. Folgheraiter, S. Yessirkepov, T. Umurzakov and R. Korabay, "NU-Biped-4 a Lightweight and Low-Power Consumption Full-Size Bipedal Robot," 2023 9th International Conference on Automation, Robotics and Applications (ICARA), Abu Dhabi, United Arab Emirates, 2023, pp. 38-43, doi: 10.1109/ICARA56516.2023.10126025.

Chapter 2

Background Research

2.1 Elastic Actuator Design Principles

Servomotors are essential components in most robotic and automated systems. They typically consist of an electric motor that converts electrical power into mechanical power, a gearbox that reduces angular velocity and increases torque, an encoder or tachometer to measure position or velocity, and a feedback loop control system that tracks a reference trajectory in terms of position, velocity, or torque. To achieve high precision, servomotor components are generally designed to have low compliance. However, in specific applications, such as the mechanical architecture of humanoid robots, elastic elements are integrated to reduce the overall stiffness and enhance safety during human-robot interactions [1].

Among various motor technologies, Brushless Direct Current (BLDC) motors are widely used in most applications. BLDC motors are favored owing to their high efficiency, reliability, and precise control capabilities. In applications requiring high torque at low speeds, the motor output shaft is often coupled with a gearbox. Gearboxes are typically used to reduce angular velocity and increase torque. The primary types of gearboxes include planetary gearboxes and harmonic drives. Direct-drive systems, which operate without a gearbox, constitute a separate category.

Planetary gearboxes are compact and provide high torque density, making them ideal for space-constrained applications, although they may introduce backlash. Harmonic drives, which are usually more expensive, have high precision and small backlash [2], and are often used in robotics where accuracy is important. On the other hand, direct-drive systems connect the motor directly to the load, offering simplicity, allowing precise torque control [3], and eliminating backlash but at the cost of requiring more powerful motors for the same torque output.

To achieve inherent compliance in the actuators, elastic elements are introduced either in series or parallel to the actuator mechanism. These elements can be mechanical, such as torsional springs, or can incorporate advanced technologies such as auxiliary motors, and pneumatic or hydraulic systems. Compliance allows the actuator to absorb and adapt to external forces, thereby enhancing the safety and performance in dynamic environments.

Compliant actuators can have either fixed [15], [16], or variable stiffness [17], [18], [19], [20]. Fixed stiffness elements provide consistent compliance characteristics,

Table 2.1: Comparison of Different Types of Actuators.

Type of Actuator	**Description**	**Advantages**	**Disadvantages**	**Applications**
Without Elastic Element [3]	Actuators without any compliant element, relying solely on motor and gearbox	- High precision and control - Simplicity in design - Direct transmission of forces	- No absorption of external forces - Potential for high impact and wear - Stiffer motion	CNC machines Industrial robots High-precision tools
Elastic Element with Fixed Stiffness [4], [5], [6]	Actuators with a constant stiffness elastic element (for example, a torsional spring)	- Improved safety and adaptability - Consistent compliance characteristics - Energy efficiency	- Limited adaptability to varying loads - Fixed dynamic response	Exoskeletons Humanoid robots Prosthetics
Elastic Element with Variable Stiffness [7], [8], [9], [10], [11], [12], [13], [14]	Actuators with adjustable stiffness (for example, a pneumatic actuator)	- Adjustable compliance - High versatility - Adaptability to varying tasks and loads	- Increased complexity - Potential for higher maintenance - More complex control systems	Soft robotics Assistive devices Adaptive robotic arms

whereas variable stiffness elements can adjust their properties in response to control inputs. Pneumatic actuators use compressed air to generate force. They can vary the stiffness by adjusting the chamber pressure acting as torsional springs with adjustable characteristics.

Integrating BLDC motors with gearboxes and elastic elements has been extensively studied to enhance actuator performance. For instance, Zhu et al. explored the benefits of using elastic actuators in lower limb exoskeletons, particularly in compensating contact stiffness while walking [21]. Extensive research to increase the torque capabilities of robotic actuators with planetary gearboxes was conducted by Crispel et al. [22] who proposed a novel planetary gearbox design. Terfurth and Parspour integrated a BLDC motor and a gearbox into a single structure [23], thus reducing the inertia of the system and backlash. Backdrivability, a feature that is especially important for collaborative robots, exoskeletons, and other robotic applications, was studied by Matsuki et al. [24].

The concept of actuator compliance has gained significant attention. Series elastic actuators (SEAs) [15], which place an elastic element in series with the actuator,

were pioneered by Pratt and Williamson in 1995 and have been shown to improve force control and energy efficiency. Parallel elastic actuators (PEAs) add elasticity in parallel to the actuator mechanism, providing similar benefits for different mechanical configurations. They were studies later, but currently they have quite an extensive implementation [4], [5], [6], [16].

Pneumatic actuators are particularly interesting owing to their capacity to dynamically adjust the stiffness based on application requirements. Shen and Goldfarb [25] demonstrated how pneumatic actuators can be tuned for different compliance levels by adjusting the air pressure, making them versatile for various applications. This variable stiffness capability has been further explored in studies focusing on enhancing the adaptability and safety of robotic systems [26], [27], and [28]. The mathematical models of pneumatic actuators are detailed in [29] and [30]. These models are based on the torque generated by pressure difference between two chambers, usually one pressurized and one non-pressurized chamber.

Implementing variable stiffness for an elastic element is a non-trivial task from the point of view of design and constructability. Nevertheless, researchers have managed to create variable stiffness systems from almost all types of elastic elements, including electromagnetic springs [7], torsional springs [8], [9], [10], [11], pneumatic actuators [12], [13], and McKibben PAM (pneumatic artificial muscles) [14]. A summary of the main types of actuators discussed above is provided in Table 2.1, which presents a comparison of their advantages, disadvantages, and application areas.

Although the integration of compliant elements offers various benefits their inherent nonlinear characteristics can negatively affect the performance of classical feedback control systems. One possible solution is to use model-based control techniques [31], [32].

2.2 ANN Modeling Approaches

Different control architectures, such as computed torque control [33], [34], nonlinear control [35], model predictive control [36], adaptive control [37], sliding mode control [38], feedback linearization control [39], and Kalman filtering require the availability of an accurate and up-to-date model of the nonlinear dynamic process. In many cases, this model needs to be adapted online while the process is operating, necessitating the integration of a module capable of collecting the state of the process, training and validating a suitable representation in real time.

Therefore it is essential to first identify a model of the system. As system complexity increases, analytical approaches are often highly challenging. This difficulty is a key driving force behind adopting artificial neural networks (ANNs) for model identification. The first attempts to use ANNs were made in the 1990s [40], [41], [42], [43], [44],

[45]. However, their performance was limited at the time, primarily due to the low computational power of the available hardware. Later, researchers discovered that ANNs are particularly efficient for modeling nonlinear systems, as such systems are difficult to describe analytically. The ability of ANNs to learn complex relationships between inputs and outputs allows them to adapt to a system's nonlinearities, making them practical tools for capturing the dynamic behavior of such systems [46], [47]. Modern approaches to using ANNs for modeling include implementing deep neural network architecture [48], Feed-Forward Neural Network (FFNN) [49], [50], [51], and Recurrent Neural Network (RNN) [52].

Echo State Networks (ESN) are the class of recurrent neural networks that are particularly useful for time series prediction and system identification. An ESN consists of a sparsely connected recurrent reservoir of neurons with fixed random weights, with only the output weights trained. This significantly reduces the computational complexity of the adapting step compared with traditional RNNs [53], [54]. ESNs have shown promising results in various fields, including robotics [55], [56] and signal processing [57]. Their simplicity and efficiency make them particularly attractive for real-time applications and systems with limited computational resources [58].

In cases where a system can be approximated as a linear dynamic component followed by a static nonlinear component, a Wiener model provides a convenient structure [59–63].

Hybrid pneumatic-electric actuators have been explored in several studies to combine the advantages of both actuation methods, and to optimize performance without compromising safety. Researchers have developed different designs to achieve this goal, for example, by integrating low-frequency pneumatic muscles with a high-frequency electrical actuator to improve the control bandwidth and reduce impedance [64], or by combining pneumatic cylinders with direct-drive electric motors for enhanced torque generation and precision [65]. Additional innovations include novel structural integrations, such as fusion hybrid linear actuators, which improve force output and disturbance rejection [66].

This work presents a neural network-based model for a variable stiffness system consisting of a BLDC motor, a planetary gearbox, and a SEA-PEA pneumatic actuator. The novelty of this study lies in the innovative use of backlash, traditionally regarded as a disadvantage in robotics applications, for the development of a compliant mechanism with variable stiffness. Furthermore, a larger backlash in the planetary gearbox can enhance the backdrivability by reducing the gear tooth meshing forces. Additionally, this actuator enables the storage of a portion of the energy of the system within the pneumatic valve, which can then be recovered during the walking gait [67]. As a result, a lower-cost setup can deliver performance comparable to that of an actuator with more expensive components.

Model-based control methods have been widely studied and applied in robotics due to their ability to leverage system dynamics for improved performance. These methods rely on mathematical models of the system to design control strategies, enabling precise and robust control even in the presence of uncertainties and disturbances. This section reviews the key model-based control approaches, their applications in robotics, and the growing role of data-driven techniques, such as artificial neural networks (ANNs), in enhancing control performance.

2.3 Adaptive Control Strategies

Model-based control methods rely on explicit mathematical representations of system dynamics to compute control inputs. These approaches are particularly effective for complex and nonlinear systems such as humanoid robots, aerial drones, and industrial manipulators. The foundation of model-based control can be traced to 1960, when R. E. Kalman published his seminal paper "A new approach to linear filtering and prediction problems" [68], which introduced the principles of optimal estimation. Since then, model-based control theory has evolved substantially, giving rise to several prominent techniques widely adopted in robotics.

Among them, Model Predictive Control (MPC) is an optimization-based strategy that predicts future system behavior over a finite horizon and computes control actions that minimize a cost function subject to system constraints. MPC is particularly effective for handling multi-input multi-output systems and constraints such as joint limits and actuator saturation in humanoid robots. Its predictive nature allows anticipatory control, making it a preferred choice for complex robotic applications [69], [70], and [71].

Computed Torque Control (CTC) is another well-established approach that employs the inverse dynamics of the robot to linearize the system. By canceling out nonlinear terms, the control problem is reduced to a linear form that can be addressed with conventional controllers such as PID. CTC has been successfully applied to humanoid robots for walking, squatting, and dynamic balancing tasks [72], [73].

Adaptive control techniques further extend this paradigm by continuously adjusting controller parameters in real time to accommodate uncertainties, parameter variations, or unmodeled dynamics. These methods are particularly suitable for humanoid robots, which often operate in environments with unpredictable ground interactions or payload changes. Adaptive control enhances robustness and has been effectively used to improve locomotion and manipulation performance [74], [75].

Another important development is Sliding Mode Control (SMC), which provides robustness against model uncertainties and external disturbances by driving system trajectories onto a predefined sliding surface. Once on this surface, the dynamics become insensitive to parameter variations. Despite its robustness, SMC may introduce

chattering—a high-frequency oscillation in the control signal—which has motivated the development of smoother sliding mode variants for humanoid applications [76], [77].

Feedback linearization represents yet another direction, where nonlinear system dynamics are transformed into an equivalent linear system through a suitable change of variables and control law. This approach simplifies controller design and has been applied to humanoid robots for gait generation, trajectory tracking, and stabilization [78].

Model-based control has been extensively employed to address key challenges in humanoid robotics. In the context of walking and locomotion, controllers such as CTC and MPC are used to generate stable and dynamically consistent walking patterns by computing joint torques that follow desired trajectories [79], [80]. For balancing and stabilization, adaptive control and SMC have demonstrated improved robustness and stability under external perturbations [81], [82], and [83]. In manipulation and interaction tasks, feedback linearization and MPC have enabled coordinated control of humanoid arms and hands, allowing precise and dexterous movement [84], [85], and [86].

While these approaches offer strong theoretical guarantees, their performance depends heavily on the accuracy of the underlying model. Developing and maintaining high-fidelity dynamic models for humanoid robots remains challenging due to their complex kinematics, compliance, and unmodeled effects. Consequently, data-driven methods—particularly those based on artificial neural networks (ANNs)—have emerged as powerful tools to complement and enhance model-based control. [87]. ANNs can learn system dynamics directly from empirical data, reducing dependence on analytical modeling. Recurrent architectures such as Echo State Networks (ESNs) have proven effective in capturing the nonlinear temporal dependencies inherent in humanoid motion [88]. Neural networks can also be embedded within predictive control frameworks to approximate system models or cost functions, improving prediction accuracy and computational efficiency, and enabling real-time MPC for high-dimensional systems [89].

Combining analytical models with neural approximations leverages the complementary strengths of both paradigms: the interpretability and stability of model-based design and the adaptability of data-driven learning. In such hybrid schemes, ANNs are often used to compensate for model uncertainties or unmodeled dynamics, while the model-based controller provides structure and stability guarantees [90]. This integration represents a promising direction toward achieving robust, adaptive, and computationally efficient control of humanoid robots.

2.4 Challenges and Future Directions

Despite the significant progress in both model-based control and data-driven methodologies, several challenges continue to constrain their effectiveness in humanoid robotics.

Model Accuracy: Achieving high-fidelity dynamic models for humanoid robots remains a major obstacle. The inherent nonlinearities, high degrees of freedom, joint compliance, and unmodeled disturbances introduce uncertainties that limit the accuracy of analytical models and degrade controller performance.

Computational Complexity: Advanced model-based control schemes, such as Model Predictive Control (MPC) or inverse dynamics optimization, demand substantial computational resources. This complexity restricts their implementation in real time, particularly for high-DOF humanoids where rapid control updates are required for stable locomotion and interaction.

Generalization and Data Efficiency: Although data-driven methods, including neural networks and reinforcement learning, have shown promise in capturing nonlinear dynamics, they often rely on large, task-specific datasets. Consequently, their ability to generalize across varying conditions, payloads, or environments remains limited, leading to inconsistent real-world performance.

Future Directions: Addressing these challenges requires a synergistic integration of model-based and learning-based paradigms. Future research should focus on developing hybrid control architectures that combine the interpretability and stability of analytical models with the adaptability of neural networks. Efforts toward efficient system identification, online learning, and sim-to-real transfer are particularly promising. Furthermore, implementing these frameworks on physical humanoid prototypes will be essential to validate robustness, energy efficiency, and scalability under real-world uncertainties.

2.5 Transition to the Thesis

The preceding sections have reviewed the fundamental design principles of elastic actuators, the use of artificial neural networks (ANNs) for dynamic modeling, and the evolution of adaptive control strategies in robotics. While significant progress has been achieved in each of these domains, the literature reveals persistent limitations when these technologies are applied to complex, compliant robotic systems such as humanoids. Conventional actuation methods—most notably harmonic drives—offer precision but lack adaptability and energy efficiency. Conversely, existing elastic actuators often struggle to provide tunable stiffness and sufficient torque capacity for full-scale robotic applications.

Similarly, although numerous studies have demonstrated the potential of ANNs

for system identification and control, most implementations remain limited to either simulation environments or simplified actuator models. The integration of neural-based models with physically compliant actuators and their subsequent deployment in humanoid robots remains largely unexplored. Furthermore, achieving stable and energy-efficient sim-to-real transfer continues to pose a major challenge for adaptive controllers operating in nonlinear, high-DOF systems.

In response to these gaps, this thesis introduces a neural-augmented modeling and control framework developed and validated across two experimental platforms. The first focuses on the design and control of a hybrid Series–Parallel Elastic Actuator (SEA–PEA) that merges pneumatic compliance with electromechanical precision, enabling dynamic stiffness modulation. The second extends these principles to a 12-degree-of-freedom humanoid robot, where an Echo State Network (ESN)-based Computed Torque Control (CTC) scheme facilitates robust trajectory tracking and reliable sim-to-real transfer. Together, these studies demonstrate how combining data-driven learning with model-based control can bridge the gap between compliant actuation and adaptive whole-body control in humanoid robotics.

Chapter 3

Description of the SEA–PEA Elastic Actuator and the Humanoid Prototype

3.1 Electromechanical and Pneumatic Description of the SEA/PEA Actuator

As shown in Fig. 3.1A, the overall architecture of the setup encompasses various components, each playing a crucial role in the system's functioning. The system is actuated by a Brushless Direct Current (BLDC) motor, Model ML 3510 by GARTT. It operates at a voltage of 24 V, has 7 pole pairs, and a speed constant of 600 rpm/V. Under its rated operating conditions, the motor generates a maximum torque of 0.3 N·m [91]. The motor is paired with a planetary gearbox with a gear ratio of 100:1 and efficiency of 77%, increasing the maximum torque to 23.1 N·m. The other components include two magnetic encoders, a pneumatic actuator, and 3D-printed mechanical parts that serve as structural elements and connectors. Additionally, the setup has a link (Fig. 1A, item 8), a load cell, a pressure sensor, a solenoid valve, an air compressor (Fig. 2), a dual motor controller (model: ODrive 3.5), a microcontroller (model: Arduino Nano), and two relays responsible for controlling the compressor and solenoid air valve.

The rigid CNC-machined aluminum holder (Fig. 1A, item 4) was used as a ground base. It has two parallel mounting plates that allow the connection of the gearbox and the pneumatic actuator mounting frames. During the assembly phase, once the mounting frames are bolted to the holder, their shafts are welded to each other to create a rigid connection from the gearbox to the pneumatic actuator and eliminate mutual displacement. The pneumatic actuator has a rotation range of 270° which exceeds the required rotation range for most joints in humanoid robots. The angular displacement of the shaft is initiated by the BLDC motor, and not by the pneumatic actuator. When the directions of the torque generated by the BLDC motor and the pneumatic element are the same, the pneumatic element facilitates rotation caused by the BLDC motor. Otherwise, when directions are opposite the pneumatic element counteracts such rotation. This behavior occurs because only one chamber of the pneumatic element is pressurized,

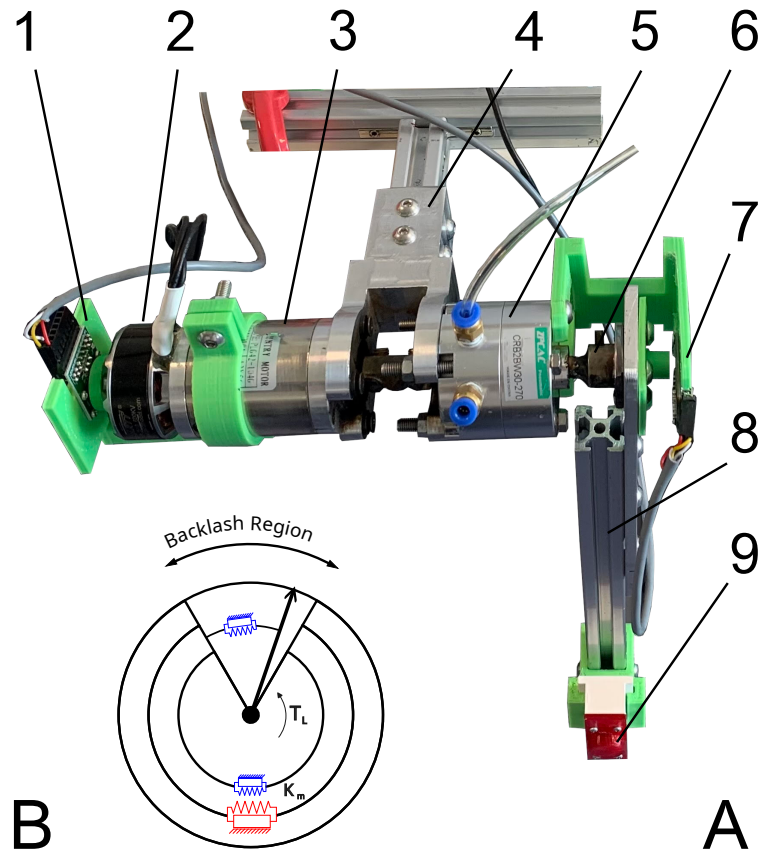

Figure 3.1: A) Overall Setup Architecture. 1-Encoder (Motor Side); 2-BLDC Motor; 3-Gearbox; 4-Aluminum Holder; 5-Pneumatic Element; 6-Metal Adapter; 7-Encoder (Pneumatic Element side); 8-Link; 9-Load Cell; B) The motor torque does not affects the system in the backlash region, but only the pneumatic actuator and external torque do.

thereby generating torque in a single direction. The BLDC motor is connected from the other side of the gearbox. To enable the application of force (torque) to the output shaft during experimental procedures, a metal adapter connects the output shaft of the pneumatic actuator to a link, which has a sensor installed at its distal end to measure the applied force.

The pneumatic component of the setup depicted in Fig. 3.2 involves pressurizing one of the chambers within the pneumatic actuator using a designated level of pressure supplied by the air compressor. The pressure sensor is responsible for measuring the pressure being generated in the system. Once the pressure reaches the desired level, a control signal is sent to the relay, activating the solenoid air valve. The valve seals the system off from the air compressor, ensuring that the pressure level remains constant.

As shown in Fig. 3.3 the gearbox and the pneumatic actuator have a common shaft because their shafts are welded. This causes the elastic element to behave both as parallel-connected and series-connected depending on whether the rotation occurs outside or within the backlash range respectively. A schematic illustration of the

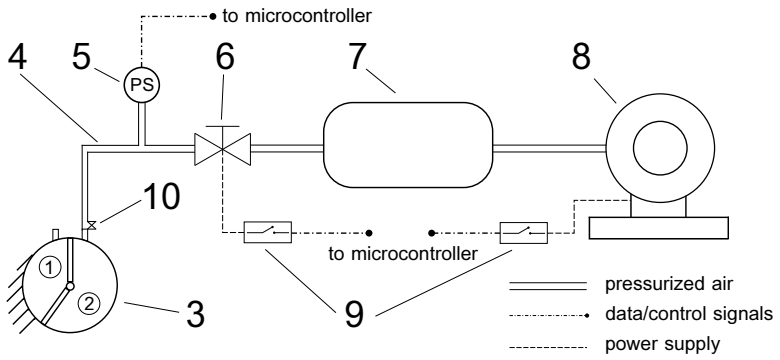

Figure 3.2: Pneumatic and Electrical Schematic Diagram. 1,2-Non-pressurised and pressurised chambers; 3-Pneumatic Element; 4-Flexible Tubes; 5-Pressure Sensor; 6-Solenoid Air Valve; 7-Reservoir; 8-Compressor; 9-Relays controlling Solenoid Air Valve and Compressor; 10-Controlled Leakage.

difference between the two modes is shown in Fig. 3.1B. In other words, within the backlash region, the system behaves as a Serial Elastic Actuator. Once it moves beyond the backlash region, it functions as a Parallel Elastic Actuator.

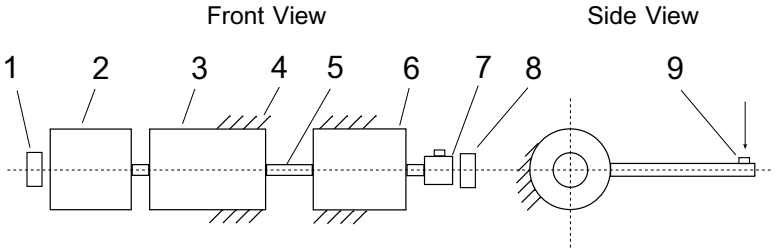

Figure 3.3: Mechanical Schematic Diagram. 1-Encoder (Motor Side); 2-BLDC Motor; 3-Gearbox; 4-Rigid connection to the ground; 5-Common Shaft; 6-Pneumatic Element; 7-Link; 8-Encoder (Pneumatic Element side); 9-Load Cell.

The setup uses an open-source ODrive motor controller [92], which is commonly employed in robotic applications. This controller effectively governs the Brushless Direct Current (BLDC) motor, offering a dependable and accurate solution that delivers the torque necessary for the experiments. Additionally, two magnetic encoders are employed to independently measure the position of the motor and the output shaft of the pneumatic actuator respectively. ODrive is connected to a desktop computer via a USB cable to establish communication.

The system utilizes a two-level software architecture (see Algorithm 1) to coordinate the execution of the experiment between high-level experiment planning and low-level hardware control. At the upper level, a Python code runs on the desktop computer. This code defines the primary sequences and parameters of the experiment, such as sampling rate, duration, pressure level, and other relevant variables. At the lower level, C code is executed on a microcontroller. This code is responsible for reading data from the pressure sensor and the load cell, transmitting this data to the desktop via a serial port

operating at 115200 baud, and controlling the relay connected to the solenoid air valve. This two-level software architecture enables seamless coordination and communication between the desktop and the hardware components.

Algorithm 1 Two-Level Software Architecture for Experimental Setup

Input: Experiment parameters (T_s, T, P)
Output: Motor angular position θ_{motor} and velocity ω_{motor}, pneumatic actuator angular position θ_{output} and velocity ω_{output}, BLDC measured current Iq_{measured}, pressure P, force F

Upper Level: Python Code on Desktop
Define primary sequence and experiment parameters:
- Sampling time T_s, s
- Duration T, s
- Pressure level P, bar

Send experiment parameters to the microcontroller via serial port
Lower Level: C Code on Microcontroller
Initialize serial communication
Set up sensors and relay control for solenoid air valve
while the experiment is running **do**
    Read position
    Read data from the pressure sensor
    Read data from the load cell
    Transmit sensor data to the desktop via serial port
    Control compressor and solenoid air valve if required
end while
Finalization
Ensure all data is transmitted and saved
Safely shut down sensors and relay

To define the actuator's physical characteristics and collect data for training neural networks the following parameters are measured from the sensors of the setup during the experiments: position θ_{motor} and velocity ω_{motor} of the BLDC motor, position θ_{output}, and velocity ω_{output} of the pneumatic actuator output shaft, measured current of the BLDC motor Iq_{measured}, pressure P obtained from a pressurized chamber of the pneumatic actuator, and force F, measured from a load cell located on the link at a distance l equal to 30 cm from the output shaft axis.

3.2 Humanoid Robot Description

3.2.1 Kinematic Structure

The bipedal robot developed in our laboratory, referred to as NU-Biped-4.7 presented in Fig. 3.4-a, features six revolute joints in each leg: three at the hip, one at the knee, and two at the ankle. Despite its height of approximately 1.0 m, the robot remains lightweight, with a total mass of only 17.1 kg (excluding the power supply). This was achieved through the use of aluminum and Polylactic Acid (PLA) components, along with custom high power-to-weight-ratio servomotors.

The hip joint axes were intentionally designed to intersect at a single point in order to simplify the inverse kinematics (IK) computation. In contrast, the ankle joint axes were arranged with a small offset L_5 as it is shown in Fig. 3.4-b and in Table 3.1, rather than intersecting, to eliminate the need for mechanical transmission systems and thereby simplify the overall design and assembly of the robot.

3.2.2 Electromechanical Description

The robot employs two distinct types of actuators, both based on GARTT brushless DC (BLDC) motors, but equipped with different transmission mechanisms. The first type utilizes GARTT ML3510 600 KV motors coupled with planetary gearboxes having reduction ratios of 100:1 and 50:1, while the second type employs GARTT ML5008 400 KV motors integrated with harmonic drive reducers featuring a reduction ratio of 80:1. Both actuator configurations were designed and assembled in the laboratory to evaluate their relative performance in terms of torque output, compactness, mechanical stiffness, and controllability.

The earlier actuator version, which employed low-cost planetary gearboxes, demonstrated satisfactory reliability and mechanical robustness during preliminary experiments. However, its primary drawback was the presence of significant backlash within the gearbox stages. This mechanical play introduced delays and oscillations in joint response, thereby complicating precise control of the humanoid prototype, especially during dynamic motions such as squatting or single-leg balancing. The

Table 3.1: Link dimensions of the NU-Biped-4 prototype

Link	**Distance [m]**
L_1	0.243
L_2	0.150
L_3	0.278
L_4	0.260
L_5	0.104
L_6	0.134

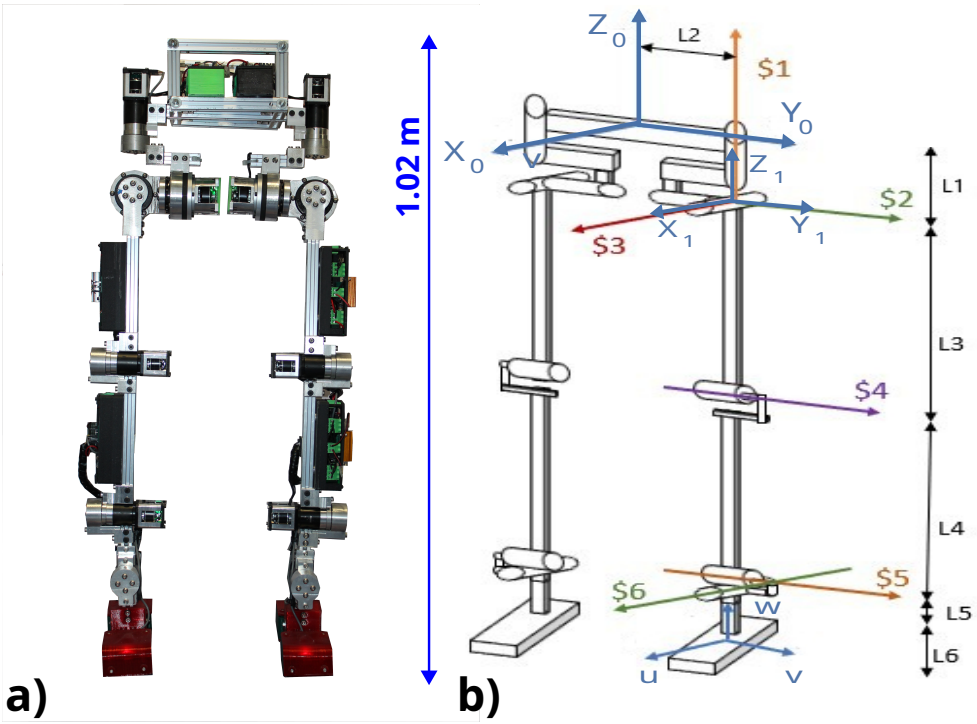

Figure 3.4: a) NU-Biped-4.7 robot prototype, 1.0 m height, weight 17.1 kg b) Kinematic model of the bipedal robot including the reference frames and screw axes.

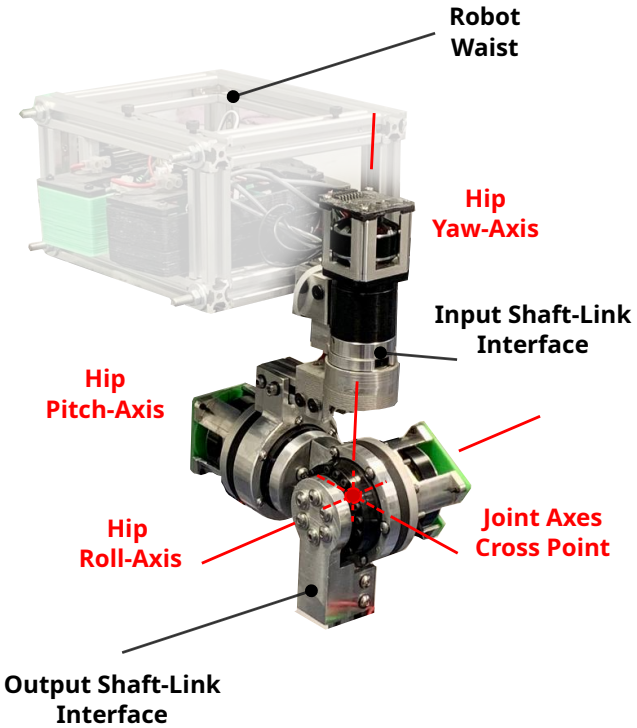

Figure 3.5: Mechanical design of the hip joints. The three joint axes intersect at a single point to simplify the solution of the IK. Each servomotor features two link interfaces: the first one connects the gearbox's body to the input link, and the second one connects the servomotor shaft to the output link.

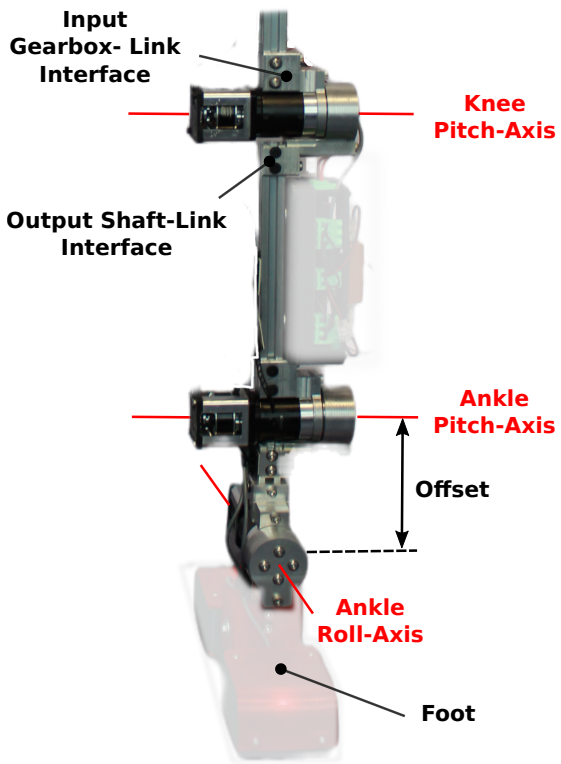

Figure 3.6: Mechanical design of the knee and ankle joints.

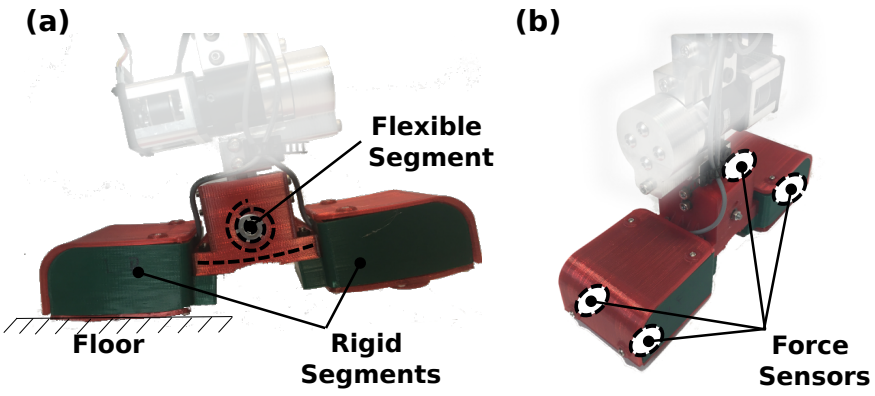

Figure 3.7: Foot Design: a) Two rigid segments made out of PLA material are connected with an elastic central segment made out of TPU material. b). Two force sensors, along with the necessary electronics, are integrated under each rigid segment.

internal structure of this actuator is illustrated in Fig. 3.8.

Each actuator of this type is based on a GARTT ML3510–600 KV BLDC motor coupled with planetary gearboxes of either $50{:}1$ or $100{:}1$ reduction ratio. The relationship between the motor's velocity constant K_V (expressed in rpm/V) and its torque constant K_t (in N·m/A) is given by

$$K_t = \frac{60}{2\pi K_V}, \tag{3.1}$$

which follows from the equivalence between the back-EMF constant K_e and the torque constant K_t, with unit conversion between rotational speed in rpm and angular velocity in rad/s.

Substituting $K_V = 600$ rpm/V into (3.1) yields

$$K_t = \frac{60}{2\pi \times 600} \approx 0.0159 \text{ N·m/A}. \tag{3.2}$$

Given the motor current limit $I = 25$ A, the nominal motor-side torque becomes

$$\tau_m = K_t I = 0.0159 \times 25 \approx 0.398 \text{ N·m}. \tag{3.3}$$

Assuming a gearbox efficiency of $\eta = 0.75$, the continuous output torques are estimated as

$$\tau_{\text{out,50:1}} = \tau_m \, N \, \eta = 0.398 \times 50 \times 0.75 \approx 14.9 \text{ N·m}, \tag{3.4}$$

$$\tau_{\text{out,100:1}} = \tau_m \, N \, \eta = 0.398 \times 100 \times 0.75 \approx 29.8 \text{ N·m}. \tag{3.5}$$

Hence, the experimentally observed value of approximately 30 N·m for the $100{:}1$ version agrees well with the theoretical estimate (3.5), validating the consistency of the torque model derived from (3.1).

To overcome the limitations associated with backlash, a new actuator design was developed employing a harmonic drive transmission. Harmonic drives are well known for their zero-backlash characteristics, high reduction ratios, and compact design, making them highly suitable for precise motion control in humanoid robots. This new actuator type, shown in Fig. 3.9, is currently installed in the hip pitch and hip roll joints, where high torque accuracy and stiffness are most critical for maintaining balance and stability during motion.

This actuator integrates a GARTT ML5008–400 KV motor with an $80{:}1$ harmonic reducer. Using the same relation (3.1) with $K_V = 400$ rpm/V gives

$$K_t = \frac{60}{2\pi \times 400} \approx 0.0239 \text{ N·m/A}. \tag{3.6}$$

With the current limit of 25 A, the motor-side torque is

$$\tau_m = 0.0239 \times 25 \approx 0.597 \text{ N·m}. \tag{3.7}$$

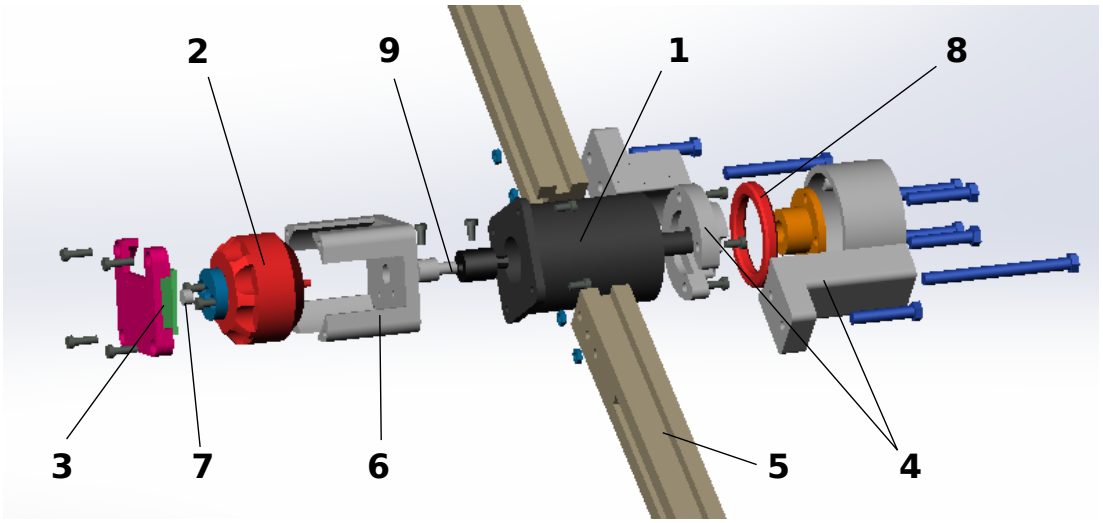

Figure 3.8: Exploded view of the servomotor where the main components are: 1) A low cost planetary gearbox. 2) A GARTT ML3510 600 KV BLDC motor. 3) A 14 bit magnetic encoder. 4) Link interfaces. 5) 20 by 20 aluminum link. 6) BLDC motor mounting with integrated heat sink. 7) Magnet mounting. 8) Ball bearing. 9) Motor-gearbox shaft interface.

Assuming a transmission efficiency of $\eta = 0.75$, the continuous output torque is estimated as

$$\tau_{\text{out,80:1}} = \tau_m \, N \, \eta = 0.597 \times 80 \times 0.75 \approx 35.8 \,\text{N·m}. \tag{3.8}$$

This higher torque output (3.8), combined with the intrinsic zero-backlash performance of the harmonic drive, significantly enhances control precision and mechanical stiffness compared with the previous planetary gearbox version, making it an optimal choice for the robot's hip joints.

Figure 3.9: New Actuator with GARTT ML5008 400 KV BLDC and Harmonic Drive.

All servomotors and onboard sensors are interconnected through a CAN bus network utilizing an MCP2512 controller as presented in Fig. 3.10, which links them to the central computational unit—a Jetson Nano. The system employs a dual-voltage power architecture, where the sensors, microcontrollers, and computing modules operate at 5 V, while the motor controllers are supplied with 24 V to ensure adequate power delivery for actuation.

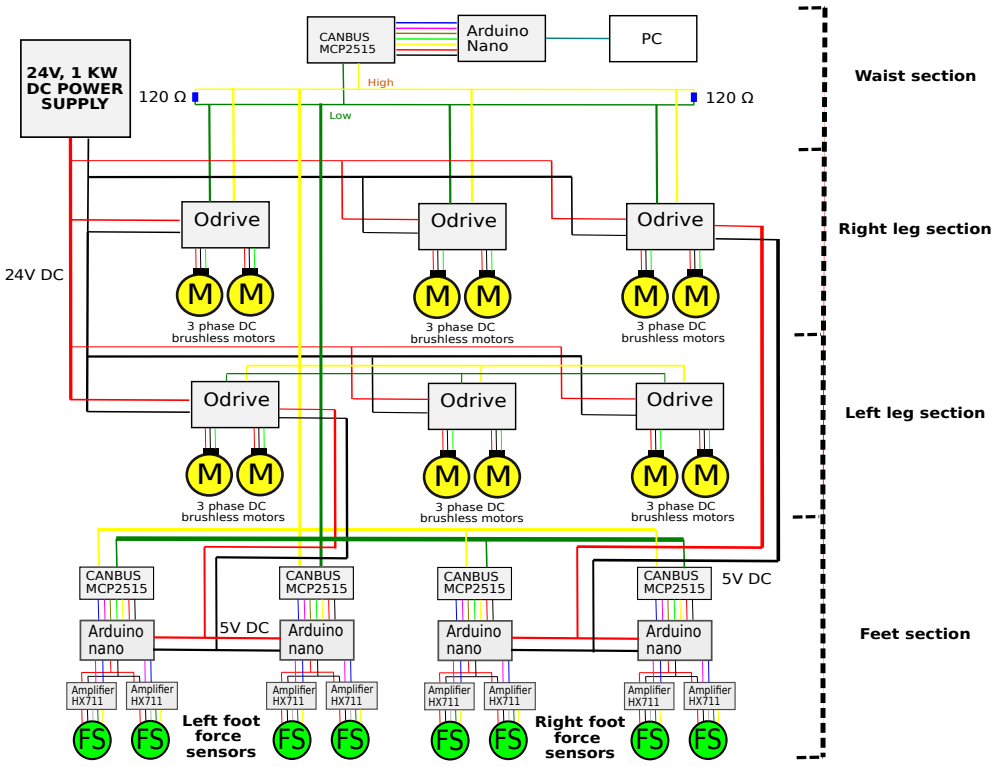

Figure 3.10: Overall electrical circuit of the bipedal robot NU-Biped-4.7.

An exact model of the NU 4.7 biped was created in FreeCAD and then imported to CoppelliaSim simulation software with the purpose of further export to URDF (Unified Robot Description Format) which is an XML-based file format used to describe the structure, kinematics, dynamics, and visual representation of a robot. The model render in PyBullet User Interface is shown in Fig. 3.11.

The humanoid robot has 12 DOFs with 6 DOFs for each leg. The experiments are conducted using squat motion that involves three joints of each leg namely hip pitch, knee, and ankle pitch. This type of motion was selected mainly because of its simplicity and stability. During this motion CoM (Center of Mass) moves along Z axis down and up and its vertical projection is located in the geometrical center of the support polygon. It can be said that the motion space of six joints is a subspace of a motion space of all twelve joints. This partial involvement of the joints still can help to achieve a planned goal of creating a Computed Torque Control (CTC) algorithm that is expected to have better efficiency compared to standard PID algorithm in term of MSE (Mean

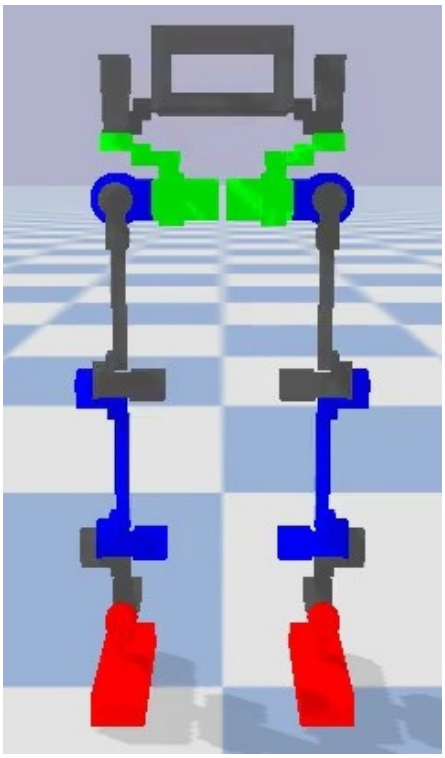

Figure 3.11: PyBullet Model of the NU 4.7.

Squared Error) between reference and actual positions, and energy consumed to follow the reference trajectory.

3.2.3 CAN Bus Data Exchange Protocol

Achieving the highest feasible sampling rate is of critical importance during experiments with the real prototype, as it directly impacts control accuracy and system stability. The total cycle consists of data processing in Python, data exchange via Serial and sensing plus execution done on an Arduino side. Current total cycle execution time equals to $12.2\,\mathrm{ms}$ which is equivalent of $82\,\mathrm{Hz}$. This is close to a planned sampling rate of $100\,\mathrm{Hz}$.

Data exchange, sensing and execution time per cycle equals to $12.0\,\mathrm{ms}$. This is split between data exhange via Serial which is $4.2\,\mathrm{ms}$, and sensing plus execution on an Arduino side which is $7.8\,\mathrm{ms}$.

The measured timings are as follows: sending commands - $5.7\,\mathrm{ms}$, reading actual data from CAN bus - $5.1\,\mathrm{ms}$, total loop time - $11.8\,\mathrm{ms}$. Deducting two values from the last one we have $1.0\,\mathrm{ms}$ for data exchange. On contrary the theoretical maximum is $4.2\,\mathrm{ms}$ at $153600\,\mathrm{baud}$.

Sent data: 12 reference positions. Received data: 12 actual positions and 8 force sensor readings.

Sending `float32` (which takes 4 bytes) is not as efficient as sending `int16`. The maximum range of `int16` is $(-32768, 32767)$.

A precision of 0.01 turns, in the worst case (for an actuator with a gear ratio of $50{:}1$), equals to:

$$\alpha_{\text{joint}} = \frac{0.01 \times 360}{50} = 0.072^{\circ} \tag{3.9}$$

The corresponding linear displacement for the leg length of $800\,\text{mm}$ is:

$$\delta_{\text{joint}} = \frac{0.01}{80} \cdot 2 \cdot \pi \cdot 800 = 0.62\,\text{mm}. \tag{3.10}$$

Both cases confirm that a precision of 0.01 turns is more than sufficient for controlling the humanoid robot. Therefore, the communication protocol uses position values in motor turns multiplied by 100 and cast to `int16`. Upon reception, these values are divided by 100 and used for further processing.

The total data exchange load shall consist of 12 reference position, 12 actual positions, 12 actual velocities, 12 measured current (Iq) values, 8 force sensor readings, and 9 IMU values representing angular position, angular velocity, and linear acceleration of the robot base link (torso) along the X, Y, and Z axes.

Chapter 4

Methodology

4.1 SEA/PEA Actuator Modeling Methodology

The proposed methodology integrates black-box and grey-box modeling techniques to represent the system dynamics accurately. The black-box approach is used to model systems with complex nonlinear behaviors that are difficult to describe mathematically. In contrast, the grey-box approach applies when the system dynamics can be expressed with equations, but parameter values require estimation. Grey-box modeling estimates initial parameter values using data sheets, CAD calculations, material properties, experiment results, or prior knowledge.

The research process for developing a neural network-based model for the electromechanical system with a pneumatic variable stiffness element includes the following stages: 1) Experimental Procedure and Parameters Identification, 2) Selection of the Model Structure, 3) Model Training, and 4) Model Validation.

The first stage identifies the system parameters, such as backlash, friction forces, and gravity effects. In addition, the angular position and velocity, applied pressure, applied external torque, and measured current of the BLDC motor are collected for ANN training. In the second stage, five different neural network architectures - namely Long Short-Term Memory Recurrent Neural Network (LSTM-RNN), Gated Recurrent Unit Recurrent Neural Network (GRU), Feed-Forward Neural Network(FFNN), Nonlinear Exogenous Model (NNX), and Echo State Network (ESN) - will be considered as model structures. Next, the models are trained and their performance is compared by Mean Square Error (MSE) and time complexity. The model with the smallest MSE and lower computational complexity is preferable. Finally, the neural networks with satisfactory performance metrics are validated and used as the dynamic system model. The process is iterative, allowing revisiting of earlier stages if needed (see Fig. 4.1).

4.1.1 Defining the Dynamic Model of the System

The electrical model of the BLDC motor is based on the voltage and current relationships in the stator windings.

The electrical dynamics of a BLDC motor in a three-phase model can be described by the following set of equations:

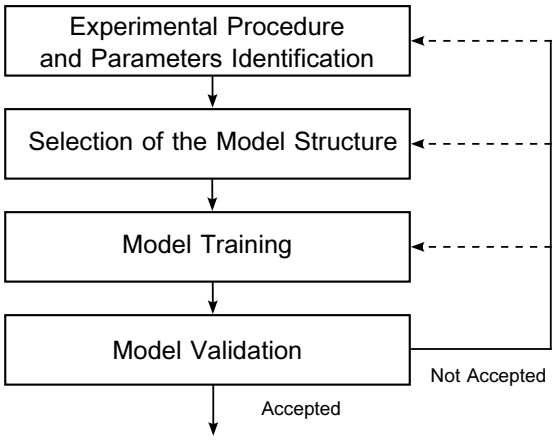

Figure 4.1: Model Development Stages.

$$V_a = L\frac{di_a}{dt} + Ri_a + e_a, \tag{4.1}$$

$$V_b = L\frac{di_b}{dt} + Ri_b + e_b, \tag{4.2}$$

$$V_c = L\frac{di_c}{dt} + Ri_c + e_c; \tag{4.3}$$

where:

- V_a, V_b, V_c are the phase voltages (in volts, V);
- L is the inductance of the winding (in Henries, H);
- R is the resistance of the winding (in ohms, Ω);
- i_a, i_b, i_c are the phase currents through the winding (in amperes, A);
- e_a, e_b, e_c are the back electromotive forces (EMFs) in each phase (in volts, V).

The back electromotive force (EMF) for each phase is proportional to the rotor angular velocity:

$$e_a = K_e f(\theta_e)\omega_m(t) \tag{4.4}$$

$$e_b = K_e f(\theta_e - 120^\circ)\omega_m(t) \tag{4.5}$$

$$e_c = K_e f(\theta_e + 120^\circ)\omega_m(t) \tag{4.6}$$

where:

- K_e is the back EMF constant (in volts per radian per second, $\mathrm{V\,rad^{-1}\,s^{-1}}$);
- $\omega_m(t)$ is the angular velocity of the motor rotor (in radians per second, $\mathrm{rad\,s^{-1}}$);
- $f(\theta_e)$ represents the back-EMF waveform.

The torque generated by the BLDC motor is given by:

$$T_e(t) = K_e \left(f(\theta_e) i_a + f(\theta_e - 120^\circ) i_b + f(\theta_e + 120^\circ) i_c \right), \tag{4.7}$$

where:

- i_a, i_b, i_c are the phase currents.

We define:

$$I(t) = f(\theta_e) i_a + f(\theta_e - 120^\circ) i_b + f(\theta_e + 120^\circ) i_c, \tag{4.8}$$

so that the motor torque equation becomes:

$$T_e(t) = K_e I(t). \tag{4.9}$$

The total torque acting on the mechanical system consists of several components: the inertial term, the viscous damping term, the elastic force term, and the load torque. The following equation describes the relationship between these components:

$$T_m(t) = J_{\text{total}} \frac{d\omega_m(t)}{dt} + B_{\text{total}} \omega_m(t) + K_{\text{total}} \theta(t) + T_L(t), \tag{4.10}$$

where:

- J_{total} is the total moment of inertia;
- B_{total} is the total viscous friction coefficient;
- K_{total} is the total stiffness coefficient;
- $\theta(t)$ is the angular displacement;
- $T_L(t)$ is the total load torque (including the torque required to move the robot joint), seen by the motor.

In turn, the total load torque applied to the motor consists of the external load torque and the torque generated by the pneumatic actuator divided by gear ratio:

$$T_L(t) = \frac{T_{\text{load}}}{N} - \frac{K_p (P_{\text{in}} - P_{\text{out}})}{N} \tag{4.11}$$

where:

- K_p is a pressure-to-torque conversion factor, which represents the effectiveness of pneumatic pressure in generating torque;

- P_{in} and P_{out} are are the input and output pressures in the actuator chambers;
- N is the gear ratio between the motor and the joint output shaft;
- T_{load} is the external load torque applied at the joint.

By combining the motor torque Equation (4.9) with the mechanical system torque Equation (4.10), we obtain the electromechanical model of the servomotor. To derive a unified equation, we set $T_e(t) = T_m(t)$, thereby equating the torque generated by the motor with the torque applied to the mechanical system. The resulting equation is as follows:

$$\begin{aligned} J_{\text{total}}\frac{d\omega_m}{dt} + B_{\text{total}}\omega_m + K_{\text{total}}\theta_m = \\ K_e I(t) + \frac{K_p(P_{\text{in}} - P_{\text{out}})}{N} - \frac{T_{\text{load}}}{N}, \end{aligned} \tag{4.12}$$

In the servomotor system, the total inertia, friction and stiffness coefficients are determined by combining the individual contributions from the motor, gearbox, load, and other components. The following equations express the total values for each parameter:

$$J_{\text{total}} = J_m + \frac{J_{\text{gb}}}{N^2} + \frac{J_{\text{load}}}{N^2} + \frac{J_{\text{pa}}}{N^2}, \tag{4.13}$$

$$B_{\text{total}} = B_m + \frac{B_{\text{gb}}}{N^2} + \frac{B_{\text{load}}}{N^2} + \frac{B_{\text{pa}}}{N^2}, \tag{4.14}$$

$$K_{\text{total}} = \frac{K_{\text{load}}}{N^2} + \frac{K_{\text{pa}}}{N^2}, \tag{4.15}$$

where:

- J_m and B_m, are the motor's moment of inertia and friction coefficient.
- J_{gb} and B_{gb} represent the contributions from the gearbox.
- J_{load}, B_{load}, and K_{load} correspond to the load's moment of inertia, friction coefficient, and stiffness, respectively.
- J_{pa}, B_{pa}, and K_{pa} denote the contributions from the pneumatic actuator, if applicable.
- N is the gear ratio.

Taking the Laplace transform of the mechanical equation, and assuming zero initial conditions:

$$\begin{aligned} J_{\text{total}} s\Omega_m(s) + B_{\text{total}}\Omega_m(s) + K_{\text{total}}\Theta_m(s) = \\ K_e I_q(s) + \frac{K_p(P_{\text{in}}(s) - P_{\text{out}}(s))}{N} - \frac{T_{\text{load}}(s)}{N} \end{aligned} \tag{4.16}$$

Rearranging the equation to solve for $\Omega_m(s)$:

$$\Omega_m(s) = \frac{K_e I_q(s) + \frac{K_p(P_{\text{in}}(s) - P_{\text{out}}(s))}{N} - \frac{T_{\text{load}}(s)}{N} - K_{\text{total}}\Theta_m(s)}{J_{\text{total}} s + B_{\text{total}}}. \tag{4.17}$$

To focus on the transfer function from the input current $I_q(s)$ to the output angular velocity $\Omega_m(s)$, we separate the effects of $I_q(s)$ and the load torque $T_{\text{load}}(s)$:

$$\begin{aligned} \Omega_m(s) = \frac{K_e I_q(s)}{J_{\text{total}} s + B_{\text{total}}} + \frac{\frac{K_p(P_{\text{in}}(s) - P_{\text{out}}(s))}{N}}{J_{\text{total}} s + B_{\text{total}}} - \\ \frac{\frac{T_{\text{load}}(s)}{N}}{J_{\text{total}} s + B_{\text{total}}} - \frac{K_{\text{total}}\Theta_m(s)}{J_{\text{total}} s + B_{\text{total}}}. \end{aligned} \tag{4.18}$$

To derive the transfer functions in Eqs. (4.19) and (4.20), we consider the linear system response to each input independently. Specifically, Eq. (4.19) is obtained from Eq. (4.18) by setting the load torque $T_{\text{load}}(s)$, pneumatic pressure difference $P_{\text{in}}(s) - P_{\text{out}}(s)$, and angular position $\Theta_m(s)$ to zero. This isolates the influence of the input current $I_q(s)$ on the output velocity $\Omega_m(s)$. Similarly, Eq. (4.20) is derived by setting $I_q(s) = 0$, $P_{\text{in}}(s) - P_{\text{out}}(s) = 0$, and $\Theta_m(s) = 0$, thus isolating the effect of the load torque. This separation of effects allows each transfer function to characterize how the system responds to a specific input. Defining $G(s)$ and $H(s)$ enables targeted analysis of control input and disturbance rejection, which is essential for controller design, tuning, and stability evaluation.

The transfer function $G(s)$ from the input current $I_q(s)$ to the output angular velocity $\Omega_m(s)$ is:

$$G(s) = \frac{\Omega_m(s)}{I_q(s)} = \frac{K_e}{J_{\text{total}} s + B_{\text{total}}}. \tag{4.19}$$

The transfer function $H(s)$ from the load torque $T_{\text{load}}(s)$ to the output angular velocity $\Omega_m(s)$ is:

$$H(s) = \frac{\Omega_m(s)}{T_{\text{load}}(s)} = -\frac{1}{N(J_{\text{total}} s + B_{\text{total}})}. \tag{4.20}$$

These transfer functions illustrate how the input current I_q and the load torque T_{load} affects the motor's angular velocity ω_m. This comprehensive model accounts for the mechanical properties of the system, including the motor, gearbox, load, and pneumatic actuator's inertia, damping, and stiffness, considering the gear ratio N.

4.1.2 Estimation of the Motor Moment of Inertia and Damping Coefficient

To accurately model the dynamic behavior of a BLDC motor, it is essential to estimate the moment of inertia (J_m) of the rotor and the damping coefficient (B_m). These parameters can be derived from the relationship between the motor's torque, angular velocity, and angular acceleration.

In the absence of external loads, the motor torque equation can be expressed as:

$$\frac{3}{2}K_e I_q = J_m \frac{d\omega_m}{dt} + B_m \omega_m. \tag{4.21}$$

The moment of inertia J and the damping coefficient B can be identified using the measured values of I_q (current), ω_m (angular velocity), and $\frac{d\omega_m}{dt}$ (angular acceleration). A simplified model that utilizes a two-axis (d-q) reference frame is used because ODrive motor controller allows to measure I_q and I_d.

4.1.2.1 Identification Steps

Measure Angular Velocity and Current Record the angular velocity (ω_m) and current (I_q) of the motor over time.

Calculate Angular Acceleration Compute the angular acceleration $\left(\frac{d\omega_m}{dt}\right)$ by differentiating the angular velocity (ω_m) with respect to time:

$$\frac{d\omega_m}{dt} = \frac{\Delta\omega_m}{\Delta t}. \tag{4.22}$$

Formulate the Linear System Equation (4.21) can be written in matrix form for multiple data points:

$$Y = Ax. \tag{4.23}$$

where Y is the vector of $\frac{3}{2}K_e I_q$ measurements, A is the matrix of angular accelerations and velocities, and x is the vector containing J_m and B_m:

$$Y = \begin{bmatrix} \frac{3}{2}K_e I_{q1} \\ \frac{3}{2}K_e I_{q2} \\ \vdots \\ \frac{3}{2}K_e I_{qN} \end{bmatrix}, \quad A = \begin{bmatrix} \frac{d\omega_1}{dt} & \omega_1 \\ \frac{d\omega_2}{dt} & \omega_2 \\ \vdots & \vdots \\ \frac{d\omega_N}{dt} & \omega_N \end{bmatrix}, \quad x = \begin{bmatrix} J_m \\ B_m \end{bmatrix}. \tag{4.24}$$

K_e is derived from the KV value provided in the technical specifications of the BLDC motor, which is equal to 600 RPM/V. The relationship between K_e and KV is follows:

$$K_e = \frac{2\pi}{60} \cdot \frac{1}{KV}. \tag{4.25}$$

Estimation of J_m and B_m Using Least Squares Solve the linear system using the least squares method to estimate J_m and B_m:

$$x = (A^T A)^{-1} A^T Y. \tag{4.26}$$

4.1.3 Neural Networks Architecture

Five different types of neural networks are implemented, trained, and validated. The following provides a brief description of each network, including its architecture and the basic formulas that define its operation.

1. Feed-Forward Neural Network (FFNN)

An FFNN comprises of an input layer, one or more hidden layers, and an output layer. Information flows in one direction—from the input to the output—without cycles or loops.

Architecture:

- **Input Layer:** Receives the input features.
- **Hidden Layers:** Neurons apply nonlinear activation functions to the weighted sum of the previous layer outputs.
- **Output Layer:** Produces the final output as the linear combination of the last hidden layer neuron outputs.

Key Formula: For a given layer l:

$$\mathbf{a}^{(l)} = \sigma(\mathbf{W}^{(l)}\mathbf{a}^{(l-1)} + \mathbf{b}^{(l)}), \tag{4.27}$$

where:

- $\mathbf{a}^{(l)}$ is the activation of layer l;
- $\mathbf{W}^{(l)}$ and $\mathbf{b}^{(l)}$ are the weights and biases of layer l;
- $\sigma(\cdot)$ is the activation function (e.g., ReLU, sigmoid).

2. Long Short-Term Memory Recurrent Neural Network (LSTM-RNN)

An LSTM is a type of Recurrent Neural Network (RNN) designed to model sequential data and capture long-range dependencies with memory cells.

Architecture:

- **Input Layer:** Receives sequential input data.
- **LSTM Layers:** Contain memory cells with gates to control information flow.
- **Output Layer:** Produces the output for each time step or the final output.

Key Formulas: For an LSTM cell at time step t:

$$\mathbf{f}_t = \sigma(\mathbf{W}_f \mathbf{x}_t + \mathbf{U}_f \mathbf{h}_{t-1} + \mathbf{b}_f), \tag{4.28}$$

$$\mathbf{i}_t = \sigma(\mathbf{W}_i \mathbf{x}_t + \mathbf{U}_i \mathbf{h}_{t-1} + \mathbf{b}_i), \tag{4.29}$$

$$\mathbf{o}_t = \sigma(\mathbf{W}_o \mathbf{x}_t + \mathbf{U}_o \mathbf{h}_{t-1} + \mathbf{b}_o), \tag{4.30}$$

$$\mathbf{c}_t = \mathbf{f}_t \odot \mathbf{c}_{t-1} + \mathbf{i}_t \odot \tanh(\mathbf{W}_c \mathbf{x}_t + \mathbf{U}_c \mathbf{h}_{t-1} + \mathbf{b}_c), \tag{4.31}$$

$$\mathbf{h}_t = \mathbf{o}_t \odot \tanh(\mathbf{c}_t); \tag{4.32}$$

where:

- $\mathbf{x}_t$ is the input at time t.
- $\mathbf{h}_t$ is the hidden state, and the all output.
- $\mathbf{c}_t$ is the cell state.
- $\sigma(\cdot)$ is the sigmoid function.
- $\odot$ denotes element-wise multiplication.
- $\mathbf{W}_f$, $\mathbf{W}_o$, $\mathbf{W}_i$ are the weight matrices for the forget, output, and input gates, respectively.
- $\mathbf{U}_f$, $\mathbf{U}_o$, $\mathbf{U}_i$ are the recurrent weight matrices for the forget, output, and input gates, respectively.

3. Gated Recurrent Unit Recurrent Neural Network (GRU)

A GRU is a type of Recurrent Neural Network (RNN) designed to model sequential data while reducing computational complexity compared to LSTM by using a simplified gating mechanism.

Architecture:

- **Input Layer:** Receives sequential input data.

- **GRU Layers:** Contain gating units that regulate information flow without separate cell states.
- **Output Layer:** Produces the output at each time step.

Key Formulas: For a GRU cell at time step t:

$$\mathbf{z}_t = \sigma(\mathbf{W}_z \mathbf{x}_t + \mathbf{U}_z \mathbf{h}_{t-1} + \mathbf{b}_z), \tag{4.33}$$

$$\mathbf{r}_t = \sigma(\mathbf{W}_r \mathbf{x}_t + \mathbf{U}_r \mathbf{h}_{t-1} + \mathbf{b}_r), \tag{4.34}$$

$$\tilde{\mathbf{h}}_t = \tanh(\mathbf{W}_h \mathbf{x}_t + \mathbf{U}_h (\mathbf{r}_t \odot \mathbf{h}_{t-1}) + \mathbf{b}_h), \tag{4.35}$$

$$\mathbf{h}_t = (1 - \mathbf{z}_t) \odot \mathbf{h}_{t-1} + \mathbf{z}_t \odot \tilde{\mathbf{h}}_t; \tag{4.36}$$

where:

- $\mathbf{x}_t$ is the input at time t;
- $\mathbf{h}_t$ is the hidden state;
- $\mathbf{z}_t$ is the update gate, which controls how much of the past hidden state is preserved;
- $\mathbf{r}_t$ is the reset gate, which determines how much of the past hidden state is forgotten;
- $\tilde{\mathbf{h}}_t$ is the candidate hidden state;
- $\sigma(\cdot)$ is the sigmoid function;
- $\odot$ denotes element-wise multiplication;
- $\mathbf{W}_z$, $\mathbf{W}_r$, $\mathbf{W}_h$ are the weight matrices for the update gate, reset gate, and candidate state, respectively;
- $\mathbf{U}_z$, $\mathbf{U}_r$, $\mathbf{U}_h$ are the recurrent weight matrices for the update gate, reset gate, and candidate state, respectively.

4. Nonlinear Exogenous (NNX) Model with Neural Network

An NNX model is a framework for modeling nonlinear systems with exogenous inputs. A neural network can be used to approximate the nonlinear function $f(\cdot)$.

Architecture:

- **Input Layer:** Receives the exogenous inputs.
- **Neural Network:** Models the nonlinear function $f(\cdot)$.

- **Output Layer:** Produces the model's prediction.

Key Formula:

$$y(t) = f\left(x(t), \theta\right) + e(t); \tag{4.37}$$

where:

- $y(t)$ is the output at time t;
- $u(t)$ is input value at time t;
- $x(t) = \Big[u(t), u(t-1), u(t-2), \ldots, u(t-n_u)\Big]$
- θ represents the neural network parameters;
- $f(\cdot)$ is the nonlinear function modeled by the neural network;
- $e(t)$ is the error term.

5. Echo State Network (ESN)

An ESN is a type of recurrent neural network that uses a fixed reservoir of neurons randomly connected to encode the dynamics of an input sequence [55], [56]. The reservoir state is combined with learned output weights to produce predictions. The mathematical representation of the system is as follows:

Initialization

- Number of reservoir neurons: N.
- Input vector dimension: M.
- Output vector dimension: O.
- Input weight matrix: $\mathbf{W}_{\text{in}} \in \mathbb{R}^{N \times M}$, with $\mathbf{W}_{\text{in}} \sim \mathcal{N}(0, \alpha)$;

  where:

  - $\mathcal{N}(\mu, \sigma^2)$ represents a normal distribution with mean μ and variance σ^2,
  - α is the variance of the normal distribution used to generate the input weight matrix $\mathbf{W}_{\text{in}}$.
- Reservoir weight matrix: $\mathbf{W} \in \mathbb{R}^{N \times N}$, with $\mathbf{W} \sim \mathcal{N}(0, 1)$, and $\mathbf{W}_{ij} = 0$ with probability $(1 - \text{sparsity})$.

The matrix $\mathbf{W}$ is scaled to have a spectral radius ρ:

$$\mathbf{W} = \mathbf{W} \cdot \frac{\rho}{\lambda_{\text{max}}}, \tag{4.38}$$

where:

- λ_{max} is the largest eigenvalue of $\mathbf{W}$,
- ρ: Target spectral radius of the reservoir weight matrix, determining the reservoir's dynamic range.

- Bias vector: $\mathbf{b} \in \mathbb{R}^N$, with $\mathbf{b}_i \sim \mathcal{N}(0, 1)$.

State Update

At each time step t, the reservoir state $\mathbf{x}(t) \in \mathbb{R}^N$ is updated using (4.39).

$$\begin{aligned} \mathbf{x}(t) = (1 - \tau)\mathbf{x}(t - 1) + \\ \tau \tanh(\mathbf{W}\mathbf{x}(t - 1) + \mathbf{W}_{\text{in}}\mathbf{u}(t) + \mathbf{b}); \end{aligned} \tag{4.39}$$

where:

- $\mathbf{u}(t) \in \mathbb{R}^M$ is the input vector at time t;
- τ: Leak rate, controlling the influence of the previous state.

Training

The reservoir states over T time steps are stored in $\mathbf{X} \in \mathbb{R}^{N \times T}$:

$$\mathbf{X} = [\mathbf{x}(1), \mathbf{x}(2), \ldots, \mathbf{x}(T)]. \tag{4.40}$$

Ridge regression is applied to find the output weight matrix $\mathbf{W}_{\text{out}} \in \mathbb{R}^{\text{output\_dim} \times N}$:

$$\mathbf{W}_{\text{out}} = \underset{\mathbf{W}_{\text{out}}}{\text{argmin}} \, \|\mathbf{Y} - \mathbf{W}_{\text{out}}\mathbf{X}\|_2^2 + \beta_{\text{reg}} \|\mathbf{W}_{\text{out}}\|_2^2, \tag{4.41}$$

where:

- $\mathbf{Y} \in \mathbb{R}^{O \times T}$ is the target output matrix.
- β_{reg}: Regularization strength in ridge regression, controlling overfitting.

Prediction

For a new input sequence $\mathbf{U} = [\mathbf{u}(1), \ldots, \mathbf{u}(T)]$:

1. Update the reservoir states using (4.39),
2. Compute the predicted output:

$$\mathbf{Y}_{\text{pred}}(t) = \mathbf{W}_{\text{out}}\mathbf{X}. \tag{4.42}$$

5. Wiener Model

The Wiener model [59–63] is a block-oriented framework that is commonly used in system identification and control engineering to represent nonlinear systems. It consists of two primary blocks connected in series as it is presented in Fig. 4.2:

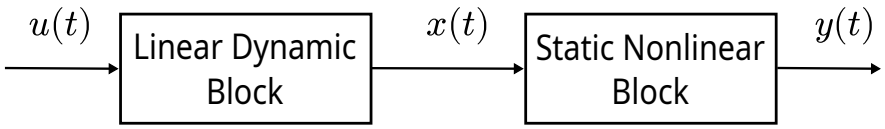

Figure 4.2: The Structure of Wiener Model.

1. **Linear dynamic block**: This block captures the system's linear dynamics, such as delays, filtering, and other linear time-invariant (LTI) behaviors. It is often represented by a transfer function $G(s)$ or $G(z)$ in the continuous or discrete domains, respectively.

2. **Static nonlinear block**: This block follows the linear dynamic block and models the system's static nonlinear characteristics. It maps the output of the linear block to the actual system output through a nonlinear function, such as an ANN, polynomial, sigmoid, saturation, or other types of nonlinear mappings.

The mathematical representation can be described as follows. If $u(t)$ is the input to the system, $y(t)$ is the output, and the linear and nonlinear components are represented by G and f, respectively, the Wiener model can be expressed as:

$$x(t) = G(u(t)) \quad \text{(linear block output)}, \tag{4.43}$$

$$y(t) = f(x(t)) \quad \text{(nonlinear block output)}; \tag{4.44}$$

where

$$G(u(t)) = \sum_{n=0}^{N} g_n u(t-n); \tag{4.45}$$

is the linear block output expressed as a convolution for discrete systems, where g_n are the impulse response coefficients.

Chapter 5

Experimental Procedures

5.1 SEA/PEA Actuator Experiment

The conducted experiments are divided into two main parts: evaluating the system's physical characteristics and collecting data for training neural networks. The first part focuses on determining the system characteristics, including the backlash angle, static and dynamic friction torque, gravity-induced torque, inertia, and stiffness. The second part involves collecting data within and outside the backlash region for neural networks training.

5.1.1 Defining Physical Characteristics

The following experiment was conducted to identify the backlash angular range. The link was located in the horizontal position. Then a small amount of torque was applied to the BLDC motor with a value of positive and negative 0.02 N·m with a cycle duration of 2.5 s. A counter torque, which opposed the BLDC torque, was applied to the link. The motor and the link encoders measured the position during the experiment with a sampling time of 0.0125 s. The difference between the displacement of the BLDC motor multiplied by the gearbox ratio and the displacement of the link represents free play in the backlash region. A graph of the experiments in Fig. 5.1 shows this difference over time. The backlash was determined as the difference between the maximum and minimum values. To ensure statistical reliability, the backlash was averaged over 100 experiments, resulting in a value of $2.66°$ (159.6 arcmin).

In addition to defining the backlash angle, this part of the experiment is aimed at evaluating the torque contribution due to gravity and static friction. The link was in one of the fixed starting angular positions: -90°, -60°, -45°, -30°, 0°, 30°, 45°, 60° and 90°. A force was applied to the link until the shaft of the pneumatic element began to rotate. The applied force was recorded and converted to torque considering the link's length of l = 0.3 m. A total of 100 experiments were conducted for each starting angular position. The resulting torque plot, with average values and standard deviation, is presented in Fig. 5.2. As can be seen, the closer the link to the horizontal position, the less torque is required to start rotating the output shaft of the pneumatic element.

The relationship between the torque required to start rotating the shaft T_{total}, the torque applied to the load cell $T_{\text{load cell}}$, and the torque due to gravity T_{gravity} is as follows:

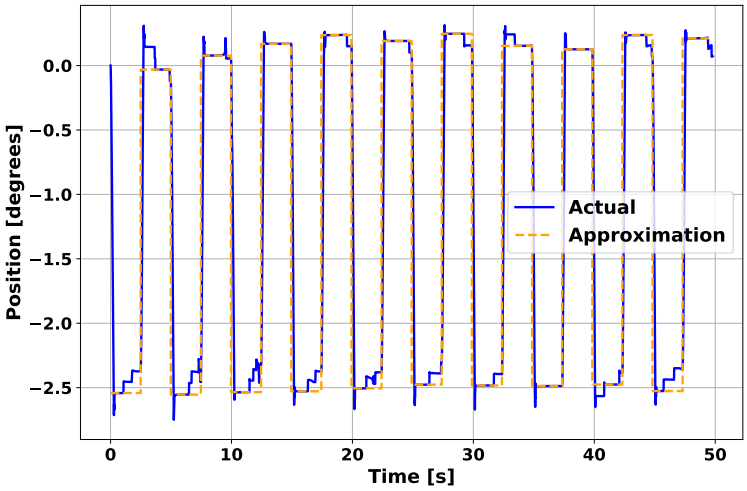

Figure 5.1: Backlash Identification.

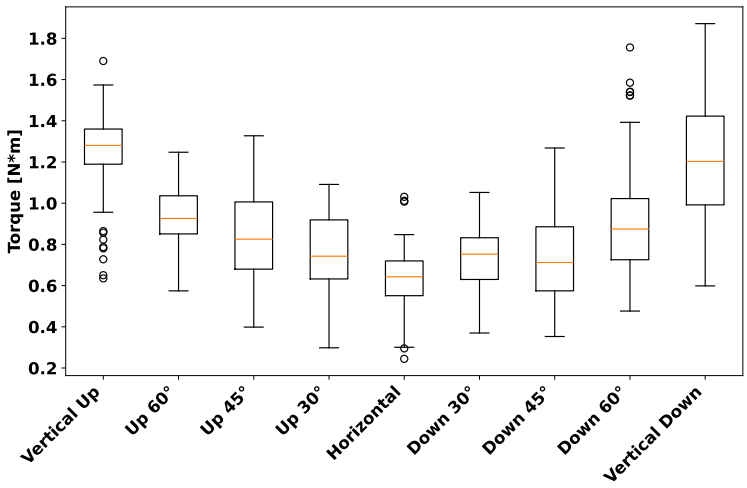

Figure 5.2: Torque to start rotating the shaft.

$$T_{\text{total}} = T_{\text{load cell}} + T_{\text{gravity}}, \tag{5.1}$$

where:

$$T_{\text{gravity}} = T_{\text{gravity max}} \cdot \cos(\alpha). \tag{5.2}$$

An optimization algorithm was run to find a $T_{\text{gravity max}}$ value that minimizes T_{total} values with the least MSE.

The resulting plot highlighting the relationship between torque and angular position is presented in Fig. 5.3. The total static torque without the gravity impact of the link (vertical position of the link) equals 1.22 N·m. The maximum gravity impact torque (horizontal position of the link) equals 0.58 N·m. These data are to be used in further experiments.

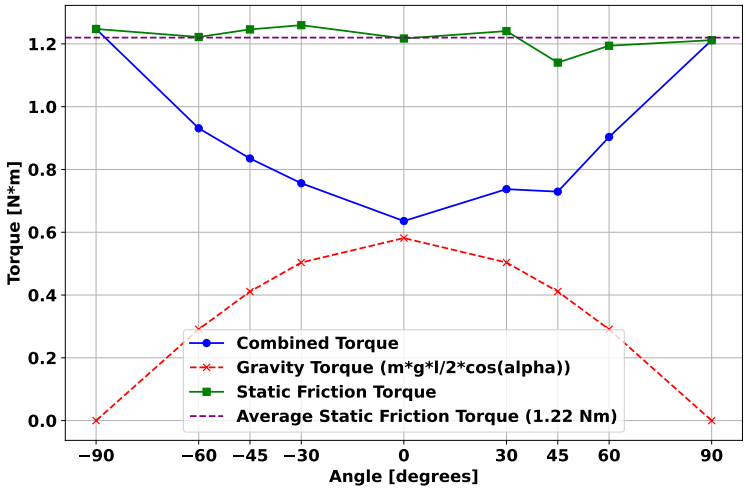

Figure 5.3: The combined (or actual) torque required to start the system rotating decreases when the link moves closer to the horizontal position with an increase in torque due to gravity.

5.1.2 Data Collection in the Backlash Region

The experiment was aimed at determining how the stiffness of the system changes with the pressure level in the pneumatic actuator chamber within the backlash region. The experimental protocol was as follows. The motor was set to a fixed position. This was achieved using position control mode; additionally, the rotor of the motor was mechanically locked to prevent rotation. The link was in a horizontal position with the load cell on its end. The pressure was supplied via the reservoir with a volume of 2 liters, which significantly increased the total system volume, initially consisting of the inner volume of pipes, a solenoid air valve, and the chamber of the pneumatic actuator. Once the pressure reached 6 bars, the solenoid air valve closed the system, and the experiment began.

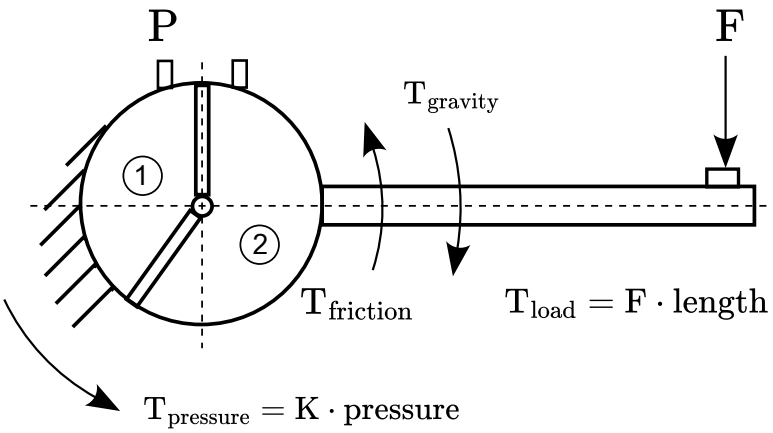

Figure 5.4: Torque Balance.

A force was repeatedly applied vertically downward according to the scheme depicted in Fig. 5.4 at a rate of approximately 2 Hz for a total time of 100 s. This was done to move the link, the pneumatic actuator, and the gearbox to the limit of the backlash. The pneumatic system was calibrated to allow gradual pressure reduction due to a controlled leak. This allowed system testing across a pressure range from 6 bars to zero bars. The collected data, including gradual pressure drop, force applied to the link, and an angular displacement of the actuator within the backlash region, are presented in Fig. 5.5. The applied force F was then converted to torque T_{load}:

$$T_{\text{load}} = F \cdot l. \tag{5.3}$$

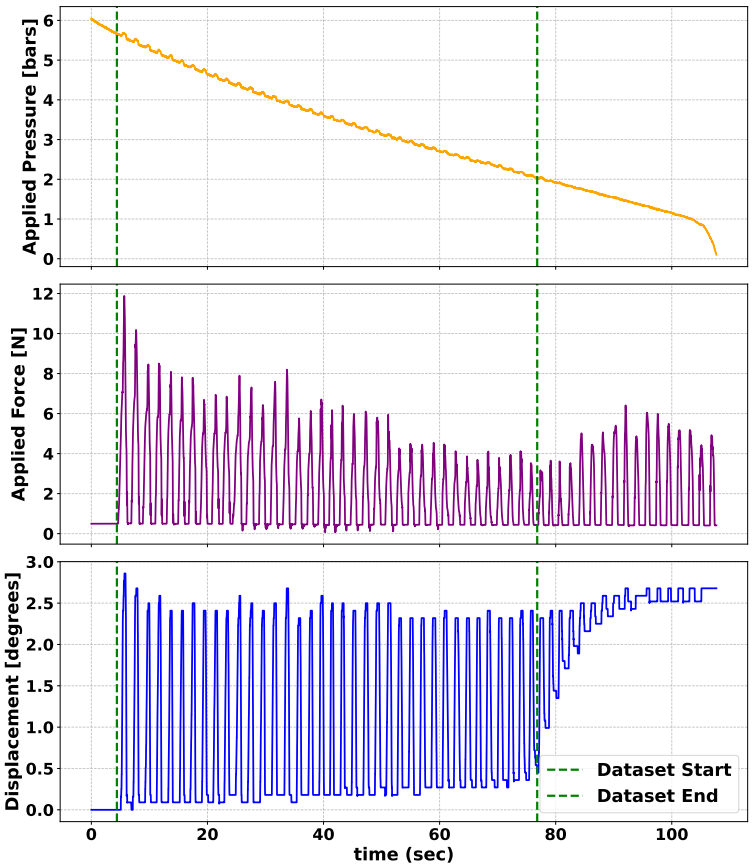

Figure 5.5: Experiment Data. The data set to train the ANNs was acquired in the range from the moment the force started to be applied until the moment when torque from the pressurized chamber was capable of moving the system back once the force was released.

As it is possible to see from the first plot in Fig. 5.5, the pressure in the system decreased from 6 bars down to 0 bars almost linearly, which ensured that every motion was done under different pressures, thus a different counter torque from the pneumatic actuator. The collected data were preprocessed to determine the respective values of the torque applied to the link capable of rotating the shaft of the actuator at the given

pressure. Fig. 5.6 shows this relationship and proves that is linear. These data were used to define the dynamic friction torque per the torque balance with unknown parameters K and $T_{\text{dynamic\_friction}}$ in Equation (5.4). Generally, the dynamic friction depends on angular velocity, especially in rotating equipment that reaches high-velocity values. This dependence is caused by heat generation that alters the viscosity of lubricant applied. Given that the experiment measured the dynamic friction when the system just started to rotate and the angular velocity was small, $T_{\text{dynamic\_friction}}$ is assumed to be constant in a low angular velocity range.

$$K \cdot \text{pressure} + T_{\text{dynamic\_friction}} = T_{\text{gravity}} + F \cdot l \tag{5.4}$$

A linear regression model was used to determine K and $T_{\text{dynamic\_friction}}$ values by minimizing the squared difference between the measured and predicted torque values. $T_{\text{dynamic\_friction}}$ = 0.22 N·m, K = 0.3587. The dynamic friction in the systems is less because more force is required to start the system rotation than to keep it moving.

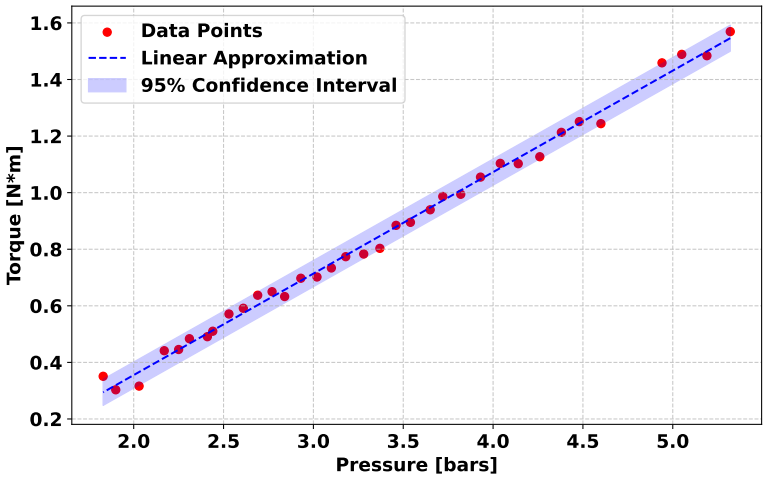

Figure 5.6: Torque vs Pressure Relationship.

The entire dataset expressed as a 3D plot, where axes represent torque, pressure, and angular position, is presented in Fig. 5.7. A slice of data, reflecting one cycle at a pressure around 4 bars, is presented in Fig. 5.8, clearly showing a hysteresis loop because part of the energy is lost in heat during compression and friction.

In addition, the stiffness of the system was defined based on the collected data. Stiffness is a measure of the resistance of an object to deformation under an applied load. In the context of rotational systems, stiffness can be defined as the ratio of the change in torque to the corresponding change in angular displacement. Mathematically, stiffness (k) is expressed in (5.5).

$$k = \frac{\Delta T}{\Delta \theta}, \tag{5.5}$$

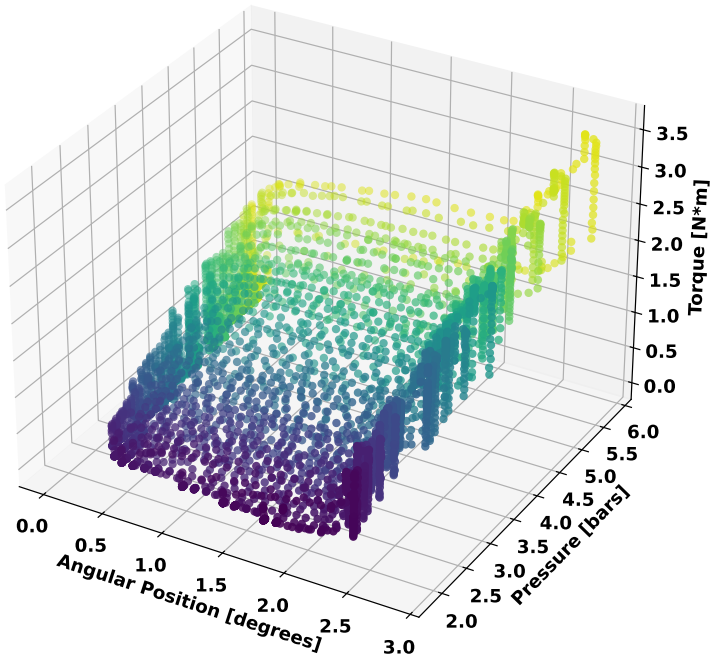

Figure 5.7: Relationship between torque, pressure, and angular position.

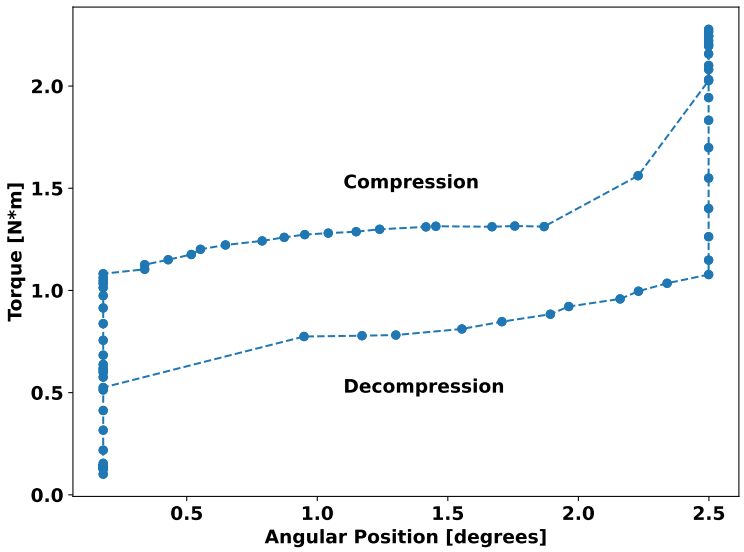

Figure 5.8: Position vs Torque Hysteresis during one cycle at a pressure around of 4 bars.

where ΔT represents the change in torque, and $\Delta\theta$ denotes the change in angular displacement. Given that the chamber pressure varies between 6 bar and 0 bar, a decrease in pressure was expected to result in a decrease in stiffness. This hypothesis was confirmed by the graphical representation in Fig. 5.9, which demonstrates a linear relationship between stiffness and pressure. In the region below 2 bar, the stiffness rapidly decreased to near zero. This is attributed to the insufficient torque generated by the pressurized chamber to overcome static friction, preventing the link from returning to its initial position.

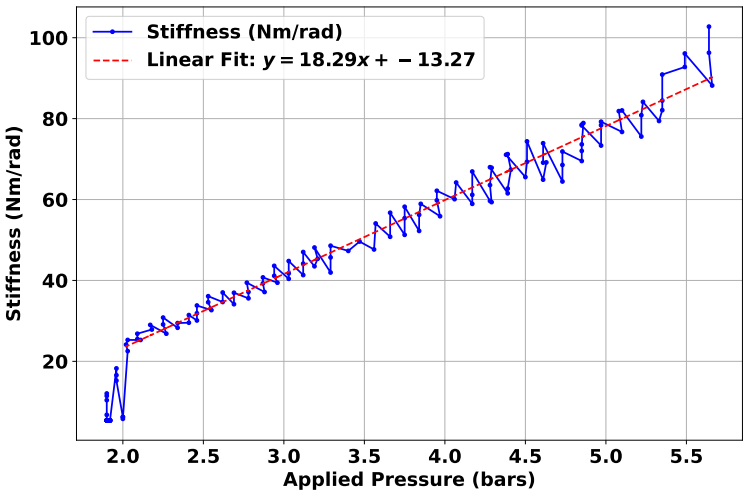

Figure 5.9: Change of the stiffness of the system with change of pressure.

5.1.3 Data Collection out of the Backlash Region

Another experiment was conducted to collect data out of the backlash region. The link was located in an upright vertical position. The BLDC motor was in a torque control mode. Then manual force was applied to the distal end of the link in such a way that the link was rotated to the down vertical position. Then the force was decreased so that the BLDC moved the link to the initial position, however, it still created a certain resistance level to this rotation. This cycle was repeated 100 times in an uninterruptable manner. During the experiment the following parameters were read and stored: link position, link velocity, pressure, applied force, Iq measured, and torque of the BLDC motor. The collected data are represented in Fig. 5.10.

5.2 Humanoid Robot Model Simulation Experiment

The experimental procedures consist of the following parts: creation of the reference trajectory using chirp function, simulating the model with the reference trajectory

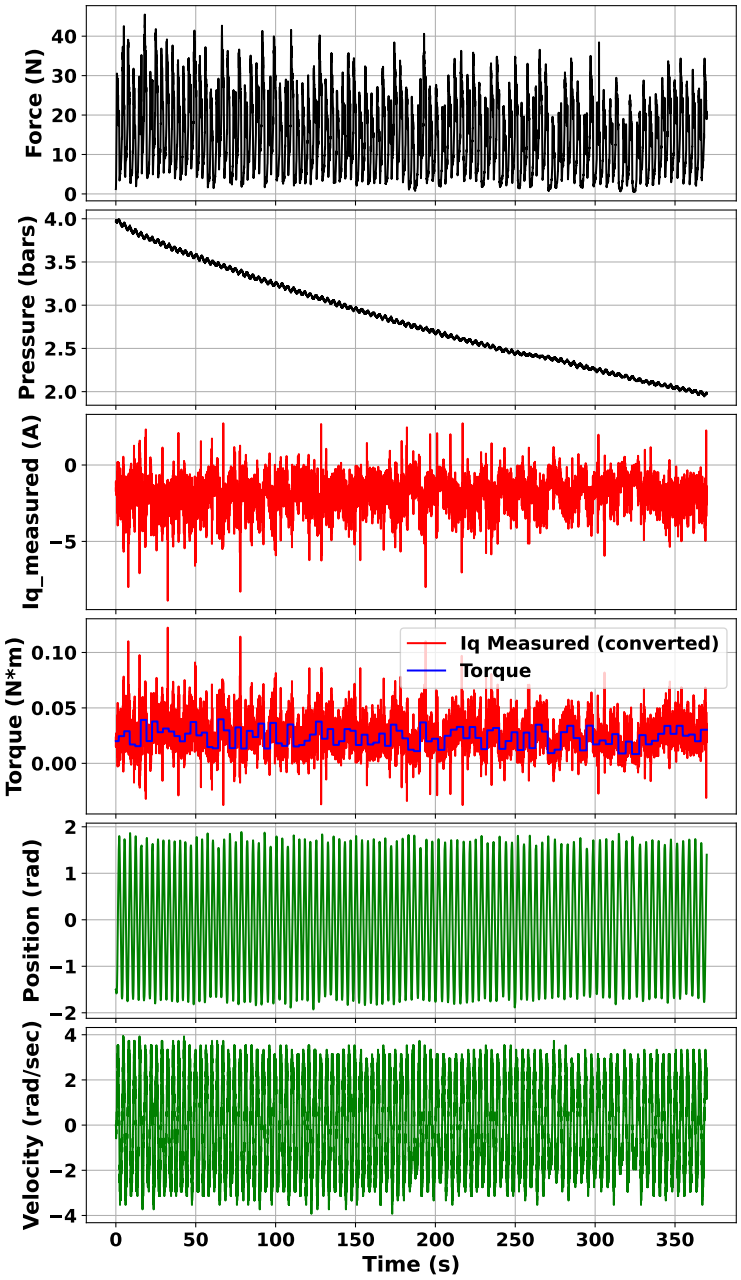

Figure 5.10: Collected Data for out of the Backlash Region.

and collecting data namely joint's position, velocity, acceleration, and torque, training different types of ANNs using collected data, and implementing ANN-augmented control algorithm.

5.2.1 Creation of the Reference Trajectory

The chirp function was used to generate the squat sequence trajectory for the base link pseudo-random motion from the starting position when z coordinate of the link is zero to the bottom position with z = -400 mm. This chirp function is overlapped with a sinusodial trajectory as per Equations (5.6)–(5.10) presented below.

Time Vector

The time vector is defined as:

$$t_n = \frac{n \cdot T}{N}, \quad n = 0, 1, 2, ..., N-1 \tag{5.6}$$

where:

- $T = 200$ sec is the total experiment time,
- $N = \frac{T}{T_s}$ is the number of samples,
- $T_s = \frac{1}{240}$ sec is the sample time.

Chirp Signal

The chirp trajectory is given by:

$$x(t) = \cos\left(2\pi\left(f_0 t + \frac{(f_1 - f_0)}{2T}t^2\right)\right) \tag{5.7}$$

where:

- $f_0 = 0.15$ Hz is the initial frequency,
- $f_1 = 1.5$ Hz is the final frequency,
- T is the experiment duration.

Time-Varying Amplitude Bounds

The maximum and minimum time-varying amplitude functions are:

$$A_{\max}(t) = A_{\text{amp,max}} \sin(2\pi f_{\text{amp,max}} t + \phi_{\max}) \tag{5.8}$$

$$A_{\min}(t) = A_{\text{amp,min}} \sin(2\pi f_{\text{amp,min}} t + \phi_{\min}) \tag{5.9}$$

where:

- $A_{\text{amp,max}} = 8.0$ is the amplitude of $A_{\max}$ variation,
- $A_{\text{amp,min}} = 0.0$ is the amplitude of $A_{\min}$ variation,
- $f_{\text{amp,max}} = 0.13$ Hz is the frequency of $A_{\max}$ variation,
- $f_{\text{amp,min}} = 0.07$ Hz is the frequency of $A_{\min}$ variation,
- $\phi_{\max} = \frac{\pi}{2}$ is the phase of $A_{\max}$ variation,
- $\phi_{\min} = \frac{\pi}{4}$ is the phase of $A_{\min}$ variation.

Amplitude-Modulated Trajectory

The amplitude-modulated trajectory is given by:

$$y(t) = \frac{A_{\max}(t) - A_{\min}(t)}{2} \cdot x(t) \tag{5.10}$$

The resulting reference trajectory for the base link is presented in Fig. 5.11.

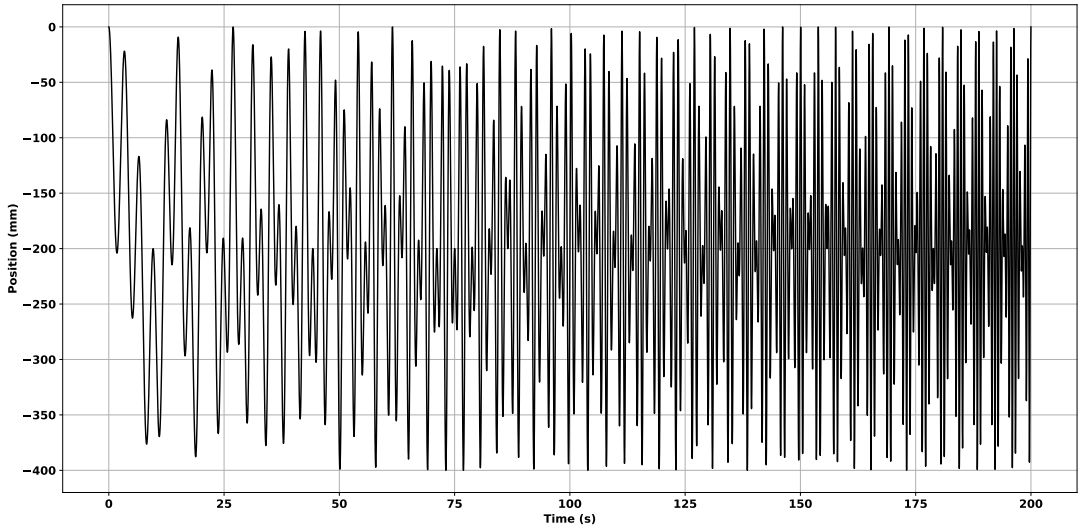

Figure 5.11: Reference trajectory for the base link using a chirp function.

Such trajectory allows to cover a wide range of frequencies and collected data which helps to increase accuracy of ANNs training process.

The base link trajectory was used to derive trajectories for six joints involved in the squat sequence, namely right and left hip pitch, knee, and ankle pitch joints. This was done using inverse kinematics (IK) equations developed in [91].

5.2.2 Simulating the Model and Data Collection

Simulating the model was done in PyBullet environment. Reference positions were sent to respective joints and data including actual position, velocity and torque were collected at a sampling frequency equal to 1/240 s. The results of this part of the experiment were stored in a CSV file that contains the following data: time stamp [s], reference position [rad], actual position [rad], actual velocity [rad/s], actual acceleration [rad/s^2] , and torque [N·m]. The actual acceleration was derived using the actual velocity and the sampling time. Torque values during the experiment are presented in Fig. 5.12. It is important to note that torque was applied not only to the joints involved in the squat motion—namely the hip pitch, knee, and ankle pitch—but also to the stationary joints, including the hip yaw, hip roll, and ankle roll. When the robot performs a squat, the primary movement occurs in the hip pitch, knee, and ankle pitch joints. However, due to the dynamics of a multi-body system, external forces such as gravity, ground reaction forces, and inertial coupling do not act exclusively on the moving joints. Even the joints that are intended to remain stationary—such as the hip yaw, hip roll, and ankle roll—are subjected to reaction torques. These torques arise because the robot's body segments are mechanically linked: forces applied at one joint propagate through the kinematic chain, resulting in torque demands on all connected joints, whether they are actively moving or not.

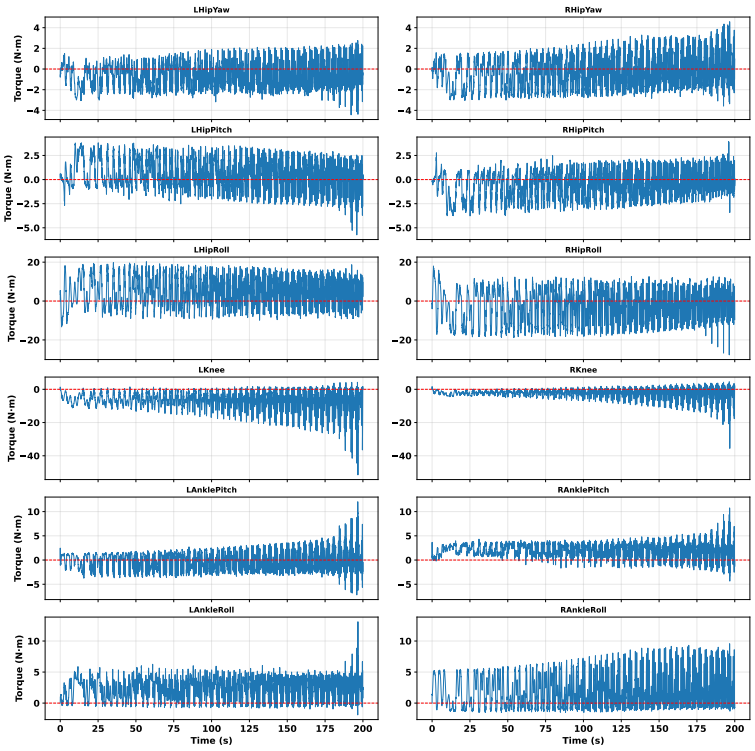

Figure 5.12: Torque during the experiment.

BLANK

Chapter 6

Results

The identified model consists of two parts. The first part represents the system in the backlash region, where the BLDC does not transfer its rotation to the link. The second part describes the system out of the backlash region where the transfer of motion from the BLDC motor to the link is rigid.

6.1 Modeling the SEA–PEA Elastic Actuator in the backlash region

Using the dataset acquired during operation in the backlash region, five models were trained: 1) an LSTM-RNN with a single layer of 10 neurons and memory 12, 2) a GRU with a single layer of 10 neurons and memory 10, 3) a FFNN with architecture 3-20-20-4-1, 4) an NNX where the NN part has architecture 3·10 (time delay)-10-10-10-1, and 5) an ESN with 470 neurons in the reservoir. All ANNs have the same three inputs θ, $\dot{\theta}$, and P, and one output T.

To ensure comparability, the total number of trainable parameters was maintained approximately equal. For this study, the difference remained at the 10% range maximum. The total number of adjustable parameters for a small LSTM with N=10 was used as a reference.

The formulas that calculate the total number of trainable parameters for each of the proposed networks are as follows:

$$\begin{aligned} TP(\text{LSTM\_RNN}) &= 4 \times (\text{N} \times (3 + \text{N}) + \text{N}) \\ &\quad + \text{N} \times 1 + 1 \\ &= 560 + 11 = 571, \end{aligned} \tag{6.1}$$

$$\text{where } N = 10$$

$$\begin{aligned} TP(\text{GRU}) &= 3 \times ((3 + N + 1) \times N + N) \\ &\quad + \text{N} \times 1 + 1 \\ &= 528 + 12 = 540, \end{aligned} \tag{6.2}$$

$$\text{where } N = 11$$

$$\begin{aligned} TP(\text{FFNN}) &= (\text{Inputs} \times L1 + L1) \\ &\quad + (L1 \times L2 + L2) \\ &\quad + (L2 \times L3 + L3) \\ &\quad + (L3 \times L4 + L4) \\ &= 80 + 420 + 84 + 5 = 589, \end{aligned} \tag{6.3}$$

$$\text{where Inputs} = 3,\ L1 = 20,\ L2 = 20,\ L3 = 4, \text{ and } L4 = 1$$

$$\begin{aligned} TP(\text{NNX}) &= (30 \times L1 + L1) \\ &\quad + (L1 \times L2 + L2) \\ &\quad + (L2 \times L3 + L3) \\ &\quad + (L3 \times L4 + L4) \\ &= 310 + 110 + 110 + 11 = 541, \end{aligned} \tag{6.4}$$

$$\text{where } L1 = 10,\ L2 = 10,\ L3 = 10, \text{ and } L4 = 1$$

$$\begin{aligned} TP(\text{ESN}) &= \text{Outputs} \times N = \\ &= 1 \times 470 = 470, \end{aligned} \tag{6.5}$$

$$\text{where Outputs} = 1, N = 470$$

The dataset was split into training and test parts in a proportion of 70% by 30% using a block-wise stratified sampling approach. Because the data reflect time-dependent actuator dynamics, a fully random split was avoided to preserve temporal continuity. Instead, the dataset was first divided into 10 equal consecutive blocks, each corresponding to a specific segment of the operating range of the actuator. From each block, a proportion of samples was allocated to both the training and test sets, maintaining the original distribution of pressure levels and dynamic states across the split. This ensures that the test set remains representative of the full range of operating conditions, while minimizing information leakage and preserving dynamic structure within each segment. Adam (Adaptive Moment Estimation) learning algorithm was used for LSTM-RNN, GRU, FFNN, and NNX. Ridge Regression (also known as Tikhonov regularization or L2 regularization) was used for ESN. The relevant hyperparameters of all the five ANNs are presented in Table 6.1.

The prediction results for all four neural network architectures are shown in Figs. 6.1–6.5.

Table 6.2 and Fig. 6.6 compare the performance of the neural networks, using the mean and standard deviation of the squared prediction error with ESN being the most

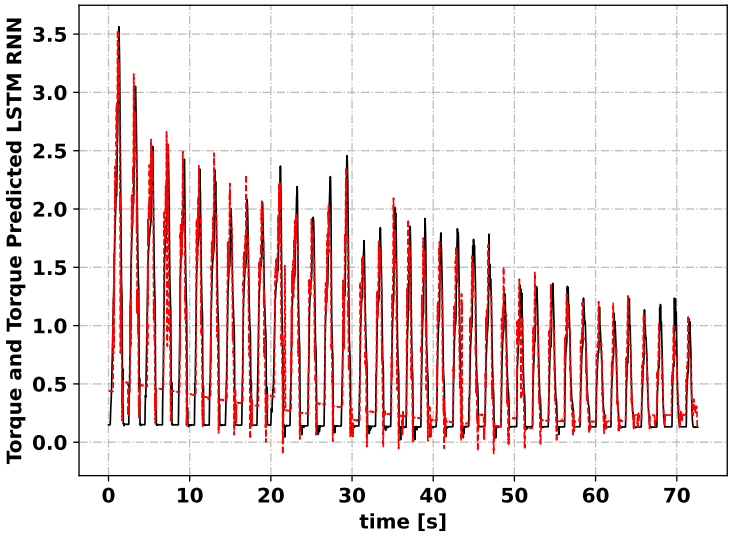

Figure 6.1: Model prediction (red dashed line) vs reference (black line) of LSTM-RNN.

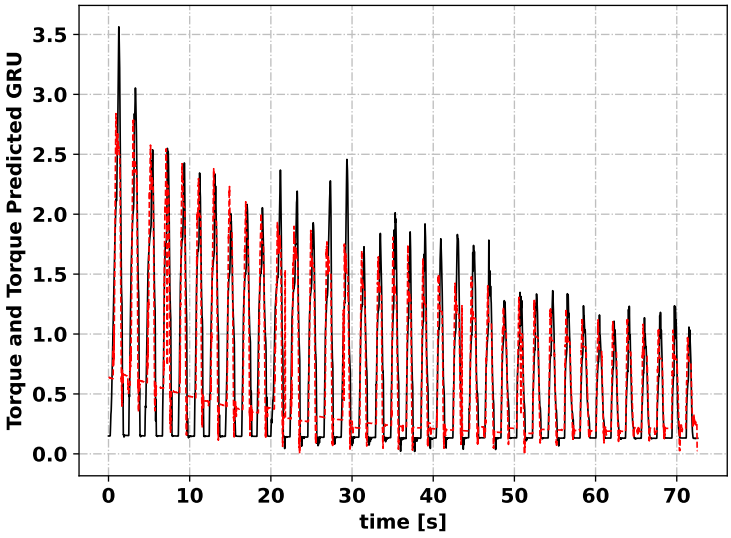

Figure 6.2: Model prediction (red dashed line) vs reference (black line) of GRU.

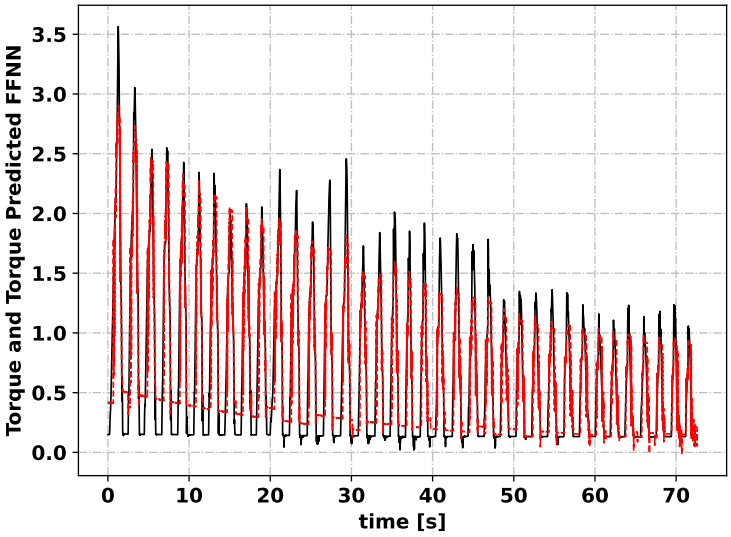

Figure 6.3: Model prediction (red dashed line) vs reference (black line) of FFNN.

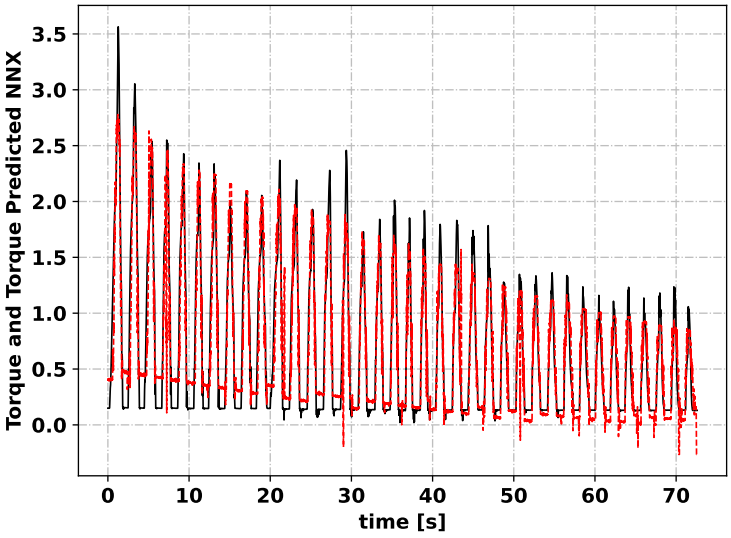

Figure 6.4: Model prediction (red dashed line) vs reference (black line) of NNX.

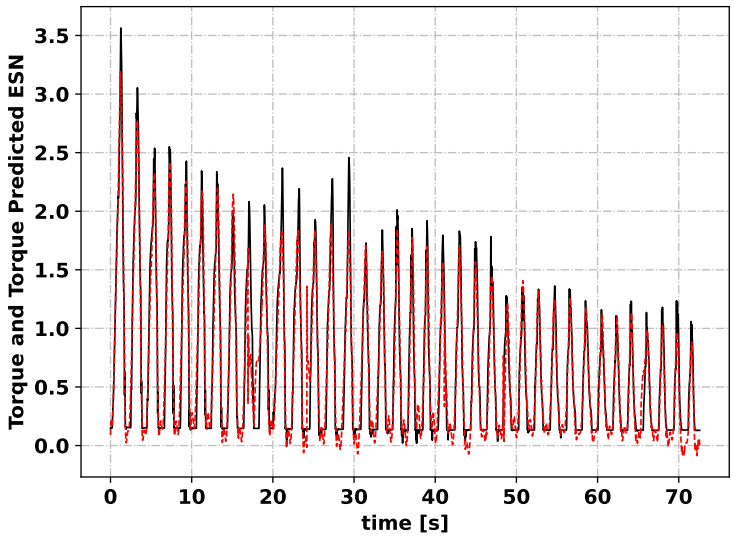

Figure 6.5: Model prediction (red dashed line) vs reference (black line) of ESN.

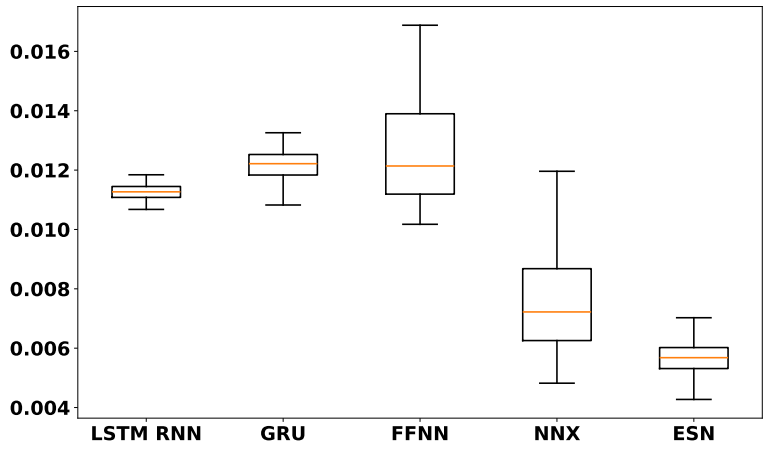

Figure 6.6: Test set MSE for different ANNs.

accurate. For the ESN the preliminary selection of optimal hyperparameters using a random grid search algorithm identified the best combination as N=550, ρ=0.8985, τ=0.11. To evaluate ESN performance, a series of training trials was conducted for hyperparameters within the following ranges: $100 < N < 550$, $0.75 < \rho < 1.25$ and $0.05 < \tau < 0.15$. The range of each hyperparameter was split into 10 intervals and the resulting MSEs are presented in Fig. 6.7. Axes X, Y, and Z are represented by N (reservoir size), ρ (spectral radius), and τ (leakage rate) respectively. The smallest MSE is reached for N=500, ρ=1.15 and τ=0.12.

NNX has the second smallest prediction error. Even though the FFNN showed the largest prediction error, its performance is comparable to that of the LSTM-RNN. This demonstrates the ability of FFNN architecture to effectively handle time-series data.

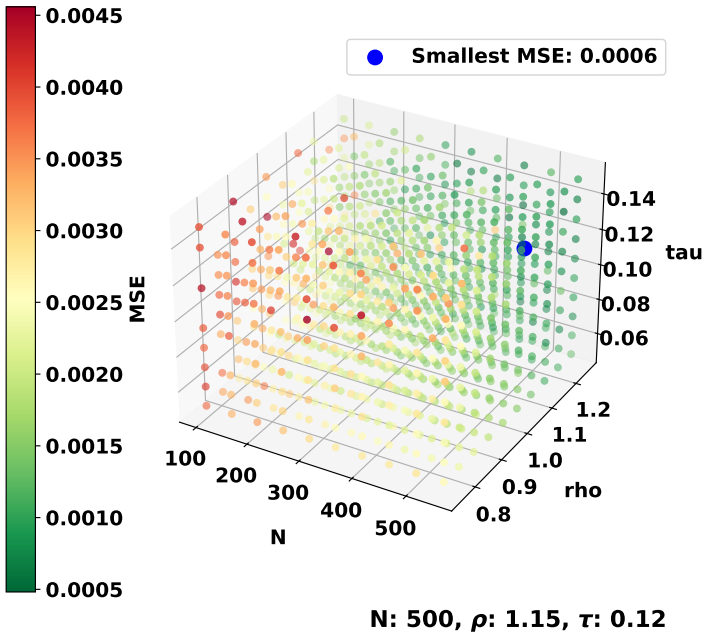

Figure 6.7: MSE as a function of N, ρ, and τ.

The comparative analysis of the efficiency of these ANNs is performed by estimating their time complexity (see section column in Table 6.2). Space complexity, which refers to the memory required to execute an algorithm, is not considered in the Thesis. Typically it is not as critical as execution time, especially in systems with very high sampling rates. The order of growth of an algorithm's basic operation count was selected as a performance criterion. It is denoted with O (Big-Oh) notation.

6.2 Modeling the SEA–PEA Elastic Actuator out of the backlash region

The out of the backlash region dataset was used as an input for the training of seven models: 1) linear model without any NN, 2) pure LSTM, 3) pure ESN, 4) Wiener LSTM, 5) Wiener FFNN, 6) Wiener FFNN with more trainable parameters, and 7) Wiener ESN. The selection of the Wiener models was mainly driven by the need to decompose the system into two simpler components. The system exhibits linear dynamics with the following nonlinear distortions caused by the pneumatic element. The first part can be identified using traditional linear methods, and the second part can be determined using learning and optimization techniques.

The formulas used to calculate the total number of trainable parameters for each of the proposed networks are as follows:

$$TP(\text{Pure Linear}) = \text{Inputs} = 3, \tag{6.6}$$

$$\begin{aligned} TP(\text{Pure LSTM}) &= 4 \times ((\text{Inputs} + \text{LSTM Layer}) \\ &\times \quad \text{LSTM Layer} + \text{LSTM Layer}) \\ &+ (\text{LSTM Layer} \times \text{Hidden Layer}) + \\ &\text{Hidden Layer} \\ &= 4 \times ((6+2) \times 2 + 2) + (2 \times 1) + 1 \\ &= 72 + 3 = 75, \\ \text{where Inputs} &= 6, \text{LSTM Layer} = 2, \text{Hidden Layer} = 1. \end{aligned} \tag{6.7}$$

$$\begin{aligned} TP(\text{Pure ESN}) &= \text{Outputs} \times N = \\ &= 1 \times 78 = 78, \\ \text{where Outputs} &= 1, \text{N} = 78 \end{aligned} \tag{6.8}$$

$$\begin{aligned} TP(\text{Wiener LSTM}) &= 4 \times ((\text{Inputs} + \text{LSTM Layer}) \\ &\times \quad \text{LSTM Layer} + \text{LSTM Layer}) \\ &+ (\text{LSTM Layer} \times \text{Hidden Layer}) + \\ &\text{Hidden Layer} + TP(\text{Pure Linear}) \\ &= 4 \times ((7+2) \times 2 + 2) + (2 \times 1) + 1 + 3 \\ &= 80 + 3 + 3 = 86, \\ \text{where Inputs} &= 7, \text{LSTM Layer} = 2, \text{Hidden Layer} = 1. \end{aligned} \tag{6.9}$$

$$\begin{aligned} TP(\text{Wiener FFNN}) &= (\text{Inputs} \times L1 + L1) \\ &+ (L1 \times L2 + L2) \\ &+ (L2 \times L3 + L3) \\ &+ TP(\text{Pure Linear}) \\ &= (7+1) \times 6 + (6+1) \times 4 + (4+1) \times 1 + 3 = \\ &= 48 + 28 + 5 + 3 = 84, \\ \text{where Inputs} &= 7,\ L1 = 6,\ L2 = 4, \text{ and } L3 = 1 \end{aligned} \tag{6.10}$$

$$\begin{aligned} TP(\text{Wiener FFNN Big}) &= (\text{Inputs} \times L1 + L1) \\ &+ (L1 \times L2 + L2) \\ &+ (L2 \times L3 + L3) \\ &+ TP(\text{Pure Linear}) \\ &= (7+1) \times 8 + (8+1) \times 6 + (6+1) \times 1 + 3 = \\ &= 64 + 54 + 7 + 3 = 128, \end{aligned} \tag{6.11}$$

$$\text{where Inputs} = 7,\ L1 = 8,\ L2 = 6, \text{ and } L3 = 1$$

$$TP(\text{Wiener ESN}) = \text{Outputs} \times N + TP(\text{Pure Linear}) = \\ = 1 \times 78 + 3 = 81, \tag{6.12}$$

$$\text{where Outputs} = 1, \text{N} = 78$$

The main hyperparameters of all seven models are presented in Table 6.3.

The comparative performance of the neural networks is presented in Table 6.4 and Fig. 6.8 with the smallest MSE for the Wiener ESN.

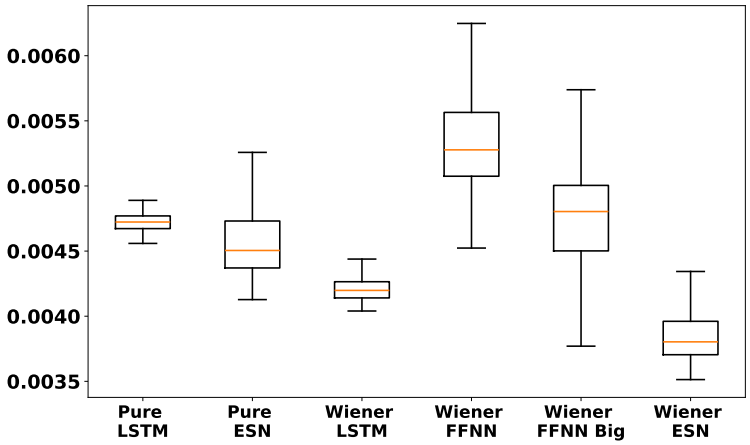

Figure 6.8: Test set MSE for different options.

6.3 Modeling the Humanoid Robot

6.3.1 Training ANNs

Four different ANN architectures were selected for comparison, namely Feed-Forward Neural Network (FFNN), Long Short-Term Memory Recurrent Neural Network (LSTM RNN), Nonlinear Exogenous Model with Neural Network (NNX), and Echo State Network (ESN). A FFNN consists of an input layer, one or more hidden layers, and an output layer. Information flows in one direction—from input to output—without cycles or loops. An LSTM is a type of Recurrent Neural Network (RNN) designed to model sequential data and capture long-range dependencies with memory cells. A NNX model is a framework for modeling nonlinear systems with exogenous inputs. A neural network can be used to approximate nonlinear functions. An ESN is a type of recurrent neural network that uses a fixed reservoir of neurons randomly connected to encode the dynamics of an input sequence. The reservoir state is combined with learned

output weights to produce predictions. All ANNs have 18 inputs (position, velocity, and acceleration) and 6 outputs (torque).

Performance of these ANNs is checked against two criteria, namely MSE and training time because then the ANNs are used in the systems that require a high accuracy and a very small computation time.

To ensure comparability, the total number of trainable parameters (TP) was kept approximately equal. As a reference, the total TP of a small LSTM with N=5 was taken.

Below are formulas that calculate the total number of trainable parameters for each of the proposed networks:

$$\begin{aligned} TP_{\text{LSTM\_RNN}} &= 4 \times (\text{N} \times (\text{inputs} + \text{N}) + \text{N}) \\ &\quad + \text{N} \times \text{outputs} + \text{outputs} \\ &= 840 + 60 + 12 = 912, \end{aligned} \tag{6.13}$$

$$\text{where } N = 5$$

$$\begin{aligned} TP_{\text{FFNN}} &= (\text{inputs} \times L1 + L1) \\ &\quad + (L1 \times L2 + L2) \\ &\quad + (L2 \times L3 + L3) \\ &\quad + (L3 \times L4 + L4) \\ &= 518 + 165 + 108 + 120 = 911, \end{aligned} \tag{6.14}$$

$$\text{where } L1 = 14,\ L2 = 11,\ L3 = 9, \text{ and } L4 = 12$$

$$\begin{aligned} TP_{\text{NNX}} &= (\text{Memory} \times \text{inputs} \times L1 + L1) \\ &\quad + (L1 \times L2 + L2) \\ &\quad + (L2 \times L3 + L3) \\ &\quad + (L3 \times L4 + L4) \\ &= 724 + 35 + 56 + 96 = 911, \end{aligned} \tag{6.15}$$

$$\text{where Memory} = 5,\ L1 = 4,\ L2 = 6,\ L3 = 7, \text{ and } L4 = 12$$

$$\begin{aligned} TP_{\text{ESN}} &= \text{outputs} \times N = \\ &= 12 \times 76 = 912, \end{aligned} \tag{6.16}$$

$$\text{where } N = 76$$

The dataset was split into training and testing parts in a proportion of 80% to 20% respectively. Given that the dataset reflects dynamic data, the split was done in a non-random way, with the first consecutive 80% of data assigned for training and the remaining 20% for testing. Adam (short for Adaptive Moment Estimation) learning algorithm was used for LSTM-RNN, FFNN, and NNX. Ridge Regression (also known as Tikhonov regularization or L2 regularization) was used for ESN.

The training accuracy results for all four neural network architectures are shown in Figs. 6.9–6.12.

The LSTM-RNN closely follows the measured torque trajectory across both low- and high-variation intervals. The predicted curve overlaps the measured torque for most of the time window, with only minor deviations during abrupt torque changes. This indicates that the LSTM effectively captures temporal dependencies and nonlinear patterns in the torque dynamics. Overall, it provides smooth and accurate predictions with minimal phase lag.

The FFNN reproduces the general trend of the torque signal but shows noticeable lag and reduced accuracy during rapid torque transitions. Since the FFNN has no recurrent or memory mechanisms, it cannot incorporate temporal context, and therefore struggles with the dynamic portions of the torque profile. This results in underestimation or overshooting during sharp peaks and troughs. The performance is adequate for slow-varying torque but insufficient for capturing fast dynamic changes.

The NNX model provides a closer match to the measured torque than the FFNN, particularly during moderately dynamic segments. It captures nonlinear variations better and reduces the amplitude errors seen in the FFNN. However, compared to the LSTM-RNN, the NNX still exhibits small discrepancies during high-frequency or abrupt torque changes, indicating partial but not full utilization of temporal information.

The ESN predicts the overall shape of the torque signal well, reflecting the reservoir's ability to encode temporal structure. During smooth or quasi-steady intervals, the predicted torque aligns closely with the measured one. However, the ESN shows occasional oscillatory artifacts or small overshoots, likely due to the reservoir's fixed random dynamics being only partially tuned to the system. Despite this, the ESN performs competitively and demonstrates strong capability in modeling the underlying torque dynamics.

Table 6.5 and Fig. 6.13 summarize the quantitative performance of the four neural network architectures. The ESN achieved the lowest MSE on both the training and test datasets, indicating the highest overall prediction accuracy. As expected for reservoir computing, it also exhibited the largest standard deviation on the test dataset, which is attributable to the random initialization of the reservoir and input weight matrices.

The LSTM_RNN and FFNN obtained the next-best results, with performance values that are very close to each other. The FFNN produced a slightly lower test MSE than the

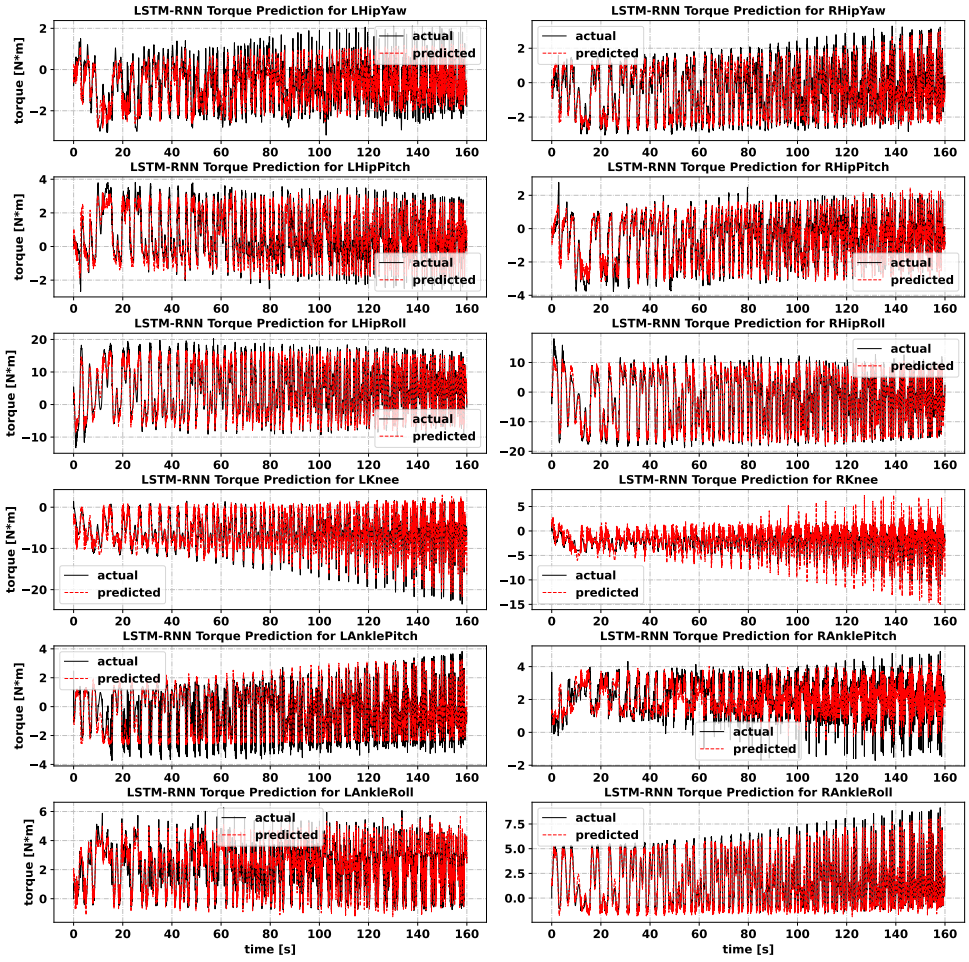

Figure 6.9: LSTM-RNN Torque Prediction.

LSTM_RNN, while the LSTM_RNN showed marginally higher stability, as reflected by its standard deviation values.

The NNX model performed the weakest among the architectures in terms of both training and testing MSE, confirming that its structure was less suited for capturing the temporal characteristics of the torque data compared to recurrent or reservoir-based models.

6.3.2 Conducting Benchmark Experiments

The main criteria for evaluation of the ANN-augmented control algorithm are MSE between reference and actual positions and energy consumed to follow the reference trajectory. This comparison is based on the benchmark values that are obtained from two experiments of following the same trajectory in PyBullet. One experiment uses POSITION_CONTROL mode and the other uses classic PID control in PyBullet TORQUE_CONTROL mode.

The second experiment includes a preliminary process of selecting the optimal PID

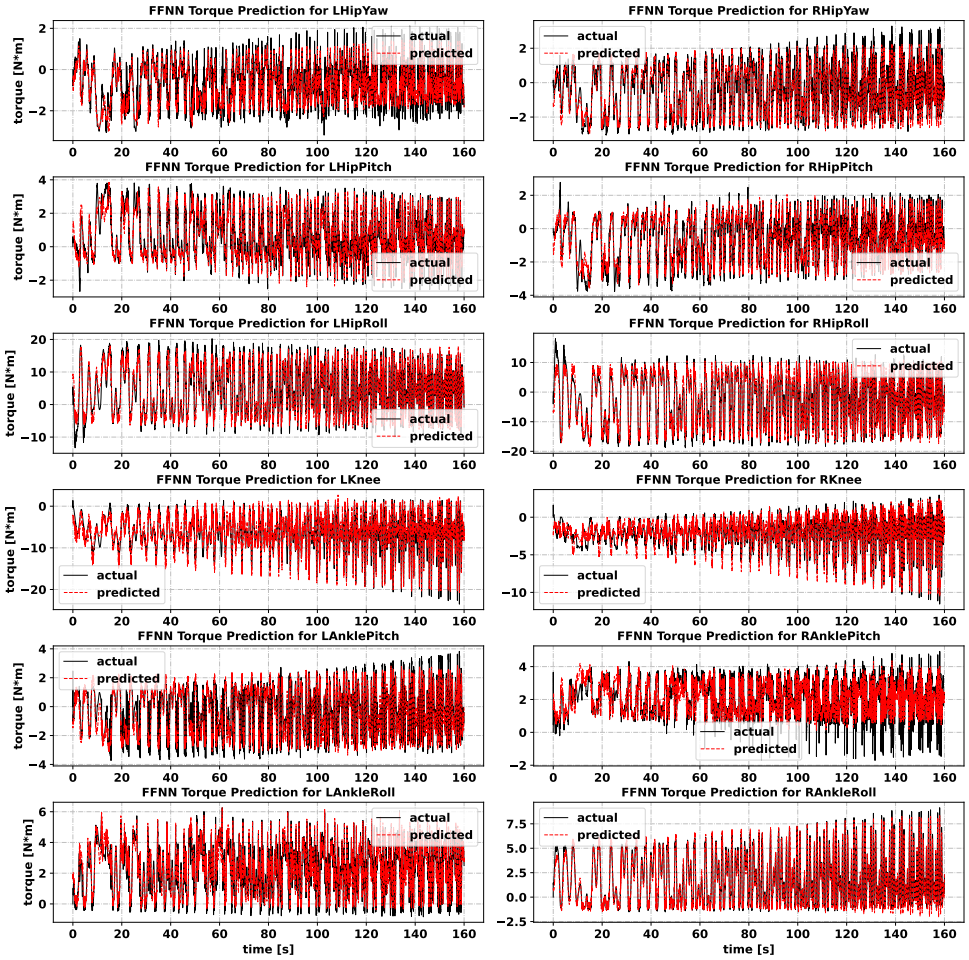

Figure 6.10: FFNN Torque Prediction.

parameters K_p, K_i, and K_d. A benchmark trajectory with two full squat cycles during 10 seconds is presented in Fig. 6.14. Positions acquired from the PyBullet model and prototype, in different timestamps of one squat sequence, are shown in Figs. 6.15– 6.16.

As described in Algorithm 2 a random grip search method is implemented in PyBullet environment for finding the optimal PID parameters.

The two stage process was implemented to increase precision of the outcome. First stage conducted random grid search in a wide range of PID parameters. Then after analysing the results, more narrow ranges are selected for processing in the second stage for fine tuning. The outcome of the first stage is presented in Fig. 6.17. As it can be seen, MSE has a high correlation level with K_p but not really depends on K_i, and K_d.

Then during the second stage the random grid search process was run again and it resulted in a scatter plot presented in Fig. 6.18.

The final optimal PID parameters are as follows: $K_p = 60.0$, $K_i = 3.5$, and $K_d = 1.0$.

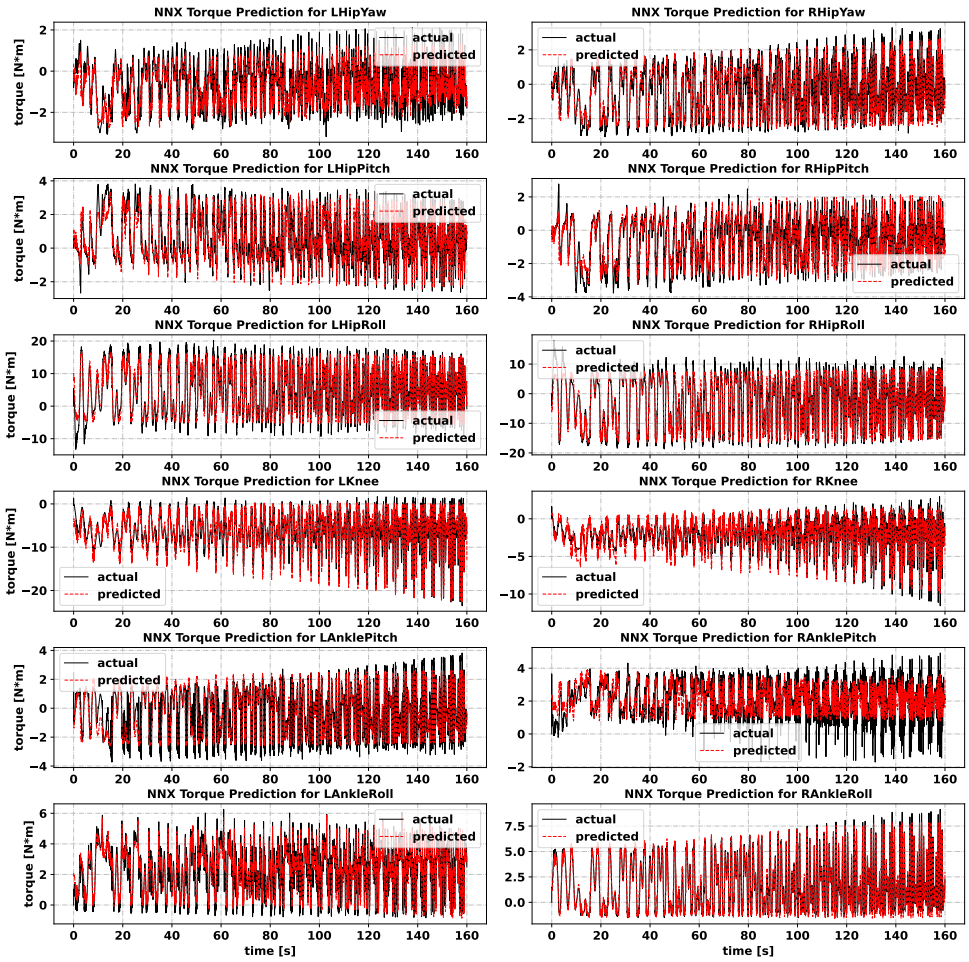

Figure 6.11: NNX Torque Prediction.

Algorithm 2 Random Grid Search for Optimal PID Parameters

Input: The benchmark trajectory τ_j where j = 1, ..., 6

Output: $K_{p\_optimal}$, $K_{i\_optimal}$, and $K_{d\_optimal}$

1: Initialise range K_p, K_i, and K_d with maximum and minimum values
2: **while** the experiment is running **do**
3: Select random values of K_p, K_i, and K_d from their respective ranges
4: Simulate the benchmark trajectory in PyBullet and records actual position, actual velocity, and torque applied at a sampling rate 1/240 s
5: Record the base link $Z_{base\ link}$ coordinate
6: **if** $Z_{base\ link} < Z_{min}$ **then**
7: Go to the next combination of the PID parameters
8: **end if**
9: Calculuate MSE
10: Append MSE, K_p, K_i, and K_d values to respective numpy arrays
11: **end while**
12: Save numpy arrays in csv file
13: Analyze csv and select the optimal PID parameters (or range of the optimal PID parameters)

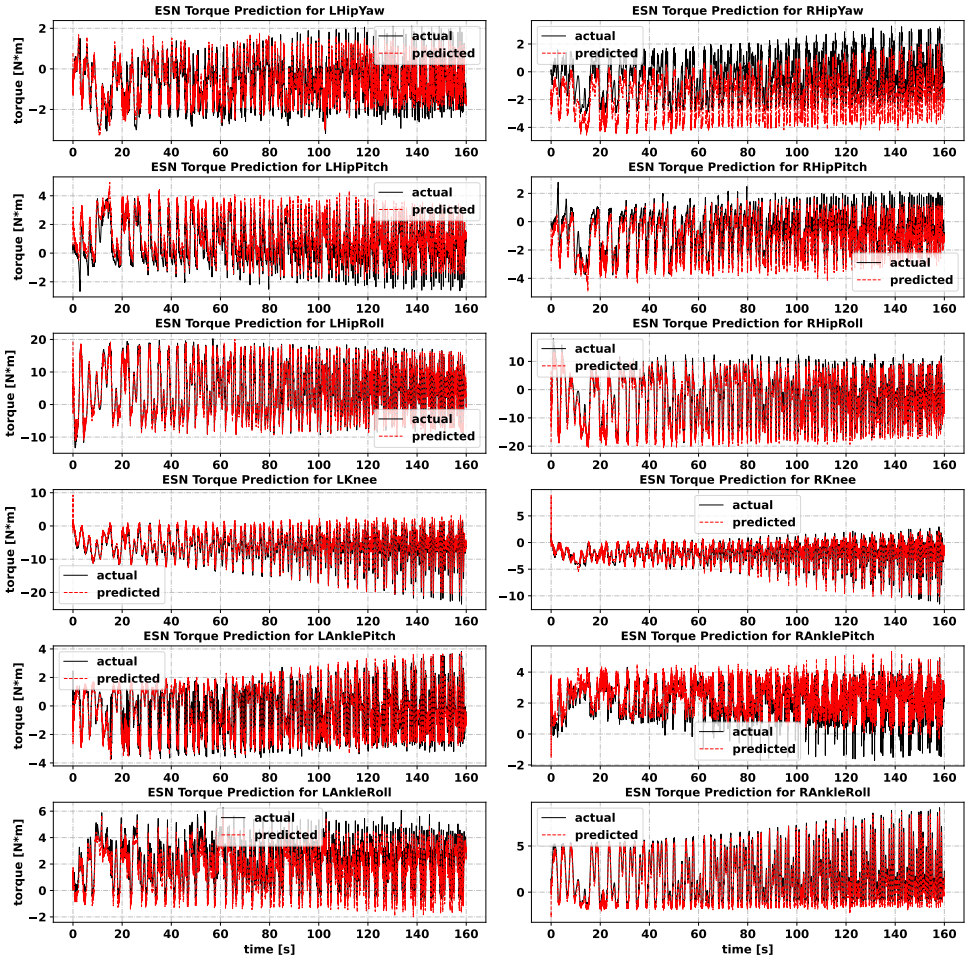

Figure 6.12: ESN Torque Prediction.

6.3.3 Implementing ANN-augmented Control Algorithm

The investigated control algorithm is very similar to Computed Torque Control because pre-trained ANN learns the system's inverse dynamics and it acts similarly to computed torque control. The difference is that CTC requires and has a precise model while the precision of ANN depends on different factors like the quality of the dataset provided for training, or in case of ESN a random initialization of the weight matrix. The control scheme is presented in Fig. 6.19. It has two loops: a standard feed-back based on PID control and a feed-forward implemented by ESN.

The ANN-augmented TORQUE_CONTROL mode employs the ESN trained in the previous section as a feed-forward module, effectively replacing the need for an explicit analytical model of the system as would be required in a classical MPC framework. By providing model-like predictions directly from data, the ESN compensates for nonlinearities and unmodeled dynamics that are difficult to capture analytically, thereby enhancing the controller's ability to track the desired trajectory.

The comparative results demonstrate that the ANN-augmented TORQUE_CONTROL

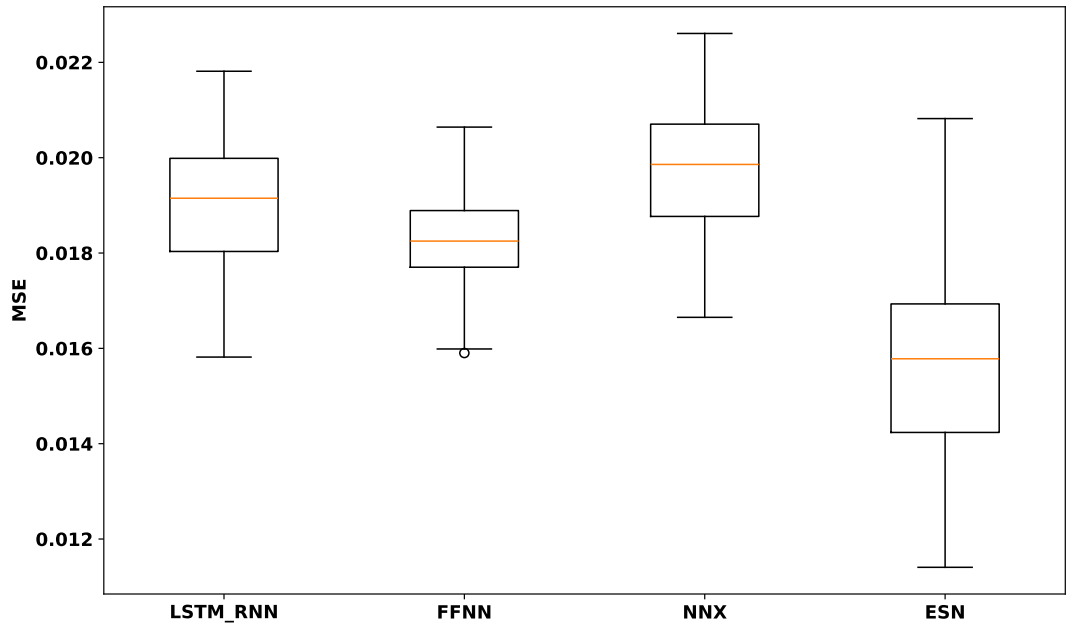

Figure 6.13: Test set MSE for different ANNs.

mode achieves superior tracking performance relative to both POSITION_CONTROL and conventional PID TORQUE_CONTROL modes. This improvement is visually evident in Figs. 6.20–6.22, where the ANN-based controller more closely follows the reference squat trajectory. Quantitatively, Table 6.6 highlights that the ANN-augmented approach reduces the mean-squared tracking error (MSE) by **66%** compared to POSITION_CONTROL and by **87%** compared to PID TORQUE_CONTROL. In addition to accuracy, the method significantly improves energy efficiency: the total energy consumption is reduced by **63%** and **58%** relative to the same two baselines.

To further assess robustness, an additional experiment was conducted by adding 4 kg of weight to the robot and repeating the squat sequence. Under this heavier load, the ANN-augmented controller continues to demonstrate favorable characteristics. Although its MSE increases, likely because the ESN was trained on data from the nominal 17.1 kg configuration, it still consumes approximately half the energy required by the POSITION_CONTROL mode. In contrast, the PID TORQUE_CONTROL mode deteriorates considerably under the increased load, exhibiting an MSE higher by **418%** and energy consumption higher by **161%** compared to the ANN-augmented mode. These results, summarized in Table 6.7, indicate that the ANN-enhanced controller generalizes more effectively to moderate variations in system dynamics than the classical torque-control baseline.

Collectively, these findings confirm that the ESN-augmented approach provides not only improved trajectory-tracking accuracy and energy efficiency under nominal

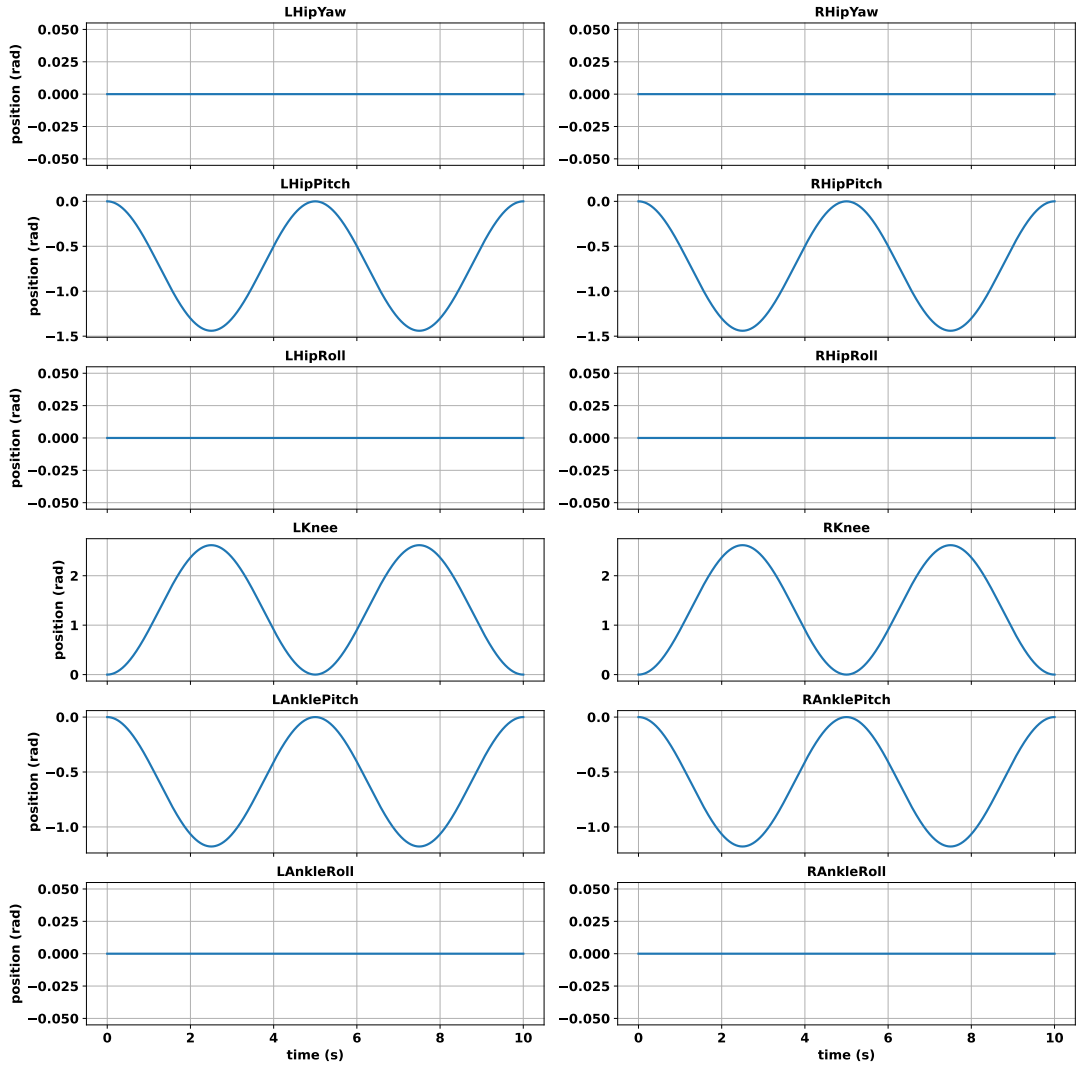

Figure 6.14: Benchmark trajectory for comparison of different control algorithms.

conditions, but also greater robustness to changes in system mass. This demonstrates the practical value of integrating learned dynamic models into torque-based control frameworks for humanoid-robot motions such as squatting.

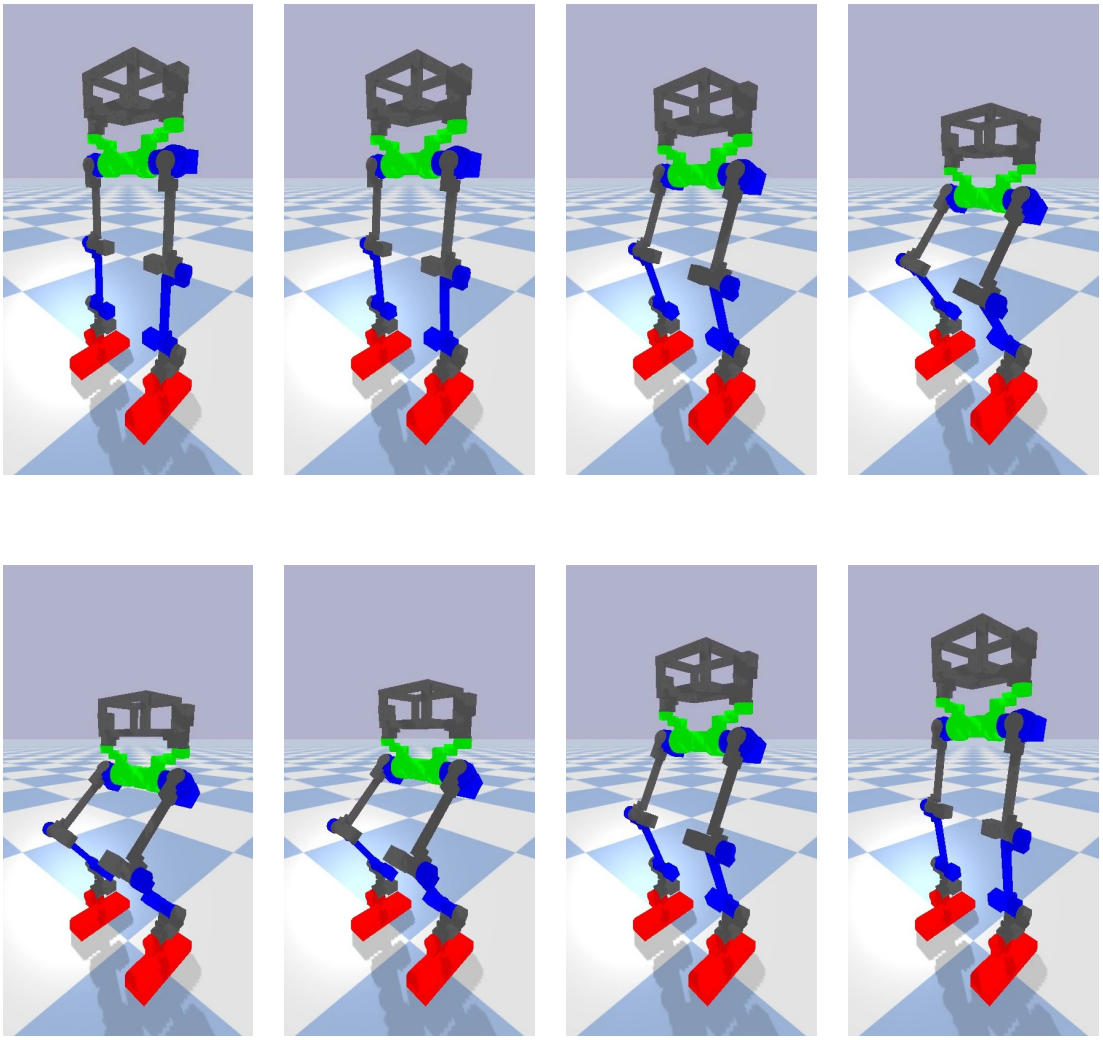

Figure 6.15: Position of the PyBullet model at the different timestamps.

Table 6.1: Hyperparameters for FFNN, LSTM-RNN, GRU, NNX, and ESN.

Hyper parameter	**LSTM-RNN**	**GRU**	**FFNN**	**NNX**	**ESN**
Learning Rate	0.001	0.001	0.001	0.001	*n/a*
Number of Trainable Parameters	571	540	589	541	470
Activation Function	tanh	tanh	tanh	tanh	tanh
Batch Size	6600	6600	6600	6600	6600
Epochs	1000	1000	1000	1000	1000
Optimization Algorithm	Adam	Adam	Adam	Adam	Ridge Regression
Dropout Rate	0.0	0.0	0.0	0.0	*n/a*
Reservoir Size	*n/a*	*n/a*	*n/a*	*n/a*	450
Spectral Radius ρ	*n/a*	*n/a*	*n/a*	*n/a*	1.25
Input Scaling α	*n/a*	*n/a*	*n/a*	*n/a*	0.1
Leak Rate τ	*n/a*	*n/a*	*n/a*	*n/a*	0.13
Reservoir Sparsity	*n/a*	*n/a*	*n/a*	*n/a*	0.1
Regularization β	*n/a*	*n/a*	*n/a*	*n/a*	0.01

Table 6.2: Performance Comparison of Different Options in the Backlash Region.

NN Type	**Time complexity**	**MSE**		**Standard Deviation**	
		Train Dataset	**Test Dataset**	**Train Dataset**	**Test Dataset**
LSTM-RNN	$O(T \cdot N^3)$	0.0037	0.0113	0.0001	0.0003
GRU	$O(T \cdot N^3)$	0.0042	0.0122	0.0002	0.0006
FFNN	$O(N^2)$	0.0032	0.0127	0.0007	0.0017
NNX	$O(T \cdot N^3)$	0.0014	0.0076	0.0009	0.0018
ESN	$O(N^2)$	0.0011	0.0057	0.0004	0.0005

Table 6.3: Hyperparameters for Pure and Wiener Models.

Hyper parameter	**Pure Linear**	**Pure LSTM**	**Pure ESN**	**Wiener LSTM**	**Wiener FFNN**	**Wiener FFNN Big**	**Wiener ESN**
Learning Rate	*n/a*	0.001	*n/a*	0.001	0.001	0.001	*n/a*
Number of Trainable Parameters	3	75	78	86	84	128	81
Activation Function	*n/a*	tanh	*n/a*	tanh	relu	relu	*n/a*
Batch Size	*n/a*	32	32	32	32	32	32
Epochs	*n/a*	10	10	10	10	10	10
Optimization Algorithm	Ridge Regression	Adam	Ridge Regression	Adam	Adam	Adam	Ridge Regression
Dropout Rate	*n/a*	0.0	*n/a*	0.0	0.2	0.2	*n/a*
Reservoir Size	*n/a*	*n/a*	78	*n/a*	*n/a*	*n/a*	73
Spectral Radius ρ	*n/a*	*n/a*	0.95	*n/a*	*n/a*	*n/a*	0.95
Input Scaling α	*n/a*	*n/a*	0.1	*n/a*	*n/a*	*n/a*	0.1
Leak Rate τ	*n/a*	*n/a*	0.066	*n/a*	*n/a*	*n/a*	0.066
Reservoir Sparsity	*n/a*	*n/a*	0.1	*n/a*	*n/a*	*n/a*	0.1
Regularization β	*n/a*	*n/a*	0.01	*n/a*	*n/a*	*n/a*	0.01

Table 6.4: Performance Comparison of Different Options Out of Backlash Region.

NN Type	**Time Complexity**	**MSE**		**Standard Deviation**	
		Train Dataset	**Test Dataset**	**Train Dataset**	**Test Dataset**
Pure Linear	$O(N)$	0.0112	0.0137	0.0000	0.0000
Pure LSTM	$O(T \cdot N^3)$	0.0029	0.0047	0.0001	0.0001
Pure ESN	$O(N^2)$	0.0025	0.0046	0.0002	0.0003
Wiener LSTM	$O(T \cdot N^3)$	0.0023	0.0042	0.0001	0.0001
Wiener FFNN	$O(N^2)$	0.0042	0.0053	0.0002	0.0004
Wiener FFNN Big	$O(N^2)$	0.0034	0.0048	0.0002	0.0004
Wiener ESN	$O(N^2)$	0.0027	0.0038	0.0001	0.0002

Table 6.5: Performance of Different ANNs

NN Type	Training Time (s)	MSE		Standard Deviation	
		Train Dataset	Test Dataset	Train Dataset	Test Dataset
LSTM-RNN	3.69	0.0075	0.0189	0.0008	0.0013
FFNN	2.62	0.0076	0.0183	0.0006	0.0010
NNX	2.75	0.0096	0.0198	0.0009	0.0013
ESN	0.19	0.0068	0.0157	0.0008	0.0019

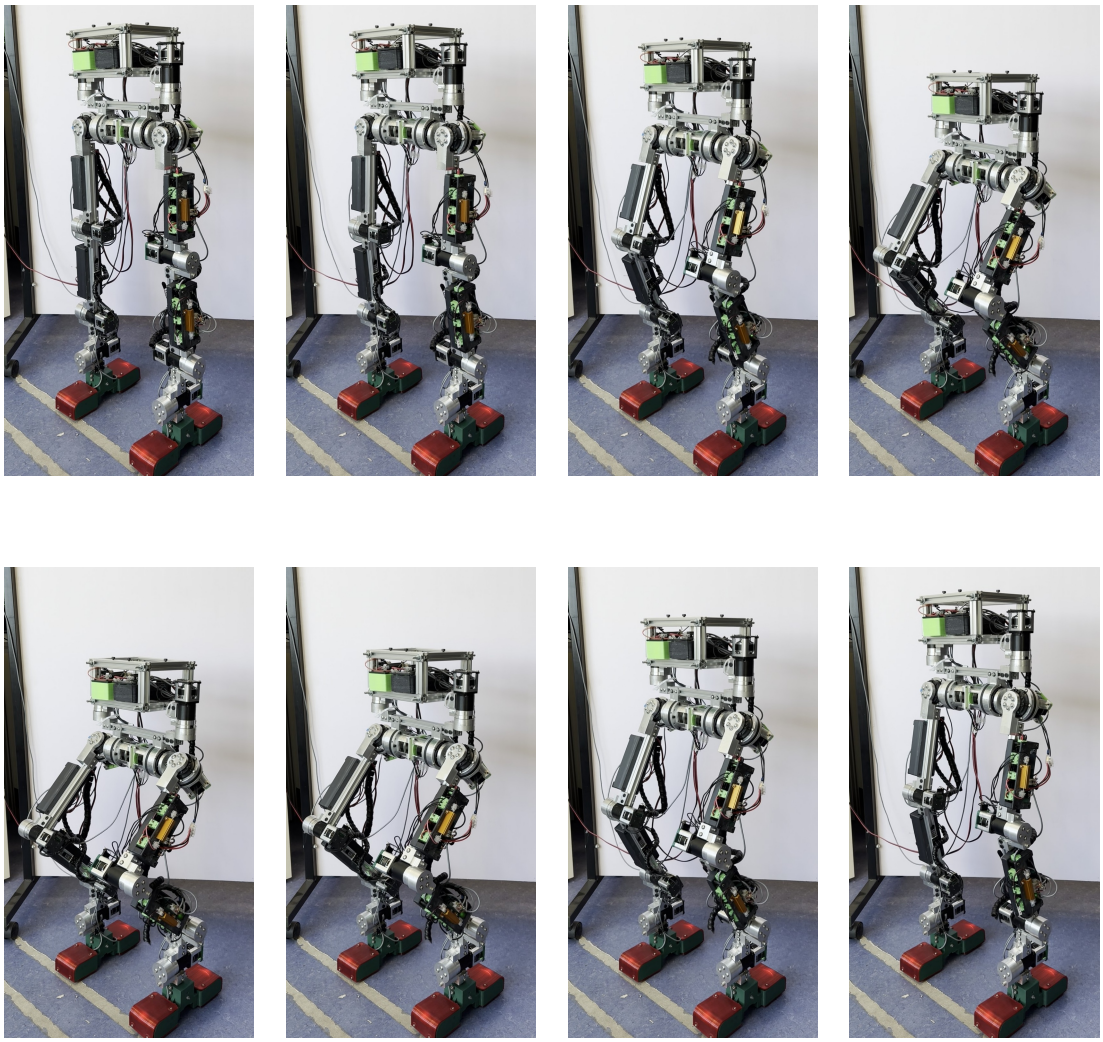

Figure 6.16: Position of the humanoid robot at the different timestamps.

Table 6.6: Comparison of Control Modes

Mode	MSE	Energy [J]
POSITION_CONTROL	0.0070	113.45
PID TORQUE_CONTROL	0.0180	100.90
ANN-augmented TORQUE_CONTROL	0.0024	42.44

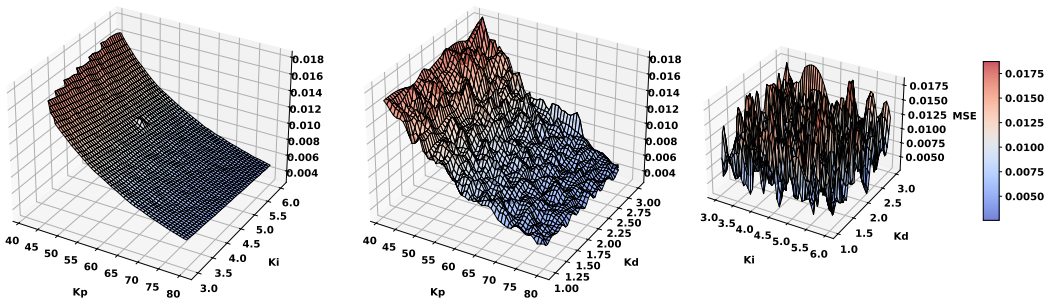

Figure 6.17: MSE as a function of the PID parameters.

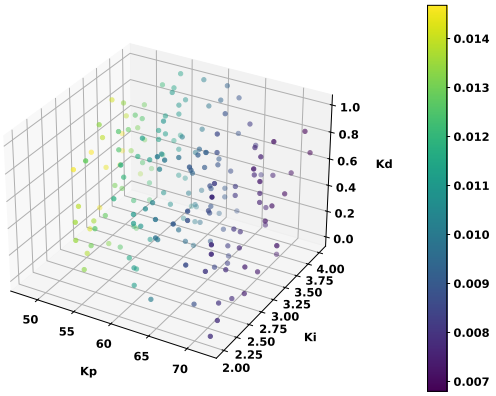

Figure 6.18: 3D Scatterplot of MSE vs PID parameters.

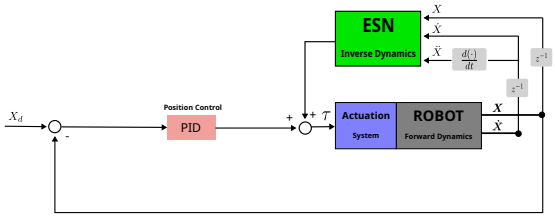

Figure 6.19: ANN-augmented Control Scheme.

Table 6.7: Comparison of Control Modes with extra 4 kg weight

Mode	MSE	Energy [J]
POSITION_CONTROL	0.0070	179.23
PID TORQUE_CONTROL	0.0461	233.86
ANN-augmented TORQUE_CONTROL	0.0089	89.56

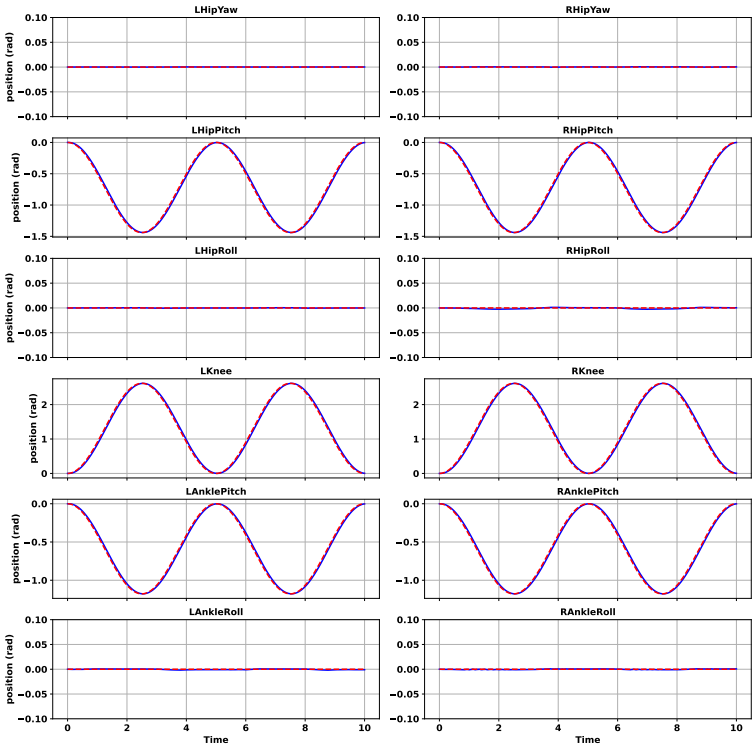

Figure 6.20: Reference vs Actual Position in POSITION_CONTROL mode.

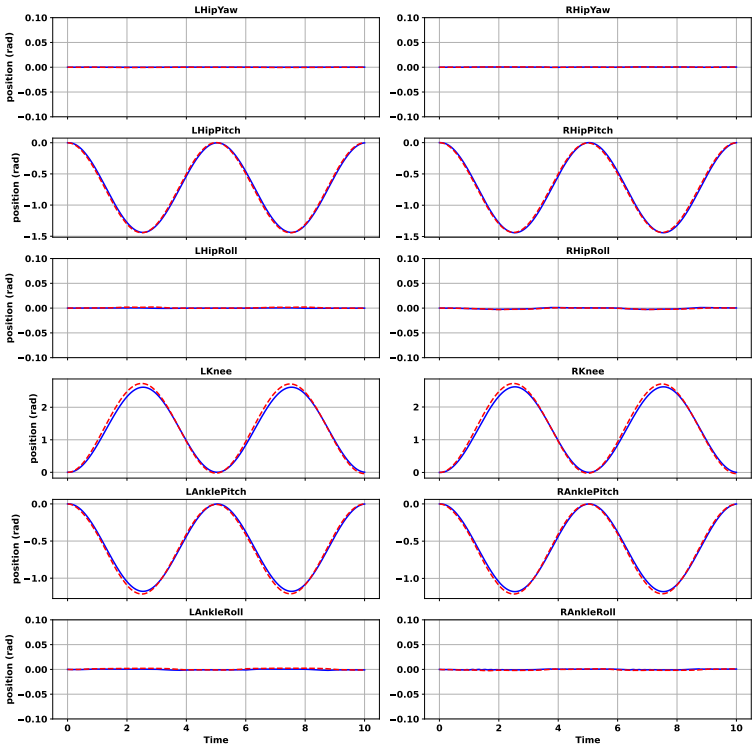

Figure 6.21: Reference vs Actual Position in TORQUE_CONTROL mode.

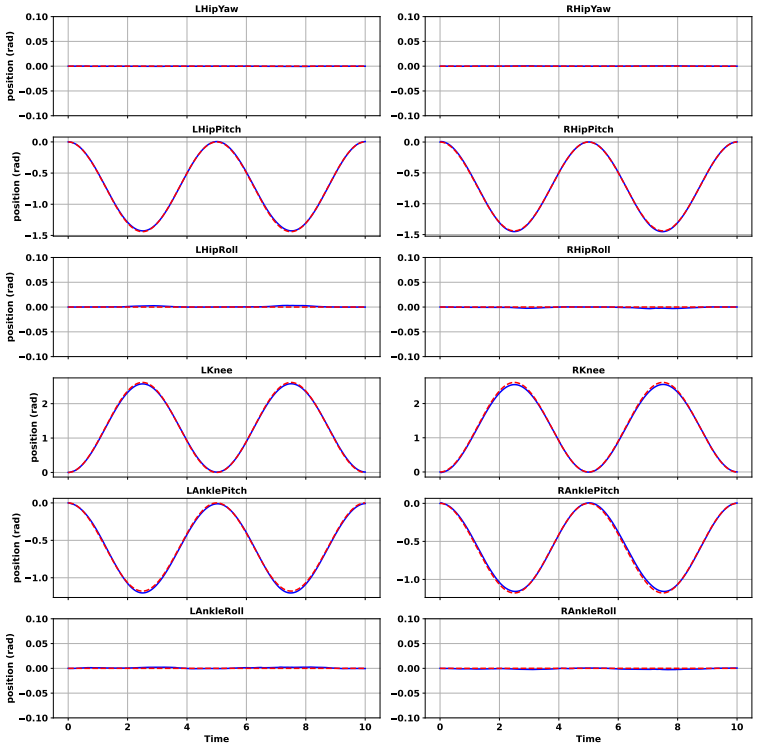

Figure 6.22: Reference vs Actual Position in ANN-augmented TORQUE_CONTROL mode.

BLANK

Chapter 7

Discussion

7.1 The SEA–PEA Elastic Actuator Key Findings

This study aims to define and model a BLDC motor-based actuator integrated with variable stiffness, unidirectional rotary pneumatic element. This architecture allows the actuator to store energy during rotation in one direction and to release it in the opposite direction. Similar research was conducted for an actuator with a linear elastic element by Folgheraiter et al [93]. Such systems have significant potential for robots, particularly for joints performing cyclical movements, such as knee, hip, and ankle pitches. The BLDC motor and gearbox generate a maximum torque of about 23 N·m, which is significantly higher than the pneumatic actuator's 0-3 N·m within 0-6 bar pressure range. This demonstrates that the pneumatic element serves as an auxiliary component that either facilitates or counteracts the rotation generated by the BLDC motor. Depending on the pressure in the chamber, the stiffness of the systems changes from zero to 100 N·m/rad.

The actuator operates in two different modes. The first mode is when the rotation range is within a backlash region. In this case, the pneumatic element behaves as a serial elastic element (SEA). By contrast, outside of the backlash region, the pneumatic actuator becomes a parallel elastic element (PEA). In addition, two more factors influence the dynamics of the systems, namely, the friction force, and gravity impact of the link, which depends on its position and orientation.

The operation of the actuator with the pneumatic element is shown in Fig. 7.1 and can be described as follows. One of the chambers of the pneumatic element is pressurized to create a unidirectional torque. Then the BLDC starts to rotate in the direction opposite to the torque from the pneumatic element until readings from the encoder of the output shaft show that the shaft also starts to rotate. This means that the teeth of the gearbox touch each other, and the system exits the backlash region. If the BLDC starts rotating in the direction opposite to the torque created by the pneumatic element, the system continues to be out of the backlash region. If the BLDC rotates in the same direction, then the system enters the backlash region during the rotation phase. Once the BLDC stops, in this case the torque from the pneumatic element compensates the backlash by making the teeth of the gearbox contact each other. The most interesting behavior can be observed when the BLDC is stationary, for example, it reaches a reference position and then maintains it. The combination of the torque generated by the external load and

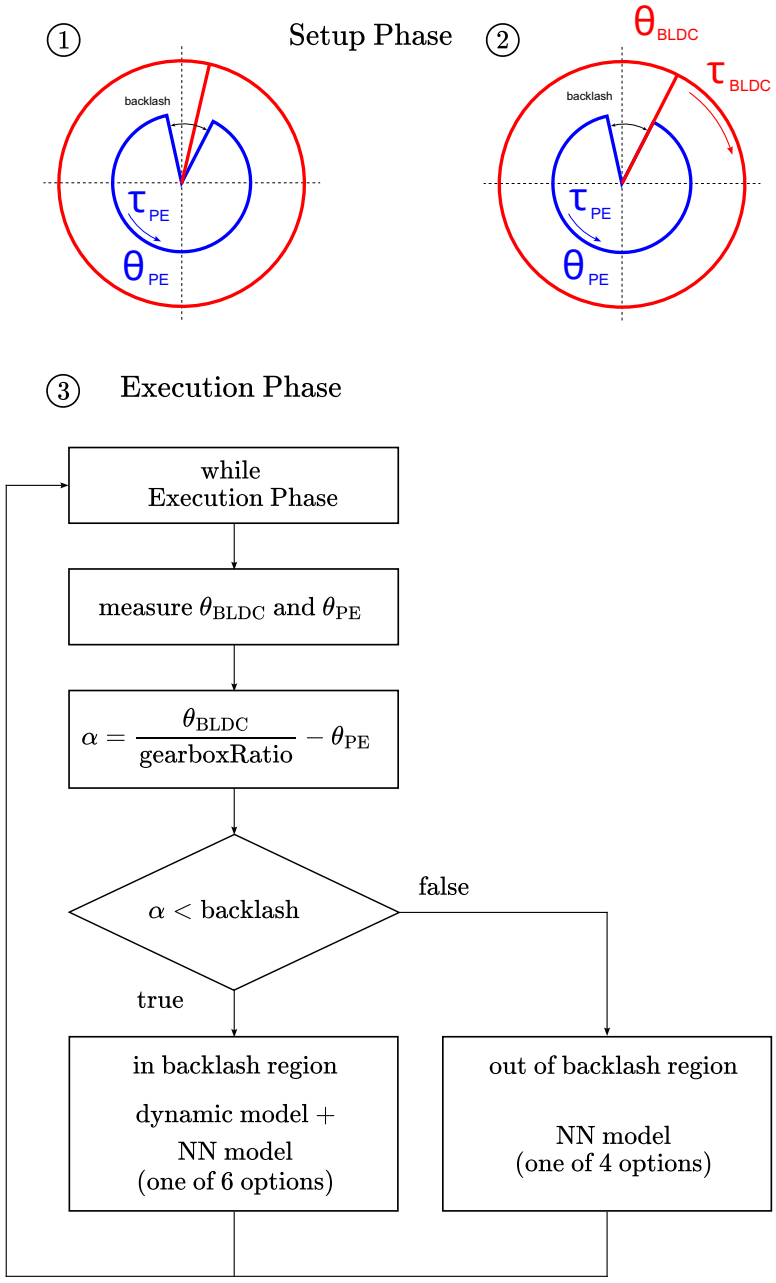

Figure 7.1: Main Algorithm. Setup Phase: (1) - Pressurizing one chamber of PE (pneumatic element), (2) - Applying torque to BLDC until PE also starts to rotate. Execution Phase: (3) - The system constantly measures the angular position of PE and BLDC. If α is less than the backlash value, the system is in the backlash region; otherwise, it is out of this region.

the pneumatic element can rotate the output shaft in the backlash region. Changing the pressure level in the pressurized chamber regulates the stiffness of the system making it more compliant when the pressure decreases.

The ESN is the best-performing neural network for models representing the actuator in the backlash and outer regions. It achieved MSE of 0.0057 and 0.0038, respectively, and demonstrated a superior prediction error compared to other ANN architectures. The ESN efficiently encodes system dynamics into a high-dimensional space using fixed, non-adaptive weights in the reservoir, making it inherently well-suited for capturing short-term, complex nonlinear dynamics without requiring memory gates. In addition to its smallest prediction error, the ESN exhibits lower computational complexity than the other models presented in the Thesis, as only the output layer is trained. For a detailed analysis of its computational complexity, refer to [46].

NNX was utilized in the backlash region and produced an MSE of 0.0076, the second-best result after ESN. This performance can be attributed to its simplicity and focus on local dependencies. LSTM did not perform as well as expected, with MSE equal to 0.0113 and 0.0042 for in the backlash and out of the backlash regions, respectively. This outcome is primarily due to LSTM's ability to capture long-term dependencies, which were less relevant in this system. The system under examination involves short-term interactions between the inputs (external torque, velocity, current, and pressure) and the output (torque). Furthermore, electro-mechanical systems often exhibit relatively fast dynamics, where the effects of past inputs decay quickly due to factors such as damping and inertia. GRU demonstrated performance comparable to that of the LSTM, with an MSE of 0.0122. FFNN was used in both experiments for comparison. In the backlash region it showed MSE values of 0.0127. Out of the backlash region it showed MSE of 0.0053 and 0.0048 for an architecture with a comparable number of trainable parameters and a second one with a much larger number of trainable parameters respectively. These results demonstrate that the FFNN is suitable for modeling sequential data, and achieving a reasonably small prediction error is a matter of using an ANN with more trainable parameters.

As a preliminary validation of the proposed approach, a verification experiment was conducted to test the hypothesis that ANNs contribute to improved actuator performance. The experiment involved tracking a sinusoidal reference trajectory with an amplitude of 30° and a frequency of four cycles over 10 s. This test was not conducted at full scale and was performed without any external load, as it was challenging to ensure consistent external loading conditions across different experimental runs. The pressure in the pneumatic actuator was maintained at a constant level of 4 bar. Three control strategies were evaluated, all of them operating in the standard Torque Control mode provided by the ODrive motor controller: a (1) conventional PID controller, (2) the same PID controller with a linear model-based compensator, and (3) with the ANNs

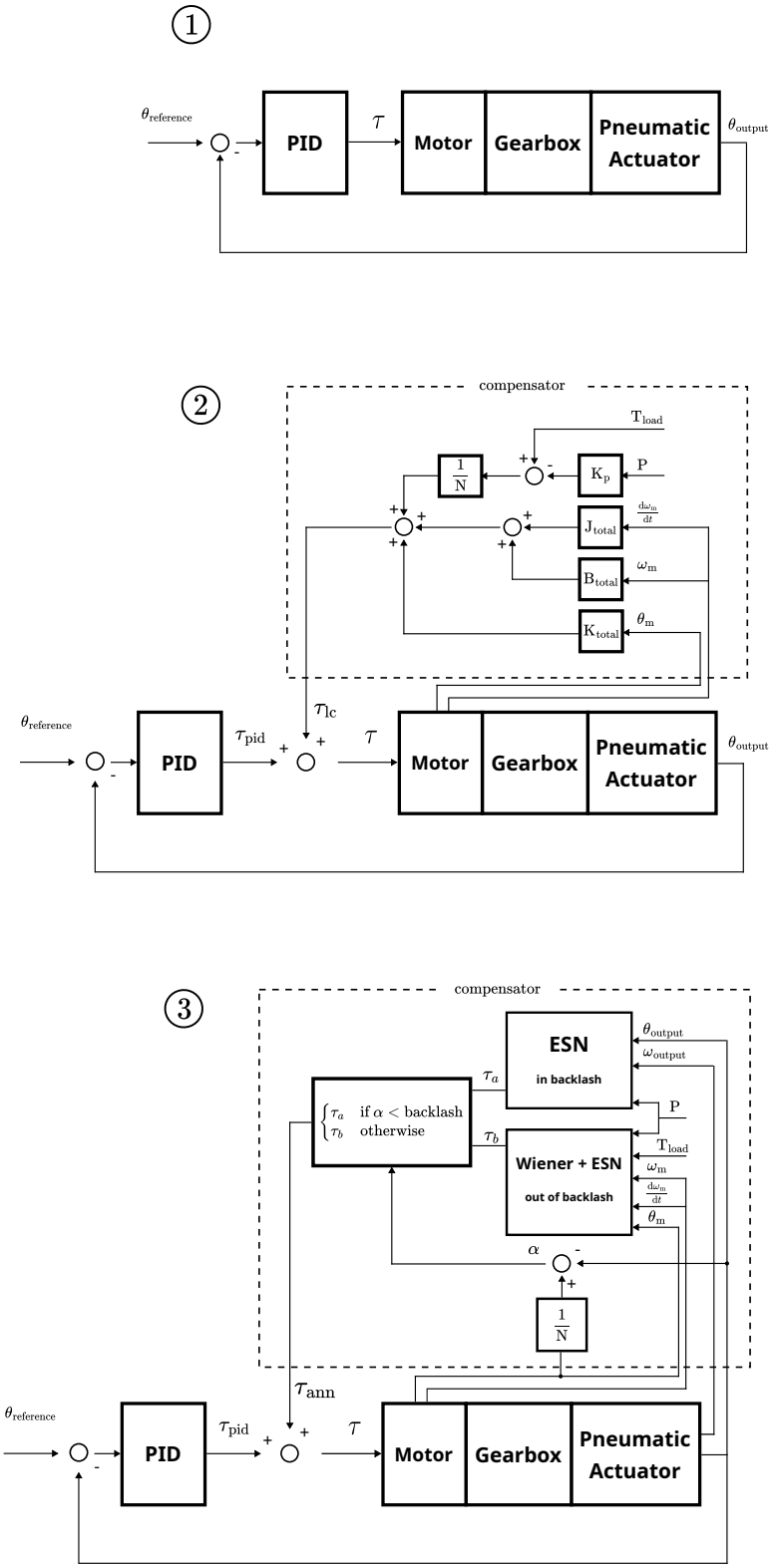

Figure 7.2: Schematic Diagram of Control Algorithms. (1) - Conventional PID control, (2) - PID control with a Linear Compensator, (3) - Neural Network-based Compensator.

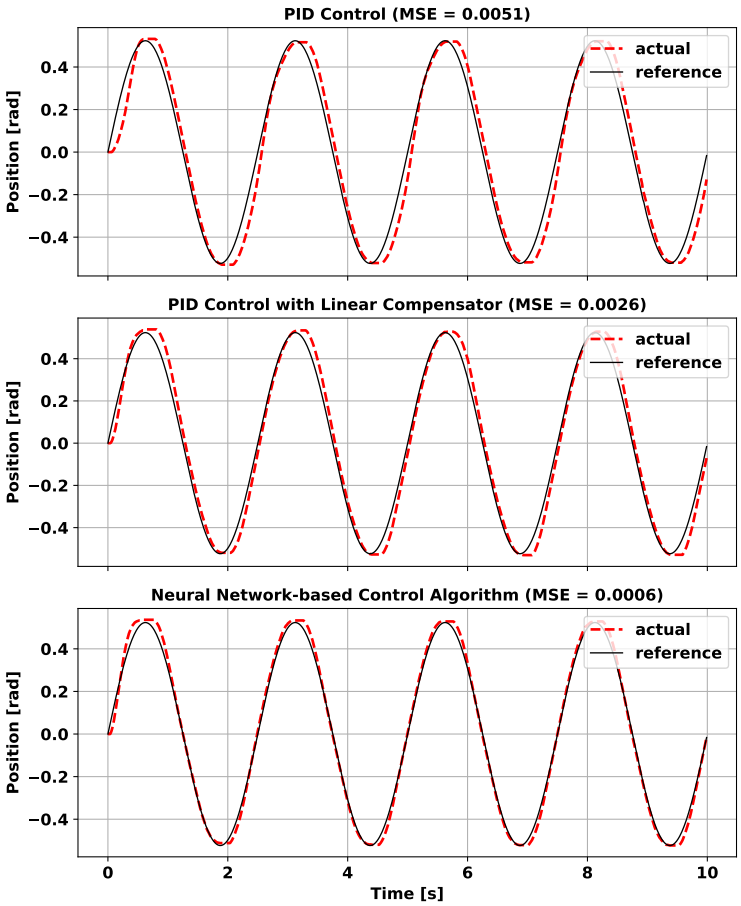

Figure 7.3: Control Strategies Performance Comparison.

acting as a nonlinear model-based compensator.

Schematic diagrams of these control modes are presented in Fig. 7.2, where (1) represents the conventional PID controller, and (2) shows a control algorithm that employs the linear compensator to enhance system performance. This algorithm is based on the equation below, which was derived from Equation (4.12):

$$\begin{aligned} \tau_{\text{lc}} = J_{\text{total}} \frac{d\omega_m}{dt} + B_{\text{total}}\omega_m + K_{\text{total}}\theta_m \\ - \frac{K_p(P_{\text{in}} - P_{\text{out}})}{N} + \frac{T_{\text{load}}}{N}. \end{aligned} \tag{7.1}$$

The Neural Network-based Control algorithm (3), also shown in Fig. 7.2, switches between two ANNs depending on whether the system is operating within the backlash region or outside of it, and uses one of the ANN outputs as a compensation torque value τ_{ann}. Here, $\theta_{\text{reference}}$ is the reference position of the parallel elastic actuator, θ_m, ω_m, $\frac{d\omega_m}{dt}$ are the motor position, velocity and acceleration respectively, θ_{output} and ω_{output} are the parallel elastic actuator position and velocity respectively, P - the pressure, and T_{load} - the torque generated by the external load.

To compare the performance of these control strategies, the MSE between the reference and actual position trajectories was used as the primary evaluation metric. The trajectories and MSE values are shown in Fig. 7.3. The Neural Network-based control algorithm demonstrated an MSE of 0.0006 compared to 0.0051 for the conventional PID control and 0.0026 for the PID control with the linear compensator.

7.2 The Humanoid Robot Key Findings

7.2.1 Benchmark Squat Sequence Experiment with the Prototype

To assess the consistency between the URDF model and the physical prototype, the real robot performed a sequence of two squat movements. The conditions were identical, with one exception: the prototype's Hip Yaw joints were locked at the time due to maintenance and repair. For the sake of accuracy, it should be noted that, although the Hip Yaw joints do not actively participate in the squat sequence, the ODrive motor controllers nonetheless applied current and torque to maintain these joints at a stationary zero position. Nevertheless, for the purpose of the experiment, this difference in the hardware configuration is believed to be acceptable.

The energy consumed by the prototype is in a range comparable to the PyBullet model, as shown in Fig. 7.4. Table 7.1 shows that 134.12 J are required for the prototype versus 179.23 J for the URDF model.

Energy Consumed by a Human During a Squat Sequence

In order to validate the results obtained from both the PyBullet simulations and the real prototype, an estimate of the energy consumed by an adult human was carried out, as presented below.

- **Body mass**: ≈ 70 kg (adult)
- **Squat depth**: ≈ 0.35 m vertical displacement of the center of mass
- **Gravity**: $g = 9.81$ m/s^2
- **Work (mechanical energy)**:

$$W = m \cdot g \cdot h \tag{7.2}$$

Substituting values in Equation (7.2) we get energy consumed:

$$W = 70 \times 9.81 \times 0.35 \approx 240 \text{ J} \tag{7.3}$$

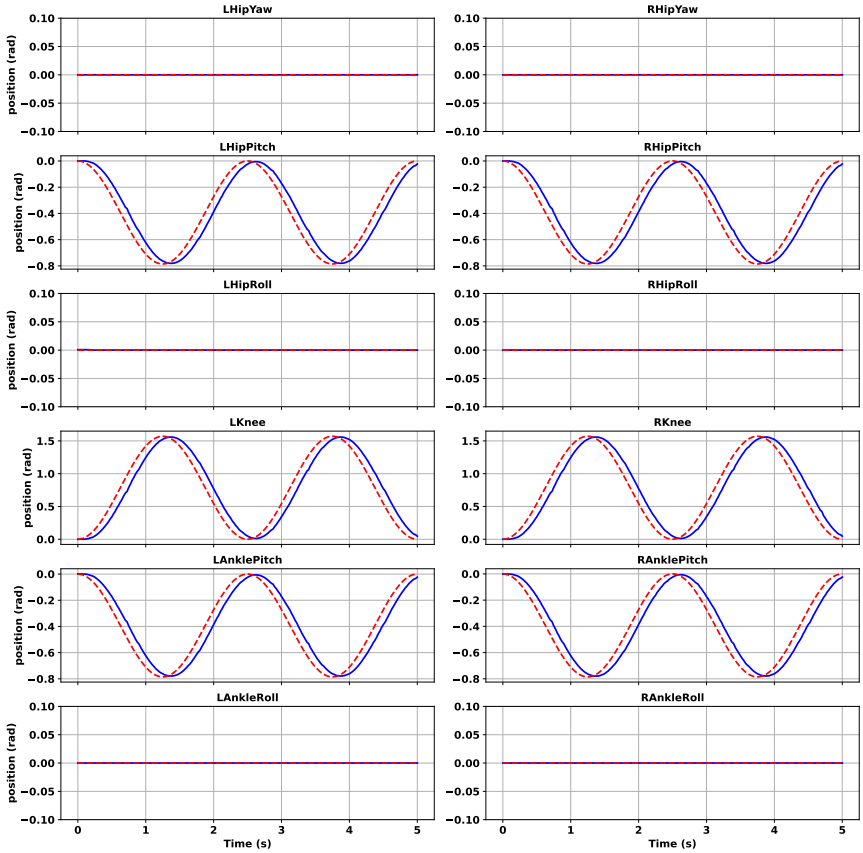

Figure 7.4: Reference vs Actual Position in POSITION_CONTROL mode for the Prototype.

Given that the weight of the prototype is 17.1 kg, we compare the squat energy per unit mass per one squat:

$$E_{\text{robot, per kg}} = \frac{134.12}{2 \cdot 17.1} \approx 3.9 \text{ J/kg} \tag{7.4}$$

$$E_{\text{human, per kg}} = \frac{240}{70} \approx 3.4 \text{ J/kg} \tag{7.5}$$

The energy consumption per weight of both the humanoid prototype and the human was found to be comparable with a difference in a range 15%, as shown in Equations (7.4)–(7.5), thereby demonstrating consistency between the two systems. It should be noted that this comparison only accounts for ideal mechanical work. Humans are not 100% efficient, and muscles consume additional chemical energy due to inefficiencies, with an estimated efficiency of roughly 25%–30% for concentric movements such as squats.

For the standard PID torque control mode, the same gains were initially used; however, it was found that the values were too high. After tuning, the experiment was

conducted with the following parameters:

$$k_p = 25.0, \quad k_i = 1.0, \quad k_d = 0.1.$$

The plot comparing the actual and reference trajectories is shown in Fig. 7.5. In the fully squatted position, the prototype descends slightly lower than intended, whereas in the fully upright position it tracks the reference trajectory more accurately. This behavior can be attributed to the fact that, during the downward squatting motion, the gravity force and the center-of-mass (CoM) motion act in the same direction, effectively amplifying the downward acceleration. A standard PID controller cannot properly compensate for these configuration-dependent gravitational effects, resulting in larger tracking errors.

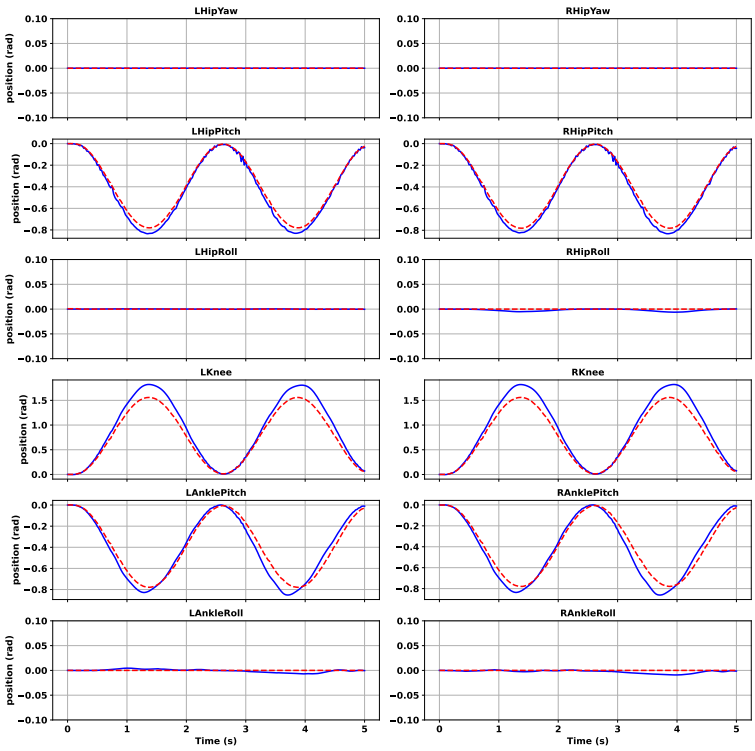

Figure 7.5: Reference vs Actual Position in TORQUE_CONTROL mode for the Prototype.

The scheme of the computed torque control (CTC) algorithm is presented in Fig. 7.6. It employs an ESN to approximate the system's mathematical model and operates using only a PD controller. To reduce noise in the acceleration inputs, the reference acceleration $\ddot{X}_d$ is used instead of the actual acceleration $\ddot{X}$, which in practice is obtained by differentiating the actual velocities provided by the ODrive motor controllers.

Compared to the standard PID approach, the CTC scheme is able to track the trajectory more accurately because it explicitly compensates for nonlinear dynamics,

including gravity and configuration-dependent inertial effects. As a result, even when gravity and the robot's CoM motion act in the same direction during the downward squat and amplify the effective load, the CTC controller can counteract these forces more effectively, which leads to improved tracking performance across the full range of motion.

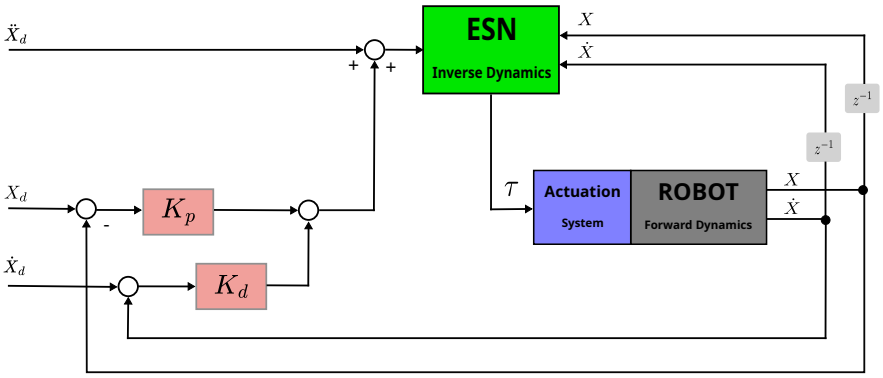

Figure 7.6: Computed Torque Control using ANN.

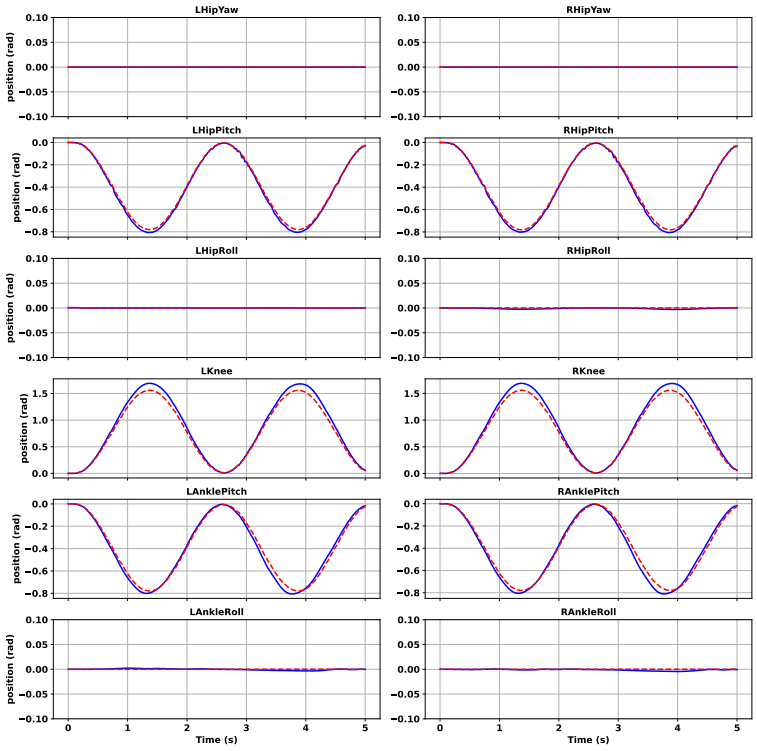

Figure 7.7: Reference vs Actual Position in CTC mode for the Prototype.

The results of adding an additional 4 kg to the robot and repeating the squat sequence under CTC control are shown in Fig. 7.8. The tracking performance exhibits noticeably larger errors compared to the baseline configuration. This degradation occurs because

the ESN model used within the CTC scheme was trained exclusively on data collected from the original 17.1 kg setup. As a result, the ESN does not accurately represent the altered system dynamics caused by the increased mass, particularly the changes in gravitational loading and inertia. Since the CTC controller relies on the ESN to estimate and cancel these nonlinear effects, mismatches in the learned model lead directly to reduced compensation accuracy and, therefore, larger trajectory tracking errors under the heavier configuration.

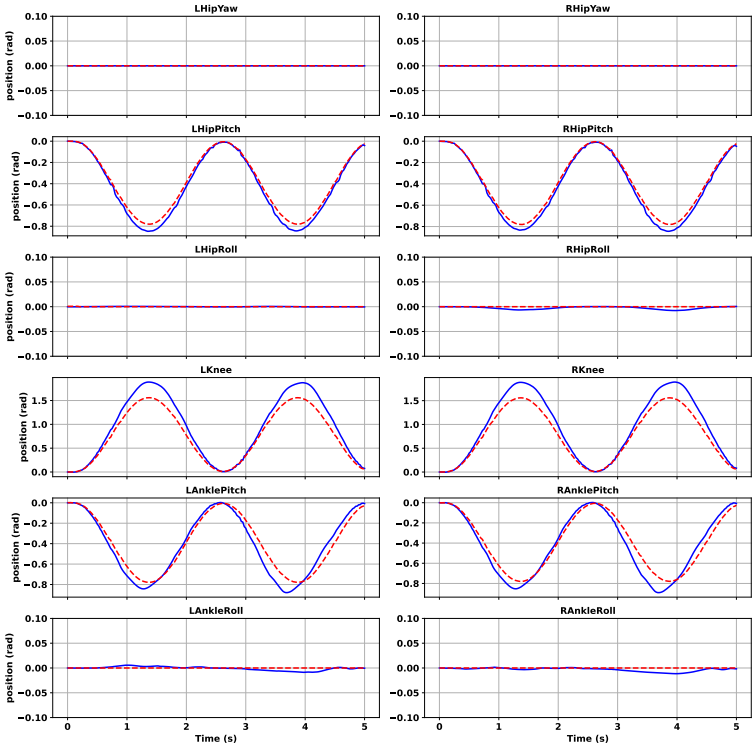

Figure 7.8: Reference vs Actual Position in CTC mode for the Prototype with extra 4 kg.

Another experiment was conducted in which the Echo State Network (ESN) was modified to support incremental learning using a recursive least squares (RLS) algorithm with a forgetting factor λ. This allowed the network's readout layer to be updated continuously as new data were collected, without retraining from scratch. The forgetting factor ensured that recent samples were weighted more heavily, enabling rapid adaptation to new conditions while retaining useful information from previous operation. The diagram is shown in Fig. 7.9.

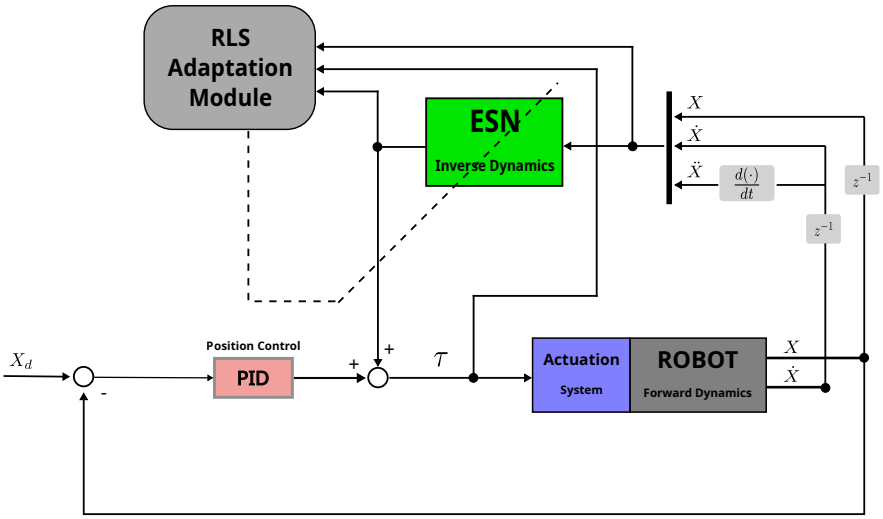

Figure 7.9: ESN adaptation using the RLS algorithm.

Table 7.1: Comparison of Control Modes for the Prototype

Mode	MSE	Energy [J]
POSITION_CONTROL	0.0808	134.12
PID TORQUE_CONTROL	0.0606	117.19
COMPUTED TORQUE_CONTROL	0.0152	76.54
COMPUTED TORQUE_CONTROL with 4 kg	0.0944	154.35
ESN_AUGMENTED_CONTROL with Adaptation Module with 4 kg	0.0276	123.79

7.2.2 Adaptation of the CPC Algorithm Using RLS-Based Sliding-Window RMSE(t)

To quantitatively evaluate the transient and steady-state tracking performance of the proposed CPC adaptation scheme, we compute a time-varying sliding-window root-mean-square error (RMSE) for each joint. The instantaneous joint tracking error is defined as

$$e(t) = \theta_{\text{ref}}(t) - \theta(t), \tag{7.6}$$

where $\theta_{\text{ref}}(t)$ is the reference trajectory generated by the CPC framework and $\theta(t)$ is the measured joint angle.

To capture local tracking performance while suppressing noise, a sliding window of duration W is applied. For each time instant t, the RMSE over the window $[t - W,\, t]$ is computed as

$$\text{RMSE}_W(t) = \sqrt{\frac{1}{|W|} \sum_{i:\, t_i \in [t-W,\, t]} e^2(t_i)}, \tag{7.7}$$

with $W = 1.5$ s, which we found provides a good balance between smoothness and responsiveness.

Figure 7.10 presents the resulting $\mathrm{RMSE}_W(t)$ curves for three controllers: PID, CTC (non-adaptive), and ESN-augmented control with an adaptation module (ESN-adaptive). The prototype performed six squat cycles, with an additional $4\ \mathrm{kg}$ load manually placed on the robot's waist exactly at the end of the third squat when the prototype was in the fully upright position. The moment of load application is explicitly marked in the figure.

Across all joints, the PID controller exhibits the largest transient deviation following the load increase and does not return to its nominal error level because the prototype begins oscillating in the sagittal plane due to errors in joint position tracking. This oscillation further amplifies the tracking error and could have caused the robot to fall if it had not been manually caught. The non-adaptive CTC controller performs better, yet shows a pronounced and persistent RMSE increase after the added mass, confirming its sensitivity to unmodeled dynamics.

In contrast, the ESN-adaptive controller demonstrates the strongest robustness: immediately after the load perturbation, the RMSE rises but subsequently decreases rapidly, settling to a low steady-state plateau. This behavior confirms that the recursive-least-squares parameter adaptation compensates effectively for the mass-induced model mismatch.

Overall, the sliding-window RMSE analysis clearly validates the benefit of incorporating RLS-based adaptation in the CPC loop, particularly in scenarios involving abrupt dynamic changes such as payload variations.

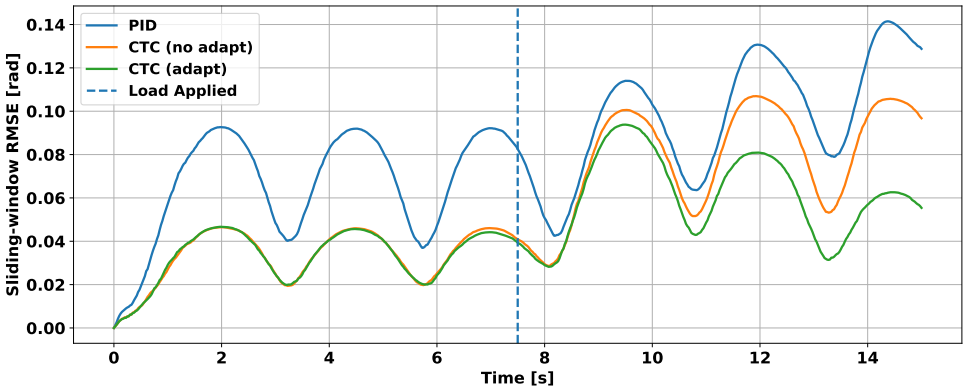

Figure 7.10: Comparison of the three experiments with extra 4 kg weight added during squatting sequence.

7.3 Limitations

While the study provides valuable findings, it does have certain limitations, which can be classified into two categories. The first group is related to the accuracy of the data obtained during the experiments. The required amount of torque can be influenced

by imperfections in the assembly of the setup's elements. The mechanical setup was made using the best available tools and some of the structural elements were produced in an industrial facility with the help of a CNC machine. It was important to align the axes of the gearbox output shaft and the pneumatic element input shaft before welding them in a coupled connection. Additionally, the estimation of the dynamic parameters for the rotating parts of the BLDC motor, the gearbox, and the pneumatic element could have been affected by factors such as the convergence to a suboptimal solution during optimization or noise introduced into the system. The second group is related to the sample size of the data used for training and validating the ANNs and the hyperparameters of these ANNs. The dataset consisted of samples collected from 100 experiments, which may be insufficient for training ANNs with many tunable parameters and could therefore increase the risk of overfitting. In addition, a more accurate URDF model of the humanoid robot, which more precisely reflects its mass and inertia characteristics, needs to be developed.

BLANK

Chapter 8

Conclusion

This study introduces a novel actuator architecture designed to balance the performance, efficiency, and cost in humanoid robots. The actuator combines a low-tolerance planetary gearbox with a pneumatic elastic element, thereby enabling a unique hybrid configuration that offers several key advantages. Unlike precision gearboxes, which reduce backlash but introduce high internal friction and limit backdrivability, the use of a lower-precision gearbox in this system allows for reduced friction losses and improved energy efficiency while maintaining good backdrivability. The pneumatic element introduces controllable stiffness, which, together with the gearbox, enables effective backlash management. This architecture provides a cost-effective solution with enhanced compliance and energy recovery, making it particularly suitable for robotic applications that require safe and adaptive interactions.

Building on this architecture, the research focuses on modeling a highly nonlinear dynamic system in which behavior is affected by backlash, gravity forces, static torques, and impact from a pneumatic elastic element. To address these challenges, this study presents a novel approach to creating a hybrid neural-dynamic model for a complex electromechanical system composed of a BLDC motor, a planetary gearbox, and a pneumatic element. The system operates in two distinct modes. In the first mode, within the backlash region, the pneumatic element functions as a Series Elastic Actuator (SEA). This mode is represented by five ANNs, among which the ESN demonstrated the best performance. Outside the backlash region, it behaves as a Parallel Elastic Actuator (PEA). This part was modeled using seven different options, including a pure linear model, two pure ANNs, and four Wiener schemes that consists of a linear dynamic block and a nonlinear static block represented by different ANNs. The linear dynamic block is described using classical electromechanical theory.

Future work will focus on the practical implementation of a model-based control system, including experimental validation and optimization under various operating conditions. The next phase extends beyond the current preliminary validation experiments by incorporating external loads and varying the stiffness of the system through adjustments in pneumatic actuator pressure. In this framework, it is expected to collect larger datasets and conduct experiments with an increased number of tunable ANN parameters. This experiment shall lead to a reduction in the required computational power and make it possible to implement a control algorithm on an edge computing device. Additionally, it is planned to explore hybrid modeling strategies

that combine neural networks with traditional methods that further reduce prediction error and enhance reliability of system performance. The system is planned to be implemented in the knee and ankle pitch joints of the humanoid robot [94], thereby contributing to improved compliance and operational safety.

On top of implementing safe and adaptive actuators, further experiments with the CTC approach will be conducted. The overall robot architecture will become more complex with the addition of an upper body and two arms. This, in turn, increases the complexity of both the kinematics and dynamics of the system. Consequently, ANN-based control algorithms represent a promising alternative to traditional analytical models of robot dynamics.

8.1 How I Will Leverage This Project In The Future

I plan to continue my research in humanoid robot locomotion, with a particular focus on reducing the risk of damage and injury during collisions with objects and humans. Another key objective is to decrease energy consumption by implementing energy storage and release mechanisms during the robot's cyclical movements. One effective approach to enhancing robot safety in such interactions is through adaptive stiffness control, which allows the system to become more compliant when needed. Using a rotary pneumatic valve as a pneumatic actuator has a few advantages including the capacity to store enough energy in its chamber, smooth and compliant motion due to air compressibility, and a simplified mechanical design with fewer moving parts. I am considering the development of an integrated actuator that combines a BLDC motor, a planetary gearbox, and the pneumatic actuator into a compact unit, aiming to reduce the overall dimensions and improve system efficiency.

In addition, I will continue to further develop the humanoid robot prototype, particularly focusing on training it to walk. This process can greatly benefit from using a CTC algorithm based on ANN implementation combined with reinforcement learning. Once the upper body of the humanoid robot is assembled, the locomotion task becomes more sophisticated and requires advanced control approaches. Experimental results have also demonstrated that a sampling rate of 40–82 Hz may be insufficient for successful operation and stable control of the robot. Therefore, the data exchange protocol must be revised to achieve a sampling rate of 150–200 Hz.

As a logical step toward unifying the two main research direction, SPEA actuator development and humanoid walking control, I plan to integrate SPEA actuators into several joints of the humanoid robot prototype. The overall kinematic model of the prototype, including the upper body, arms, and head, is shown in Fig. 8.1. The lower body kinematic structure is overlaid on an image of the existing NU 4.7 prototype. In the future implementation, the same general arrangement will be preserved, with SPEA

actuators replacing the standard actuators in the knee and ankle pitch joints.

These joints were selected because they play an important role in the walking cycle, and introducing two levels of compliant actuation will allow the system to function as an effective shock mitigation mechanism. This compliance reduces the impact forces when the robot's feet contact the ground, which directly contributes to gait stability and decreases the transmission of vibrations and shocks to the upper body and head. These locations are planned for the installation of the Jetson Orin Nano and the computer vision systems.

Once the concept is proven to be viable, especially considering the challenges of locating an additional power source such as an air compressor and the required pneumatic infrastructure including pipes, valves, and pressure sensors, the implementation of SPEA actuators will be extended to the torso and arms. Introducing a SPEA actuator in the torso pitch joint will add an additional layer of compliance between the walking surface and the robot's head, where the LiDAR and cameras are located.

Similarly, the integration of SPEA actuators in the shoulder joints and wrists will further enhance the compliance and safety of the upper body. In the shoulder joints, SPEA actuation will provide controlled flexibility during arm swing and interaction with external objects, reducing the transmission of sudden forces to the torso and improving overall dynamic stability. In the wrists, the compliant behavior of SPEA will enable safer contact handling, better shock absorption during manipulation tasks, and improved robustness when dealing with unstructured environments. Together, these additions will contribute to smoother upper limb motion and greater resilience of the sensing and computing hardware housed in the upper body.

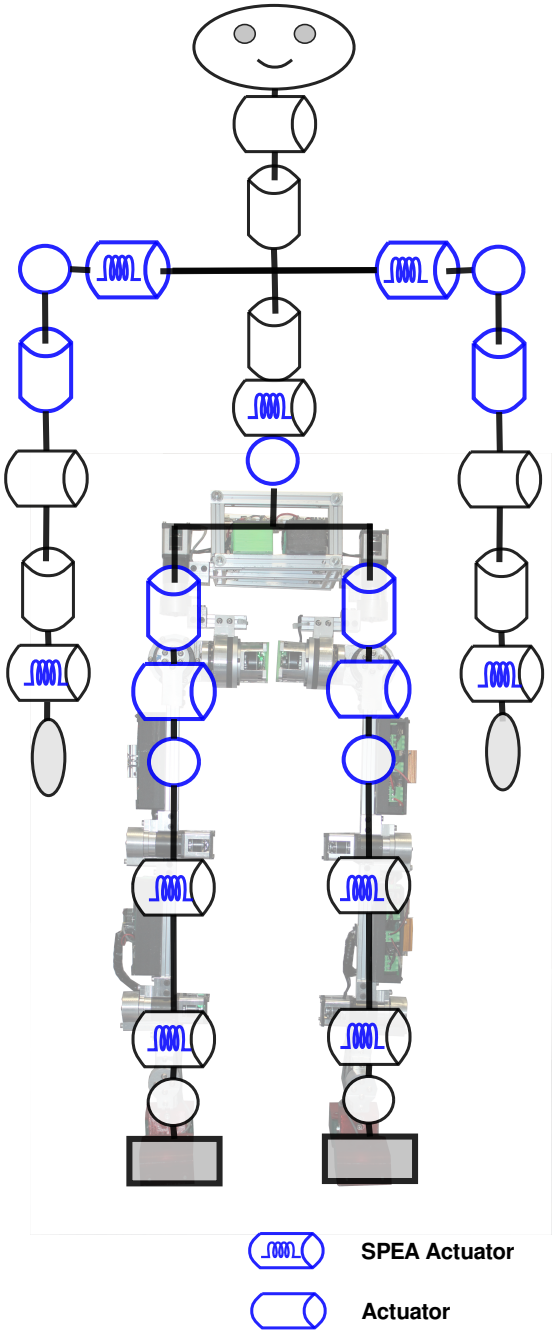

Figure 8.1: Future Implementation of SPEA Actuators in the Humanoid Robot Prototype.

Bibliography

[1] K. D. Truong, A. K. L. Luu, N. P. Tong, V. T. Duong, H. H. Nguyen, and T. T. Nguyen, "Design of series elastic actuator applied for humanoid," in *2020 International Conference on Advanced Mechatronic Systems (ICAMechS)*, pp. 23–28, 2020.

[2] S.-w. Le and G.-j. Li, "Study of resolution for harmonic drives controller with friction in precision robotic system," in *2011 Eighth International Conference on Fuzzy Systems and Knowledge Discovery (FSKD)*, vol. 4, pp. 2404–2407, 2011.

[3] G. Kenneally, A. De, and D. E. Koditschek, "Design principles for a family of direct-drive legged robots," *IEEE Robotics and Automation Letters*, vol. 1, no. 2, pp. 900–907, 2016.

[4] P. Beckerle, T. Verstraten, G. Mathijssen, R. Furnémont, B. Vanderborght, and D. Lefeber, "Series and parallel elastic actuation: Influence of operating positions on design and control," *IEEE/ASME Transactions on Mechatronics*, vol. 22, no. 1, pp. 521–529, 2017.

[5] B. Penzlin, L. Bergmann, Y. Li, L. Ji, S. Leonhardt, and C. Ngo, "Design and first operation of an active lower limb exoskeleton with parallel elastic actuation," *Actuators*, vol. 10, no. 4, 2021.

[6] A. Kakogawa, S. Jeon, and S. Ma, "Stiffness design of a resonance-based planar snake robot with parallel elastic actuators," *IEEE Robotics and Automation Letters*, vol. 3, no. 2, pp. 1284–1291, 2018.

[7] J.-Y. Chang, H.-P. Yang, C.-S. Jang, H.-T. Yeh, and T.-M. Liu, "A new compliant actuation: Electromagnetic spring uses in parallel elastic actuator," in *2016 16th International Conference on Control, Automation and Systems (ICCAS)*, pp. 1549–1552, 2016.

[8] C. W. Mathews and D. J. Braun, "Design of a variable stiffness spring with human-selectable stiffness," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 7385–7390, 2023.

[9] C. W. Mathews and D. J. Braun, "Parallel variable stiffness actuators," in *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 8225–8231, 2021.

[10] X. Qu, D. Cao, Q. Wang, and Y. Li, "Design and research of flexible joint with variable stiffness based on torsion spring," in *2019 IEEE International Conference on Mechatronics and Automation (ICMA)*, pp. 325–329, 2019.

[11] Y. Xu, K. Guo, J. Li, and Y. Li, "A novel rotational actuator with variable stiffness using s-shaped springs," *IEEE/ASME Transactions on Mechatronics*, vol. 26, no. 4, pp. 2249–2260, 2021.

[12] X. Zang, J. Song, Y. Liu, and X. Zhou, "Design of a pneumatic variable stiffness ankle joint," in *2018 IEEE International Conference on Intelligence and Safety for Robotics (ISR)*, pp. 291–296, 2018.

[13] Y. Sun, P. Tang, D. Dong, J. Zheng, X. Chen, and L. Bai, "Modeling and experimental evaluation of a pneumatic variable stiffness actuator," *IEEE/ASME Transactions on Mechatronics*, vol. 27, no. 5, pp. 2462–2473, 2022.

[14] Y. Yamamoto, N. Matsunaga, and H. Okajima, "Robust variable stiffness control of mckibben type pneumatic artificial muscle arm by using multiple model error compensators," in *2017 17th International Conference on Control, Automation and Systems (ICCAS)*, pp. 957–962, 2017.

[15] G. Pratt and M. Williamson, "Series elastic actuators," in *Proceedings 1995 IEEE/RSJ International Conference on Intelligent Robots and Systems. Human Robot Interaction and Cooperative Robots*, vol. 1, pp. 399–406 vol.1, 1995.

[16] S. Toxiri, A. Calanca, J. Ortiz, P. Fiorini, and D. G. Caldwell, "A parallel-elastic actuator for a torque-controlled back-support exoskeleton," *IEEE Robotics and Automation Letters*, vol. 3, no. 1, pp. 492–499, 2018.

[17] N. G. Tsagarakis, I. Sardellitti, and D. G. Caldwell, "A new variable stiffness actuator (compact-vsa): Design and modelling," in *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 378–383, 2011.

[18] A. Jafari, N. G. Tsagarakis, B. Vanderborght, and D. G. Caldwell, "A novel actuator with adjustable stiffness (awas)," in *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 4201–4206, 2010.

[19] S. Lan and Z. Song, "Design of a new nonlinear stiffness compliant actuator and its error compensation method," *Journal of Robotics*, vol. 2016, no. 1, p. 7326905, 2016.

[20] S. Wolf, G. Grioli, O. Eiberger, W. Friedl, M. Grebenstein, H. Höppner, E. Burdet, D. G. Caldwell, R. Carloni, M. G. Catalano, D. Lefeber, S. Stramigioli, N. Tsagarakis, M. Van Damme, R. Van Ham, B. Vanderborght, L. C. Visser,

A. Bicchi, and A. Albu-Schäffer, "Variable stiffness actuators: Review on design and components," *IEEE/ASME Transactions on Mechatronics*, vol. 21, no. 5, pp. 2418–2430, 2016.

[21] Y. Zhu, J. Yang, H. Jin, X. Zang, and J. Zhao, "Design and evaluation of a parallel-series elastic actuator for lower limb exoskeletons," in *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1335–1340, 2014.

[22] S. Crispel, P. L. García, E. Saerens, A. Varadharajan, T. Verstraten, B. Vanderborght, and D. Lefeber, "A novel wolfrom-based gearbox for robotic actuators," *IEEE/ASME Transactions on Mechatronics*, vol. 26, no. 4, pp. 1980–1988, 2021.

[23] J. Terfurth and N. Parspour, "Integrated planetary gear joint actuator concept for wearable and industrial robotic applications," in *2019 Wearable Robotics Association Conference (WearRAcon)*, pp. 28–33, 2019.

[24] H. Matsuki, K. Nagano, and Y. Fujimoto, "Bilateral drive gear—a highly backdrivable reduction gearbox for robotic actuators," *IEEE/ASME Transactions on Mechatronics*, vol. 24, no. 6, pp. 2661–2673, 2019.

[25] X. Shen and M. Goldfarb, "Simultaneous Force and Stiffness Control of a Pneumatic Actuator," *Journal of Dynamic Systems, Measurement, and Control*, vol. 129, pp. 425–434, 01 2007.

[26] Y. Sun, P. Tang, D. Dong, J. Zheng, X. Chen, and L. Bai, "Modeling and experimental evaluation of a pneumatic variable stiffness actuator," *IEEE/ASME Transactions on Mechatronics*, vol. 27, no. 5, pp. 2462–2473, 2022.

[27] X. Shen and M. Goldfarb, "Independent stiffness and force control of pneumatic actuators for contact stability during robot manipulation," in *Proceedings of the 2005 IEEE International Conference on Robotics and Automation*, pp. 2697–2702, 2005.

[28] N. Herzig, R. Moreau, A. Leleve, and M. T. Pham, "Stiffness control of pneumatic actuators to simulate human tissues behavior on medical haptic simulators," in *2016 IEEE International Conference on Advanced Intelligent Mechatronics (AIM)*, pp. 1591–1597, 2016.

[29] M. Sorli, L. Gastaldi, E. Codina, and S. de las Heras, "Dynamic analysis of pneumatic actuators," *Simulation Practice and Theory*, vol. 7, no. 5, pp. 589–602, 1999.

[30] M. Dağdelen and M. Sarıgeçili, "An experimental method for estimating combined friction torque in vane type pneumatic semi rotary actuators," *Sakarya University Journal of Science*, vol. 24, no. 6, p. 1272–1283, 2020.

[31] S. M. A. Cruz, A. Stefani, F. Filippetti, and A. J. M. Cardoso, "A new model-based technique for the diagnosis of rotor faults in rfoc induction motor drives," *IEEE Transactions on Industrial Electronics*, vol. 55, no. 12, pp. 4218–4228, 2008.

[32] S. Saxena and Y. V. Hote, "Advances in internal model control technique: A review and future prospects," *IETE Technical Review*, vol. 29, no. 6, pp. 461–472, 2012.

[33] R. Luo, Z. Hu, M. Liu, L. Du, S. Bao, and J. Yuan, "Adaptive neural computed torque control for robot joints with asymmetric friction model," *IEEE Robotics and Automation Letters*, vol. 10, no. 1, pp. 732–739, 2025.

[34] F.-J. Lin, M.-S. Huang, C.-Y. Hung, and Y.-C. Chien, "Intelligent computed torque control with recurrent legendre fuzzy neural network for permanent-magnet assisted synchronous reluctance motor," *IEEE Access*, vol. 11, pp. 54017–54028, 2023.

[35] G. Cheng, W. Yu, and J.-g. Hu, "Improving the performance of motor drive servo systems via composite nonlinear control," *CES Transactions on Electrical Machines and Systems*, vol. 2, no. 4, pp. 399–408, 2018.

[36] L. B. Armenio, E. Terzi, M. Farina, and R. Scattolini, "Model predictive control design for dynamical systems learned by echo state networks," *IEEE Control Systems Letters*, vol. 3, no. 4, pp. 1044–1049, 2019.

[37] B. Li, L. Yan, X. Gao, and C. Gerada, "Robust adaptive control based on variable boundary for a twin-motor cable-driven system," *IEEE Transactions on Industrial Electronics*, vol. 69, no. 7, pp. 7054–7063, 2022.

[38] J. Zhai and Z. Li, "Fast-exponential sliding mode control of robotic manipulator with super-twisting method," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 69, no. 2, pp. 489–493, 2022.

[39] M. Sadiq, R. Hayat, K. Zeb, A. Al-Durra, and Z. Ullah, "Robust feedback linearization based disturbance observer control of quadrotor uav," *IEEE Access*, vol. 12, pp. 17966–17981, 2024.

[40] M. M. Polycarpou and P. A. Ioannou, *Identification and control of nonlinear systems using neural network models: Design and stability analysis*. Citeseer, 1991.

[41] K. S. Narendra and K. Parthasarathy, "Neural networks and dynamical systems," *International Journal of Approximate Reasoning*, vol. 6, no. 2, pp. 109–131, 1992.

[42] E. B. Kosmatopoulos, M. M. Polycarpou, M. A. Christodoulou, and P. A. Ioannou, "High-order neural network structures for identification of dynamical systems," *IEEE transactions on Neural Networks*, vol. 6, no. 2, pp. 422–431, 1995.

[43] E. B. Kosmatopoulos, P. Ioannou, and M. Christodoulou, "Identification of nonlinear systems using new dynamic neural network structures," in *[1992] Proceedings of the 31st IEEE Conference on Decision and Control*, pp. 20–25, IEEE, 1992.

[44] A. M. Shaw, F. J. Doyle III, and J. S. Schwaber, "A dynamic neural network approach to nonlinear process modeling," *Computers & chemical engineering*, vol. 21, no. 4, pp. 371–385, 1997.

[45] 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.

[46] M. Folgheraiter, A. Yskak, and S. Yessirkepov, "One-shot bipedal robot dynamics identification with a reservoir-based rnn," *IEEE Access*, vol. 11, pp. 50180–50194, 2023.

[47] N. Todorovic and P. Klan, "State of the art in nonlinear dynamical system identification using artificial neural networks," in *2006 8th Seminar on Neural Network Applications in Electrical Engineering*, pp. 103–108, 2006.

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

[49] S. Lu and T. Basar, "Robust nonlinear system identification using neural-network models," *IEEE Transactions on Neural Networks*, vol. 9, no. 3, pp. 407–429, 1998.

[50] K. J. Nidhil Wilfred, S. Sreeraj, B. Vijay, and V. Bagyaveereswaran, "System identification using artificial neural network," in *2015 International Conference on Circuits, Power and Computing Technologies [ICCPCT-2015]*, pp. 1–4, 2015.

[51] P. Gautam, "System identification of nonlinear inverted pendulum using artificial neural network," in *2016 International Conference on Recent Advances and Innovations in Engineering (ICRAIE)*, pp. 1–5, 2016.

[52] M. Schüssler, T. Münker, and O. Nelles, "Deep recurrent neural networks for nonlinear system identification," in *2019 IEEE Symposium Series on Computational Intelligence (SSCI)*, pp. 448–454, 2019.

[53] B. Zhang, D. J. Miller, and Y. Wang, "Nonlinear system modeling with random matrices: Echo state networks revisited," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 23, no. 1, pp. 175–182, 2012.

[54] J. Qiao, F. Li, H. Han, and W. Li, "Growing echo-state network with multiple subreservoirs," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 28, no. 2, pp. 391–404, 2017.

[55] C. Hartland and N. Bredeche, "Using echo state networks for robot navigation behavior acquisition," in *2007 IEEE International Conference on Robotics and Biomimetics (ROBIO)*, pp. 201–206, 2007.

[56] P. Arena, L. Patané, and A. G. Spinosa, "Structural and input reduction in a esn for robotic navigation tasks," in *2019 IEEE International Conference on Systems, Man and Cybernetics (SMC)*, pp. 3531–3536, 2019.

[57] Y. Xia, B. Jelfs, M. M. Van Hulle, J. C. Principe, and D. P. Mandic, "An augmented echo state network for nonlinear adaptive filtering of complex noncircular signals," *IEEE Transactions on Neural Networks*, vol. 22, no. 1, pp. 74–83, 2011.

[58] A. Shamsah, K. Agarwal, S. Kousik, and Y. Zhao, "Real-time model predictive control with zonotope-based neural networks for bipedal social navigation," in *2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 13741–13748, 2024.

[59] Y. Yang, Z. Ding, and F. Li, "Parameter identification of wiener model based on lstm neural network," in *2023 42nd Chinese Control Conference (CCC)*, pp. 1315–1319, 2023.

[60] I. M. Iqbal and N. Aziz, "Comparison of various wiener model identification approach in modelling nonlinear process," in *2011 3rd Conference on Data Mining and Optimization (DMO)*, pp. 134–140, 2011.

[61] C. Qi, J. Lin, Y. Wu, and F. Gao, "A wiener model identification for creep and vibration linear and hysteresis nonlinear dynamics of piezoelectric actuator," *IEEE Sensors Journal*, vol. 21, no. 24, pp. 27570–27581, 2021.

[62] Z. Zhang, D. Zhang, H. Zheng, T. Huang, and Y. Xie, "Identification of a precision motion stage based on the hammerstein-wiener model," in *2019 Chinese Control Conference (CCC)*, pp. 1637–1642, 2019.

[63] M. Folgheraiter, "A combined b-spline-neural-network and arx model for online identification of nonlinear dynamic actuation systems," *Neurocomputing*, vol. 175, pp. 433–442, 2016.

[64] N.-H. Kim, J.-M. Kim, O. Khatib, and D. Shin, "Design optimization of hybrid actuation combining macro-mini actuators," *International Journal of Precision Engineering and Manufacturing*, vol. 18, pp. 519–527, 2017.

[65] B. Rouzbeh, G. M. Bone, G. Ashby, and E. Li, "Design, implementation and control of an improved hybrid pneumatic-electric actuator for robot arms," *IEEE Access*, vol. 7, pp. 14699–14713, 2019.

[66] Y. Nakata and T. Noda, "Fusion hybrid linear actuator: Concept and disturbance resistance evaluation," *IEEE/ASME Transactions on Mechatronics*, vol. 28, no. 4, pp. 2167–2177, 2023.

[67] M. Folgheraiter, S. Yessirkepov, and T. Umurzakov, "Nu-biped-4.5: A lightweight and low-prototyping-cost full-size bipedal robot," *Robotics*, vol. 13, no. 1, 2024.

[68] R. E. Kalman, "A new approach to linear filtering and prediction problems," 1960.

[69] M. M. Sotaro Katayama and Y. Tazaki, "Model predictive control of legged and humanoid robots: models and algorithms," *Advanced Robotics*, vol. 37, no. 5, pp. 298–315, 2023.

[70] E. Dantec, M. Naveau, P. Fernbach, N. Villa, G. Saurel, O. Stasse, M. Taix, and N. Mansard, "Whole-body model predictive control for biped locomotion on a torque-controlled humanoid robot," in *2022 IEEE-RAS 21st International Conference on Humanoid Robots (Humanoids)*, pp. 638–644, 2022.

[71] J. A. Castano, C. Zhou, Z. Li, and N. Tsagarakis, "Robust model predictive control for humanoids standing balancing," in *2016 international conference on advanced robotics and mechatronics (icarm)*, pp. 147–152, IEEE, 2016.

[72] A.-N. Sharkawy and P. Koustoumpardis, "Dynamics and computed-torque control of a 2-dof manipulator: Mathematical analysis," *International Journal of Advanced Science and Technology*, vol. 28, no. 12, pp. 201–212, 2019.

[73] B. Wang, Y. Wang, J. Huang, Y. Zeng, X. Liu, and K. Zhou, "Computed torque control and force analysis for mechanical leg with variable rotation axis powered by servo pneumatic muscle," *ISA Transactions*, vol. 140, pp. 385–401, 2023.

[74] R. Beerens, H. Nijmeijer, and P. van Zutven, "Adaptive control on bipedal humanoid robots," 2014.

[75] A. T. M. Amin, A. H. A. Rahim, and C. Y. Low, "Adaptive controller algorithm for 2-dof humanoid robot arm," *Procedia Technology*, vol. 15, pp. 765–774, 2014. 2nd International Conference on System-Integrated Intelligence: Challenges for Product and Production Engineering.

[76] J. Yu, S. Zhang, A. Wang, W. Li, Z. Ma, and X. Yue, "Humanoid control of lower limb exoskeleton robot based on human gait data with sliding mode neural network," *CAAI Transactions on Intelligence Technology*, vol. 7, no. 4, pp. 606–616, 2022.

[77] D. C.-O. I. S. M. Sanchez-Magos, M. Ballesteros and I. Chairez, "Terminal sliding-mode control of virtual humanoid robot with joint restrictions walking on stepping objects," *Cybernetics and Systems*, vol. 51, no. 4, pp. 402–425, 2020.

[78] E. Karadogan and R. L. Williams II, "The robotic lumbar spine: Dynamics and feedback linearization control," *Computational and Mathematical Methods in Medicine*, vol. 2013, no. 1, p. 985248, 2013.

[79] C. M. C. O. Valle, R. Tanscheit, and L. A. F. Mendoza, "Computed-torque control of a simulated bipedal robot with locomotion by reinforcement learning," in *2016 IEEE Latin American Conference on Computational Intelligence (LA-CCI)*, pp. 1–6, 2016.

[80] N. T. Alberto, M. Mistry, and F. Stulp, "Computed torque control with variable gains through gaussian process regression," in *2014 IEEE-RAS International Conference on Humanoid Robots*, pp. 212–217, 2014.

[81] J. D. Gamba, A. C. Leite, and R. Featherstone, "Robust balancing control of a spring-legged robot based on a high-order sliding mode observer," in *2020 IEEE-RAS 20th International Conference on Humanoid Robots (Humanoids)*, pp. 384–391, 2021.

[82] K. Yamamoto and Y. Nakamura, "Switching feedback controllers based on the maximal cpi sets for stabilization of humanoid robots," in *2009 9th IEEE-RAS International Conference on Humanoid Robots*, pp. 549–554, 2009.

[83] S. M. Ahmed, C. M. Chew, and B. Tian, "Standing posture modeling and control for a humanoid robot," in *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 4152–4157, 2013.

[84] F. F. A. Silva and B. V. Adorno, "Whole-body control of a mobile manipulator using feedback linearization based on dual quaternions," in *2016 XIII Latin American Robotics Symposium and IV Brazilian Robotics Symposium (LARS/SBR)*, pp. 293–298, 2016.

[85] J. de la Casa Cárdenas, A. S. García, S. S. Martínez, J. G. García, and J. G. Ortega, "Model predictive position/force control of an anthropomorphic robotic arm," in *2015 IEEE International Conference on Industrial Technology (ICIT)*, pp. 326–331, 2015.

[86] C. Khazoom and S. Kim, "Humanoid arm motion planning for improved disturbance recovery using model hierarchy predictive control," in *2022 International Conference on Robotics and Automation (ICRA)*, pp. 6607–6613, 2022.

[87] Z.-S. Hou and Z. Wang, "From model-based control to data-driven control: Survey, classification and perspective," *Information Sciences*, vol. 235, pp. 3–35, 2013.

[88] M. Folgheraiter, A. Yskak, and S. Yessirkepov, "One-shot bipedal robot dynamics identification with a reservoir-based rnn," *IEEE Access*, vol. 11, pp. 50180–50194, 2023.

[89] S. Bhattacharya, T. K. Maiti, S. Dutta, A. Luo, M. Miura-Mattausch, and H. J. Mattausch, "System simulation for robot control based on ai approach," in *2018 14th IEEE International Conference on Solid-State and Integrated Circuit Technology (ICSICT)*, pp. 1–4, 2018.

[90] D. Katic, A. Rodic, and E. Jose Bayro-Corrochanoy, "Hybrid control algorithm for humanoid robots walking based on episodic reinforcement learning," in *World Automation Congress 2012*, pp. 1–6, 2012.

[91] M. Folgheraiter, S. Yessirkepov, T. Umurzakov, and R. Korabay, "Nu-biped-4 a lightweight and low-power consumption full-size bipedal robot," in *2023 9th International Conference on Automation, Robotics and Applications (ICARA)*, pp. 38–43, 2023.

[92] ODrive Robotics, "Odrive: High performance motor controller for robotics." https://odriverobotics.com/, 2024. Accessed: 2024-09-11.

[93] M. Folgheraiter, J. de Gea, B. Bongardt, J. Albiez, and F. Kirchner, "Bio-inspired control of an arm exoskeleton joint with active-compliant actuation system," *Applied Bionics and Biomechanics*, vol. 6, no. 2, p. 542562, 2009.

[94] T. Umurzakov, S. Yessirkepov, and M. Folgheraiter, "Fast prototyping and testing of a new full scale bipedal robot," in *2022 7th International Conference on Robotics and Automation Engineering (ICRAE)*, pp. 215–221, 2022.