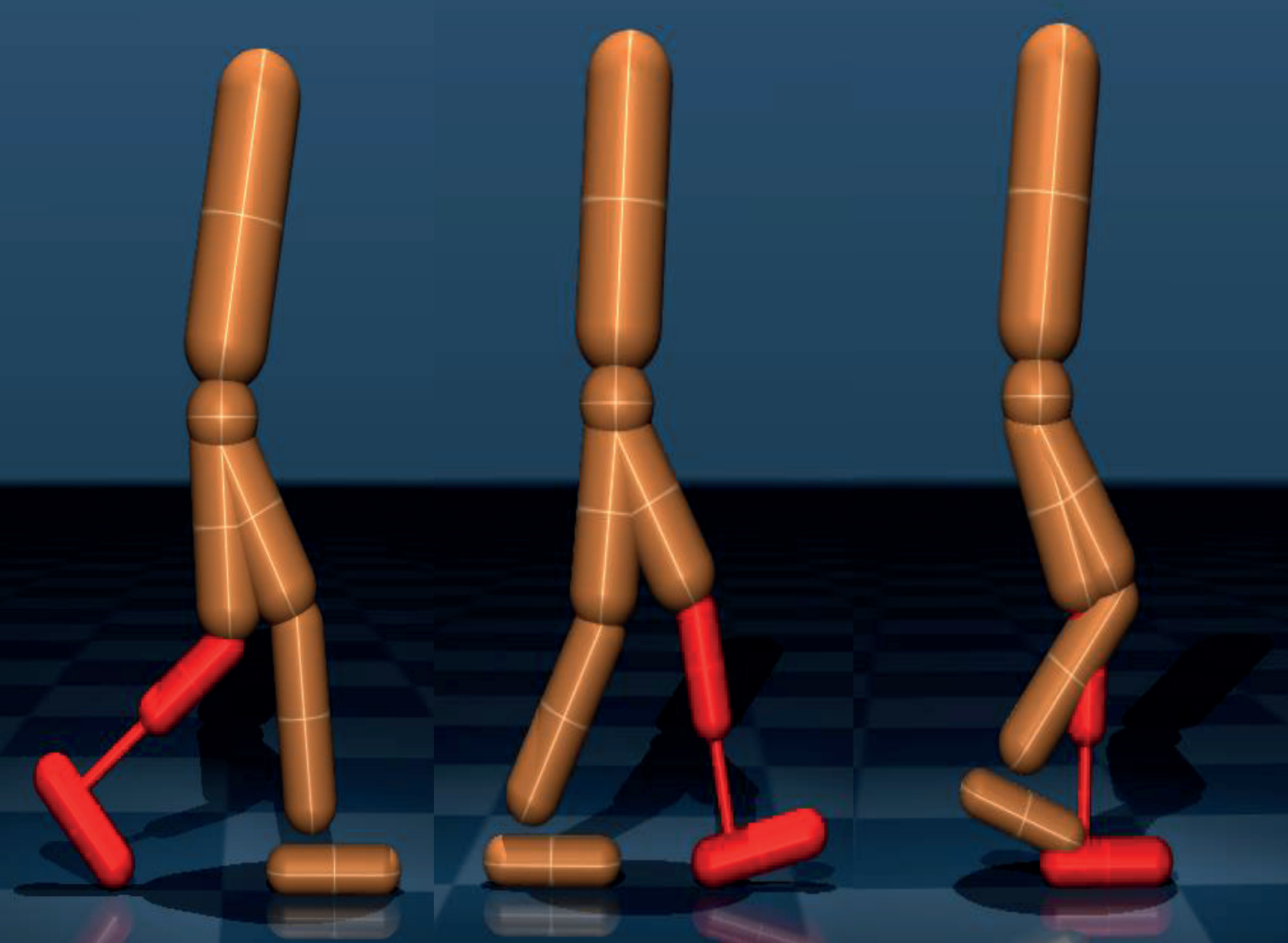
MSc Thesis in Electrical Engineering, Mathematics and Computer Science
Delft University of Technology

Online IC parameter adjustment of an active knee prosthesis using Reinforcement Learning with frequency-domain state representations

Can Çetindağ
2023



Can Çetindağ: *Online IC parameter adjustment of an active knee prosthesis using Reinforcement Learning with frequency-domain state representations.* (2023)

The work in this thesis was carried out with:

Reboocon Bionics B.V.

Delft Center for Systems and Control
Delft University of Technology

Supervisors: Assoc. Prof. Peyman Mohajerin Esfahani
Asst. Prof. Robert McAllister
Ir. Manav Penubaku

Preface

This master thesis has been written to fulfill the graduation requirements of the *Signals and Systems* track of *Electrical Engineering Mathematics and Computer Science* at the *Delft University of Technology*. I have produced this work in collaboration with *Reboocon Bionics B.V.* and *Delft Center for Systems and Control* within seven months from November 2022 to June 2023.

The prosthesis domain is a domain I had a close interest in since my amputation in 2017. When I got involved recently two years ago, the first thing I realized was the gap between research-oriented scientific works and commercially available products. This project was an excellent opportunity to address this gap as a middleman between the T.U. Delft and Reboocon Bionics.

Until now, my background mainly concentrated on offline learning problems, where data availability was always the primary issue on specific applications. The prosthesis domain is such an application due to the difficulties involving (i) reaching amputees, (ii) logistics of collecting data, and (iii) variety in walking styles. In this atmosphere, online learning shines bright with its learning capabilities through the data stream.

The potential of reinforcement learning always intrigued me. Especially in the robotics domain, where an agent always needs to interact with its surroundings and occasionally with its user, reinforcement learning is a natural solution to many tasks. The problem in the scope of this thesis, personalization and adaptation of the prosthesis, particularly falls into this group of tasks. It is also proven to be a potent approach through the simulation experiments conducted for this thesis.

Throughout the thesis process, I had support from various parties, which all greatly contributed to the final product. I want to thank Assoc. Prof. Robert McAllister and Manav Penubaku for their algorithm-related and domain-related supervision, respectively. My weekly meetings with each of them ensured the smooth synchronization of the research and commercial side of the task in concern. I also want to thank Assoc. Prof. Peyman Mohajerin Esfahani and Shiqian Wang for their precise comments and suggestions throughout this thesis.

Finally, I want to present my gratitude to my family and friends, who always kept their support and appreciated my work.

Can Çetindağ
Delft, The Netherlands
19.08.2023

Contents

List of Figures

List of Tables

List of Algorithms

1. Introduction

According to the World Health Organization (WHO), an estimated 30 million people worldwide live with limb loss, with projections indicating a substantial rise due to aging populations, increased incidents of chronic diseases, and traumatic injuries. An estimated 1.7 million people suffer from lower extremity limb loss, which is projected to double by 2050 if current trends persist. Apart from the physical challenges caused by being an amputee, amputees are threatened with their psychological well-being as they fail to integrate the society.

Recently, a significant amount of research has been concentrated on improving the quality of life of lower extremity amputees using active knee prostheses. Different than their passive counterparts, an active knee prosthesis can support the user by exerting positive work through the motor it contains. They affect the life of amputees both ability-wise and health-wise. Using active prostheses, amputees can adapt to different walking speeds and different terrains seamlessly, which is only possible through the controllable torque output of the device. On top of that, the torque exerted by the device compensates the lost muscles of the amputee. Thus, less stress on the other joints of the body is experienced by the user. Consequently, the metabolic cost of walking and the risk of having long-term health problems drop significantly.

Although the mentioned advantages attract the research community's attention, the dominant group of products in the industry are still more traditional, passive devices. One of the main reasons behind the delay in adopting these active devices is the complex control algorithms that should be utilized.

Impedance Control (IC) Law is the fundamental law in Human-Robot Interaction (HRI) literature as it allows safe interaction between humans and robots. In standard IC law, the torque applied by the motor is controlled by three parameters: *Stiffness Coefficient (K)*, *Setpoint Angle* (θ_s) and *Damping Coefficient (C)*. Throughout this thesis, these parameters will be referred to as *IC parameters*.

The naturally emerged control strategy for active devices is to divide an ambulation mode into phases (or states) and control the torque through a set of *IC parameters* assigned for each phase (Sup et al. [2008]). This is called *Finite State Machine Impedance Control (FSM-IC)*. Even though the number of phases is a design choice, the usual practice is having three or four phases. This results in up to 12 IC parameters to tune, which is only for level-ground walking. This number exponentially increases with the ambulation modes (level ground, stairs, ramp, etc.) the prosthesis tries to cover. Having so many IC parameters does not pose a problem, but it starts to become one as they need to be *personalized* and *adaptive*.

A personal FSM-IC controller is a natural direction in the future of active prosthesis control to address *interpersonal variations*. Each able-bodied (non-amputee) individual has a unique gait pattern that even can be used as a biometrics technique (Boyd and Little [2005]). It is mainly affected by features, such as height and weight of the individuals, but also influenced by unquantifiable features, such as health, ability, and habits. The variance in gait pattern

is even higher in amputees with additional features such as stump length, stump condition, etc. As a result, it is not realistic to assume that a single set of IC parameters would fit all amputees.

On the other hand, an adaptive controller is another natural direction to address *intrapersonal variations*. Currently, FSM-IC implementations utilize a static set of IC parameters. This approach assumes an amputee has an identical walking gait under any circumstances, but this assumption does not hold in real life. In real life, walking gaits can vary by many short-term and long-term factors. A stumble or being tired can introduce short-term variations, while a wound on the stump or significant weight change can introduce long-term variations.

A robust framework against the *interpersonal* and *intrapersonal* variations is needed to unveil the full potential of active prostheses in the market. The main goal of this thesis is to address this need by utilizing a reinforcement learning (RL) algorithm to adjust the IC parameters. By its nature, RL *agent* learns a *policy* (π) that map *states* (x) to *actions* (u) through its interactions with the *environment*. In this context, (a) policy is the personalized and adaptive control strategy, (b) states are the observations of the gaits, (c) actions are the adjustments on the IC parameters, (d) agent is the knee prosthesis and (e) environment is the human-prosthesis system. This framework is a promising approach because it effectively lifts the limitations of the current approaches by introducing personalization and adaptiveness.

Other researchers have previously studied the proposed framework concerning the same problem. The novel approach of this thesis lies in the proposed state definitions. Previous works attempted to encode the walking gait by their time-domain features. This approach is straightforward to implement but cannot extract trend information. Conversely, this thesis preserves the trend information by encoding the gait using frequency domain features. It is believed that the impact of this approach would be beyond the active prosthesis domain and can be translated to many robotics tasks that contain periodic trajectory following.

Within the scope of this thesis, *MuJoCo (Multi-Joint dynamics with Contact)*, an advanced physics simulation software that can handle contact forces, is utilized to create a multi-body model of a walking human-prosthesis system. Although similar simulations are used for prosthesis control, to the best of our knowledge, this is the first one that includes ground reaction forces (GRF). Including GRF plays a vital role in increasing the fidelity of the simulation. This simulation is used as the RL environment for the algorithm, and the analyses are made through the simulation.

Consequently, this thesis contributes to the literature with the following points.

1. Achieving a personalized and adaptive controller through an IC parameter adjustment algorithm that addresses interpersonal and intrapersonal variations.
2. Introducing a state representation with frequency domain features, and proving its superiority over the time domain features.
3. Introducing a multi-body human gait simulation that includes the GRF to achieve higher fidelity standards.

The thesis is structured as follows. First, Chapter 2 mentions the existing literature in the domain. Chapter 3 contains all the information about the method, from the simulation environment to algorithm details. Results are presented in Chapter 4 and reflected in Chapter 5. Some possible directions are also identified in Chapter Chapter 5. Finally, the thesis is concluded with Chapter 6.

2. Literature Review

2.1. Prosthesis Personalization

The most basic approach for prosthesis personalization (either knee or ankle) is the trial and error-based manual tuning of the parameters by the practitioners (Sup et al. [2009, 2010]). There are several problems with this approach. First, it is costly regarding time and effort (Lenzi et al. [2018]; Wen et al. [2019]). It solely relies on uneducated feedback from the user and visual observations of the practitioner (Sup et al. [2008]; Gao et al. [2021]), which may introduce bias to the process. Furthermore, it is not scalable due to the cross-interactions of the parameters (Gao et al. [2021]). These problems, which arise from manual parameter tuning, prevent powered prosthetics from reaching their full potential.

Few techniques attempted to solve the problems presented above. First, researchers tried to take the biological features as a reference either through a musculoskeletal model (Pfeifer et al. [2012]) or through biological impedance measurements for the ankle (Rouse et al. [2014]) and for knee (Tucker et al. [2017]). However, these methods require high domain knowledge in biomechanics, and they are not scalable to be applied in arbitrary prosthetic clinics. The second is to reduce the number of parameters rather than automatically personalizing them (Simon et al. [2013, 2014]). Although these methods address some of the problems related to time and effort, it is unclear whether they could be effective for each amputee (Gao et al. [2021]). On top of that, they still rely on uneducated user feedback and visual observations of the practitioner, which potentially introduce bias. One other technique that automizes the process is to utilize a cyber expert system (Wang et al. [2013]; Huang et al. [2016]) to hardcode the decisions of particular prosthetists, but this does not remove the possible bias as well, even though it reduces the effort by automizing the process.

All the techniques discussed above result in some sets of parameters for each ambulation mode. However, these techniques fail to address either the personalization or adaptiveness aspect or both. This thesis aims to approach the IC parameter personalization problem using reinforcement learning (RL) and achieve a more dynamic set of parameters with gait-to-gait adjustment to expose the full potential of powered devices.

2.2. Reinforcement Learning in Prosthesis Personalization

Although RL has attracted substantial research from several domains such as control, computer science, and psychology for a long time, the reflection of these researches in real-life applications is recent as one or two decades. Especially after proving itself through games that require discrete decision-making, such as chess (Acher and Esnault [2016]) and GO (Silver et al. [2017]), RL started to gain momentum in industry implementations. In robotics, RL emerged as the naturally appealing solution for complex problems (Singh et al. [2022]).

More elaborately, a robotic unit in concern is a one-to-one physical projection of the abstract *agent* in the usual RL setting, which makes RL highly suitable for robotics applications.

RL is an appealing candidate for achieving personal and adaptive active knee prosthesis control. First, a group of RL algorithms mainly focus on cases where it is tough, if not impossible, to derive a mathematical description of the process. This is called *model-free reinforcement learning*, and it solely relies on the user's interaction with the environment in its decision-making strategy. Considering the difficulties of modeling the level ground walking of each amputee, a model-free RL approach shines brightly as a candidate approach to personalize the controller. Secondly, by using RL, one can avoid having a static set of parameters and achieve adaptiveness by responding to the variations across gaits.

For the last half-decade, significant effort has been put into applying reinforcement learning to IC parameter personalization problem for powered prosthetic devices through the studies by Wen et al. [2019]; Gao et al. [2021, 2019]; Li et al. [2019]; Gao et al. [2020]; Wen et al. [2020]; Alili et al. [2021]; Wu et al. [2022]; Li et al. [2022a,b]; Liu et al. [2022]. The studies by Wen et al. [2019] and Gao et al. [2021] set the fundamentals of the approach, where the authors tested their approaches in real-time and simulation, respectively. It should be stressed that none of these studies evaluated their implementation through quantitative performance measures but only through convergence.

Wen et al. [2019] initiated RL application in active knee prosthesis using Actor-Critic RL algorithm. They used neural networks as their approximators and prosthesis kinematics as their features. Their follow-up work Wen et al. [2020] uses the same framework to conclude that the prosthetic kinematics alone are insufficient to ensure improved performance, and the wearer-prosthesis interaction should be investigated through some symmetry measures.

The rest of the studies focus on the policy iteration algorithms. Policy iteration algorithms aim to find the optimal policy simply by sampling the acting policy, evaluating the samples, and improving the policy. Gao et al. [2019] is the first study to utilize this approach, and they used polynomial basis functions as their approximators. They achieve convergence within $60-80$ gaits. Li et al. [2019] improved the convergence time to 40 gaits by utilizing quadratic basis functions and offline training. In their implementation, offline training should be carried out with the data collected from a *similar* patient. This raises a limitation since it is not always possible to have access to similar patients. To accommodate online and offline training, the study is extended within the study of Li et al. [2022b]. The algorithm is observed to converge within 120 steps with online training. However, this is not tested in the real environment.

A target should be identified for the system to create a complete RL framework. The initial attempts targeted healthy human gait knee trajectory for the RL agent. However, the efficiency of the approach is questionable due to the possible mismatch between healthy and amputee gaits. To address this issue, Li et al. [2022a] applied Bayesian optimization on top of the work of Gao et al. [2019] to identify the importance of the target features and achieved convergence within $14.1 \pm 4.5 mins$. On the other hand, Liu et al. [2022] utilized inverse reinforcement learning (IRL) to personalize the cost function by learning cost function weights. Their approach converged within 20 gaits.

The fundamental difference between these studies and this thesis is how the controller is personalized. The personalization achieved by these studies is through *updating* the IC parameters, where this thesis utilizes an *adjustment* approach. The main advantage of the adjustment approach is to achieve adaptiveness, as the controller can choose different adjustments for each gait.

2.3. Modelling Techniques

Numerous models in the literature address level ground walking from several perspectives concerning different aspects of it. The most basic models concern only single side and single phase, primarily based on a basic double pendulum for swing and an inverted double pendulum for stance (Luengas et al. [2015]; Ranzani [2014]).

As the aim of the model gets more complex, like a full gait concerning both sides, bipedal models start to emerge. Most models in the literature simplify the head, arms, and trunk as a single entity called *HAT* (Ranzani [2014]; Minh et al. [2020]; Shandiz et al. [2013]). Bipedal models can be classified according to the segments they include. 5-link models (Minh et al. [2020]) include HAT, both thighs and shanks, whereas 7-link models (Shandiz et al. [2013]) also include both feet. These models can be 2D or 3D, depending on their degrees of freedom (DoF).

The modeling part of this thesis does not concern with perfect replication of the level-ground walking. Instead, the aim is to create an environment to apply RL. Gymnasium (Towers et al. [2023]) is a project that provides various benchmark environments to train RL algorithms. The *Humanoid-v04* model is one of those environments with 13 body parts and 17 joints. This customizable humanoid model is modified for this thesis in line with the bipedal models available in the literature.

3. Methodology

The methodology of this thesis is presented using five parts. In the first part, an introduction to the current state of the practice and some key concepts in active prosthesis control are given, along with a mathematical formulation. The second part entails creating an environment suitable for training the reinforcement learning algorithm. A detailed breakdown of the custom reinforcement learning environment is given by underlining key limitations and associated implementation choices. The third part is dedicated to the learning process, where the implementation choices related to the algorithm are discussed. A thorough discussion of the chosen state vector, action vector, cost function, and Q function is included in this part. The pseudocode for the algorithm is also presented, along with the remaining implementation details. Finally, the evaluation metrics to evaluate the algorithm are presented. Through a structured approach encompassing these parts, this section aims to provide a comprehensive understanding of the problem, the solution methodology, and the technical aspects of the algorithm employed to address the problem.

3.1. Problem Formulation

A solid understanding of the mathematics behind a problem is critical in any scientific study. In active prosthesis control, it is essential first to comprehend the FSM-IC strategy to formulate the problem effectively. Therefore, this subsection is divided into two parts. The first part outlines the general principles of the FSM-IC strategy, providing a foundation for understanding the mathematical modeling of the approach presented in the second part of this subsection.

3.1.1. Finite State Machine with Impedance Control (FSM-IC)

A finite state machine (FSM) is a mathematical model representing a system as a set of states where the system transitions from one state to another in response to input signals. FSMs have numerous applications in control systems, computer science, and many other fields.

On the other hand, impedance control is a technique used in robotics to control the interactions between a robot and its environment by creating a virtual spring-damper system. It involves adjusting the robot's impedance, or resistance to movement, in response to the forces it experiences. Impedance control is often used to achieve precise and compliant interactions between a robot and its environment by using the law in Equation 3.1,

$$\tau = -K(\theta - \theta_s) - C\dot{\theta} \tag{3.1}$$

where K is the stiffness coefficient, θ_s is the set-point angle and C is the damping coefficient. Together they constitute the *IC parameters*, which are the main concern of this thesis. θ and $\dot{\theta}$ are the angular position and velocity of the joint, respectively. In the context of this thesis, they applied to the knee, and they will be referred to as knee kinematics. Finally, the torque the motor applies is depicted by τ.

An active prosthetic knee typically consists of a motorized joint controlled by an electronic system. Combining a finite state machine with impedance control results in a robust control system for an active knee prosthesis. In this approach, the finite state machine represents the high-level control of the system, and it is used to switch between different IC parameters based on the system's current state. On the other hand, impedance control adjusts the robot's impedance in response to the forces it experiences by utilizing the IC parameters selected by FSM. This system can incorporate a large set of modes, such as standing, walking, and going up or down stairs, as well as adjusting the knee's resistance to movement, allowing it to respond appropriately to different walking speeds, slopes, and other factors in a natural, efficient, and safe way. The system is schematized in Figure 3.1.

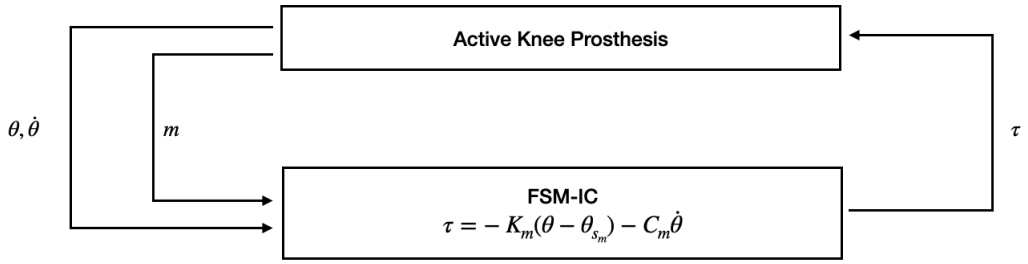


Figure 3.1.: Traditional control of an active knee prosthesis through FSM-IC control strategy. θ and $\dot{\theta}$ are the knee kinematics extracted from the knee, m is the system's state, and τ is the torque applied to the motor due to FSM-IC.

In the context of level ground walking, which is the scope of this thesis, states of the system are defined as the *phases* of the level ground walking. As depicted in Figure 3.1, each phase has its own IC parameters. The design choices purely determine the number of phases, but the standard approach is to have three to four phases. This thesis adopts a three-phased approach, and the phases are presented in Figure 3.2.

A full *gait* is the pattern of movement and coordination of limbs during one locomotion cycle. In this context, the trajectory given in Figure 3.2 is the knee trajectory of a healthy individual during walking. This trajectory starts from the toe-off, the instant the toe is off the ground, to the next toe-off.

3.1.2. Mathematical Description of Level-ground Walking

Walking, while often considered a simple activity, is a complex process that involves coordination between various parts of the body, when an individual walks, the hips, knees, ankles, and feet must be orchestrated to maintain balance and propel them forward.

Like many other body components, the central nervous system also controls joints, and the control strategy is not very different from the already introduced impedance law in Equation

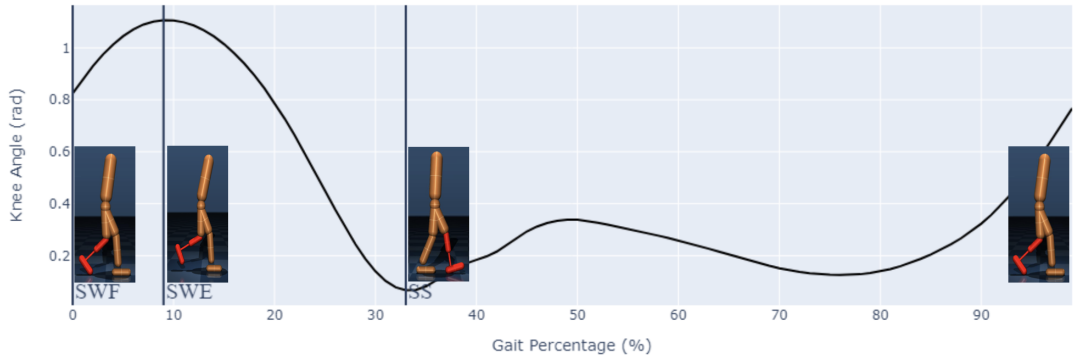


Figure 3.2.: Knee angle trajectory of a healthy human during level-ground walking. Divided into three phases, *Swing Flexion (SWF)*, *Swing Extension (SWE)*, and *Stance (SS)*.

3.1. To make sure the perfect collaboration and synchronization, the stiffness and damping coefficients (K and C in Equation 3.1, respectively) of all involved joints are controlled continuously and simultaneously by our nervous system through contraction and relaxation of muscles.

Thus, a general mathematical model of the joint control system of a healthy individual can be described with high dimensional state information (x) and a high dimensional control input (u). The state information contains all the information gathered by the sensory organs, and the control input is the continuous-time adjustments made on the stiffness and damping coefficients by the muscles.

This thesis makes three significant simplifications to accommodate a similar mathematical description to active knee prostheses control. Firstly, the state space is absent from sensory organ information, naturally occurring with prostheses that only utilize mechanical sensors. Secondly, the control input is only limited to the adjustments on the robotic knee as it is the only electronically controllable joint for an above-knee amputee. It should be noted that although it is possible to control the ankle joint with an active ankle, this thesis focuses on the active knee with a passive ankle. Different from the physiological system, utilizing the impedance law allows controlling the equilibrium angle (θ_s in Equation 3.1) and the stiffness and damping coefficient. Lastly, the FSM-IC control strategy only allows adjustments for each phase, meaning that a discrete-time control strategy is adopted instead of the continuous-time control employed by the nervous system.

In line with the above simplifications, the problem drops to a discrete-time problem for each three-phase described in Figure 3.2. To ease the notation, the system will be defined for only one phase, but the actual implementation consists of three identical systems running

parallel, one for each phase. This system can be described as,

$$\begin{aligned} x^+ &= f(x,u) \\ x &= L(y) \\ u &= \pi(x) \end{aligned} \tag{3.2}$$

where x is the system's state, representing the gait. Representing the gait through the exact measurements, y is straightforward, but this approach comes short mainly due to the resulting huge state vector drastically increasing the computational cost. Instead, the state is approximated through a feature extraction function L. This function is applied to the y, which is the collection of the knee angle measurements in the following form,

$$y = \begin{bmatrix} \theta(0) \\ \theta(1) \\ \vdots \\ \theta(T) \end{bmatrix} \tag{3.3}$$

where $\theta(\cdot)$ is the consecutive knee angle measurements and T is the period of the corresponding gait. The action, or the control input, is the IC parameter adjustments taken under the control policy π. x^+ is the next state, extracted from the following gait that occurred through an unknown system dynamics f. The overall system is schematized in Figure 3.3.

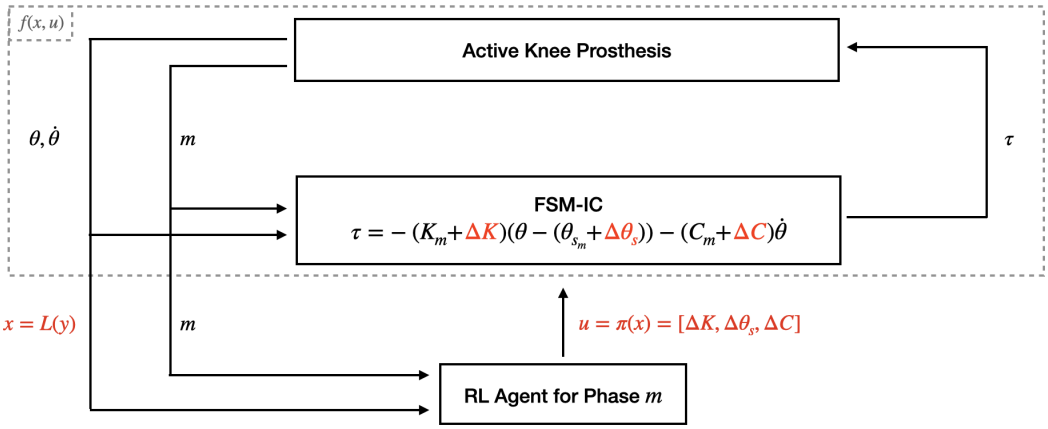


Figure 3.3.: Control of an active knee prosthesis enhanced by RL framework. State x is the encoded information from the previous gait, and action u is the appropriate adjustment on the IC parameters.

The system's dynamics is a nonlinear unknown function, which depends on the prosthesis user. It can be affected by quantifiable factors such as height, weight, and segment lengths, as well as by non-quantifiable factors such as ability, walking style, etc. These are some examples of permanent factors. However, there are also temporary factors due to imperfect sockets, terrain conditions, or tiredness. Thus, although walking does follow some basic commonalities, the function f is a highly personal function that is difficult to derive for each individual.

This thesis aims to enhance the control of prostheses by adjusting the IC parameters during each gait cycle based on measurements from the previous gait cycle. The presented system description in Equation 3.2 and in Figure 3.3 stresses this thesis's two main concerns. First, how to represent the gait, meaning the structure of the feature extraction function, L. This is discussed in detail in Section 3.3.1. The second is deciding the IC parameter adjustments, meaning determining the policy. This process is conducted through the RL framework, which is discussed from Section 3.3.2 to Section 3.3.4.

3.2. Environment

An environment dedicated to running the algorithm is one of the most fundamental components of a reinforcement learning (RL) project. Thus, within the period of this thesis, a significant amount of effort is dedicated to creating and customizing the environment. This environment is created by *Multi-Joint dynamics with Contact (MuJoCo)* (Todorov et al. [2012]), which is a physics engine that primarily focuses on multibody dynamics and accommodates most of the benchmarking environments in the field of RL (Towers et al. [2023]).

The created environment is a whole that contains both the *model* and the *simulation*. In this context, the model is the body's overall configuration designed to conduct the walking motion. At the same time, the simulation is the act of walking, including the implemented control strategy. The main trade-off of creating an environment is between fidelity and feasibility. Considering this trade-off, some key requirements for both model and simulation are decided. Both components of the environment will be discussed in this section by laying out the corresponding key requirements. Finally, the overall limitations of the environment are presented in this section.

3.2.1. Model

The model that this thesis utilized is a modified version of the 17 DoF Humanoid model of Gym Towers et al. [2023], which is one of the benchmarking environments for RL applications.

The main requirements of the model can be listed as follows,

1. Model should include aspects to investigate the symmetry
2. Model should be compatible with the embedded design of the controller.
3. Model should differentiate between amputated and healthy legs.

The Humanoid model has been reduced to a 2D model. This was a natural simplification since a prosthetic knee already lacks information about the center of mass and stability. However, symmetry can be evaluated through the duration of the swing of both legs. To investigate symmetry, a bipedal model is required. Thus, the Humanoid model was reduced to a 7-link model, which includes feet, shanks, thighs, and HAT (combined entity of head-arms-trunk). The first requirement is satisfied by the modification made to the Humanoid model.

As stated in Section 3.1.1, the current state-of-the-practice control strategy for active prosthesis is FSM-IC. Although MuJoCo allows direct torque feeding to the joints, it is not feasible in real-time human-prosthesis interaction due to safety concerns. Thus, all the joints of the model are controlled with FSM-IC by Equation 3.1.

To preserve the fidelity of the algorithm, the definitions of the phases for the active knee prosthesis (labeled with green in Figure 3.4) should be identical to the predetermined phases shown in Figure 3.2. The definitions of the phases are somewhat flexible for the healthy-side joints (labeled with blue in Figure 3.4) and prosthetic ankle (marked with yellow in Figure 3.4). For these joints, gait is divided into eight phases to increase the control precision. Passive ankles are based on springs and dampers. For this joint only, θ_s is set to zero in Equation 3.1 to represent a spring-damper system. The second requirement is also satisfied by defining the phases of the active knee prosthesis loyal to the predefined phases.

The number of phases and parameters for each type of joint is summarized in Table 3.1. While configuring the model, the segment lengths are adjusted for an $80kg$ person with a height of $1.80m$, using the ratios presented in Winter [2009]. The third requirement is also satisfied by controlling different joints with different strategies and adjusting the segment lengths accordingly.

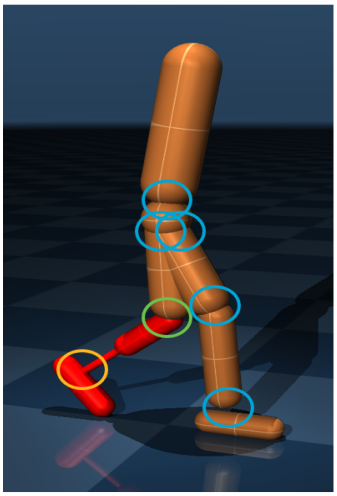


Figure 3.4.: The custom model created in MuJoCo. Blue circles depict the five healthy side joints, yellow circle depicts the passive ankle prosthesis, and green circle depicts the knee prosthesis aimed to control.

Joint Type	Count	Number of Phases	IC Parameters per Phase	Total IC Parameters	Noise
Healthy Joints	5	8	3	120	Yes
Passive Ankle Prosthesis	1	8	2	16	No
Active Knee Prosthesis	1	3	3	9	No
Total				*145*	

Table 3.1.: Number of parameters that should be initialized to apply the FSM-IC strategy to all joints.

3.2.2. Simulation

Simulation is essential to the RL environment to assess the algorithm's effectiveness. The followings are decided as the key requirements for the simulation.

1. Simulation should interact with the environment through the *ground reaction forces (GRF)*.
2. Simulation should successfully complete at least three gaits.

3. Simulation should include some uncertainty in the process.
4. Simulation should realize a faulty gait.

Ground reaction forces (GRF) are one of the most important aspects of walking. Previous works that concern the same problem utilized an environment without this aspect. Their environment consists of a sliding lower body without interacting with the ground. However, it is crucial to include GRF in the simulation to prove that the approach has reciprocity in the real-time application. One of the significant contributions of this thesis is the customized RL environment that includes GRF. This was possible due to the strong contact dynamics capabilities provided by MuJoCo.

The crucial part of training an RL algorithm is facing the consequences of its actions. The simulation requires a single gait to take an action and a second gait to evaluate that action. Considering the simulation starts from the same phase, specifically Swing Flexion, the simulation should have at least three gaits to cover the first and second gait pair for all phases. This requirement is satisfied by fine-tuning all 145 parameters in Table 3.1 by a genetic algorithm without noise. These fine-tuned parameters are accepted as the *base values*.

These base values of parameters are designed such that the model could successfully have three gaits with good hip and knee trajectories. Another crucial point of training an RL algorithm is to present diverse experiences to the agent to learn the expected outcomes of different state and action pairs. However, with constant base values, a physics engine produces the same behaviors. The diversity is achieved by adding Gaussian noise to the healthy-side IC parameters, and it is controlled by a hyperparameter called *noise factor*, which is a factor that adjusts the standard deviation of the Gaussian. Although the simulation starts from the same configuration, this noise allows a variation in the models' interaction with the environment, and thus, different experiences are observed in each simulation. This is also loosely related to representing different walking styles among individuals. All in all, by adding noise to the healthy-side IC parameters, the third requirement is satisfied. The prosthetic side was absent from the noise, as the prosthesis is expected to act the same across all patients.

Given the complexity of the walking task, having a faulty gait with a multi-body model is inevitable. The current experiment should reset and start over in case of a faulty gait. One caveat of this simulation strategy is defining what is faulty gait. There is no such mathematical description, so defining some *termination conditions* is decided. These conditions are kept as broad as possible to not over-fit a walking pattern by over-restricting conditions. These conditions are,

- If a gait takes more than 1.5 seconds, which indicates the model is stuck,
- If there is an irregular contact between a body and ground,
- If a foot lands behind the other foot.

These defined conditions allow the simulation to satisfy the final requirement.

3.2.3. Limitations

It should be stressed that the environment aims not to replicate human walking perfectly but to evaluate the RL algorithm. That said, the trade-off of fidelity and feasibility raises a few limitations.

In Section 3.1.2, the similarity between the healthy individual nervous control and the impedance control is underlined. Muscles adjust the stiffness and damping of the joints by contracting and relaxing the muscles connected to them. This process is continuous and, thus, can react to sudden changes very effectively. However, mimicking this would require adopting a musculoskeletal model, in which muscle behavior is also modeled. In line with the *fidelity vs. feasibility* trade-off, it is decided to exclude the muscle behavior and only uses a multi-body model. Considering that the prosthetic knee has no direct connection with the muscle system, this is a valid direction, as the compromised fidelity does not affect knee control. However, this loss in fidelity results in sub-optimal gaits, as there is no continuous compensation on the healthy side but constant stiffness and damping for each phase throughout an experiment.

Finally, as mentioned previously, specific noise is put on the healthy-side base IC parameters to create diversity during learning. Before starting each simulation, randomly corrupted healthy-side parameters are chosen and kept constant throughout the corresponding phase. New healthy-side IC parameters are determined when the simulation terminates and restarts. These selected corrupted parameters will be referred to as *nominal* parameters throughout the rest of the thesis, as these are the effective parameters of the simulation in that the adjustments are applied. That said, although three gaits were guaranteed with the base parameters, it is not always the case with the nominal parameters. This disrupts the continuity of the data and causes two issues, one in the training phase and one in the evaluation phase. First, it disturbs the learning process as the RL agent cannot resolve the future effects of its action. Second, in the same way, cost-based analysis becomes impossible as the long-term cost of the action vanishes. In other words, an action can be sub-optimal for the following gait and be optimal for further gaits, but the simulation terminates before experiencing those gaits.

3.3. Reinforcement Learning Framework

As the problem is formulated and the environment is introduced, there is enough foundation to continue with the algorithmic details, inspired primarily by Li et al. [2022b]. This section encompasses a complete RL framework, including (i) descriptions of state and action vectors, (ii) details on cost and Q functions, (iii) approximation of Q function, (iv) applying the policy and taking action, (v) implementation details and pseudocode for the algorithm, and finally (vi) the evaluation metrics.

3.3.1. State and Action Vectors

To the best of the author's knowledge, one of the most significant novelties of this thesis is how the state is described. Previous works (Wen et al. [2019]; Gao et al. [2019] and the follow-up works mentioned in Chapter 2) attempted to represent the state in the time domain by its extrema. The extrema are encoded by their deviation in location (ΔD) and their deviation in magnitude (ΔP) from the health individual knee trajectory as shown in 3.5. This approach loses a significant amount of information about the trend the trajectory follows.

In the previous studies, the time domain approach is utilized, and gait is divided into four phases, where peaks correspond to the boundaries of the phases. A single pair of the deviation in value and duration of the corresponding peak is used as the features, meaning each

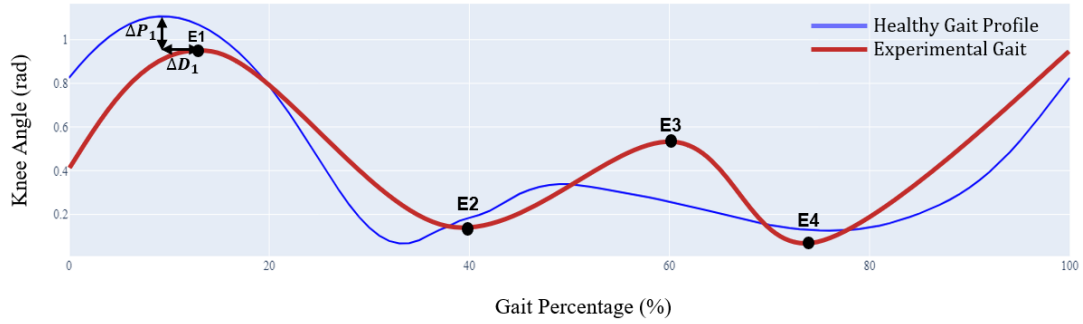


Figure 3.5.: Knee trajectory of a single gait with indicated time-domain state features.

state has only two features. This approach is not desirable for two reasons. Firstly, the peaks do not coincide with the phase transitions shown in Figure 3.2. Secondly, the adjustments on the impedance parameters rely only on a single peak, which means the information from approximately two third of the gait is discarded.

To avoid the above-mentioned information loss, this thesis utilized a frequency-domain approach. It is well known that linear combinations of sines and cosines can describe a periodic signal. This is called Fourier Series and is defined in the following way,

$$f(\zeta) = \sum_{n=0}^{\infty} a_n cos(n\zeta) + b_n sin(n\zeta) \tag{3.4}$$

where ζ is a general dummy variable, n is the integer multiple and (a_n, b_n) pair is the coefficients of the corresponding frequency. This can be used to approximate the signal in the following way,

$$f(\zeta) \approx \sum_{n=0}^{N} a_n cos(n\zeta) + b_n sin(n\zeta) \tag{3.5}$$

where N is the number of harmonics used to approximate the function.

Knee trajectory is inherently periodic during a constant locomotion mode, so using Fourier Series is a valid approach to approximate the knee trajectory. To validate this approach, the approximation performance to the number of harmonics is investigated with Figure 3.6. Figure 3.6 suggests that the approximation error converges with a few pairs of harmonics (a_n and b_n). During the experimentation of this thesis, four to eight harmonic pairs are used.

Each harmonic can be represented by the pair (a, b). Thus, to represent a trajectory with N harmonics, $2N$ elements are required in the state vector. The first harmonic is always the DC offset of the signal, meaning b_0 is zero for every signal. Therefore there is no new information from b_0. The resulting $2N - 1$ elements are sufficient to represent the gait with N harmonics.

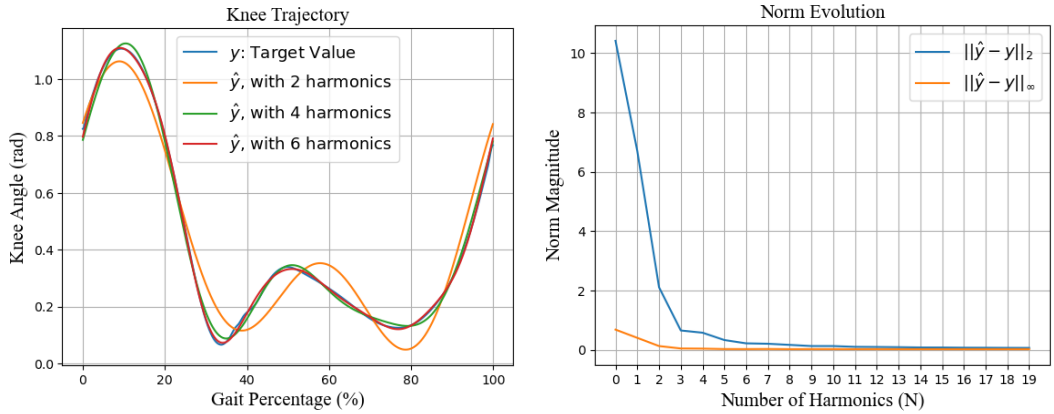


Figure 3.6.: Performance of the Fourier Series to approximate the target knee trajectory with the increasing number of harmonics. y is the collection of a gait's angle measurements, as defined in Figure 3.3. Left: through knee angle trajectory, right: through L_2 *norm* (blue) and L_∞ *norm* (orange)

The features decided to be the difference between the experimental harmonics (a_n, b_n) and the target harmonics $(\overline{a_n}, \overline{b_n})$, e.g. in Equation 3.6. However, the target harmonics of each RL agent are different, as each is concerned with a shifted version of the knee trajectory. As a signal shift, its harmonics change as well. Table 3.2 are the target values of the harmonics for each RL module.

$$f_{a_n} = \Delta a_n = a_n - \overline{a_n} \tag{3.6}$$

where f_{a_n} is the feature corresponding to a_n. On the other hand, each RL module is concerned with a shifted version of the knee trajectory.

	$\overline{a_0}$	$\overline{b_0}$	$\overline{a_1}$	$\overline{b_1}$	$\overline{a_2}$	$\overline{b_2}$	$\overline{a_3}$	$\overline{b_3}$	$\overline{a_4}$	$\overline{b_4}$	$\overline{a_5}$	$\overline{b_5}$	$\overline{a_6}$	$\overline{b_6}$	$\overline{a_7}$	$\overline{b_7}$
SWF	-0.851	0.0	-0.297	-0.205	-0.147	-0.240	0.064	-0.059	-0.005	0.017	-0.014	-0.013	0.0003	-0.008	0.003	0.001
SWE	-0.851	0.0	-0.361	-0.015	-0.279	0.031	-0.067	-0.056	0.017	-0.007	0.009	0.017	0.002	0.008	-0.003	0.002
SS	-0.851	0.0	-0.0371	0.359	0.281	0.0049	0.067	-0.055	0.018	-0.003	0.0186	-0.003	0.001	-0.008	-0.0003	-0.004

Table 3.2.: Target harmonic values for each RL module up to 8^{th} harmonic.

Amputee knee trajectory is not entirely identical to the healthy knee trajectory. Studies show that to ensure comfortable walking, walking symmetry should somehow include in the picture Wen et al. [2020]. Thus, adding a feature corresponding to the walking symmetry is decided. The symmetry metric is evaluated by comparing the swing time of both legs through a symmetry index (Viteckova et al. [2018]) given in Equation 3.7.

$$Sw_{SI} = \frac{2 \cdot (Sw_s - Sw_p)}{(Sw_s + Sw_p)} \tag{3.7}$$

where Sw_s is the duration of the healthy leg swing, and Sw_p is the duration of the prosthetic leg swing. As a result, the final state vector is formed in the following way.

$$x = \left[\Delta a_0,\ \Delta a_1,\ \Delta b_1,\ \Delta a_2,\ \Delta b_2,\ \cdots,\ \Delta a_{N-1},\ \Delta b_{N-1},\ Sw_{SI}\right]^T \tag{3.8}$$

Control input (u), on the other hand, has only three elements. Each element corresponds to the adjustment on one of the IC parameters applied for the following phase and has the form presented in 3.9.

$$u = \left[\Delta K, \Delta\theta_s, \Delta C\right]^T \tag{3.9}$$

Although control input is more straightforward to define, the effect it represents in the system should be stated clearly. Embedding the control input into the impedance law, adjustable impedance law can be achieved in Equation 3.10.

$$\tau = -(\hat{K} + \Delta K)(\theta - (\hat{\theta}_s + \Delta\theta_s)) - (\hat{C} + \Delta C)\dot{\theta} \tag{3.10}$$

where $\hat{K}$, $\hat{\theta}_s$, $\hat{C}$ are the nominal values for stiffness, equilibrium angle, and damping coefficient, respectively. ΔK, $\Delta\theta_s$, ΔC are the adjustments on the corresponding nominal IC parameter. θ and $\dot{\theta}$ are the knee kinematics, and finally, τ is the torque that the motor should apply.

By design, it is decided to keep the adjustments $\pm 10\%$ of the nominal value of the corresponding IC parameter. A joint discrete action space, $\mathcal{U}$, a $21x21x21$ grid with boundaries of $\pm 10\%$ is created. It should be noted that the nominal parameters are different for each phase, and the adjustments are made at every phase transition for the following phase only, which means that the actions are not updates on the parameters but temporary adjustments.

RL can learn a problem with a stream of data online. The basic idea is to explore the policy by evaluating the outcome of the actions taken previously. Throughout this thesis, the following notation will be used to represent the complete data stream,

$$\mathcal{D} = \{x_n, u_n, x_n^+\}_{n=0}^{n=N-1} \tag{3.11}$$

where N is the size of the data stream. A single sample tuple, which is the most fundamental piece of information, can be defined as follows,

$$d = (x, u, x^+) \tag{3.12}$$

where (x, u) is the first state-action pair, and x^+ is the following state following the simulated system dynamics.

3.3.2. Cost Function and Q-value Function

Considering the state and action definitions presented in Equation 3.8 and in Equation 3.9, respectively, the objective is to achieve the smallest state possible with an action that has minimal energy expenditure. Thus, the single-gait cost function represents this objective for a single (x, u) pair is defined as follows.

$$g(x,u) = x^T R_x x + u^T R_u u \tag{3.13}$$

where R_x and R_u are predefined diagonal matrices with positive values, both becoming positive semi-definite (PSD) matrices. The associated Q function (Watkins [1989]) in the discounted infinite horizon setting is,

$$Q(x_k,u_k) = g(x_k,u_k) + \sum_{n=k+1}^{\infty} \gamma^{n-k} g(x_n, \pi(x_n)) \tag{3.14}$$

$$= g(x_k,u_k) + \gamma Q(x_{k+1}, \pi(x_{k+1})) \tag{3.15}$$

$$Q(x,u) = g(x,u) + \gamma Q(x^+, \pi(x^+)) \tag{3.16}$$

where γ is the hyperparameter to adjust the importance of the future costs, called the *discount factor*. π is the policy that the system employs, and x_+ is the next state following the system dynamics $f(x,u)$, assuming f is deterministic. Despite introducing noise into the system, the deterministic assumption was maintained due to the lack of uncertainty within a given simulation. Equation 3.16 can be rewritten using the equality $u^+ = \pi(x^+)$ in the following way.

$$Q^\pi(x,u) = g(x,u) + \gamma Q^\pi(x^+, u^+) \tag{3.17}$$

Equation 3.24 is called the Bellman Equation (Bellman [1954]), and it is one of the foundations of dynamic programming and reinforcement learning. The ultimate objective of this function is to achieve the optimal Q function that minimizes the long-term cost of the system. This can be defined as

$$Q^*(x,u) := g(x,u) + \gamma \min_{u^+ \in \mathcal{U}} Q^*(x^+, u^+) \tag{3.18}$$

The policy that achieves this objective is the optimal policy π^*, which can be defined as in Equation 3.18.

$$\pi^*(x) := \arg\min_{u \in \mathcal{U}} Q^*(x,u) \tag{3.19}$$

Realize that, feeding Equation 3.19 to Equation 3.18 falls back to Equation 3.16 with the optimal versions of the Q function and π, which can be expressed as in Equation 3.20.

$$Q^*(x,u) = g(x,u) + \gamma Q^*(x^+, \pi^*(x^+)) \tag{3.20}$$

where (x,u) is the first state-action pair and $g(x,u)$ is the associated stage cost. γ is the discount factor, x^+ is the next state following the system dynamics $f(x,u)$, Q^* and π^* are the optimal Q function and optimal policy, respectively.

3.3.3. Q-value Function Approximation and Iteration

To find an exact solution for the Bellman Equation (Equation 3.17), the Q function must be known. In most real-time problems, however, Q is not known exactly and should be approximated. For this thesis, linear combinations of quadratic basis functions are selected to parameterize Q as follows at a given update stage (Lagoudakis and Parr [2003]),

$$\hat{Q}^{(i)}(x,u) = \psi(x,u)^T r^{(i)} = \psi^T r^{(i)} \tag{3.21}$$

where $\psi(x,u)$ is the basis functions with a fixed form. r is the weighing vector, the $\hat{Q}$ is the approximated Q function and i is the index of the update iteration. Although the approximation shown in Equation 3.21 can be achieved by an advanced universal approximator as neural networks, this thesis utilized quadratic basis functions. The main reason is to avoid the uncertainty that would surface using many free parameters and to have more transparent behavior.

Due to the approximation given in Equation 3.21 and the noise introduced to the system, the strict equality in Equation 3.16 is not necessarily true anymore and turns into the following form (Lagoudakis and Parr [2003]).

$$Q^{(i)}(x,u) \simeq g(x,u) + \gamma \min_{u^+ \in \mathcal{U}} Q^{(i)}(x^+, u^+) \tag{3.22}$$

The mismatch between the left and right-hand sides arises as a natural metric called Bellman error (BE). This mismatch represents how the Q function of the i^{th} iteration is off from the optimal Q function defined in Equation 3.18 (Baird [1995]). BE can be defined in the following way for a single sample,

$$BE = \left| Q^{(i)}(x,u) - g(x,u) - \gamma \min_{u^+ \in \mathcal{U}} Q^{(i)}(x^+, u^+) \right| \tag{3.23}$$

and for all samples available, the average BE can be defined as,

$$\overline{BE} = \frac{1}{N} \sum_{k=0}^{N-1} \left| Q^{(i)}(x_k,u_k) - g(x_k,u_k) - \gamma \min_{u_k^+ \in \mathcal{U}} Q^{(i)}(x_k^+, u_k^+) \right| \tag{3.24}$$

where N is the number of the available data, k is the data index.

Defining the Q function as in Equation 3.21 inherently assumes that the Q function is a linear combination of quadratic basis functions. To propagate the problem into a linear system form ($Ar = b$), Equation 3.21 can be substituted into 3.22,

$$\psi(x_k, u_k)^T r^{(i)} \simeq g(x_k, u_k) + \gamma \psi(x_k^+, u_k^+)^T r^{(i)} \tag{3.25}$$

Notice that the this expression is defined for k^{th} datum, (x_k, u_k, x_k^+), and a datum does not include u_k^+. u_k^+ is a policy-dependent entity that varies at each update as $r^{(i)}$ changes, which is equivalent to $\pi^{(i)}(x_k^+)$. Equation 3.25 can be transformed into Equation 3.26 to express this dependency.

$$\psi(x_k, u_k)^T r^{(i)} \simeq g(x_k, u_k) + \gamma \psi\left(x_k^+, \pi^{(i)}(x_k^+)\right)^T r^{(i)} \tag{3.26}$$

where $\pi^{(i)}$ is the effective policy of the i^{th} update iteration, parameterized by $r^{(i)}$. The expression can be ordered further in the following way. To ease the notation, $g(x_k, u_k)$, $\psi(x_k, u_k)$ and $\psi\left(x_k^+, \pi^{(i)}(x_k^+)\right)$ will be expressed as g_k, ψ_k and $\psi_{k+}^{(i)}$, respectively.

$$(\psi_k^T - \gamma(\psi_{k+}^{(i)})^T) r^{(i)} \simeq g_k \tag{3.27}$$

Inspired by the literature (Lagoudakis and Parr [2003]), both sides can be projected onto the space spanned by the basis functions by multiplying both sides with ψ_k

$$\psi_k(\psi_k^T - \gamma(\psi_{k+}^{(i)})^T) r^{(i)} \simeq \psi_k g_k \tag{3.28}$$

Equation 3.28 is expected to hold $\forall d \in \mathcal{D}$. This can be expressed in the matrix form as follows.

$$\tilde{A} r^{(i)} \simeq \tilde{b}^{(i)} \tag{3.29}$$

where $r^{(i)}$ is the weight vector of i^{th} iteration that is expected to solve the linear system. Using all the data available, $\tilde{A}$ and $\tilde{b}$ can be approximated as (Li et al. [2022b]; Lagoudakis and Parr [2003]),

$$\tilde{A}^{(i)} = \frac{1}{N} \sum_{k=0}^{N-1} \psi_k(\psi_k^T - \gamma(\psi_{k+}^{(i)})^T) \tag{3.30}$$

$$\tilde{b}^{(i)} = \frac{1}{N} \sum_{k=0}^{N-1} \psi_k g_k \tag{3.31}$$

where N is the number of data available to calculate $A^{(i)}$ and $b^{(i)}$, ψ and $\psi_+^{(i)}$ are the quadratic basis functions associated with the pair (x, u) and $(x^+, \pi^{(i)}(x^+))$, respectively. g is the cost of the corresponding pair, and γ is the discount factor. Although the b calculation looks like it is not dependent on the update index, it is inherently dependent as the data stream changes at every calculation.

If the chosen parameterization is perfectly accurate for the system, Equation 3.29 should hold with the optimal weight vector, r^*. However, $\tilde{A}$ and $\tilde{b}$ are empirical approximations achieved with the available data stream $\mathcal{D}$, which is the collection of tuples of (x, u, x^+). The approximation error can be derived as,

$$E^{(i)} = \tilde{A}^{(i)} r^{(i)} - \tilde{b}^{(i)} \tag{3.32}$$

where $E^{(i)}$ is the approximation error in the i^{th} iteration. Inspired by the works of Bertsekas [2011], this can be used to update the weight vector leading to a gradient descent approach shown in Equation 3.34.

$$\Delta r^{(i)} = \tilde{A}^{(i)} r^{(i)} - \tilde{b}^{(i)} \tag{3.33}$$

$$r^{(i+1)} = r^{(i)} - \eta \Delta r^{(i)} \tag{3.34}$$

where η is a hyperparameter to adjust the learning step, called the learning rate. It is common to include a time dependency on the learning rate to ensure the learning process convergence. This time dependency is included in the following way,

$$r^{(i+1)} = r^{(i)} - \eta^{(i)} \Delta r^{(i)} \tag{3.35}$$

$$\eta^{(i)} = \frac{\eta}{\rho i} \tag{3.36}$$

where i is the number of updates and ρ is a factor to adjust the decaying rate. Due to the early results and inspiration from the machine learning literature, it is also decided to add a momentum rate, μ, to reinforce the search for the optimal weight vector, r^*. By adding momentum, the final update rule has the form presented in 3.37.

$$r^{(i+1)} = r^{(i)} - \eta^{(i)} \Delta r^{(i)} - \mu \Delta r^{(i-1)} \tag{3.37}$$

where μ is the momentum rate and the $\Delta r^{(i-1)}$ is the update from the previous iteration.

This section encompasses the approximation and iteration to solve the problem. To summarize, each iteration has three steps:

1. Evaluate the policy (Equation 3.25-3.31)
2. Improve the policy (Equation 3.32-3.37)
3. Repeat

3.3.4. Applying the Policy

Revisiting Equation 3.21, the approximation for each iteration can be expressed in the matrix form as follows,

$$Q^{(i)}(x,u) = \begin{bmatrix} x \\ u \end{bmatrix}^T H^{(i)} \begin{bmatrix} x \\ u \end{bmatrix} \tag{3.38}$$

$$= \begin{bmatrix} x \\ u \end{bmatrix}^T \begin{bmatrix} H_x^{(i)} & H_{xu}^{(i)} \\ H_{ux}^{(i)} & H_u^{(i)} \end{bmatrix} \begin{bmatrix} x \\ u \end{bmatrix} \tag{3.39}$$

where $H^{(i)}$ is a weighing matrix that is achieved by reordering $r^{(i)}$ at any update iteration. The objective of the chosen policy is to minimize the expression given in Equation 3.38. Mathematically, it can be expressed as,

$$\pi^{(i)}(x) = \arg\min_{u \in \mathcal{U}} Q^{(i)}(x,u) \tag{3.40}$$

$$= \arg\min_{u \in \mathcal{U}} \left\{ \begin{bmatrix} x \\ u \end{bmatrix}^T \begin{bmatrix} H_x^{(i)} & H_{xu}^{(i)} \\ H_{ux}^{(i)} & H_u^{(i)} \end{bmatrix} \begin{bmatrix} x \\ u \end{bmatrix} \right\} \tag{3.41}$$

$$= \arg\min_{u \in \mathcal{U}} \{u^T H_u^{(i)} u + 2x^T H_{xu}^{(i)} u\} \tag{3.42}$$

where $\pi^{(i)}(x)$ is the policy at the i^{th} update iteration. The minimization problem depicted in Equation 3.42 is a convex problem as long as $H_u^{(i)}$ is positive semi-definite (PSD). However, it may not necessarily be PSD, particularly during the initial phases of the training process. That considered, $H_u^{(i)}$ is projected to the PSD cone by using eigenvalue decomposition to enforce convexity of 3.42. Thus, the convex optimization library called *cvxopt* is used to solve the problem. Appendix A shows the library's notation and the utilized implementation.

Ideally, as the training session proceeds, a better approximation of the Q function should be achieved, which results in the convergence of the policy to the optimal policy.

$$\pi^{(i)} \rightarrow \pi^* \tag{3.43}$$

Notice that the result of the convex optimization is continuous while the action space is discrete. The grid points closest to the continuous solution are chosen as the candidate solution and compared manually using Equation 3.21. This corresponds to comparing only eight points, the $2x2x2$ sub-grid in the action space, as opposed to the entire grid of $21x21x21 = 9,261$ points.

3.3.5. Algorithm and Implementation Details

Until now several hyperparameters are already discussed. These are the noise factor, number of harmonics, learning rate, momentum rate, and decaying rate. This subsection is dedicated to wrapping all the rest of the hyperparameters and concisely presenting a full list.

Punishment is essential in reinforcement learning, as it penalizes the state-action pairs with bad consequences. This is usually inherently included in the algorithm through the cost function definition. Although that is also the case for this thesis, *faulty gaits* cause a slight problem. First, faulty gaits after an action do not have a full gait, making it impossible to extract the states. Second, this thesis aims not to have fewer faulty gaits but to have good ones, which makes it mathematically questionable to include them in the cost function. Although one can argue that they are correlated, it does not have one-to-one correspondence due to the imperfections in the environment. In the end, it is decided to assign a *punishment state* and *punishment action* in case of a faulty gait, where the magnitude of the punishment is adjusted with *punishment factor*.

Exploration vs. exploitation is the trade-off at the heart of every RL implementation. Exploration randomly searches the action space, while exploitation refers to benefiting the current policy. A random exploration element should be included while taking action to explore the action space. During the implementation, *exploration factor* is used to adjust the exploration weight, representing the possibility of having a random action. Random action is uniformly chosen by the grid points that are closest to the actual solution.

Before proceeding to the algorithm, three quantities are highly common in machine learning projects, which are batch size (N_b), target sample size (N_s), and update threshold (ϵ). N_b is the number of samples required to conduct an update, and N_s is the maximum number of samples to terminate the training. ϵ, on the other hand, is the minimum update required to continue the training. All the hyperparameters and quantities mentioned in this chapter are summarized in Table 3.3.

σ_N	Noise Factor
n	Number of Harmonics
γ	Discount Factor
η	Learning Rate
μ	Momentum Rate
ρ	Decaying Rate
f_p	Punishment Factor
f_e	Exploration Factor
N_b	Batch Size
N_s	Target Sample Size
ϵ	Update Threshold

Table 3.3.: List of hyperparameters and quantities.

To create a solid foundation for the following chapters, it is beneficial to summarize the terminology. From particular to general,

- a *phase* is one of the three distinct parts (Swing Flexion, Swing Extension, Stance) of a gait,
- a *gait* is a full cycle of locomotion, which is level-ground walking in the scope of this thesis,
- a *sample* is two consecutive gaits, and the action between them as depicted in 3.12,
- a *simulation* is the collection of samples from the initialization of the model to its termination due to a faulty gait,
- an *update* occurs when there are N_b new samples from consecutive simulations,
- the *training session* is the collection of simulations until the end of training. Training sessions can be concluded in two ways which target sample size might be achieved or until the update magnitude is below the update threshold,
- the *test session* is the period from the termination of the training to the termination of the experiment,
- an *experiment* is the whole process.

The training procedure used to train the RL agents is given in Algorithm 1. It should be stressed again that three distinct RL agents are corresponding to each phase, which all train in parallel.

3.3.6. Evaluation Metrics

Evaluating the success of an algorithm is as important as the algorithm itself. One of the most significant challenges of this project is to assess the results, mainly because of two sets of reasons. First, as underlined in Section 3.2.3, the inability to evaluate an action's long-term effects is due to the termination of simulations. Second, the generic difficulties that surface with online evaluation, such as instability in the early stages and change in data at every training session.

One of the most common ways to evaluate an RL algorithm is to monitor the long-term costs of an action. The long-term is quantified by *the horizon* in the literature to determine how far the algorithm should investigate an action's quality. However, it is most convenient for sequential tasks, where the agent faces the consequences of its action without disruption, such as playing chess or go. Due to the limitations already mentioned in Section 3.2.3, this is not the case with the environment utilized in this thesis. The faulty gaits require the simulation to restart with completely new parameters, where the action taken in the previous simulation does not influence the new samples. In training a chess bot, this problem is analogous to playing an opponent who flips the board whenever the bot sacrifices a pawn. While we can still collect a large quantity of data by repeatedly restarting the game, we can rarely observe the long-term effects of any decisions.

As the cost-based analysis is out of the picture, it is decided to have a two-step evaluation. The first step is to investigate if the controller converges to policy, and the second is to evaluate if it converges to a good policy.

Convergence of the controller is relatively straightforward and ensured by the decaying learning rate presented in 3.36. Convergence of the controller means converging to a policy and a converging r vector. To investigate the convergence of the controller, Δr, the change

Algorithm 1 Online Training

Input:
Hyperparameters ▷ Table 3.3

Initialize:
initialize $\pi^{(0)}, r^{(0)}, H^{(0)}$
Sample Count, $n_s = 0$

select healthy-side IC parameters
start experiment
while $n_s < N_s$ **and** $\Delta r > \epsilon$ **do**
knee trajectory = []
while knee trajectory $\not\supset$ full gait **or** not faulty gait **do**
add data to knee trajectory ▷ Collect knee trajectory data
end while

if first gait **then**
extract state vector x ▷ Section 3.3.1
take action $u = \pi^{(i)}(x)$ ▷ Section 3.3.4 ▷ Equation 3.13
else
if gait is faulty **then**
set x^+ to punishment state ▷ Section 3.3.5
set u^+ to punishment action ▷ Section 3.3.5
else
extract next state vector x^+
take next action $u^+ = \pi^{(i)}(x^+)$
end if
▷ Equation 3.13
create sample tuple (x, u, x^+) and add to $\mathcal{D}$
iterate experience $(x, u) = (x^+, u^+)$
increment n_s

if $mod(n_s, N_b) == 0$ **then**
calculate $\tilde{A}$ and $\tilde{b}$ ▷ Equation 3.30, 3.31 using all $\mathcal{D}$
calculate $r^{(i+1)}$ ▷ Equation 3.37
$\Delta r \leftarrow |r^{(i+1)} - r|$
$r \leftarrow r^{(i+1)}$
end if
end if

if faulty gait **then**
terminates the simulation
reselect healthy-side IC parameters
restart simulation
end if
end while

in the r vector is plotted. The expected outcome from this plot is a decaying curve, ideally converging to zero.

To investigate the performance of the converged policy, it is required to take a step back to Equation 3.23, the bellman error for a sample. The implication of Equation 3.23 is that the expected cost of the current state-action pair is equal to the sum of the actual cost of the current state-action pair and the expected cost of the next state-action pair. The ultimate metric to evaluate the system is the average bellman error over all available samples, given in Equation 3.24. Two variations of this metric are investigated, online and offline.

The online BE is calculated using the samples collected until that update, as it is impossible to break the causality. However, the offline BE is calculated retroactively over all samples collected throughout the experiment. Nonetheless, it is expected both to converge with each other at the end of the experiment.

The distinction between the two settings surfaces due to insufficient samples in the early stages of the online setting. The low number of data leads to poor BE approximation and unreliable profiles. The online setting helps investigate the number of samples required to observe convergence in the BE. On the other hand, the offline setting allows to examine the ultimate performance improvement.

The initial controller has a diagonal H matrix in the offline setting. Looking at the Equation 3.42, one can realize that the second term is initially zero as $H_{xu}^{(0)}$ is all zero-matrix. On top of that, only quadratic actions are non-zero as $H_{u}^{(0)}$ is diagonal as well. Thus, the initial controller is the naive controller, with no adjustment actions on the nominal parameters. This aligns with the current active knee prosthesis control approach, wherein constant IC parameters are utilized for a given patient. Hence, the offline bellman error decrease represents the ultimate improvement introduced by the RL algorithm into the system. The percentage change is calculated with the following formula,

$$\Delta BE(\%) = 100 \times \frac{BE_i - BE_f}{BE_i} \tag{3.44}$$

where $\Delta BE(\%)$ is the percentage change, which indicates improvement when it is positive. BE_i and BE_f are the initial and final bellman error values, respectively.

4. Results

This thesis aims to achieve personal and adaptive active knee prosthesis control by introducing an RL framework to adjust the IC parameters. Thus, the experimentation primarily concentrated on assessing the improvement introduced by the RL framework. The novel approach utilized by this thesis is representing the gaits in the frequency domain using harmonics of the signal. As opposed to the time-domain representations used by previous studies, it is aimed to encode more information about the overall trend of the gait by adopting a frequency-domain approach. The secondary results are compiled concerning the validity of the frequency-domain approach by comparing it with the time-domain approach within the same experimental setup.

All results are presented in a three-columned fashion, each column representing an RL agent associated with a phase. The order is SWF, SWE, and SS from left to right. The reflections on the results are kept broad in this chapter but detailed in Chapter 5.

4.1. RL Framework Performance

This section utilized all the tools mentioned in Section 3.3.6 to analyze the RL framework's impact on performance. An extensive hyperparameter search has not been conducted in this study because our objective does not prioritize finding the optimal hyperparameters. This decision is based on the understanding that the simulation-to-real-life correspondence will likely be limited. Instead, a sufficiently good set of hyperparameters is utilized, presented in Table 4.1.

σ_N	1%
n	6
γ	0.7
η	0.7
μ	0.9
ρ	0.9
f_p	0.1
f_e	0.2
N_b	100
N_s	8000
ϵ	0.0001

Table 4.1.: Hyperparameters used in experiments.

The first evaluation tool is the convergence presented in Figure 4.1. This tool ensures the agent converges to a policy regardless of whether it is a good policy. To investigate the overall trend throughout a reasonable number of updates, ϵ kept small.

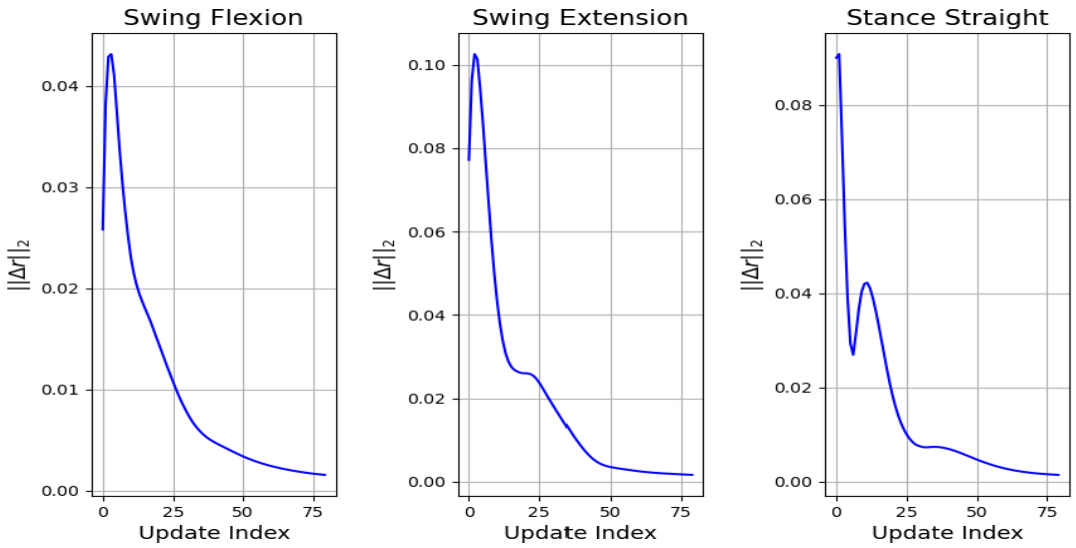


Figure 4.1.: Convergence of the algorithm for each RL agent.

Figure 4.1 indicates that all the agents converged within 60 iterations. Since batch size is 100 samples, each module is reasonably converged with 6000 samples.

The online and offline bellman errors (BE) are presented together in Figure 4.2 to emphasize the convergence behavior of the error. They are the results of the same experiment with Figure 4.1. The initial high fluctuation on online BE is expected, reflecting the poor approximation when less data is available. The convergence of the online BE occurs within $25 - 40$ iterations.

Although the magnitude of improvement varies from agent to agent, the decaying behavior of the offline BE is visible for each agent. Percentage improvements are presented in Table 4.2.

Agent	SWF	SWE	SS
Improvement	3.38%	16.74%	6.31%

Table 4.2.: Improvement introduced by RL framework for each agent.

4.2. Frequency vs. Time Domain Representations

In Section 3.3.1, it is mentioned that the previous works attempted to represent the gait in the time domain through its extrema as illustrated in Figure 3.5. To understand the influence

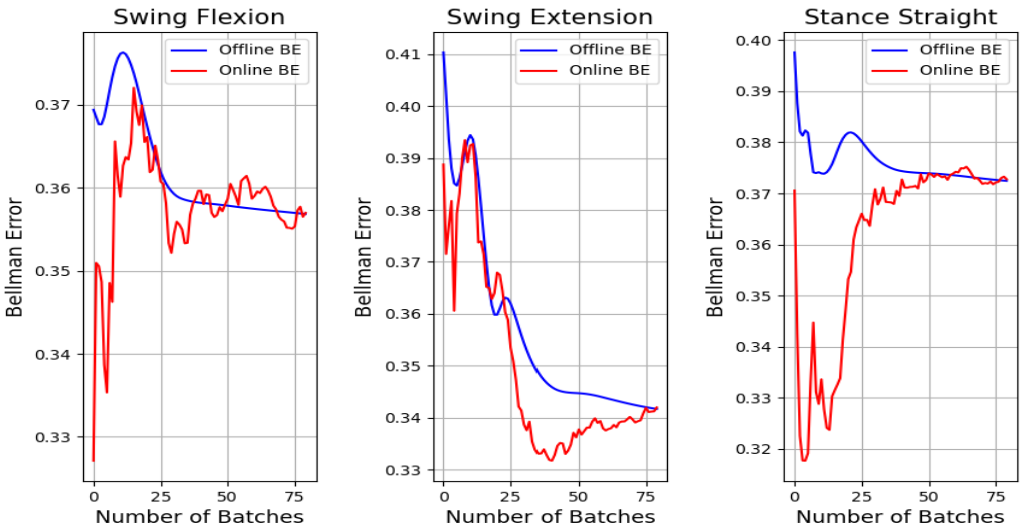


Figure 4.2.: The online and offline bellman error for each agent.

of the frequency domain approach proposed in this study, it is necessary to compare the performances of both approaches. As the main concern is performance, only the offline BE metric is used to conduct this comparison, and relative performances are investigated through the same plot.

For a controlled comparison, each hyperparameter in Table 4.1 is kept the same, but the gaits are represented via extrema points. The state vector with this time-domain approach can be constructed as follows,

$$x := \left[\Delta D_1, \Delta D_2, \Delta D_3, \Delta D_4, \Delta P_1, \Delta P_2, \Delta P_3, \Delta P_4, Sw_{SI}\right]^T \tag{4.1}$$

where ΔP_i and ΔD_i are defined as in Equation 4.2, and Sw_{SI} is identical to its definition in Equation 3.7.

$$\Delta P_i := \frac{P_i - \overline{P_i}}{\overline{P_i}} \qquad \Delta D_i := \frac{D_i - \overline{D_i}}{100} \tag{4.2}$$

where $\overline{P_i}$ and $\overline{D_i}$ are the target value and position of the extremum, respectively. P_i and D_i, on the other hand, are the simulation value and position of the corresponding extremum. Notice that the state definition given in 4.1 has nine features, as the knee trajectory is always supposed to have two peaks and two valleys. Thus, while comparing the two approaches, having a different number of features is inevitable due to the harmonic approach's flexibility.

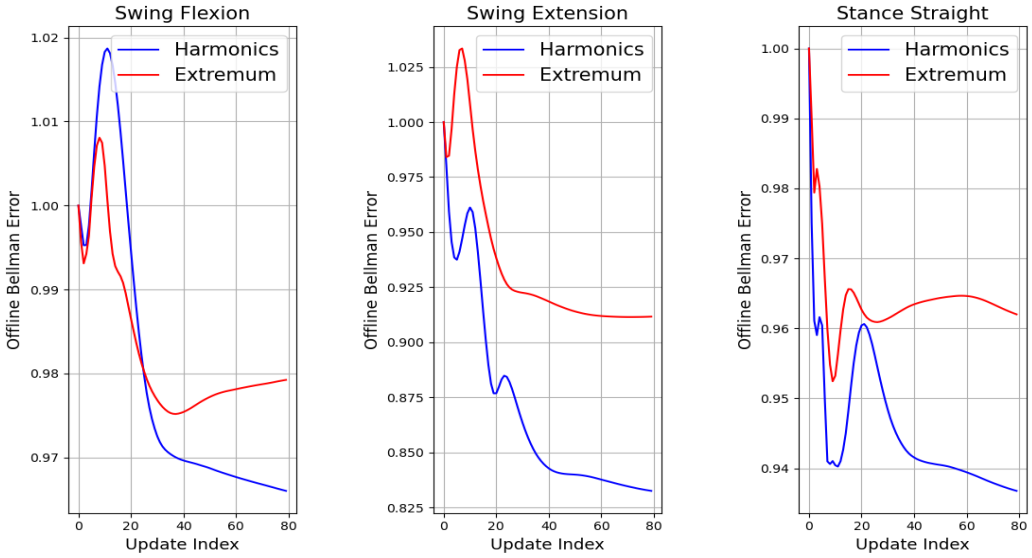


Figure 4.3.: Offline bellman error for frequency domain (blue) and time domain (red) state representations.

Changing the definition of the state vector directly affects the scale of the BE, as the cost defined in Equation 3.13 includes a state term. It is decided to normalize the offline BE by the initial value so relative performance can be seen clearly. Figure 4.3 presents the offline BE over the updates. Table 4.2 is also extended by the results from the extremum representation and shown in Table 4.3.

Agent	SWF	SWE	SS
Frequency Domain	3.38%	16.74%	6.31%
Time Domain	2.09%	8.82%	3.80%

Table 4.3.: Improvement introduced by RL framework for each agent by different state representations.

The clear superiority of the frequency domain representation can be seen in both Figure 4.3 and Table 4.3. It is unrealistic to expect the offline BE to vanish entirely due to the *parameterization error* introduced by the Q value approximation given in Equation 3.21. The error value converged by the frequency domain representation is lower, meaning that the overall performance of the controller is better than the controller trained through time domain representation.

5. Discussion

Results suggest that the convergence will occur within 60 iterations, corresponding to 6000 samples. The training requires around 6000 seconds ($\sim$ 100 minutes) of walking to converge. This is worse than the reported results in previous studies, primarily because of two reasons. First, the simulation employed for this thesis includes the ground reaction forces, which significantly increase the problem's complexity. Second, the state space used by the previous works had a lower dimension with the expense of losing full-gait information. Although the studies reported better convergence rates, they do not present a measure for performance.

If the convergence behavior persists in real-time implementation, it is unrealistic to ask a patient to walk for 100 minutes. In that case, the RL framework would require a more intelligent integration to the system. It should also be noted that the bellman error converges within $25 - 40$ iterations depending on the agent. This suggests the bellman error approximation is sufficiently good before the convergence of updates.

Regarding performance, there is an improvement for all the agents. The average bellman errors are decreased by 3.38%, 16.74%, and 6.31% for SWF, SWE, and SS agents, respectively. The magnitude of the improvements aligns with the complexity of the movements they represent. Swing extension, for example, corresponds to the second half of the swinging of the leg. In classical biomechanics, this phase is usually modeled as a double-pendulum, a relatively easy multibody system. This simplicity can justify the corresponding agent's significantly better performance. Similarly, due to its opposition to gravity, the Swing Flexion agent has less improvement than others. The trend is also observed with the time domain representations, with remarkably larger improvement on the swing extension agent.

Comparing the two representation approaches, the frequency domain representation that this paper proposed has significant superiority over the time domain representation. Frequency domain features achieve 1.29%, 7.92%, and 2.51% better performance in SWF, SWE, and SS phases, respectively. SWE agent achieves almost double the improvement with frequency domain representation, while SWF and SS have around 1.6 times the improvement over time-domain representations. This shows that the signal's harmonics are more appropriate features to represent the gait. This exciting result is not limited in this specific application. The robotics domain mainly deals with real-time periodic tasks, and this representation can be utilized in almost all robotic tasks that deals trajectory following with periodic signals.

5.1. Limitations

Real-time correspondence has always been the highest priority throughout the critical decisions of this thesis, especially during the creation of the environment. It is always intended to avoid the advantages of having a simulation environment and embrace the disadvantages.

Although some of them are mentioned throughout the report for different purposes, a few critical aspects regarding this approach are summarized below.

1. Healthy joints of an amputee individual can compensate for the missing limb through the contraction and relaxation of the muscles. In the environment, however, we have utilized constant IC parameters for all healthy joints, eliminating human factors and potentially improving the process.
2. After each termination, healthy side joint IC parameters are chosen with a Gaussian Noise on the base IC parameters. This allows the introduction of variation in data and loosely demonstrates the different physiological attributes of different individuals. The real-time implementation, however, is expected to train the same user, where the variety will naturally occur through the variation across gaits and the healthy side joints are expected to have more consistent behavior.
3. The limitations of the simulation prevent the continuous walking action during the experiments, and the simulation has been restarted as faulty gaits occurred. This fact directly affects the training process as the long-term effects of a particular action cannot be observed.
4. The faulty gaits are defined very broadly to avoid forcing the model to have predefined gaits. This results in a considerable variation from the healthy knee trajectories and a more challenging learning process. The real-life amputee knee trajectories with active devices have some standards and are much closer to healthy knee trajectories.

Despite the generic challenges of real-time implementation, the promising outcomes and the above-listed differences encourage the pursuit of such implementation.

5.2. Future Works

Although the results suggest that this framework is already promising for achieving a personal and adaptive controller, room for improvement is inevitable.

Throughout the thesis, all algorithmic choices aimed to reduce complexity and maintain explainability. Using quadratic basis functions is one example, whereas any universal approximator could have been utilized. More careful basis function choices can help the algorithm to capture more fundamental information.

For this study, the target trajectories are chosen to be healthy human gait trajectories. However, amputee walking and able-bodied walking does not have one-to-one correspondence. Thus, the real-life comfort of the algorithm still needs to be investigated through clinical studies. A possible future direction can be a method to extract personal target trajectories. A musculoskeletal simulation for amputees would be a handy tool to extract target trajectories and could also be extended to other ambulation modes. Such a simulation would be invaluable in the active knee prosthesis domain.

6. Conclusion

This thesis aims to utilize an RL framework to achieve personal and adaptive active knee controllers using a novel frequency domain state representation approach. This research encompasses a complete proof of concept study from creating the environment to evaluating the proposed algorithm. In Chapter 1 and Chapter 2, the problem and the previous approaches have been introduced. Later in Chapter 3, the system is mathematically described, and the RL framework has been thoroughly explained by mentioning the motivations behind key decisions. Chapter 4 is dedicated to the results of the experiments, and results are reflected in Chapter 5.

The thesis revolves around two main hypotheses. First, the proposed reinforcement learning framework is suitable for learning the system dynamics and taking action accordingly. The second is that walking gait, as an inherently periodic task, can be better represented by frequency domain features.

Through simulations, the proposed RL framework has proven to be a potential solution to achieve personal and adaptive controllers. The RL approach decreased each agent's average bellman error. Especially for SWE, a staggering 16.74% improvement has been achieved. The frequency domain state representation has also proven superior to the time domain representations. Again for SWE, frequency-domain features achieved double the performance with respect to time-domain features. On top of it, this approach is not only limited to active knee prosthesis control but can be applied to many systems that deal with periodic trajectory-following tasks.

As an initial attempt to integrate RL into the active knee prosthesis controller, the results are already promising. One possible research direction is enhancing performance with personalized cost functions or target values, while another can be increasing the complexity of the underlying model and using tailored basis functions. Another concrete leap for the prosthesis domain can be investing time and effort in a high-fidelity musculoskeletal model to increase the simulation to the real-time correspondence of such research.

A. CVOXPT Implementation

The convex optimization library *cvoxpt* uses the following notation.

$$\min_{x} \ \frac{1}{2}x^T P x + q^T x \tag{A.1}$$

$$\textit{subject to} \quad Gx \leq h \tag{A.2}$$

where the correspondence to this project is as follows,

$$x \leftarrow u, \ \ P \leftarrow \frac{1}{2}H_u^{(i)}, \ \ q^T \leftarrow x^T H_{xu}^{(i)} \tag{A.3}$$

$$G \leftarrow \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ -1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & -1 \end{bmatrix} \quad h \leftarrow \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} \tag{A.4}$$

Notice that the inequality constraints lead to a solution in $[-1, 1]$, whereas the adjustment was designed to be within $\pm 10\%$. Appropriate scaling is applied to scale the input to $[-10, 10]$.

Bibliography

Acher, M. and Esnault, F. (2016). Large-scale analysis of chess games with chess engines: A preliminary report. *arXiv preprint arXiv:1607.04186*.

Alili, A., Nalam, V., Li, M., Liu, M., Si, J., and Huang, H. (2021). User controlled interface for tuning robotic knee prosthesis. pages 6190–6195. cited By 2.

Baird, L. (1995). Residual algorithms: Reinforcement learning with function approximation. In *Machine Learning Proceedings 1995*, pages 30–37. Elsevier.

Bellman, R. E. (1954). *The Theory of Dynamic Programming*. RAND Corporation, Santa Monica, CA.

Bertsekas, D. P. (2011). Temporal difference methods for general projected equations. *IEEE Transactions on Automatic Control*, 56(9):2128–2139.

Boyd, J. E. and Little, J. J. (2005). Biometric gait recognition. In *Advanced Studies in Biometrics: Summer School on Biometrics, Alghero, Italy, June 2-6, 2003. Revised Selected Lectures and Papers*, pages 19–42. Springer.

Gao, X., Si, J., Wen, Y., Li, M., and Huang, H. (2021). Reinforcement learning control of robotic knee with human-in-the-loop by flexible policy iteration. *IEEE Transactions on Neural Networks and Learning Systems*, 33(10):5873–5887.

Gao, X., Si, J., Wen, Y., Li, M., and Huang, H. H. (2020). Knowledge-guided reinforcement learning control for robotic lower limb prosthesis. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 754–760. IEEE.

Gao, X., Wen, Y., Li, M., Si, J., and Huang, H. (2019). Robotic knee parameter tuning using approximate policy iteration. In *Cognitive Systems and Signal Processing: 4th International Conference, ICCSIP 2018, Beijing, China, November 29-December 1, 2018, Revised Selected Papers, Part I 4*, pages 554–563. Springer.

Huang, H., Crouch, D. L., Liu, M., Sawicki, G. S., and Wang, D. (2016). A cyber expert system for auto-tuning powered prosthesis impedance control parameters. *Annals of biomedical engineering*, 44:1613–1624.

Lagoudakis, M. G. and Parr, R. (2003). Least-squares policy iteration. *The Journal of Machine Learning Research*, 4:1107–1149.

Lenzi, T., Cempini, M., Hargrove, L., and Kuiken, T. (2018). Design, development, and testing of a lightweight hybrid robotic knee prosthesis. *The International Journal of Robotics Research*, 37(8):953–976.

Li, M., Gao, X., Wen, Y., Si, J., and Huang, H. H. (2019). Offline policy iteration based reinforcement learning controller for online robotic knee prosthesis parameter tuning. In *2019 International conference on robotics and automation (ICRA)*, pages 2831–2837. IEEE.

Li, M., Liu, W., Si, J., Stallrich, J., and Huang, H. (2022a). Hierarchical optimization for control of robotic knee prostheses toward improved symmetry of propulsive impulse. *IEEE Transactions on Biomedical Engineering*.

Li, M., Wen, Y., Gao, X., Si, J., and Huang, H. (2022b). Toward expedited impedance tuning of a robotic prosthesis for personalized gait assistance by reinforcement learning control. *IEEE Transactions on Robotics*, 38(1):407–420. cited By 11.

Liu, W., Wu, R., Si, J., and Huang, H. (2022). A new robotic knee impedance control parameter optimization method facilitated by inverse reinforcement learning. *IEEE Robotics and Automation Letters*, 7(4):10882–10889.

Luengas, L. A., Camargo, E., and Sanchez, G. (2015). Modeling and simulation of normal and hemiparetic gait. *Frontiers of Mechanical Engineering*, 10:233–241.

Minh, V. T., Tamre, M., Musalimov, V., Kovalenko, P., Rubinshtein, I., Ovchinnikov, I., and Moezzi, R. (2020). Model predictive control for modeling and simulation of human gait motions. *International Journal of Innovative Technology and Interdisciplinary Sciences*, 3(1):326–345.

Pfeifer, S., Vallery, H., Hardegger, M., Riener, R., and Perreault, E. J. (2012). Model-based estimation of knee stiffness. *IEEE transactions on biomedical engineering*, 59(9):2604–2612.

Ranzani, R. (2014). Adaptive human model-based control for active knee prosthetics. Master's thesis, ETH Zürich.

Rouse, E. J., Hargrove, L. J., Perreault, E. J., and Kuiken, T. A. (2014). Estimation of human ankle impedance during the stance phase of walking. *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, 22(4):870–878.

Shandiz, M. A., Farahmand, F., Osman, N. A. A., and Zohoor, H. (2013). A robotic model of transfemoral amputee locomotion for design optimization of knee controllers. *International Journal of Advanced Robotic Systems*, 10(3):161.

Silver, D., Schrittwieser, J., Simonyan, K., Antonoglou, I., Huang, A., Guez, A., Hubert, T., Baker, L., Lai, M., Bolton, A., et al. (2017). Mastering the game of go without human knowledge. *nature*, 550(7676):354–359.

Simon, A. M., Fey, N. P., Finucane, S. B., Lipschutz, R. D., and Hargrove, L. J. (2013). Strategies to reduce the configuration time for a powered knee and ankle prosthesis across multiple ambulation modes. In *2013 IEEE 13th International Conference on Rehabilitation Robotics (ICORR)*, pages 1–6. IEEE.

Simon, A. M., Ingraham, K. A., Fey, N. P., Finucane, S. B., Lipschutz, R. D., Young, A. J., and Hargrove, L. J. (2014). Configuring a powered knee and ankle prosthesis for transfemoral amputees within five specific ambulation modes. *PloS one*, 9(6):e99387.

Singh, B., Kumar, R., and Singh, V. P. (2022). Reinforcement learning in robotic applications: a comprehensive survey. *Artificial Intelligence Review*, pages 1–46.

Sup, F., Bohara, A., and Goldfarb, M. (2008). Design and control of a powered transfemoral prosthesis. *The International journal of robotics research*, 27(2):263–273.

Sup, F., Varol, H. A., and Goldfarb, M. (2010). Upslope walking with a powered knee and ankle prosthesis: initial results with an amputee subject. *IEEE transactions on neural systems and rehabilitation engineering*, 19(1):71–78.

Sup, F., Varol, H. A., Mitchell, J., Withrow, T. J., and Goldfarb, M. (2009). Preliminary evaluations of a self-contained anthropomorphic transfemoral prosthesis. *IEEE/ASME Transactions on mechatronics*, 14(6):667–676.

Todorov, E., Erez, T., and Tassa, Y. (2012). Mujoco: A physics engine for model-based control. In *2012 IEEE/RSJ international conference on intelligent robots and systems*, pages 5026–5033. IEEE.

Towers, M., Terry, J. K., Kwiatkowski, A., Balis, J. U., Cola, G. d., Deleu, T., Goulão, M., Kallinteris, A., KG, A., Krimmel, M., Perez-Vicente, R., Pierré, A., Schulhoff, S., Tai, J. J., Shen, A. T. J., and Younis, O. G. (2023). Gymnasium.

Tucker, M. R., Shirota, C., Lambercy, O., Sulzer, J. S., and Gassert, R. (2017). Design and characterization of an exoskeleton for perturbing the knee during gait. *IEEE Transactions on Biomedical Engineering*, 64(10):2331–2343.

Viteckova, S., Kutilek, P., Svoboda, Z., Krupicka, R., Kauler, J., and Szabo, Z. (2018). Gait symmetry measures: A review of current and prospective methods. *Biomedical Signal Processing and Control*, 42:89–100.

Wang, D., Liu, M., Zhang, F., and Huang, H. (2013). Design of an expert system to automatically calibrate impedance control for powered knee prostheses. In *2013 IEEE 13th International Conference on Rehabilitation Robotics (ICORR)*, pages 1–5. IEEE.

Watkins, C. J. C. H. (1989). Learning from delayed rewards.

Wen, Y., Gao, X., Si, J., Brandt, A., Li, M., and Huang, H. (2019). Robotic knee prosthesis real-time control using reinforcement learning with human in the loop. *Communications in Computer and Information Science*, 1005:463–473. cited By 3.

Wen, Y., Li, M., Si, J., and Huang, H. (2020). Wearer-prosthesis interaction for symmetrical gait: A study enabled by reinforcement learning prosthesis control. *IEEE transactions on neural systems and rehabilitation engineering*, 28(4):904–913.

Winter, D. A. (2009). *Biomechanics and motor control of human movement*. John wiley & sons.

Wu, R., Yao, Z., Si, J., and Huang, H. (2022). Robotic knee tracking control to mimic the intact human knee profile based on actor-critic reinforcement learning. *IEEE/CAA Journal of Automatica Sinica*, 9(1):19–30. cited By 7.

Colophon

This document was typeset using LaTeX, using the KOMA-Script class `scrbook`. The main font is Palatino.