UTS UNIVERSITY OF TECHNOLOGY SYDNEY

Improving Performance of EMG-driven Ankle-Foot Orthosis Through Passivation & Reinforcement Learning

by Bradley Beck

Thesis submitted in fulfilment of the requirements for the degree of

Doctor of Philosophy (C02018)

under the supervision of Associate Professor Steven Su, & Professor Joanne Tipper

University of Technology Sydney
Faculty of Engineering and Information Technology

April 2023

I, Bradley Beck, declare that this thesis is submitted in fulfilment of the requirements for the award of Doctor of Philosophy, in the School of Biomedical Engineering at the University of Technology Sydney.

This thesis is wholly my own work unless otherwise referenced or acknowledged. In addition, I certify that all information sources and literature used are indicated in the thesis.

This document has not been submitted for qualifications at any other academic institution.

This research is supported by the Australian Government Research Training Program.

Signature: Production Note: Signature removed prior to publication.

Date: 2023/03/03

Acknowledgements

Australian Government Research Training Program Stipend for providing funding throughout the project.

University of Technology Sydney for providing the equipment and resources necessary to complete the project.

Principal supervisor Associate Professor Steven Su for assistance with the technical aspects of the project, and always being available when needed.

Co-supervisor Professor Joanne Tipper for guidance on report, project management, and assessment matters, and providing an outside perspective.

Robert Karimagako and GreenGrid Energy Systems for providing an internship opportunity to implement learned skills in a practical setting.

Australia New Zealand Control Conference 2022 for publishing paper based on work in this thesis (Comparison of Constant PID Controller and Adaptive PID Controller via Reinforcement Learning for a Rehabilitation Robot).

This research was supported in part by the UTS Interactive High Performance Computing facility.

Abstract

The human musculoskeletal system is known to weaken as we age, with deterioration more prominent at the limb joints. As the lowest joint of the body, the ankle joint must support the full weight of the body during all activities and movement tasks, making it very susceptible to injury and cartilage degeneration over time. The use of robotic rehabilitation devices for assisting in physiotherapy is not a new concept but has been steadily improving over the past decades, and discussion of how improvements can be made through electromyography, control systems, and artificial intelligence is the focus of this project.

Utilising passivity within control systems has shown to remove redundancy requirements and allows less conservative controller design, improving performance. Passivating any system through an automated script that utilizes linear matrix inequalities and optimisation techniques allows this efficient design, while also minimizing perturbation of the original system to retain reliable simulation and control. An optimal input-output pairing is also revealed through this analysis, allowing improved robotic design. Originally this technique was assumed to be applicable to all systems but was found to be more restrictive during research. A lemma was proposed to encapsulate the limitations which stemmed from the relative degrees of the transfer functions within the system.

Reinforcement learning developed an adaptive PID and adaptive admittance controller that could better handle uncertain interactions within a human-robot interfacing system. The adaptive PID controller was successful in producing better reference tracking abilities than traditional PID controllers in changing environment situations. The adaptive admittance controller was also shown to be a viable approach, but was generally outperformed by a switching controller.

Reinforcement learning was also used to try and classify motions from electromyography signals to act as the input to a rehabilitation robot. Using secondary data, convolutional neural networks were able to perform this classification to a high accuracy, and produce a feature dataset that could be used to classify motions. The aim of the reinforcement learning was to identify the essential features needed for classification, and prune the rest to create a more efficient categorisation system. Results showed reinforcement learning was not well suited for the task of state space reduction, and the more traditional machine learning techniques provided high enough accuracy results that reinforcement learning was deemed unnecessary for this task.

Overall, this project aimed to combine the use of admittance control, reinforcement learning, and passivity-based process control to improve safety and ease-of-use of wearable ankle-foot orthoses.

Keywords: Ankle Rehabilitation, Ankle-foot orthosis, EMG Signals, Adaptive admittance control, Reinforcement learning, Passivity-based process control, Reinforcement learning-based classification

Table of Contents

Chapter 1 - Introduction

1.1. Introduction

Advancements in medical technologies are increasing the average life expectancy of almost every country's population and improving the quality of life of their citizens. While this is undoubtedly a benefit to humans as a species, an aging population leads to new problems that must be addressed, especially for countries with falling birth rates. One of the largest problems is the degradation of lower limb functionality throughout life; as bodies age they become more frail and susceptible to injury, disease, or simply become weaker due to lifestyle changes. A reduction to freedom of mobility will in most cases cause a decrease to the quality of life of an individual, as noted by the correlation between functional independence and quality of life (Araujo et al. 2016). For many people, lower limb issues are not a matter of if, but a matter of when, as approximately 1 in 9 Australians will suffer some form of osteoarthritis (OA) and over 50 000 requiring total knee replacement surgery with OA cited as the principal diagnosis (Health and Welfare 2020). This value has been increasing every year, along with total hip replacements, and only identifies cases where medical intervention was deemed appropriate; many others are also likely suffering from less severe but manageable conditions. It is also only one form of lower limb disability; other conditions such as hemiplegia/hemiparesis (a side effect caused by stroke, which is also extremely prevalent in Australia) is also correlated with an increase in age (Kelly-Hayes 2010). Diseases such as stroke or OA or lower limb injuries have the potential to cause serious impairments to an individual's ability to perform locomotive tasks. It is therefore important to develop solutions to these problems that would allow those who have lost their lower limb functionality to recover to a point where they are no longer inconvenienced. These solutions may come in many different forms, such as surgery, but all methods will inevitably require some form of physical therapy to help the patient. For this reason, this thesis focuses on the physical therapy component of the multi-stage solution for lower limb recovery.

For injuries or arthritis specifically a surgical procedure known as total joint replacement can be performed to improve mobility of the affected joint. Lower limbs benefit from this procedure greatly as the hip, knee, and ankle joints support the weight of the upper body. With knee and hip replacements being so common it is important to look at why ankle replacements do not also follow this trend, since it can be found that ankle injuries are just as common, if not more

so. The lower reported cases of ankle surgeries can be explained by a combination of several factors. The belief that ankle injuries are less severe, and therefore not worth investing time into professional recovery may reduce the number of people who seek medical assistance. The biggest issue, however, is that there is no reliable assistance that can be implemented for long term improvements. Unlike total knee or hip replacements, total ankle replacements are unable to last for extended time periods, often failing and requiring additional surgical intervention within several years (Stratton-Powell 2018). Hip and knee replacements are reported to remain successful (require no follow-up intervention) for over 10 years in greater than 90% of the patients, yet for ankle replacements only 66-84% (variation based on country) remain successful after 10 years (Perry et al. 2022). Due to the less reliable medical resolutions, physiotherapy becomes much more important for ankle rehabilitation, which gives justification for the focus of this paper being strictly ankle rehabilitation over other lower limb joints. By designing a control system for an ankle-foot orthosis (AFO) that is able to efficiently monitor, adjust, and assist ankle motions of a subject, it is possible to achieve several advantages of physiotherapy without requiring a physiotherapist present, effectively reducing the workload on the institution. The workload of the physiotherapy industry is expected to increase alongside the aforementioned aging population. Technologies that can reduce the industry burden should be invested in now, before they become overwhelmed and the quality of rehabilitation offered is negatively affected. There are three types of wearable robotic devices that are commonly referred to in literature: orthoses, exoskeletons, and prostheses. Although definitions for these terms are somewhat malleable and change between papers, the definitions used within this paper are that prostheses replace a portion of a limb, orthoses support a limb with a physical ailment, and exoskeletons enhance a fully functioning limb.

With an understanding of why advancements to rehabilitation are necessary, how to achieve these goals becomes the next focus. The design of any robotics system will always contain both a mechanical/hardware aspect and a software aspect, where both aspects will be fundamentally related and will affect the design process and final outcome. The hardware used in rehabilitation robotics is important to consider as it will be in direct contact with humans, making safety a primary concern. Although there are a variety of physical options that can be implemented, the focus at this stage of the project is on the control engineering of the orthosis device, which falls in the category of software development. Hardware design will not be intensely considered and any effects the hardware would have on the software are temporarily ignored. The literature review will cover a broad scope of assistance devices ranging from treadmill-based rehabilitation

to end-effector robotics to wearable orthosis design. The control theory of the system determines how the system reacts to a specific input signal which represents a state. An orthosis that is capable of determining the users intended motion and produces an assistive force to reach the predicted goal can be used to guide ankle motions to what is considered a healthy gait. Through repeated use, similar to physical therapy, an ankle rehabilitation robot can assist the recovery of ankle disabilities. This could be applied for post-surgery patients similar to how rehabilitation therapy is assigned currently, but may also be used as a preventative measure for OA patients to help reduce the forces in the ankle joint. Osteoarthritis is due to the degeneration of joint cartilage, causing swelling and pain. By both reducing the forces and helping guide the user to a gait that helps evenly distribute the forces, the user may be able to slow further degeneration which is caused by the uneven force distributions. Ramsey and Hamilton (1976) showed that a 1 mm shift in lateral motion for the talus, which can occur due to injuries and fractures, can lead to a 42% change in loading pressure. Over long periods of time this loading pressure is what causes OA, so by redistributing pressure and providing joint torque assistance the degradation of cartilage may be reduced. The redistribution would be implemented by a control system that is able to measure forces and produce appropriate actuator motions to redirect the ankle orientation. This can be done through traditional control theory or it may utilize newer technologies such as artificial intelligence. Both will be discussed throughout this report and compared within the literature review.

For subjects with reduced muscle force production capabilities, the orthosis must be able to supply enough additional torque for the ankle joint to reach its intended orientation. For cases where the rehabilitation device must do all or most of the work, the process is described as “passive rehabilitation”. This is important at the early stages of rehabilitation where patients do not have enough muscle strength to move unassisted. Alternatively, when the device is able to offer minimal assistance and all motion can be produced from the muscles, the process is known as “active rehabilitation” (Meng et al. 2015). Active rehabilitation is much more effective at motor training, improving motor plasticity (Lotze et al. 2003), and brain plasticity (Rossini and Dal Forno 2004), so the long-term goal of all rehabilitation should be to transition from passive to active rehabilitation until assistance is no longer necessary. With long-term progression in mind, techniques that require large amounts of data collection become more viable, such as using machine learning to compose a dynamical model of the specific user’s motions. Due to the nonlinear properties of human motions, changing in reference to the environment and varying from subject to subject, the implemented control system must have some form of adaptability.

Some proven beneficial examples of these approaches include fuzzy tuning (Yang et al. 2016), adaptive admittance control (Yao et al. 2018), and switching regimes (Artemiadis and Kyriakopoulos 2011). The necessity to customise the control system of a robotic rehabilitation opens up a large field of machine learning to potentially utilise for system improvements.

As advancements occur in hardware fields such as energy storage/efficiency, sensory technology, and actuation technology, the design of control systems must also continue to evolve and improve. A generalised control framework is provided by Tucker et al. (2015) to display the control system as a hierarchical structure and determine what each layer provides to the overall functionality. Although the control systems covered by the literature review will not be described in this hierarchical structure, it is beneficial to consider how all the physical components interact and identify what aspects of the control system can be associated to that specific relation. This report hopes to cover the structure and implementation of control systems that will allow an AFO to provide assistive physical rehabilitation through the literature review presented below.

1.2. Summary of Thesis

To identify potential improvements and effectively develop the control systems of a generic rehabilitation robot it is important to have a deep understanding of existing technologies. A literature review of the different approaches used in the past illuminates the technical fields that are most important for further development. This literature review is presented in Chapter 2 and details the less-technical aspects of the overall project such as device comparisons or practical implementation considerations. From the information collected, the experimental tasks were formulated into different stages that would allow compartmentalised activities and research to be performed in such a way that each experiment block contributed to the overarching project but did not depend on previous or future experiments to avoid progression bottlenecking. Each experimental block was documented in an individual chapter, making each experimental chapter independent from one another and should (in theory) be comprehensible as a standalone report. More technical details are recorded in smaller chapter-specific literature reviews when appropriate.

Chapter 3 is the first experimental block chapter which focuses on passivity-based process control and the control configuration selection of multi-input multi-output control systems. More specifically, this chapter aims to use optimisation engineering to guarantee a system meets the mathematical definition of a passive system for its positive applications to system

stability and human safety. The control configuration selection dictates the links between system inputs and system outputs, and the effects these pairings have on the overall system. This can be measured through mathematical conditions and recordings, which allows optimisation tools to be used once again to determine the best input-output pairing for guaranteeing system passivity with minimal changes. A mathematical lemma was produced to determine whether any given control system could apply this technique to find a meaningful solution. The inspiration for this chapter was utilising the mathematical certainty that a passive system coupled with a passive system will remain passive, and that the human ankle is passive under most circumstances. If the robotic device can then also be forced to be passive through optimisation, then the human-machine coupled system will also be passive and will allow the system to be much easier to control overall. The control configuration selection was inspired by the use of EMG-to-force decoders and their susceptibility to crosstalk between muscle activations. By determining which muscle cluster should contribute most to different robotic motions the control of the system can be more conservative and susceptible to unpredicted movements. Choosing the optimal control configuration selection is equivalent to selecting the best muscle cluster to robot movement pairing.

Chapter 4 focuses on the implementation of reinforcement learning in control systems contexts. Reinforcement learning can improve the performance of traditional controllers, or act as a replacement entirely. Both methods were attempted to compare system ability to track a reference signal that would dictate the movement of a rehabilitation device constructed by former students, known as the PaddleBot. This comparison aimed to ascertain which method was best suited for rehabilitation, or at the very least determine if they were viable. Using reinforcement learning to replace the traditional controller is sometimes referred to as “end-to-end” learning, which was documented but was not the focus of the chapter. Using reinforcement learning to create an “adaptive controller” was the true focus, and was performed for both a PID controller and an admittance controller for several different environments. The goal was to determine under which operating circumstances the adaptive controllers outperformed their more traditional constant counterparts. The inspiration for researching adaptive controllers is the constantly changing interactions between humans and machines that will require changes in the control loop if optimal performance is desired. The forces a human can exert will vary greatly between subjects, but will also vary over the course of an exercise routine, so it was believed that an adaptive controller that could adjust as necessary would be important.

Chapter 5 implemented more reinforcement learning to convert electromyography (EMG) signals into user intended motions. The use of bio-signals for machine learning classification is not a new idea and has been implemented many times over, however reinforcement learning for this purpose was much less prevalent throughout the literature. Experiments were performed to compare reinforcement learning to a convolutional neural network for classification of EMG data that was sourced from the internet and modified for the task at hand. The inspiration from this chapter once again stemmed from the changing bodily output during exercises and the necessity to categorise motions at any stage. The more traditional classification methods will depend entirely on the training and validation stages of development, so attempts to utilise reinforcement learning may hold some benefits in long term applications. The chapter acts as a discussion for the limitations of reinforcement learning and what components of a rehabilitation device may or may not exploit this technology. Although the equipment for EMG collection was available and a script was created for real-time recording, the secondary dataset was believed to result in more reliable outcomes for the purposes of the chapter due to its rich diversity of participants and proven applications in machine learning.

Despite the title of this thesis isolating the ankle as the point of interest, the purpose of identifying points of improvement within robotic control systems can be applied to a wider scope. The ankle remains the dominant focus for its passive properties discussed in the literature, however other joints may also possess this characteristic and will be able to benefit from these developments as well, with the objective of improving quality of life for those dependent on rehabilitation robotics. With this in mind, the applications of each chapter and their connections as a whole forms a stronger, more cohesive narrative. The input-output pairing of chapter 3 was designed with minimising muscle crosstalk, but the muscle movements (the outputs) were fairly limited due to the hinge nature of the ankle. Ankle adduction/abduction, and medial/lateral rotation motions can occur, and as such the method can still improve rehabilitation by resisting these motions. Other joints such as the wrist may be more directly able to utilize the new techniques, as muscle activation per finger or unique hand motions enable more obvious output changes.

It is for similar reasoning that hand motions were used for EMG classification in chapter 5 (along with the availability of the data). The novelty of the chapter does not come from classification, but the analysis of reinforcement learning in the optimisation of classification. As the kinds of movement are not a key variable in this analysis, the difference of upper limb motions over lower limb motions will not negatively affect results.

Finally, a discussion of each chapters results are presented in chapter 6. How each experimental block contributes to the project overall, how the results can be combined, and what this research can potentially lead to, are all reviewed to conclude the dissertation.

Chapter 3 benefits from the reinforcement learning work of chapter 4 by eventually using reinforcement learning to confirm the control configuration selection, or search for a configuration that leads to a smaller deviation in the system. The work in chapter 5 would also ideally be able to simplify plant models represented by nonlinear function approximators to a point where they could also apply the optimiser method of input-output pairing, although this may not be possible due to the black box nature of neural networks and nonlinear control structure.

Chapter 4 benefits from chapter 3 by pre-establishing the optimal input-output pairing of the plant, as the design of the control system will be dependent on this structure. By identifying the best pairing to guarantee a passive system as close as possible to the original system, the adaptive controllers designed in chapter 4 will be able to control the real-world robotic device despite being trained through simulations. The benefits of chapter 5 show that limb motions can be classified in real-time, which would allow reference signals to the adaptive controllers to be determined by participant current movement intent, as well as further experience into reinforcement learning and optimising the training structure. Although not achieved, the simplification of the classifier would allow faster updating and remove delays within the system, which would in turn positively benefit stability of the control system.

Chapter 5 benefits from chapter 3 by using the decentralised inputs as the source of features for the classifier, rather than indistinguishable features from a convolutional neural network, potentially improving the chances of feature reduction. Chapter 4 show the similarities between upper and lower limb data and open a direct path to applying adaptive control to upper limb rehabilitation.

Each of these chapter links build the foundation to vast amounts of future work that may be researched. Although the chapters are only tangentially related, they should all work in tandem and complement each other to result in a more robust control system that uses optimal designs.

All research chapters rely on experiments that do not involve human participants, instead favouring simulated data within chapters 3 and 4, and secondary datasets for chapter 5. This lack of user participation is predominantly due to the COVID-19 lockdowns, making human interactions completely unavailable for majority of the project both legally and ethically.

Chapter 2 - Literature Review

2.1. Existing Technologies

There are many different technologies that go into the proper design of a rehabilitation robot, ranging from robot sensor design, to control theory, to material sciences. Each component contributes something fundamental to the overall system design to create a functioning product. A short review of the already existing robotic devices as a whole is provided below, followed by a review of one specific method of human-machine interfacing in the form of electromyography.

2.1.1. Rehabilitation Robotics

Devices created to assist a human in performing physical tasks have existed for most of human history. A traditional pirate's peg-leg is an example of very early artificial limbs and would fall into the category of rehabilitation device. As material technologies improved so did these prosthetics, and the inclusion of computer technologies have transitioned the field again by allowing more intelligent and natural interactions. Rehabilitation robotics has been in development for over half a century with papers published in the 1960s that utilised powered orthoses. Naturally over time publications became more frequent and advanced, and established effective forms of human-robot interaction through different feedback channels (Meng et al. 2015). Most of the early stages of rehab robotics were location restrictive as they had to be anchored to a specific position, whether it be for power supply reasons, feedback monitors, or the device was simply too large. This limitation is still pervasive today, but has been lessened by improvement to battery technology, computing power and shrinkage, or may simply be non-applicable to certain kinds of devices such as treadmill based devices. Both mobile and non-mobile devices were researched for the project, however mobile devices unsurprisingly align closer to the goals of an ankle-foot orthosis device. It has been recorded that wearable robots are more suitable for gait training, while platform robots are better suited for ankle exercises only (Zhang, Davies, and Xie 2013).

Non-mobile devices are designed to exist within professional physiotherapy environments to be used in conjunction with a professional, rather than a replacement. They provide a well-

structured set of exercises that help both user and professional complete the tasks in a reliable, repeatable manner while also collecting potentially useful user data. One common device is the MIT-MANUS system (Hogan et al. 1992) developed in the early 1990s by the Massachusetts Institute of Technology for upper limb hemiparesis patients. The MANUS helped guide the user through visual feedback to complete horizontal arm movements along with assistive or resistive forces that could change depending on the user. A double blind study showed this simple act of using the robotic device that assisted in motions increased elbow and shoulder motor function (Hidler et al. 2005). These improvements were dissipated at later stages, but the robotic device still produced a more rapid improvement (which is very important for faster patient turnover and avoiding overwhelming the facility). For similar results for the lower limb, the Lokomat is a treadmill-based rehabilitation device that has had success for several decades (Colombo et al. 2000). As stroke survivors cannot support themselves immediately after the stroke, a device that supports their body weight and assists them in even minor leg movement as soon as possible has enormous benefits to rehabilitation speed. Other treadmill-based devices have been developed with similar ideas, as not requiring a therapist to help support the subject allows training times to be increased dramatically. The internal control of these devices have also developed over time, with techniques such as using a moving window rather than an exact reference having revealed improvements to usability and more active participation from subjects (Duschau-Wicke et al. 2010). All the devices shown above have produced results showing the benefits of rehabilitation robotics, however they are all large machines suitable only for rehabilitation centres.

Figure 2.1. LokomatPro: A moving treadmill based rehabilitation robot developed by Hocoma

Mobile devices better fill the previously discussed definitions of prostheses, orthoses, and exoskeletons. The primary goal of these devices is to attach to a person and allow them to perform their regular daily tasks with assistance, rather than depending on a predetermined exercise routine with a physiotherapist. It may remain attached temporarily to assist in rehabilitation, or it may remain as a more permanent solution (as a prosthetic would). The Hybrid Assistive Leg (HAL) is a lower limb orthosis that was developed for rehabilitation by providing assistive torques to the hip and knee, but remained mobile. A design similar to the HAL, as seen in Figure 2.2, but with more focus on the ankle rather than hip and knee was the imagined device while constructing the overall project. Its use of inertial measurement sensors and ground force sensors allow personal gait details to be collected for future improvements to the system design or simply customising the rehabilitation process for the individual. HAL-5 includes EMG electrodes, so using EMG signals to predict user intent and assist in motions is already an option for control assistance. The commercialisation of HAL-5 occurred around 2013, although HAL-5 supports the upper body as well as the lower limbs, so it has more applications than the HAL-3 which is exclusively lower limb. The internal control of the device is not publicly available, but the existence and success of the HAL products give credence to an ankle-focused device of similar nature.

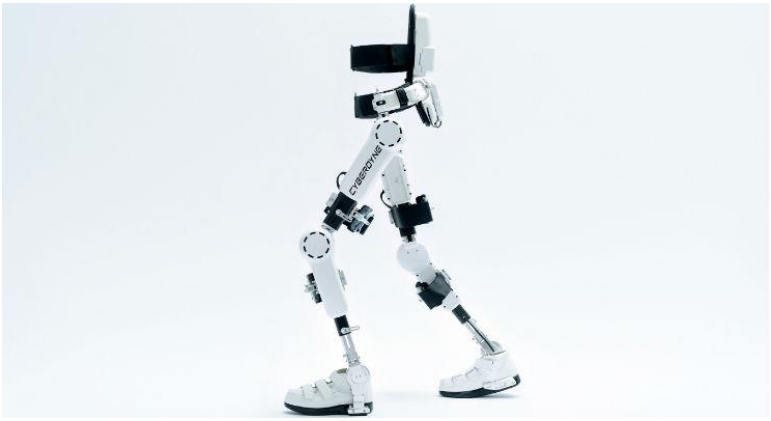

Figure 2.2. Hybrid Assistive Leg 3 rehabilitation device for lower limb spinal cord injuries (Roboticsbiz 2020)

An alternative device, the ReWalk, is a similar product to HAL-3 but specifically targeted towards paraplegics (no leg use) that enables standing and locomotion. Once again, power is provided to the hip and knee while the ankle is left passively affected. No justifications as to why this decision was made for either product is easily available, so specific research into the human ankle and its role in gait is necessary. The ReWalk power supply comes from a backpack which may be inconvenient in some situations, and is naturally very expensive. Hardware design is not the focus of this project or literature review, so these shortcomings are ignored at this point in

time. The control systems of the devices are what is being analysed for potential improvements and gaps in the research. Tucker et al. (2015) published a review focusing on these control strategies that are discussed below.

Tucker proposes a generalized control framework to help identify the interactions between the user, the device, and the environment, along with a controller hierarchy that can be used to guide project experimental structure. The top level incorporates the estimation of a users intent, which is not always reliably achievable or well-defined. The mid level transforms these estimations to a pre-defined operational state of the device, effectively converting the ambiguous goals into a clearer numerical goal. The low level acts as the classical controller to transition the device towards the established desired state. For the future experimental works, estimating user intent is most often achieved through the use of electromyography or electroencephalography (EEG), as the electrical signals are generated before any muscle activations occur. The delay between these two events is what allows device activation to occur in time to properly assist in movement and act as a predictor of motion. EMG is discussed in more detail in a later subchapter, but EEG has also been used for rehabilitation purposes (Al-Quraishi et al. 2018). For lower limb control EEG was considered less desirable as electrodes attached to the head must be required to connect to the device on the legs. For the purposes of daily living, it is unlikely subjects would be excited about wires across the entire body, or even just the head-device if wireless transmission was possible (although not researched, this would likely slow transmission down to a point of negatively impacting predictive behaviour). With the addition of excessive hair causing worse readings, EEG was disfavoured in the project and was not intently researched. Most research related to the high level of control hierarchy fell into the EMG-to-force or EMG-to-position category, which will be focused on in a later experimental chapter. The high level control simply distinguishes between multiple predefined discrete categories of intention, such as “walking”, “running”, “standing”, etc. and the conversion from this state to intended force or position output occurs at the mid level. Both levels can make use of bio-signals to direct the estimation (simple classification) or state calculation (numerical output). The most commonly referenced method for these tasks is the use of artificial intelligence, specifically neural networks (Zhang et al. 2019). The different approaches will be discussed in more detail in section 2.4.

Reporting the hardware components of these rehabilitation devices is the most common form of literature for the field, with several published reviews focusing on these aspects. What is less commonly reported is the analysis of which scenarios a device would be best suited for.

Most devices discuss use after spinal cord injury, but rehabilitation is necessary for lower limb injury and arthritis as well, and these differences are rarely outlined. For knee osteoarthritis, pre-surgery therapy is recommended in almost all cases and can often eliminate the necessity for surgery altogether (Grande 2016). This source has a clear conflict of interest as the author is a physiotherapist actively selling therapy sessions and no external publication could be found clearly sharing these claims, however other papers have discussed the non-surgical management of lower limb arthritis (Kon et al. 2012). Generally speaking, most rehabilitation robotics focus on corrective maintenance and aim to recover lost limb functionality after injury or disability. Preventative rehabilitation does exist and would also likely benefit from the same control system improvements proposed in this project, but the overall focus will remain on ambulatory recovery to help limit the scope of research necessary for progress.

As mentioned by Stratton-Powell (2018), total ankle replacement is underdeveloped when compared to other lower limb joint replacements; a trend which continues into the wearable prosthetics. This may be caused by the lower frequency of ankle surgery leading to less interest in the topic, as well as the alternate surgery of ankle fusion which is simpler but more motion-restricting. Naturally research will fill the areas that call for research, so the need for a knee rehabilitation device for post knee replacement surgery will drastically outnumber the need for post ankle surgery if the primary ankle surgery does not encourage joint motion after the fact. Focusing on the ankle in this project provides a sense of novelty within the rehabilitation setting, as it is the least common lower limb joint mentioned within the literature. The development of an ankle rehabilitation device may in turn lead to TAR being more common, ideally cascading into more interest into the area. Despite this, there is still a lot to learn from other robotic joint devices, with the techniques developed often directly applicable.

Discussions of self-perception are also an important factor for widespread use in general society, and have already been identified as practical issues (Kobetic et al. 2009), however these questions fall outside the scope of the current project.

2.1.2. Electromyography Technologies

2.1.2.1. General EMG use & application

When looking into robotic rehabilitation technologies, one of the most crucial steps is the human-machine interface (HMI). How the robot interacts with a living human and how the human interacts with the robot must make many considerations ranging from how data will be

collected to the physical safety of those involved. The human body produces many electrical signals known as bio-signals which can be detected using electrode sensors attached to the skin. Access to electrical signals that directly relate to human behaviour without any medical procedures is an extremely beneficial method for interfacing with a robot due to the intuitive nature of using human output electrical signals as the input signals for robotic control. Electromyography (EMG) is a specific type of bio-signal that is produced by the musculoskeletal system, creating a signal linked to a human's intended motion. Many robots designed to be worn by a human; whether they be prostheses (Au, Berniker, and Herr 2008), orthoses (Fleischer, Reinicke, and Hommel 2005), or exoskeletons (Song et al. 2013); use EMG control due to the intuitive approach of using the human's natural intentions as the input. EMG signals are generated and can be detected before any muscle forces. This delay between activation and force output is called electromechanical delay (Begovic et al. 2014), allowing EMG signals to act as a predictor of intent and provide any assistance forces to help complete actions the subject may be unable to achieve unaided. The activation of arm muscles was found to be heavily related to the arm's force manipulability, which simply means that, given a specific arm posture, identifying specific arm muscle actions through EMG signals was able to predict the force generated on three separate axes (Artemiadis et al. 2011). These findings for the upper limbs will also be applicable for lower limb force generation, although the decoder will need to be adjusted. The method for model training is not clear from the paper, however a linear hidden-state model is mentioned, which is believed to be some form of linear classifier in high-dimensional space.

In reference to rehabilitation and robot control, issues arise with EMG signals when trying to standardise between subjects of varying conditions. Healthy subjects will naturally have a stronger EMG signal than subjects who suffer from some form of disability such as spinal cord injuries or hemiplegia (which seems to be attributed to a limited descending pathway (Song et al. 2013) and an absence of spontaneous activity (Chokroverty and Medina 1978)) which makes feature extraction from signals more difficult for those who need rehabilitation the most. Many papers use healthy subjects as the experimental subjects which may bias the results towards favouring EMG techniques where they are not as applicable. These papers are still significant for the development of control and hence will be included within the literature review.

For the purposes of a general rehabilitation robot for the ankle, EMG signals definitely play a role. Osteoarthritis causes a reduction in range of motion (Steultjens et al. 2000), but there are few papers which examine the EMG signals of OA patients. The restrictions to motion may then

be caused by an unwillingness to perform larger motions (as a human's natural aversion to pain) or due to the weakened muscles that occur because of OA. EMG signals can also be used as an input for alternate control systems such as switching regimes or dynamics model estimation, effectively using the EMG signals to classify the type of movement or the physical parameters of the measured muscle. Artemiadis and Kyriakopoulos (2011) used a switching regime model to allow continuous motion through 3D space as opposed to the more common discrete control which limits positions to a finite and predetermined state in space. The switching regime is also customisable for each individual to better estimate motions associated with their specific EMG production. This approach is extremely beneficial for clinical application, as EMG signal production will change between users based on a myriad of factors that may be out of the tester's control (any obstacles between electrode and muscle, such as body hair or fat) or within their control but difficult to perfectly replicate (electrode placement). A controller that only allows positioning at predetermined discrete orientations would not be acceptable for activities of daily living (ADL) so creative EMG applications must be applied for improvements to robustness and smoothness of robotic motion. Instead of relying on one predetermined muscle model such as the Hills model (Miller 2018), by using both EMG electrodes and mechanical sensors, EMG signals can be collected in conjunction with their produced kinematics and create an estimation model through machine learning (Zhang et al. 2012). Using additional angle sensors allows a supervised learning approach with a back propagation neural network, as well as validate the artificial intelligence system once properly trained. The results from this approach can be seen in Figure 6 and 7 of Zhang et al. (2012) for both healthy and spinal cord injury patients, showing estimation more than adequate enough for rehabilitation purposes. This technique can also be used to create numerous decoding models to associate EMG signals with the corresponding motion. EMG pattern recognition is often used for simple classifications of gestures in real time (Zhang et al. 2019). Alternate machine learning techniques such as convolutional neural networks or reinforcement learning have also been developed for hand gesture classification (Song et al. 2018).

This technique can assist adaptive control systems as the identification of intended motion can help determine what motions are likely to follow and what robotic assistance should be provided. Although most papers do not seem to be Markovian in nature and each state prediction does not rely on the previous state, human behaviour does (Artemiadis and Kyriakopoulos (2011) explicitly mentioned that their experiment had no Markovian dynamics, although previous cited switching regimes did). A human is unlikely to transition from sitting

down to an immediate stair climb action, instead involving both a “stand-up” action and a “flat surface walk” in between. Real-time gesture technologies can therefore be useful as a switching variable, allowing a switching regime or adaptive control system to best select the appropriate control system or dynamical model. Systems that allow easy training of muscle models are likely to be very useful for rehabilitation purposes as they can be customised to the subject and be updated throughout their progress and changing gait patterns. These models designed through EMG signals are known as EMG-Driven musculoskeletal models (EDMM) which are commonly used in conjunction with controllers aiming to detect a subject’s effort put in (Yao et al. 2018). EMG signals may also be combined with a variety of other sensors such as an inertial measurement unit (IMU) sensor to help identify intent. Using only two IMUs on the foot and shank allowed ankle power estimated through the use of a random forest model machine learning algorithm (Jiang et al. 2019). This EMG alternative is validated through further motion tracking and force plate recordings and requires no inverse dynamics calculations or heavily monitored environment. Despite majority of this project focusing heavily on EMG as the dominant system input, it is important to note the success and efficiency of other data collection. It is noted, however, that for stroke-affected individuals often show an imbalance between power output from both legs, so IMU sensors must be placed on both legs to account for any asymmetries within rehabilitation patients – a consideration not present in the healthy subjects used in the aforementioned paper. Similar considerations for osteoarthritis patients must be made.

Most papers seem to have reached consensus on how to process EMG signals into a meaningful data. Wavelet analysis which involves full wave rectification, normalization of maximum voluntary contraction (MVC) level, band pass filtering and envelope calculation is present in multiple papers, including reviews of EMG signal processing (Chowdhury et al. 2013; Raez, Hussain, and Mohd-Yasin 2006). EMG signals can be used at several different stages in the control loop design, a basic diagram of which is shown in Figure 2.3. One very common application is to use EMG signals as the input reference signal to the control system, setting the system to converge to a direct measurement of the human’s intent. If the EMG signal can be decoded into a meaningful expression of intended force, intended joint torque, or intended joint angle, then external sensors can compare measurements to the calculations and actuators can be used to assist in achieving the desired reference signal value.

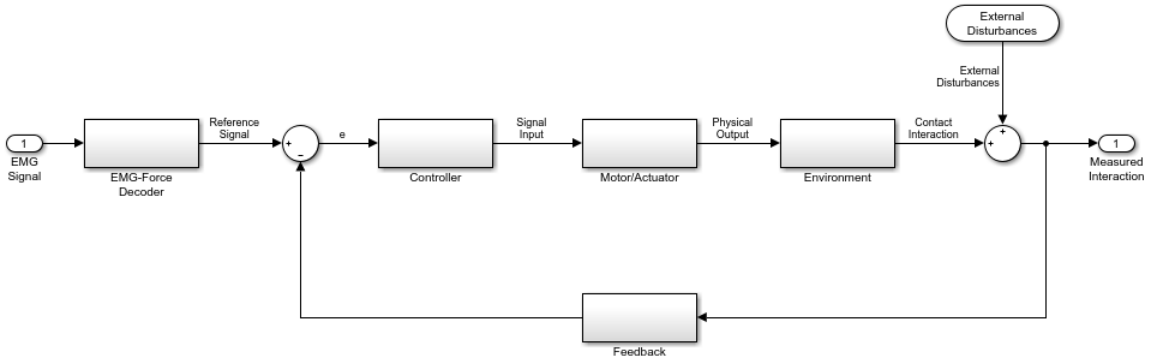

Figure 2.3. Generic block diagram of an EMG-based control system. EMG signals act as the control input to the system which is then decoded into a desired force interaction value. The closed loop component of the system acts as a basic reference tracking loop and aims to minimise the error between the reference signal and the measured output signal

An alternative use for EMG is to use the measurements to formulate an approximation of the dynamics of the human body. As previously discussed, there are several different approaches to creating a mathematical model for physical representation of the human musculoskeletal system. By using EMG signals as an input during a training session the model can be adjusted to an individual to provide more accurate interactions. For instance, during different types of muscle contractions the muscle stiffness will change. Interacting with a non-stiff environment such as stepping on dry sand will result in a different muscle reaction than a stiff environment such as a concrete footpath, so EMG sensory values can also be used to predict which environment it is interacting with and change the controller to better suit the situation. Unfortunately, as with all machine learning techniques, the final model is only as good as the training data. This means that the training data used to create the dynamical model must include information about every kind of interaction the model may experience; a very difficult task to achieve.

The Hill's muscle model is a classic model for muscle contraction that allows predictions of muscle forces during various motions and can be considered the standard for muscle model analysis (Hill 1938). The nonlinear nature of the model and high number of parameters makes analysis difficult, but the model can be improved using EMG signals to better adapt to the individual and estimate real-time muscle forces. This model is known as a Hill-type neuromusculoskeletal model (HNM), but other simpler models can also be constructed such as the linear proportional model (LPM) which predicts forces proportional to EMG levels. HNM and LPM are all EMG-driven models (making them both forms of EDMM) for the human environment but produce different levels of controllability when implemented. Ao, Song, and Gao (2017)

showed that HMN generated smoother movements when compared to the LPM, while also reducing the amount of physical effort required by the user. This reduction of effort bodes well for long-term improvements to ankle injuries, decreasing the loads that are generated within the joint that may cause deterioration. The EDMM designed by Yao et al. (2018) could be subdivided into three sub-models: an EMG-to-activation sub-model, a musculoskeletal geometry sub-model, and a Hill-based muscle-tendon sub-model. When this model was used in conjunction with an adaptive admittance controller the system was able to successfully track joint angle and torque and provide assistance based off user intent. This paper is very similar to our end-goal where we will attempt a design of a different control method that will allow for tracking in multiple dimensions, combining theory used for MIMO control systems.

Instead of trying to model the dynamics of a system an alternate approach can be taken: use the EMG signals to design a control law that causes the entire closed loop system to produce the correct results. The model required to control the dynamical system will often be less sophisticated than the model required to predict the physical system (Brunton 2018). This method still uses EMG signals to achieve the goal of a smooth and robust robotic device, however due to the trial-and-error nature of changing the control law it is best suited for simulations. This is to avoid damage to the device or compromising the safety of any humans interacting with the device.

2.1.2.2. Problems with EMG collection

According to Raez, Hussain, and Mohd-Yasin (2006) the most important concerns about EMG signal collection is the signal-to-noise ratio and the signal distortion, notably in the frequency domain. These issues are obviously still important and fundamental for EMG control, however this paper was written over a decade ago and technological advancements have reduced the noise and distortion effects. Other papers reviewed the pre-processing techniques for reducing noise and identifying the most common noise sources with appropriate reduction techniques (Chowdhury et al. 2013). Inherent noise, electromagnetic noise, electrode movement artefact, and crosstalk are all different sources of noise that must be eliminated as best as possible.

Mentions of electrocardiogram interference were also made, which would need to be considered for upper limb rehabilitation devices but will not affect lower limb EMG readings. Although EMG is ubiquitous in use for motion estimation it still suffers from many practical problems. The most important issue comes from the time-dependence of EMG signals which

cause an individual to produce different amplitudes of electrical signals depending on the state of the subject. As discussed in (Zhang et al. 2019) the amplitude of an EMG signal is a key parameter required for the calculation of many different results desired for feature extraction. Often related to fatigue, as the subject becomes tired the EMG signals decrease in power, shown through holding of MVC for extended periods of time (Moritani, Muro, and Nagata 1986). The changes to amplitude must be considered in any transformation, such as the EMG-to-muscle intent decoder, that depends on this value. General noise also makes feature extraction more difficult, but smart selection of features to extract can minimize noise-related issues (Phinyomark, Phukpattaranont, and Limsakul 2012). The presented features, modified mean frequency & modified median frequency, showed a tolerance to white Gaussian noise and effective functionality in weak EMG signals. This indicates that careful planning of the feature extraction process for EMG signals can expand the scenarios in which EMG pattern recognition can be implemented. The experiment was conducted with a finite number of muscle motions and worked in discrete space, so a combination of these specific features in continuous motion identification has yet to be seen. The core concept, however, is very promising and shows one approach to noise robustness design.

More practical problems also arise in the data collection methodology for surface electromyography. Electrode motion, sweat production, and even blood flow are all known to cause changes to EMG measurements (Fleischer, Reinicke, and Hommel 2005). These changes will occur both within individual patients between experiments, but also between patients; body structure plays an important role in the EMG signals produced and how reliably they can be collected. Physical factors such as subcutaneous fat thickness cause uncertainties in measurements to grow (Nordander et al. 2003), making certain experiments suitable only for specific demographics. This makes the design of experiments and wearable devices very difficult and a “one-size-fits-all” approach, bordering on impossible. It is also a limitation on single-input control systems that will not be able to process all relevant information that is required for a reliable and accurate response. Despite these issues with surface EMG collection it is still one of the best methods for collecting bio-signals in a rehabilitation setting due to its non-invasive nature.

Fleischer, Reinicke, and Hommel (2005) discuss the limitations of using EMG signals as the exclusive input and their inability to consider external forces and torques that may act on the robot during standard operation. Use of additional sensor types such as inertial measurement units can help supplement the robot to better represent the realistic system dynamics. When a

transformation model is designed using EMG signals and joint angle measurements it was shown to be important to limit the order of the model, with noise contribution becoming amplified and reducing overall quality for orders higher than 20 (Zhang et al. 2012). This approach is quite beneficial but is limited to the training data collected for the EMG-angle decoding transformation. Zhang et al. in particular restricted motion training to a small number of participants and only one movement mode in the form of treadmill walking. Although a good start, this does not cover all movement modes that may be necessary for rehabilitation and definitely does not cover all movements for ADL. This is a microcosm of the issues from the machine learning approaches of EMG control; the control system will only be as good as the training data method. No matter how good these models are they will still limit the user's final number of actions performed. This is not as much of a problem in reality than it is on paper, as most sufferers of OA or ankle injuries are older people who only need basic movement options to retain a high quality of life.

Muscle crosstalk makes a one-to-one relationship between EMG signal and motion difficult to identify, especially for the more intricate motions and high degree-of-freedom models such as the hand. Merlo et al. (2009) claimed that "crosstalk occurrences and amplitude on the brachioradialis muscle were mainly due to finger and wrist extensor muscles". Crosstalk issues are not as prevalent in ankle muscles, making them less critical to functionality of an ankle robot but still worth considering to improve the robustness of the EMG data collection method and muscle decentralisation. The muscles that allow ankle dorsiflexion are all extrinsic muscles, originating on the femur, tibia, or fibula. The tibialis anterior and gastrocnemius are the primary muscles for the dominant ankle degree of freedom, but two other muscles also contribute a non-negligible amount, all of which are in close proximity. When combined with other smaller muscles also found in the lower limbs it becomes clear why reducing crosstalk becomes important for estimating force production from EMG readings. If trying to determine EMG readings from smaller muscles then cross-talk becomes more of a problem due to the larger muscles dominating the area and electrodes placement being interrupted (Heloyse Uliam et al. 2012). These issues result in contamination and makes it often impossible to accurately determine where physiological signals are originating.

2.2. Human Considerations

2.2.1. Rehabilitation effectiveness

When designing a robotic system with the intention of physically interacting with humans the safety of all persons involved in its enactment becomes the most important consideration. From a control theory perspective, stability of a system will be strongly correlated with safety due to instability representing unbounded signals. These unbounded signals would in turn produce unbounded physical properties such as motor torques of dangerous levels. Establishing a rigid range of actions that can be performed by the actuators will prevent potential damages but the point of operation may change between users. For instance, the stiffness of a limb joint will vary between humans, especially when comparing a healthy subject with an injured in need of physiotherapy, whether it be neurological (Bressel and McNair 2002) or musculoskeletal (Altman et al. 1986). Altman has studies on osteoarthritis for the hip, knee, and hand, but none on the ankle. The stiffness is recorded for all other joints and as such it is reasonable to interpolate stiffness to ankle osteoarthritis as well.

This variation between humans leads to the practice of explicitly setting upper and lower bounds being an unsatisfactory method for determining a safe performance range; individual properties must be considered. The forces that can be applied by the rehabilitation device is the critical factor in rehabilitation performance, but due to safety concerns the controllers must be designed quite conservative to limit forces and velocities (Atashzar et al. 2020). These restrictions curtail potential performance, especially for subjects further into their rehabilitation process and require higher forces. This trade-off is a necessary sacrifice as any injuries impacted during rehabilitation is likely to cause progress to regress, perhaps to an even worse than the initial state depending on the damage. The design process is always trying to find new techniques to help improve the system controller without any sacrifice to the user's safety, such as passivity-based approaches.

The term 'passivity' is fairly generic and is used in several fields to define different concepts. Passive rehabilitation is performed when all forces that act upon the body are externally produced and the internal muscles of the body are not involved in the motion of rehabilitation exercises. This is contrasted to active rehabilitation where the motions are performed by the internal muscles without external assistance, as a healthy musculoskeletal body would. However in the context of this report the term passivity is most often used in reference to energetic passivity, where the amount of energy being produced by a system must be non-positive. When coupled with a passive system, a second system that adds energy to the original has the potential

to cause instability while a system that consumes energy cannot, and as such passive systems are preferred for stability purposes. Lee and Hogan (2016b) detail that both energetic passivity and mechanical impedance are critical features that must be considered when designing a physically human-interactive robot. Thus to achieve coupled passivity both the ankle-foot orthosis and the human ankle joint must be passive. It is also noted that improving our understanding of the human neuromuscular side of this interaction would allow improvements in the design of the robotic AFO controller.

2.2.2. Energetic passivity and interaction forces

Energetic passivity is present in most environments a robot will interact with as they will be constructed from fundamentally passive mechanical components such as springs and dampers. However the human body does not meet these criteria and is able to produce its own mechanical energy from the chemical energy provided by the digestive system. Furthermore, when focusing on a specific joint, feedback delays between the joint and the brain have the potential to violate system passivity. This problem is worsened in rehabilitation settings for those who suffer neurological disorders that affect nerve communication, such as stroke patients. Passivity of a joint describes the generation of mechanical work solely at the interaction point between systems and can be characterised using passivity analysis (Lee and Hogan 2016b), the mathematical details of which will be discussed at a later section. A second paper by Lee and Hogan (2016a) considers the quantitative characteristics of this passivity analysis for a test group of young healthy subjects. These tests were not performed to replicate ADL but to measure simple muscle activation while sitting. This paper is not supposed to provide all-encompassing passivity analysis but is simply a base for future studies on neurologically impaired subject passivity to be compared to.

Mechanical impedance describes the relationship between joint angle displacement and the associated force/torque that was required to achieve this result. An accurately identified mechanical impedance is a critical factor in dynamic interactions with external objects as it will dictate how much force will be applied to achieve the desired goal. This mechanical impedance will change depending on whether the muscles are being stimulated, with increased impedance in all directions during activation (Hogan 1984) and specifically during locomotion (Lee, Rouse, and Krebs 2016). As the mechanical impedance of the robotic system can be designed through an impedance controller, identifying the mechanical impedance of the ankle interaction port is important for improving performance. Investigations into the human ankle joint found the

mechanical impedance to vary over the gait cycle with increases in impedance around the heel strike and significant decreases as the foot leaves contact with the ground and begins the swing phase. These behaviours can be approximated to a second order model with stiffness, damping, and inertia (Lee, Krebs, and Hogan 2014). A highly passive ankle would require less passivity within the AFO which makes the design less restrictive in terms of energy dissipation and could allow performance improvements. As previously mentioned, the passivity of any given person will vary, especially when comparing healthy individuals to neurologically impaired individuals, and as such the design of an impedance controller must be user specific. As a patient improves their neuromuscular response system throughout rehabilitation they are likely to progress towards a healthier and more passive ankle joint and as such the controller must also change with the progress of the user.

This leads to a need for adaptive impedance controllers and artificial intelligence techniques to achieve the best possible controller. This approach has been attempted by (Bingjing et al. 2019) who used reinforcement learning to adjust and individualise the parameters for an admittance controller. These parameters represent the stiffness, damping effects, and inertia of the system; together dictating the interaction forces produced between the robotic system and the environment (which in the case of rehabilitation is the human). The mathematical details of the admittance controller will be discussed later, however Bingjing et al. (2019) performed simulations of manual parameter adjustment to present the effects on joint angular displacement. The results show that an increase in stiffness or damping will reduce the angular displacement, most likely due to both these parameters acting as a form of resistance to a driving force. Higher damping values also lead to an increase in rise time, meaning that the system takes longer to achieve its peak displacement when more damping is present in the system. The inertia of the system showed no strong changes to angular displacement, but simulations were limited to small variations in inertial values due to the undesired effects to the system damping ratio and natural frequency. These two physical properties of a system can dictate how the system responds to disturbances and are important considerations for system stability, so large changes to the inertial coefficient may undermine the simulation results in representing the real physical system. Experiments were performed for both passive and active rehabilitation, both of which are necessary stages of development because of the aforementioned changes in energetic passivity and mechanical impedance.

2.3. Control Theory

2.3.1. Control theory vs. Artificial intelligence

A robotic device that behaves in accordance with its environment must have sensors to collect information about critical components of the environment to act as the inputs to the robot. It must also use some form of actuator to determine how the robot will interact with the environment, acting as the output of the system. A mathematical model to describe the relationship between inputs and outputs is necessary for the desired behaviours to occur and allow appropriate responses to be calculated. In traditional control theory this model is called a 'transfer function' and is simply a collection of algebraic expressions structured as a matrix. If a system has only one input and one output the system is described as single-input single-output (SISO) and the transfer function will return a scalar value. For more complicated systems such as rehabilitation devices with multiple sensors and higher degrees of freedom in the range of motion, a multi-input multi-output (MIMO) system is required. A MIMO system with *r* inputs and *m* outputs will require a transfer function matrix in the dimension space of R^{mxr}. The element of the transfer function matrix located in the i^{th} row and the j^{th} column will contain a transfer function relating the j^{th} input to the i^{th} output. A real-world physical relation like this would often be represented by a differential equation, such as what is seen for modelling equations of motion or electrical circuit analysis. These differential equations can be converted into algebraic equations through the Laplace transform, converting a function of time to a function of complex frequency $s = \sigma + i\omega$. Transfer functions are presented in the s-domain which allows simpler solving and calculation of coupled systems (Nise 2011).

Transfer functions are difficult to construct for highly complex systems as the mathematical models will not be able to contain all the nuances of the system. As many real-world systems are nonlinear in nature, or have time-variant properties, a transfer function will not be able to appropriately model system behaviour. This limitation will have direct effects on the design of a rehabilitation robot that relies on measurements from the human body to determine motion. The intended goal of using surface electromyography sensors as system input and returning the predicted motion the subject intended as system output would result in a highly nonlinear model. Using machine learning techniques such as backpropagation neural networks can be used for model estimation (Zhang et al. 2012). The construction of such a transfer function would be extremely difficult to the point of being infeasible while neural networks are much easier to construct and train. Further complications arise from the constantly changing abilities of the human body which causes time-dependence in the modelling. Muscle fatigue from

overuse will decrease the average force exertion over relatively short timeframes such as a single exercise session. Fatigue can be detected through EMG signals by a decrease in EMG amplitude and how long a high-frequency signal can be sustained, as higher frequency signals directly relate to the MVC (Yousif et al. 2019). Other bodily functions likely to occur during exercise such as moisture on the skin will affect the sEMG input readings which will affect the input side of the sEMG-force relation (Basmajian and Luca 1985).

An alternative method to constructing a mathematical model is to utilize artificial intelligence techniques to construct this input-output transformation. Artificial neural networks are one such technology that allows nonlinear functions to be constructed and automatically adjusted by using collected data to slowly train the network to produce desired results. Methods of supervised learning (Zhang et al. 2019) and reinforcement learning (Wu, Saul, and Huang 2020) are both represented in the literature and show functioning systems that allow the estimation of user intention from their sEMG measurements. Supervised learning is best suited for when large amounts of sEMG data has been collected along with the recorded motion output that each signal produced (or desired to produce, depending on what is trying to be decoded). Reinforcement learning does not need this large amount of labelled data to develop the model; instead only a set of behavioural rules in the form of a reward function is needed to develop the model. For these reasons reinforcement learning is preferred for the development of a rehabilitation robot when mass data collection is not feasible. One direct negative of using neural networks (NN) instead of constructing a transfer function model is the fact that the system acts as a black box; no meaningful information can be extracted from the NN to understand the transformation. The musculoskeletal system will not be better understood by the results this method produces; however effective transformations can be achieved regardless. Another limiting factor for NN is the necessity for large amounts of data to effectively train the network. This training process is also susceptible to biases within the data or overfitting the data and learning exact patterns rather than the general relationship between input and output that would be more widely applicable (Sutton and Barto 2018).

2.3.2.State space representation

When discussing a control system it is important design the mathematical model to represent the physical model as accurately as possible. For any given system, the state vector $x(t) \in R^{nx1}$ represents all the critical states of a system at time t. The output vector $y(t) \in R^{mx1}$ denotes all the physical measurements that are taken from the system. Using state space representation allows a process to be defined by how both the state vector x, and the output vector y, change over time with respect to a given input vector $u(t) \in R^{rx1}$, as seen below:

$$\dot{x} = Ax + Bu \tag{2.1}$$

$$y = Cx + Du \tag{2.2}$$

A state space representation can be summarised by the collection of matrices $\{A, B, C, D\}$ that satisfy equations (2.1) and (2.2). These matrices refer to the system in the time domain and are labelled as follows with the following dimensions:

State matrix: $A \in R^{nxn}$

Input matrix: $B \in R^{nxr}$

Output matrix: $C \in R^{mxn}$

Feedthrough matrix: $D \in R^{mxr}$

State space representation can be changed by altering these matrix values without affecting the transfer function of the system, G(s), which is the mathematical realisation of how the input transforms into the output. The calculation for the transfer function is performed in the Laplace domain:

$$G(s) = C(sI - A)^{-1}B + D \tag{2.3}$$

A transfer function aims to mathematically model the physical processes of change in a real-world scenario. As such the system transfer function is often referred to as the “process” of the system, or the “plant” if actuation is involved (these terms are often used interchangeably). As there are an infinite number of ways for a state space representation to result in the same transfer function, when constructing a state space representation from a given transfer function there are several canonical forms that can be used to standardise the practice and create a beneficial state space representation. Each canonical form that may be implemented will contain trade-offs that must be considered in reference to the specific model being proposed and the intended purpose of the system. The controllability canonical form and the observability canonical form are the most common and are best suited for systems that are desired to be fully

controllable or fully observable, respectively. The controllability canonical form is used within this project as it guarantees full controllability which is important in in rehabilitation robotics. It must be possible to reach any point in state space for full range of motion and more importantly every state variable must be adjustable by the system to avoid any safety concerns. The controllability canonical form does not imply observability of the system is unimportant, nor does it suggest the system is not fully observable, as these concepts are not mutually exclusive. Full observability is also highly beneficial to the system and should be attempted as well. If as system is both fully observable and fully controllable then control becomes much simpler and easier to work with.

Benefits of working with state space representations include the ability to represent nonlinear systems or time-variant systems (Nise 2011: 118). State space representation also scales better than transfer functions for larger amounts of inputs and outputs. While focusing on ankle rehabilitation the number of inputs and outputs are very unlikely to become numerous enough to be of concern, but future developments should take this note into consideration. For more complex input data arrays or additional degrees of freedom in robotic actuation, state space representation may become the more appropriate design tool.

2.3.3.Passivity-based process control theory

Passivity is an inherent characteristic of a physical system that describes the energy behaviours of an input-output relationship. A system may be characterised as passive if the process dissipates energy over time rather than producing it, which would be described as a non-passive process. Knowledge regarding energy absorption behaviour can significantly help to reduce the conservatism of a specific control algorithm. Systems that do fall into the passive category have special characteristics that can be used in the design of a controller; using these characteristics is known as passivity-based process control (Bao and Lee 2007). One of the most important benefits of passivity-based process control comes from the passivity theorem which states that the combination of two passive systems through negative feedback guarantees a passive and stable closed loop system (Zhang, Bao, and Lee 2002). This has dramatic implications for rehabilitation robotics if it can be guaranteed that the human body component of the human-robot connection is itself passive. If this condition is met then the design of a passive controller for the robotic device will guarantee coupled stability. Even if the human component cannot be modelled perfectly, as long as it is approximate and it retains its passivity then the controller can be designed to guarantee stability by designing the controller to also be passive. This will be true

in the presence of unmodelled dynamics, meaning a simplified model of the human can be used with no added potential for instability (given that these unmodelled dynamics do not violate the passivity of the system). These benefits carry through to testing different controller designs in simulation. If said controller reaches stability in simulations on an approximate model then there is a guarantee that same controller can be implemented into the real-world physical system hardware and still remain stable (Douglas 2021). This is a great benefit both for safety concerns in early testing stages and for avoiding any damages to the hardware.

Unfortunately, for the purpose of rehabilitation only the robotic component of the overall system can be designed to be passive; the fundamental requirement of passivity of the human cannot be controlled. This condition is also not as straightforward as identifying which joint is being coupled to the system. Each joint must have passivity determined individually, which will be vary between specific motion patterns. Atashzar et al. (2020) discovered a correlation between a subject's posture and the passivity margins (a measurement of how far from passivity/non-passivity the system is). The types of muscle contractions (agonist/contracting vs antagonist/relaxing) have a direct effect on this correlation. As different motions will produce different muscle contractions it becomes evident there is a direct link between motion pattern and joint passivity. These conclusions came from tests on the hip joint, however the ankle joint also changes the muscle contractions present depending on the motion. The kinematics and dynamics of an ankle vary between walking, running, stair ascent, and stair descent; all kinds of actions that would be present within activities of daily living (Hyodo et al. 2017). Even changes to the slope of the environment have been shown to affect the dynamics of the ankle which would in turn affect the muscle contraction behaviours (Rábago, Aldridge Whitehead, and Wilken 2016). These environment-dependent variables will need to be considered in future works but walking on a flat surface will be the predominant focus of the project at this point in time.

Variances in passivity between subjects is also a consideration, although these differences can be closely aligned to age demographics rather than the individual. Time delays can result in non-passive behaviour by causing an accumulation of energy in a closed-loop system (Atashzar et al. 2020). These time delays are caused by the time delay between the human brain and the activation of a muscle, and as such this non-passive behaviour is more common in demographics with slower central nervous system response time such as the elderly or neurologically impaired people(Hunter, Pereira, and Keenan 2016). As these time delays are more expected in an older population it is reasonable to expect less passive behaviours overall, however this is

contradicted by Chesworth and Vandervoort (1989) which shows no significant differences in passive stiffness and torque between three groups of healthy women separated by age bracket. The oldest age group consisted of fifteen women between the ages of 61 and 80, all of which were classified as healthy. It is possible that this subject selection excluded subjects that did suffer from slower central nervous system response times, as joint mobility is known to decline more so for males and rapid mobility reduction for ages above 90 (James and Parker 1989). It is also possible the term 'passivity' is being used in a different context and the current reading of these two papers is simply a misinterpretation.

As a response to the energy accumulation caused by time delay, techniques such as adding adaptive damping have been used in the past to dissipate energy (Zhang and Cheah 2015), and as such the ideas behind using passivity-based control for rehabilitation purposes has been shown to be plausible for the human hand and wrist (Atashzar et al. 2017). Atashzar et al. (2020) also mention the user-dependent nature of passivity maps: a tool used to compute dependencies of muscle contractions, geometry, posture, and dynamical behaviour. For specific results of ankle passivity, Lee and Hogan (2016a) discuss the differences between subjects younger than 37 in the passivity of the ankle for quasi-static (remaining still with small, slow perturbations from the robot) and steady state dynamic muscle activation. These tests do not accurately simulate real-world movement and as such may not describe ankle passivity during walking tests. By tracking a percentage of the MVC value, data is collected to relate EMG data, ankle torques, and ankle velocities. This data is then used to determine passivity through curl analysis and passivity analysis for the quasi-static and steady state dynamic tests, respectively. Results showed unimpaired human ankle joints were passive for low frequencies (less than approximately 2 Hz) for most subjects, however around 30% produced nonpassive behaviour when the soleus muscle was being observed. Behaviour became strictly dissipative over the frequency range (10-20 Hz), and dissipative behaviour began falling off for higher frequencies. Discovering these operating frequencies show general passive behaviours and as such we hypothesise the passivity of other specific motions can be determined, or at the very least guided, by analysing their operating frequencies. As the ankle is expected to be strongly passive in the 10-20 Hz range, designing the robotic controller to also be passive would be a conservative design that may negatively impact performance. We can compensate by reducing the passivity restrictions on the controller in this range, allowing the controller to vary over frequencies. A switching regime to change between a discrete number of controllers may be implemented for a pre-determined number of frequencies ranges.

2.3.4. Controllers (PID, impedance, admittance)

Admittance controllers & impedance controllers

An admittance controller will control how much a system is able to be admitted through physical space given some force as an input. In other words, the controller will 'admit' a force to enter the system which will in turn produce motion. This system is intended to move and will use forces as an input to determine these motions. An impedance controller will control how much a system impedes motion, meaning its output will be the force used to impede motion. These two concepts are complementary to one another, where if one system acts as an impedance then the other must act as an admittance. For the fusion of a human ankle and an AFO, if the rehabilitation device is intended to encourage motion from the ankle it is receiving the torque produced from the muscles and produces an appropriate angular displacement to assist in reaching the desired angle. This scenario requires an admittance controller for the AFO, as opposed to an impedance controller which would be better suited for strengthening muscles that are already capable of reaching their desired angles. The parameters for said controller will be user-specific and must be designed for each individual to effectively function.

One method for the tuning of the admittance controller parameters is to use reinforcement learning methods that can produce values that change over time based on the collected sensor data during rehabilitation, such as joint angle error and human-robot contact force (Bingjing et al. 2019). Bingjing et al. use a discrete action space that simplifies the assistive torque supplied to the user to be either on or off in any given direction. This technique seems better suited for passive rehabilitation where the user cannot provide any torque through the musculoskeletal system. However as discussed earlier, active rehabilitation which encourages muscular use is the more effective form of rehabilitation for ADL. If the goal of the rehabilitation is to stretch joint motions to their limits and expand the full range of motions, then this approach is very effective and has been shown to benefit stroke patients (Waldman et al. 2013). For rehabilitation that aims for precision of joint angle and signal reference tracking, this system may struggle to achieve a satisfactory performance level.

Rehabilitation exercises where the subject aims to follow a predetermined ankle angle path have been performed for both passive and active rehabilitation to develop more intricate joint motions (Zhang, Xie, et al. 2018). Active rehabilitation required adaptive admittance parameters for stiffness (K) and damping (B), along with inertial mass (M), automatically adjusting to the subject's exerted force/torque levels to produce the appropriate resistive force from the robot.

This resistive force is calculated based off the error between current position and desired position, as well as the aforementioned parameters, and is shown by the control law:

$$F(t) = M\left(\ddot{X}(t) - \ddot{X}_r(t)\right) + B\left(\dot{X}(t) - \dot{X}_r(t)\right) + K\left(X(t) - X_r(t)\right) \quad (2.4)$$

The variable X represents the measured state variable vector containing displacement values of the system. The variable $\mathrm{X_r}$ represents the reference state variable vector containing all the desired state variable values that the system will aim to converge towards. These variables can be replaced with angular measurements (converting the force measurement to a torque measurement) directly. The maximum angular displacement possible was also saturated to prevent injury during operation. This limitation will vary between subjects, however their experiment was only performed on two subjects and as such this did not affect the setup or design procedure. The admittance control law is a simple equation of motion relating the human-robot interaction force (or torque in this case) with the robot's inertia, damping, and stiffness properties. This is the base for most impedance and admittance controllers, with various details of setup presented in other literature (Song, Yu, and Zhang 2019). Song et al. give mathematical insight into how these parameters can be chosen given some real-world system requirements.

Although most rehabilitation devices use basic reference tracking which will push the robot to a set position, a more effective method is to set an acceptable range of positions to better meet each individuals' abilities (Duschau-Wicke et al. 2010). The range around the tracking path is referred to as the "tunnel" or "window", which encouraged active participation from the user indicated through the produced EMG data. With the combination of graphical feedback instructions, the majority of users with incomplete spinal cord injury were able to actively control gait timing. The authors do note that all tests were performed on a treadmill with constant velocity, so although the control of gait timing was successful it was under heavily controlled environments and dependent on the swing phase of gait (as the stance phase was moved by the treadmill). For non-treadmill walking an additional dimension for windowed path control may be necessary to achieve positive results. The window idea can be adapted from spatial windows directing position an acceptable interactions force or output torque windows, where equation (2.4) must equal a value within the predetermined range.

A Proportional-Integral-Derivative (PID) controller is the most commonly utilised controller in industry settings due to their simplicity and effectiveness. By inputting a desired value, the system will measure the distance between goal and current state and use this error to drive the system in a way that reduces error. The error, it's derivative, and it's integral are all used in calculating the actuation signal, although how much each contributes is generally manually tuned and selected. Due to the nature of this signal transformation, the output of the PID controller always has the same sign as the input signal, and the error and output will share a direction (the derivative component can be set to always be outweighed by the remaining components). This is a defining characteristic of a passive system, so all PID controllers are passive by default and allow the invocation of the passivity theorem (Zames 1966). With the passivity theorem guaranteeing stability between coupled stable systems, having the PID controller be guaranteed passive opens many development directions, including designing a controller that passivates the closed loop system. Several papers exist discussing the creation of PID passivity-based control which exploits the passive properties of PID controllers (Romeo et al. 2021). There have also been publications of using passivity theorem-based ideas to guarantee finite incremental gain for two nonlinear systems connected through negative feedback (Chaffey 2022).

Adding delay into a PID controller can cause the input and output signals to become out of phase, so if delay causes phase of ±90° then the controller will lose its passive properties, potentially causing energy build-up and unstable behaviours. This is mentioned to elucidate the reality that PID controllers can cause issues if not properly designed. For multi-input multi-output systems the PID system becomes more complex. As there are more than one inputs or outputs, the calculation of error is no longer a simple subtraction unless the system is decoupled such that one input directly controls one output. Decoupling, or decentralisation, allows each error to be calculated separately and simply employing multiple PID controllers for single-input single-output systems becomes trivial. Mathematically this PID controller would be a diagonal matrix of the individual controllers, and all the benefits of passivity will remain. Overall, PID controllers have many positive qualities as long as they are appropriately designed to retain their passive characteristics.

2.3.5. Control Configuration Selection

MIMO systems can be represented by transfer function matrices which represent the contribution of each input signal to each output signal. As the system is being designed, the inclusion of specific input or output signals can help structure the system in a way that makes control much easier and the mathematics much simpler. From here, choosing each input output pairing can have surprising effects on stability due to feedback loop interactions, and aiming to decouple these loops as best as possible is fairly common when robustness is desired. There are methods for making this selection, and are studied under the topic of control configuration selection (CCS). The most basic CCS method is known as the relative gain array (RGA) which has developed many extensions over the decades since its conception. The equation for the standard RGA of a standard system G(s) is:

$$\Lambda(G(s)) = G(0).* G^{-T}(0) \tag{2.5}$$

As the system is MIMO and represented by a matrix the multiplication ".*" signifies element-by-element multiplication. All rows and columns of Λ will sum to unity, and the element $[\lambda_{ij}]$ will give a scale invariant measure of the dependence of output i on input j. In the most simple terms, λ_{ij} represents the gain between input j and output i when all other loops are open, divided by the gain between input j and output i when all other loops are closed. The best pairing will be revealed by which element λ_{ij} is closest to 1. If the values are too large then the RGA indicates high closed loop sensitivity which will lead to high difficulty in controlling the system. If the values are negative the RGA indicates instability of the system. It can be noted in the equation above that the system is only being observed at a frequency of 0, and as such the RGA does not reveal any information about system dynamics. To have this information embedded in the CCS it is better suited to use the dynamic relative gain array (DRGA) method. The calculation of the DRGA is similar to the RGA, but observes how the relative gains vary with respect to frequency, as some pairings may be preferred at specific frequency ranges. The equation for the standard DRGA is:

$$\Lambda\big(G(s)\big) = G(s).* G^{-T}(s) \tag{2.6}$$

Despite calculating the RGA by default, the DRGA is not always preferred. The RGA is independent of any controller design and disturbances, which is not true for the DRGA. This example is simply to underline the various approaches in CCS and there is no "one-size-fits-all" approach to CCS method selection; how the pairing is determined will be dependent on the system and what characteristics are most important to the designer.

Publications on CCS methods that employ passivity are also in existence, however only one paper seems readily available and referenced (Bao et al. 2007) and is described as “newly introduced interaction analysis” from a textbook published in 2009 (Khaki-Sedigh and Moaveni 2009). This passivity-based approach does not rely upon diagonal dominance (suggesting loop decoupling) and instead use the degree of passivity to determine best closed loop performance under decentralised control. A large portion of the experimental work performed later used this publication as a reference and guiding point. With the notable gap in literature utilising this passivity-based CCS methodology, and the results showing new CCS methods can be created to focus on specific system characteristics, it becomes clear the design of a new passivity-based CCS method will likely be novel with benefits to the project overall.

2.3.5.1. Transfer function approximation

Some CCS methods require a transfer function to be in a specific form, such as a strictly proper matrix, or a matrix fraction description. For systems with time delay contained within an exponential function, it is often required to find an approximation of the system in the standard polynomial representation of a transfer function. A Padé approximation can effectively perform this estimation, and for the purposes of control systems, a first order approximation produces practical results.

$$e^{-\theta s} \approx \frac{1-\frac{\theta}{2}s}{1+\frac{\theta}{2}s} \tag{2.7}$$

By using this approximate equivalence, any transfer function with time delay can be converted into a strictly polynomial form to allow for more flexible transfer function manipulation.

2.4. Artificial Intelligence in Rehabilitation

Artificial Intelligence is a fairly generic term that is used for whenever a computer is tasked in observing a situation and predicting details about the information perceived without an equation producing absolute certainty. It aims to replicate the tasks traditionally performed by humans, such as identifying images or not walking into walls; tasks easily performed by humans but not by computers or robots. Machine learning is a type of artificial intelligence that helps categorise data that it has never experienced by comparing it to large amounts of similar data that it has experienced. The data is required to be structured when training, and the features

that are to be extracted are selected manually by the architect. Deep learning is a subset of machine learning where these features are not manually selected and the network chooses them itself, creating a black box system that is difficult to analyse. Deep learning can be used for optimisation problems that are based on collecting data, while classical control is optimisation that is based on detailed models of the system and the physical dynamics. As it is practically impossible to create a detailed model of every individual's unique motion patterns for rehabilitation purposes, deep learning becomes a highly favoured tool in customising rehabilitation.

2.4.1. Reinforcement Learning Algorithms

Reinforcement learning (RL) is heavily used in robotic design for its effectiveness in producing a system able to interact with an environment that does not have a pre-established model. Model-free reinforcement learning allows a model of the environment to be generated through a simulation essentially equivalent to trial and error. When designing a controller for a system, tuning the parameters to achieve a specific behaviour can be difficult. However by utilising RL to tune these parameters, the desired behaviours can be maximised and general patterns can be identified to learn overarching desires rather than individual intricacies. Complex, nonlinear relationships can be implemented into robotics through this method that would assist in the development of a general controller. The first stage in using RL is to select the appropriate algorithm for the designated problem-at-hand. If the system can be interacted with through a finite number of discrete actions (such as cardinal direction inputs or button clicks) then a deep Q-Network (DQN) is best suited. If the system requires a more continuous value (such as a torque ranging between 0 and 10 Newton-metres) then some form of an actor-critic network will be beneficial. For brevity, only DQN and the twin-delayed deep deterministic policy gradient (TD3PG) algorithms will be discussed in detail below, however other algorithms or forms of AI have been used for rehabilitation purposes, such as proximal policy optimisation (Anand et al. 2019) or fuzzy logic (Yang et al. 2016).

2.4.1.1.Deep Q-Network

A DQN can have multiple system outputs, but the values these outputs can take will be limited by the DQN design. This limitation does hinder DQN applicability for determining assistance levels (which often come in the form of a continuous torque value) but would be well suited for more classification tasks that may occur during rehabilitation, such as mode switching. As different physical tasks will require different kinds of assistance, a DQN would be able to observe the given state and predict which kind of motion is being performed to trigger a change in behaviour. Switching regimes are fairly common in rehabilitation for the purposes of decomposing complex physical tasks into smaller and easier-to-control tasks (Huang et al. 2020) or aiming to remove time-dependence from systems such as EMG decoding (Artemiadis and Kyriakopoulos 2011). Although neither of these papers utilise RL to build the switching regimes, they do present the benefits of smaller control systems for specific situations. This exact scenario could be designed with a DQN in mind, as rather than creating an algorithm to determine current state the neural network will return which state will accumulate the highest long term reward. There are noted problems with switching regimes such as the transition between states potentially causing 'bumps' in the control. For any human-interfacing system this unintended motion is concerning and potentially dangerous, so improvements become necessary. Huang et al. (2021) is able to implement these improvements through the creation of a generic realisation of each subsystem to switch between, rather than completely isolated subsystems. The creation of this generic subsystem is also something that may be achievable with other RL techniques, although not using the DQN algorithm.

The DQN architecture works the same as other reinforcement learning techniques by performing randomly selected actions and recording the resulting reward function accumulation to determine how successful a given state-action pairing is for achieving the goals. This information is retained in the form of a Q-value, which simply acts as a function that takes "current state" and "follow-up action" as inputs and returns a value. By calculating and comparing every Q-value for a given state (cycling through possible actions) the best action is then determined by the highest Q-value. This technique has been used in other subtasks of rehabilitation, such as for the classification of EMG motions and extraction of key features from EMG signals (Song et al. 2018). DQN is the primary focus of Chapter 5 in this dissertation.

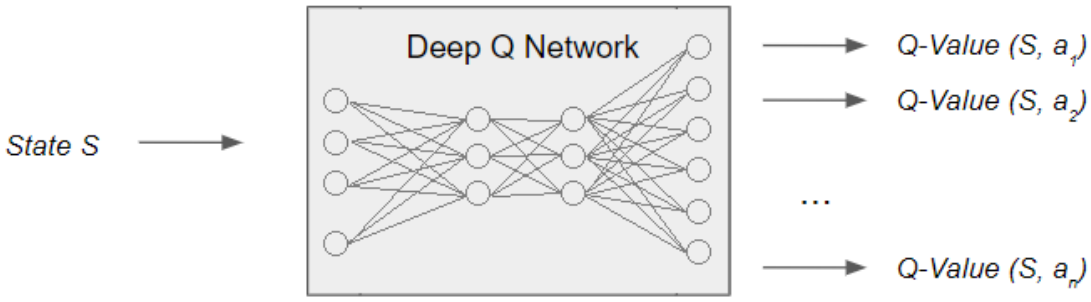

Figure 2.4. Deep Q-Network architecture for converting state recordings to action-defining Q-values

2.4.1.2.Twin-Delayed Deep Deterministic Policy Gradient

The other heavily featured RL algorithm within this dissertation is the TD3PG algorithm for producing continuous signals. Capable of producing continuous actions rather than predetermined set values like the DQN, the TD3PG is much better suited for tasks that require more precision in the decision making. Both algorithms are used to calculate a Q-function through an iterative updating of Bellman equations and reward calculations, but the TD3PG also learns a policy simultaneously (which is essentially a function that determines which action is best suited for a given state input) from these Q-functions. The TD3PG is simply two deep deterministic policy gradient (DDPG) networks operating simultaneously to generate two different Q-functions. By then selecting the more conservative Q-function the transitions in network learning will be slower, but more robust to sudden changes and overestimating specific state-action combinations which will often lead to non-optimal behaviour or even policy failure through divergence. Other changes to DDPG include only updating the policy every few step instances, and adding noise to the continuous target action to avoid exact action values being rated extremely highly and small deviations rated poorly (imagine the difference between an impulse signal and a sinc function signal). All techniques combined result in a more robust RL agent with improved performance over DDPG (OpenAI 2018).

For robotic control and human interaction specifically, the input to the TD3PG agent is often an amalgamation of the robotic device's current actuator positioning (often angular values for joint-focused work) and any sensor recordings relating to the human, such as EMG recordings or inertial measurement unit recordings. Naturally more inputs lead to a more complex system, so as few observations as reasonably possible is the most efficient design approach. It has been demonstrated that a 2 degree-of-freedom robotic arm can be EMG-controlled through actor-critic based RL using only 4 recordings: wrist angle, elbow angle, EMG signals 1 and 2 (Pilarski et

al. 2011). Although not for rehabilitation purposes, the underlying engineering is similar enough to draw meaningful conclusions. Rehabilitation papers use similar techniques, such as (Huang et al. 2015) which uses the force and position of each finger as the state used to control a 5 degree-of-freedom rehabilitation device used for developing fine motor control. A clear correlation between the degrees of freedom and number of observed values is present throughout all the literature. However for larger motions when there are human-machine interactions present, to guarantee safety more measurements are required, such as angular velocity along with the omnipresent angular displacement.

An RL agent capable of fine tuning continuous values can also be used in combination with control engineering to help select the appropriate control parameters for use in rehabilitation.

2.5. Controller Tuning with Reinforcement Learning

2.5.1. Adaptive tuning of controllers

Tuning the controller parameters before system operation is the traditional method of control, as the assumption that the operating environment will remain constant throughout. For complex and time-dependent systems this will result in poor performance as a constant controller is unable to account for new disturbances, signals of a different magnitude, or a new interaction that requires different responses. Parameters must change during operation, known as online or adaptive control, as the system allows the controller to adapt to changes in the environment.

To appropriately guide the user, a general understanding of the expected environmental interactions must be known when designing the controller. For rehabilitation, the environment that the robot interacts with is the human body, which is capable of changing its dynamic properties such as muscle stiffness or maximum force output during gait and basic activities. These changes are based on current actions being performed (Hyunglae and Hogan 2013; Li and Ogden 2012), muscle fatigue (Toumi et al. 2006), or even changes to their surroundings such as temperature (Cornwall 1994). For traversing in everyday life the obstacles interacted with will also play a role, as walking on a slope is biomechanically different from level ground walking (Pickle et al. 2016). Differences between users will need to be adjusted for, as some users may prefer a tighter tracking performance while others prefer a larger margin of error allowing for personal input to play a larger role in rehabilitation. These changes should affect the power output from a robotic device, however a classic controller may struggle to adapt as necessary

while an adaptive controller will allow these changes to be addressed by the control loop. From the perspective of the controller-tuning AI, the robotic system would also be considered the environment and any changes in its functionality would also need to be addressed by the agent. Cases of the agent adapting to the robotic system failing or performing abnormally would be considered a form of fault-tolerant control. Adaptive PID controllers have been constructed using fuzzy logic (Sam and Angel 2017)-(Ibrahim 2002), and with recursive least square algorithms (Fahmy, Badr, and Rahman 2014), both of which showed improvements to reference tracking over the non-adaptive PID controller model. Adaptive impedance across a gait cycle has also been shown to reduce complications for users suffering from lower limb impairments such as drop foot (Blaya and Herr 2004).

The final goal of the reinforcement learning in this chapter is to produce an appropriate mechanical admittance for a robot at any given time. This objective is guided by the discussion provided by Lee and Hogan (2016b), which in turn was discussing results presented by Colgate and Hogan (1988). This discussion states that the driving point impedance (calculated by the input velocity divided by the output force of the system) of a linear system being positive real can guarantee coupled stability for a passive Hamiltonian environment. As the positive real condition is equivalent to energetic passivity, if a controller can be designed to retain a positive real impedance then coupled system stability will be easily achieved. One small complication arises in the description of the system being linear; no such guarantee is possible for robotic behaviours in general. This requirement isn't focused on in either paper and as such it is temporarily ignored. In the case that linearity is a fundamental condition, linearization of the nonlinear system can be performed to test the passivity and stability retention properties.

2.5.2. Reinforcement learning algorithms for controller tuning

Using a reinforcement learning agent to learn the relation between user input and necessary parameter changing behaviours will allow an adaptive controller to be implemented to best suit the environment and interaction behaviours. Previous approaches for the design of an adaptive impedance controller have shown that the frequency of impedance parameter updating will have an effect on controllability and fluency of operation, as mentioned by Bingjing et al. (2019) in reference to using a "discrete impedance parameter" for a welding-assistance robot (Erden and Marić 2011). This description of discrete impedance controller seems to be referring to a constant value for the mass and damping coefficients used during welding simulations as opposed to the variable impedance control when not simulating welding. Regardless, later works in rehabilitation showed inferring human intention to continuously adjust parameters

could improve flexibility when compared to fixed parameters if different patients are considered. This is a prime situation for RL to adjust parameters and the functions that determine them based on the physical movements being performed.

A plethora of different RL-based techniques for automated controller tuning are documented in non-rehabilitation fields. Kim et al. (2010) implemented a Natural Actor Critic (NAC) algorithm with a recursive least squares filter to learn the optimal impedance parameters for robotic contact tasks such as opening a door or movement throughout space. For daily activities this result is extremely beneficial as the interaction forces were shown to lower after training, even in uncertain environments. Fuzzy logic has also been used in conjunction with RL to convert the traditionally discrete Q-learning agents to continuous state space problems, known as fuzzy Q-learning, and was able to reduce the state-action space at the cost of a guaranteed convergence (Kofinas and Dounis 2019). Although this approach is unlikely to be used for the purposes of rehabilitation, the combination of fuzzy logic and RL is definitely a strategy that should be monitored. Other more basic techniques such as DDPG (Liu et al. 2019) and radial basis function (Guan and Yamamoto 2021) are also reported upon, but simply encourage the use of RL in rehabilitation without any specific importance.

Within reinforcement learning terminology an important distinction to make is between model-free and model-based approaches. A model-based approach requires the system to have a perfect model of its environment, including transition probabilities for stochastic environments. With this knowledge, the agent will have prior knowledge to what states it can transition into and which actions will cause that specific trajectory. For any real-world scenario an accurate environment model is rarely available and in many cases not possible to obtain. Tests have been performed to try and compare model-based and model free RL in the field of autonomous racing, showing significantly better results from the model-based deep learning algorithm, Dreamer (Brunnbauer et al. 2021). As the environment is much simpler in map tracing, this result does not offer much insight for rehabilitation but does articulate several considerations when transitioning from simulations to real robots, such as using a consistent starting point for a moderate portion of training data and random placement for the remainder. It is also noted that implementing deep reinforcement learning into the real world is more difficult than simulations and less documented.

Adaptive controllers have shown to be able to reject disturbances during online operation (Howell and Best 2000). As this does not require any detailed knowledge of the system being controlled, it is a useful alternative to the model-based methods previously discussed. Although

more data is required as it is less data-efficient, the ease of implementation drastically makes up for this shortcoming. It is also noted that the results of the continuous action reinforcement learning automata (CARLA) were superior to older manual PID tuning techniques such as Ziegler-Nichols. This kind of RL is most favoured for systems where changes can be anticipated (like gait). One of the dangers of relying heavily on data is the possibility of biases in the data causing biases in the trained system, or overfitting to the data available. Studies on the effects of overfitting have discussed the increased sensitivity to environmental perturbations, as well as the similarities to an under-explored environment making diagnostics less clear-cut (Zhang, Vinyals, et al. 2018). The problem of overfitting becomes a larger problem for data collection, as any small environmental changes in real-world gait must be present in the training data or human safety becomes compromised. Small bumps in a footpath may have never been experienced by an agent trained on data collected exclusively within a gait laboratory, and as such the data collection stage of RL development becomes increasingly important for rehabilitation purposes, if the rehabilitation device is itself not restricted in potential environments. Adding model uncertainty speeds learning; aims to incorporate model of reference signal into the learning process; assuming a linear model.

Chapter 3 - Decentralisation of MIMO Integral Controller Using Passivation

3.1. Introduction

With technology expanding at greater speeds than ever and becoming more intertwined with every aspect of society, the control systems that can be constructed are becoming more complex and ever-present. Smart devices have become common in both business and residential settings, allowing data aggregation and analysis on topics never before possible. Connecting these devices into a networking system allows dramatic improvements to society; whether it be through a smart grid appropriately scheduling devices to minimize economic cost (Venayagamoorthy et al. 2016), or by simply making device communication more user friendly. The Internet of Things (IoT) is a global network that connects smart devices to the internet and makes a great example of how device interconnectedness can drastically increase the services that can be offered by automated systems and the societal benefit they provide. These services can be more efficiently scheduled and run when access to guiding data is available, which is also true for systems that are isolated from the greater IoT but communicate within a smaller localised network. Although these benefits are vast, the interconnections of smart devices bring complexities to a system that must be addressed in order to retain a functioning system. Control theory dealing with multi-input multi-output (MIMO) systems is well equipped to deal with issues such as device coupling, where one input inadvertently affects outputs that are not desired. Additionally, the obvious effect of adding more devices to a system is that the number of potential points of failure also increases, so the system must avoid any critical failures that may occur due to these coupling effects. Stability of a system can be compromised due to input-output (IO) coupling and signal time delay (Cao 2014). For a MIMO system with transfer functions for every IO relation, every transfer function must be stable to guarantee internal stability of the coupled system. For a system with n inputs and m outputs the MIMO transfer function can be represented as a matrix with each component being its own transfer function.

$$G(s) = \begin{bmatrix} g_{11}(s) & \cdots & g_{1n}(s) \\ \vdots & \ddots & \vdots \\ g_{m1}(s) & \cdots & g_{mn}(s) \end{bmatrix} \tag{3.1}$$

From this information, two important tasks are derived to best design these complex systems: minimizing the number of inputs and outputs the system uses without negatively affecting the overall performance, and, simplifying the IO relationships to make the control system more manageable and efficient. These tasks are not required for single-input single-output (SISO) models which are much more easily controlled. If all coupling effects can be minimized to a level that any crosstalk can be effectively ignored, then a MIMO system can be perfectly described as multiple SISO systems. This setup is known as a "decentralized controller" as each SISO controller does not need to interact with one another and no central control is necessary. This would be represented by G(s) in (3.1) becoming a diagonal matrix where $g_{ij}(s) = 0 \; for \; i \neq j$. The decentralised control structure is much more common than the centralised approach, especially in the last few decades (Bao and Lee 2007: 2). To successfully design a decentralised controller it is important that the correct input be paired with the appropriate output, which is not always immediately obvious due to the aforementioned signal coupling. This step is known as "control configuration selection" or "loop pairing" and must occur before any controller tuning takes place. Many different techniques of loop pairing have been developed over several decades with new techniques still being proposed relatively recently (Huilcapi et al. 2019; Bao et al. 2007). For any papers discussing the control configuration selection it is important to keep in mind that control systems discussed must already be decentralised for the loop pairing to work. Therefore, a method that decentralises a coupled MIMO system can act as a very useful tool in control design.

An analysis of a system from the perspective of energy transfer can help determine system stability and provide an intuitive reasoning behind system responses. Through this lens, specific properties arise that categorise systems depending on their energy properties. A system is said to be dissipative if the amount of energy that is stored within the system over a finite time interval is less than the amount of energy that was supplied to the system in the same time period. A system is said to be passive if it is dissipative with respect to a specific supply rate, which will be discussed later (Bao and Lee 2007). This definition of passivity brings with it many beneficial properties that can be exploited to decentralise a system while retaining system stability. This is especially important for the purposes of rehabilitation, where certain human joints are passive systems that will allow cohesion between human and machine. This information acted as the catalyst into this research, fundamentally tying the results with ankle rehabilitation. As a passive system is also minimum phase, ensuring passivity also removes the need for excessive fail-safe designs. This chapter focuses on the decentralisation of a MIMO

controller using these passivity properties as a proof-of-concept to be used in future complex MIMO systems.

Large portions of this chapter rely on esoteric mathematically-intensive control theory that is not applicable to other chapters of the dissertation. To better group the relevant information, a secondary literature review is included within this chapter to detail the theory behind passivity-based process control. A breakdown of the aims and objectives of this chapter are discussed in "Objectives" and the practical implementations of discussed theory to achieve these goals are discussed in "Methodology". This chapter aims to establish an easy and repeatable method for converting real-world control plants into a passive system as close to the original as possible, and determine the best input-output pairing configuration for this transformation. With these new designs, real-world robotic systems will be able to retain passivity when coupled with specific human joints such as the ankle. From there, controller development can proceed with less limitations and more assurance of safety and stability during operation, allowing redundancies in the design to be removed which may have been negatively impacting performance.

3.2. Literature Review

3.2.1.Matrix Fraction Descriptions

Transfer functions will often be constructed from polynomial numerators and denominators, where the degree of the denominator is larger than the degree of the numerator if the transfer function is considered strictly proper. With an n-by-n transfer function matrix containing n^2 transfer functions it is sometimes beneficial to restructure the system matrix into two separate n-by-n matrices that separate numerators and denominators. This is known as a matrix fraction description (MFD) and can be implemented in two ways: a left MFD will multiply the inverse of the denominator matrix $D(s)$ to the left-hand side of the numerator matrix $N(s)$, while a right MFD will multiply the inverse of $D(s)$ to the right-hand side of $N(s)$. Within this paper, whenever an MFD is referenced it is referring to a right MFD, for which the equation is provided:

$$G(s) = N(s)D^{-1}(s) \tag{3.2}$$

A right MFD can be obtained from a state space description of a given system by designing stabilizing state-variable feedback (Goodwin, Graebe, and Salgado 2000: 597). Matrix fraction

descriptions are useful for the decomposition of complex proper transfer functions which are often found representations of physical systems. For the purposes of this project, if a process is strictly passive then it can be stabilised by a decentralised passive controller. A decentralised controller is easier to construct by finding the MFD and restructuring the denominator matrix to be diagonal. From this form, a state space representation which guarantees the decentralised characteristics will be producible. This method is described by Kailath (1980: 403) who utilises the controller form realisation for a right MFD.

3.2.2.Passivity

All control systems will have energy transfer from the system in both inward and outward directions. Energy directed inward is stored in the system which may cause instability, while energy directed outward is distributed to the environment. This energy rate of change is called the supply rate $\mathrm{w}(\mathrm{t})$ and is dependent on the input $\mathrm{u}(\mathrm{t})$ and output $\mathrm{y}(\mathrm{t})$ of the system. A storage function $\mathrm{S}(\mathrm{x}(\mathrm{t}))$ describes how much energy is stored within the system at any given time-dependent state $\mathrm{x}(\mathrm{t})$, and can be used to determine system behaviour with the supply rate. If the amount of energy stored within the system over a time period $\mathrm{T} = [\mathrm{t}_0, \mathrm{t}_1]$ is less than the integral of the supply rate over the same period of time, then some of the energy supplied has been lost in the transmission. A system is said to be dissipative if this condition is met for all $\mathrm{t}_1 \geq \mathrm{t}_0 \geq 0$.

$$S\big(x(t_1)\big) - S\big(x(t_0)\big) \leq \int_{t_0}^{t_1} w(u(t), y(t))\, dt \qquad (3.3)$$

If a system is not dissipative, then excess energy is being generated from somewhere and the system may be inherently unstable if this energy build-up cannot be controlled. Excessive energy will take some form such as heat or vibrations that may negatively affect the system, so a dissipative system is preferred. However, it is important to note that a dissipative system does not guarantee stability. If a storage function does not consider every critical state, then any state not considered has the potential to cause instability. An example of this would be a state vector that contains variables related to motion would be able to predict instability due to motion, but energy causing an increase in system temperature would not be considered by the storage function and could lead to an unstable temperature and potential system damage.

A passive system is a specific kind of dissipative system where the initial energy storage $\mathrm{S}(0)$ is equal to 0 and the supply rate is defined as the product of inputs and outputs:

$$w\big(u(t), y(t)\big) = u^T(t)y(t) \tag{3.4}$$

This restrictive supply rate shows that the number of inputs must match the number of outputs for the definition of passivity to be met. The passive system is said to be lossless if the equality of (3.3) holds with (3.4), or strictly passive if the strict inequality holds. Passive systems have several properties that can help identify them and will help in control. One such property is that a passive system output signal is bounded to a phase shift within [-90°, 90°] from the sinusoidal input. This is a by-product of the power consumption behaviour; the input and output values are power conjugates and when multiplied together must return a positive value to show power consumption rather than power generation. A second useful property is that the eigenvalues of a passive system are guaranteed to be non-negative, however a system with non-negative eigenvalues is not guaranteed to be passive unless the system is symmetric. By simply viewing the plot of eigenvalues over all frequencies of the system (which can be turned symmetrical by $(A + A^T)/\,2$), determining passivity is as easy as confirming no negative eigenvalues on the produced graph. The physical interpretation of an eigenvalue being positive also ties to the power consuming behaviour of the system. In the frequency domain, this property is equivalent to confirming positive real condition (that the transfer function is a positive real function).

3.2.3.Passivity Indices

The degree of passivity, or passivity index of a system is a measurement of how much passivity is within a system, with a higher passivity index indicating a system is more passive. From the inequality in (3.3) it can be seen that a system is dissipative if not all energy supplied to the system is stored, however the amount of energy that is dissipated is not specified. A system that dissipates more energy would be considered more dissipative. Similarly, a passive system is considered more passive for systems with a larger inequality between the values. Naturally, subtracting value from the supply rate while maintaining the storage function will push the inequality closer to an equality and make the system less passive, while adding value will make the system more passive. As the supply rate function is dependent on both input and output there are two ways to adjust the supply rate: change the output through the inclusion of a static feedforward system, or change the input through the inclusion of a static feedback system. These techniques to achieve passivity are known as input feedforward passivity (IFP) and output feedback passivity (OFP), respectively. By using IFP or OFP, a passive system can be made non-

passive and a non-passive system can be made passive, which allows all the passivity-based stability conditions to be applied more globally.

Input feedback passivity, for a given passive system H, feeds the input signal *u* through a static gain *v* (*v* > 0), and subtracts from the output signal *y*. Due to the passivity of H and the static nature of the additional system, the combined system H̃ retains the storage function *S(x)* and the input signal *u*, but with the output $\tilde{y} = y - vu$. This structure can be seen as a block diagram in Figure 3.1(a). Assuming H̃ is passive, and the storage function for time t_i denoted as S_i, the passivity inequality can be expressed as:

$$S_1 - S_0 \leq \int_{t_0}^{t_1} (u^T y - \nu u^T u)\, dt \tag{3.5}$$

Equation (3.5) shows that higher values of ν will reduce the difference between stored energy and supplied energy over a time period [t_0, t_1] for the combined system H̃, and H is dissipative with respect to the supply rate function provided. This shows that the original system H has an excess of passivity of the amount ν, denoted as IFP(ν), as this is how much passivity can be compensated by H before the inequality is violated. The passivity must be provided by H and not the feedforward gain as the supply rate component from the feedforward block -$w_{ff}(u, y_{ff}) = -u^T y_{ff} = -u^T(\nu u) = -\nu u^T u$ is guaranteed to be non-positive. This violates what is known as the positive real condition. Every passive system is positive real, so identifying that a system is not positive real confirms the system is not passive.

Assuming H is non-passive, similar arguments can be made using a negative ν value such that the supply rate for H̃ is provided by $\tilde{w}(u, y) = u^T y + \nu u^T u$. In this case it is said that H has a lack of passivity of the amount ν, denoted by IFP(-ν), and is compensated by the feedforward gain (which no longer violates the positive real condition as ν is guaranteed to be negative) to force H̃ to be passive.

Output feedback passivity for a given passive system H, feeds the output signal *y* through a static gain *ρ* (*ρ* > 0), and adds to the input signal *ũ,* which is the input to the combined system H̃. The storage function and output signal *y* are retained, but the input to H becomes u = ũ + ρy. This structure can be seen as a block diagram in Figure 3.1(b). Assuming H̃ is passive, and the storage function for time t_i denoted as S_i, the passivity inequality can be expressed as:

$$S_1 - S_0 \leq \int_{t_0}^{t_1} (u^T y - \rho y^T y)\, dt \tag{3.6}$$

Equation (3.6) is the output feedback equivalent to (3.5) and shows that the original system H has excess passivity of the amount ρ, denoted as OFP(ρ). Similarly, assuming H is non-passive, a negative value of ρ will produce the supply rate $\tilde{w}(u,y) = u^T y + \rho y^T y$ and show a lack of passivity in H, denoted by OFP(-ρ), that can be compensated by the feedback gain to result in a passive H̃.

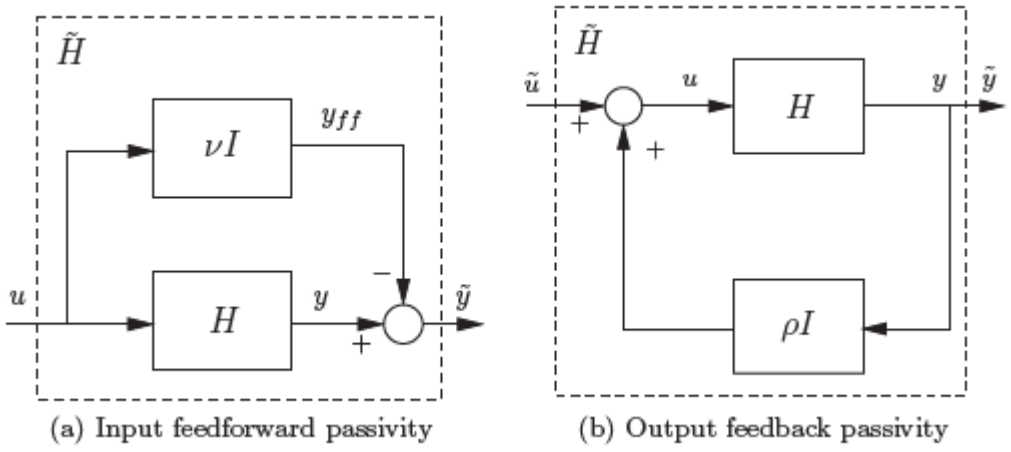

Figure 3.1. (a) Subtraction of a feedforward gain for input feedforward passivity, and (b) Addition of a feedback gain for output feedback passivity (Bao and Lee 2007)

3.2.4.Passivation via LMIs

If a system is not passive but lacks IFP or OFP, then the process can be converted to a passive system by adding the appropriate IFP or OFP components mentioned above to push the system towards passivity. This process is known as passivation and has many beneficial effects on the control of a system since passive systems controlled by a passive controller are also guaranteed to be passive. An important caveat is that an unstable system cannot be passivated by IFP and must either be stabilised beforehand, or use OFP to passivate. Similarly, OFP cannot passivate a system with non-minimum phase. These restrictions are briefly mentions to clarify that both IFP and OFP can be useful depending on the circumstances and system being worked upon, but neither are applicable in every scenario.

Obtaining the appropriate feedforward system can be achieved through the application of linear matrix inequality (LMI) techniques. An LMI is a mathematical representation of linear inequalities used to represent constraints of a system, often used in the field of optimisation.

The Kalman-Yacubovich-Popov (KYP) lemma is an LMI that relates the state space representation of a system to the stability and passivity of the system. Under the assumption that a system output equation has no feedthrough (the state space matrix D=0) the KYP lemma can be simplified to the positive-real (PR) lemma, which states the following:

A stable linear time invariant system is given by the state space equations provided in (2.1) *and* (2.2)*. The system is passive if and only if there exists a positive definite matrix P such that:*

$$\begin{bmatrix} A^T P + PA & PB - C^T \\ B^T P - C & -D - D^T \end{bmatrix} < 0 \qquad (3.7)$$

For matrix inequalities, matrix $P \in \mathbb{R}^n$ is considered positive definite if the following strict inequality is true:

$$x^T P x > 0, \qquad \forall x \in \mathbb{R}^n \qquad (3.8)$$

With inequality (3.7) showing passivity is dependent on state space representation it becomes clear that changing the state space matrices of a system can affect the passivity. Specifically, the output y is dependent on the output matrix C in the same manner that the output y is dependent on the feedforward gain used for IFP. By adding an additional feedforward block C_{ff} in parallel to the existing output matrix block C, the new combined system $\tilde{C} = C + C_{ff}$ can be appropriately selected to guarantee (3.7) and (3.8) are true and the new system is passive.

3.3. Methodology

3.3.1.Controller feedback architecture

For a system with a process G, the addition of a feedforward system G_{ff} allows for the joint transfer function $\tilde{G}$ to become a passive system. However as the process of the system is representative of its physical properties, the joint process should ideally be as close to the original as possible for best modelling practices. It is possible to consider the additional feedforward process as part of the controller rather than part of the plant. A semi-equivalent closed loop system can be constructed by redirecting the feedforward process into a feedback process creating an inner loop with the base controller K. Both designs can be seen in Figure 3.2, showing the joint process $\tilde{G}$ in (a) and the joint controller $\tilde{K}$ in (b). The feedback equivalent system shows a more intuitive structure with all control structures grouped together into a standard closed loop configuration often found in control theory literature. By calculating the transfer function equivalence of the total closed loop system (from r to ỹ/y) it becomes evident the system is not perfectly equivalent. The feedforward addition of G_{ff} produces an additional term in the transfer function at the expense of the guaranteed passivity of $\tilde{G}$. It can be noted, however, that the denominator term of the transfer function remains the same (shown below) and as such both systems can be guaranteed to have the same stability properties which is the predominant factor being analysed. Through the lens of stability both transfer functions are equivalent and guarantee passivity.

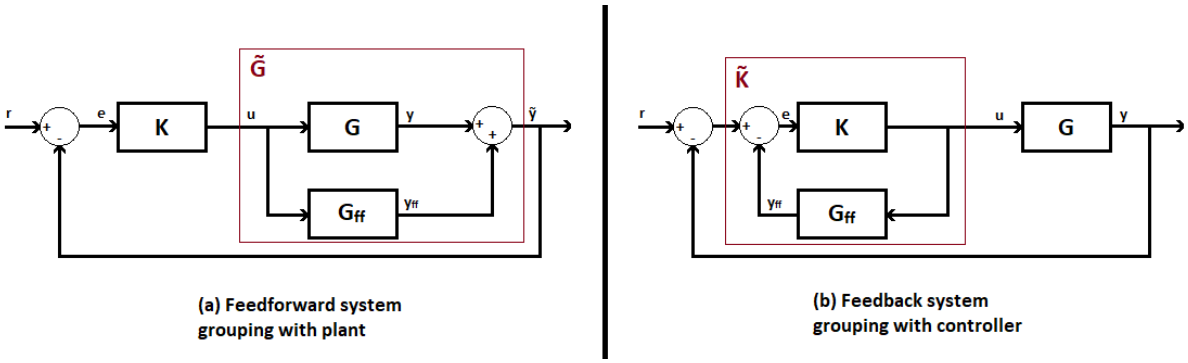

Figure 3.2. General block diagram structure for stability and passivity equivalence using: (a) A feedforward controller G_{ff}. (b) A feedback controller G_{ff}.

$\tilde{y} = y + y_{ff};\quad y = Gu;\quad y_{ff} = G_{ff}u$ $u = Ke;\quad e = r - \tilde{y};$ $u = K\left(r - \left(Gu + G_{ff}u\right)\right)$ $u = Kr - KGu - KG_{ff}u$ $u + KGu + KG_{ff}u = Kr$ $u = \frac{Kr}{1 + KG + KG_{ff}}$ $\tilde{y} = G\left(\frac{Kr}{1 + KG + KG_{ff}}\right) + G_{ff}\left(\frac{Kr}{1 + KG + KG_{ff}}\right)$ $\tilde{y} = \frac{GKr}{1 + KG + KG_{ff}} + \frac{G_{ff}Kr}{1 + KG + KG_{ff}}$ $\tilde{y} = \left(\frac{GK + G_{ff}K}{1 + KG + KG_{ff}}\right)r$ $G_{CL} = \frac{\tilde{y}}{r} = \frac{GK + G_{ff}K}{1 + KG + KG_{ff}}$	$y = Gu;\quad u = Ke;\quad e = r - y - y_{ff};$ $y_{ff} = G_{ff}u$ $u = K(r - y - G_{ff}u)$ $u = Kr - Ky - KG_{ff}u$ $u + KG_{ff}u = Kr - Ky$ $u = \frac{Kr - Ky}{1 + KG_{ff}}$ $y = G\left(\frac{Kr - Ky}{1 + KG_{ff}}\right)$ $y = \frac{GKr - GKy}{1 + KG_{ff}}$ $y + \frac{GKy}{1 + KG_{ff}} = \frac{GKr}{1 + KG_{ff}}$ $y = \frac{\frac{GKr}{1 + KG_{ff}}}{1 + \frac{GK}{1 + KG_{ff}}}$ $G_{CL} = \frac{y}{r} = \frac{GK}{1 + KG_{ff} + GK}$

For a SISO system all transfer functions are commutative and no discrepancy between GK and KG exist. For a MIMO system the multiplication of matrices must be performed more carefully, but as each matrix can be set to act as a left-hand side or right-hand side multiplication the transfer function can be manipulated to act accordingly. The transfer function also clarifies the desire for an optimally small G_{ff}, as this term tends towards 0 the two transfer functions converge.

The selection of a base controller is an important step that must occur before any IO pairing takes place. A proportional-integral-derivative (PID) controller is a commonplace controller with widespread application and comes with the beneficial property of being guaranteed to be passive (Romeo et al. 2021). A PID controller is a SISO system, however many can be combined

to a MIMO system with each individual PID being placed in the diagonal elements of the MIMO transfer function. This diagonal property guarantees the controller is decentralised with no crosstalk between signal inputs and outputs; a strictly 1-1 relationship is preserved. To retain this decentralisation, the diagonal property the feedback controller must not be broken by the feedback controller G_{ff}. This is achievable by structuring G_{ff} to be diagonal as well, which can be proven using the standard negative feedback loop equation for MIMO systems:

$$\widetilde{K} = \frac{K}{I + K * G_{ff}} \tag{3.9}$$

The base controller K represents the MIMO PID controller, the feedback controller G_{ff} represents the passivating controller, and the closed-loop controller $\tilde{K}$ would represent the equivalent transfer function matrix of the combination of these two controllers.

Structuring the feedback controller to be diagonal can be done by constructing the state space representation matrices {A, B, C} to all be block diagonal. If this condition is met, then the transfer function calculated through (3.2) guarantees a diagonal process. The control canonical form is constructed in such a way that the state matrix A and the input matrix B are both block diagonal by definition if the right matrix fraction description of the transfer function contains a diagonal denominator matrix. For an n-by-n MIMO system the denominator matrix will also be of size n*n. Each column will have n transfer functions that have potentially different polynomial degrees. The maximum degree for the i^{th} column is called the i^{th} column degree and is represented by k_i. With this information, the state space representation can be constructed to guarantee a diagonal transfer function, as such:

$$A_c = A_c^0 - B_c^0 D_{hc}^{-1} D_{lc} \tag{3.10}$$

$$B_c = B_c^0 D_{hc}^{-1} \tag{3.11}$$

$$C_c = N_{lc} \tag{3.12}$$

where A_c^0 and B_c^0 are block diagonal matrices formed from n matrices which vary in size depending on the column degree:

$$A_c^0 = block\ diag \left\{ \begin{bmatrix} 0 & 0 & \dots & 0 & 0 \\ 1 & 0 & \dots & 0 & 0 \\ 0 & 1 & \dots & 0 & 0 \\ \vdots & \vdots & \dots & \vdots & \vdots \\ 0 & 0 & \dots & 1 & 0 \end{bmatrix}, k_i \times k_i, i = 1, \dots, n \right\} \tag{3.13}$$

$$B_c^0 = block\ diag\left\{\begin{bmatrix}1\\0\\\vdots\\0\end{bmatrix}, k_i \times 1, i = 1, \dots, n\right\} \quad (3.14)$$

The coefficient matrices D_{hc}, which contains the highest coefficient of each element of the denominator matrix $D(s)$; and D_{lc}, which contains the remaining lower coefficients for each element, will be diagonal and block diagonal respectively as a result of the denominator matrix $D(s)$ being diagonal. An example of this is provided in 3.3.3.3 for clarity. These conditions lead to the transfer function $G_c(s)$ being diagonal if the matrix C_c is also block diagonal, which can be mathematically deduced through equations (2.3), and (3.10) through to (3.12). Using this derived constraint of C_c, finding a solution to the feedforward system can be achieved by identifying this as an optimisation problem. By employing an optimisation solver the output matrix C_c can be chosen such that it must be block diagonal, which guarantees the condition that our feedback controller (G_{ff} in Figure 3.2) is diagonal and our closed loop controller K̃ remains diagonal and appropriately designed for a decentralised system. The optimisation toolbox YALMIP (Lofberg 2004) is used in conjunction with the optimisation solver MOSEK to generate the results in MATLAB 2021a.

3.3.2.Process control configuration selection

With the structure of the controller and feedback controller known, the system process can be permuted in such a way that the input-output relations can be changed as desired. Swapping the ith and jth rows of the process transfer function will effectively relabel the ith and jth outputs of the system to one another while still representing the same physical system. If no controller is present, performing this action does not affect the system at all and can be freely performed. However once the system is coupled with a controller the ordering of inputs and outputs becomes important. Changing the ordering of the process but not the controller will result in a different control configuration where the controller relates inputs to different outputs. This is known as control configuration selection (CCS) and must occur before the controller is designed, since the design will be dependent on this IO coupling. With the prior restriction that the controller must be diagonal, a controller can be designed for every CCS to determine which process is best suited for the desired operation conditions.

Since it is subjective and dependent on the user it is difficult to identify any CCS as the undisputed best performing configuration, although some can be disregarded as definitely

worse. For this research, the CCS that required the least amount of control adjustment to become both stable and passive was favoured. This condition can be mathematically determined by the output matrix C_{ff} from the feedback controller G_{ff}. The state matrix A_{ff} and input matrix B_{ff} are set to be direct copies of the state and input matrices of the process G to simplify the addition of the two transfer functions:

$$A_{ff} = A; \quad B_{ff} = B; \quad C_{ff} \textit{ will be calculated by the optimisation algorithm}$$

From Figure 3.2(a) the effect of the feedforward controller can be seen as a parallel addition of transfer functions which is achieved mathematically by $G_{tot} = G + G_{ff}$. Using the state space representation and the previously defined values of $A_{ff} = A$ and $B_{ff} = B$, the combined transfer function becomes:

$$\begin{aligned} G_{tot} &= C(sI - A)^{-1}B + C_{ff}\left(sI - A_{ff}\right)^{-1}B_{ff} \\ &= C(sI - A)^{-1}B + C_{ff}(sI - A)^{-1}B \\ &= \left(C + C_{ff}\right)(sI - A)^{-1}B \end{aligned} \tag{3.15}$$

From this representation it becomes clear that a smaller value of C_{ff} will result in less change to the overall system. The optimisation solver will aim to minimize the spectral norm of C_{ff} to find an output matrix value that satisfies all the constraints of the system while also resulting in the least overall variance from the original process. The spectral norm of a real matrix M is defined as the square root of the maximum eigenvalue of $M^H M$, where M^H is the Hermitian (complex-conjugate transpose) matrix of M.

$$||M||_2 = \sqrt{\lambda_{max}(M^H M)} \tag{3.16}$$

$M^H M$ is guaranteed to have real eigenvalues and as such the spectral norm represents the largest scaling factor caused by the transfer function. MIMO systems will have multiple scaling/gain factors in the frequency domain, and so system gain over frequency is defined as the value that maximizes gain for an associated input (Goodwin, Graebe, and Salgado 2000). By running the optimisation solver for each CCS and comparing spectral norm values it can be determined which CCS is able to handle the least amount of change from the original system while still achieving the requirements of passivity. The CCS that produces the smallest possible spectral norm will be selected as the optimal input-output pairing as it will best represent the unmodified physical system that will be able to be decentralised and controlled.

3.3.3. Software Development

With the overall goal of creating a controller that is able to guarantee system stability and passivity in the most efficient way possible, it is required to develop automated software in stages that build off the previous stages. As the system is required to create a control system that contains several characteristics, each characteristic is separated into a separate subtask to help structure the software development and guarantee each characteristic is met sufficiently. The stages of development can be described as: guaranteeing system stability for safety and functionality purposes, guaranteeing system passivity for safety with human coupling, controller diagonalisation to guarantee appropriate state-space representation for the final step, and the combination of all three steps for each IO pairing to identify the optimal stable-passive-diagonalised controller system. Each stage is detailed below with the underlying ideas of the implementation, however the stages of development and troubleshooting are not discussed in detail as they do not offer any insight to the overarching goals and objectives of the project.

3.3.3.1. Progression stage 1 - Stabilisation

The first task was to become familiar with the required optimisation software YALMIP and develop a practical understanding of how to effectively utilize LMIs in MATLAB. This task was performed by constructing a script that guaranteed stability of a system using full state feedback. For full state output feedback for the system the output vector y must equal the state vector x, and as such the output matrix C must equal unity ($C = 1$) and the feedthrough matrix D must equal zero ($D = 0$). Alternatively, by feeding the state vector directly to the feedback line (shown in Figure 3.3) the output of the system can remain a separate value while still benefiting from the full-state feedback. By solving the state space equations (2.1) and (2.2) for this system we find the new equation:

$$\dot{x} = (A - BK)x \tag{3.17}$$

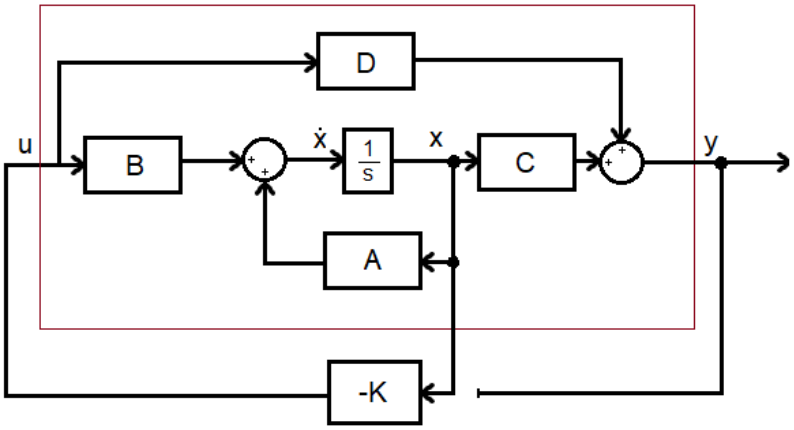

Figure 3.3. Block diagram structure of a system utilising full-state feedback without utilising full-state output feedback. The system in the red box can be stabilised by appropriately setting feedback gain K

For a system with the state vector x, the energy within the system is described by the function $V(x)$ and is calculated as:

$$V(x) = x^T P x \tag{3.18}$$

where $P \in \mathbb{R}^{nxn}$ is symmetric and will be dependent on the system. As this value function represents the physical meaning of contained energy it can also be reasoned that V(x), and therefore also P, must be greater than zero (since no system can have negative energy). Similarly, for the system to be stable the rate of change of system energy must be negative to represent an energy dissipation behaviour. A system with a positive energy derivative would represent the accumulation of energy which is not infinitely stable. This condition can be represented as the inequality below, where the matrix Q is simply a buffer from zero to remain a set distance from the instability condition:

$$\frac{d}{dt}V(x) = \dot{V}(x) < Q \tag{3.19}$$

From this inequality, by applying the product rule and substituting in the traditional state space equation $\dot{x} = Ax$ it is possible to derive the Lyapunov inequality which guarantees system stability in the form of a linear matrix inequality:

$$\dot{V}(x) = \dot{x}^T P x + x^T P \dot{x}$$

$$= (Ax)^T P x + x^T P (Ax)$$

$$= x^T A^T P x + x^T P A x$$

$$= x^T (A^T P + P A) x$$

These values can be reduced into the inequalities known as the Lyapunov stability condition, which states that a differential equation is stable if and only if there exists a matrix P such that:

$$A^T P + PA < Q \tag{3.20}$$

$$P > 0 \tag{3.21}$$

If the A matrix in (3.20) is substituted with the appropriate matrix $\tilde{A} = (A - BK)$ from equation (3.17) then a bilinear inequality is formed as the goal of the task is to design controller K to guarantee stability of the system, and P must be determined to exist. As neither P nor K are known values, the multiplication of these unknown variables causes the bilinear inequality that cannot be solved to an optimal solution. The inverse matrix P^{-1} can be projected onto this bilinear matrix inequality to allow for some substitutions and simplify the calculation:

$$P^{-1}[(A - BK)^T P + P(A - BK)]P^{-1} < P^{-1}QP^{-1}$$

$$P^{-1}A^T PP^{-1} - P^{-1}K^T B^T PP^{-1} + P^{-1}PAP^{-1} - P^{-1}PBKP^{-1} < P^{-1}QP^{-1}$$

$$P^{-1}A^T - P^{-1}K^T B^T \ + AP^{-1} - BKP^{-1} < P^{-1}QP^{-1}$$

By now substituting matrix $R = P^{-1}$ and $S = -KP^{-1}$, and remembering that P is symmetric, the inequality becomes a new set of linear matrix inequalities. It can be trivially shown that if a matrix is positive definite then it is invertible, and the inverse matrix is also positive definite. This guarantees the positive definite constraint of P is retained for the new matrix R:

$$RA^T + S^T B + AR + BS \ - RQR < 0 \tag{3.22}$$

$$R > 0 \tag{3.23}$$

As LMIs are convex functions this form can be easily solved through the use of a convex solver. However the solver is only able to minimize one matrix while we want both P and K to be minimized. To achieve this we must convert the algebraic equation in (3.22) into a matrix using the Schur complement, which states that a block matrix M will have the equivalent Schur complement:

$$M = \begin{bmatrix} A & B \\ C & D \end{bmatrix} \tag{3.24}$$

$$M/D \coloneqq A - BD^{-1}C \tag{3.25}$$

By observing the equation in (3.22) it can be deconstructed to have the necessary form for the Schur complement in (3.25), which means the equivalent matrix M can be defined as:

$$M = \begin{bmatrix} RA^T + S^T B + AR + BS & R \\ R & -Q^{-1} \end{bmatrix} < 0 \quad (3.26)$$

The optimiser command in MATLAB uses two parameters: constraints and objective. The constraints parameter is a set of LMIs that detail the restrictions placed on the system. For the example of stability, the two constraints include the LMI (3.22) and (3.23) which were derived from the Lyapunov inequality (3.20), and (3.21). As the variables A and B are predetermined by the system, only values for R and S are calculated by the solver. The value of Q is determined by the designer depending on their desired stability margin, but for testing purposes was set to a zero matrix. These constraints determine the feasible set: a set of every possible unknown variable combination that guarantees every constraint is satisfied. The unknown variables are known as standard decision variables within YALMIP. From this set, the optimiser aims to find the optimal solution with respect to the objective, which in this stability example, is the inequality (3.37) as it contains all the necessary decision variables R and S. The objective variable will almost always be represented by a function of the decision variables that are able to be set by the optimisation software with the aim of minimising said function by altering these variables. Optimal values for P and K matrices can then be calculated from the results of the optimiser and their direct substitutions. A generic representation of the optimiser function is as follows:

Minimise: Objective

Subject to: Constraints

3.3.3.2. Progression stage 2 - Passivation

The next two tasks involved aiming to passivate a system similar in structure seen in Figure 3.2(a), with the additional restriction that the system G is fully controllable, fully observable, and stable. Although these restrictions limit the number of cases that this passivation method can be applied, by using the techniques developed in stage 1, even systems that are unstable can be stabilised to then further be passivated.

Passivation was split into two different approaches to determine how the additional system would combine with the existing system. For the first case the state space matrices A and B are guaranteed to be equivalent between the default system and the added feedforward system ($A = A_{ff}$ and $B = B_{ff}$), and as such the state vector is equivalent between systems and simple

vector addition is possible. This results in the final number of states for the total system to remain the same. For the second case the state matrices are not guaranteed to be shared between systems. This leads to the total system state matrices becoming a block diagonal matrix of both state matrices $A\ and\ A_{ff}$, while the input matrix becomes a vertically concatenated matrix from both input matrices $B\ and\ B_{ff}$. This results in the final number of states for the total system to be the sum of each state vector, which for our experiment meant a doubling of states. Increasing the number of variables to calculate causes much less efficient optimisation algorithms and will be exacerbated by any dimensionality issues that will occur in the future.

$$A_{tot} = \begin{bmatrix} A & 0 \\ 0 & A_{ff} \end{bmatrix}; \quad B_{tot} = \begin{bmatrix} B \\ B_{ff} \end{bmatrix}$$

As the feedforward system is completely within the designer's control, having the control matrices equate in both cases can always be achieved. This is why both methods are valid and reveal helpful information about system passivation.

Both approaches use the optimisation solver to choose an output matrix C_{ff} for the additional system G_{ff} that would cause the total system G_{tot} to become passive. The passivity condition can be achieved through using the KYP lemma as the constraints to the optimisation problem, seen in (3.7) and (3.8). The objective was originally set to the output matrix C_{tot} which would then calculate the feedforward matrix $C_{ff} = C_{tot} - C$, however an alternative approach was tested using a new scalar value γ that acted as an upper limit to the spectral norm of C_{tot}. By optimising, and hence minimising, the upper limit of eigenvalues of the output matrix, the overall disturbance caused by the system perturbation can be suppressed to a value as small as possible. As this is a more direct optimisation of desired behaviour it is expected that results from this method will be superior for the purposes of the overall project, however both methods are tested for the purposes of analysis. Details of this implementation are provided in section 3.3.3.4, which is the stage of the project the technique was developed. The spectral norm optimisation was retroactively applied to the attempts at passivation to confirm its functionality. An important note for this section is that the output matrix being determined by the optimiser is non-diagonal or non-block diagonal. This is the primary distinction between stages 2 and 3.

Tests to confirm the PR lemma functioned as intended and were integrated into the automation script by plotting eigenvalues of each transfer function over frequency $s = j\omega$. With the knowledge that a passive system will have positive eigenvalues for all values of ω it can be easily visually determined whether a system is passive or not. Some systems were deemed non-

passive due to negative eigenvalues, but on closer inspection of the data, these values were extremely small in magnitude and were attributed to rounding errors in the program. Over several system tests, all behaviours produced met their expectations, giving great confidence in the reliability of this method for guaranteeing passivity.

3.3.3.3. Progression stage 3 - Diagonalisation

The next stage in development was to make use of matrix fraction descriptions to guarantee a specific state space representation. A MATLAB script was created to take any MIMO proper transfer function as an input and return the MFD with a diagonal denominator matrix as the output. This structure, when deconstructed using the techniques outlined by Kailath (1980: 403) will always generate a state matrix A and input matrix B that are block diagonal (see (3.13) and (3.14)). Through some fairly trivial algebra it can then be proven that the transfer function of this state space realisation, when calculated from (2.3), will be diagonal if the output matrix C is chosen such that it is also block diagonal. An example of the MFD deconstruction for the coefficient matrices $D_{hc}(s)$ and $D_{lc}(s)$ is provided below:

$$N(s) = \begin{bmatrix} n_{11} & n_{12} \\ n_{21} & n_{22} \end{bmatrix}$$

$$D(s) = \begin{bmatrix} a_4 s^4 + a_3 s^3 + a_2 s^2 + a_1 s + a_0 & 0 \\ 0 & b_2 s^2 + b_1 s + b_0 \end{bmatrix}$$

$$D_{hc}(s) = \begin{bmatrix} a_4 & 0 \\ 0 & b_2 \end{bmatrix}; \qquad D_{lc}(s) = \begin{bmatrix} a_3 & a_2 & a_1 & a_0 & 0 & 0 \\ 0 & 0 & 0 & 0 & b_1 & b_0 \end{bmatrix}$$

$$k = [4 \quad 2]$$

By performing these steps and substituting these values into equations (3.10) to (3.14) we can guarantee the new system G_{ff} will be diagonal and will only affect the direct input-output relations with no crosstalk. The output matrix C_{ff} can be guaranteed block diagonal when using the optimisation solver by setting the standard decision variables only in the appropriate positions of the matrix with constant values of 0 in all others. The system is able to solve for C_{ff} and produce a real matrix with block-diagonal structure, and as such it is possible to restrict the feedforward system G_{ff} to be diagonal and avoid crosstalk. The intention of the overall experiment is to find the smallest feedforward system that guarantees passivity while also assuring the addition of a feedforward perturbation does not cause any signal crosstalk within the system. A feedforward system that does not change the output at all would be optimal, and as such minimizing the output matrix of the feedforward system is the best approach to achieve this minimal divergence. The two major developments of this stage

included an automated script to calculate an MFD from a transfer function that had a diagonal denominator matrix structure, and the extraction of appropriate coefficient values to substitute into a restricted state space equation that guarantees a diagonal transfer function.

A final note is to understand that not every system will have a solution set when constrained by so many inequalities. In some cases the optimiser will return an "infeasible solution" which suggests that the input system is not capable of being passivated under these strict conditions of diagonal control. Solutions in this category must be analysed separately, and discussion as to why the solution is infeasible is required.

3.3.3.4. Progression stage 4 – Pairing Selection

The last stage of progression is to alter the existing script to find the minimal control matrix for each permutation of IO pairings. This is, although fairly simple to implement, a computationally intensive task for systems with large amounts of inputs and outputs. For n inputs and outputs there will be n-factorial permutations that require cycling, with each optimisation during testing taking approximately 5 seconds to complete. Under these assumptions, any system with greater than six inputs and outputs will take over 6 hours to compute. All experiments were performed on systems with three or less inputs and outputs.

The method for minimizing the output matrix was also adjusted at this stage in the project. Instead of minimizing the matrix C, the goal is to now minimize the spectral norm of C, effectively minimizing the energy output of the system. This conclusion was reached as the spectral norm of a matrix is equal to the maximum singular value of the matrix, which in turn is equal to the square root of the largest eigenvalue of the matrix multiplied by its complex-conjugate transpose as shown in (3.16). The eigenvalues of a system are inherently connected to the stability and energy state of a system, as they are used to determine if a fixed point is stable or unstable in differential equations.

To find the pairing that would produce the smallest spectral norm the optimiser must aim to minimize the maximum eigenvalue of $C^T C$. Unfortunately the optimiser requires the minimising parameter be a well-defined variable and the eigenvalues of a matrix must be calculated separately, thus cannot be used as the minimizing parameter of the solver. Instead, a new parameter γ was created with the intention to minimize gamma and add the constriction $C^T C < \gamma$. This constraint will effectively minimize the spectral norm through the suppression of its upper bound γ. To utilize this within the convex solver, the algebraic inequality

must take the form of a linear matrix inequality. This was achieved through another implementation of Schur's complement. The algebraic inequality is rearranged into the form $M/A \coloneqq D - CA^{-1}B$, as follows:

$$\gamma I > C^T C$$

$$\gamma I - C^T(I^{-1})C > 0$$

$$M = \begin{bmatrix} I & C \\ C^T & \gamma I \end{bmatrix} > 0 \qquad (3.27)$$

By combining the upper bound γ and the output matrix C into one constraint, both variables will determine the feasible set while also finding the smallest spectral norm that also satisfies the other constraints assuring passivity and stability. The output matrix C_{tot} being suppressed in this experiment is the combined output matrix for the original system G and the feedforward system G_{ff}. The PR lemma (3.7) and (3.8) is also still used as a constraint to retain overall system passivity, and as the total system is guaranteed passive, this guarantees the feedforward system is also passive if the original system is assumed non-passive. Since the original system is constant and cannot be changed, suppressing the combined system only suppresses the feedforward system component in $C + C_{ff}$, as seen in (3.15). Once the optimisation was completed the spectral norm value was saved in an array and the next IO permutation was run through the optimiser through a simple rearranging of the appropriate matrices. A basic example of these permutations is shown below, with a generic input u and output y subject to a diagonal transfer function.

$$\begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix} = G\{1,2,3\} * u = \begin{bmatrix} g_1 & 0 & 0 \\ 0 & g_2 & 0 \\ 0 & 0 & g_3 \end{bmatrix} * \begin{bmatrix} u_1 \\ u_2 \\ u_3 \end{bmatrix} \qquad (3.28)$$

Due to the diagonal nature of the transfer function it is clear that the output y_1 is only affected by the input u_1, y_2 only affected by u_2, and y_3 only affected by u_3. If the pairing needs to change, then altering the order of the rows of the transfer function will cause this redirection of inputs and outputs. If the base transform function G is shown above with the permutation $G\{1,2,3\}$, this symbolises the transfer function with the rows of order 1, 2, 3. To swap input 2 to control output 3 and input 3 to control output 2, the transfer function must be changed to the permutation $G\{1,3,2\}$, as presented below:

$$\begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix} = G\{1,3,2\} * u = \begin{bmatrix} g_1 & 0 & 0 \\ 0 & 0 & g_3 \\ 0 & g_2 & 0 \end{bmatrix} * \begin{bmatrix} u_1 \\ u_2 \\ u_3 \end{bmatrix} \quad (3.29)$$

It can be trivially proved that changing the row order of a matrix is achieved by multiplying the base matrix by an identity matrix with shifted row order to match the desired row order. As the transfer function can be calculated as in (3.15), it then becomes clear that to change the IO pairings only the row order of the C matrix must be permuted. As the IO pairings of the original plant are what determines the control configuration selection, only G must be permuted and not G_{ff}. From this, only C will be altered between permutations while C_{ff} will remain block diagonal for every iteration to retain the diagonal nature of G_{ff}. For a 2-IO system, the optimisation will be performed on the system twice, with the only difference between run-throughs being the orientation of the C matrix from the base transfer function. This matrix will have a structure similar to:

$$C\{1,2\} = \begin{bmatrix} a & b & c & d & e \\ 0 & 0 & 0 & f & g \end{bmatrix}; \qquad C\{2,1\} = \begin{bmatrix} 0 & 0 & 0 & f & g \\ a & b & c & d & e \end{bmatrix}$$

Once all control configurations were completed, the pairing that resulted in the lowest spectral norm value for C_{ff} was deemed the best configuration and chosen as the system best suited for decentralisation of a MIMO system. To give credence to the recommended input-output pairing that was recommended by this optimisation technique, several other CCS methods were implemented for several example systems to compare recommendations.

3.3.3.5. Procedure Summary

An overall summary of how each developmental step is employed for the overall project is given below. To determine the CCS that results in the smallest spectral norm while also leading to decentralisation through passivation, the following steps are performed:

1. Given an m-by-m MIMO system plant G, confirm stability or use full-state feedback to achieve stability
2. Construct a right MFD of the system such that the denominator $D(s)$ is diagonal. Use $D(s)$ to choose the state space matrices A_c and B_c provided in (3.13) and (3.14)
3. Use an optimisation solver to determine an output matrix C_{ff} to be used in conjunction with the aforementioned matrices A_c and B_c such that a new transfer function G_{ff} is diagonal and the combined system $G_{tot} = G + G_{ff}$ is passivated through feedforward system addition.

4. Repeat step 3 for every possible permutation of rows in the plant G transfer function matrix to represent each possible CCS/IO pairing
5. For each pairing, calculate the spectral norm of the optimised C_{ff} matrix and compare values to identify the pairing that results in the lowest spectral norm
6. Design a MIMO PID controller for the selected CCS pairing through standard tuning methods

The optimisation software is licenced by MOSEK for semidefinite programming, while YALMIP is an open-source MATLAB optimisation toolbox. For the purposes of this project these tools are used to create variables that can be automatically calculated given specific conditions and goals. The optimize function is the main draw of this approach to passivation, as it allows linear constraints to be listed together to frame the feasibility set and then uses the goal to find the optimal value from that set. “Optimize(constraints, goal)” will find the feasible set of states that meet every condition contained in “constraints”. These constraints most often exist in the form of an LMI, although simple scalar constraints are also valid inputs. Once the feasible set is identified, the state that returns the minimal value of the “goal” variable is returned from the optimiser. Using this solution for C_{ff} allows the calculation of G_{ff} that guarantees passivity for all frequencies and presents the adjusted mathematical model and the original model (which represents the real-world physical system) as closely aligned as possible. A closer and better model results in more accurate simulations and the minimal amount of excess passivity helped reduce the conservative design features of the controller which may have led to unwarranted energy dissipation and lower performance.

One potential point of contention is how the minimal passivity is applied across the frequency domain. The optimiser guarantees the system is passive across all frequencies, however it is not clear that the minimal excess passivity is added to the system for all frequencies or only for the frequency bands that required the adaptation. If a system is passive at all frequencies excluding a small frequency band, the system may add enough passivity to all frequencies to remove this non-passive behaviour. Although the minimisation of the spectral norm should prevent this, and some experiments do seem to function as intended, some transfer functions are simply given extreme amounts of passivity to guarantee passivity without any frequency-dependent bumps. No clear explanation for this has been found, however since these results produce equally large spectral norms they are naturally filtered out during the control configuration selection process.

MATLAB scripts were built for each step to prove each concept is achievable as an isolated experiment. All steps were then combined into one single script to confirm the workflow can always receive a MIMO plant and return a stable, passive, decentralised MIMO plant as the output. All these steps allow the MIMO PID controller to appropriately control each IO pairing as individual SISO systems; crosstalk between variables will be reduced as much as possible. For the purposes of comparison, a MATLAB script was created to try and replicate the results displayed by (Bao et al. 2007). This script was successfully created, however slight deviations between some examples were present. These differences showed a change in frequency bandwidth by approximately a factor of 10, however this difference was present for all tests and as such the pairing conclusions that were made from the test remained the same. Due to the fact that pairing conclusions were the same as the paper, and the functionality of this test was not of high priority for the project as a whole, the reasons for the discrepancy was not heavily researched.

3.4. Results

Each stage of development relied upon small testing examples to show each proof-of-concept working as intended on at least one model. Although the examples may not be completely inclusive of all possible cases and not as rigorous as an all-encompassing mathematical theorem, the results presented do show the viable application of the underlying theory for some base cases. Scenarios with obvious exceptions will be explicitly mentioned with potential justifications of the differing results, however there are likely to be many different control systems that do not function as intended that are not discussed here. The goal of this section is simply to achieve basic applicability in some base cases.

Although the results would be more meaningful if the examples were based around robotic systems or ankle-foot dynamics, the formulation of these examples is a non-trivial task and sourcing existing transfer functions was not easily achievable. To progress with the true goal of this chapter, the development of a CCS method relying on optimisation tools, the system models used for testing are simple and sourced from other academic literature. The development of a mathematical model for an ankle-foot robot can be achieved, but will be dependent on the physical properties of the machine such as mass and length. It is therefore best practice to confirm the ideas of the mathematics before committing to a robotic build that would incur financial costs and restrict future designs. The robotic design was planned for a future stage of

the overall project, however due to the COVID-19 pandemic this plan was altered to better suit the available facilities such as simulation work.

3.4.1. Stage 1 – Stabilisation

By implementing the LMI restrictions derived from the Lyapunov stability inequality onto a simple system it is possible to directly compute the appropriate decision variables that stabilise the system. The following system was used for testing purposes:

$$A = \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}; \quad B = \begin{bmatrix} 0 \\ 1 \end{bmatrix}; \quad C = [1 \quad 0]; \quad D = [0] \tag{3.30}$$

This system is unstable, which can be determined by observing the Bode plot and the phase and gain margins of the system. Present in Figure 3.4, the magnitude of the system is seen to be approximately 0 dB for all frequencies below 0.3 Hz. The phase of the system is approximately -180° for all frequencies below 0.03 Hz. These two characteristics combined suggest the system is unstable for frequencies lower than 0.03 Hz and has a very small phase margin for frequencies less than 0.3 Hz which suggests the system is close to instability for low frequencies. Alternatively, calculating the transfer function equivalence for the system reveals the characteristic equation of $s^2 - 5s - 2$ which has a root on the right-half plane and is therefore unstable (calculations not included as they are trivial and unimportant).

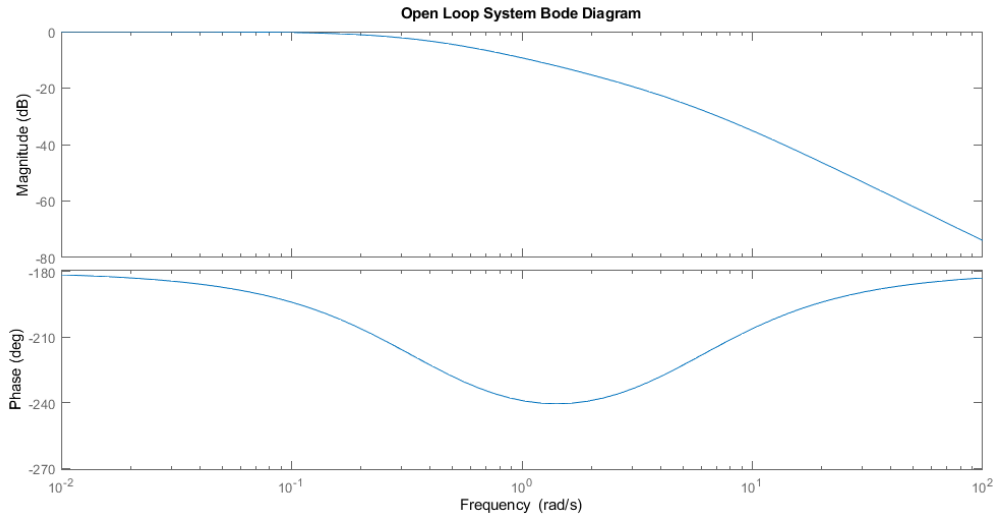

Figure 3.4. Bode plot of the open loop system presented in state space representation in (3.30). Gain and phase margin suggest the system is unstable for small frequency values

Running the optimiser to calculate the values of P and K results in one optimisation cycle for each element in the objective matrix M. Three of the four return an error reporting the problem is “primal-infeasible” due to an unbounded objective function. The other result returns no errors and reports the problem is “primal and dual feasible” with an optimal solution achieved. If no

objective is assigned and the optimiser is simply required to find a feasible solution, then no errors are produced and values for P and K are calculated. These values are similar to the previous calculation, although there are small differences in the values.

$$P_M = \begin{bmatrix} 2.5655 & 1.1471 \\ 1.1471 & 1.4184 \end{bmatrix}; \qquad K_M = [7.8054 \quad 6.4688];$$

$$P_{noM} = \begin{bmatrix} 2.4078 & 1.0048 \\ 1.0048 & 1.4030 \end{bmatrix}; \qquad K_{noM} = [7.2294 \quad 6.1128]$$

It is unclear what causes the discrepancy between results and why the approach with no objective resulted in smaller gain values. It is possible that the additional terms in the matrix M that had no bound caused the other values to adjust in non-optimal ways, while the lack of objective allowed the algorithm to simply select the optimal values for the decision variables. Since the removal of the Schur complement in the objective left it unused a final alternate approach was invented, using the Schur inequality (3.26) as a constraint in place of the inequality (3.22). Initial results showed a drastic change in P values to be multiplied by 10^{10} and K values to reduce to approximately 5.5, however this was caused by the singular nature of Q and the inclusion of its inverse in the new constraint. By adjusting the zero matrix to the identity matrix with a coefficient of 0.001 the values converged back to reasonable values:

$$P_{Mcon} = \begin{bmatrix} 0.0158 & 0.0076 \\ 0.0076 & 0.0066 \end{bmatrix}; \qquad K_{Mcon} = [12.6965 \quad 9.7507]$$

The change in Q matrix made no significant difference to the previously calculated K or P values, so the results of using the Schur inequality is a decrease in the P matrix as a trade-off for an increase in K values. As P values have no effect on the control of the system directly this result is considered worse and the previous approach of simply not utilising the Schur complement seems to be superior. Due to the complexity of the optimiser, this inconsistencies between constraint/objective structures will not be looked into further.

Regardless of the error messages, values for P and K are returned that satisfy both the constraints and do stabilise the system when used as closed loop feedback gain. The Bode plot for the new closed loop system is shown below with a phase of -180° for large frequencies greater than 10 Hz, where the magnitude has already decayed to a value of around -30 dB. This shows a very stable system with a gain margin of over 30 dB and an infinite phase margin since gain magnitude never crosses the 0 dB line. The closest point occurs around 2.4 Hz with gain at -2.7 dB and a phase of -76.6°; the system is very stable at all frequencies.

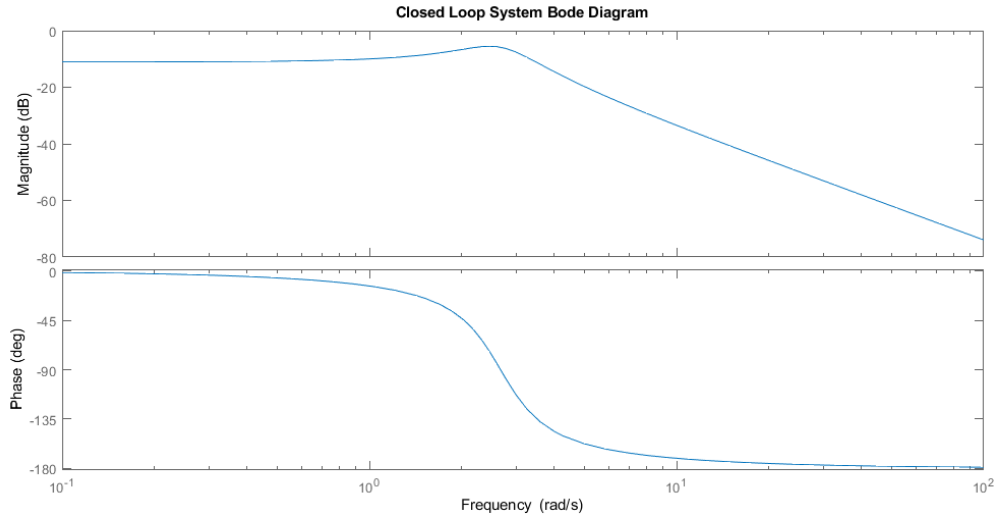

Figure 3.5. Bode plot of the closed loop system presented in state space representation in (3.28) combined with full-state feedback K. Gain and phase margin suggest the system is stable for all frequency values

The creation of this new feedback system relied on the premise that full-state feedback was utilised, yet based on the output matrix C in (3.32) and the dimensions of these two values, it appears full-state output feedback was not utilised, with instead the output only being equal to the first state. This is due to the structure seen in Figure 3.3 where the state vector is fed back directly without relying on system output. This is being reiterated as for more complex systems with more than one input and output can cause confusion in how values are calculated. Another example involving two inputs and one output was tested and also showed a stable system from the optimised K values.

This test showed the ability to use optimisation solvers to determine feedback gain values that would guarantee the specific system characteristic of stability. With the proof of concept successful, the next stage was to find optimal values that would guarantee system passivity.

3.4.2. Stage 2 – Passivation

A) Equivalent State Spaces

As discussed earlier, the addition of a feedforward system can be simple if the state space of both systems are equivalent, as it allows the easy combination of transfer functions. Applying this method first, the KYP lemma was applied as a constraint to a SISO system with the combined output matrix C_{tot} as a decision variable while all other state space matrices were set constant.

$$A = \begin{bmatrix} -4 & 0 & 0 \\ 0 & -3 & 0 \\ 0 & 0 & -1 \end{bmatrix}; \quad B = \begin{bmatrix} 1 \\ -4 \\ 5 \end{bmatrix}; \quad C = [1 \quad 2 \quad 3]; \quad D = 0 \tag{3.31}$$

The example system was selected to be stable, full controllable, and fully observable without any alteration, but not passive. These were the necessary conditions of passivation, and were tested by checking no eigenvalues were positive, and checking the controllability and observability matrix were both full rank. Once confirmed, the YALMIP optimiser selects an optimal value for C_{tot} that is restricted to the constraints of (3.7) and (3.8), and from that value the additional feedforward output matrix C_{ff} can be calculated from the equation (3.15) as $C_{tot} = C + C_{ff}$. To confirm the resulting system was passive, the eigenvalues of each system $(G, G_{ff}, and\ G_{tot})$ were calculated and plotted with respect to frequency.

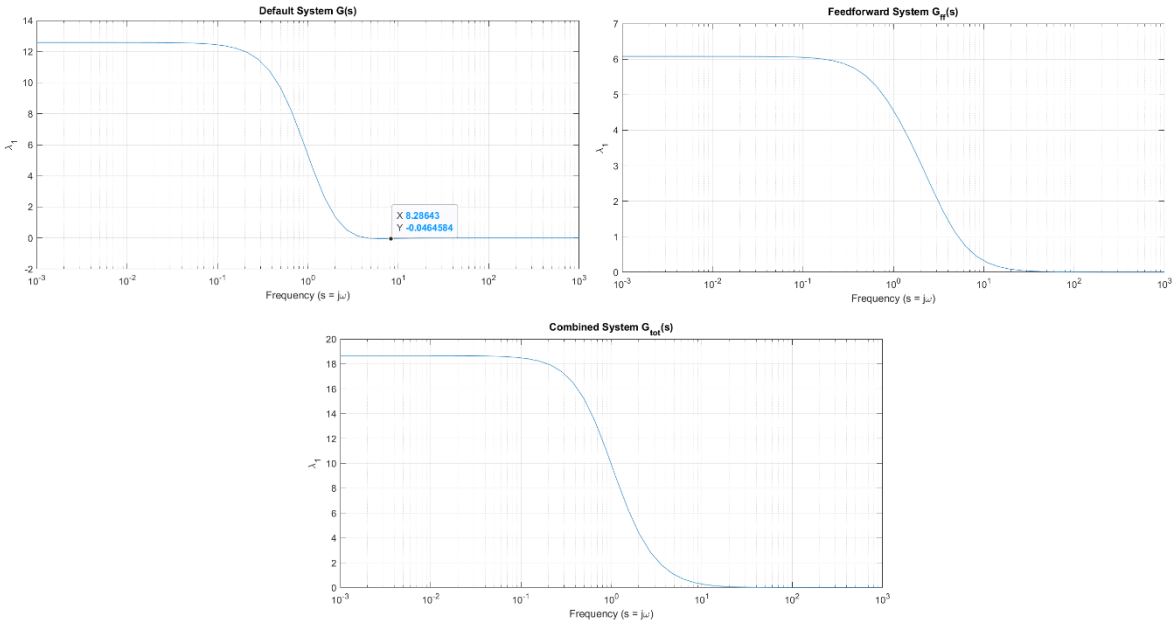

Figure 3.6. System passivity represented by frequency-dependent eigenvalues for the objective of C_{tot}. The default system shows slightly negative eigenvalues between 3-10 Hz. The feedforward system and combined system show strictly positive eigenvalues across all frequencies

The original system was shown to be very close to passive with only a small frequency band producing negative eigenvalues. The addition of the feedforward system does push the system passive, however the eigenvalue change at low frequencies is not a negligible change. These results show a passivation that was successful, but seemingly non-optimal. The optimiser identifies the objective C_{tot} as unbounded and therefore infeasible. Changing the objective to be more directly in line with the goals $|C_{tot} - C|$ is used to ensure the objective is not unbounded (from the absolute function) and that there is minimal change to the output of the

system. This change does allow for an optimal solution to be produced with a feedforward output matrix of $C_{ff} = [0 \quad -3.4659 \quad 1.0708]$ but the eigenvalues produced are even larger.

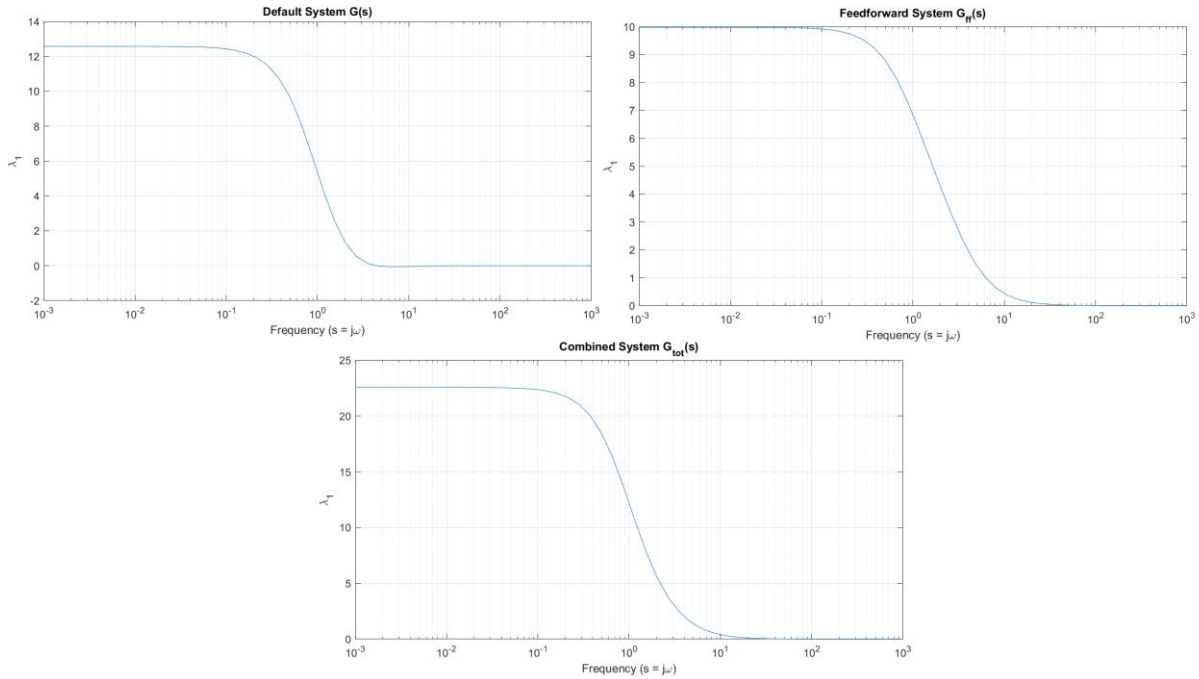

Figure 3.7. System passivity represented by frequency-dependent eigenvalues for the objective of $|C_{tot} - C|$. Both feedforward and combined systems are passive with all positive eigenvalues, but with higher values than the previous objective approach

From this it can be assumed that the approach of minimising the difference between the non-passive system output and the newly passivated system output does not result in the most minimal change to passivity. Instead it allows the addition of large amounts of passivity so long as the final output is similar to pre-adaptation. It is difficult to optimise with respect to the eigenvalues directly, so a new approach of utilising the upper bound of output matrix eigenvalues, as seen in (3.27), was attempted, hereby known as the "gamma method". The Schur inequality was added to the list of the constraints while the value for γ was set as the objective to minimise. As the goal is to find the feedforward system that changes the eigenvalues a minimal amount while still resulting in a passive system, the C matrix that should be substituted into the Schur constraint is C_{ff}. If the goal was to determine the minimal eigenvalues for C_{tot} across all frequencies, then the C matrices should merely be swapped within this constraint. Results of the minimised C_{ff} approach are shown in Figure 3.8 to be successful in producing a small feedforward system that minimises the divergence between the default and combined systems. A small disclaimer must be made in that the eigenvalues are not all strictly

positive, and for higher frequencies the calculated values are negative with insignificant magnitudes of -1.3x10^{-10}. It is believed that this is the result of rounding issues within the MATLAB eigenvalue calculator and can therefore be ignored.

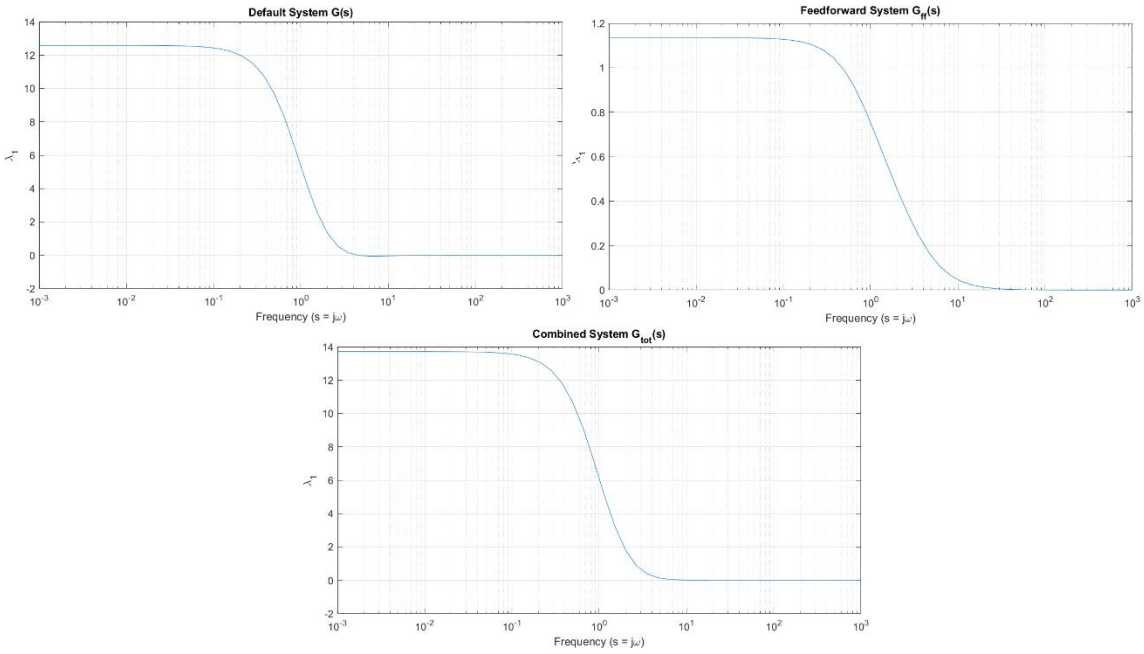

Figure 3.8. System passivity represented by frequency-dependent eigenvalues for the objective of γ. The feedforward system is shown to have much smaller values at all frequencies to result in the combined system being as close as possible to the default system while also being (approximately) passive

The resulting feedforward output matrix is calculated to be $C_{ff} = [0.1081 \quad -0.3243 \quad 0.1351]$ which is also quite minimal in its effects on the system output. Overall these results for the SISO system are promising and show a valid approach to passivating stable systems. This test was also repeated on a MIMO system to confirm consistent behaviour. The state space representation and final eigenvalue plots are shown below to once again be successful in passivation by adding a minimal feedforward system.

$$A = \begin{bmatrix} -4 & 0 & 0 \\ 0 & -3 & 0 \\ 0 & 0 & -1 \end{bmatrix}; \quad B = \begin{bmatrix} 1 & -2 \\ 3 & -4 \\ 5 & -6 \end{bmatrix}; \quad C = \begin{bmatrix} -1.5 & -3 & 0.5 \\ 4 & -1 & -1.5 \end{bmatrix}; \quad D = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix} \tag{3.32}$$

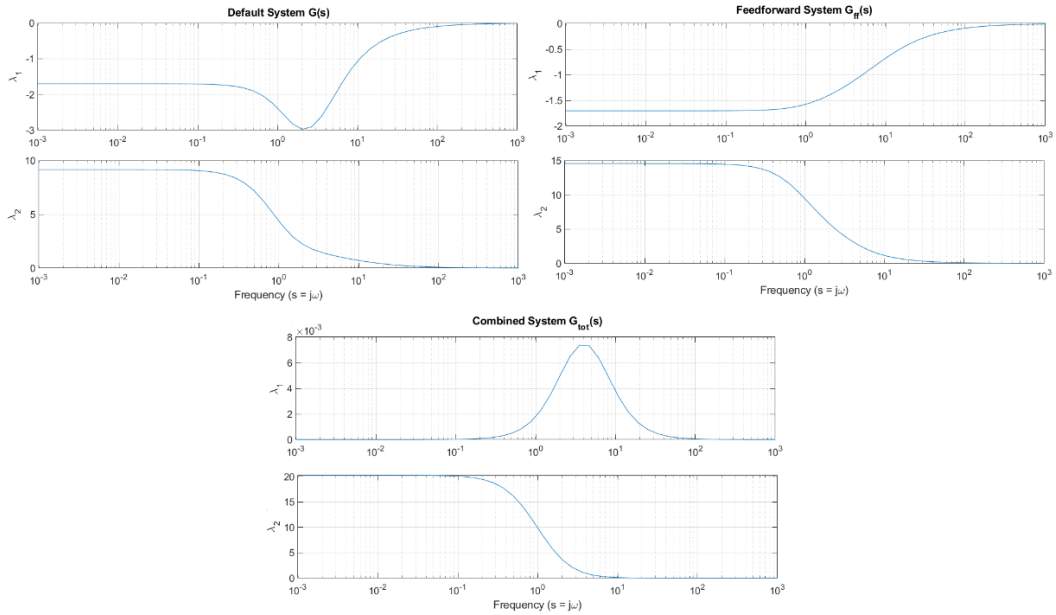

Figure 3.9. Frequency-dependent eigenvalues of $(H+H^T)/2$ for each control system G(s), $G_{ff}(s)$, and $G_{tot}(s)$ to visualise system passivity. The combined system is shown to have strictly positive eigenvalues across all frequencies and as such is a passive system. Neither the default system or the feedforward system are passive systems on their own

As the system has two inputs and two outputs the transfer function is represented by a 2x2 matrix, hence each system now producing two eigenvalues. Another side effect of the MIMO system is the added complexity of combining systems, as it can be seen that directly adding the eigenvalues from the default and feedforward system will no longer result in the combined system eigenvalues. In this case the combined system has a smaller λ_1 than its two components which suggests crosstalk between the inputs and outputs.

B) Non-equal State Spaces

When state matrix and input matrix A and B are not equal, then the simplification of the overall transfer function in (3.15) becomes much more complicated. By calculating the controllable canonical form of the system allows the creation of a new system that can be concatenated with the previous system. Although they have the same number of states it is still a valid approach to system combination. The system tested is the same as shown in (3.31), with the controllable canonical form equal to the system seen in (3.33).

$$A_{ccf} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ -12 & -19 & -8 \end{bmatrix}; \quad B_{ccf} = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}; \quad C_{ccf} = [151 \quad 69 \quad 8]; \quad D_{ccf} = 0 \tag{3.33}$$

By using the controllable canonical form as the feedforward system, but with the output matrix replaced by the optimiser, the eigenvalues produced by the system have minimal difference after passivation, seen in Figure 3.10. The passivation method was the same as the original method, using the output matrix as the objective to minimise, on the system below:

$$A_{tot} = \begin{bmatrix} A & 0 \\ 0 & A_{ccf} \end{bmatrix}; \quad B_{tot} = \begin{bmatrix} B \\ B_{ccf} \end{bmatrix}; \quad C_{tot} = [C \quad C_{ff}]; \quad D_{tot} = 0 \tag{3.34}$$

The previous method did optimise toward $|C_{tot} - C|$ as to minimise the produced feedforward matrix C_{ff}, however in this case this is not possible as C_{tot} and C share a matrix structure. Instead, the optimiser aims to minimise $|C_{tot}|$, as C_{ff} is the only variable within the objective and should therefore be the only component to be affected.

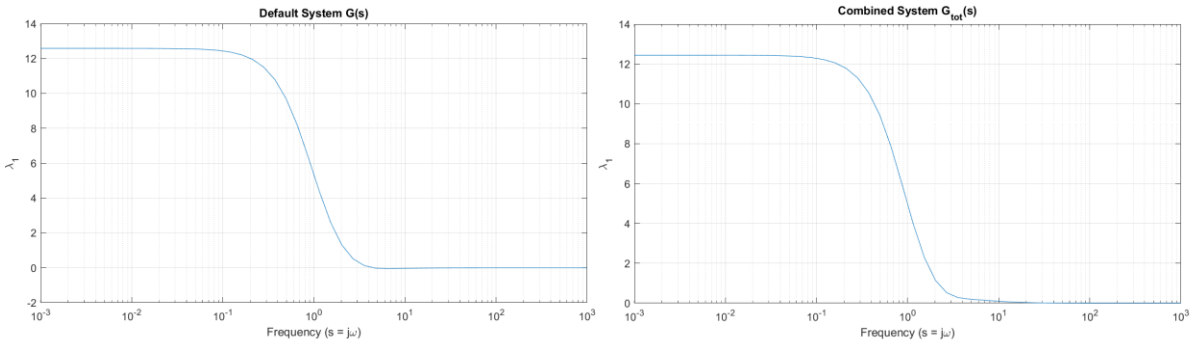

Figure 3.10. Frequency-dependent eigenvalues for original control system G(s) and the concatenated control system $G_{tot}(s)$ to visualise system passivity. Both systems are shown to be nearly identical, with the small dip to negative values around 3 Hz removed for the combined system, hence showing passivity

Several alternative permutations of the block structures in (3.34) were attempted, with the controllable canonical form tested in the first position and the default form in the second, and with both positions filled by the same form. Each of these attempts resulted in unique optimised C_{ff} values and eigenvalue plots confirming passivity. Many had large increases to the eigenvalues and were therefore discarded as non-desired. However the attempt with both blocks being filled by the controllable canonical form resulted in fairly reasonable eigenvalues with the maximum peak at the same frequency the non-passivity occurred in the original system. This suggests the optimiser was trying to add passivity only where necessary.

For the application of the gamma method the previous constraints and objective was used, with the only difference now being the state space representation and the state dimensions being increased. The optimiser worked as intended and suppressed the values of the feedforward

matrix to $C_{ff} = [-0.0003 \quad -0.0763 \quad 0.6155]$ with minimal changes to the eigenvalues while also achieving passivity. Different block permutations were again tested, but none had any noticeable improvements to the default-controllable combination of (3.34). The MIMO system was also tested and confirmed to successfully passivate the system, although this was done without a controllable canonical form model as it was too difficult to figure out for the minimal information it offered.

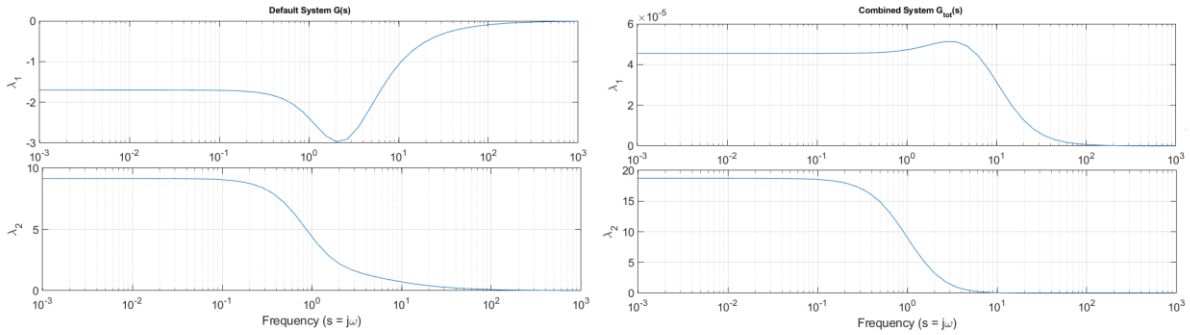

Figure 3.11. Frequency-dependent eigenvalues of (H+HT)/2 for original control system G(s) and the concatenated control system $G_{tot}(s)$ to visualise system passivity. The default system is shown to have one negative eigenvalue which is completely compensated in the combined system.

C) Summary

Both methods of state space structure were shown to be able to apply the KYP lemma and guarantee passivity of the final combined system. The primary difference between methods is the resulting number of states which may cause computational problems for systems with large numbers of states in both systems that cannot be reduced. When creating a Bode plot for both systems (3.31) (replacing C with $C + C_{ff}$) and (3.34) it becomes clear there is very little difference between the systems, except where the combined states have a slightly higher gain for all frequencies. It is currently unclear what causes this difference, however it is clear that the block method leads to the least amount of change in the Bode response. Despite this, the combined method is much easier to implement and will scale better for larger systems. The designer must select their preferred method depending on their system.

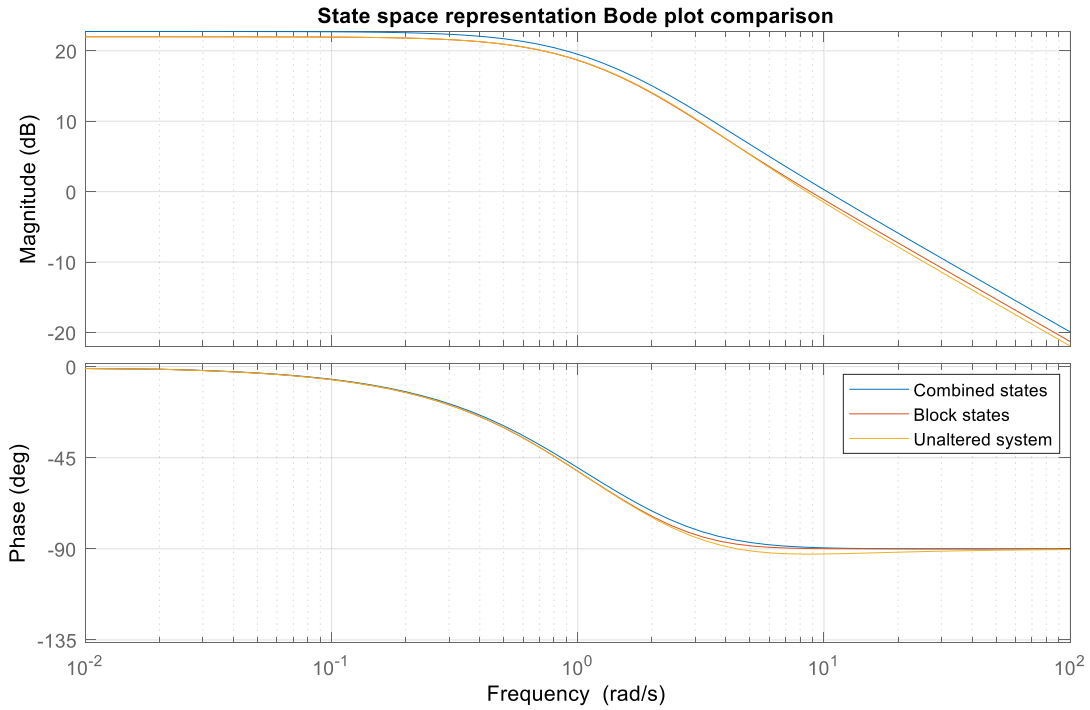

Figure 3.12. Bode plot comparing state matrix equivalence to block diagonal state matrix

Overall, this experiment confirmed that a stable system can be passivated using the optimisation techniques discussed earlier. When combined with the first experimental results, it is now clear that a system can be stabilised with full state feedback, and then passivated using feedforward system addition.

3.4.3. Stage 3 – Diagonalisation

Due to the nature of this section there are very little results to present, with all goals simply being to create a matrix with an appropriate diagonal or block-diagonal form. Nevertheless, it was confirmed that this form could be readily achieved by utilising the equations (3.10)-(3.12) at the cost of increasing the state vector dimensions. To provide a better understanding of the diagonalisation method, an example from the textbook "Linear Systems" by Kailath (1980) is presented below with the transfer function (3.35) taken from their example 6.2.1.

$$G = \begin{bmatrix} \frac{s}{(s+1)^2(s+2)^2} & \frac{s}{(s+2)^2} \\ \frac{-s}{(s+2)^2} & \frac{-s}{(s+2)^2} \end{bmatrix} \tag{3.35}$$

Each transfer function has more than one matrix fraction description and does not need to be presented in a diagonal form. One simple MFD for this transfer function is:

$$N = \begin{bmatrix} s & 0 \\ -s & s^2 \end{bmatrix}; \quad D = \begin{bmatrix} 0 & -(s+1)^2(s+2) \\ (s+2)^2 & (s+2) \end{bmatrix}$$

If this form of MFD was used to determine the state space matrices A_c, B_c, and C_c from (3.10)-(3.12) the produced matrix of $(sI - A_c)^{-1}B_c$ will not be block diagonal, as seen in the LHS of Figure 3.13. This can be changed by modifying the MFD to contain a diagonal denominator matrix, as follows:

$$N = \begin{bmatrix} s & s \\ -s(s+1)^2 & -s \end{bmatrix}; \quad D = \begin{bmatrix} (s+1)^2(s+2)^2 & 0 \\ 0 & (s+2)^2 \end{bmatrix}$$

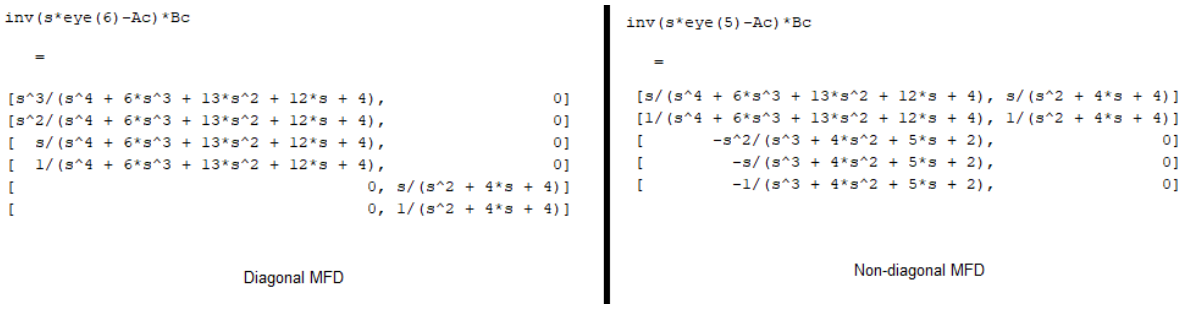

Figure 3.13. Comparison of transfer function component (sI-A_c)B_c for different matrix fraction description forms. The diagonal MFD has a degree of 6 and is block-diagonal, while the non-diagonal MFD has a degree of 5 and is not block-diagonal*

From this state it becomes clear that the final component of the transfer function, C_c, will cause the transfer function $G(s) = C(sI - A)^{-1}B$ to be diagonal when it is block diagonal itself. Additionally, it can be seen that each element of the matrix per column shares a denominator term and only differs in the numerator which is simply the complex frequency variable s raised to sequentially different powers. Multiplying this matrix by a free matrix on the left hand side allows the complete freedom to construct any strictly proper transfer function in the appropriate diagonal elements of the resulting matrix. From this knowledge, using the optimiser to determine the best values to include while still remaining block diagonal allows the guarantee that the new transfer function for the feedforward system will be diagonal and avoid input-output mixing. Using the feedforward system state matrices in the KYP lemma returns a functioning result to guarantee the additional system is passive. Using the combined system of $C_c + C_f$ returns an “unknown error” message, with the eigenvalue plot showing non-passive traits of the combined system. Utilising the gamma method for optimisation replaced the “unknown error” message with an “infeasible error” message, which seems to be expressing that passivating the system with a diagonal transfer function is not mathematically possible. A

workaround to this is to optimise for a total output matrix with the objective being to minimize the absolute values in the off-block-diagonal positions to get as close to diagonal as possible. As the optimiser is limited to only one objective this prevents the use of the gamma method, but does return a feasible result. This approach results in the following output matrix C_f, with non-zero off-diagonal values.

$$C_f = \begin{bmatrix} 1.1525 & 1.3516 & 0.1424 & 0.4154 & 0.0000 & 0.5109 \\ 2.0000 & 1.3054 & 0.1814 & -0.4512 & 3.7106 & 1.1189 \end{bmatrix}$$

It seems to be the case that there is no guarantee that a system can be passivated exclusively through the inclusion of a diagonal controller. This outcome seems misinformed, as the sum of a small matrix and a dominating diagonal matrix will approximate the diagonal matrix, which can itself be designed to be passive. This can also be seen through the effects of the eigenvalues of the combined matrix; by setting all diagonal elements to be largely positive so the sum of all columns is also positive at all frequencies, the eigenvalues are guaranteed to be positive at all frequencies which confirm passivity. The relatively small off-diagonal elements will only contribute a small amount to the eigenvalues, as seen through the Gershgorin circle theorem.

As further analysis, a simple MIMO transfer function was calculated by hand to compare to the MATLAB code for troubleshooting purposes. Subsequently, a manually designed diagonal transfer function was added to the system to prove passivation was possible. The transfer function in question is provided below, with the control canonical form state space:

$$G(s) = \begin{bmatrix} \frac{1}{10s+1} & \frac{0.6}{10s+1} \\ -\frac{0.6}{s+1} & \frac{1}{10s+1} \end{bmatrix} \quad (3.36)$$

$$A = \begin{bmatrix} -1.1 & -0.1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & -0.1 \end{bmatrix}; \quad B = \begin{bmatrix} 0.1 & 0 \\ 0 & 0 \\ 0 & 0.1 \end{bmatrix}; \quad C = \begin{bmatrix} 1 & 1 & 0.6 \\ -6 & -0.6 & 1 \end{bmatrix}$$

By replacing the output matrix C with a newly designed matrix C_{ff}, a new transfer function can be constructed equal to:

$$G_{ff}(s) = C_{ff}(A - sI)^{-1}B = \begin{bmatrix} c_1 & c_2 & 0 \\ 0 & 0 & c_3 \end{bmatrix} \begin{bmatrix} \frac{s}{(10s+1)(s+1)} & 0 \\ \frac{1}{(10s+1)(s+1)} & 0 \\ 0 & \frac{1}{10s+1} \end{bmatrix}$$

$$= \begin{bmatrix} \frac{c_1 s + c_2}{(10s+1)(s+1)} & 0 \\ 0 & \frac{c_3}{10s+1} \end{bmatrix}$$

This transfer function, when added to the original transfer function G(s) in a feedforward structure, can be set to dominate. The total transfer function $G_t = G + G_{ff}$ was used to calculate the eigenvalues of $G_t(j\omega) + G_t^H(j\omega)$ and found all eigenvalues could indeed be forced to be positive, coinciding with a forced passivation of the system. In the case where the added transfer function was set to $\frac{K}{10s+1}$ on both diagonal elements of G_{ff} then the eigenvalues were mostly positive, with some small negative values with magnitudes less than 0.017 (for $K=10^2$) and $1.7\text{x}10^{-7}$ (for $K=10^6$) for higher frequencies. Altering the value of K would shift the frequency at which these negative values first occur, but never completely removed them. It is currently unclear whether this very small passivity violation is caused by rounding issues within MATLAB or are an indication that passivity is not always possible through diagonal addition. Further testing is required to identify limitations and functioning cases, but a potential explanation is expressed in section 3.5.2 with the short explanation being that the imaginary component in the off-diagonal elements of $G_t + G_t^H$ causes issues and should be removed by setting the imaginary off-diagonal components of G_t to be symmetric (while the real component is not restricted).

Changing the transfer function to $\frac{Ks}{10s+1}$ does guarantee positive eigenvalues and hence passivity, however this is not achievable by setting the C_{ff} matrix values. Setting the transfer function to $\frac{Ks}{(10s+1)(s+1)}$, which is possible, shows a clear and unambiguous dip in the eigenvalues of G_t around 0.4 Hz. These results suggest the small negative values produced in the other transfer function cases are not rounding error as they follow a similar path. As stated before, this seems to contradict the Gershgorin circle theorem and the interpretation of these results remains unclear, but discussed in later discussion. It is still believed that passivation is possible with the diagonal addition, but only under stricter conditions that may reduce the applicability of this pairing method.

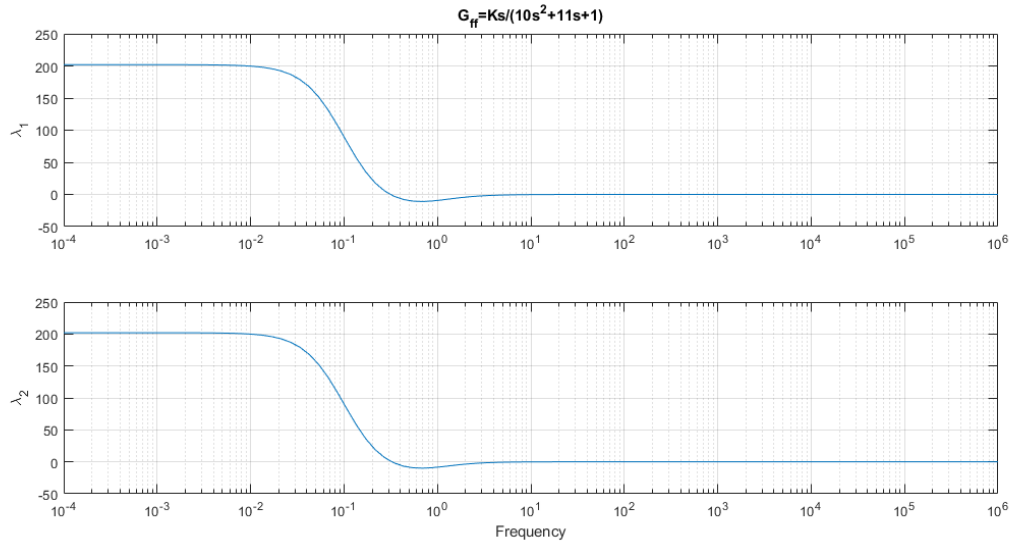

Figure 3.14. Eigenvalues of the system $G_t + G_t^T$ where $G_t = G + G_{ff}$ and $G_{ff} = 100s/(10s^2 + 11s + 1)$. The eigenvalues showing a clear zero crossing to negative values equate to a non-passive system

3.4.4. Stage 4 – Pairing Selection

Due to the nature of control configuration selection it is difficult to identify the "correct" solution. This new proposed method, hereby referred to as the "Optimiser method", aims to recommend the CCS that reduces crosstalk between inputs and outputs while also minimizing the divergence from the physical system caused by the controller. This goal is not suitable for every scenario and the intention of the system must be considered before choosing which CCS method is best suited. The results of different pairing methods can be collected and the final pairing recommendations can be compared to one another as a feasibility check. Identifying and explaining discrepancies between the proposed and existing methods may help to solidify the scenarios in which the optimiser method is best suited. The existing methods tested in this paper include the relative gain array (RGA) for its commonplace usage and simplicity, the dynamic relative gain array (DRGA) for its ability to consider the frequency dynamics of the system over the RGA, and the passivity-based method proposed by Bao et al. (2007) for the benefits of comparing with another passivity-focused method. The pairing results for each of these methods are shown in Table 3.1 for several examples found throughout the literature.

3.4.4.1.Control Configuration Selection Methods & Recommendations

The RGA method is the simplest of the CCS methods, however it comes with noticeable drawbacks. The elements of an RGA will give a scale-invariant measure of the dependence of output j on input i, effectively describing the relative gain between inputs and outputs. RGA helps analyse fundamental steady state closed loop properties such as stability and robustness but does not consider the dynamics. As all real-world transfer functions will be dependent on the Laplace variable, looking only at the case for $s = 0$ will not show important properties of the system. A system must also be non-singular for the RGA to be calculable, so some systems cannot utilise this method.

The DRGA method fixes these issues by simply calculating the equivalent to the RGA over a specified frequency range to show that the relative gain of one pairing may increase or decrease depending on frequency. If the system is singular at steady state, varying the frequency value s has the potential to produce a non-singular matrix to allow the DRGA to be calculated.

Using passivity takes the idea of frequency dependence a step further, as one of the definitions of a passive system is that the eigenvalues of the system are all positive for all frequencies. Therefore, all frequencies must be inherently considered for a system to be defined as passive, and the system dynamics will be represented through these results. This allows a direct IO pairing to be recommended as opposed to a pairing that fluctuates based on frequency ranges. The passivity-based method proposed by Bao aims to produce pairing schemes that are decentralised integral controllable (DIC) and passive on a frequency band as large as possible. Decentralised integral controllability is simply the determination of whether a system is controllable with a decentralised integral controller, viewed through the metric of closed-loop performance, and thus can be used in determining which IO pairings are practical. If the system is DIC, then the addition of a diagonal controller to the MIMO system becomes much easier and it is possible to achieve stable and offset-free control of the overall closed-loop system by tuning every loop separately (Bao and Lee 2007: 92). If the system is passive for a larger bandwidth, then the steady state gain that can be applied to the system within that band can be infinite without negatively affecting stability. These two goals of DIC and steady state passivity result in a system with good performance that can be effectively controlled for many input signals. By testing every possible pairing scheme for two specific conditions, any pairing that will not result in these goals being met can be immediately disregarded, leaving only the viable pairings to be compared. These two conditions are as follows:

1) The system must be passive at zero frequency, to guarantee the frequency band begins at 0. If the scaled passivity index at zero frequency is greater than zero, then the system is not DIC and the pairing is rejected
2) The controller of the system must have output constraints (such as physical or economic limitations) that do not limit the control of the system or require large controller gain. If plant elements at zero frequency are larger than the output constraints, the process outputs will not be controllable through input manipulation and the pairing is rejected

Any pairings that meet both these conditions can then calculate the passivity index and plot against frequency to identify the pairing that results in the largest frequency bandwidth of a zero passivity index. This result simply states that no additional feedforward or feedback systems need to be added to the system to achieve passivity.

Table 3.1 - Input-Output Pairing Recommendations from Different Methods

Example	RGA	DRGA	Passivity-based Method	Optimiser Method
Kailath 6.4.1 [2x2] [no delay]	All equal	$(u_1\text{-}y_2)$, $(u_2\text{-}y_1)$ for all frequencies	N/A	$(u_1\text{-}y_2)$, $(u_2\text{-}y_1)$
Henrion 3.4.1 [2x2] [no delay]	All equal	$(u_1\text{-}y_1)$, $(u_2\text{-}y_2)$ for all frequencies	N/A	$(u_1\text{-}y_2)$, $(u_2\text{-}y_1)$
McAvoy 1.2 [2x2] [time delay]	$(u_1\text{-}y_1)$, $(u_2\text{-}y_2)$	$(u_1\text{-}y_2)$, $(u_2\text{-}y_1)$ for frequencies > 0.03	$(u_1\text{-}y_1)$, $(u_2\text{-}y_2)$	$(u_1\text{-}y_2)$, $(u_2\text{-}y_1)$
Bao 6.1 [3x3] [time delay]	$(u_1\text{-}y_1)$, $(u_2\text{-}y_3)$, $(u_3\text{-}y_2)$	$(u_1\text{-}y_1)$, $(u_2\text{-}y_3)$, $(u_3\text{-}y_2)$*	$(u_1\text{-}y_1)$, $(u_2\text{-}y_3)$, $(u_3\text{-}y_2)$	N/A
Bao 6.3 [2x2] [time delay]	$(u_1\text{-}y_1)$, $(u_2\text{-}y_2)$	$(u_1\text{-}y_2)$, $(u_2\text{-}y_1)$ for frequencies > 0.09	$(u_1\text{-}y_2)$, $(u_2\text{-}y_1)$	$(u_1\text{-}y_1)$, $(u_2\text{-}y_2)$

* Pairing recommended for all frequencies except for a small frequency band around 1 Hz

Each CCS method was programmed in MATLAB 2021a. If the pairing method produced equal scoring for different pairings, then no one score was better than the others and “All equal” is displayed to convey that any pairing can be chosen without negative consequence (according to that method). Some methods are incompatible with a specific example due to either limitations in the technique or technical issues. For these scenarios “N/A” is substituted as the pairing recommendation to represent the lack of meaningful results. For methods that do result in a

recommended pairing, the j^{th} input paired with the k^{th} output will be represented by $(u_j - y_k)$. The DRGA results were observed over the arbitrary frequency range of 10^{-4} Hz to 10^4 Hz to contain any important features of the relative gains. The details of each example are not mentioned within the table, which simply gives a general pairing recommendation. The passivity-based method was replicated to the best ability, however when comparing examples there were discrepancies between the results produced and the results presented in the original paper. These discrepancies were seen in the frequency bandwidth that allows an infinite steady state gain, but did not seem to affect which pairing scheme is recommended as each pairing was equally affected. The passivity-based method shows which frequencies can be operated at for good control, while our proposed optimiser method shows which system must be adjusted the least amount (and how much that amount is) to be well controlled at every frequency. Further developments to the method could allow specified frequency bands to be defined to allow more conservative deviations from the original system.

3.4.4.2. Examples

Case 1 – Kailath (1980: 407)

The first example tested was sourced from the textbook "Linear Systems" under the chapter "State space realisations and matrix fraction descriptions of multivariable systems". The transfer function is presented in (3.35).

$$G = \begin{bmatrix} \frac{s}{(s+1)^2(s+2)^2} & \frac{s}{(s+2)^2} \\ \frac{-s}{(s+2)^2} & \frac{-s}{(s+2)^2} \end{bmatrix}$$

This example was selected for its polynomial properties and structuring the design of the optimiser method. As such, its practical applications for control are limited and do not represent any commonly seen transfer functions for control systems but still provides some insight into variation of CCS methods. The case provided by Kailath does not have a reference tracking property commonly seen from low-pass filters; instead the system output tends towards 0 regardless of the input.

The transfer function matrix is singular at frequency $s = 0$, so RGA analysis is incompatible with the example due to the dependence on the matrix inverse. Generalisations exist for singular matrices, such as the using the alternative generalised matrix inverse proposed by Uhlmann (2019). By implementing this technique the unit-consistent RGA shows an equal effect from one

input to both outputs, while the other input has no effect on either output. This example showcases the restrictive applicability of RGA and its inability to handle zeros at steady state frequency. As the singularity only exists for zero frequency it is possible to view the relative gains at small frequencies to analyse the system. These results appear in the DRGA method, which will present the limits of the DRGA as frequency approaches 0. From these results the pairing recommendation for RGA will equal the pairing recommendation from the DRGA, which in this case is a cross-diagonal pairing (input 1 to output 2 and input 2 to output 1).

The passivity-based method also fails to produce meaningful results for the Kailath example. For the two pairings possible for the 2x2 example both pairings are rejected; the off-diagonal pairing is rejected due to not being DIC while the diagonal pairing is rejected due to the controller output restrictions. With no recommendation of pairing is possible the method fails to provide any insight for control. The controller output restrictions are based around arbitrarily defined upper limits, so adjusting these values until the conditions have been satisfied show that the diagonal pairing does result in a larger frequency bandwidth of passivity (since the off-diagonal pairing has no passivity bandwidth) and this pairing could be recommended under specific conditions. This pairing recommendation is not recorded in Table 3.1 however, since the upper limit constraints had to be set to greater than 10 million, which is considered an unreasonable upper bound. This result does produce a point of interest in that the passivity-based method recommends a different pairing scheme to both DRGA and the proposed method.

As a 2x2 system the only possible control configuration selections are the diagonal pairing $(u_1 - y_1)$ and $(u_2 - y_2)$ or the off-diagonal pairing $(u_1 - y_2)$ and $(u_2 - y_1)$. DRGA recommends the off-diagonal pairing, to which our proposed pairing method agrees. Comparing the additional feedforward systems necessary for passivation show the diagonal pairing resulted a matrix with a spectral norm of 3.65×10^3 compared to 2.95×10^3 from the off-diagonal pair. Under the hypothesis that spectral norm represents both diversion from the natural system and the energy necessary to achieve this shift, the off-diagonal pairing is shown to be the more conservative CCS. The resulting paring recommendation matches the DRGA recommendation. However since there are only two possible pairing schemes and only two CCS methods were applicable to this example, the efficacy of the proposed method is still unsupported.

Case 2 – Henrion and Šebek (2009)

The second example was taken from a research report chapter detailing matrix fraction descriptions for polynomial matrices:

$$G = \begin{bmatrix} \frac{s}{(s+1)^2} & \frac{-s}{(s+1)^2(s+2)^2} \\ \frac{s}{(s+1)^2} & \frac{s(s^2+s-1)}{(s+1)^2(s+2)^2} \end{bmatrix} \quad (3.37)$$

This example is very similar to the previous case, however the transfer function for the second input and output has multiple zeros not located at $s = 0$. With one of the zeros located on the right-hand plane the system will become unstable at higher gains, restricting the potential controllers that can be used to control the MIMO system as a whole. The right-hand plane zero also guarantees the system is not minimum phase, which will affect the state size of the state space representation used within the optimiser CCS method.

The transfer function is once again singular at zero frequency which prevents the classical RGA calculation. The DRGA method shows a perfect diagonal pairing, with input 1 affecting output 1 completely with no crosstalk from input 2, at frequencies greater than 5 Hz. This is represented through a relative gain of 0 or 1 between unlinked or linked inputs and outputs, respectively. For lower frequencies the relative gain values transition from 1 to 2 for linked connections (in this case the diagonal pairing) and from 0 to -1 for the unlinked. This suggests the RGA would agree with the DRGA pairing recommendation, but would be doubly sensitive to input changes affecting the output at lower frequencies.

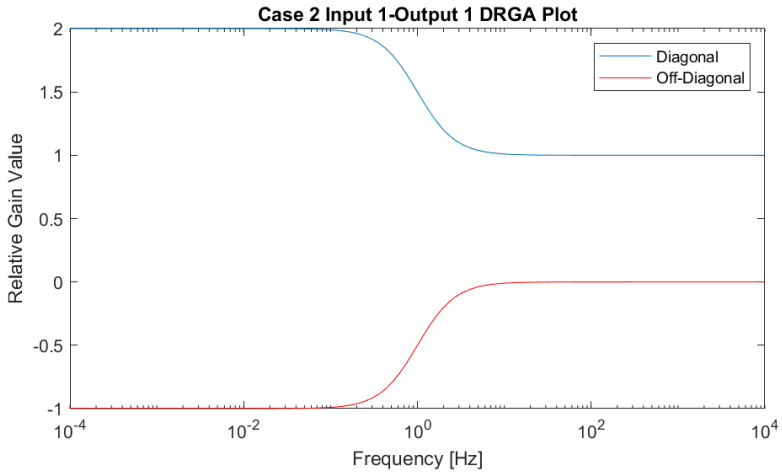

Figure 3.15. Dynamic Relative Gain Array for system described by (3.37) for input 1-output 1 pairing (blue) and input 1-output 2 pairing (red)

The passivity method produces similar issues to case 1 where both pairing options are rejected before it is possible to compare results. As with example 1, the diagonal pairing is rejected due to an arbitrary controller output limitation while the off-diagonal is rejected for not being DIC

and not being passive at zero frequency. By adjusting the controller output upper limits to 5000 the diagonal pairing can be made viable. This limit is more reasonable than example 1, but without context for what this value physically represents it is impossible to determine if the example will be effectively controllable through linear multi-loop control. The passivity method has certain limitations as to where it can be applied but has been proven as effective for systems that do conform to its restrictions. In the case of not rejecting this result, both DRGA and the passivity method suggest a different pairing from the optimiser method. With all established CCS methods recommending diagonal pairing, our method recommends an off-diagonal pairing for the drastically lower spectral norm of 1.80 compared to the diagonal pairing spectral norm of 2.26×10^3. The optimisation toolbox confirms this pairing is feasible with the optimal solution given by a matrix with the aforementioned spectral norm.

Conclusions are difficult to derive from both examples 1 and 2 as neither transfer function is traditionally used in control, but both were discussed due to their importance in creating the proposed method and its script, as well as establish cases where the proposed method both agrees and disagrees with other CCS methods, showing a potential benefit to analysing the passivating feedback system.

Case 3 - Mc Avoy et al. (2003)

The third example tested was referenced in several papers, which seemingly originated in a paper discussing new DRGA techniques:

$$G = \begin{bmatrix} \frac{5e^{-40s}}{100s+1} & \frac{e^{-4s}}{10s+1} \\ \frac{-5e^{-4s}}{10s+1} & \frac{5e^{-40s}}{100s+1} \end{bmatrix} \quad (3.38)$$

Each transfer function has the general structure of a low-pass filter (first order proper transfer function component) with time delay (exponential component) and as such is well suited for reference tracking systems. For the proposed optimiser method each transfer function must be represented as a polynomial in both numerator and denominator to structure the MFD correctly. The exponential component representing time delay must therefore be approximated into polynomial form using the Padé approximation. A first order Padé approximant was used to estimate the time delay at $s = 0$. Bao's passivity method plots the scaled passivity index on the frequency band [0, π] which are all close enough to $s = 0$ that the Padé approximate is applicable to the test. To reduce system differences between CCS methods, this Padé

approximation was used in every method for calculations. The non-approximated examples were also tested and confirmed similar results to their approximations to ensure this approach was justified.

The RGA results recommend a diagonal pairing, which matches the DRGA results for frequencies less than 10^{-2} Hz. At approximately 0.02 Hz the relative gain between inputs changes rapidly and settles for frequencies greater than 0.1 Hz to recommend the off-diagonal pairing. This shows the dynamics of the system do play an important role in the pairing selection and as such why the RGA is not always reliable. However since the DRGA only gives recommendations for specific frequencies, using one of the passivity-based methods provides a pairing scheme that should consider all frequencies.

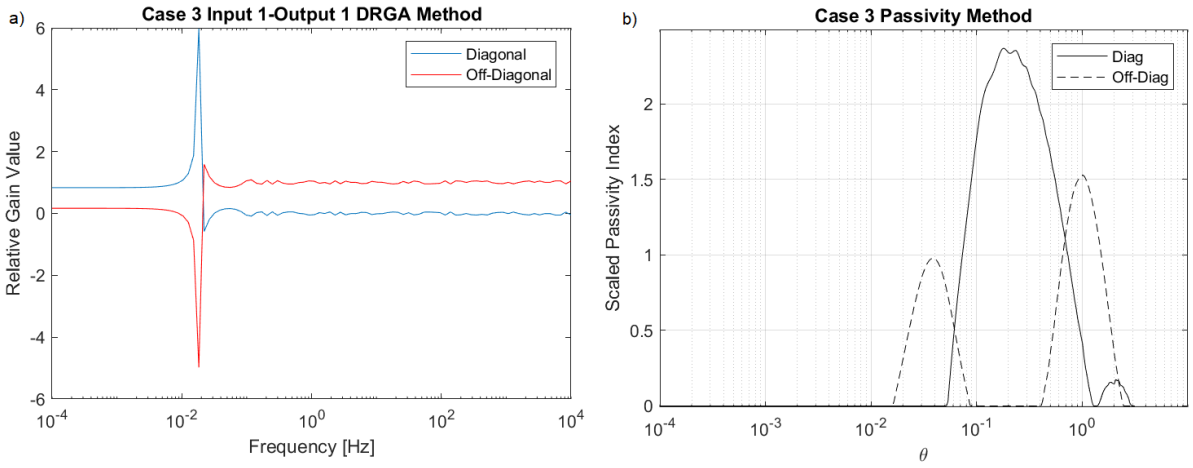

Figure 3.16. a) Dynamic Relative Gain Array for system described by (3.38). b) System passivity index based on frequency shows the diagonal pairing has a larger frequency bandwidth where the system requires no excess passivity and as such has the faster response and larger range of operation for offset-free control

The passivity method presents the diagonal pairing as the superior scheme, and as such disagrees with the DRGA results. This conclusion is based off the scaled passivity index plot seen in Figure 3.16 by the diagonal results remaining at zero for a larger frequency band. Since established CCS methods use different metrics to determine a recommendation it is not uncommon for different pairing schemes to be produced. This also shows that newly developed methods may offer additional perspectives that can benefit the control design process. The optimiser method shows that the off-diagonal pairing requires an additional feedback system with a spectral norm of 1.80×10^{4} while the diagonal pairing has no possible feedback matrix that can satisfy all the constraints of the optimiser and as such the solver returns an infeasible solution. As a result, only the off-diagonal pairing results in a system that is guaranteed passive and decentralised.

Case 4 – Bao et al. (2007)

This example represents a 3x3 distillation column and was referenced by Bao as a point of comparison between their passivity-based method and the generalised dynamic relative gain method:

$$G = \begin{bmatrix} -\frac{1.986e^{-0.71s}}{66.67s+1} & \frac{5.24e^{-60s}}{400s+1} & \frac{5.984e^{-2.24s}}{14.29s+1} \\ \frac{0.0204e^{-4.199s}}{5s+1} & -\frac{0.33e^{-1.883s}}{3.904s+1} & \frac{2.38e^{-1.143s}}{10s+1} \\ \frac{0.374e^{-7.75s}}{22.22s+1} & -\frac{11.3e^{-14.78s}}{35.66s+1} & -\frac{9.881e^{-1.59s}}{11.35s+1} \end{bmatrix} \quad (3.39)$$

The Padé approximation was used for this example as well, which resulted in each transfer function being strictly proper with the numerator containing a degree of 1 while the denominator is of degree 2, represented below:

$$G = \begin{bmatrix} \frac{1.986s-5.594}{(66.67s^2+188.8s+2.817)} & \frac{-5.24s+0.1747}{(400s^2+14.33s+0.3333)} & \frac{-5.984s+5.343}{14.29s^2+13.76s+0.8929} \\ \frac{-0.0204s+0.009717}{5s^2+3.382s+0.4763} & \frac{0.33s-0.3505}{3.904s^2+5.147s+1.062} & \frac{-2.38s+4.164}{10s^2+18.5s+1.75} \\ \frac{-0.374s+0.09652}{22.22s^2+6.734s+0.2581} & \frac{11.3s-1.529}{35.66s^2+5.825s+0.1353} & \frac{9.881s-12.43}{11.35s^2+15.28s+1.258} \end{bmatrix} \quad (3.40)$$

The number of potential pairing schemes for an n-by-n system is given by n-factorial, and as such the addition of an input/output increases the system complexity dramatically. The RGA and DRGA matrix are constructed in such a way that the sum of each row or column will equal unity. For 2x2 systems this results in each row being the complement to one another and only one row needs to be analysed. However for a 3x3 system this is no longer the case and each row must be observed to identify all relevant information. These results, displayed in Figure 3.17, show input 1 is the predominant factor for output 1 across all frequencies, input 3 is the predominant factor for output 2 across all frequencies excluding the small frequency band around 0 Hz where input 3 is marginally better, and input 2 is the predominant factor for output 3, excluding the same small bandwidth around 0 Hz. This can all be summarised by the recommendation of using the “1-1, 2-3, 3-2” pairing. These results were obtained through the Padé approximation and not the time-delay form of the transfer function. Without using the Padé approximation the DRGA results were terribly noisy and produced graphs that were difficult to parse, although the recommended pairing did not change between forms.

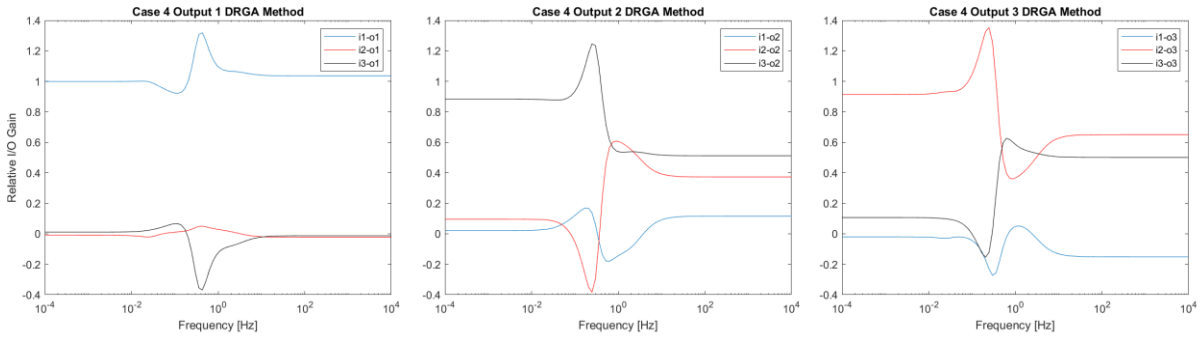

Figure 3.17. Dynamic Relative Gain Array for system described by (3.40). Results show a clear preferred pairing of i1-o1 (LHS), i3-o2 (MID), and i2-o3 (RHS) with a small deviation bandwidth around 1 Hz which swaps the input 2 and 3 pairings

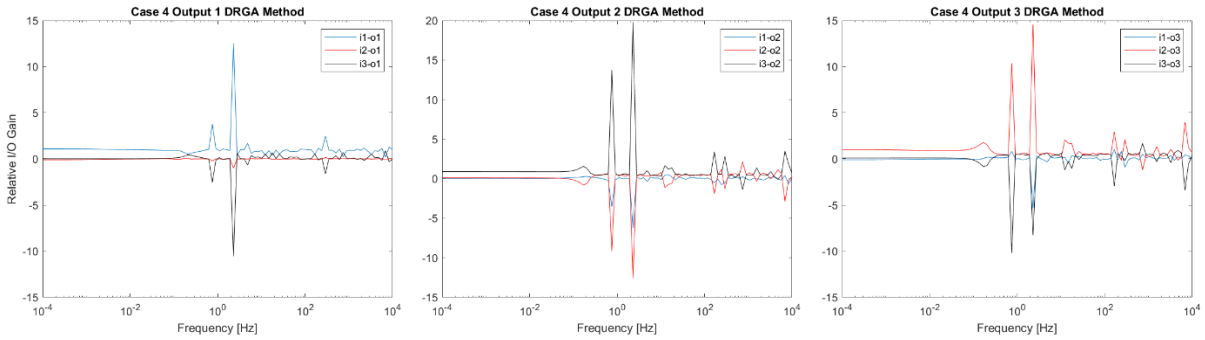

Figure 3.18. Dynamic Relative Gain Array for system described by (3.39). Results show a less clear preference between inputs and outputs without the Padé approximation. Pairing of i1-o1 (LHS), i3-o2 (MID), and i2-o3 (RHS) seem to remain the dominant pairings

The passivity-based method confirms this pairing schematic, where out of the six possible combinations, only the “1-1, 2-3, 3-2” and the “1-3, 2-1, 3-2” pairings satisfy both conditions for the passivity-based method. When compared, the former pairing is shown to have a larger frequency bandwidth than the latter, resulting in the same recommendation to the DRGA method. The results from Bao’s method are presented in their original paper but do not perfectly match the results produced when trying to replicate the method.

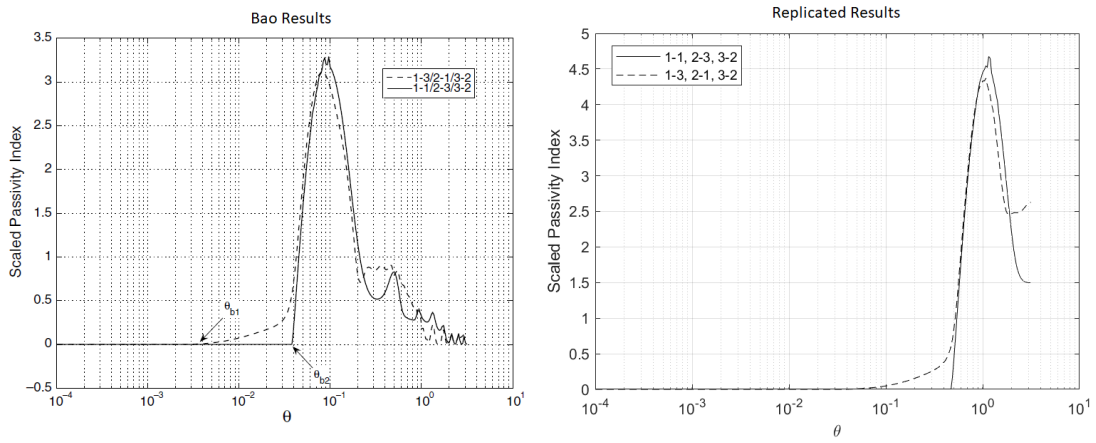

Figure 3.19. Comparison of the example results (left) and its replicated results (right) for the passivity-based pairing method proposed in Bao et al. (2007)

It can be seen that the bandwidth frequency for the preferred pairing occurs at $\theta_{b2}=3.7x10^{-2}$ in the original paper, but $\theta_{b2}=4.6x10^{-1}$ in the replicated results. This suggests the replicated code does not have 100% fidelity to the original paper, although the conclusions do agree for each case tested.

An important note is that the results for the passivity method were produced using the non-approximation transfer function with exponential components. When the Padé approximation was used, the inferior pairing does not pass the conditions and the only remaining pairing of "1-1, 2-3, 3-2" becomes the recommendation by default. In this instance the approximation did not affect the recommendation, but this does show that it can influence the results in non-negligible ways and the results must be observed carefully for any system discrepancies. It also means that while the DRGA method required the approximation to produce clean results, the passivity method required the opposite. This adds an additional layer of variability between methods and makes comparisons slightly less direct. It is unlikely this causes any major issues in calculations, and as such it is considered reasonable, however it is another factor that must be considered during analysis.

The optimiser method employs the symbolic toolbox in MATLAB for solving the optimisation problem, however issues arise due to the high precision used within the transfer function coefficients. Attempts to resolve this through rounding were unsuccessful and the produced results included extremely large coefficients that could not be utilized by the optimisation solver. Each pairing returned the result that the problem was infeasible and not solvable, however this violates the law that a system can be passivated if the process lacks input feedforward passivity or output feedback passivity, and as such the issue is believed to be strictly

caused by limitations in the created script. For this reason no meaningful input/output pairing suggestion from the CCS method were produced from this test. This restriction seems to be caused by software issues and is not emblematic of any underlying mathematical restrictions. Further research can be committed to finding solutions to this problem, but at this time it was not deemed fundamental to the overall project. This example was included to highlight that the proposed method does have limitations and is not suitable to every system, similarly to the other available methods.

Case 5 - Bao et al. (2007)

A second example was sourced from Bao et al. (2007) as a 2x2 transfer function with comparable time delays for each element:

$$G = \begin{bmatrix} \frac{-2e^{-s}}{10s+1} & \frac{1.5e^{-s}}{s+1} \\ \frac{1.5e^{-s}}{s+1} & \frac{2e^{-s}}{10s+1} \end{bmatrix} \quad (3.41)$$

This transfer function is very similar to case 3, however the time delay for each element is smaller and closer to zero, making the Padé approximation more accurate. Similarly to case 3, the RGA and DRGA results show a recommendation for diagonal pairing for low frequencies and off-diagonal pairing for higher frequencies. The frequency of this transition is approximately 0.09 Hz. Our optimiser method agrees with this higher-frequency recommendation as the diagonal pairing leads to an optimal solution from the optimisation solver and a drastically smaller spectral norm of 88.7 compared to 4.35×10^4. In this scenario both pairing schemes were solved with one being superior, as opposed to example 3 which displayed one pairing causing an infeasible problem to form and no solutions possible. The passivity-based method disagrees with this recommendation, instead favouring the off-diagonal pairing. The bandwidth of the diagonal pairing is found to be $\theta_b = 5.7 \times 10^{-2}$ (3×10^{-2} in paper) while the off-diagonal has a bandwidth of $\theta_b = 2.6 \times 10^{-1}$ (1.2×10^{-1} in paper). There is no distinguishable difference between the base transfer function and the Padé approximation, as was expected.

3.4.5. Stage 3/4 Corrections

Several of the previous examples resulted in the optimiser claiming the KYP lemma was infeasible and no solutions existed to satisfy both LMIs (3.7) and (3.8). For the KYP lemma to be applicable, the system must be strictly stable. The cases 1-5 in section 3.4.4.2 were analysed to show that all systems were indeed strictly stable before feedforward alteration and the KYP

lemma should apply as long as G_{ff} does not cause instability (which it should not due to feedforward control not affecting stability). The issue of infeasibility is intensified by the additional Schur constraint (3.27) that is used for constructing an objective to minimize. This suggests that not every system was able to be passivated through feedforward addition alone and the goal of proving every system to be passifiable was not met, and even disproven. Once this was discovered, it was determined that this method was more restrictive than previously thought and the following restrictions were included to guarantee feedforward passivity:

1. For the system G_0, the relative degree of the off-diagonal elements (k_{od}) must be greater than the relative degree of the diagonal elements (k_d): $[k_{od} > k_d]$.
2. For the system G_0, the relative degree of all of the off-diagonal elements must be greater than or equal to 2 $[k_{od} \geq 2]$.

The first restriction guarantees the diagonal elements dominate at higher frequencies and cause the system as a whole to become diagonal dominant. For a frequency value contained within an open-bounded finite range $\omega \in (0, \omega_0)$, the values of G_0 will also be finite and the addition of G_{ff} can always be selected to produce a diagonally dominant $G = G_0 + G_{ff}$ with sufficiently high K values within G_{ff}. This result is then able to use the Gershgorin circle theorem to guarantee eigenvalues are all positive (and hence passive) within the bounded frequency range.

As the limit of ω approaches infinity, since the off-diagonal elements have larger relative degrees they will tend towards zero faster than the diagonal elements. We can then make the generalised claim that, for high ω values of an n-by-n transfer function, the system is diagonally dominant:

$$|g_{ii}| > \sum_{j=1, j\neq i}^{n} |g_{ij}| \qquad | \, i, j = 1, 2, \dots, n$$

$$2|g_{ii}| - \sum_{j=1}^{n} |g_{ij}| > 0$$

For the visualisation of the Gershgorin circle theorem, the diagonal elements g_{ii} act as the centres for each eigenvalue disk and the sum of off-diagonal magnitudes act as the radius for each disk. The inequality directly shows that for each eigenvalue disk, the radius will be less than the distance to the origin and as such the entire disk will be contained on whichever side of the plane the centre lies upon. The gain values within G_{ff} must be set to achieve positive values at low frequencies to guarantee the centre lies on the positive plane.

The second restriction is derived from the state space structure of the feedforward system G_{ff} guaranteeing each diagonal element is a strictly proper transfer function. As a strictly proper transfer function can have a minimum relative degree of 1 and a maximum relative degree equal to the denominator degree, then the feedforward system will also share these relative degree limitations. The summation of two transfer functions will have a relative degree equal to the transfer function with the smaller relative degree. A simple example is provided below:

$$reldeg\left(\frac{1}{s^{\alpha}}+\frac{1}{s^{\beta}}\right)=reldeg\left(\frac{s^{\beta}+s^{\alpha}}{s^{\alpha+\beta}}\right)=(\alpha+\beta)-\max(\alpha,\beta)=\min(\alpha,\beta)$$
$$=\begin{cases}\alpha & | \quad \alpha<\beta \\ \beta & | \quad \beta\leq\alpha\end{cases}$$

To guarantee that system $G=G_0+G_{ff}$ has diagonal elements with relative degrees less than the off-diagonal elements, then simply choose the relative degree of G_{ff} (k_{ff}) to be less than G_0 off-diagonal relative degrees (that is, $k_{ff}<k_{od}$). Due to the aforementioned minimum relative degree limitation of 1, it is therefore necessary that the original system G_0 has off-diagonal elements with a relative degree strictly greater than 1. In other words, to guarantee the first condition to always be true, $k_{od}\geq 2$.

As one condition results in the other being true, if either of these conditions are met then the system would be able to be passivated using diagonal feedforward addition. To support this, the following lemma is supplied:

Lemma 3.1 (Diagonal Feedforward Passivity Lemma)

Let the linear time-invariant system $G\in\mathbb{R}^{n\times n}$ be stable. If each non-diagonal element of the system has a relative degree greater than or equal to 2, then the system is guaranteed to be able to achieve passivity through the summation of a diagonal feedforward system $G_{ff}\in\mathbb{R}^{n\times n}$ that has a relative degree less than off-diagonals.

Comparing relative degrees can be performed across matrix rows or columns. In cases where the system is not symmetric and the condition is met for one dimension but not the other, then the comparison can be performed on the symmetric matrix generated by $G+G^H$. When examined, the previous case examples were shown to not meet these conditions and therefore had no guarantee of reaching passivity. As examples of these conditions, both positive and negative cases for each condition are presented below:

3.4.5.1. Restrictive Application Tests

Test A – Both conditions met

A system that satisfies both conditions is seen below:

$$G_0 = \begin{bmatrix} \frac{1}{10s+1} & \frac{0.6}{(10s+1)(3s+1)} \\ \frac{-0.6}{(s+1)(0.5s+1)} & \frac{1}{10s+1} \end{bmatrix} \quad (3.42)$$

The system is stable and satisfies both conditions of relative degrees of the off-diagonals being greater than or equal to 2 and the diagonal relative degrees. Assuming lemma 3.1 is true, this system should be able to be passivated though the summation of a system G_{ff} that contains a strictly proper transfer function on the diagonal elements, where the denominator polynomial for column j is constructed from the lowest common multiple of the denominators in the j^{th} column of G_0. For the system of (3.42) the feedforward system will have the generic form:

$$G_{ff} = \begin{bmatrix} \frac{K(\mu_1 s+1)(\mu_2 s+1)}{(10s+1)(s+1)(0.5s+1)} & 0 \\ 0 & \frac{K(\mu_3 s+1)}{(10s+1)(3s+1)} \end{bmatrix}$$

Experiments were performed by varying the values of $\mu_1, \mu_2, \& \mu_3$ to directly cancel different poles to determine the effects on passivity. For each of these tests the constant gain K was swept from 0 to 3 in intervals of 0.5. The following table is a summary of which configurations of μ resulted in the system $L(j\omega) = \left(G_0 + G_{ff}\right) + \left(G_0 + G_{ff}\right)^H$ producing strictly positive eigenvalues (indicating passivity of $G = G_0 + G_{ff}$).

Table 3.2 - Passivity Dependence of Pole Cancellation and Gain Value

Variables	μ_3	
(μ_1,μ_2)	3	10
(10, 1)	TRUE for K≥2.5	TRUE for K≥1
(10, 0.5)	FALSE	TRUE for K≥2
(1, 0.5)	FALSE	FALSE

The results show that cancelling the more dominant poles (closer to the origin, with higher μ values) resulted in a larger deviation from the original (and non-passive) system. Intuitively, this

makes sense as the non-dominant poles do not contribute as much to the system for higher frequencies.

For the cases that are marked as FALSE, as the experiment on tested up to K=3 it is not an assured result. However by observing the plots it becomes clear that the general pattern involves a single zero crossing followed by an asymptote towards zero on the negative side; the K value changes to frequency of the zero crossing, but never removes it completely. Assuming this pattern remains for all K values then these systems will never achieve passivity. For the other configurations that do achieve passivity for higher K values the negative asymptote behaviour is not present and a temporary dip below zero is what causes the non-passivity. This dip can be effectively removed with a high enough K value to avoid the double zero crossing altogether. The figures below show the eigenvalue plots of two cases to show a successful passivation when the most dominant poles were cancelled (Figure 3.20) and when the least dominant poles were cancelled (Figure 3.21).

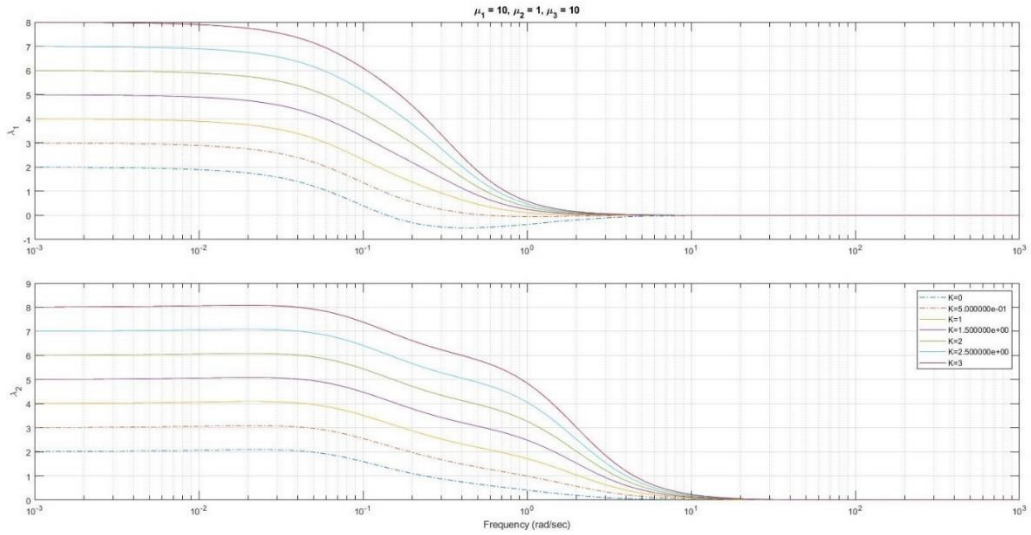

Figure 3.20. Eigenvalue plot showing system passivity dependence on K values with zero placement of μ_1=10, μ_2=1, μ_3=10. Passivity is confirmed for values of K≥1

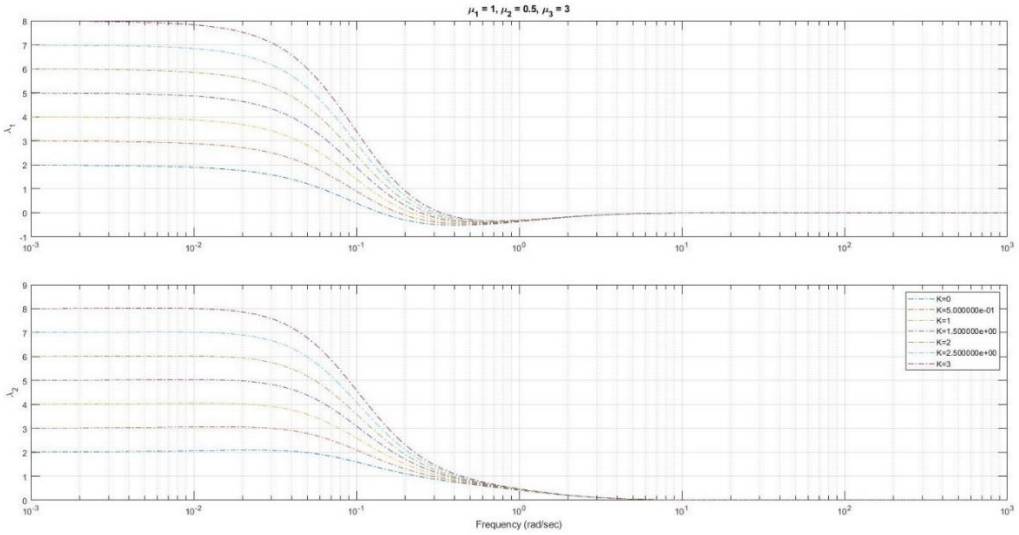

Figure 3.21 Eigenvalue plot showing system passivity dependence on K values with zero placement of μ1=1, μ2=0.5, μ3=3. Passivity is not confirmed for any values of K

Systems that are non-passive are represented by dotted lines, while systems that are passive are represented by solid lines. The asymptotic behaviour is difficult to see within the plots, but the negative dip is much clearer for all values of K in the latter image. Should the system be passifiable with higher values of K, that value would be much higher than for the case of appropriate zero placements.

An additional experiment was performed by flipping the rows of the default system to fail both conditions, then repeating the experiment. This resulted in very similar responses with the same zero configurations being passifiable, which bodes well for the application of the stage 4 permutations to calculate system spectral norm.

Test B – Neither condition met

The system tested for neither condition being satisfied was the simplistic model (3.36) that was previously used in determining validity of diagonal passivation.

$$G_0(s) = \begin{bmatrix} \frac{1}{10s+1} & \frac{0.6}{10s+1} \\ -\frac{0.6}{s+1} & \frac{1}{10s+1} \end{bmatrix}$$

This system has every element characterised by a relative degree of 1, and as such neither of the conditions are met. Under the premise of lemma 3.1, this implies there is no guarantee that the system will be passifiable using strictly feedforward addition. The lemma makes no claims

about the negative cases, and as such it may still be possible for this to achieve passivity. To find the appropriate eigenvalues the new matrix L_0 is calculated:

$$L_0(\omega) = G_0(j\omega) + G_0^H(j\omega) = \begin{bmatrix} \frac{2}{100\omega^2+1} & -\frac{6.6j\omega}{10\omega^2+9j\omega+1} \\ \frac{6.6j\omega}{10\omega^2-9j\omega+1} & \frac{2}{100\omega^2+1} \end{bmatrix} \quad (3.43)$$

Rationalising these transfer functions can be done to have a completely real denominator in all elements. This does not seem to reveal any key details of the system, but is included below for the sake of completion:

$$L_0(\omega) = \begin{bmatrix} \frac{2}{100\omega^2+1} & \frac{-66j\omega^3-59.4\omega^2-6.6j\omega}{100\omega^4+101\omega^2+1} \\ \frac{66j\omega^3-59.4\omega^2+6.6j\omega}{100\omega^4+101\omega^2+1} & \frac{2}{100\omega^2+1} \end{bmatrix}$$

When the eigenvalues of L are calculated across frequency values $\omega \in [10^{-4}, 10^4]$ there are negative values seen starting at approximately 0.16 rad/sec, hits local minimum around 0.5 rad/sec, then asymptotically approaches 0 from the negative side. This behaviour is only present for the first eigenvalue, with the second eigenvalue being strictly positive at all frequencies. This only clarifies the default system G_0 is not passive; tests of feedforward addition to produce $G = G_0 + G_{ff}$ where

$$G_{ff} = \begin{bmatrix} \frac{K(\mu s+1)}{(10s+1)(s+1)} & 0 \\ 0 & \frac{K}{10s+1} \end{bmatrix}$$

were performed to identify the effects of a varying gain K and of shifting the position of the cancelling zero found in the first element. The zero position, set by μ, was held constant while K was scanned from 0 to 10 in intervals of 1. This test was performed several times with the value of μ varying from 0.01 to 100 increasing by a factor of 10 each test. The figure below presents a summary result for primary eigenvalues of $L(\omega) = G(j\omega) + G^H(j\omega)$ for values $\mu = 0.01, 1, 100$ to visually represent the effects of pole cancellation on passivity. The secondary eigenvalue was strictly positive in all cases and as such was not considered important enough to include in the figure, although there was noticeable differences between μ values. While the first two plots for $\mu = 0.01,1$ followed a similar 'low-pass' curve, the plot for $\mu = 100$ produced a 'band-pass' curve centred on 0.3 rad/sec and a bandwidth of 1 decade. This is not likely to be important for this case, but is mentioned to reiterate that both eigenvalues contain valid information for system analysis.

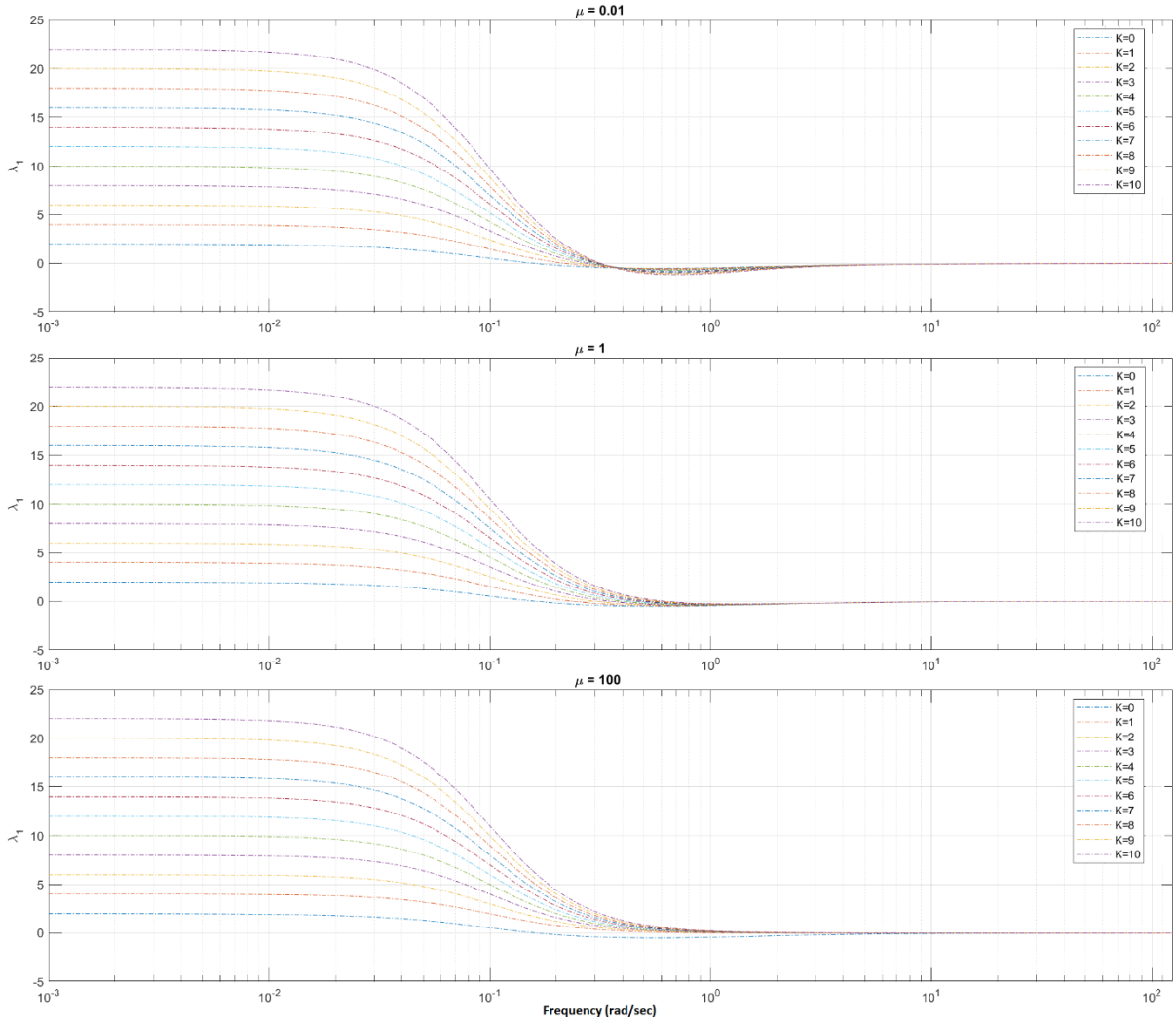

Figure 3.22. Primary eigenvalue curves of $L(j\omega)$ to determine effects of zero placement and K-gain values on system passivity

The lower values of μ show a more prominent 'dip' into the negative values, which suggests that cancelling the dominant poles causes a greater push towards removing the negative characteristics and achieving passivity, without actively completing this goal.

Failure to passivate the test B system does not guarantee that other systems that do not meet the conditions will also fail to be passivated. Strictly speaking, as the experiment was non-exhaustive in the values of K and μ tested, these results do not even guarantee that THIS system is unable to be passivated. Previous examples have shown successful passivity for K values with both upper and lower bounds, meaning if the intervals of K are too large it would be possible to completely skip over a viable K range. The failure to achieve passivity does lend itself to some meaningful analysis when compared to the previous example, which yields contrary results. The previous example was very easily passivated for many different zero placements with K-values

not fully explored. The opposing results suggest, at the very least, that following the conditions creates a system that has a more malleable passivation range which would be extremely beneficial for general control and operation robustness.

3.4.5.2.Effects of correction

With all example systems being confirmed stable then according to the KYP lemma, the system can be passivated if and only if there is a feasible solution set to the linear matrix inequality constraints (3.7) & (3.8). If YALMIP returns the conclusion that the constraints produce an infeasible solution set then the system cannot be passivated without modification. The modification of a feedforward addition was assumed to be all was necessary to achieve passivity for all systems, however results implied this was not the case. Lemma 3.1 was developed to outline the conditions in which a system could be guaranteed passive with a feedforward diagonal transfer function addition, but in doing so found that cases 1-5 did not meet either of the conditions necessary to invoke the lemma's claim of guaranteed passivity. All the experimental results that were collected were left in this thesis as examples of negative results to show the limitations of the proposed optimisation pairing method. This is especially important as three out of five cases contained first order systems which are the most commonly used representation of physical processes, suggesting this method is not well suited for simple models.

For the system in test A with the transfer function (3.42) the optimiser method confirms the problem to be "primal and dual feasible" with an optimal solution if the Schur constraint is not included; with the additional constraint the solver still claims the problem is feasible with an optimal solution, however the final system G is shown to be non-passive. Even with the simple addition of an objective to minimize the solver claims the problem is feasible yet produces a non-passive solution; the inclusion of an objective should not affect the feasibility set, only the selection of the solution from the set. It should be impossible for the problem to be labelled as feasible but produce a non-passive system, yet a non-passive solution claimed to satisfy the KYP lemma. The difference in eigenvalue plots is shown below:

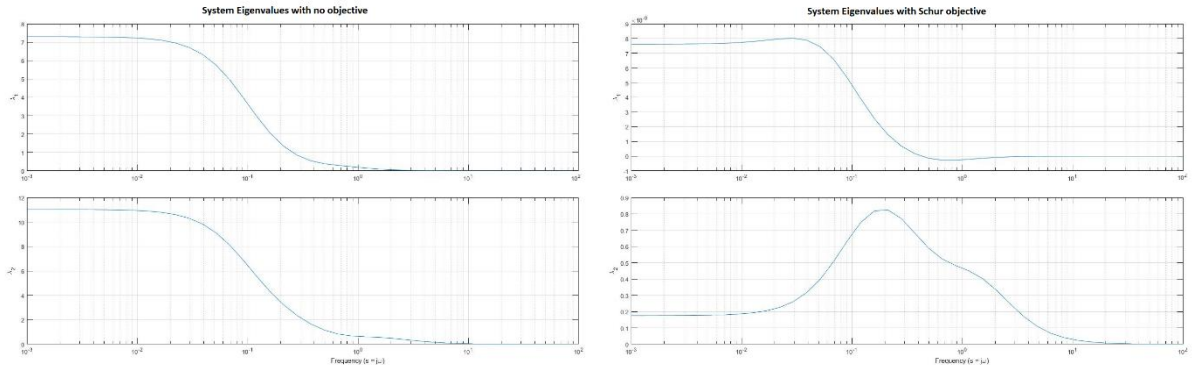

Figure 3.23. Comparison of passivity between system with no objective (left) and objective to minimize γ under the constraint $\gamma \leq CC^T$ (right). Passivity is shown present for the no-objective problem but not present for the γ-objective problem

The calculated norm values were also heavily influenced by the objective, as they should be. For the no-objective result the norm was calculated at 257.987, for the system with no additional Schur constraint but an objective function minimising the sum of all elements of C the norm was calculated at 80.713, and for the system with the additional Schur constraint and the objective of minimising the spectral norm of C the norm was calculated at 78.073. These results are promising for the minimisation of spectral norm, however the issue of non-passivity remains with no clear explanation of how these results are being produced. The negative value does have a magnitude of less than $1x10^{-9}$, but follows the negative asymptote for all high frequency values and is unlikely to be caused by rounding issues.

As the solver claims the problem is feasible yet produces a result that should not be possible it becomes very difficult to troubleshoot the causes of failure. Due to time constraints no further experimentation or analysis can be committed to this problem and the conclusion of the results must simply end stating that the optimisation method shows promise for spectral norm minimisation and achieving passivity, but the current method for implementation contains errors that may hinder results or suggest contradictory solutions.

3.5. Discussion

3.5.1. Existing pairing methods and applications

The overarching goal of this chapter was to develop a method of input-output selection that can be performed easily while also guaranteeing passivity of the final combined system in the most conservative approach possible. Each stage of development led to the final script that produced the results seen in Stage 4, which compared the recommendations with existing control configuration selection methodologies. Which pairing to select will inevitably come down to the designer, where each of these CCS methods act simply as an analysis tool to identify features which are desired within their system. Each method contains its own benefits and are more suitable for specific systems, so the proposed method does not supersede the existing technologies but may offer new analysis techniques to improve system passivity, controllability, and stability. For instance, Bao et al. (2007) focuses on finding the pairing that maximises the frequency bandwidth that a system is already passive. While most techniques aim to minimize IO interactions, Bao's method does not focus on this due to the evidence that some interactions may actually help reject load disturbances. It also does not rely on diagonal dominance, which is a present factor in the developed optimiser method for cases that are not easily passivated. The passivity-based method is beneficial if a system is only needed to be passive in a specific frequency band of operation, however if the system must be strictly passive then further changes must be made.

The more simplistic methods of relative gain array and dynamic relative gain array are beneficial for cases where system models are accurately known and linear (Khaki-Sedigh and Moaveni 2009) but can be expanded to weakly nonlinear systems (Kadhim, Birk, and Arranz 2016). None of these advanced implementations were required for the cases used in comparison testing, but do show the benefit of using existing techniques: they have been developed, expanded, and applied to a plethora of cases that reduce the inherent restrictions of the technique. Despite all this additional support and documentation, these techniques do not focus on the passivity of the system which is the primary focus of the project. Generalised dynamic relative gain is another pairing criterion that determines total interaction potential, and would be better suited for system analysis looking for minimal input-output crosstalk (Huang, Ohshima, and Hashimoto 1994). Should generalised dynamic relative gain be used as the pairing method, any interactions between signals would be represented negatively regardless of the destabilising effect of the cross-talk. Form these examples it becomes clear there is no way to definitively state which pairing method should be used, and the decision must be made based on the scenario at hand.

The results in Table 3.1 only provide the final pairing suggestion without any remarks as to why these suggestions were made. Without further exploration this makes the results fairly abstract and difficult to picture, so a step response is provided in Figure 3.24 to show the response for each pairing. These response graphs are directly copied from Mc Avoy et al. (2003) and show the system response for when the first input is a step response and the second input is zero (left column), and when the first input is zero and the second input is a step response (right column). These responses are for when the system (3.38) is in a closed loop system with a tuned PI controller (with different PI parameter values for each pairing). The response shows the effects of crosstalk between inputs and outputs as well as the general ability to track the reference input signal. For a step on input 1 it can be seen the diagonal pairing reaches transience faster than the off-diagonal pairing but causes a much larger disturbance in output 2 for a longer duration. For a step on input 2 the off-diagonal pairing has faster tracking but causes a longer sustained disturbance in output 1. An important feature of the system is that both diagonal elements have a large time delay of 40 seconds while the off-diagonal elements only have a delay of 4 seconds. The off-diagonal pairing is able to exploit this fast response time to a much higher degree; this is stated as being valuable enough to outweigh the poorer y_1 response, although no justifications are given for this claim. Interestingly enough, the exact same example was used by Khaki-Sedigh and Moaveni (2009) [Example 2.3.3] as justification for the RGA suggestion of the diagonal pairing.

This example is a good observation as to why pairing selection is no simple question. Where the RGA and Bao's passivity method recommend the diagonal pairing and would achieve better responses on y_1, the DRGA and optimisation method select the off-diagonal pairing that minimises crosstalk in y_2.

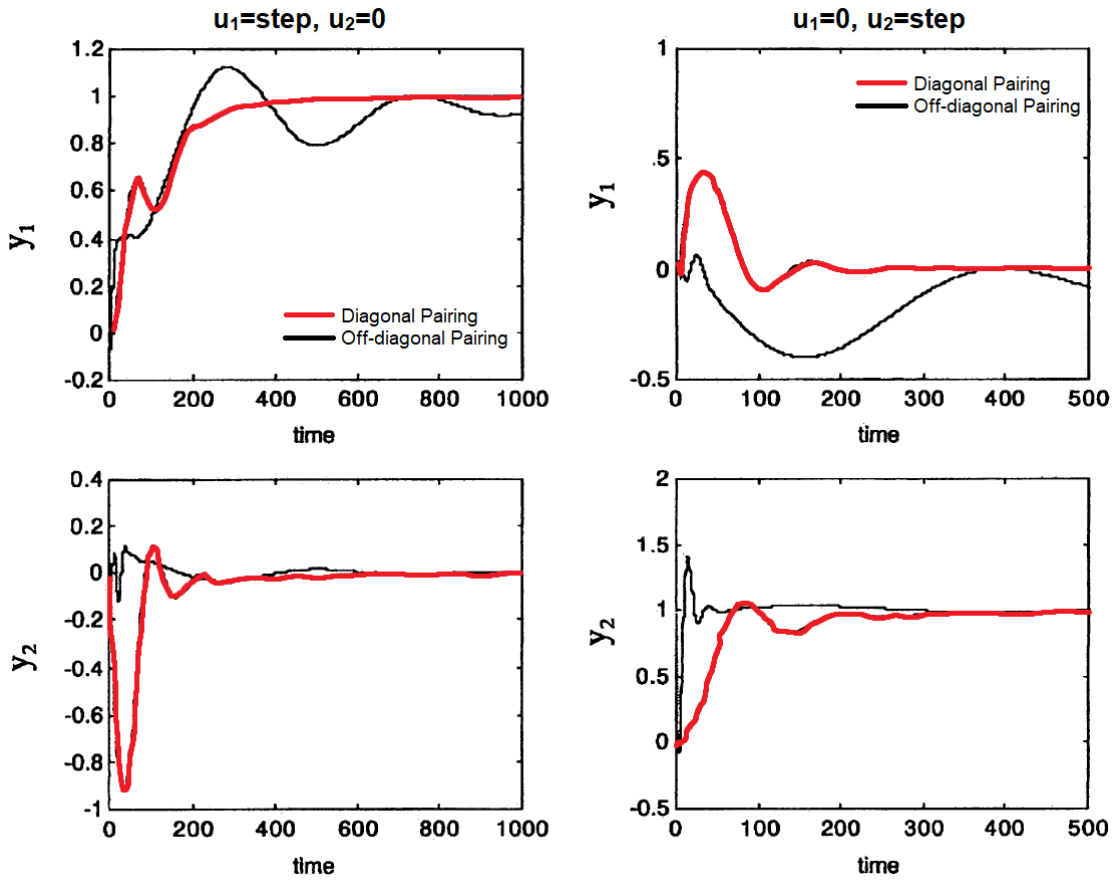

Figure 3.24. Step response comparison of different pairing selections for the McAvoy (2003) example used in Case 3. Time is presumed to be measured in seconds and the system output amplitude is unitless

The pairing methods used to generate the results are far from an exhaustive list. The RGA-based techniques can be expanded upon, many of which are mentioned in chapter 3 of the textbook by Khaki-Sedigh and Moaveni (2009). As these techniques are not applicable to the current problem they are not discussed in detail, however for future expansion these techniques may provide valuable assistance for system development.

The benefits of passivity-based control have been studied in-depth by Bao in several different papers focusing on aspects such as robust control using sector stability (Bao et al. 2000), failure-tolerant control (Bao, Zhang, and Lee 2002), and input disturbance suppression for nonlinear systems (Su, Bao, and Lee 2006). Research was also published discussing the important properties of decentralised integral controllability and its relation to passivity (Bao, James McLellan, and Fraser Forbes 2002). These papers show the applications of passivity-based process control are quite expansive to many different efforts and support the belief that rehabilitation engineering would thrive with these control technologies. A theorem is presented that supports the efforts being made by the optimiser method relying on a diagonal feedforward

controller to passivate a system (Theorem 1 of (Bao et al. 2007)). The only additional requirement set by this theorem that was not explicitly set within the optimiser method was that the feedforward transfer function must also be passive. This was not included as it was believed to be inherently true given the system was guaranteed to be stable and diagonal which would guarantee positive eigenvalues. The paper also extends to claim and offer proof that once the system has been stabilised and passivated, then there exists a MIMO controller that can also retain passivity; this conclusion is hugely beneficial for reference tracking systems and can be applied to the optimiser method as well. The benefit of the optimiser method over Bao's passivity-based method is that where the passivity method makes a recommendation of IO pairing, the optimiser method provides a recommendation of the feedforward passivity system along with the pairing. The design aspect of the feedforward system was not the focus of Bao's experimental pairing method and as such was not addressed in the paper. For the optimiser method, the resulting combined system will in theory be a small perturbation from the original system that can then be controlled as if it were the original system.

3.5.2. Optimiser method shortcomings

A. Passivation

One potential point of contention of the optimiser method is how the minimal passivity is applied across the frequency domain. The optimiser guarantees the system is passive across all frequencies, however it is not clear that the minimal excess passivity is added to the system for all frequencies. If a system is passive at all frequencies excluding a small frequency band, the system may add enough passivity to all frequencies to remove this non-passive behaviour. Although the minimisation of the spectral norm should prevent this, and some experiments do seem to function as intended, some transfer functions are simply given extreme amounts of passivity to guarantee positive eigenvalues with very minimal frequency-dependent considerations. To avoid this behaviour the added transfer function should be constructed to have eigenvalues that follow a 'bandpass' curve with very small eigenvalues for the frequency ranges already passive and within the non-passive range eigenvalues as small as possible to achieve passivity. No clear explanation for the poor results have been found; since these results produce larger spectral norms than a more restricted addition, they should be naturally filtered out through convex optimisation during the control configuration selection process. Despite this, if this behaviour occurs for all possible pairings then the final result will produce a system that has excess passivity and would lead to an overly conservative controller design. More

examples must be tested to try and find a common factor that may cause this issue. Finding working examples in itself can be quite difficult due to the personalised aspect of pairing selection and no ground-truth pairing is available or even quantifiable.

B. Dimensionality scaling

One consideration to be made is how well each pairing method is able to scale with a system's dimensions; as the number of inputs and outputs increase, how long will each pairing method require to return a favoured selection. For the most basic method of RGA, the elementwise matrix multiplication will cause the number of calculations to scale with quadratic growth, but overall time will not likely be a problem for a system with less than 100 inputs and outputs. DRGA follows a similar runtime behaviour with the added dependence on frequency range. Both the passivity-based method and the optimiser method require calculations from every possible permutation of input-output pairings. This will lead to the number of calculations scaling factorially and, when combined with the more intensive computations required, the computation time will become infeasible for relatively low input counts. Bao's method has opportunities to prune the number of calculations by immediately skipping any system that does not meet the conditions of being decentralised integral controllable, which may speed up overall computation times but does not address the mathematical foundation of the number of pairings scaling as a factorial. Similar pruning may be possible for the optimiser method by disregarding any system that does not meet either of the conditions of relative degree. This has the potential to drastically decrease the number of computations, as only one element not meeting the conditions would eliminate every possible combination sharing that element. Unfortunately as the conditions are not logically biconditional, failing the condition check is not enough to be certain the system cannot be passivated and it may still be worthwhile to calculate the spectral norm and passivity end result. It may still be expedient to perform experiments into the negative conditional cases to identify recurring patterns or resulting systems, potentially leading to discovering large subsections of pairings that are guaranteed to be non-optimal.

Along the same lines of dimension scaling, it is important to understand the limitations of the conditionals for higher dimensions. As stated earlier, the relative degree of the off-diagonal elements must be greater than 1 and the relative degree of the diagonal elements. While easily visualised for a 2x2 system, larger matrices show that all non-diagonal elements must meet these conditions, not just the off-diagonal elements (diagonal elements from bottom-left to top-right). These elements will make up the majority of the overall system and can be an extremely limiting constraint for real-world modelling. Again, although these conditions are not necessary

to allow the system to achieve passivity, they are sufficient and would allow much stronger assurances during the design process.

C. System modelling

To accurately represent a physical system, the relationship between inputs and outputs must be represented mathematically through state values using physics first principles. Although this is true for every control system, more complex systems are much harder to model accurately and as the optimiser method is best suited for complex systems it becomes more difficult to reliably implement on real-world examples. Simple systems may not benefit from the minimal perturbation goals of the optimiser technique as much as complex systems would, and may favour the more basic pairing methods like RGA. Along with this avoidance of complex systems, due to the constraints outlined in lemma 3.1 the modelling of the system is guided towards higher-order transfer functions which adds further restrictions on the system modelling and pairing method selection.

The final potential problem with forming the system model is the current calculation method for the system matrix fraction description. Due to unalterable behaviours within the MATLAB symbolic toolbox, the format of transfer function polynomials is always converted to avoid decimal values and will instead favour rational fraction representations. When calculating the MFD, this representation becomes an issue as the gain values become scaled to much higher values that are less manageable. For example, the transfer function

$$H = \frac{-0.2}{(18.3s + 1)(5.6s + 1)}$$

needs to be represented through the MFD, which can be done in an infinite number of forms. The desired form is shown on the left, while the MATLAB form is shown on the right, below:

$$\frac{-0.2(28.3s + 1)(0.62s + 1)}{(18.3s + 1)(5.6s + 1)(28.3s + 1)(0.62s + 1)} =$$

$$= \frac{-10(31s + 50)(283s + 10)}{(183s + 10)(28s + 5)(283s + 10)(31s + 50)}$$

These transfer functions are equivalent mathematically, but cause problems when aiming to extract coefficient values. For instance, the coefficient of the highest denominator degree of the desired transfer function is 1798.1 while the MATLAB highest degree coefficient is 44 952 852, which is 25000 times larger. Although the scaling factor can be easily calculated and the coefficients rescaled to their desired values, the symbolic toolboxes resistance to decimal values

continues to cause problems throughout in system representations. As the structure of the feedforward system is directly dependent on the MFD, these results directly affect the optimiser output for determining magnitude values for the feedforward output matrix. Converting to always represent the denominator matrix as a monic polynomial was attempted with no fruitful endpoint. It is believed that these cascading issues do have negative impacts on the optimiser's ability to determine optimal solutions as numerical problems have been flagged as an error in several experiments throughout the project. Numerical problems occur when magnitudes between variables is too large for the solver to appropriately compare values.

3.5.3. Optimiser method benefits

Although the previous sub-chapter focused on all the negative aspects of the proposed optimiser method, there are negative aspects to every other pairing method and positive aspects to the optimiser method. Structured singular value (SSV) approaches allow pairing analysis over frequency ranges of interest and produce a different interaction measurement. This may have benefits, but for the purposes of passivity analysis the entire frequency range must be explored, making the optimiser method the better suited interaction quantifying method. SSV is also required to have a block-diagonal controller and all theorems require the assumption that G_0 and G_{ff} have the same number of poles in the right-half plane. This makes SSV more restrictive in application than the optimisation method, showing even restrictive methods to be useful to the field.

Another benefit is the use of a diagonal feedforward system allows the system to become diagonal-dominant for lower frequencies regardless of relative degrees. Becoming diagonal dominant allows guaranteed passivity through the Gershgorin circle theorem, centring the eigenvalues in excessively passive states. It is mentioned by Bao that diagonal dominance is often relied upon in pairing methods and that they strived to avoid this, although no detail was given as to why. Mentions are made of signal interactions potentially rejecting load disturbances, obviously less applicable in a diagonally dominant system with minimal interactions. However for human-robot interactions the disturbances may be generated by the human which should not be rejected, and signal interactions are generally undesirable, as such avoiding diagonal dominance is not a perceived as important for the task at hand. From this, along with the previously developed results, diagonalization should always be possible. In cases where the feedforward system not be dependent on the MFD structure and a relative degree of 0 is possible, this method would allow even further applications as the conditions for guaranteed passivity would be nullified.

3.6. Conclusion

A multi-input multi-output system is able to represent complex real-world physical systems that have their behaviour dictated by multiple variables. In the case where these variables affect multiple outputs simultaneously, any controller designed to manage the system must consider these interactions, and are labelled as “centralised controllers”. A “decentralised controller” is able to decouple these variable interactions and isolate specific system characteristics to be controlled by specific system inputs. Although both controllers are feasible and have advantages, decentralised control is more commonly employed for its ease-of-implementation and robust nature. Decentralising a controller therefore allows better disturbance rejection for most cases and would be especially useful for systems relying on inconsistent signals such as electromyography in rehabilitation settings. There are further underlying system properties that are beneficial to basic control and would assist in human-robot interactions. These properties, such as stability and passivity, can be augmented into the system through mathematical techniques, effectively transforming the system to one which would be better suited for specific tasks such as rehabilitation engineering. Where stability guarantees a system’s state values remain finitely bound, system passivity guarantees energy dissipation rather than accumulation and guarantees stability during system coupling. The additional properties of passive systems also reduce the need for fault-tolerant designs or disturbance rejecting controllers, as the worst case scenarios failures or disturbances cause would still not violate the stability of the system. For late-stage development of robotics, this could lead to a more ergonomic and economic design, with less hardware components required for operation. This chapter discussed these techniques and how they can be implemented in practice to convert a non-stable non-passive system to a state that can be effectively decentralised and controlled by a standard MIMO controller for reference tracking or force restricting purposes while also guaranteeing system passivity.

A MATLAB script was created to allow systems that lacked stability or passivity to acquire these properties through full-state feedback gain and feedforward passivity gain, respectively. Results showed that several examples of non-passive systems could be pushed to passivity through this method when performed either manually or using an optimisation toolbox known as YALMIP. Further analysis found that not every system was passifiable using a diagonal feedforward system, leading to the discovery of conditions that guaranteed passification was possible. A system that has relative degree of the diagonal elements greater than or equal to 1 and relative degree of the non-diagonal elements greater than or equal to 2 is guaranteed to have a diagonal

feedforward system that pushes the system to being passive. This condition, presented as a lemma, show the mathematical certainty of the method, however when implemented through the optimiser technology the results were self-contradictory. Systems were shown to pass the KYP lemma which guarantees the system is passive, yet eigenvalues of the system were shown to be negative for certain frequency bands implying the system was not passive. In optimisation terms, the optimisation solver claimed the applied constraints produced a feasible solution set, yet the solution it would recommend did not fall within the feasible set. The contradictory results were present for several (but not all) examples and further analysis is required to determine the source of this discrepancy between theory and practice. In spite of this, as the KYP lemma was seen functioning appropriately and was considered the more robust test, the optimisation method for pairing determination was deemed successful with further research justified. This new pairing method will in theory allow a robotic system to be designed passive to allow better coupling with a human musculoskeletal system which is also expected to be passive. The decentralised nature of the system will allow specific EMG clusters to act as the signal inputs and specific motions to act as the output. Future works will continue to try and model existing robotic devices present, apply the pairing method to real-world examples, and identify the best input-output pairing for controlling a rehabilitation robot with respect to passivity.

The unconventional use of using passivity-based process control in tandem with optimisation techniques is what sets this work separate from its counterparts. While passivity-based process control has been used to determine the best input-output pairing, this was applied to systems that had some configurations passive by default. The innovative step used here is to determine how far each pairing is from becoming passive at the cost of minor alterations. Where the former is able to identify which pairing has a small interaction potential between inputs (potentially causing uncontrollability in some outputs), the latter is able to identify which pairing can reach passivity and naturally couple with a human ankle. The limitations discovered do inhibit the widespread use of the passivity optimisation method, but within the realm of robotic design, as long as information about the inputs and outputs is known before designing the system, it is possible to design a system such that the method is guaranteed to be applicable.

Chapter 4 - Using Reinforcement Learning to Construct Adaptive Passive Controller

4.1. Introduction

Reinforcement learning (RL) is a form of machine learning that has been developing at a rapid rate for the past few decades, with its applications spreading to every field of research. Due to the nature of how RL converges on a solution, any problem that can be solved through a trial-and-error approach is well suited to use RL. Control engineering can benefit from these new techniques in cases where the plant processes change over time or are highly nonlinear and difficult to model. A traditional Proportional-Integral-Derivative (PID) controller is the most common of all controllers, found in many day-to-day technologies and industrial machines. By improving the controller, the resulting system output may be closer to the desired performance; depending on the system this may have a positive effect on costs, applicability, or customer satisfaction. In the case of robotics, the PID controller is used in a closed-loop system to reduce the error in reference tracking. A robot instructed to follow a specific reference path through physical space will be able to follow said path much closer for a well-tuned controller. Rehabilitation robotics can be designed to guide a user through physiotherapy exercises and as such would benefit from a reference tracking control system. For a user with limited motor ability the robotic device would contribute majority of the work, such as the initial prototypes of the LOKOMAT (Colombo et al. 2000) or the wearable ankle device presented by Jamwal et al. (2014). Users with stronger muscle outputs would be provided other forms of feedback, such as visual stimulus (Duschau-Wicke et al. 2010), to guide them towards the appropriate exercise motions, and would still benefit from reference tracking but with additional considerations of 'disturbances' caused by the operator. The tuning of a PID controller has no correct solution and is often performed through experience and trial-and-error adjustments. To supplement this approach, established tuning methods do exist, such as Ziegler-Nichols (Ziegler and Nichols 1993) or by the use of an auto-tuner (Mathworks 2022b). This makes PID tuning an opportune problem for RL, especially if the PID is designed to be adaptive and changing over time. By training an artificial intelligence to perform the tuning task itself, a system that deals with uncertainty in its environmental interactions can develop all the benefits of an expert re-tuning the system as it is needed during operation. The adaptive controllers are also not limited to

simple PID controllers. Impedance or admittance controllers are extremely beneficial in regulating the interaction forces of a system, and as such an adaptive admittance controller will be able to alter how much force is permitted at any given state. These states are defined by dozens of potentially user-specific system measurements and are able to incorporate nuance that has the potential to improve the performance and safety of the system overall. As mentioned by Lee and Hogan (2016b), knowledge of both energetic passivity and mechanical impedance are critical for ensuring coupled stability between human and robot. It is mentioned that the driving point impedance of a system being energetically passive is a sufficient condition to guarantee that this system, when coupled with a stable passive system, will remain stable. For the specific example of an ankle-foot orthosis (AFO), as the human ankle is considered stable and passive, coupling the human ankle to an AFO will remain stable if the AFO can remain energetically passive.

Improving the traditional controller design by tuning the parameters with RL allows system design to optimise towards a predefined behaviour desired by the system architect. Adaptive controllers allow for parameters to change throughout system execution to better suit the current conditions. Tuning these parameters based off input data allows RL to make decisions from imperceptible patterns in data that humans would never notice. Using a neural network (NN) to read system observations and return an appropriate control signal to the physical system model is an alternative method and perfectly analogous to using a classical feedback controller. In the case where the output of the NN connects directly to the plant and the classical controller is bypassed, the system is described as “end-to-end” learning. It is also possible to use the NN in conjunction with a classical controller, with the output signals acting as the adaptive controller parameters instead of bypassing the controller completely. This method may seem unnecessarily indirect, however end-to-end learning has been reported to perform worse than adaptive controller tuning (Lee, Lee, and Yim 2020). It is for this purpose that RL is used to help in the design of rehabilitation robotics. Attempts at both end-to-end learning and adaptive parameter tuning are implemented in this chapter to help develop an understanding of the reinforcement learning process, and compare the results of each system with reference tracking ability. The development of an RL agent was performed for three separate system configurations involving an ankle rehabilitation robot and tracking a desired position reference signal. The initial experiment aimed at developing an end-to-end system to directly control torque. The second experiment focused around constructing an adaptive PID controller by using the RL agent to tune the controller gain values and observe how well they adapted under changing

environment conditions. The work on RL-based PID control was published as a paper under the title “Comparison of Constant PID Controller and Adaptive PID Controller via Reinforcement Learning for a Rehabilitation Robot” and presented at the Australia & New Zealand Control Conference 2022. The third experiment expanded upon the second experiment by adding an adaptive admittance controller to interact with an environment that the robot has no direct control over, nor any prior knowledge on what will be encountered. The final experiment is best suited for rehabilitation purposes as human-robot interactions contain large amounts of uncertainty in the environment that must be accounted for in the artificial intelligence control system. Using RL for control tuning purposes is not a new application, but the novelty of the experiments in this chapter comes from the analysis and comparison of results between control options to best determine when to use each technique for a specifically designed ankle rehabilitation robot.

A tangential experiment was originally planned to use collected lower limb sEMG data to predict and categorise the types of ankle motions that were being performed during standard locomotion and daily activities. This EMG-to-motion decoder would have allowed sEMG data to function as the system input such that a user’s intended motion trajectory would act as the reference signal to the control loop. The rehabilitation robot would then reposition itself to match with the user’s intended ankle position and assist in rotational motions. The amount of assistance provided by this robot could be adjusted through admittance control and would allow both passive and active rehabilitation exercises to be performed. The use of sEMG for gesture classification has already been proven feasible for real-time technology applications (Zhang et al. 2019), however instead of using supervised learning to train the network this experiment planned to use reinforcement learning. Unfortunately due to COVID-19 and time constraints this experiment was deprioritised in favour of experiments that did not require close physical contact with multiple participants. However progress was still made and discussed further in Chapter 5 - . The overarching goal of this chapter is to develop practical skills in reinforcement learning to produce a control system that may be implemented into a rehabilitation robot. This control system must be able to appropriately assist the user in moving their ankle towards a predefined angle with varying degrees of interaction force. The system must also retain its energy-passivity to guarantee the total system, when coupled with the user, remains passive. This approach will be conducted through an adaptive admittance controller that applies forces to retain passivity and minimise excess passivity throughout the simulation. Future works would

aim to retain this passivity and minimal excess passivity during the gait cycle for the system coupled with a user.

This chapter aims to explain the differences between classical control methods and reinforcement learning-based control methods such that future rehabilitation robotics will be able to make more informed decisions on how to approach design. Ideally, this impact will lead to a better understanding of when more advanced controller implementation is warranted and will allow more diverse rehabilitation devices to exist in their better suited niche operating conditions.

4.2. Methodology

This chapter focuses on identifying where in the ankle rehabilitation robot design process could benefit from reinforcement learning and hypothesising how that implementation may be performed. Through the literature review, these stages were identified as replacing the controller with the function approximator, automatically tuning an adaptive controller, and converting bio-signals to user intention. Experiments and simulations were planned for each of these stages for the purposes of confirmation that each idea could be used in practice for this project's long-term goals, as well as developing experience in the artificial intelligence field. The first experiment focused on developing the basic skills to implement an actor-critic reinforcement learning agent to track a step or sine wave reference signal. The second experiment, developing an RL agent to adaptively tune a traditional controller, utilized a PID controller to determine robot torque values that would improve reference tracking. The third experiment expanded upon the second experiment with an admittance controller to help interacting with environments with uncertainty. This expansion aims to improve overall safety and stability for the user. The reference signal for each system would represent angular displacement, making a step input represent aiming for a specific angle and a sine wave input represent a continuous back and forward motion of the joint similar to a passive rehabilitation exercise.

All simulations were performed using MATLAB 2021a (v9.10) and Simulink (v10.3) and relied heavily on the reinforcement learning toolbox. A Simscape multibody model representing a 1 degree-of-freedom robot available on-campus was used as the plant in the control loop. This robotic device, referred to as the PaddleBot, is designed to allow a user to attach their foot to a large platform which can be rotated around a fixed point in space. As the input to the model is

a torque value and the output is an angular position value the model can be summarised as a double integrator. The motor used in the device is a MAXON EC45 brushless 250W motor (part: 136207) with a nominal torque of 331 mNm and a stall torque of 2540 mNm. This motor is combined with a MAXON planetary gearhead GP 62 A (part: 110508) which has a max continuous torque of 50 Nm. An image of the PaddleBot can be seen in Figure 4.1, while the Simscape multibody model can be seen in Figure 4.2.

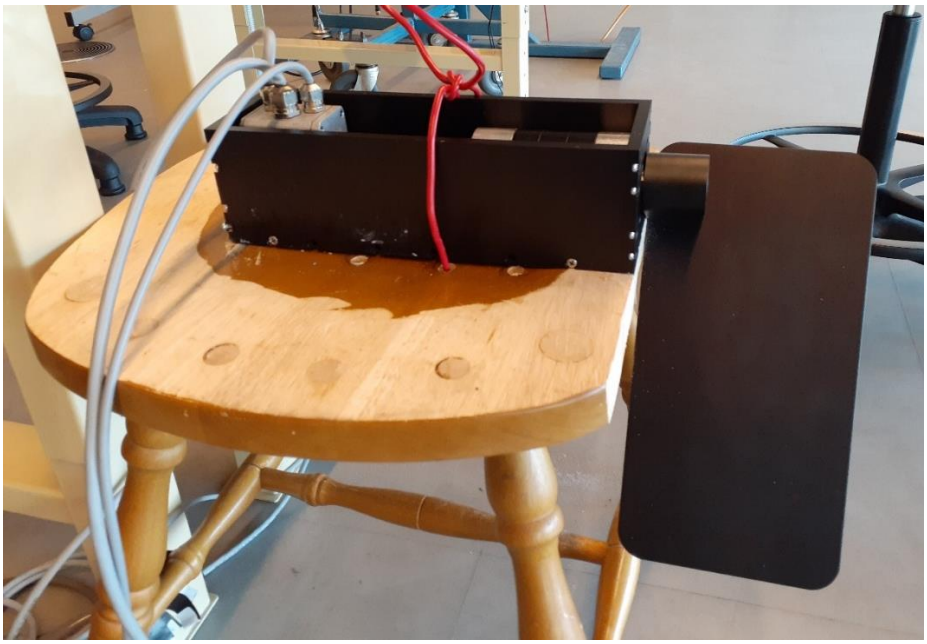

Figure 4.1. PaddleBot robotic device mounted on knee-high footstool

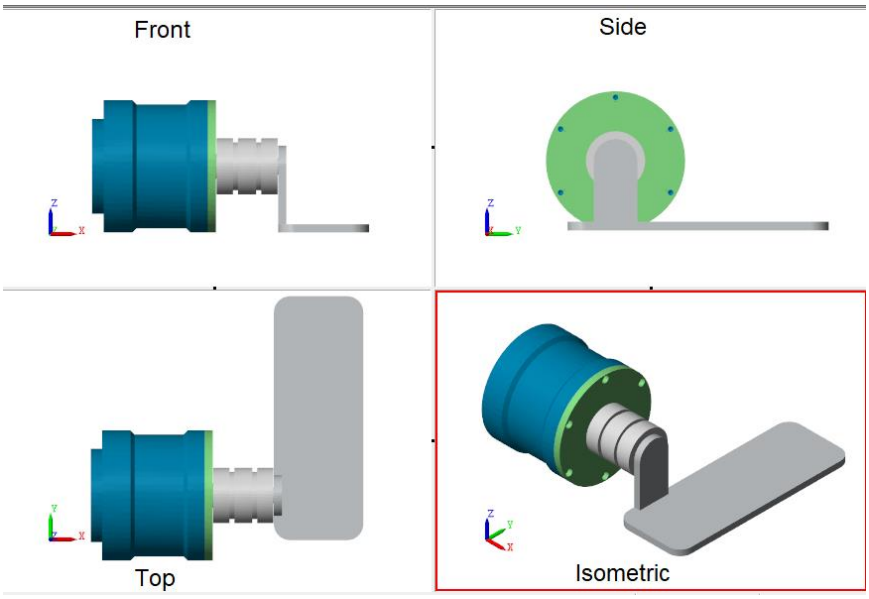

Figure 4.2. Simscape multibody model of PaddleBot displayed in orthographic format with an isometric view presented in the bottom right. The Simscape model was utilized by all Simulink experiments

4.2.1.Estimation of PaddleBot Transfer Function

An estimation of the transfer function was constructed for the purposes of understanding the behaviours of the system. The plant requires a torque as an input and returns an angular position, which can be calculated by the physical relation between angular acceleration and torque:

$$\tau = I * \ddot{\theta} \tag{4.1}$$

The moment of inertia is calculated primarily from the foot paddle as it is the dominant mass of the system and hangs displaced from the axis of rotation. The moment of inertia can be approximated using the parallel axis theorem for a rectangular prism. The connecting arm mass is ignored for simplicity, as only the polynomial structure of the transfer function is important, not the overall details:

$$I = \frac{1}{3}ml^2 + mr^2 \tag{4.2}$$

where l is the foot-length of the paddle and r is the distance between the axis of rotation and the foot paddle. By using the Laplace transform on (4.1) the transfer function can be algebraically constructed as:

$$G(s) = \frac{\theta}{\tau} = \frac{1/I}{s^2} \tag{4.3}$$

This shows that the motor component of the PaddleBot acts as a double integrator. However as the paddle mass will also be affected by gravity, additional torque will be present at the system input that can be calculated through the equation of motion of a simple pendulum affected by gravity:

$$\tau_g = mgr * \sin\theta \tag{4.4}$$

$$\tau = I\ddot{\theta} + mgr\sin\theta$$

The nonlinear behaviour of the sine function causes problems for transfer function construction, but a linear estimation can be made such that $\sin(\theta) \approx \theta$ for small angles. The estimate transfer function is a simple gain present on a negative feedback loop (negative as the gravity component g is conventionally negative) and becomes much easier to represent as a linear polynomial fraction. The reference tracking ability of the Simscape model, the nonlinear transfer function, and the linearised transfer function were compared to confirm whether an accurate transfer function had been constructed and whether substitutions between each model could be

utilised. Each plant was placed in a PID-controlled closed loop with equal PID controllers, with each loop parallel within Simulink, which can be seen in Figure 4.3 below. The same input signal was fed to each loop, with the output response recorded by a scope. The moment of inertia (I), along with the gravity gain coefficient (mgr), were estimated through calculations and trial and error until the output response from the nonlinear transfer function (TF1) sat approximately equal to the Simscape model. The linearised transfer function (TF2) utilised these constant gain values and simplified into a single transfer function through closed-loop feedback equivalence:

$$G_{CL}(s) = \frac{G}{1 + GF} = \frac{12}{s^2 + 13.2} \tag{4.5}$$

The results of the reference tracking showed extreme resemblance between the PaddleBot model and TF1, which suggests the transfer function model is accurate and a reasonable replacement to the Simscape model if ever necessary. The linearised model also resembles the Simscape model, however deviations are present, especially at higher angular measurements. This is to be expected due to the approximation being used only being applicable at small angles. Therefore, TF2 cannot be reliably used as a substitute plant unless only minimal motions are to be expected during operation.

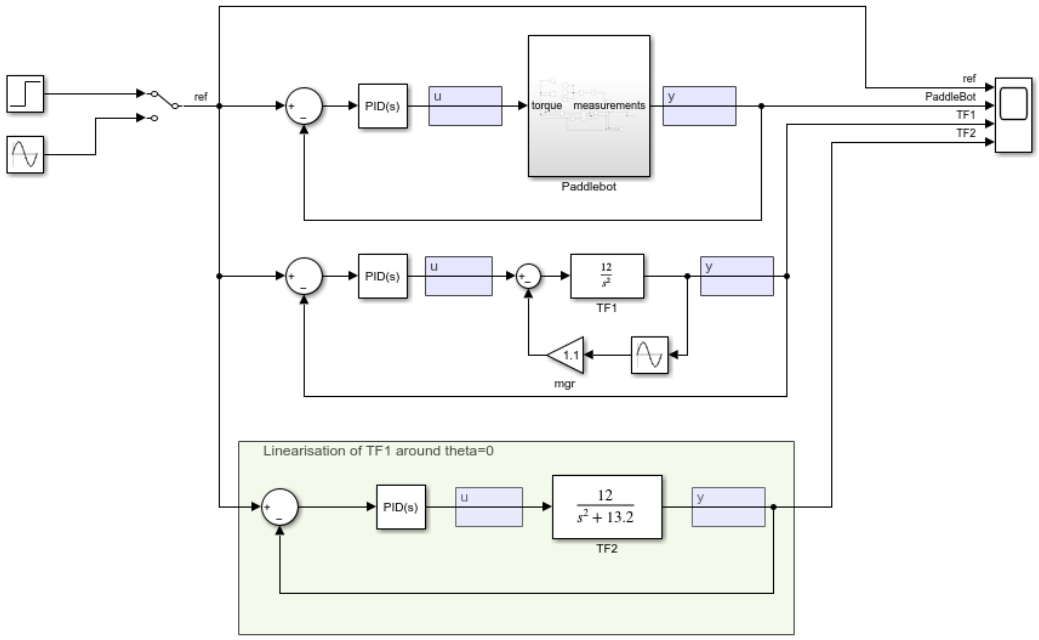

Figure 4.3. Simulink test for transfer function equivalence of the Simscape PaddleBot model, the nonlinear transfer function approximation (TF1), and the linearised transfer function approximation (TF2)

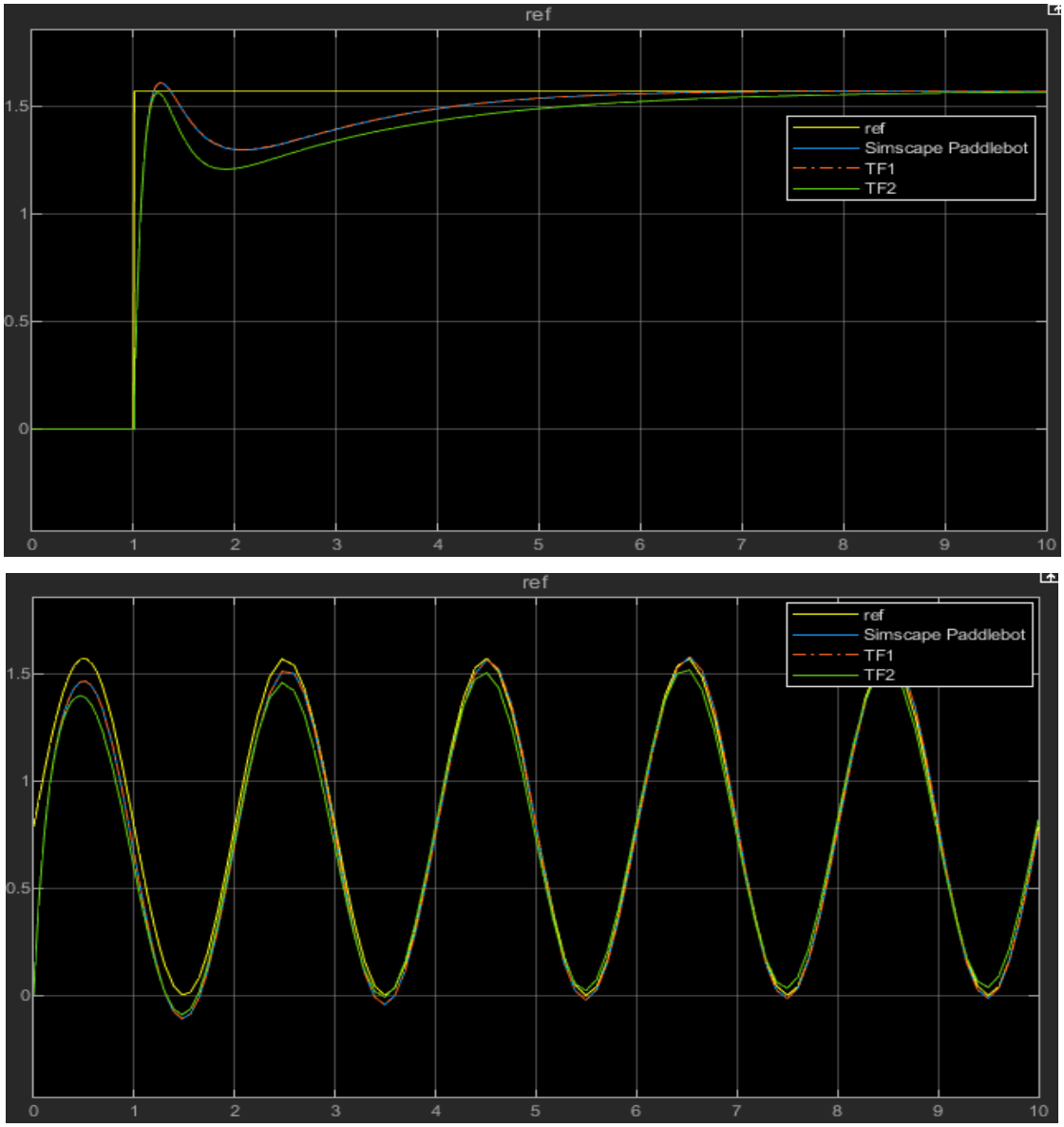

Figure 4.4. Comparison of output response from the Simscape PaddleBot model (blue), nonlinear transfer function approximation TF1 (orange), and linearised transfer function approximation TF2 (green) for a step input (top) and sine input (bottom)

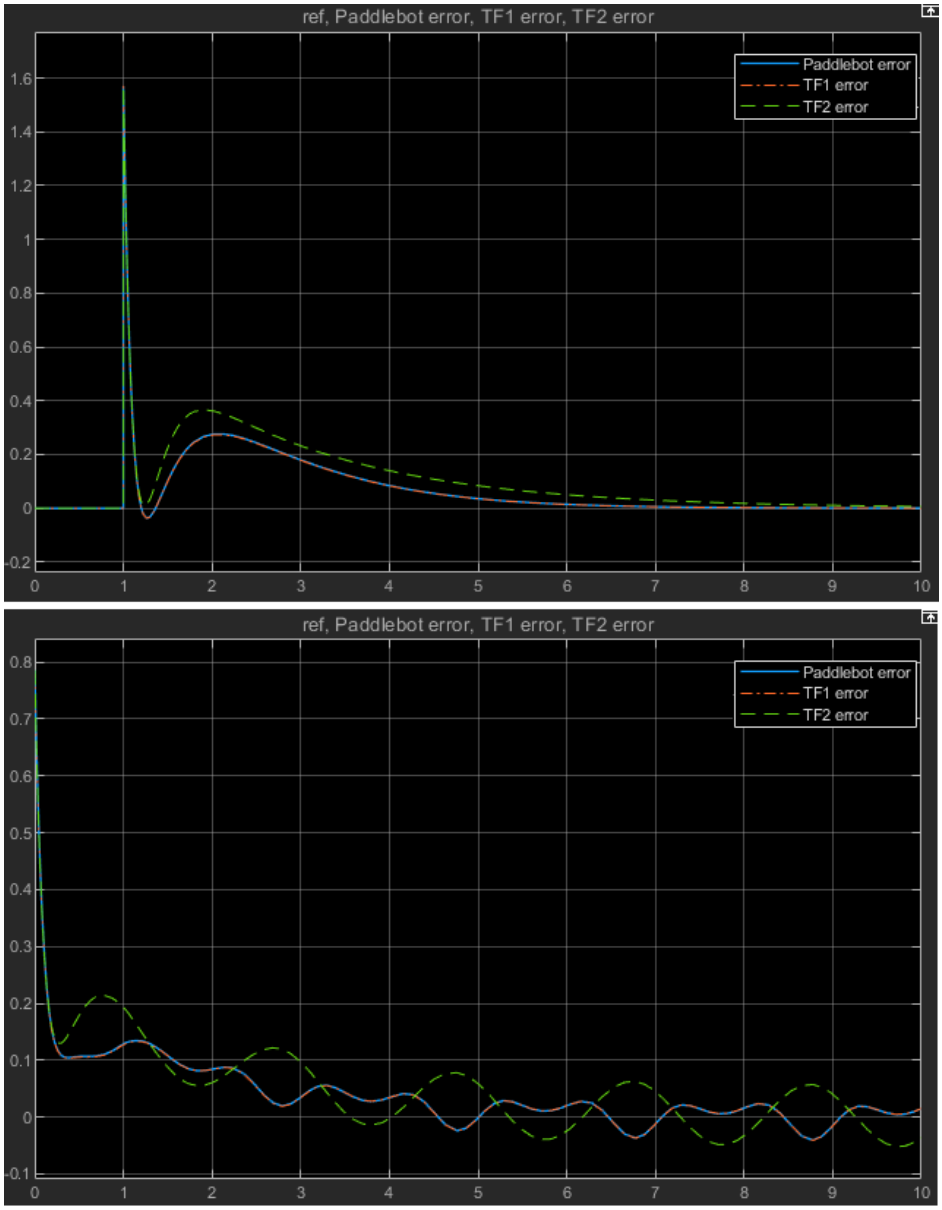

Figure 4.5. Comparison of error from the Simscape PaddleBot model (blue), nonlinear transfer function approximation TF1 (orange), and linearised transfer function approximation TF2 (green) for a step input (top) and sine input (bottom)

The purposes of creating these model approximations was originally inspired by a need to better understand the system response and how to modify the model within a simulation. These modifications could be reasoned as changes in the environment that affects output response from the system and would be better handled by adaptive controllers. The confirmation that the PaddleBot was not a linear transfer function is also beneficial in analysis as it can now be definitively stated that applying a gain to the input of the PaddleBot will not be equivalent to applying an equal gain to the output. This equivalence would be the case for linear systems but not for nonlinear systems, making control more difficult.

4.2.2. Stage 1 - Direct Reference Tracking via Reinforcement Learning

A basic reference tracking control loop measures a process variable and utilizes negative feedback to compare the measurement to a desired value represented by a reference signal. The error between these two values is used to determine the input signal to the system plant, allowing the plant output to reduce the aforementioned error signal, slowly converging to an acceptably small level, depending on the system and designer. This structure was used as the foundation for the Simulink model; the Simscape model acts as the system plant and returns a measured angle to subtract from the reference signal that represents the desired angle of the PaddleBot foot pedal. The produced error term is used in conjunction with angular velocity measurements and driving torque recordings as the foundational observations of an actor-critic reinforcement learning agent, as well as being the independent variables of the reward function. The trained agent is then able to transform the observations from the plant into a torque driving signal which feeds to the input of the plant. The Simulink model of this closed-loop system is provided below:

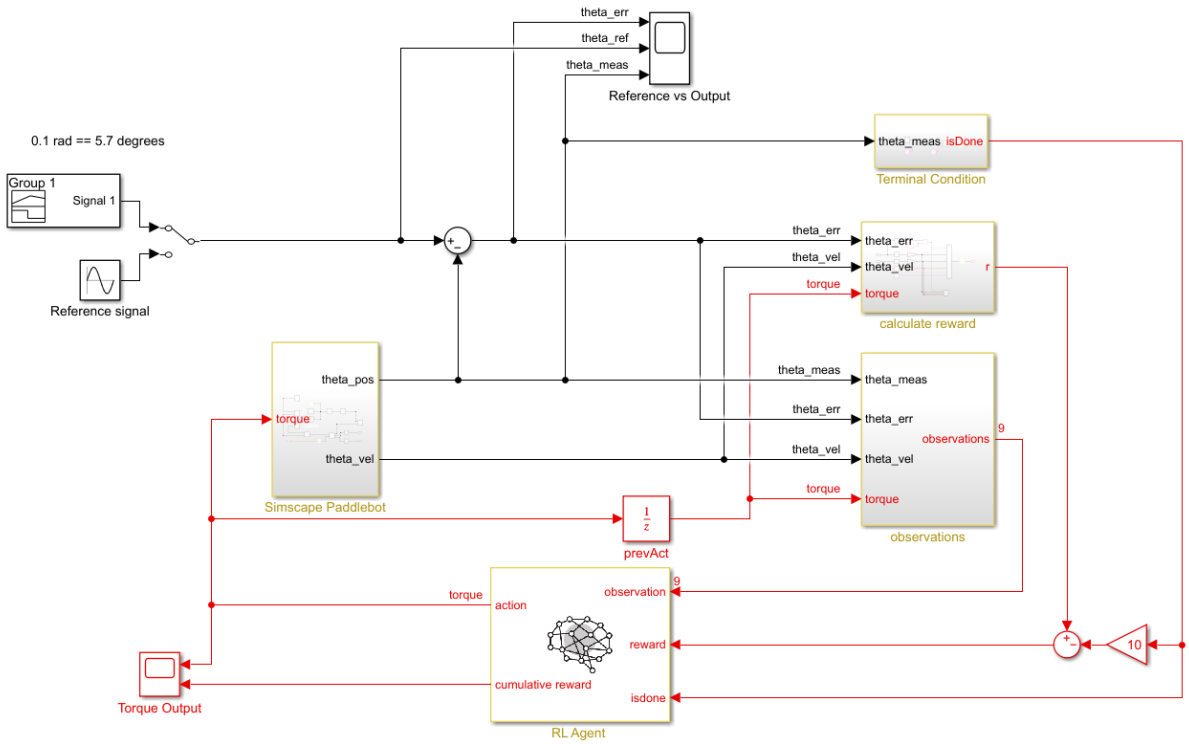

Figure 4.6. Simulink model for direct reference tracking utilizing the reinforcement learning agent "RL Agent" with 9 observations and 1 action

Traditional ankle range of motion for walking has an upper bound of around 30°, with a larger range of 70° for more strenuous activities (Brockett and Chapman 2016). Rehabilitation exercises should be used to help recover a full range of motion and as such should push the boundaries of each individual. For initial testing an input sine wave of amplitude 0.1 radians (≈5.7°) was used for training, as this stays well within the expected range of motion. Tracking a sine wave would result in a smooth and constant back-and-forward motion of the ankle, helping stretch the muscles. The other form of motion tracking that is desired is for step input values that essentially instruct the robot which angle it should currently rest at. This results in more abrupt motions that would be less common in ADL, but still have utility in rehabilitation. System response to both sine wave and step inputs were recorded for the purposes of analysis. As the RL agent is only able to control the torque applied to the PaddleBot and nothing else, changing torque to minimize the angular error is how the system functions. The potential actions the agent can perform can be limited to match real-world limitations or to improve safety. The torque signal was limited to a range of ±3 Nm for both safety concerns of the connected subject and the simulation of the existing equipment. Although the physical robot can produce higher torque values, a smaller limit was thought reasonable for initial testing and reaching small-amplitude reference signals as discussed earlier.

This first experiment acted predominantly as a learning experience and many aspects of the design may not be optimal, but achieve acceptable results nonetheless.

4.2.2.1.Observations

As observations act as the input to the RL neural network system with no memory, the observations at any given time sample are the only pieces of available data to the system to determine the appropriate action. As the action space of this system is the torque input to the PaddleBot, any measurements that may affect the input torque must be present in the observations. The most obvious observation is the error measurement between current angular position and desired angular position. Without this observation the system would have no way of identifying whether the goal was currently reached. From this point it is reasonable to expect the derivative and integral of the error to also be useful measurements to let the system know how fast the error is changing or how long an error has existed.

The measured angle and measured velocity are also used as observations despite covering redundant information; the angle measurement can, in theory, be calculated from the error observation if the reference signal is constant. However as this angle represents ankle angle it is important for the system to determine current angle for safety purposes. For instance, if a reference signal is misplaced at a potentially unsafe value (such as 180° from rest) the error signal will not be sufficient on its own to alert an issue. Including the direct measured angle as an observation, along with its derivative allows better responses to the system reaching the edge cases of acceptable operation. Measured velocity is redundant due to the derivative error observation, but is included nonetheless to structure the agent to use all available sensor readings. The measured angular velocity can be used to calculate angular acceleration through another derivative block to use as an observation, avoiding the necessity for a double derivative signal.

The final observations for the system are taken from the actions of the RL agent at a previous timestep, along with the discrete derivative to determine how the supplied torque values change over time. This allows the system to know what action led to the current states and will help determine which actions should be taken to reach desired states the system has not experienced beforehand.

In total, there are 9 observations used for this RL agent, however known redundancies are present and simplifications can be implemented to reduce the neural network complexity. These simplifications were not implemented as the development of this Simulink model was not of high priority, and the computational effort of the system training is not of key importance for this project. Future stages of the project involving MIMO systems will need to make these considerations more rigorously as computing power and training time become more restricting.

Table 4.1 – End-to-End Reinforcement Learning Experiment Observations

Name	Symbol	Description
Error	θ_e	Angular difference between desired angular value and measured angular value
Error Integral	$\int \theta_e \, dt$	Integral of angular error to determine duration of error across time
Error Derivative	$\dot{\theta}_e$	Derivative of angular error to determine rate of change of error across time
Measured Angle	θ	Measured angular value of PaddleBot
Meas. Angle Derivative	$\dot{\theta}$	Derivative of measured angle to determine rate of change of measured angle across time
Meas. Velocity	θ_v	Measured angular velocity of PaddleBot
Meas. Acceleration	$\dot{\theta}_v$	Derivative of measured angular velocity to predict PaddleBot angular accelerations
Previous Torque	τ_{t-1}	Torque value applied to the PaddleBot at previous time step
Torque Derivative	$\dot{\tau}_{t-1}$	Discrete derivative of torque value applied to PaddleBot to determine changes in applied torque across time

4.2.2.2.Reward function

Reward shaping is arguably the most important task in reinforcement learning, as the reward function is the sole guide for how the system determines what it is trying to achieve. A positive term in the reward function should be tied to any measurements that move the system towards more favourable states, while a negative term should be tied to any measurements that move the system towards unfavourable states. For reference tracking, the primary goal is to minimize the difference between the reference signal and the measured signal, so using the error value as a negative reward value will train the agent to minimize the error, in turn maximizing the returned reward. The scale of each of these terms must be tuned such that each term in the reward function is appropriately weighted to which measurement values are most important to system functionality. For rehabilitation purposes it is desirable to minimize the velocity in which the patient is required to move since fast or sudden motions may cause discomfort. However the user must still reasonably track the reference signal and as such the negative component related to velocity (and acceleration) will be weighted less than the error component. For similar reasons to velocity, the input torque to the robotic device should also be minimized for human safety concerns and the general reduction of energy used by the system.

The angle error, angle velocity, and input torque are the three measurement components used in the final reward function, although the error term was constructed as a piecewise function and two additional components were included to achieve an appropriate reference tracking system. One component is a constant positive reward of +0.1 to encourage further exploration and avoid early termination. The other component adds a penalty if the current position in state space is not worth exploring, again to avoid early termination. The latter component is related to the isdone function: a function that determines early episode termination during training and is discussed in a later subsection. The complete reward function is shown in (4.6) which describes the reward received at time instance t.

$$r_t = r_e - 0.01\theta_v^2 - 0.1\tau_{t-1}^2 + r_{isdone} + 0.1 \tag{4.6}$$

The reward associated with the measured error of the system is represented as r_e and takes the form of a piecewise function depending on if the measured error is less than a pre-determined threshold. By identifying whether the error value is less than a threshold and providing a reward for being close to the goal, the number of viable states for the system to aim towards increases dramatically and assists with the exploration of state space. For angular reference tracking, the threshold is set to 0.1 radians (≈5.7°). This value was selected as a 5° error in a real world scenario would be close enough for a subject to be considered close to their desired angle.

$$r_e = \begin{cases} 1 - 0.1\theta_e^2, & |\theta_e| \leq 0.1 \\ -0.1\theta_e^2, & |\theta_e| > 0.1 \end{cases} \tag{4.6}$$

The error value, measured in radians, has a scaling factor of 0.1 simply to keep cumulative reward values low and remain at a value comparable to the other penalty values. When also considering the error values will be measured in radians and have values less than 1, the error term, when squared, will be several orders of magnitude smaller than the constant +1 value received when within close proximity. This results in the system valuing low error very highly when compared to the remaining terms.

As discussed earlier, the velocity of the system is not one of the main concerns for the system and as such is scaled down further than the other terms. This results in the velocity of the system not dictating behaviour by much, but is still considered. The isdone reward function is a simple constant depending on the current angluar measurement. If the device has rotated over 90° in either direction, a large penalty is applied to singal this behaviour is extremely discouraged. This limitation is included from the fact that the human ankle cannot bend to such high degrees and the device may cause injury if these values occurred in actuality. The actual human range of

motion was not used to account for variations between users and prevent early termination occurring too frequently during training.

$$r_{isdone} = \begin{cases} 0\,, & |\theta| \leq 1.57 \\ -10, & |\theta| > 1.57 \end{cases} \quad (4.6)$$

The isdone function causes early termination of any given training episode if some condition is met during simulation, and as such this event must also result in a large reward penalty to discourage the system from terminating early on purpose. The cost of early termination must be large enough that the penalty accumulated across one episode is preferrable and exploration is encouraged.

4.2.2.3.Early episode termination

During training, every episode will run a simulation and attempt to maximize the reward by changing the actions it performs at each time stage. By trialling different actions from the action space, the state trajectory of the simulation will also change. For systems with continuous action space and state space there will be a large number of combinations that lead to infeasible or potentially dangerous states. If the system begins exploring these areas of state space that have no practical utility the system must identify this and stop simulation early. If this act is not done, the training of an agent will require potentially exponential increases in computing time and resources. Reducing the state space that is explored will improve training times, however it also eliminates many potential behaviours. Therefore these behaviours must only be eliminated if there is no scenario in which these states are desired or favourable.

For rehabilitation robotics, these behaviours will represent actions that cause harm to the subject. If the robot forces constant rotation in one direction, or any values above the specific user's range of motion, severe damage may be done to the ankle musculoskeletal system. For this reason, the early episode termination is set to trigger if the angle measured exceeds ±90° in either direction. Although this threshold is much too large for practical application, threshold changes can be easily implemented to be more in line with traditional ankle ranges, and even customised between individuals. If contact force between robot and human exceeds a threshold due to the mechanical resistance of the human, then free motion has been impeded and the device should no longer apply external force. This approach was not implemented for this experiment as it was not needed for simulation functionality.

4.2.2.4.Reinforcement Learning Algorithm & Agent Topology

The construction of the reinforcement learning agent depends heavily on the tasks that will be performed and the environment-generated interactions. With the need for a continuous action space for applied torques, as well as a deterministic policy, a deep deterministic policy gradient (DDPG) algorithm was implemented. Although twin delayed deep deterministic policy gradient (TD3PG) also fits the criteria it was not implemented as it is believed the environment simplicity did not warrant the additional computing resources. As DDPG is an actor-critic algorithm, the neural networks for both actor and critic must be designed manually to best suit the task at hand. The actor network takes the measured states (referred to as observations) as the input to the network and returns an action value, which in this case will be a singular scalar value to represent torque applied to the device.

The number of hidden layers within the actor NN, as well as the number of neurons per layer, were chosen by considering the mathematical theory of back-propagation neural networks as well as some less formal design tips. Determining the correct number of hidden layers has no correct solution, however with the knowledge that increasing the NN size will increase computation requirements and basic nonlinear functions can be estimated by a single hidden layer network, the number of hidden layers used for a relatively simple task should always remain less than 3, especially if relying on back propagation (Sutton and Barto 2018: 183). Traditionally only 1 hidden layer is sufficient, however 2 hidden layers were included in this experiment as the added computation time was not an issue. The number of neurons per hidden layer were chosen to fall between the number of input neurons and the number of output neurons, most commonly the mean. As the system contains 9 observations and only 1 action, a relatively small value of 6 neurons per hidden layer was selected. The construction of the critic network followed the same design procedure but with the input of the network including all observations and actions while the output produced a training value that adjusted and updated the actor network. The structure varied by having two paths that were eventually combined through elementwise summation, a technique employed by a provided MATLAB reinforcement learning example (Mathworks 2022e).

Some common issues to consider when designing a neural network include the vanishing gradient problem. As backpropagation of a NN updates weights through a calculation that utilizes the gradient of the following nodes, if a node has a small partial derivative valued between 0 and 1 then chaining through multiple hidden layers causes these derivatives to multiply and converge to 0. If this occurs the weights of the NN will no longer update and any

further training will see no improvements to performance. As DDPG is a policy gradient method the critic network is susceptible to this issue, however by utilizing appropriate activation functions such as the rectified linear unit (ReLU) and using a low number of hidden layers can minimize the potential effects of the vanishing gradient. Advantages of the leaky ReLU activation function were considered but considered unnecessary for this stage in the project.

The other potential problem that was considered when designing the network was overfitting or underfitting data. If a network does not have enough neurons then not all training data can be retained in the NN, with the most recent training data overwriting the existing values of the system. This problem is known as underfitting, where the network is ill-equipped to reliably fit the existing data to an appropriate output. Overfitting stems from the opposite end of the problem, where there are too many neurons in the system and all the training data is learned exactly. The goal of RL is to learn the general patterns contained within the training data, so learning the extreme details of a small set of data will not allow the system to identify broad strokes of unseen but similar data in the future. These potential problems are what directed the design choices for number of hidden layers and hidden neurons; computing time was another, albeit minimal, consideration. Training was performed on a personal home computer overnight, and as such the training time was irrelevant as long as it remained under 8 hours.

4.2.3. Stage 2 - PID Control Parameter Tuning via Reinforcement Learning

The design of the PID controller tuning agent was very similar to the direct reference tracking as they shared the objective of minimizing an error value between set-point and process variable. The difference arose in the action space of the agent, which instead of outputting a torque value directly, would output 3 time-varying coefficient values that would act as the gains for a Proportional-Integral-Derivative controller. By comparing Figure 4.6 to Figure 4.7 it becomes apparent the only difference between the control loops is the addition of the PID block, which acts as a controller to the Simscape PaddleBot plant. The effects of this controller in the original design can be considered as subsumed by the RL agent. However, by explicitly including and adjusting gain values for the controller the system is able to achieve its goals with more precision.

The training options were set to train the system for 2500 episodes, but an early termination criterion was included to reduce the effects of overfitting. This training option ended system training if the average reward value for the past 10 episodes exceeded a score of 20. As the

reward function predominantly revolved around negative rewards (penalties) if the system was able to return a positive value then tracking was being performed to a reasonable level that justified ending the training. The score of 20 was chosen arbitrarily to provide a small buffer from 0 to guarantee desired behaviours.

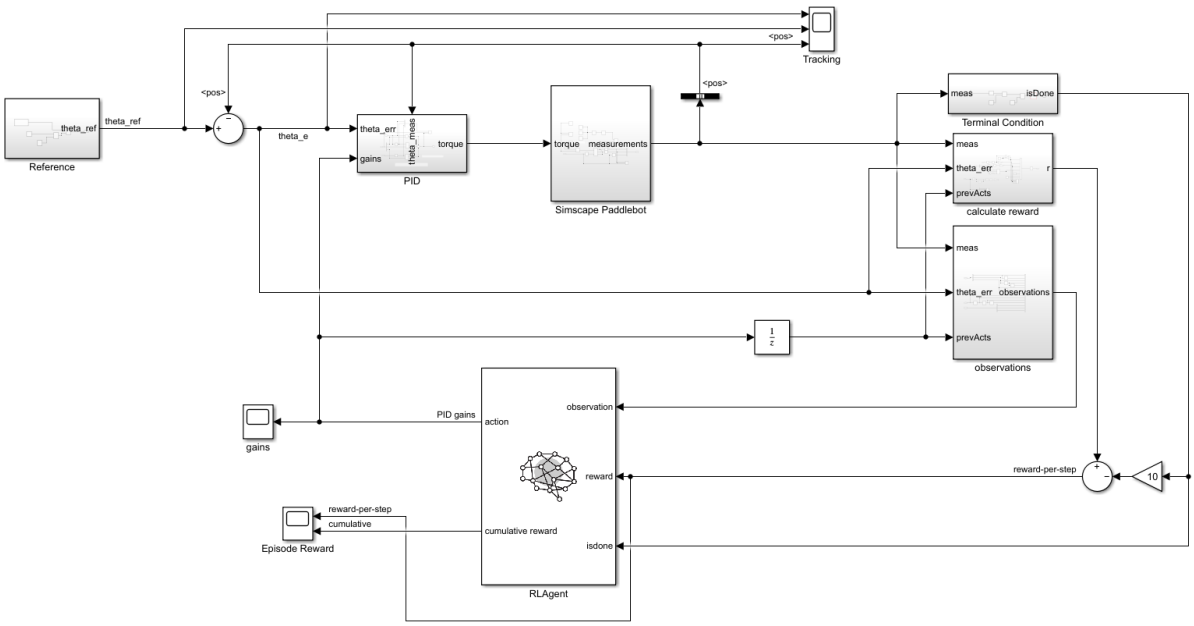

Figure 4.7. Simulink model for PID controller reference tracking utilizing the reinforcement learning agent "RL Agent" with 15 observations and 3 actions

Small changes were made to the training of this model when compared to the basic reference tracking model. The largest change was the inclusion of initial state variation, allowing the device to be better prepared for beginning simulations in non-optimized positions. This is an important feature to include before porting to the hardware, as there is no guarantee a human test will begin at exactly 0° and may have some inherent angle the system must compensate for before starting. This variation was set to randomly begin at any angle between -0.25 and +0.75 radians (-14.3° to +43.0°) with a uniform distribution. This range, with dorsiflexion described in the negative direction and plantarflexion described in the positive direction, was chosen for its approximate coverage of the traditional range of motion for a healthy ankle (Brockett and Chapman 2016). Small variations to the gravity vector directions were also made to remove a slight angle present in the original model, although this change is unlikely to have any measurable effect.

The original experiment used torque as the action and as such was easily restricted to a range of ±3 Nm. For the PID model using gains as the action space removed this ability, and indirectly restricting torque by restricting gain values was not possible due to the mathematical relation shown in (4.7), describing the mathematical structure of a PID controller. The actions produced

$$\tau = K_P * \theta_e + K_I * \int \theta_e \, dt + K_D * \frac{d}{dt}\theta_e \tag{4.7}$$

by the RL agent are the PID gain variables $K_P, K_I, \& \, K_D$, yet even when bound the torque output will be unbounded due to the dependence on the angular error terms and derivatives.

As a side note, the equation in (4.7) is for generic PID controllers but is not accurate for the used PID controller. For this experiment the derivative of the error signal ($d\theta_e/dt$) was replaced with the derivative of the measured signal ($d\theta/dt$) to avoid the step within the signal and causing drastically high derivative values. This substitution can be made as the error signal is calculated by $\theta_r - \theta$ and the derivative of the reference signal will equal zero, allowing the correct derivative to be calculated without what is known as "derivative kick".

Multiple different upper-bound values were tested, and the final maximum gain restriction was chosen to be 100, while the lower limit was set to 0. As an RL agent with continuous action space uses random variation in actions as the method of exploration, the standard deviation of this noise distribution must be mathematically related to the action range to appropriately explore all options. It is for this reason the action range of 100 was adopted, as when it was combined with the equation recommended within the MATLAB noise model documentation (see (4.9)), a reasonable standard deviation that was small enough to converge on local maximum reward values yet large enough to explore the action space was derived. The remaining changes to the observations and reward function were required to adapt to the new task and will be discussed below.

4.2.3.1.Observations

The basic principles to observation selection are used again for the PID reference tracking task. Variables that would likely directly affect or correlate to the error value should be included. The training reference signal was a multi-step function ranging from +0.4 to -0.2 radians, which caused derivative measurements to become unreasonably large. To avoid this issue, instead of using the derivative of any measurements, multiple time-instances of these measurements were taken. As the neural network has access to multiple time instances, as well as the sample time,

it is able to calculate the discrete derivative if necessary. The primary measurement, angle error, was measured at four consecutive time intervals. This allowed $\theta_e(t)$, $\theta_e(t-1)$, $\theta_e(t-2)$, and $\theta_e(t-3)$ to act as observations to the system.

The measured angle and was also taken as an observation for the same safety concerns as the previous experiment. As the measured angle is not as important as the angular error measurement it was only taken at two consecutive time intervals: $\theta(t)$ and $\theta(t-1)$. Measured angular velocity falls under the same category, however three instances were recorded: $\theta_v(t)$, $\theta_v(t-1)$, and $\theta_v(t-2)$. This is likely an unnecessary number of observations given the low impact that angular velocity measurements have on the reward function, but was simply chosen for ease of transition from a previous attempt iteration. As this observation was predicted to be the least relevant, it is expected any connections in the NN will have minimal weights and can be pruned in future iterations if computation requirements become a problem.

Previous torque measurements were taken for three consecutive time intervals as the change in torque values is an important consideration in robotics and their longevity. For the calculation of torque acceleration a minimum of three measurements are required: $\tau(t-1), \tau(t-2), \& \tau(t-3)$. Torque measurements start one time interval after the other measurements as it is considered the output of the feedback loop and therefore cannot be used to calculate itself without creating an algebraic loop.

Similarly, the actual output of the RL agent are the PID gain values which are included as observations. The values from the previous time instance, $K_P(t-1), K_I(t-1), \& K_D(t-1)$ are included simply to allow the agent to know the previous actions taken, which allows it to slowly learn which actions result in state transition patterns.

Table 4.2 – PID Reinforcement Learning Experiment Observations

Name	Symbol	Description
Error	$\theta_e(t)$	Angular error at time t
	$\theta_e(t-1)$	Angular error at time t-1
	$\theta_e(t-2)$	Angular error at time t-2
	$\theta_e(t-3)$	Angular error at time t-3
Meas. Angle	$\theta(t)$	Measured angle at time t
	$\theta(t-1)$	Measured angle at time t-1
Meas. Velocity	$\theta_v(t)$	Measured angular velocity at time t
	$\theta_v(t-1)$	Measured angular velocity at time t-1
	$\theta_v(t-2)$	Measured angular velocity at time t-2
Input Torque	$\tau(t-1)$	Motor input torque at time t-1
	$\tau(t-2)$	Motor input torque at time t-2
	$\tau(t-3)$	Motor input torque at time t-3
Previous Gains	$K_P(t-1)$	Proportional gain for PID controller at time t-1
	$K_I(t-1)$	Integral gain for PID controller at time t-1
	$K_D(t-1)$	Derivative gain for PID controller at time t-1

4.2.3.2.Reward Function

Constructing the reward function was also similar between experiments but with several changes made to better suit the agent setup, as well as incorporate newly developed techniques and experience. The reward function can be divided into five separate terms, each of which depend on one of the following key variables: angular error, angular velocity, applied torque, simulation runtime, and early termination status. The full equation is expressed in (4.8) with the piecewise error reward and the piecewise torque reward expressed in (4.8) and (4.8), respectively. The early termination penalty, r_{isdone}, remains unchanged from (4.6) (although the threshold measurement was converted from radians to degrees), applying a penalty if the measured angle reaches unsafe values for a human ankle.

$$r_t = r_e + r_\tau + r_{isdone} - |0.01\theta_v| + 0.5 \tag{4.8}$$

$$r_e = \begin{cases} 5 - |\theta_e|\,, & |\theta_e| \le 5° \\ -|\theta_e|, & |\theta_e| > 5° \end{cases} \tag{4.8}$$

$$r_\tau = \begin{cases} -|0.01\tau_{t-1}|\,, & |\tau_{t-1}| \le 3 \\ -|\tau_{t-1}|, & |\tau_{t-1}| > 3 \end{cases} \tag{4.8}$$

The most important change between the two experiments is the change from angular units being measured in radians to being measured in degrees. This change has the effective result of multiplying the reward values by a factor of 180/π, making the error reward and velocity reward contribute more to the overall value and increasing their importance to the system. From this change, any values that were squared were changed to absolute values to not over-penalise these states.

The error reward is strictly a penalty if the absolute error measurement is greater than a predefined threshold value, currently set at 5°. If absolute error measurement is less than this threshold then a positive reward is provided to encourage the system to reach and explore these states. The threshold value 5° was arbitrarily chosen as a value that could be considered "close enough" to reduce the exploration performed in unreasonable or redundant states.

The torque reward also acts as a strict penalty if the applied torque is greater than the 3 Newton-metre restriction previously established. As there is no way to create a hard limit of applied torque for this experiment, a soft limit is incorporated into the reward function. This piecewise function decreases torque penalty by a factor of 100 if this soft limit is not violated, and as such the RL agent will learn not to violate the limit unless the error penalty becomes so large that the system determines violating the soft limit to reduce the error penalty is mathematically practical. The real-world limitations of the motor-gearhead combination being utilized in the demonstration robotic device is a maximum continuous torque of 50 Nm which exceeds the healthy ankle dorsiflexion values for all demographics, but is less than values for plantarflexion (Moraux et al. 2013). For subjects over the age of 18 the weakest average ankle torque measurement was 19.2 Nm, produced by females between ages 60-69 for right-ankle dorsiflexion. To avoid any injuries to potential subjects the soft torque threshold for the robotic device should not exceed this value. Several tests were performed with a threshold torque $\tau_{th} = 20\ Nm$, however results did not differ greatly from the previous threshold value $\tau_{th} = 3\ Nm$, so findings from the initial value are hereby referenced.

4.2.3.3.Early Episode Termination

Early episode termination is identical to the previous experiment setup, triggering an end to an episode if the PaddleBot rotates to a state that would cause harm to the participant. Current threshold is set to ±90° for simplicity. Although the range of motion for a healthy ankle is notably more narrow than this range, it is undesirable to cause episode termination extremely frequently, as this will likely cause the training to fall into a local minima and not explore further options.

4.2.3.4.Reinforcement Learning Algorithm & Agent Topology

With a continuous action space for gain values, a TD3PG algorithm was used. This neural network structure is identical to the DDPG structure implemented within the basic reference tracking system, with the notable exception that two critic networks are trained side-by-side and the more conservative update is used for each learning iteration. The learning rate for the actor network and both critic networks remained at 10^{-3}. Tests were performed with all learning rates changed to 10^{-4}, however the results showed less convergence towards an optimal policy, possibly explained by the system falling into a local minima and settling for less optimal results. As the agent is expected to value correct tracking at all stages of the simulation and the simulation is relatively short, the discount factor that reduces the value of late-stage rewards is set to 1. This results in no reward discounts for rewards that happen in the distant future, but this setting is unlikely to affect the convergence of the training due to each episode lasting at maximum 100 time instances.

For a discrete action space system an RL agent would randomly select one of the finite actions that could be taken and analyse the resulting reward accumulated. As this system is required to output a continuous action, randomly selecting a finite action is not possible due to the infinite degree of refinement that may be taken by the system. Instead, a random action is taken and is then compared to the same action taken with an additional noise signal to slightly perturb the action taken. It can then be determined if this action returned better or worse reward values over the episode and slowly adjust the actions taken in the future accordingly. The exploration of the continuous space is achieved with random Gaussian noise combined with the learned action values. The standard deviation of this noise will be proportional to the action range of the system to guarantee the action range is explored to a reasonable degree throughout training. This proportionality is recommended to be between 1% and 10% of the full action range (Mathworks 2022c). The formula for the exploration noise standard deviation (σ^2) is dependent on system sampling time (T_s) and the action space range (A_{lower}, A_{upper}):

$$\sigma_{EN} = \frac{\alpha * (A_{upper} - A_{lower})}{\sqrt{T_s}} \tag{4.9}$$

where α is the scaling factor to be chosen between 0.01 and 0.1, depending on how much exploration is favoured in the training process. For this experiment, α was set to 0.05 to appropriately explore the action range. A small noise decay rate of 0.0001 was present across the training session but unlikely to have caused any noticeable noise variation within the limited training session.

For the agent's network architecture, the number of hidden layers remained the same: 2 hidden layers for the actor network, 2 for the observation path of the critic network, and 1 for the action path of the critic network. The number of neurons per hidden layer remained equal to the average of number of actions and number of observations; with 15 observations and 3 actions, each hidden layer contained 9 neurons. The topology of each neural network can be seen in Figure 4.8. The addition layer for the critic network symbolises elementwise summation via colour coded connections.

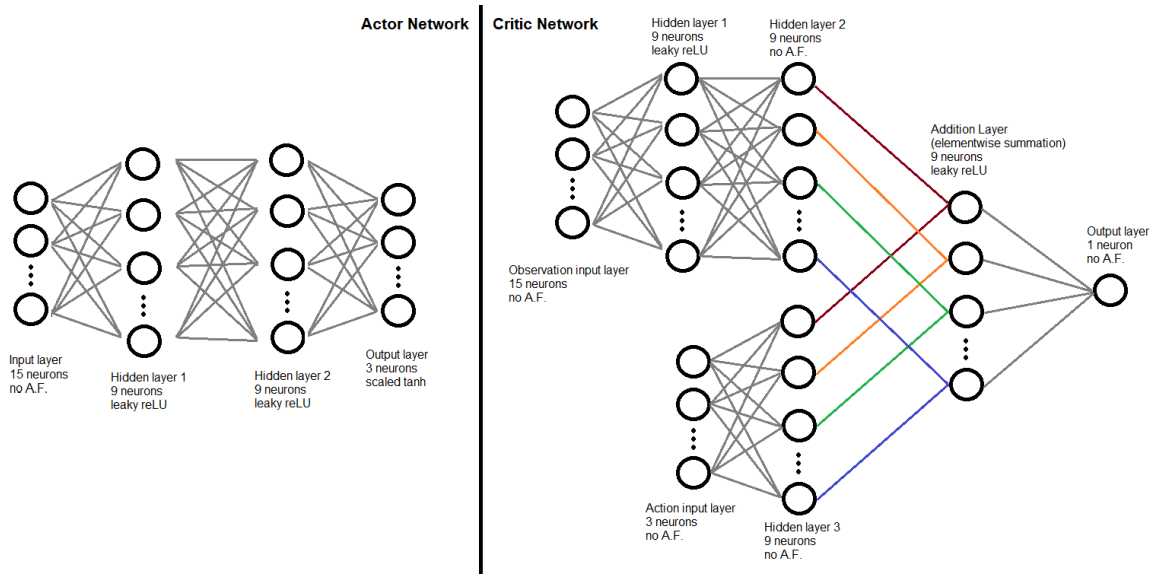

Figure 4.8. Neural network architecture for the PID reinforcement learning agent with number of neurons and activation functions labelled for each layer. Layers with no activation function are labelled as "no A.F.". The actor network is displayed on the left and the critic network is displayed on the right

Leaky ReLU activation functions were used for the hidden layers with a scaling value of 0.1. This change allows negative values to be passed through the network without being zeroed, but reduces their magnitude by a factor of 10 each node. By using this activation function the effects of the vanishing gradient problem are reduced, albeit slightly.

4.2.4. Stage 3 - Admittance Control Parameter Tuning via Reinforcement Learning

The final experiment utilizing single-input single-output systems was structured nearly identically to the PID control parameter tuning experiment, however the control loop of the system was expanded to better represent the desired functionality. A simulated external environment was added after the PaddleBot plant to determine the interaction force between the robot and any objects that may impede its motion. This environment is intended to

represent the human user that moves their ankle in tandem with the PaddleBot. As such, the force experienced will be determined by whether the robot is still able to freely move through space, or if the ankle range of motion has been reached and a resistive force is encountered. An admittance controller was implemented along the feedback line to convert the interaction force into an adjusting position signal to prevent the system tracking a reference signal that would cause harm to the user. The admittance controller uses the previous RL techniques to adaptively set the controller parameters. Figure 4.9 shows the new control loop including the simulated environment and adaptive admittance controller. A simplified diagram of the system is shown in Figure 4.10 to help clarify variable descriptions and show how the admittance controller will adjust the signal the system converges to, from the original reference signal (*theta_r*) to the new command signal (*theta_c*). The distinction between these two signals is important as there are now two error signals that will determine system behaviour. The command error is what will determine the distance from the position goal. The reference error is still an important measurement in admittance control, as it dictates the environment interaction forces and can be set as needed to achieve a pre-desired interaction force during steady state operation.

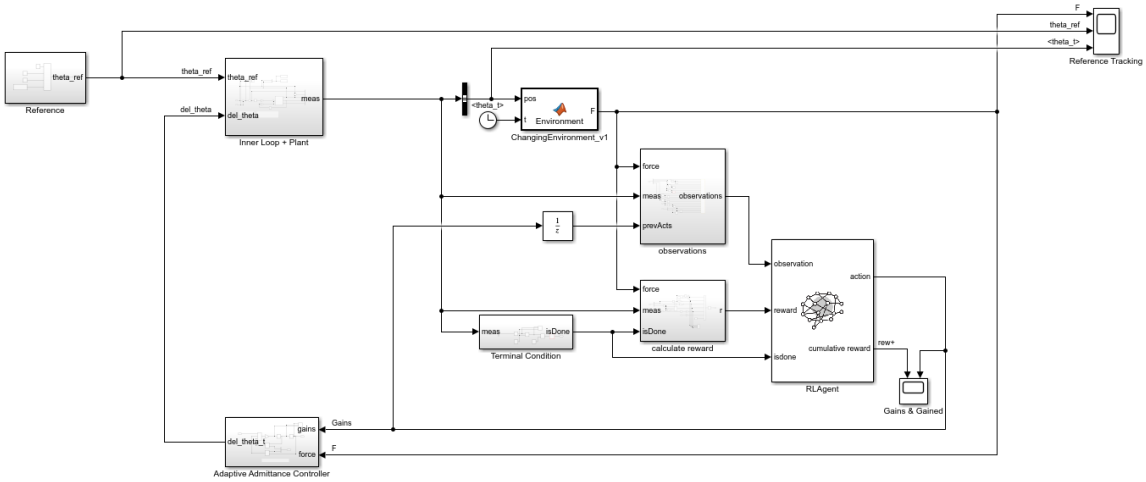

Figure 4.9. Control loop utilizing reinforcement learning for an adaptive admittance controller interacting with a changing environment

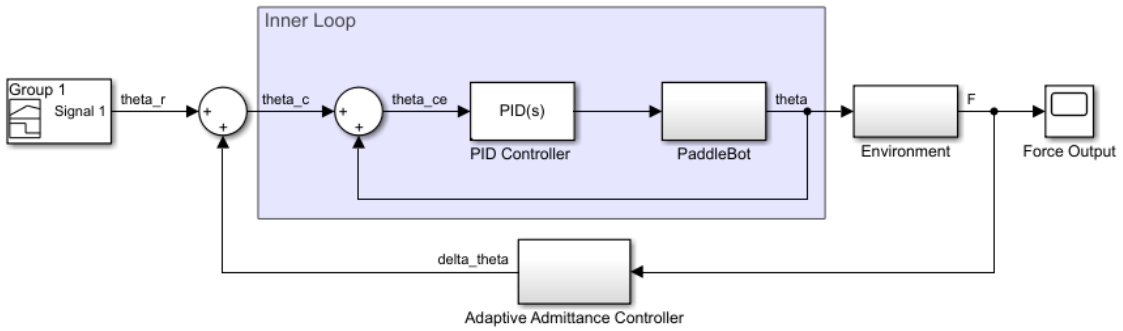

Figure 4.10. Simplified diagram of adaptive admittance control loop implemented. The adaptive admittance controller is determined by the reinforcement learning agent

The adaptive PID controller from stage 2 was replaced with a constant PD controller. The integral component of the controller was removed due to the PaddleBot plant acting as a double integrator; a double integrator following a PID controller can lead to a reduction in the phase stability margin and may cause the system to become non-minimum phase. The gain values were chosen through trial and error for a system that provided a rise time of approximately 1 second for each step change. These values were set to $K_p = 75, K_d = 15$ for majority of tests. The adaptive element of the control loop was transferred to the admittance controller, whose parameters represent physical properties of the controller: inertial mass (M), damping (B), and stiffness (K). The admittance controller takes the measured environment interaction force and returns a trajectory correction signal ($\Delta\theta$). This signal is added to the reference signal (θ_r) which results in the new command signal (θ_c). The command signal becomes the input to the inner loop and acts as the desired value the system converges towards in the same role the reference signal filled in the previous experiment. The error signal is therefore calculated by the difference between the command signal and the measured signal, although it is labelled as the command error for clarity (θ_{ce}), as seen below:

$$\Delta\theta = \theta_c - \theta_r \tag{4.10}$$

$$\theta_{ce} = \theta_c - \theta \tag{4.11}$$

The trajectory correction signal is generated by the admittance controller which follows the equation of motion discussed in (2.4). As the system achieves tracking and the command error signal converges to zero, the correction signal will converge to the following value:

$$\lim_{\theta_{ce} \to 0} \Delta\theta = \theta - \theta_r \tag{4.12}$$

Assuming the system has perfect tracking then combining (2.4) and (4.12) will allow the impedance behaviour of the system to match the mathematical model provided, which can be represented by (4.13). This equation can be rearranged for calculating the correction signal double derivative which can then be integrated for calculating the output of the admittance controller and determining the appropriate trajectory correction $\Delta\theta$.

$$\mathrm{F(t)} = M\Delta\ddot{\theta} + B\Delta\dot{\theta} + K\Delta\theta \tag{4.13}$$

$$\Delta\ddot{\theta} = M^{-1}(F(t) - B\Delta\dot{\theta} - K\Delta\theta) \tag{4.14}$$

There are three parameters in total that can be adjusted to control system impedance, and as such the action vector from the RL agent must contain 3 values. However it has been shown that changes to the inertial mass can lead to negative impacts on system stability. Varying M values lead to rapid changes in the damping ratio and natural frequency of the system (Bingjing et al. 2019). It was also revealed that changes to M, when compared to changes to B or K, had a relatively small impact on the resulting correction signal. As such, it was decided that the inertial coefficient would be excluded from the adjustable parameters, focusing only on stiffness and damping. This advice has been adopted for the previously specified reasons, along with the benefit of reducing the dimensions of the action vector of the reinforcement learning agent from 3 to 2. It is unclear how the inertial mass parameter was calculated by Bingjing et al., and as such system inertia was set to equal 1 for simplicity of calculations.

The action range was set such that each action would be a continuous value bound between 5 and 150. The upper limit was implemented to prevent searching an infinite space, and higher stiffness values lead to larger interaction forces so the maximum value of 150 will restrict these forces to reasonable levels for human experiences. The lower limit was implemented as to match the physical meanings of the gain values (a negative spring constant or damping coefficient is physically nonsensical). The lower limit of 5 was selected as previous experiments with the limit of 0 led to scenarios where tracking error would not diminish over time when transitioning from out-of-range operation to in-range operation. Having a positive gain value for both stiffness and damping at all times will prevent this behaviour and will cause steady state error to decrease over time.

4.2.4.1.Observations

The observations used for this experiment followed the same principles as the former experiment, taking measured angle, angular velocity, reference error, and input torque at three

adjacent time intervals so the neural network was able to observe how the system evolved over time based on the actions taken. Additionally, the angular error between the current position and the admittance controller-adjusted signal (the command signal) and the interaction forces were also included as observations to allow the environment interaction effects to also be considered during operation. An additional time instance of the previous gain values were also included for each parameter, to ideally assist in identifying the effects of changing the action output at any given time.

Table 4.3 – Adaptive Admittance Reinforcement Learning Experiment Observations

Name	Symbol	Description
Command Error	$\theta_{ce}(t)$	Difference between θ_c and θ at time t
	$\theta_{ce}(t-1)$	Command error at time t-1
	$\theta_{ce}(t-2)$	Command error at time t-2
Reference Error	$\theta_e(t)$	Difference between θ_r and θ at time t
	$\theta_e(t-1)$	Reference error at time t-1
	$\theta_e(t-2)$	Reference error at time t-2
Meas. Angle	$\theta(t)$	Measured angular displacement of plant at time t
	$\theta(t-1)$	Measured angle at time t-1
	$\theta(t-2)$	Measured angle at time t-1
Meas. Velocity	$\theta_v(t)$	Measured angular velocity of plant at time t
	$\theta_v(t-1)$	Measured velocity at time t-1
	$\theta_v(t-1)$	Measured velocity at time t-2
Input Torque	$\tau(t-1)$	Input torque of plant at time t-1
	$\tau(t-2)$	Input torque at time t-2
	$\tau(t-3)$	Input torque at time t-3
Interaction Force	$F(t)$	Interaction force between plant and environment at time t
	$F(t-1)$	Interaction force at time t-1
	$F(t-2)$	Interaction force at time t-2
Previous Gains	$K(t-1)$	Admittance controller stiffness gain at time t-1
	$K(t-2)$	Admittance controller stiffness gain at time t-2
	$B(t-1)$	Admittance controller damping gain at time t-1
	$B(t-2)$	Admittance controller damping gain at time t-2

4.2.4.2.Reward Function

The reward function also follows the same principles as the previous experiments but has adjusted values to appropriately balance each reward term. The error penalty remained unchanged and still provides a positive reward if current position is close to the desired position, however due to the interactions with the environment preventing the plant from tracking the reference signal in all scenarios, an error penalty will accumulate even in cases where the system is behaving exactly as desired, signalling that the reward function has been poorly constructed. Attempts were made where the error penalty was dependent on the command error (θ_{ce}) rather than the reference error (θ_e), however this led to a "chasing a moving target" problem where the agent learned to set the command signal to easy-to-track states as it was able to control the experienced forces through the actions. The results no longer resembled reference tracking and as such this approach was abandoned and reference error was used in the reward calculation, as it was for the adaptive PID controller experiment. Although meaningless, these results are presented below in Figure 4.11 as they are interesting and show the quirk of reinforcement learning behaving in a manner not intended but perfectly matching the specified reward function.

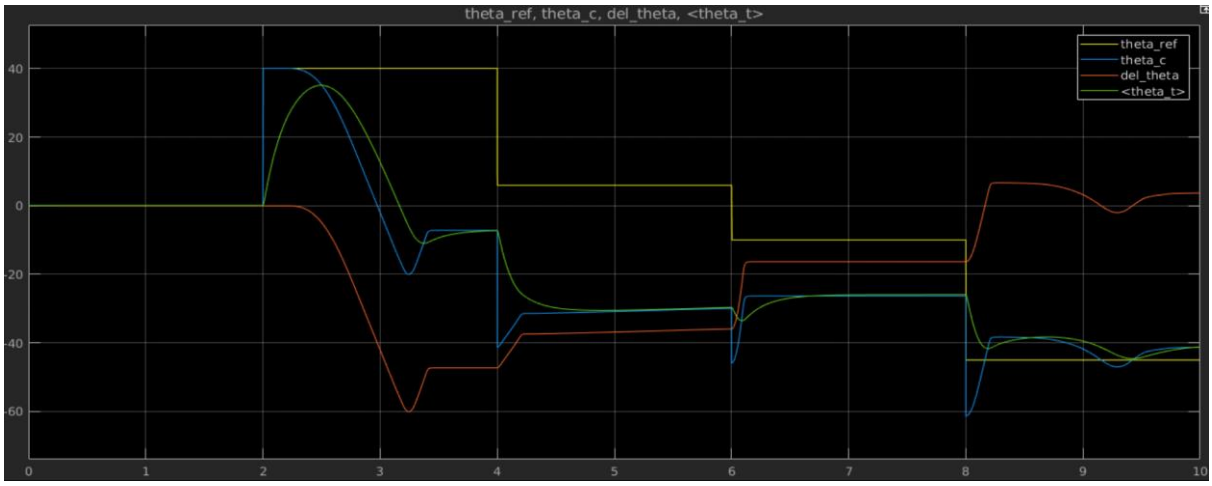

Figure 4.11. Reference tracking ability of RL agent when the reward function aims to minimise error between measured angle and command signal rather than measured angle and reference angle

Punishing error between the reference and the measured signal will lead to large reward penalties if the reference is beyond the environment-restricted range of motion. To discourage perfectly tracking the reference signal, the interaction force penalty that is experienced when outside the range of motion must supersede the accumulated error penalty. As the environment is set to the equivalent of a spring with a stiffness of 20 N/degree, the forces generate increase linearly for every degree the PaddleBot passes the range of motion. This leads to fairly large

force measurements that will easily dominate the error penalty term, even when multiplied by the constant factor of 0.2 to prevent the term from dominating all others. The velocity penalty was increased by a factor of 10 (from 0.01 to 0.1) due to the addition of new terms effectively drowning out the previously miniscule value. Despite this change, velocity is still the smallest contributor to the overall reward.

The threshold for torque penalty was increased from 3 Nm to 20 Nm. This change is to more accurately represent the torque that would be acceptably applied to a human ankle. Although most ankles are able to generate much higher torque values it was determined by Moraux et al. (2013) that this torque value varies drastically between age and demographics. For this reason the lowest torque value was chosen as to guarantee no individual would be subjected to a torque that their muscles would be incapable of producing on their own. This threshold is not a hard limit for the system, but simply reduces the experienced penalty by a factor of 100 if the applied torque remains lower. This should, in theory, lead the agent to avoid providing actions that would lead to excessive torque under ideal scenarios. Since this is not a hard limit, the torque applications must be watched carefully for safety concerns. The early episode termination penalty is triggered under the same conditions and was similarly amplified by a magnitude of 10 to prevent the term from being made insignificant in the presence of all the other rewards and penalties combined.

The force equation mentioned earlier is calculated according to the following equation:

$$F_{env} = \begin{cases} 20(\theta_l - \theta), & \theta \leq \theta_l \\ 0 \quad , & \theta_l \leq \theta \leq \theta_u \\ 20(\theta - \theta_u), & \theta \geq \theta_u \end{cases} \tag{4.15}$$

For convention, the PaddleBot rotating in the plantarflexion direction (bending foot away from tibia) is designated the negative and the dorsiflexion direction (bending foot towards tibia) is designated the positive direction. For the example a user was imagined with a maximum dorsiflexion of 30° and a maximum plantarflexion of 45°. Therefore the lower and upper bounds for the range of motion were set to $\theta_l = -40°$ and $\theta_u = +25°$, respectively. The 5° discrepancy was included as a buffer to try and prevent injury and increase response time for the command signal reacting to the environment. A positive force corresponds to pushing the PaddleBot in the dorsiflexion direction and negative force pushes in the plantarflexion direction, so the experienced forces will resist motion outside of the pre-determined range. Although this force calculation is not an exact simulation of real world physical impedance and there may be issues

in safety for sudden changes between conditions, these issues are reserved for future system improvements as the main focus of the experiment is the adaptive admittance control.

From the environment-force interaction equation from (4.15), it can be seen that while the system is within the acceptable range of motion there are no interaction forces. This is not true in practice as there will be interaction forces between the human foot and the robot whenever either body is attempting to move. This interaction force will likely be minimal and necessary for correct operation, so it was ignored for the purposes of the reward function. Every degree past the range of motion increases the resistive force by 20 N to make violating the range of motion very difficult.

The reward received at every time interval t is calculated through the following reward function:

$$r_t = r_e + r_\tau - 0.2|F_{env}| - 0.1|\theta_v| + 1 + r_{isdone} \tag{4.16}$$

$$r_e = \begin{cases} 5 - |\theta_e|\,, & |\theta_e| \leq 5° \\ -|\theta_e|, & |\theta_e| > 5° \end{cases} \tag{4.16}$$

$$r_\tau = \begin{cases} -0.01|\tau_{t-1}|\,, & |\tau_{t-1}| \leq 20 \\ -|\tau_{t-1}|, & |\tau_{t-1}| > 20 \end{cases} \tag{4.16b}$$

$$r_{isdone} = \begin{cases} 0\,, & |\theta| \leq 90° \\ -100, & |\theta| > 90° \end{cases} \tag{4.16c}$$

4.2.4.3.Early Episode Termination

Early episode termination is identical to the previous experiment setup, triggering an end to an episode if the PaddleBot rotates to a state that would cause harm to the participant. Current threshold is set to ±90° for simplicity. Some experiments used a threshold of ±60° to be slightly more realistic in physical restrictions, but no noticeable difference to training was apparent.

4.2.4.4.Reinforcement learning algorithm and agent topology

Training an adaptive admittance controller uses very similar methodology to training the adaptive PID controller. The TD3PG algorithm was used for the production of a continuous action space and previously successful results. Learning rates for all networks initially remained at 10^{-3} due to the system similarities and the previously successful training results, however some tests were performed with a learning rate of 10^{-4} as the training progress indicated the system was converging to a solution very quickly and a smaller learning rate may help find an alternate solution. The discount factor was changed from 1 to 0.99 to discount later rewards.

This change was made due to a recommendation from the OpenAI Spinning Up tutorial (OpenAI 2018) which states that a non-unity discount factor increases the likelihood of convergence, stemming from the guarantee that the reward sum will converge. Conceptually, the reward discount factor can be justified with the understanding that rewards at the beginning of the simulation are guaranteed, but later rewards may not happen due to early termination. Noise exploration was calculated using the same formula of (4.9) but with $\alpha = 0.08$ as it was believed more exploration was necessary to find the optimal gain values. Lower exploration was tested as manual parameter tuning showed low gain values were best suited in specific cases of environmental interaction, but high values were best suited for other cases so the higher exploration noise was eventually implemented. The experience buffer size was also increased from 10,000 to 100,000 to retain more experiences throughout training. The full training time was set to 4000 episodes which each generate 100 experiences, so setting the experience buffer to preserve 25% of all experiences at any one time seemed like a reasonable ratio to both learn from the training and avoid catastrophic forgetting of previously learned behaviours.

Neural network topology for the actor network was retained between experiments, with the only change being the number of neurons for each hidden layer. The first hidden layer utilised 16 neurons, while the second hidden layer only used 8 neurons to try and avoid overfitting that had occurred in previous stages of development. The critic network was reshaped to be more symmetrical and allow both observations and actions to parse through the same number of neuron layers. Figure 4.12 shows the unedited structure of the network.

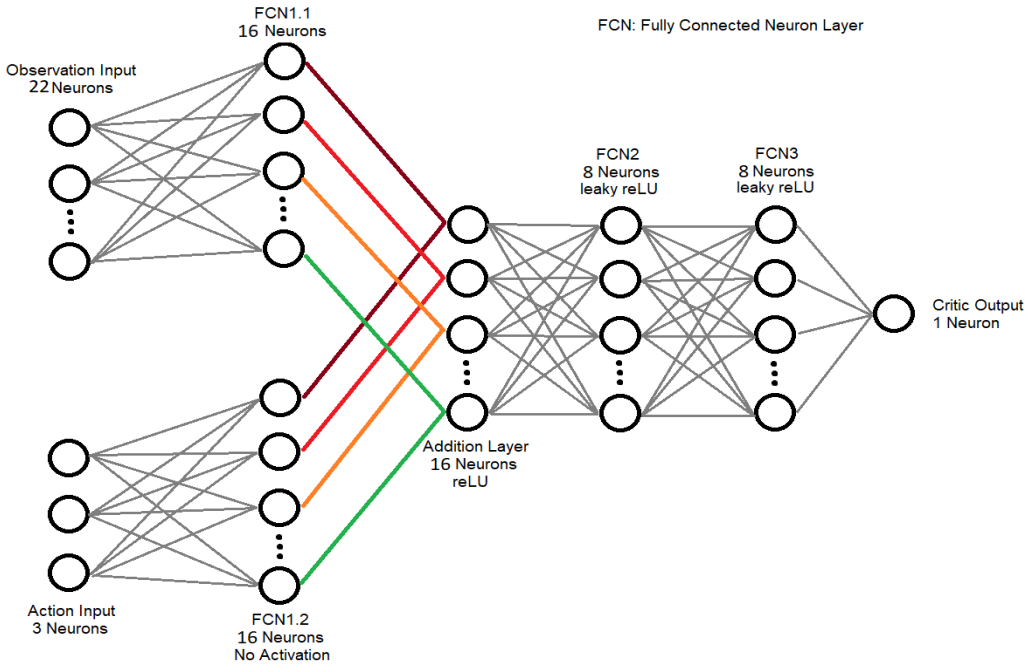

Figure 4.12. Neural network architecture for the Admittance reinforcement learning agent critic network with number of neurons and activation functions labelled for each layer. Layers with no activation function are labelled as "no Activation". The actor network architecture was unchanged from Figure 4.8, but with 22 input neurons, 16 hidden neurons/layer, and 2 output neurons

Note: This alternate architecture was trained and tested but showed worse RMSE scores than the original architecture, so the original architecture for the adaptive PID controller was utilised for calculating the RMSE values presented later in this chapter. The new architecture was left in this report to show alternate architecture attempts were made.

4.3. Results

The ability to track a reference signal for a system was measured through the calculation of the error between the reference signal (also referred to as the set point) and the measured signal (also referred to as the process variable) at any given time. To compare reference tracking systems an average score across one full simulation is required. The root mean square error (RMSE) can be used for this purpose as it will represent how large the error measurements were and how long they persisted in the system.

$$RMSE = \left(\frac{1}{T}\sum_{i=0}^{T}(r_i - y_i)^2\right)^{\frac{1}{2}} \tag{4.17}$$

For a single experimental simulation, the error at every discrete time interval can be calculated to find the average error for the simulation. Each simulation has an equal simulation time T_f and sampling time T_s. The total number of samples can then be calculated as $T = T_f/T_s$. In this experiment the simulation time was set to 10s and the sampling time to 0.1s. Any large divergences caused by overshoot or undershoot will increase the RMSE calculation, which will then be used as the comparison between reference tracking methods, as well as a measurement of how each method functions with various input signals. The three most common input signal types include the step, the impulse, and the sinusoidal wave. No impulse input tests were performed as they do not have as much practical application in rehabilitation. Each input signal can be tested to calculate the RMSE to determine if the controller is able to reliably direct the physical plant to behave in the desired motion patterns. Three different input signals were tested for each stage of the experiments:

1. A basic sine wave with amplitude of 0.1 (≈5.7°), a frequency of 3 rad/sec, and a bias and a phase of 0. The input signal was labelled “Basic Sine”
2. An offset sine wave with an amplitude of 0.25 (≈14.3°), frequency of 2π rad/sec (exactly 1 Hz), a bias of -0.15 (≈8.6°), and a phase of 0. This input was structured to crudely represent a regular healthy walking gait and was labelled as “Gait Sine”
3. A multi-step function equal to: 0 from t=0 to t=2, 40° from t=2 to t=4, 6° from t=4 to t=6, -10° from t=6 to t=8, and -45° from t=8 to t=10; labelled “Multi-step”

Two sine wave inputs were tested as the general range of motion for an ankle joint can vary dramatically between humans, especially those suffering from disability, so having data to compare under the lens of frequency analysis or amplitude analysis may lead to better understanding for future user’s preferred motion patterns.

The primary goal of this chapter was to use artificial intelligence to develop a controller. Therefor to justify this approach, the produced adaptive PID controller must surpass the traditional PID controller, or the adaptive admittance controller must surpass the traditional admittance controller. The admittance comparison will not be as clear-cut as the PID case and revealed through RMSE recordings, as perfect tracking is not intended behaviour and additional restraints must be formed to incorporate important information such as range of motion violations and environmental interactions. Excluding stage 1, the RL agent was trained two separate times: once using a multi-step input signal (hereby referred to as the step-trained adaptive system), and once with the gait sine wave (hereby referred to as the gait-trained adaptive system). Each agent is expected to have higher RMSE recordings for input signals that differ from its training

signal as the agent has not experienced these exact situations. However, as long as each control loop is compared to its counterpart with the same input signal, this issue is not believed to be negatively impactful and can be accounted for easily in the future. Varying the input signal while using agents trained with alternate input signals also has the benefit of showing whether the adaptive controllers can learn from one specific input and adjust for alternative input signal forms, as would be the case for real-world motion tracking and rehabilitation exercises.

Note: All RMSE measurements are taken in degrees rather than radians to amplify the differences in measurements, as values between experiments were very similar in many cases.

4.3.1. Stage 1 - Direct Reference Tracking (End-to-end)

The original reference tracking agent constructed does not employ any form of controller, and instead contains the controller within the RL agent to output the applied torque value directly. For this reason, there is no comparable control loop that comparisons can be made in reference to. The reference tracking ability can be seen in Figure 4.13 for a basic sine wave, a biased sine wave, and a multi-step function.

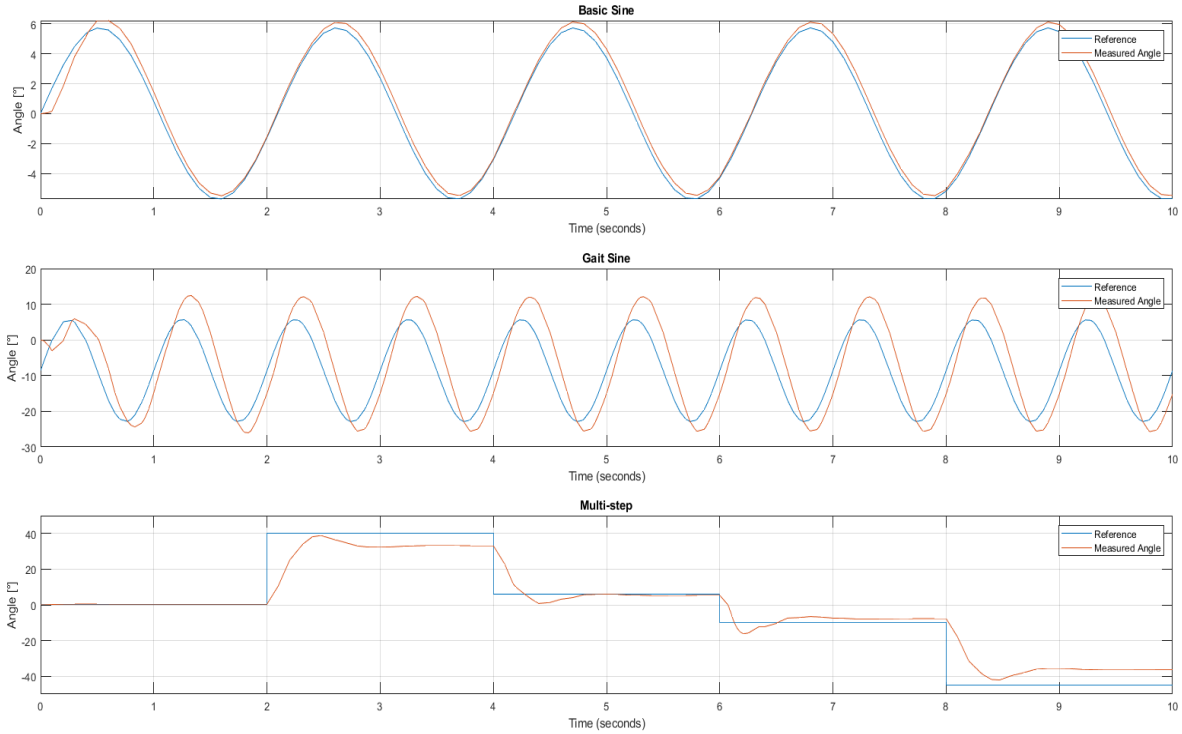

Figure 4.13. Response output from the RL direct controller for the: basic sine wave (top) gait sine wave (middle) and multi-step function (bottom) as the system input signal. RMSE values of Basic, Gait, and Multistep are: 0.454, 6.597, and 9.105, respectively.

Tracking for the basic sine wave shows small error at all points of the simulation, but the gait sine wave shows degeneration of tracking with a larger amplitude (6° for the basic sine versus 14° for the gait sine) and higher frequency (3 rad/s, or 0.48 Hz for the basic sine versus 2π rad/s, or 1 Hz for the gait sine). Naturally a reference tracking system involving larger numbers will have larger absolute error values that are used to calculate the RMSE, so it is important to determine whether the increase in error can be attributed to the larger input signal amplitude or whether it is simply the result of the RL agent training. This concept is covered deeper for the adaptive PID stage in section 4.3.2.3, but for this experiment the RMSE values for the basic sine wave were clearly affected by both frequency and amplitude. Holding amplitude at 0.1 radians, increasing the frequency from 3 rad/sec to 2π rad/sec caused the RMSE value to increase from 0.4540 to 2.6951. Holding frequency at 3 rad/sec, increasing the amplitude from 0.1 to 0.25 caused the RMSE value to increase from 0.4540 to 0.8230. These results suggest most errors present are caused by the physical limitations of the PaddleBot and its inertia causing a delay in reference signal tracking for faster movements.

The step-function tracking also behaves reasonably, with a clear trade-off between overshoot and steady state error occurring at each step. The first step at t=2 and forth step at t=8 show no overshoot but leaves a non-negligible steady state error, while the second step at t=4 and third step at t=6 have much less of a steady state error at the cost of overshoot. Since there is no distinguishable difference between types of error within the reward function there is no reason for either approach to be favoured by the RL agent other than which tends to produce less error and torque penalties. The agent was trained using a slight variation of the multi-step function, with only 3 steps rather than 4. The training steps after t=4 were held for 3 seconds which may have biased the agent to learn to favour steady state error at earlier times as it expects them to last less time. Although possible, the steady state error for the final step seems to contradict this hypothesis and from visual analysis it seems the steady state error should always be considered the worse of the two options. Another possibility the results may suggest is the difference comes from the magnitude of the step, with larger steps requiring more torque that is discouraged, and so a steady state error is produced instead. The step drop between step 1 and step 2 is 34° while the drop between step 3 and 4 is 35°, so although the step amplitude may play a role it is also unlikely to be the sole contributing factor. The final possibility for the agent's current reasoning between overshoot and steady state error is simply that training had not found the optimal solution before training ended and results presented are a suboptimal but functioning

reference tracking system. Further experimentation can easily be completed to corroborate this theory, but it is not considered important enough to commit any further resources at this time.

Regardless of the error type, these errors will sum up in the recordings of the RMSE, which were recorded to be later compared to the adaptive controller techniques. The basic sine input had the smallest value of 0.4540, likely due to the small amplitudes and low frequencies. The gait sine wave had an RMSE value of 6.5972 and the multi-step function resulted in the highest RMSE value of 9.1049. This does seem to match the visual results presented above, and supports the earlier claim that error is predominantly caused by the physical inertia of the robot being unable to keep up with the step transitions and higher frequency motions.

4.3.2. Stage 2 - Adaptive PID Controller Reference Tracking

4.3.2.1. Constant environment RMSE results

The adaptive PID controller showed promising results with a clear tracking pattern shown for a variety of input reference signals. When utilising identical input signals, this tracking was noticeably better than the end-to-end direct controller discussed above. However to justify the use of an adaptive PID controller it must be compared to a classical PID controller with constant gain values. These parameters were calculated from the average gain values from the step-trained adaptive PID experiment. These values could have been calculated manually, but by using the average of the values the RL agent selects it becomes much clearer whether changing the gain values during simulation has any merit. By varying the input signal type the average PID values change as well, and as such the results for the constant PID controller are split into three separate categories, showing three different constant PID systems. Only the gait-averaged systems will have reference tracking plots provided below as the other two input signals are similar and do not provide any additional information.

The basic sine reference tracking for each of the trained agents and the constant PID are shown below in Figure 4.14, with each approach providing almost exact tracking. The step-trained agent has a tiny offset and the gait-trained agent has mild oscillations during the first peak, but overall each approach could reasonably be utilised for low-frequency sine tracking.

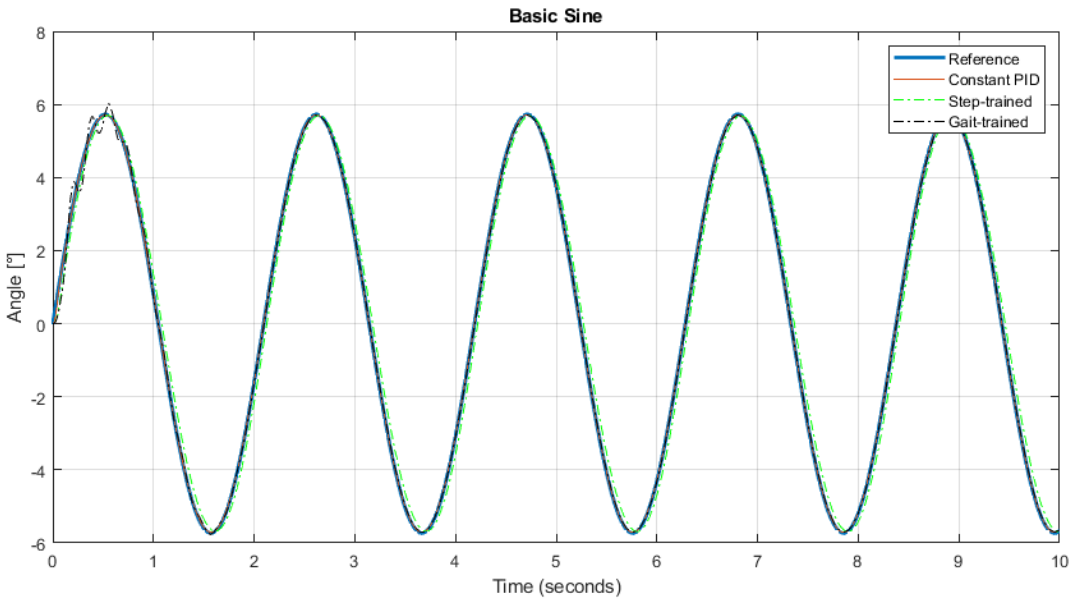

Figure 4.14. Response output for tracking the basic sine wave input in a constant environment. Contains reference tracking for constant PID (orange), step-trained adaptive PID controller (green), and gait-trained adaptive PID controller (black)

The gait sine reference tracking is presented in Figure 4.15, with successful tracking for each approach. The gait-trained agent repeated the mild oscillatory behaviour at the beginning of the simulation, causing a larger overshoot than last time, but settled very quickly. The step-trained agent also has the previous problem of slight offset to a worse degree. The absolute difference between the step-trained response and the reference signal is still no more than 2° at any given time which would likely be indistinguishable to a human positioning their ankle, so all approaches could reasonably be utilised for approximate walking gait-pattern tracking.

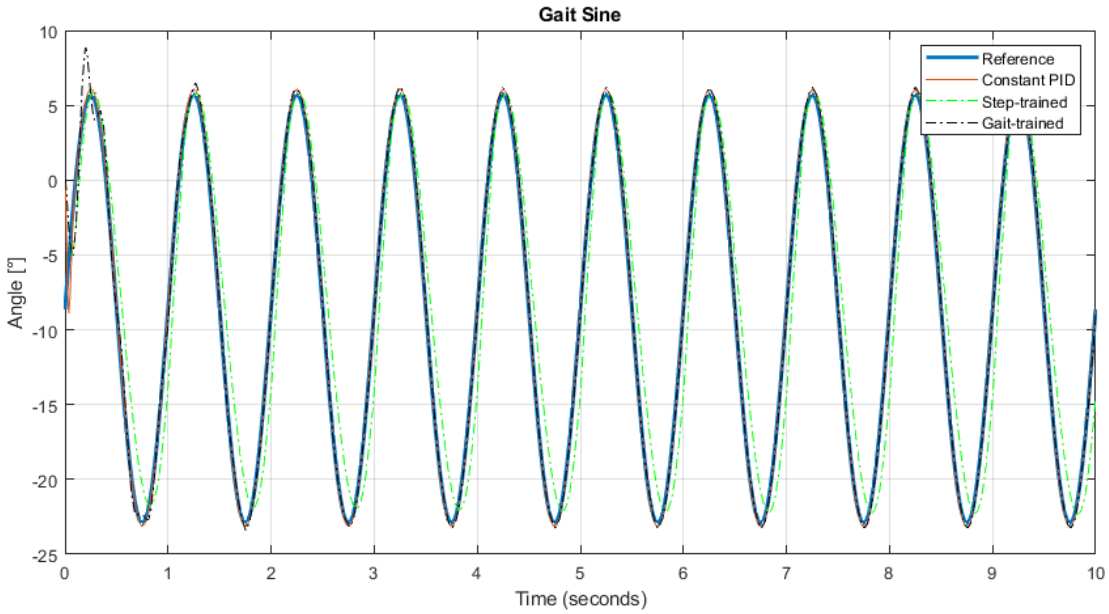

Figure 4.15. Response output for tracking the gait sine wave input in a constant environment. Contains reference tracking for constant PID (orange), step-trained adaptive PID controller (green), and gait-trained adaptive PID controller (black)

The multi-step reference tracking comparison is contained in Figure 4.16 and shows the constant PID controller with very good tracking. When compared to the adaptive PID controllers, the constant PID clearly outperforms the gait-trained agent and very similarly to the step-trained agent. The adaptive PID has smaller overshoot percentages but have a slightly longer settling time; there is no 'better' result between these as it will simply come down to preference, and in practice the difference between these results would be imperceptible for rehabilitation exercises. The gait-trained agent has large divergences from the previous two approaches, with much larger overshoots and oscillations for each of the step transitions. This is likely due to the gait-trained agent learning to always expect an upcoming oscillation during its training session, so it tries to counteract these predicted motions that never occur for the step input.

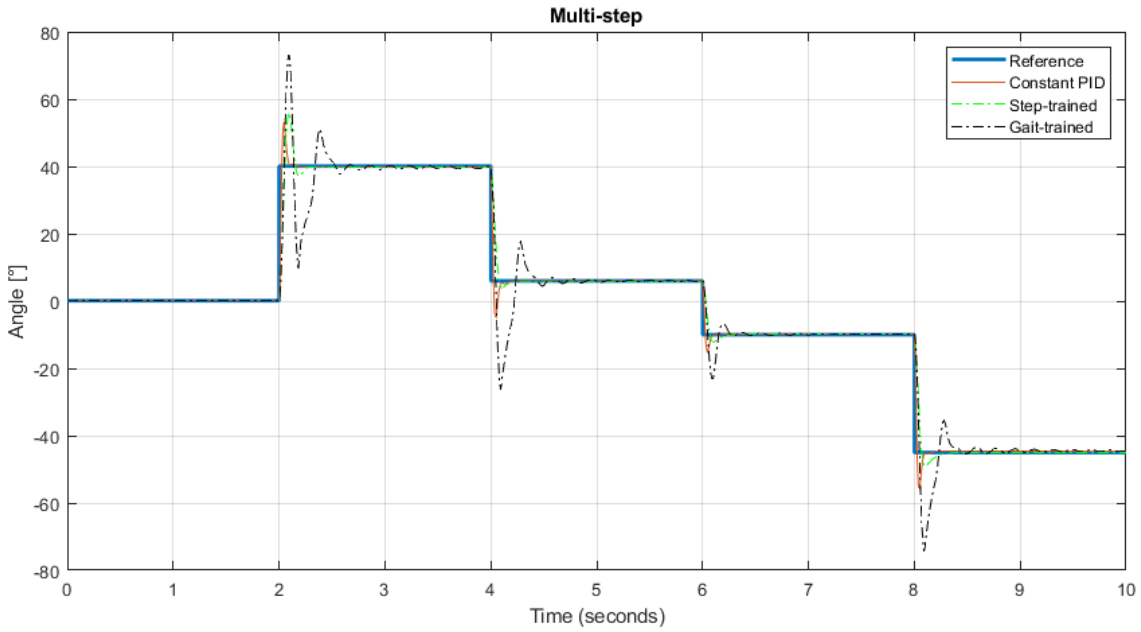

Figure 4.16. Response output for tracking the multi-step function input in a constant environment. Reference tracking for constant PID (orange), step-trained adaptive PID controller (green), and gait-trained adaptive PID controller (black)

This result clearly indicates the importance of the training data available to the RL agent as the step-trained agent performs exceedingly better than the gait-trained agent for the input it was trained with, and the gait-trained agent performed slightly better for the two sine-based inputs.

Table 4.4 – RMSE measurements for a constant environment [°]

Input Signal	Adaptive PID		Constant PID		
	Step-trained	Gait-trained	Avg. Basic	Avg. Gait	Avg. Step
Basic Sine	0.4020	0.1247	0.0143	0.0145	0.0141
Gait Sine	3.3927	1.0659	0.8859	0.8834	0.8857
Multi-step	6.6700	9.1187	6.4884	6.4807	6.4836

The root mean square error values aim to summarise the tracking results shown in the above graphs as a single value to further the comparisons. The results for the static environment RMSE values are presented in Table 4.4 and confirm that the adaptive controller performed better for the input signals associated with its training. Results also show that the constant PID controller outperformed both adaptive controllers regardless of input signal (with respect to the RMSE). When considered with the additional complexities of adaptive controllers, these results suggest a constant PID controller is better suited for simple unchanging environments under all conditions.

4.3.2.2. Changing environment RMSE results

With the constant PID outperforming the adaptive PIDs for constant environments it was theorised that adaptive PID controllers would be better suited for an environment that changes during operation and would require system retuning. With respect to the PID controller, the environment is considered anything not directly determined by the controller. For more advanced system interactions such as the admittance controller, this would include the force interactions and physical restrictions, but since the PID system is structured to be simple only the PaddleBot can be considered the environment. To simulate a changing environment, a step function that multiplied the torque input to the PaddleBot by a value of 1 for $t<6$ and 0.01 for $t\geq6$ was included into the control loop. This step drop effectively reduces the applied torque to the plant to 1% of its original value and causes the system to become non-linear, which reinforcement learning is well suited to handle. This change can be explained in a real-world example by a failure in the power supply, causing the motor to apply less torque than what the controller desires. From this perspective, using RL-based adaptive control can be considered a form of fault-tolerant control.

The same two adaptive PID controllers were used from the previous stage, and as such neither system was 'trained' to expect a change in environment. The agent trained using the multi-step function was again labelled "Step-trained" while the agent trained with the gait sine wave was labelled "Gait-trained". Training using the basic sine wave was deemed unnecessary and unlikely to present any new insights. These adaptive PID controllers were compared to a constant PID which had gain values that were calculated from the average gain values of the adaptive step-trained controller acting upon the gait sine input. The average gain values for the basic sine and the multi-step input were not recorded for this section as they do not vary enough to justify discussion. The constant PID controller in this experiment is identical to the previous experiment. New averages were not taken for the changing environment system as it was believed that the sudden change in gain values would cause gain average to be a less representative measurement than the previous simulation.

The basic sine input in Figure 4.17 shows all controllers have minimal divergence from the reference signal before the environment change, with the slight oscillation and offset from the gait-trained and step-trained agents from last experiment remaining (as the first 6 seconds of these results are identical to the previous results). After the environment change at 6 seconds there is a small disturbance present in the two adaptive controllers and a much larger disturbance for the constant PID controller. Where the adaptive controllers have some

undershoot and do not reach the full peaks of the sine wave, the constant PID drastically undershoots the reference and leads to a much larger error. The sine wave swaying remains even when the applied torque is drastically reduced, which may be contributed to gravity causing a pendulum-effect on the PaddleBot, as these oscillations are centred on the origin representing a direct down position in line with gravity. This does not seem to be the whole truth, as the final sine trough does have an increased magnitude, which suggests the PID controller is still aiming to control the system. This is also supported by looking at the remaining input reference tracking plots in Figure 4.17 and Figure 4.18.

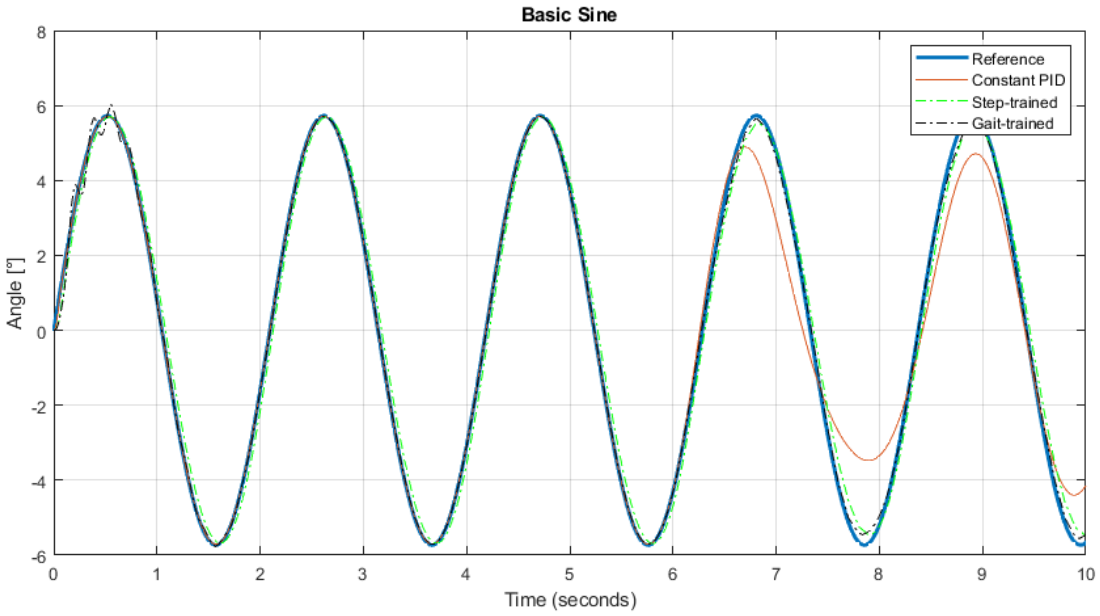

Figure 4.17. Response output for tracking the basic sine wave input for a changing environment at t=6

The gait input shows a more severe deterioration at the environment change, with the adaptive controllers overshooting the reference signal consistently. The gait-trained agent generally has less overshoot than the step-trained agent on the upswing, but more overshoot on the downswing where the step-trained agent has very little overshoot at all; both controllers keep the motion in-phase with the reference signal. According to the RMSE values the gait-trained agent is favoured over the step-trained agent, likely as the squaring of error leads to a favouring of small consistent error over large temporary overshoot error. The constant PID controller both overshoots and ends up completely out of phase with the reference signal, which is a very poor result and potentially distressing for rehabilitation purposes. The magnitude of the overshoot does decrease over time and the constant PID controller may eventually settle if given enough time, but this is not guaranteed and is not worth exploring if an adaptive PID controller can prevent the issue altogether.

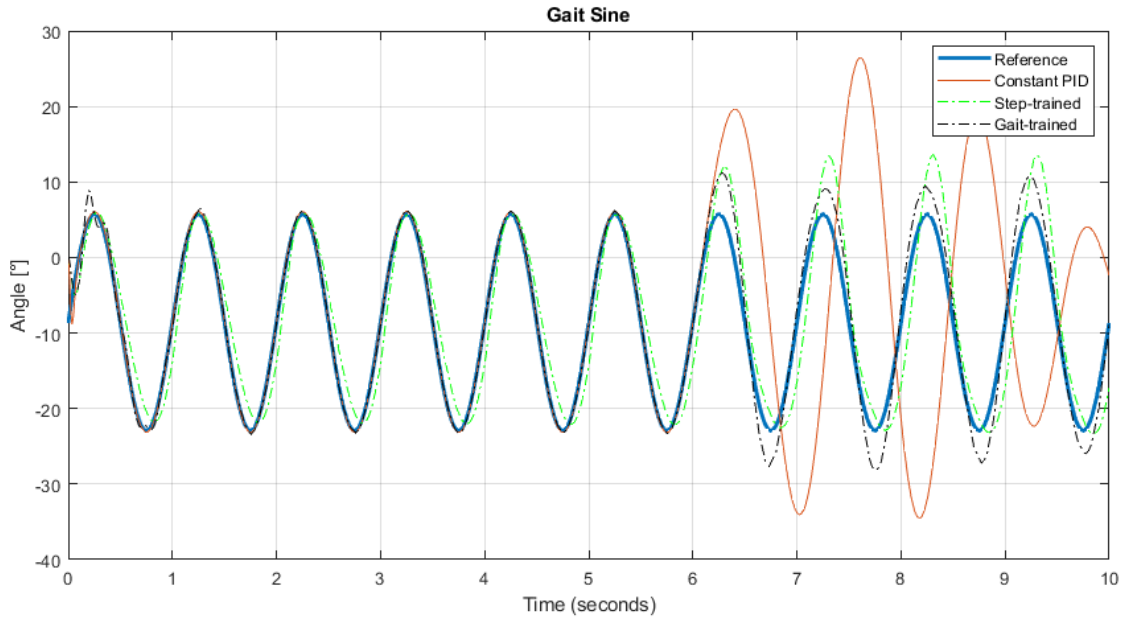

Figure 4.18. Response output for tracking the gait sine wave input for a changing environment at t=6

The multi-step response after the environment change shows both the constant PID and the gait-trained adaptive PID controllers fail to appropriately stabilise and track the signal. Although the gait-trained agent does have decreasing oscillations it would take too much time to reach steady state and should be considered a failure. Additionally, the constant PID controller does not seem to converge at all within the 2 second window available, which again should be considered a failure. The behaviour does not oscillate about the origin, so at the very least these results confirm that the controller is still functioning and causing some form of motor torque to be applied to the robot.

The step-trained adaptive PID results in very good tracking performance even after the environment change, with each step having minimal steady state error. This suggests a reinforcement learning agent is able to adjust the PID controller to new gain levels that would better suit the system as it changes. Since the change in environment was not present during training it suggests any changes to the environment could be potentially adjusted for with no required knowledge of what the change will be. This result shows RL has potential to adjust a rehabilitation robot to adapt to changes in human-robot interactions as an activity takes place; a very beneficial ability in the face of constantly changing force outputs, muscle stiffness, and unseen disturbances.

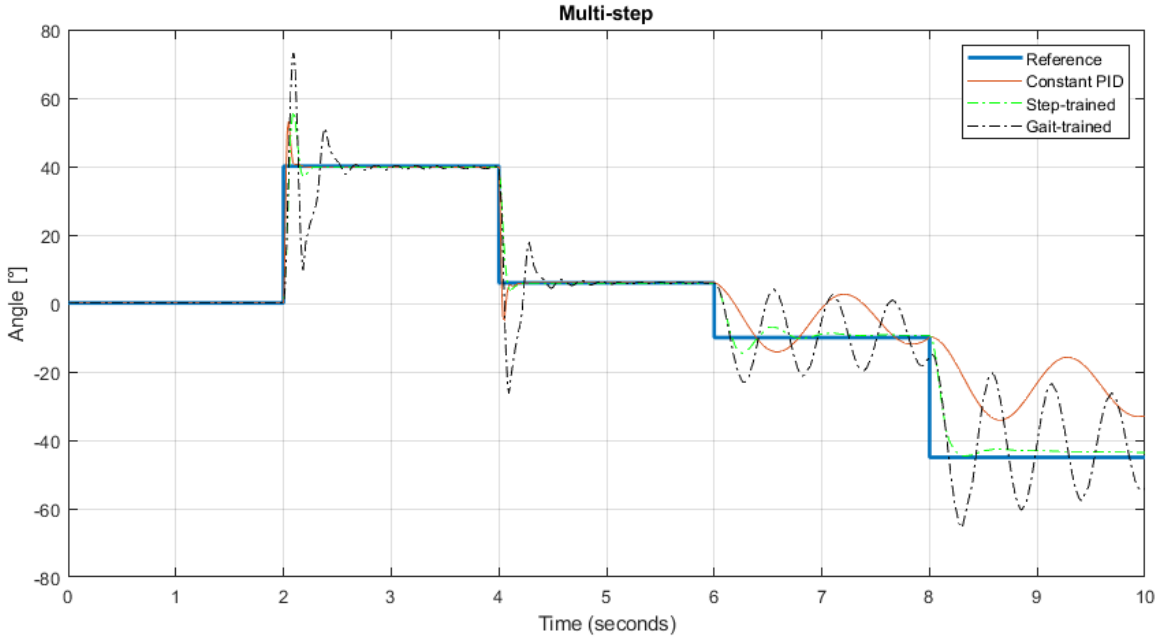

Figure 4.19. Response output for tracking the multi-step function input for a changing environment at t=6

The results for the changing environment RMSE values are presented in Table 4.5 and show the adaptive PID superiority for a nonlinear or time-variant system. Training dependence was one again highlighted as an important consideration when designing the RL system, as the gait-trained agent performed better for sine-based inputs and vice versa for the step-trained agent. The graphs suggest the improvement between gait-trained and step-trained agents was drastic for the multi-step input, which is difficult to observe directly from the RMSE measurements. Overall, the adaptive PID controllers outperformed the constant PID controller for every input signal, regardless of the training setup used. The adaptive PID controllers could be improved further by including a variety of input signals during training to better prepare the system for these predicted scenarios.

Table 4.5 - RMSE for a changing environment [°]

Input	Adaptive PID		Constant PID
	Step-trained	Gait-trained	Avg. Gait
Basic Sine	0.4119	0.1611	0.7113
Gait Sine	4.3908	2.3477	15.3714
Multi-step	7.0602	10.9052	11.9725

4.3.2.3. Effects of Input signal

To gain a deeper understanding of how both signal amplitude and frequency affect the RMSE, further experiments were performed using a sine wave to detail the results. The input amplitude A was varied between 0.1 and 0.75 to represent the full range of motion expected of an ankle. Healthy subjects would likely have larger amplitudes in range of motion compared to subjects with motion restrictions, and as such both ends of the amplitude results correspond to important real-world scenarios. The frequency ω varied between π/2 and 4π radians per second to produce repetitive motions that require between 4 seconds and 0.5 seconds, respectively. Higher frequency recordings of 10π were performed to collect data on the robotic model limitations at higher frequencies, but as they do not correspond to any reasonable rehabilitation practices they are omitted from this report. All values calculated with a bias of $B = -0.15$ and a phase value of $\phi = 0$ using the equation for the input signal below:

$$u(t) = A * \sin(\omega t + \phi) + B \quad (4.18)$$

Each calculation was performed using the step-trained adaptive PID controller to determine the RL agent effectiveness for new input signals. An identical experiment was also conducted using the constant PID controller which showed smaller RMSE results for every amplitude-frequency combination when compared to the adaptive PID results. These results can be observed in Table 4.7, however they are of less importance as these results do not illuminate the behaviour of the RL agent and its ability to compensate for input variations.

Table 4.6 - Amplitude-Frequency Effects on RMSE for Adaptive PID [°]

Frequency (ω) Amplitude (A)	π/2	π	2π	4π
0.1	0.906	1.042	1.456	2.412
0.25	1.140	1.798	3.392	6.067
0.5	1.824	3.581	6.855	10.208
0.75	2.657	5.353	10.034	13.602

Table 4.7 - Amplitude-Frequency Effects on RMSE for Constant PID [°]

Frequency (ω) Amplitude (A)	π/2	π	2π	4π
0.1	0.857	0.857	0.861	0.955
0.25	0.860	0.857	0.883	1.361
0.5	0.872	0.858	0.961	2.288
0.75	0.889	0.859	1.084	3.310

These values are plotted in 2D to try and identify a pattern that could be used to normalise RMSE values for different input signals. Figure 4.20 shows these trends for the step-trained adaptive controller with fairly linear amplitude-dependent growth irrespective of frequency; linear behaviour also remains for frequency-dependence for low amplitudes, but becomes more quadratic for higher amplitudes. Figure 4.21 shows the constant PID controller produces fairly linear growth for low frequencies, however a threshold seems to exist somewhere between 2π and 4π that causes large errors to grow. This may be biased by the extremely low RMSE recordings compressing the data points to appear linear but since the absolute RMSE values are so small the reasoning does not seem overly important; in practice any signal amplitude will not produce large errors so long as the frequency remains reasonable with reference to human motion.

For the constant PID controller, additional measurements were taken for 3π, 6π, 8π, and 10π to illustrate the quadratic relation between frequency and RMSE, although they were not plotted for consistency. These results recommend using the constant PID controller for any input signal with a low frequency which will be highly applicable for any human-machine interacting system. The adaptive controller is generally better suited for nonlinear models and time-dependent environments with uncertainty.

The general result of the adaptive PID controller returning higher RMSE values for sine inputs can be seen by comparing the two figures.

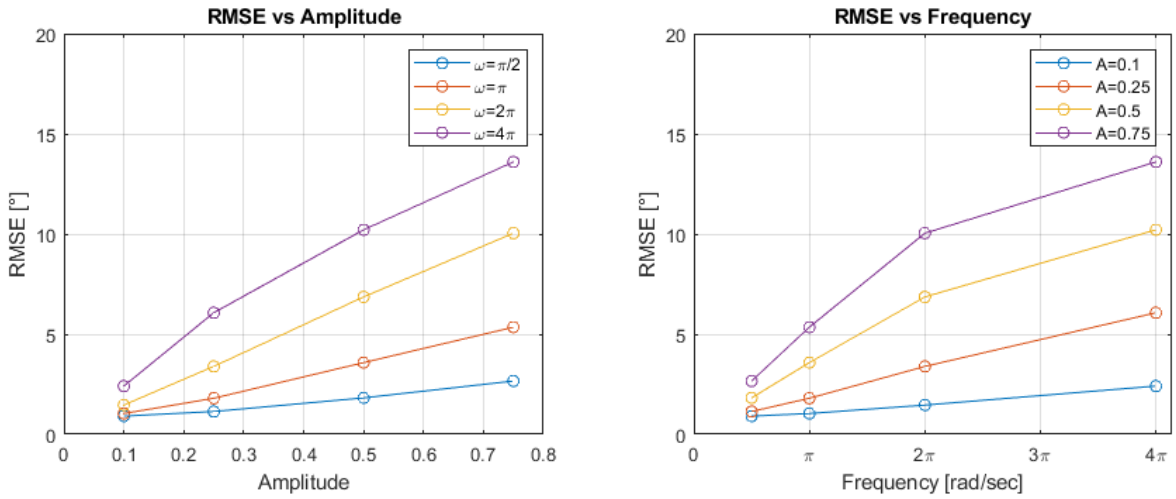

Figure 4.20. RMSE dependence on input signal amplitude (left) and frequency (right) for the adaptive PID controller

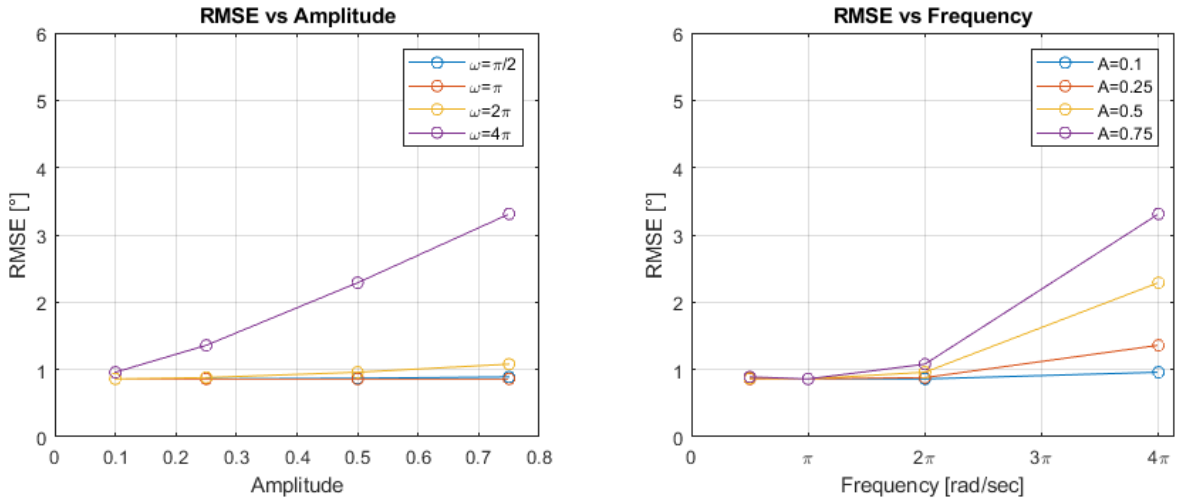

Figure 4.21. RMSE dependence on input signal amplitude (left) and frequency (right) for the constant PID controller

4.3.3. Stage 3 - Adaptive Admittance Controller Reference Tracking

The adaptive admittance controller performed the same basic experimental setups as the previous adaptive controller, by testing the ability of the system to track three different types of input signals: basic sine, gait sine, and multi-step function. The primary difference in this case is that an obstacle limits motion at pre-determined positions which should stop full tracking from being possible, so a direct RMSE value will be less representative of effective performance. The obstacle is referred to as the environment, and is utilised in three separate scenarios to represent a human body with the parameters of Range of Motion (RoM) and environmental stiffness (K_e):

1. The environment remains constant with the PaddleBot having unimpeded motion within the bounds of [-40°, +25°] and the environment having a stiffness coefficient K_e of 20 N/degree
2. The environment begins in a stiff state, with K_e = 40 N/degree and RoM = [-15°, +15°]. Halfway through the simulation at t=5 the environment relaxes and becomes more flexible as a human would during stretching, with K_e = 20 N/degree and RoM = [-40, +25]
3. The environment begins in a relaxed state, with K_e = 20 N/degree and RoM = [-40°, +25°]. At t=6 the environment stiffens and becomes less mobile as to represent a human suffering injury or fatigue, with K_e = 40 N/degree and RoM = [-15, +15]

As the primary goal of these experiments is to determine the benefits of adaptive controllers, each of these environments were simulated using a variety of different admittance controllers:

The *constant method* utilised an admittance controller with constant damping and stiffness parameters throughout the entire simulation. These values were equal to $K = 66.38$ and $B = 59.20$ which were derived from the average values of the adaptive method discussed later. This approach is the most simple of all methods and functioned as the control for all future experiments.

The *switching method* behaved similar to the control method, but changed the admittance parameters between two different modes depending on whether the system was inside or outside the RoM. While the PaddleBot was recorded inside the RoM the controller was set to "rigid" mode where $B = K = 140$ to encourage close tracking of the reference signal. When outside the RoM the controller was set to "malleable" mode with $B = K = 10$ to try and prevent the system from pushing past the range of motion that represents the humans comfort limits that a stiff system would. In this mode, external forces should in theory allow external forces to manipulate the system to a higher degree.

The *adaptive method* used reinforcement learning to continuously adjust the admittance parameters every 0.1 seconds based on the current observations. The actor and critic networks used the architecture displayed in Figure 4.8 trained in the new environment simulation. A second adaptive method (labelled "adaptive 2") was also trained and tested with higher levels of exploration and the newer critic architecture shown in Figure 4.12, but results for RMSE recordings were not as successful.

4.3.3.1. Constant Environment

Using the root mean square measurement for the numerical comparison, it is important to identify where the environment prevents complete tracking of the reference signal. For the case of the constant environment, only the multi-step function exceeds the viable range of motion, so only the multi-step input is expected to be affected by the admittance controller. To account for this, the RMSE value of the multi-step is recorded a second time, but will measure error between current position and range of motion limit when the reference signal exceeds this. This will help determine error while the system is in the standard RoM, and how far from the limit it is positioned when operating outside the RoM. Since the system should be pressed against the physical limit when the reference surpasses the limit, the error during these states should ideally converge to zero. This second RMSE recording is labelled as “clipped step” in Table 4.8 below. The visual tracking graphs for the three input signals are shown below as well. Only the multi-step function is affected by the physical limitations, so the only figure to include a visual representation of these restrictions is Figure 4.24.

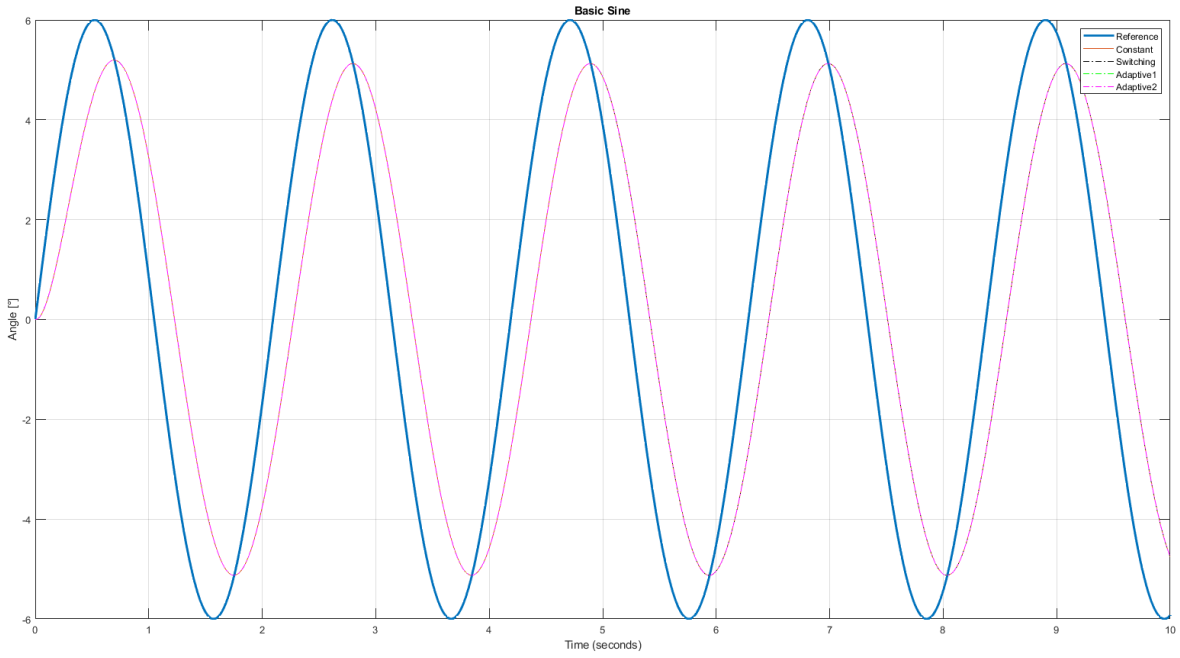

Figure 4.22. Response output for tracking the basic sine wave input in a constant environment. Contains reference tracking for constant admittance (orange), switching admittance (black), and step-trained adaptive admittance controllers 1 (green) and 2 (magenta)

The basic sine wave with an amplitude of 6° shows that every control attempt was essentially identical with no meaningful differences. As the signal never approaches the RoM limits there is never any external force generated and no altering signal generated. This shows the intuitive

conclusion from this fact, in that the admittance controller is never employed and the gain values do not affect the final outcome at all. The signal lag is caused by the constant PD controller which cannot be changed by the RL agent or the switching method. Manual tuning of this controller did show improvements of RMSE, but as these experiments were designed to show the effects of admittance controller tuning, the sub-optimal tracking was not considered a problem and was simply held constant between tests.

The gait input signal also shows identical outputs due to the input signal not being large enough to interact with the environment. The manual tuning of the PD controller showed improvements for the gait signal as well, so the signal lag and amplitude attenuation can be attributed to a derivative gain set higher than optimal.

The multi-step function is the only input signal that leads to environment interaction and as such is the only signal that shows differences in tracking. By including the physical bounds on the response output plot it becomes clear that each controller is working as intended (with varying degrees of success) by pushing the current position back towards the desired range of motion. The constant controller leads to the least amount of adjustment as it stays relatively close to the reference signal at each step, regardless of boundaries. For the out-of-range steps at t=2 and t=8, the switching method returns back to 5° of the respective bound within 1 second and remains steady after that. This is the desired behaviour, but unfortunately leads to error within the following in-range step as seen at t=4.

The two adaptive controller methods also show reasonable reaction to reducing out-of-range operation, with adaptive 1 having a faster response time at the cost of more overshoot. Visually it seems that adaptive 2 has the best response behaviour, but the RMSE values for the clipped multistep signal show adaptive 1 had less error overall. In fact, adaptive 1 showed the smallest (or equal smallest) RMSE value for every input signal, although the difference was negligible for the sine-based signals.

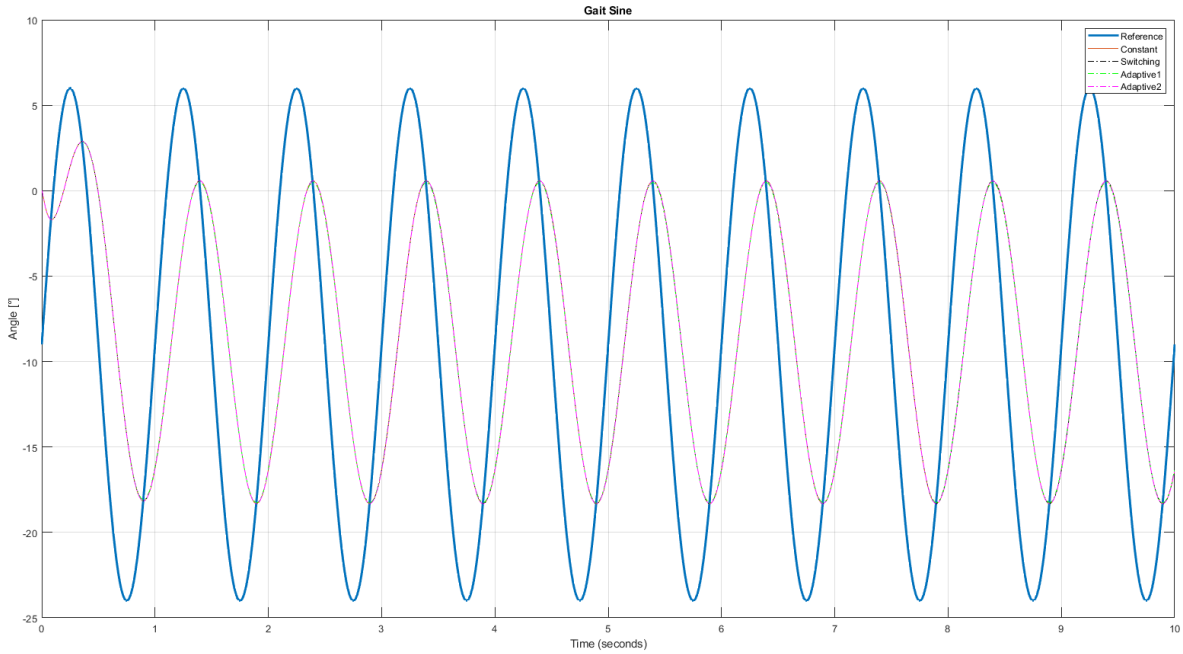

Figure 4.23. Response output for tracking the gait sine wave input in a constant environment. Contains reference tracking for constant admittance (orange), switching admittance (black), and step-trained adaptive admittance controllers 1 (green) and 2 (magenta)

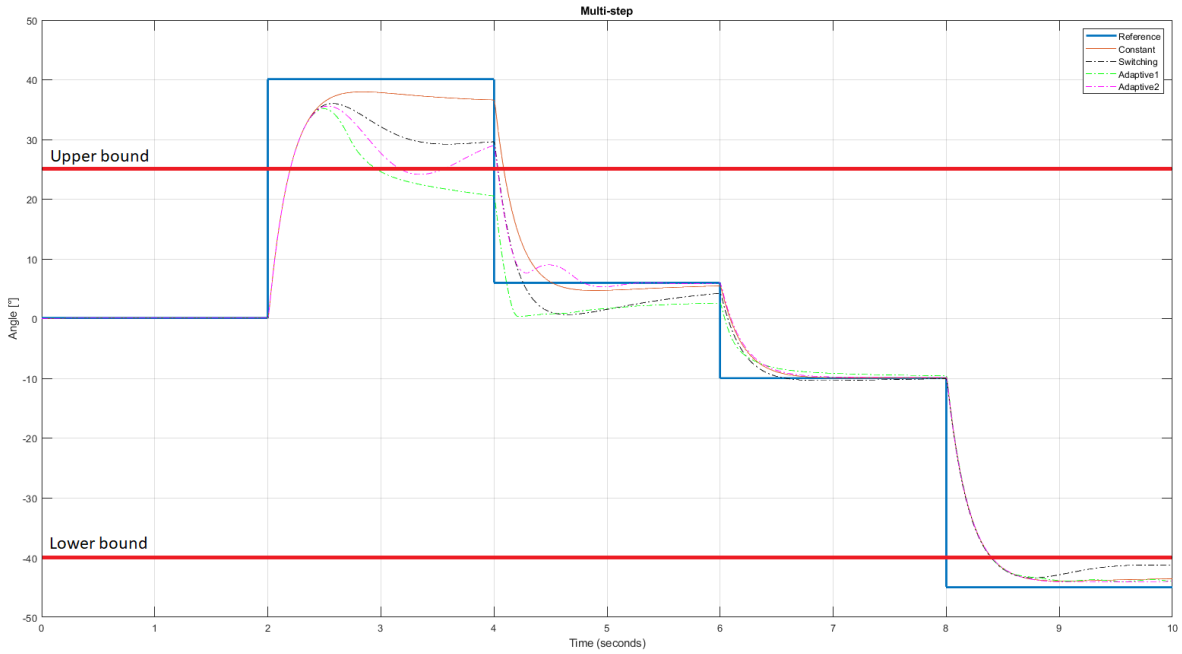

Figure 4.24. Response output for tracking the multi-step function input in a constant environment. Contains reference tracking for constant admittance (orange), switching admittance (black), and step-trained adaptive admittance controllers 1 (green) and 2 (magenta). The physical restrictions representing the human ankle range of motion are shown in red

Table 4.8. RMSE measurements of admittance controller for constant environment

Input Signal	Control Method			
	Constant	Switching	Adaptive 1	Adaptive 2
Basic Sine	2.148	2.148	2.148	2.148
Gait Sine	8.287	8.287	8.243	8.297
Multi-step	7.964	8.344	9.477	8.850
Clipped Step	7.907	6.402	5.856	6.145

As the multi-step row reports the error between position and reference with no consideration to environmental impact the data in this row has less importance than the others. If ignored, it becomes clear that the adaptive controller performed noticeably better than the non-reinforcement learning based methods for the step response, and slightly better for the gait input.

4.3.3.2. Changing Environments

The two changing environments were simulated to determine how each admittance controller would handle a change in environment that effectively changed the amount of interaction forces generated by the environment. The basic sine input is omitted from the following figures and tables as it did not reach the environment and was not affected by these changes; every result remained 2.148 for both environments. Additionally, as both environments have a more limited range of motion at some point in time (±15°) the gait input signal is also affected by the environment and requires a “clipped gait” recording for the same reasons discussed earlier for the multi-step function.

Observing Figure 4.25 shows a larger difference between control methods during the more restrictive period, as seen by the larger spread of the sine wave amplitudes for times before 5 seconds. Once the environment restriction is effectively removed, each control method converges and the differences become negligible.

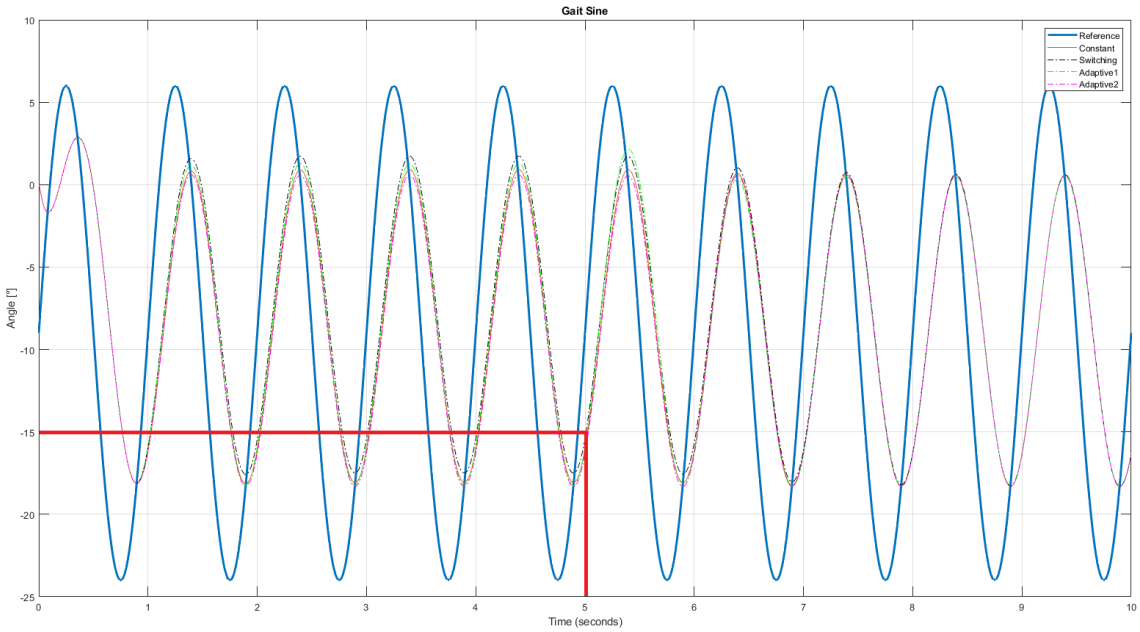

Figure 4.25. Response output for tracking the gait sine wave input in a changing environment (environment 1). Contains reference tracking for constant admittance (orange), switching admittance (black), and step-trained adaptive admittance controllers 1 (green) and 2 (magenta). The physical restrictions representing the human ankle range of motion are shown in red

The multi-step function shows the effects of the environment much clearer. Comparing Figure 4.26 to Figure 4.24, the more restrictive environment shows three of the four methods being pulled closer to the physical limitations. The adaptive 1 method is the only method that did not cause a significant drop in the final value during the first step, suggesting the adaptive admittance controller 1 is more resistant to environmental change than the other controllers. Conversely, the second adaptive agent seems to suffer from the opposite problem and produces large overshoots for the more restrictive environment that had not been experienced during training. The switching method behaves somewhere between the two adaptive models, and visually seems to be the best performing tracking, especially when considering the bounds were established with an extra 5° buffer included, so the results resting within 5° of the environmental limit is an acceptable result. The extra stiff environment also leads to a drop in tracking ability for the second step for all controllers, although the drop is more severe for the two RL methods.

These results show the adaptive methods producing reasonable results for an environment they were not trained in, but it does not show how the methods deal with an environment that changes during operation until the transition at t=5 seconds. The change in environment does not seem to have any immediate or dramatic effects on current position. Since the admittance controller only depends on environmental interaction and there is no environmental interaction

at the time change, this result is completely expected. Even if the time change happened during out-of-range operation the changes would be minimal, as the contribution of the admittance controller should be less than the PID controller. The final step also falls to out-of-range operation where the switching method is the only result to show any environmental impact response.

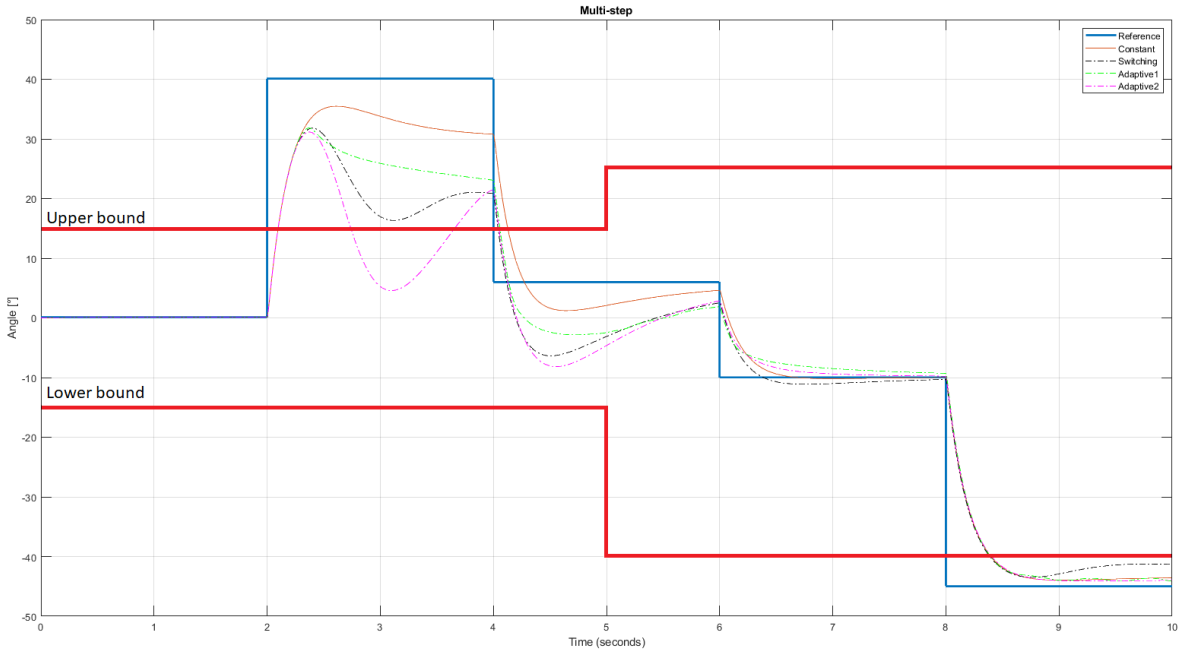

Figure 4.26. Response output for tracking the multi-step function input in a changing environment (environment 1). Contains reference tracking for constant admittance (orange), switching admittance (black), and step-trained adaptive admittance controllers 1 (green) and 2 (magenta). The physical restrictions representing the human ankle range of motion are shown in red

The summary RMSE table for this first changing environment is show below. However as the gait wave and the multi-step function both operate out-of-range temporarily, the clipped-input data is more relevant for proper rehabilitation operation. With this in mind, the switching method produced the best RMSE values for both the input signals.

Table 4.9. RMSE measurements of admittance controller for changing environment 1

Input Signal	Control Method			
	Constant	Switching	Adaptive 1	Adaptive 2
Gait Sine	8.277	8.288	8.226	8.266
Multi-step	8.088	11.035	9.614	13.501
Clipped Gait	7.603	7.541	7.553	7.621
Clipped Step	9.215	6.789	7.438	7.251

The exact same tests were performed for a new changing environment, which begins flexible and becomes stiff at time t=6 seconds. All behaviours from this test remained consistent with the previous changing environment: The gait wave controllers produced more variant results during the stiff period and more conforming results during the relaxed period.

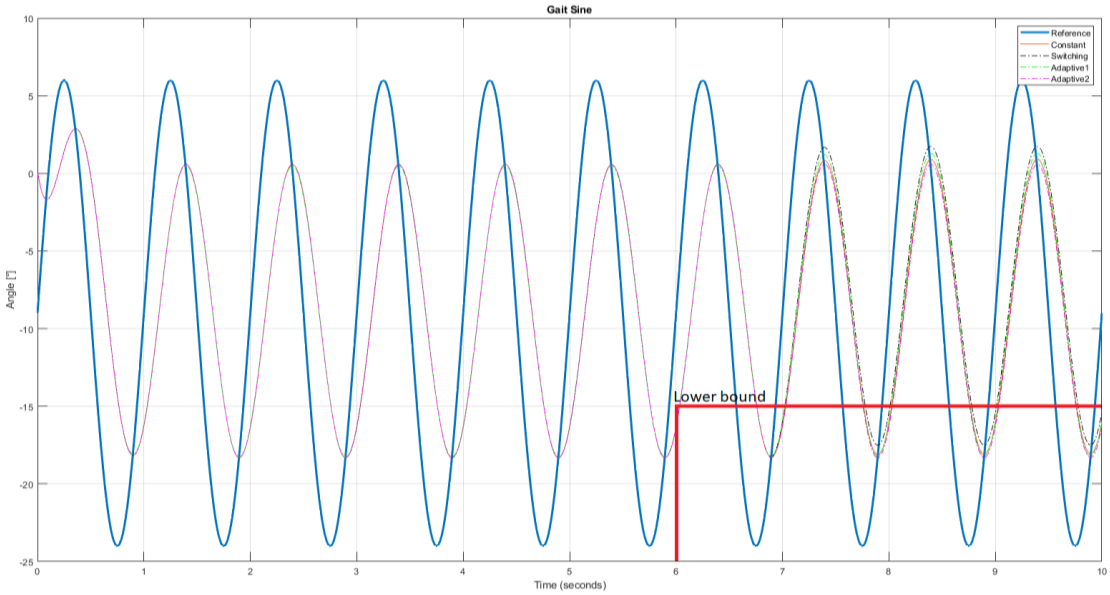

Figure 4.27. Response output for tracking the gait sine wave input in a changing environment (environment 2). Contains reference tracking for constant admittance (orange), switching admittance (black), and step-trained adaptive admittance controllers 1 (green) and 2 (magenta). The physical restrictions representing the human ankle range of motion are shown in red

The multi-step function once again showed the adaptive controller 1 being much more resistant to environmental interactions, but with more oscillations that stayed worse than the constant controller. The adaptive controller 2 again produces large swings and overshoot from the physical limitations. Neither solution is desirable for rehabilitation unless the environment is sufficiently soft and joint stretching is the primary objective.

The switching method once again showed the best practical result, with the system pulling back to the environmental limit once the interaction forces become non-zero. As the switching method merely changes the stiffness and damping values between two constant values, the results of this method combined with its simplicity make it a very favourable approach to pseudo-adaptive admittance control. The summary RMSE table for the second changing environment is presented in Table 4.10 below.

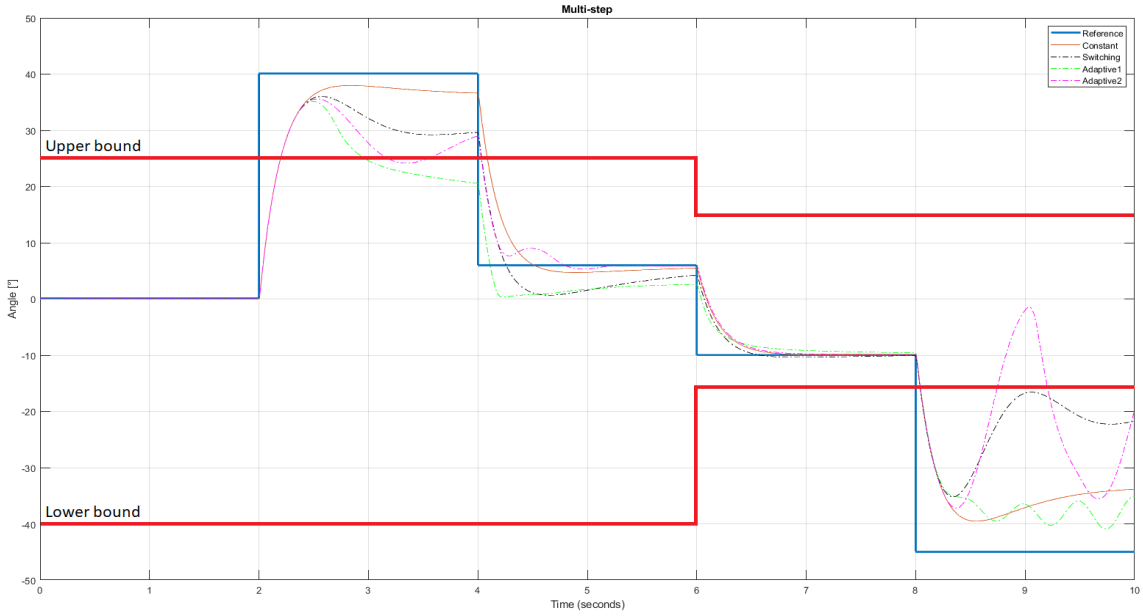

Figure 4.28. Response output for tracking the multi-step function input in a changing environment (environment 2). Contains reference tracking for constant admittance (orange), switching admittance (black), and step-trained adaptive admittance controllers 1 (green) and 2 (magenta). The physical restrictions representing the human ankle range of motion are shown in red

Table 4.10. RMSE measurements of admittance controller for changing environment 2

Input Signal	Control Method			
	Constant	Switching	Adaptive 1	Adaptive 2
Gait Sine	8.282	8.292	8.218	8.271
Multi-step	8.727	12.665	9.949	13.467
Clipped Gait	7.758	7.713	7.694	7.769
Clipped Step	11.567	6.944	10.715	7.931

Reinforcement learning based adaptive control does not seem to be reliably better than the constant method, which was not even tuned for optimal performance (the first adaptive average was used without change). The switching method shows a lot more promise and produces more reliable results for the inputs (when considering the clipped context). As the clipped values better represent real-world desires for physical therapy, the switching method showed the best RMSE results for the multi-step input by a wide margin and was only minimally worse for the gait input to the point that the difference is negligible.

From these two changing environments the switching method can be labelled as the preferred admittance controller method with reasonable certainty. This method could be considered both constant and adaptive control as it does adapt to its environment, but with much less precision than the RL-based methods for adaptive control.

4.4. Discussion

Each of the stages of development showed a different niche of control systems engineering that could be filled by reinforcement learning to varying degrees of success. The results of Stage 1 showed direct end-to-end control of the 1 degree-of-freedom PaddleBot was possible, but with fairly unsmooth tracking. The response looks 'bumpy' in comparison to the future PID tests which suggest two possible reasons: First, end-to-end control sets the input torque value directly and will therefore not be as smooth as a torque value that is calculated by an error calculation and it's integral and derivative. The nature of end-to-end may simply trade off the smoothness inherent in PID control for its direct control. The second possible explanation is that the Stage 1 testing was the practical introduction for reinforcement learning and is likely to be far from optimally implemented. As the project progressed and understanding developed the general application of reinforcement learning became better and produced superior results. The bumpy response may simply be a result of a poorly designed reinforcement learning controller that may be improvable with hyperparameter adjustments and a longer training session. Although no further testing will be performed to evaluate this hypothesis, the tests in stage 1 still provide enough confidence to consider future end-to-end RL approaches a viable strategy, as there is no conclusive evidence that end-to-end is always inferior to controller tuning approaches.

The remaining two stages focused on tuning an adaptive PID and adaptive admittance controller, respectively.

4.4.1. General PID analysis

4.4.1.1. Comparison of adaptive PID controller versus constant PID controller for constant environment

The results of the step-trained RL agent that varies the PID gain parameters over time, when compared to a PID controller that uses the average of these time-dependent gains show that the constant gain PID controller performed better for every signal input case when the environment was static. Both systems performed best when the input signal was a basic sine

wave, with RMSE values less than 0.5° for both systems. Although the adaptive PID performed worse than the constant PID, the minimal error shows that, in practicality, both systems function to an acceptable degree for the purposes of rehabilitation where small angular discrepancies are unlikely to have large impact on output of human safety. The discrepancy between adaptive and constant controllers is more prominent for the gait sine input, where the RMSE is almost 4 times larger for the step-trained adaptive PID (3.39 vs 0.88). These results are unexpected as the adaptive PID controller should be more flexible and able to achieve anything the constant PID controller is able to accomplish. One possible explanation is a non-optimal reinforcement learning model setup that requires more observation measurements to appropriately choose the actions. As the RMSE is calculated with the overall simulation time, a potential moving average of error would help identify RMSE measurements and allow better estimations of long-term results. As a potential fix: including a negative reward component to discourage gain variation the system would develop a 'gain inertia' to the action space and would try to minimize any variations in gain when possible. This may also result in RMSE values closer to the constant PID, although it does not address any underlying issues with the model that may have caused the differences in results, and simply aims to replicate a different result.

The step input is the notable exception to the rule of constant PID outperforming the adaptive PID by a large factor. Constant PID and adaptive PID results were very similar in RMSE outcomes (3.36 vs 3.32), with a difference that could be ignored in practice for rehabilitation purposes. A variation of the multi-step input was the signal used during training and as such it is reasonable to assume the RL agent is more adept at minimising error for step-based inputs. An agent trained using a gait cycle reference showed better results for the adaptive basic and gait inputs compared to the adaptive basic and gait inputs trained using the step; the tracking for the multi-step function produced worse, but still reasonable, RMSE measurements. This shows that training the system with the appropriate input does reduce error, however the constant PID still outperformed the gait-trained adaptive PID for a gait input. The discrepancy between these two results does diminish, which confirms the idea that the step-trained results were closer to the constant results for the multi-step input due to the training, and not an inherent property of step-based functions versus sine-based functions.

These tests do not utilize noise or disturbance signals, two aspects of practical robotics that may cause deterioration in overall control. Although classical PID controllers act as disturbance rejection quite effectively they are not as capable for noise rejection due to the nature of the feedback loop system. Noise causes the system to lose its linear properties and as such the PID

with constant gain values becomes unable to handle noise perturbations. Reinforcement learning is much better suited for nonlinear systems and as such it is expected that the adaptive PID controller would produce better results for a system with included noise. RL is also more capable in dealing with uncertainty and changing systems, whether these changes occur in the environment or the plant. As the training setup utilized a rigid and unchanging environment which does not appropriately represent a human body, the results became biased towards systems that performed well for simple interactions. Each of these reasons combined give a reasonable explanation for the results favouring the classical constant PID controller over the adaptive PID controller. It also identifies the scenarios in which adaptive PID controllers would be better suited (Fahmy, Badr, and Rahman 2014), and how RL can assist in the tuning (Lee, Lee, and Yim 2020; Qin et al. 2018).

4.4.1.2. Effects of environmental change

The change in environment that occurs after 6 seconds of simulation is shown to have negative effects on the tracking ability of both adaptive and constant PID controllers, with the adaptive controllers handling the change noticeably better. For cases where the agent was trained with the appropriate input signal, reference tracking remained highly effective throughout the simulation.

Alternate forms of environmental change were briefly explored: multiplying the PaddleBot input signal by a factor of 0.01 and 2 showed a representation of an undercompensating actuator or an overcompensating actuator, and multiplying the PaddleBot output signal by a factor of 0.01 and 2 showed a representation of a faulty sensor that returns angular readings much smaller or twice as large as reality. The attenuated sensor test showed poor tracking for constant PID control and improved but still low-quality tracking for the adaptive controller. The amplified sensor test showed much faster response times for both controllers, but this increase led to a much noisier and volatile adaptive controller. For the purposes of rehabilitation neither of these controllers would be suitable as they would likely cause harm to the user from the rapid snap motions, so the adaptive controller is unable to act as fault-tolerant control for a failing sensor. This was due to the adaptive controller depending on the measured position angle as an important observation and without it, the calculations within the neural network cannot be performed. The amplified actuator test showed similar results to the amplified sensor test, with response times reduced and faster motions produced. For identical reasons, these fast motions are not desirable for a rehabilitation system. The adaptive controller may be able to minimise

these negative outcomes with further training and reward tuning, but this was not followed upon for the purposes of this project. The attenuated actuator environmental change was the successor of the tests and all results presented were based off this approach.

The gait-trained adaptive controller returned better RMSE values for the sine-based inputs and the step-trained adaptive controller returned better RMSE values for the step-based input. Despite this, the step-trained controller is considered more successful overall as the RMSE values do not incorporate all tracking details, and while the step-trained agent was able to provide reasonable tracking for sine-based inputs the gait-trained agent was not able to provide reasonable tracking for the step-based inputs. The step-trained agent was more successful in a wider variety of cases, where each input signal was followed well enough for ankle motion exercises. The effects of the environmental change can be most clearly seen through the multi-step input signal. Comparing the static environment experiment to the changing environment experiment show a longer and larger transient state when the motor signal had been reduced, but was still able to settle relatively quickly with only a small steady state error by utilising the adaptive controller.

The constant controller, however, is incapable of tracking the step function and simply oscillates for the remainder of the simulation. The oscillations do not oscillate perfectly about the origin and as such the motion is unlikely to be a simple gravity-affected pendulum, some component of motion must be being produced by the controller. As the simulation only continues for 4 seconds post-environment change it is possible that the constant PID controller would slowly improve over time and eventually reach good tracking. Each input signal did show a still-changing signal that had not settled to a steady-state behaviour. Again, by considering the real world applications of the system, long periods of time will not be present to allow any settling behaviours. Results are empirically better for adaptive controllers even in the presence of unforeseen circumstances, so no further research was conducted into the long-term behaviours of the constant controller. The adaptive controllers were not trained with this change of environment and as such it is believed they would be well-suited for any other environmental changes that may occur during operation. The RMSE values in Table 4.5 show that the gait-trained adaptive controller performs almost as bad as the constant controller (10.91 vs 11.97). This finding is important in understanding the training stages of adaptive control and that improved tracking is not a guarantee for all circumstances.

Human-robot interactions fall under the high-uncertainty category due to the differences between users as well as the internal mechanics of the human body changing during exercise

and muscle activation. Current setup does not analyse any environmental interaction which removes one of the largest sources of uncertainty from the system entirely. As rehabilitation robotics is defined by its human interactions, and a human environment is guaranteed to change over time, any environment included in testing must also be time-dependent or include uncertainty. The model not using any environmental feedback is not a good representation of real human interactions, where it has been shown that internal stiffness will vary over a traditional gait cycle and general ankle motion. Regardless of these experimental shortcomings, both adaptive PID controllers and classical PID controllers returned RMSE values small enough to justify their application in rehabilitation robotics where input signal amplitudes and frequencies will be restricted for human safety concerns.

4.4.2. Admittance controller analysis

4.4.2.1. Shortcomings and Potential Expansions

The results for the admittance controller tests showed that only the multi-step function input had a noticeable affect on the system response, as both sine-based inputs remained within the a standard human ankle range throughout the simulation. These inputs were chosen to remain consistent between PID and admittance testing, along with their proximity to real-world motions. Unfortunately following these principles led to uninformative results for the admittance testing, and if future experiments are conducted the input signals should be adjusted to guarantee interaction with the environment. Alternatively, the environment could have been modified to meet the same guarantee, but for similar reasons the environment was established to resemble the real-world limitations of the human ankle and reducing it further would have negatively impacted this approximation. Because of these limitations in the results, most discussion focuses on the multi-step function results.

For the comparison of adaptive control and constant control: When an interaction force is experienced by the system, the constant controller leads to the least amount of adjustment to the desired position, as it stays relatively close to the reference signal for each jump in value and disregards majority of the experienced forces by exceeding the environmental boundaries. For scenarios where the reference signal is strongly controlled and designed by the operator (such as in passive rehabilitation with pre-determined exercise patterns) close reference tracking may be acceptable, but for future stages where the reference signal is determined by artificial intelligence (such as transforming bio-signals into a desired angle position to act as the

reference) this may be considered unsafe. A poorly classified reference may lead to the system tracking out-of-range to the point of harm, and the constant controller would not contribute enough of a component to the command signal that would ensure human safety. Contrary to this, the adaptive controllers were shown to be able to change their response depending on the experienced forces. This would allow the controller to act non-linearly in cases where the force values continued to rise, and could prevent this much more reliably. Through reinforcement learning especially, the force component of the reward function can be directly tuned to adjust how much the system should favour or disfavour these experienced forces. The poor performance may be attributed to a lack of manual tuning for the constant parameter values. Even if this was the case, as the stiffness and range of motion of each participant would vary, so would the interaction forces. As such, it would be a requirement to tune these parameters in reference to these environmental states for every system run-through and would require professional input to the system. With the goal of making these technologies more widespread, removing the need for professional input is an important contribution that can be achieved, or at the very least partially provided by using adaptive controllers.

Analysis of the tracking results and RMSE values show performance was reasonable but did retain large amounts of error at all points. This was more noticeable in the sine-based inputs, which had a consistent lag between input and output, along with some amplitude attenuation. The cause of this lacklustre performance can be attributed to the constant PD controller used in the inner loop. The PD controller was held static at $K_P = 75, K_D = 15$ which was not tuned for optimal tracking and was instead set for the step response rise time of approximately 1 second. This led to not being very good for sine wave inputs, as the error derivate caused significant changes to the produced input torque values. Reducing K_D in the constant admittance controller model led to immediate improvements in tracking. $K_D = 5$ would have been a better value to use for the sine waves, but caused the rise time of the step response to be much shorter than desired. This rapid motion was undesirable for safety concerns, especially for cases with environmental interaction where rapidly experienced forces would fail to be incorporated by the controller in time.

Additionally, the RMSE values seem to have relatively consistent differences between control methods, so it is believed that even if the tracking was substantially improved, this improvement would be present in each of the control methods and none would benefit substantially more than any other. Therefore the superior controller method would have likely remained consistent between tests even if this change was implemented and a better PD controller was used. This

claim is conjecture, so further research into better inner loop control may be worthwhile. One potential answer may come from allowing the reinforcement learning-based methods to have both the admittance controller and the PD controller be adaptive simultaneously, requiring the RL agent to produce 4 actions for the gain parameters of $K_P, K_D, B, \& K$. Some small attempts were made to implement this idea once all other stages had been completed, but no results were successful.

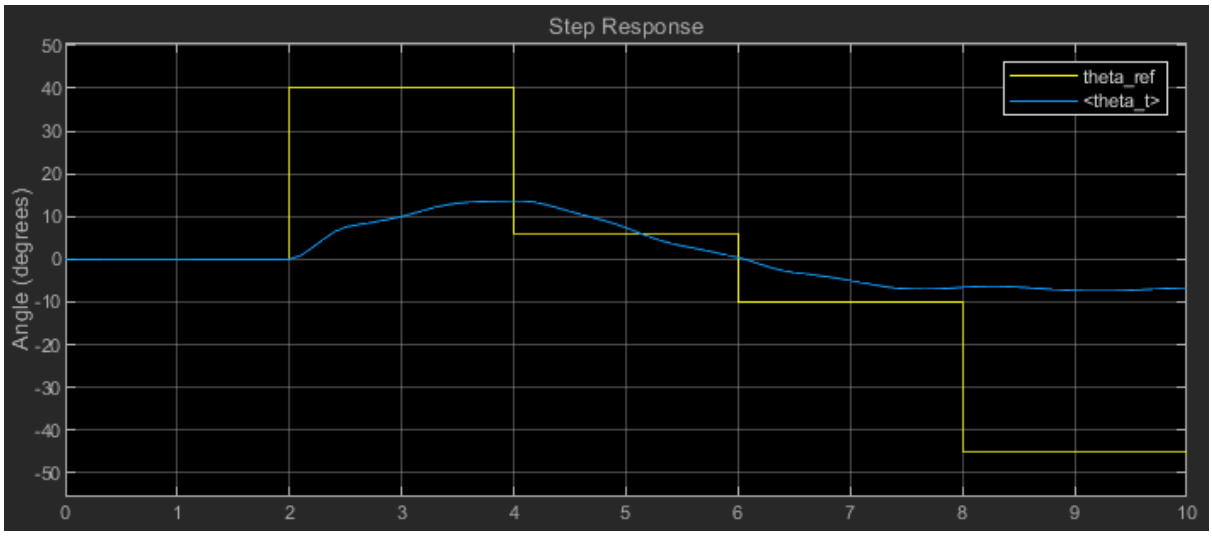

Figure 4.29. Reference tracking ability for an RL agent controlling the admittance controller gains B and K, and the PD controller gains Kp and Kd. Trained for 5000 episodes

Two primary training sessions were performed with the only hyperparameter changed being the learning rate. A learning rate of 0.001 led to the agent immediately terminating via the terminal condition of >60° PaddleBot positioning. A learning rate of 0.0001 led to agent trying to avoid motion altogether, with a bare minimum attempt at reference tracking, as seen in Figure 4.29. This result shows some promise that may have improved with a longer training session or a rebalancing of the reward function, but as this was not a focus of the chapter no real attempts were made as of writing. Should this project be continued, utilising adaptive behaviours in both controllers is the recommended follow-up approach.

4.4.2.2. Effects of changing environment

The effects of the changing environment were much less prominent for the admittance controller than it was for the PID. The most likely explanation for the massive discrepancy is that the change in the environment was simply much lower impact to the overall system, as the environment in the admittance represented external impedances to the system while the

environment in the PID test was the motor appliance to the PaddleBot controller directly. The change in environment for the PID directly affected the control loop's ability to track reference by reducing the amount of angular acceleration applied to the robot, while a change in the admittance environment only affected the amount of force experienced by any given angular displacement that would feed back into the system and be affected by the internal system gains. Although this form of environmental change is more likely to be of interest for this overall project as it can represent the different materials a human will come into contact with through activities of daily living, it will very likely result in a less pronounced change in the control loop when compared to the direct robot environmental changes.

For the results of these changing environments, although neither of the adaptive controllers are optimal, the results do show clear evidence that online tuning of admittance controller parameters can have a large impact on system response, and that adaptive admittance controllers do have the potential to reduce experienced interaction forces during operation. This conclusion is reached simply by observing the different response plots for when external forces are experienced (such as Figure 4.24 and Figure 4.26). Regardless of which response had better tracking or smaller RMSE values, the effects of the adaptive controller can be seen through the dramatic differences between the constant and adaptive controller plots. For optimal behaviour further tuning and more robust simulations are required, but for the purposes of identifying the feasibility of reinforcement learning-based adaptive control the experiments confirmed the approach as viable. The simulated environment was also overly simplistic by experiencing a force only outside of the standard range of motion, where the range of motion was set as a hard limit. Real world range of motion is much more variable and be more likely to slowly change over the course of an exercise session. The overall adaptive response could therefore be improved with a more realistic environment that could be modelled through collected data. The modelling of this environment would be a large task in and of itself, as subject age and gender demographics would have a large effect on the final result, so multiple different environments must be calculated from multiple diverse sets of subjects. This falls far outside of the scope of this project, but would be greatly beneficial to the field of rehabilitation research for many parties.

4.4.3. Training statistics

Each stage of development used different training session parameters to cater to the current system and reward function. Since each stage was customised it is difficult to compare between methods in terms of success. Therefore only a brief description of each training session used to

generate the results will be provided to highlight any notable features and potentially assist in any attempts to replicate the results.

The stage 1 training was the introduction to reinforcement learning and as such did not employ any advanced techniques. The system was trained for 2000 episodes with the system beginning to improve around episode 1400 and stabilising around episode 1600. For a well designed agent the calculated reward and the Q0 reward should converge to a solution. This behaviour is present within the training shown below, which is encouraging for the design of this agent. As stage 1 utilised end-to-end control and the action space was a torque signal limited between 0 and 20 Nm, it is surprising that the reward increase did not occur for over 1000 episodes when compared to the later training statistics.

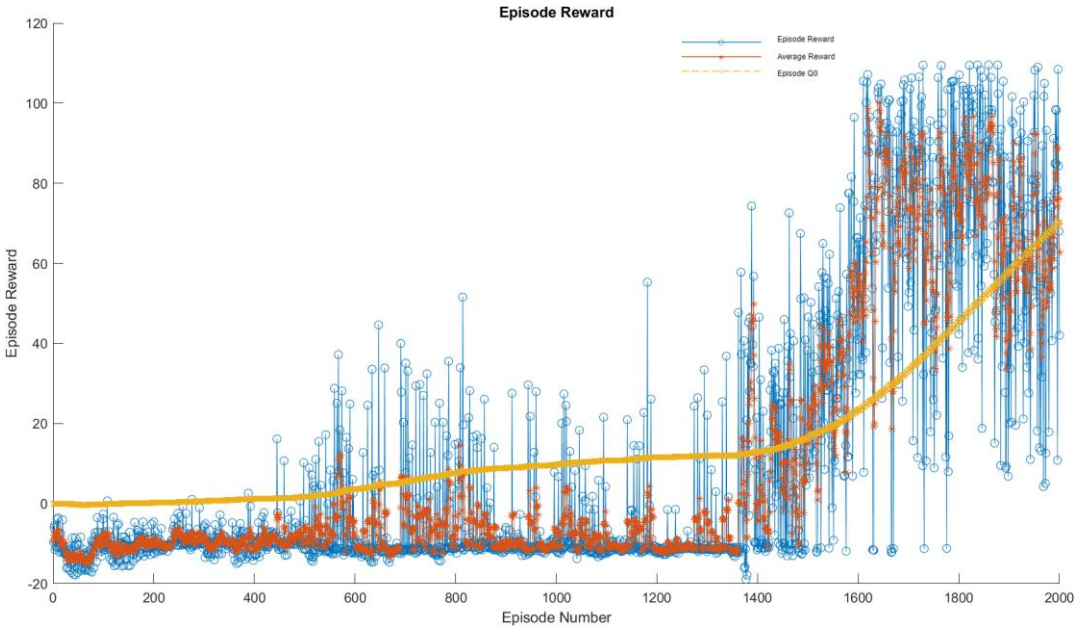

Figure 4.30. Training results for Stage 1 experiment: Direct reference tracking. Episode reward is presented in blue, 5-episode moving average is presented in orange, and Q0 is presented in yellow

The stage 2 training switched the RL algorithm from DDPG to TD3PG and included an early termination condition such that if the average reward of training surpasses 20 then the training finishes. When implemented without the early termination, the same RL system eventually suffered from critical forgetting and was unable to perform the appropriate tasks. The early termination helps avoid overfitting and reduce training time, as well as leads to a converging Q0 value. The reward threshold was selected as the reward function was almost entirely negative components, so only very close tracking would produce positive results, with 20 being a small buffer from 0. This early termination condition was met during the final training attempt after

only 450 episodes, with the agent learning most drastically within the first 50 episodes and between episodes 200-300.

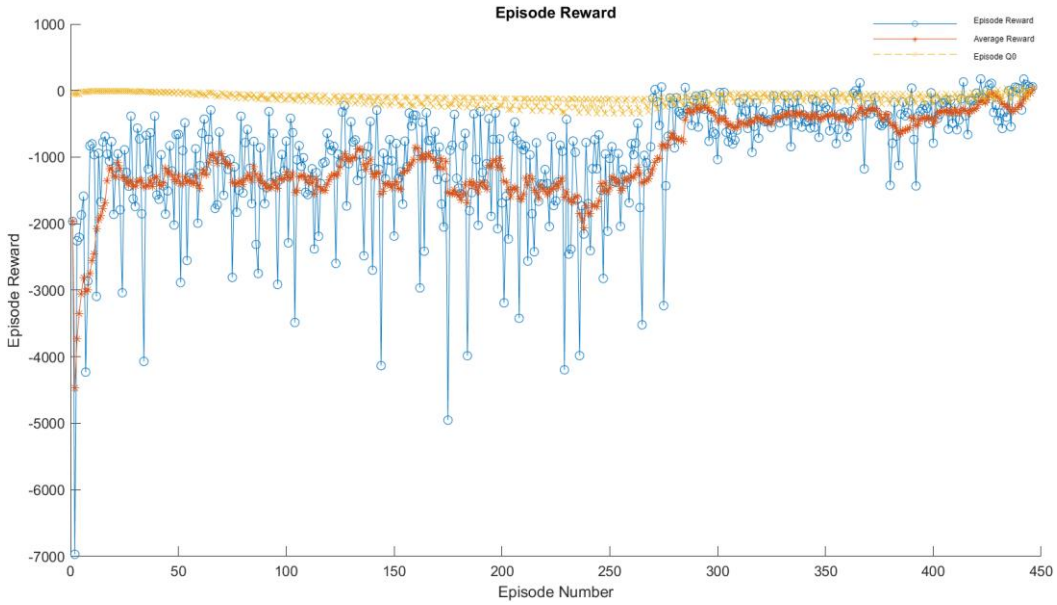

Figure 4.31. Training results for Stage 2 experiment: Adaptive PID controller reference tracking. Episode reward is presented in blue, 10-episode moving average is presented in orange, and Q0 is presented in yellow

Stage 3 showed a more steady increase in cumulative reward than the previous 2 stages, with the reward increasing from episode 0 to 2000 where the system began to remain steady. A drastic drop does temporarily occur around episode 3000 but the system recovers before the end at episode 5000. This is an example of training for too long and risking the system overfitting. The training duration of the adaptive admittance model was set higher than the previous stages as the control loop contained more components interacting and more observations were required for calculation. Results seem to suggest this was unnecessary and a maximum training time of 2000 episodes would have functioned equivalently. The Q0 values did converge for the system at around episode 2000, but later diverged to a point where the plot became distorted so Q0 was omitted from Figure 4.32. This further supports the conclusion that early termination is an essential component of a viable training schematic, but the conditions will change depending on the reward function and the expected behaviour in idealised performance.

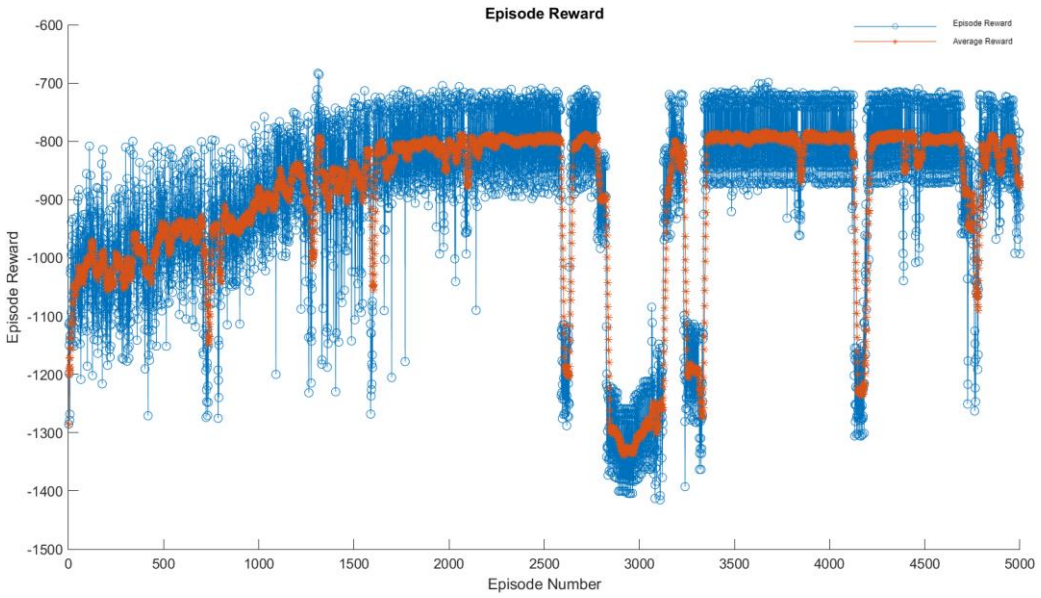

Figure 4.32. Training results for Stage 3 experiment: Adaptive admittance controller reference tracking. Episode reward is presented in blue, 10-episode moving average is presented in orange, and Q0 is presented in yellow

4.4.4.Potential improvements

The results produced have their use but may not be perfect representations of real-world applications. To better transition between simulation and practical robotics some further considerations should be implemented. An example of the measurements presenting an incomplete real-world understanding is the RMSE values inability to distinguish between angular overshoot or undershoot when pushing a joint to the maximum comfort angle. Undershooting the goal will not achieve the stretching sensation required to increase motion but also will not cause any discomfort to the user. Overshooting will pass the point of comfort and cause potential discomfort or even physical damage to the joint. For these reasons a slower response time with notable undershoot is preferred over a fast response time with small overshoot, but this cannot be represented through purely RMSE measurements. The differences could be represented through the reward function for reinforcement learning, increasing the penalty for errors that change sign (generated by overshoot). As the reference may change to be above or below the current position using just the sign will not provide enough information and even a single sign change may be caused by the change in reference signal. For the simulation stage the reference transitions can be predictably scheduled and accounted for, but for later stages of operation where the reference signals are generated from the user bio-signals the determination of what caused the error sign change becomes more difficult. Regardless of the generated reference signal, should two sign changes occur within some predefined timeframe

then overshoot is guaranteed to occur. From a larger perspective, penalising error sign changes within the reward function should lead to a reduced number of oscillations within the system response. Similar arguments can be constructed for determining the negative impact of steady state error. Early stages of rehabilitation will not require overly precise positioning and a small amount of offset from the reference signal will be unlikely to negatively impact the recovery of the subject. An early implementation of this idea can already be seen through the reward function providing positive reward for measurements within a ±5° threshold of the reference signal. In summary, to improve the system to be better suited for practical interactions with a human, the reinforcement learning agent should be adjusted to discourage fast response times and overshoot, instead favouring rise times closer to 1 second and allow moderate steady state errors.

Other issues with the RMSE recordings is their dependence on the reference signal amplitude. Naturally the signals with larger amplitudes produce larger RMSE values, making comparison between different input signals more difficult. This amplitude-RMSE relationship was discussed in section 4.3.2.3 for the adaptive PID controller, but no tests were performed for stages 1 or 3 as it was deemed unnecessary. To help compare between different magnitude signals the values could be normalised before the RMSE values are calculated, which would require prior knowledge of the reference signal and potential operating frequencies. As the types of motions being performed would remain relatively consistent between users this model is feasible, and would be a requirement for later stages in the project where human interaction is beginning and RMSE values need to become more homogenised.

Finally an improvement that could be made to both the adaptive controller techniques would be to incorporate the changing environment into the training sessions. As it stands the agent training retained a constant environment and relied on the natural adaptability of a neural network to predict the best actions. If the agents were trained to appropriately predict non-specific changes to the environment then the final agent would likely be better equipped to perform successfully in hitherto unseen situations. For the PID example, instead of training with a fully functioning actuator the training session could train with the actuator functioning correctly 50% of the time and 50% of the time have signal attenuation of a random amount. This would in theory lead to better ability to reject uncertainties from the system, however for this example the results were already good enough for operation that the increase in training time may be unjustified. For the admittance example the increased variations of environment changes becomes more useful. Changes in the environment stiffness or the boundary states will

be present in all real-world examples and will likely change over time at a more gradual rate than what was implemented in the results section. Although possible to include in the model, this variation was excluded for the purposes of simplicity and ease of implementation. Expanding on the changing environment and including it within the training session may lead to a much better agent able to predict experienced forces rather than learning the one specific linear equation that was used previously. The ability to predict forces from very short interactions will be a crucial feature of the late-project system. Interactions with foreign objects will vary the amount of experienced force (pushing against a sponge vs against a rock) which will occur during general activities outside a controlled environment. The system must be able to provide the necessary assistance on soft grass even if the system was intended for use on smooth hard concrete. As such, learning to adapt to a changing environment is crucial for late-stage rehabilitation. The downside to adding stochastic behaviour into the training session is that training will require longer to observe all the possible operating states and may prevent solution convergence.

4.5. Conclusion

Reinforcement learning can be used to tune controller parameters to better adapt to a changing environment or couple with a human participant at any given point in time. As the system is better able to adjust the coupled system, the overall safety and operation will be improved which is a highly desired outcome for rehabilitation robotics. The testing of this controller parameter tuning was performed on both a PID controller and an admittance controller.

Proportional-Integral-Derivative controllers are used in a plethora of technologies to assist in minimizing differences between desired states and current states. Using reinforcement learning to automatically tune a PID controller with no prior knowledge of the system was shown to be a viable approach to controller design rather than requiring professional experience or the use of an external tuning tool. A twin-delayed deep deterministic algorithm was used for basic reference tracking of a single degree-of-freedom robot designed to help guide a human ankle joint for rehabilitation purposes. An adaptive PID controller capable of adjusting control based on the measurements from the environment can also be produced from RL, although results showed lower tracking performance for the adaptive controller than the constant controller in time-independent environments. This result is believed to have occurred due to the simplicity of the model, with no environmental uncertainty or noise to appropriately adapt to. When a

change to the environment was simulated the constant PID controller failed to track the reference signal to a reasonable degree while the adaptive controller was able to recover and complete the simulation with minimal error. Reference tracking was found to be heavily dependent on reference signal amplitude and frequency, with root mean square error values increasing linearly with both variables when using the adaptive PID controller. The constant PID controller showed less dependence on the amplitude and frequency, but did also increase error recordings for high frequencies. Desired amplitudes and frequencies were relatively small and slight deviations would not cause any harm to the subject or negatively impact the recovery process. As such, small RMSE values were used as the identifier for acceptable performance. Overall both controller approaches resulted in RMSE values small enough to be considered acceptable for the purposes of rehabilitation in simplistic models and environments. The constant controller was shown to be unacceptable for changing environments and the adaptive controller was shown to be acceptable only if it had been appropriately trained.

The use of reinforcement learning in control systems is not a novel application, however there was limited rigorous documentation of its effects in changing environments or input signals. The focus of adaptability of changing environments transformed the regular application of reinforcement learning into a valuable source of information for future researchers. The custom-designed robotic device led to unique results that are naturally only applicable to this project, however the teachings of the chapter are for determining the circumstances in which RL is worth implementing. This will carry to other control plants, as the analysis highlights what factors to consider and what data to collect for any reference tracking robotic system.

Admittance controllers allow the measured interaction forces with the environment to determine how the system will respond to such forces. By relying on reinforcement learning to adjust the gain parameters of the system in real-time, it was theorised that the admittance controller would be able to better conform to a safe operating range of motion and improve the general tracking ability of the closed-loop system. As with the PID controller, human interaction would introduce time-variant features with uncertainty to the environment and would therefore require an adaptive nature to appropriately handle operation. In practice there were no clear improvements between the reinforcement learning-based admittance controllers and the standard switching admittance controller, which alternated gain parameters based exclusively off the measured interaction forces. The comparisons were quantified through the RMSE measurements, taking into account the environment was simulated to restriction free motion. The reinforcement learning method did provide viable results, but does not seem justified in this

particular application as the added complexities of implementation do not improve general tracking ability. Despite these seemingly negative results, the outcome is still beneficial to the overall development as it further guides the rehabilitation device progression path by contributing new information.

For the complex task of human-robot coupling and performing a large variety of physical exercises, these results highly recommend the use of an adaptive controller within the rehabilitation robot to provide the best assistance possible, with reinforcement learning being only one potential technique.

Chapter 5 - Convert sEMG Signals to User-Intended Position via Reinforcement Learning

5.1. Introduction

The design of traditional control systems requires a mathematical model to represent the physical system that is in need of controlling; this will describe how the system interacts with the environment. These models are usually represented by transfer functions or state space representations, but may struggle to accurately encapsulate complex systems that need to be controlled. For these cases, a function approximator such as a neural network is capable of encoding all the necessary information into the system model without the designer having to manually create the characterisation. This allows machine learning to become a very powerful tool in identifying potential relationships between many inputs and many outputs. The electrical signals produced by the human body from muscle activation has an extremely nonlinear, time-dependent relationship with the produced motions and interaction forces, however machine learning is able to take a myriad of input data and predict the output motions without the designer ever rigorously discovering the mathematical relationship. As most relationships in reality are nonlinear and time-dependent, machine learning techniques are widely applicable in all fields and types of data, while control theory often relies on the linearisation of complex systems. This is true for simple hinge joints such as the ankle as well, as the foot can move on other axes with a limited degree. Creating a linear model for even basic ankle movements can be difficult and shows the necessity for machine learning in a rehabilitation setting.

Machine learning has already been used for many forms of EMG classification due to its pattern recognition abilities. A simple artificial neural network (ANN) with just 1 hidden layer is capable of real-time gesture classification (Zhang et al. 2019), but requires the manual choosing of signal features to pass into the network. A convolutional neural network (CNN) can avoid the feature selection step and learn the features directly from each convolutional layer. High accuracy classification has been proven to be achievable with easy-to-use EMG sensors that could be used outside of a laboratory environment (Allard et al. 2016). As CNN technologies use 2D data as the input, most time-domain signals are converted into a time-frequency domain such as spectrograms or wavelet transforms. The number of features that can be extracted from the CNN will depend on the network structure and is generally decided by the architect.

To simplify the network would be equivalent to removing observable features of the input signal which may decrease the classification accuracy if not enough features are observable, however it may also increase accuracy and decrease computation requirements if the system is overly complex and contains too many features. Determining the correct number of features is difficult, but it has been shown that feature reduction can be performed using reinforcement learning (RL) to eliminate the unnecessary features that do not meaningfully improve classification accuracy (Song et al. 2018). Using reinforcement learning for classification is a fairly unusual application, however for optimising a classifier and including information of gesture magnitude (with more numerical results rather than categorical) it becomes more appropriate.

Song et al. (2018) acts as the primary inspiration for the experimental work performed in this chapter, although a majority of their work in agent deep Q-network (DQN) design seems to stem from Janisch, Pevný, and Lisý (2019) (or a previous version of the same publication), which in turn is based on the paper by Dulac-Arnold et al. (2011). Dulac-Arnold focused on the sequential feature choosing for classification. EMG data will be converted into a feature data vector such that the RL agent is able to find the minimal amount of features per EMG signal that can be used to classify a basic gesture. Once this behaviour is learned an RL agent will be utilized to determine precise motions with the fewest features possible, and then compared to alternative gesture classifiers to validate the approach of RL for classification.

The experiments performed utilised upper limb EMG data for classifying hand motions, rather than lower limb data for classifying ankle positions. There are key differences between these types of classification as the ankle motions will be more limited than hand motions, however it will be enough for the purposes of identifying the stage of gait the user is currently completing. Determining the exact degree of ankle flexion or extension may be beneficial in some cases, but it is more difficult to predict accurately, and position of gait cycle maps well to basic hand gestures. It is for these reasons that hand gesture classification was deemed similar enough for the purposes of feature extraction and model simplification. Collecting enough data to train a reliable network with no biases is a monumental task itself, which when combined with the obstacles of the COVID-19 pandemic led to this option not being pursued and instead using the readily available large dataset for hand motions.

5.2. Methodology

5.2.1. Electromyography Data Collection & Pre-processing

Constructing an artificial intelligence for classification purposes requires a large amount of data to appropriately train the respective model. Due to the natural variances within human EMG production the data collection stage becomes especially important to guarantee the final product is able to learn these variations. To best cover this complication, data should be collected from a wide variety of subjects from differing demographics at various stages of exhaustion. Unfortunately collecting a new dataset became logistically and ethically infeasible due to the COVID19 pandemic and its restrictions. For this reason an existing EMG database was used for training and testing purposes, which was provided by MATLAB (Mathworks 2022a), but originally collected by Goge and Chan (2005) with the intention of normalising data to compare different forms of pattern recognition (Chan and Green 2007). The EMG data was collected from the right forearm using 8 Duo-trode silver chloride electrodes (Ag-AgCl) attached to the skin surface of 30 different subjects. Each subject performed one trial which consisted of executing 7 distinct hand gestures four times each in a random order. This trial was repeated 6 times per subject within one session, and returned on 3 later days to repeat the session. The provided dataset contains 720 EMG trial data files that each contained approximately 94 seconds of 8-channel EMG recordings. Each gesture was performed for an approximate 3 second period on four separate occasions per trial, with 5 seconds of no actions at the beginning and end of recording to avoid data cut-off. Each file was sampled at 3 kHz within a bandwidth of 1-1000 Hz.

No mention of the duration between trials is given, so it is unclear whether subsequent trials will suffer from any forms of exhaustion or muscle fatigue. However since the exercises being performed are non-intensive and the study is not looking at the effects of muscle fatigue it is assumed that there was enough time between trials to completely recover and no fatigue effects are present in the training data. For rehabilitation purposes this may lead to a biased training set, as the system must be able to similarly categorise gestures performed at full strength and partial exhaustion. This missing element in available data will be ignored for the remainder of this chapter, however any future data collection must take this into consideration for an all-encompassing training dataset. Although the data collected from the upper limb cannot be used to train an ankle intention decoder, the viability of EMG classification can be confirmed and documented to justify future expansions and experiments for the rehabilitation project.

The Chan dataset was used within the MATLAB example “Classify Arm Motions Using EMG Signals and Deep Learning” which provided proof that classification was possible using a long

short-term memory (LSTM) recurrent neural network structure. With the final goal of the chapter being to develop a real-time classification method from EMG signals to a desired joint position, the underlying technology of audio-based classification was adapted to EMG signals due to their similar signal input behaviour. Audio classification converts a time-domain digital signal into a spectrogram displaying the frequency-domain features as a 2D image, then uses a CNN as an image classifier. This technique can be applied almost directly to EMG classification, however the addition of multiple signal channels increases the information density of the input signal and some adjustments must be made.

5.2.1.1. Data Filtering & Selection

For the purposes of gesture classification each input signal must correspond with one single gesture, and as such each EMG trial file was split into individual EMG motion files. As previously mentioned, each EMG trial contained 7 gestures performed 4 times each, with empty recordings at the beginning and end of the file for padding purposes. As such an expected 21 600 motion files should be produced, each falling into one of the following motion categories:

TABLE 5.1 – GESTURE DESCRIPTIONS

Gesture	Description
EndData	The additional padding data found at the beginning and end of each EMG file
HandClose	The act of closing the right hand
HandOpen	The act of opening the right hand
WristFlexion	The act of bending the wrist towards the anterior of the forearm
WristExtension	The act of bending the wrist towards the posterior of the forearm
Pronation	The act of rotating the wrist/forearm to face the ground
Supination	The act of rotating the wrist/forearm to face the sky
Rest	No action

Each trial datafile also had paired labelling data describing the time period for each gesture, along with the appropriate gesture label. Using these time periods to automatically split the data, results show that only 18199 data samples were produced. This means either the completion of each gesture four times per trial was not strictly adhered to, or when the same gesture is performed back to back the time period provided combines them into one action.

Some of the split gesture files had much longer time periods which suggests the gesture was performed multiple times or for a duration longer than instructed. Although this does not confirm some motion files contain repeating gestures it does reinforce the assumption, as well as raises concerns of validity for the unexpectedly long motion files. Regardless of the reasoning the long motion files were filtered out of the datastore as each input file must share the same dimensions to be passed into the neural network and normalising data with large variation can lead to a decrease in learning capability. Additionally, padding data to match the larger data samples also has detrimental effects on training, therefore it is best practice to remove the questionable data from the datastore. All motion files that extended past the duration of 10000 ticks (approximately 3.3 seconds sampled at 3 kHz) were removed from the datastore. All remaining files were padded to exactly 10000 ticks to normalise the EMG motion data to a form that can be readily processed. The shortest file contained 5629 ticks of EMG data and was produced by a wrist flexion gesture. As this file required 4371 ticks of padding data the final file would contain 43% padding. As every other file would require less padding, this ratio is considered acceptable for the purposes of this experiment. Additionally, it has been documented that the position of the EMG sensors during initial data collection would not be able to reliably record the motions of supination and pronation. For this reason all files corresponding to supination and pronation, along with EndData which represented no motions, were also removed from the reinforcement learning datastore. However, supination and pronation data was used for the training of the CNN classifier, where both categories were combined and labelled as “Unknown” to allow the classifier to default to a setting when previously unseen or potentially meaningless data is encountered.

The final data selection stage reviews the necessity for all 8 channels of EMG data per gesture. Goge and Chan (2005) discuss the effects of each EMG channel’s effect on classification accuracy and determine that the most important channels are channel 3, 5, 6, and 8; channel 1 is deemed the least important. The original paper used auto-regression coefficients to parameterise the data and used linear discriminant analysis for the classification. Since different methods are being used for classification the effects of each channel may be different, however the channels described as “important” were produced by the electrodes closer in proximity to the largest part of the muscle, giving the results a biological explanation. Only the four most important EMG channels were extracted from the EMG data to allow comparable EMG data to be personally collected within the UTS laboratory which is limited to 4 electrode sensors. Data reduction also has the added benefit of simplifying the data used in the system such that real-time classification

will require less computational power and may run better than using the full 8 channels. The predicted loss to classification accuracy will be minimal, with only a 2% difference outlined in the original paper.

The final datastore file count to be used for the reinforcement learning component of the experiment was 9904 with each file containing only 4 channels rather than the original 8. This is enough data to use in machine learning and establish a viable training, validating, and testing dataset. The motion gestures, along with their file count, are provided in the table below:

Table 5.2 – Gesture File Distribution for Classification

Gesture	HandClose	HandOpen	Rest	Supination	Pronation	WristExtension	WristFlexion	Total
Count	1870	1959	2058	0	0	1906	2111	9904
%	18.9%	19.8%	20.8%	0%	0%	19.2%	21.3%	100%

5.2.1.2. Data Processing for Convolutional Neural Networks

The remaining dataset must now be converted from raw time-domain signals into a 2-dimensional image to be used by a convolutional neural network for its image recognition capacities. The image produced is a time-frequency representation of the data to show the frequency domain features of the data and how they evolve over time. These representations are generically called scalograms, and can be produced through several different methods. One common method to achieving this is to use a continuous wavelet transform (CWT) on the time-domain signal and plot the absolute values from the CWT coefficients. An alternate method is to use a sweeping window function over the time-domain signal to extract the frequency-domain information from the signal using the fast Fourier transform (FFT), as seen in audio recognition. Since the EMG classification approach was inspired by audio recognition and CWT had been shown to work for electrocardiogram (ECG) data, both approaches were attempted as a comparison to determine whether both methods were viable for EMG data.

The differences between Fourier transforms and wavelet transforms is that a wavelet is localised in both time and frequency while a Fourier transform is localised entirely within frequency. In general, the Fourier transform can be considered a unique case for wavelet transforms where the resulting signal is represented as the sum of sinusoids while in a generic wavelet transform the resulting signal has no such guarantee.

A) Continuous Wavelet Transform

The wavelet transform was performed using tools provided by the Wavelet Toolbox (Version 5.6 for MATLAB R2021a). The transformation makes use of a filter bank which is simply a collection of bandpass filters of equal magnitude but different peak frequencies. The number of filters contained within the filter bank will be determined by how many voices per octave the user desires. An octave is simply a measurement of frequency which increments by one for every doubling of frequency, and as such a higher "voices per octave" setting will result in the frequency analysis to be separated into smaller intervals. Each filter is known as a wavelet and is used for the CWT calculation through convolution.

For the experiment the wavelet shape used was the Morse wavelet with a time bandwidth of 60. Both these values were the default and left unchanged due to their success in ECG classification. The voices per octave were set to 12 for similar reasons, although later tests showed no loss of quality when this value was reduced to 8. The produced CWT filter bank is displayed below, which shows the filters that will be convolved with the EMG time-domain signal to produce the resulting spectrogram.

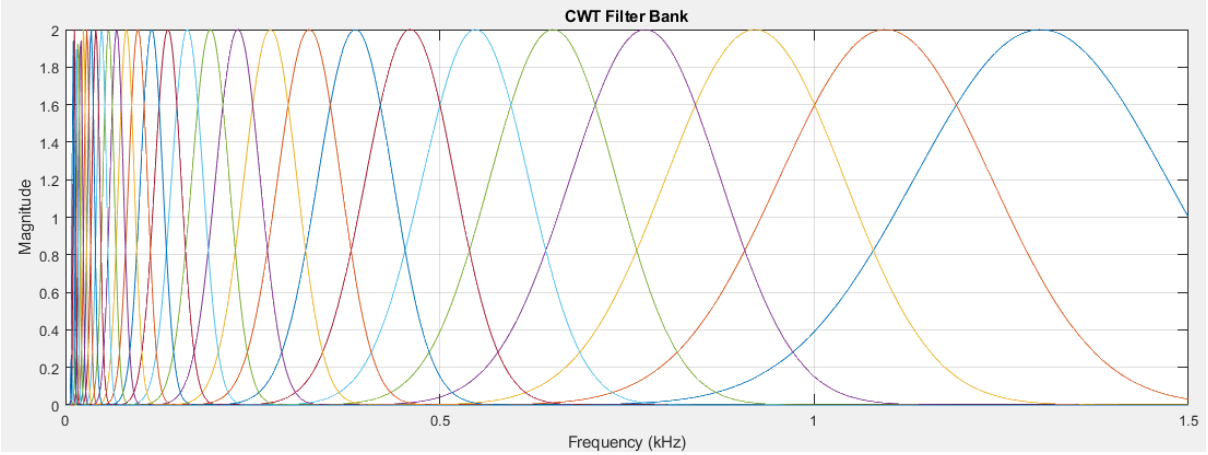

Figure 5.1. Continuous Wavelet Transform filter bank used to produce 2D images representing the EMG data in the time-frequency domain

B) Fast Fourier Transform

The Fourier transform is the mathematical transformation that allows a time-domain signal to be represented in the frequency domain to identify all the varying-frequency components. When the Fourier transform is applied to a full-length signal the frequency analysis is produced, but no information about the temporal distribution of each frequency component is generated. Simply put, if a signal has a high-frequency component near the end of a signal, as may be

present in EMG motion data, then the corresponding Fourier transform would display the high-frequency component but would be unable to determine when in occurred during the signal sample. To combat this, the Fourier transform can be applied to discrete windows of the signal to localise the frequency components.

The fast Fourier transform (FFT) is an efficient application of the Fourier transform for discrete signals and is used for the audio feature extractor provided by the Audio Toolbox (Version 3.0 for MATLAB R2021a). The feature extractor can output several sound-based spectrograms such as the Bark spectrum or the mel spectrum to be used for audio classification. However due to EMG signals not being audio-based signals it is more appropriate to use a simple linear spectrogram to best represent the spectral output. A Hann window function was used with a window length of 25 milliseconds that would shift by 10 milliseconds across the time axis to complete a convolution multiplication with the entire EMG signal. The frequency range of 0-500 Hz is displayed in 257 frequency bands, although this is compressed to 224 pixels during image production. The data was down-sampled to 1 kHz which sets the Nyquist frequency to 500 Hz, allowing the frequency range to display all necessary information of the initial signal.

For audio recognition only 50 frequency bands are generated for a Bark scale spectrogram which produces high fidelity. This implies a reduction of frequency bands could be imposed with little negative impact on the EMG classifier accuracy, although it will not be attempted at this time. A reduction in voices per octave for CWT or down-sampling for FFT could achieve this if future experiments struggle to generate frequency analysis fast enough for real-time robotic control. A comparison of the produced CWT spectrogram and the FFT spectrogram can be seen in Figure 5.2. The FFT spectrogram displays frequency information between 0 Hz and 500 Hz linearly along the y-axis, while the CWT spectrogram displays frequency information between 0 Hz and 1.5 kHz logarithmically, as there are more low-frequency filters than high frequency filters.

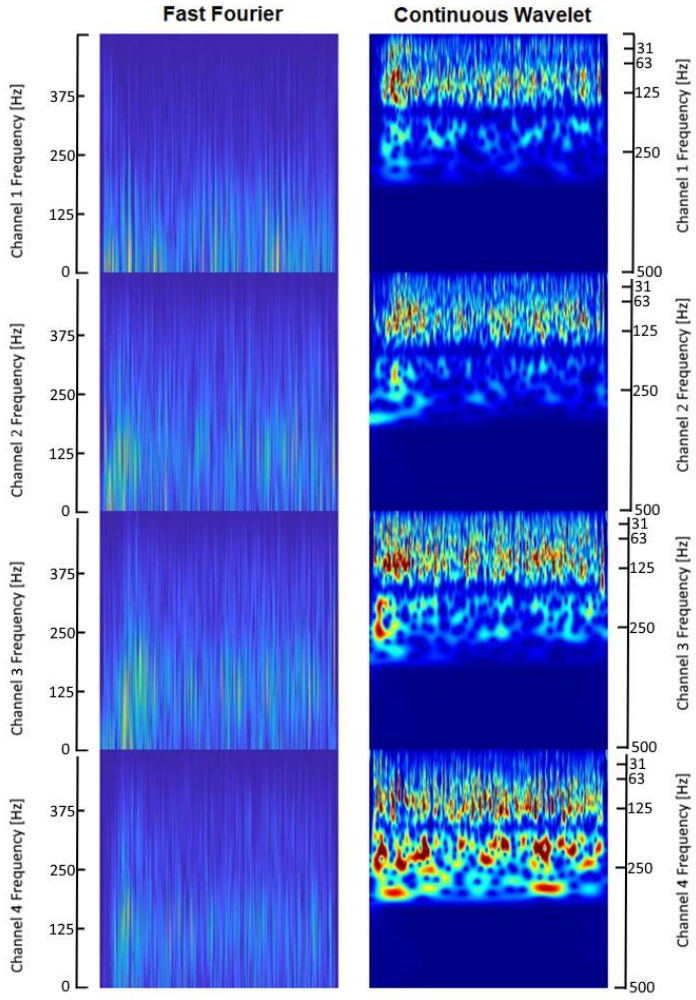

Figure 5.2. Produced spectrogram of the 'Hand Close' motion using fast Fourier transform (left) and continuous wavelet transform (right). The four EMG channel signals are presented stacked vertically, with the x-axis representing time and the y-axis representing frequency distribution.

5.2.2. Feature Reduction via Reinforcement Learning

Once all EMG data has been processed and converted into spectrogram image data it can be used as the input to a convolutional neural network used as a classifier. Instead of selecting specific features from the data such as the mean absolute value or root mean square, the CNN is able to produce deep features that a human may not be able to recognize as significance.

5.2.2.1. CNN architecture & Feature Extraction

The CNN architecture was copied from the speech command recognition example provided by MathWorks (Mathworks 2022d) which is constructed using an image input layer followed by three convolutional blocks. A convolutional block contains a 2D convolution layer, a batch normalisation layer, a rectified linear unit (ReLU) activation function layer, and a max pooling layer. Each layer contains padding to produce an output retaining the same dimensions as the

input, except for the max pooling layer which reduced both dimensions by a factor of two. The number of filters per block, and thus the number of feature maps produced per convolution block, were equal to 12, 24, and 48, respectively. Two more convolutional blocks follow these layers, however both have 48 filters each and no max pooling layer follows the ReLU layers. Finally a maxpool layer is used to find the maximum value for every row of the remaining feature maps. These values are passed through a 20% chance dropout layer and are then classified using a weighted output layer with the softmax activation function. Initial testing produced extremely high classification rates which, when combined with a lack of confidence in expertise, led to no alterations of the CNN architecture hyperparameters. The GoogleNet architecture was also adapted and tested for the purposes of classification, however due to the much more complex network structure the training sessions required much longer to complete. The final verification accuracy was also slightly lower than the speech-based network and as such the GoogleNet architecture was not pursued further.

Training options also remained relatively unchanged from the audio example, with the only noticeable change being a reduction in the minibatch size from 128 to 64 due to memory issues. The adaptive moment estimation (adam) algorithm is used with an initial learning rate of $3x10^{-4}$ that drops by a factor of 10 after 80% of training has concluded to help achieve more precise gradient descent. The training data used is a subset of the datastore discussed in Table 5.2, however supination and pronation data is combined into an “unknown” category to encompass any previously unseen gestures that may not be identifiable through the current EMG sensor configuration.

Once the general network is shown to be a reliable classifier for the spectrogram data it can be used to extract the features that it uses to classify the gestures. As each filter for each convolutional block can be considered as producing a 2D feature map, initially these filter maps were planned to act as the features. Unfortunately no reliable method could be found for representing an entire feature map by a single scalar value. If the maps were to be used in their entirety with each pixel activation being accounted for there would be a total of over 1 million features if taken from the max pool layers and over 3.6 million if taken from the convolution layers. Even if each filter is averaged to find the average feature map of one block, the number of features would remain in the hundreds of thousands. As all of these approaches would be infeasible for observations into an RL agent, an alternative feature extraction method is required.

An additional fully connected layer is added to the output of the dropout layer and is labelled as the ‘feature vector’ layer. The layer contains 100 neurons that, when activated, will produce scalar values that are shaped through the previous layer features. The activation values can be extracted for every image input, allowing a single image sample to produce a single vector containing the number of desired features. To create the feature vector database each existing spectrogram image corresponding to a valid gesture must be fed through the CNN that was proven to be a reliable classifier. However since the neural network architecture has been changed the system must be retrained and validated to confirm functionality has not been supressed to the point of unreliable classification. Once the feature vector database has been established it will be used as the input to a reinforcement learning agent environment.

In the current experiment only 100 features were produced, compared to the 288 features implemented by Song et al. (2018). Alternative tests were performed with a higher feature count of 180, however this led to a complete deterioration of training and a classification accuracy equivalent to random guessing. These tests were performed using the FFT spectrogram images and were not compared to the CWT images as it was deemed unnecessary. Tests proceeded forward using only 100 features to confirm the proof-of-concept that was being attempted in feature reduction. Future tests can be performed with higher features if the input data is detailed enough to require higher features. The position of the feature vector layer was also experimented with to try and improve the quality and relevance of the features being extracted. The fully connected layer was placed both before and after the dropout layer to try and find the most robust feature set possible. A dropout layer is generally used to improve the reliability of the classification, allowing the network to experience states which may be subject to errors or small discrepancies in the input data. Intuitively it was believed that the feature vector should occur before the dropout layer as it would be less subject to randomisation and have more complete information. When this structure was tested, training results showed a complete inability to learn from the input images, while training performed reliably when the feature vector layer was positioned after the dropout layer. Additionally, the effects of including a ReLU function with the feature vector layer were examined to determine its inclusion necessary.

Once the DQN network structure was fortified the final feature vectors were collected by passing every valid CWT and FFT spectrogram through the trained network to activate the neurons within the feature vector layer. These activation values were extracted and saved as individual feature vectors, along with the correct gesture label appended at the bottom of each column vector.

Each of the vectors produced by the CWT and FFT spectrograms were stored within their appropriate feature matrix. With a total of 9904 valid gestures each matrix has the dimensions of 101x9904. Most tests utilized the CWT feature matrix as it was created before the FFT features were extracted. Once the FFT features were also created the CWT was still the primary feature source as it had shown a higher accuracy during the CNN classification and was therefore believed to be slightly more reliable.

The feature vector layer within the CNN did have a ReLU activation function at its output, however the features extracted occur before the function is employed and as such many features have negative values. Finally, the feature values were mean-normalised to guarantee each feature vector had a feature mean of 0 and a standard deviation of 1.

$$F' = \frac{F - \mu_F}{\sigma_F} \tag{5.1}$$

By standardising the observation data the effects of gradient descent become much smoother and will likely converge faster to the optimal solution (Narasimhan 2021). Unity normalisation was also attempted in previous versions (all features were compressed between the values of 0 and 1), however using this data was eventually phased out due to mean normalisation being referenced more frequently within machine learning literature.

5.2.2.2. RL Agent Reward Shaping for Feature Reduction

Classification is generally better suited for machine learning in the form of supervised learning, while reinforcement learning is more adept at environment exploration or problems that could in theory be solved through trial and error. As such, the use of RL to simply classify human gestures is unnecessarily complex; the goal of using reinforcement learning for this experiment is to optimize the supervised learning that occurs within the previously established CNN. The extracted feature vectors that contain every feature value are used as the input to the RL network to try and learn to predict gestures based off as few features as possible.

A single feature vector $F = [f_1\ f_2 \dots f_n]$ is chosen at random to be used every episode, while a growing subset $\bar{F}$ is used to represent only the features that have been observed so far within the episode. A masked feature vector $x \in \mathbb{R}^n$ contains the feature values that have been observed so far within the episode in the correct index position, with zeros everywhere else. To distinguish between an unobserved feature and a feature with the value of 0, a second mask

vector $m \in \mathbb{R}^n$ is used to identify whether a feature has been observed (represented by value 1) or not (represented by value 0) at any given point in time.

$$x_i = \begin{cases} f_i & f_i \in \bar{F} \\ 0 & otherwise \end{cases} \quad (5.2)$$

$$m_i = \begin{cases} 1 & f_i \in \bar{F} \\ 0 & otherwise \end{cases} \quad (5.3)$$

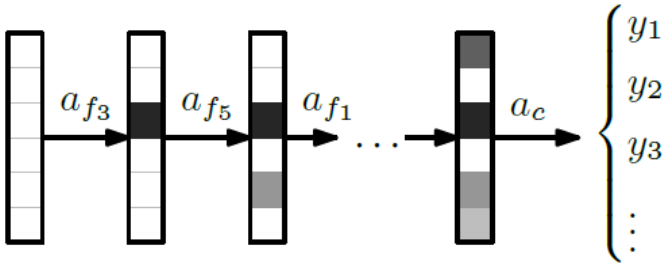

Figure 5.3. Illustration of feature observation sequence. Each individual feature within the masked feature vector has the chance to be observed through action a_f. A classification is performed at the end of the sequence through action a_c. Each action is based on the current masked feature vector state. Image sourced by (Janisch, Pevný, and Lisý 2019)

A combination of x and m act as the observations to the RL agent to determine the action that will be taken. A single action is output with a discrete value within the action range $A = \{1: n + g\}$ where n is the total number of features, and g is the total number of possible classifications. The action space can be subdivided into two subsets: A_f which represents the action of observing a new feature, and A_c which represents the action of classifying the feature vector as a specific gesture based off the currently observed features. At each time step the agent will be able to choose either to observe an individual feature from the currently loaded sample F, or it will choose to predict the motion based on its current observations. The feature f_i will be added to $\bar{F}$ if the corresponding action $a_i \in A_f$ is the selected action. The gesture g_i will be compared to the true gesture label $y \in \mathbb{N}$ and the episode will be terminated if the corresponding action $a_i \in A_c$ is the selected action.

As the agent observes more features its observations become richer in information to base its prediction on, and will be able to predict the correct gesture more frequently. To find the minimal the number of features required to appropriately classify, observing a new feature will have a cost in the form of a negative reward value. A negative cost is also applied to incorrect classification to discourage the action of misclassifying the input gesture from the correct gesture label y. A baseline reward function is displayed below:

$$r_t = \begin{cases} 0 & a = y \mid a \in A_c \\ -1 & a \neq y \mid a \in A_c \\ -0.001 & a \in A_f \end{cases} \quad (5.4)$$

The reward function is designed to be completely negative which will push the agent to end the simulation as fast as possible to avoid accumulating the negative reward. The cost of observing a feature should be less than the cost of incorrect classification to guarantee observing and correct classification has a higher average return than immediate random classification. The only exception to this is if a previously observed feature is reobserved, effectively not gaining any new information about the environment and current state; if the action taken corresponds to a feature already contained within the observed subset $\bar{F}$ then the system must respond differently. These actions are to be considered illegal and were removed from the potential action space in the original paper. However the original paper was written using python 3.6 and this option does not seem to be available for the MATLAB 2021a Reinforcement Learning Toolbox, so the illegal actions must be accounted for through other methods. There are three potential approaches to restricting or discouraging the reobservation of a feature:

- Increase the reward cost of an observation if that feature has already been observed. This approach should allow the system to converge to a non-repeating solution as it discovers the higher rewards available for unique observations.
- Redirect the illegal action to a legal action with an increased cost. This approach has the added benefit of continued state space exploration even from illegal actions, but will slow down learning the benefits of unique observations and runs the risk of relying on the action redirect to find new observations.
- Illegal actions receive a large one-time penalty cost and cause a simulation termination. Ending each simulation early will drastically reduce training time but will also slow down state space exploration, as each training episode will not collect experience on as many state-action pairs.

Smaller test experiments were performed utilizing each of these approaches for a relatively small number of episodes before implementing the first and third option for the final experiment. The results of these experiments are discussed in 5.3.3.1. The reward value is calculated each time step and transferred to the agent along the ‘reward’ line, while the early termination condition is transferred along the ‘isdone’ function shown in Figure 5.4.

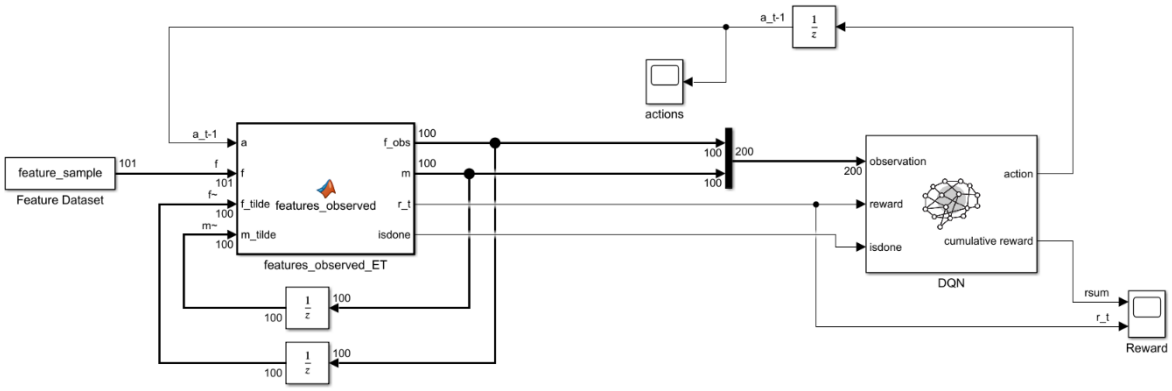

Figure 5.4. Simulink model for the final EMG classification experiments via reinforcement learning

Various attempts at alternate reward functions were attempted to try and better shape the agent behaviour. Replacing the neutral +0 reward with a +1 or +10 reward for correct classification should not drastically change the training, as a single positive reward at a terminal condition should not open any exploitation loops and should simply increase tolerance to accrued negative reward along the path to the terminal state. The tolerance threshold can be calculated by finding the average reward return for classification along with the cost of feature observation. For the classification task involving five gestures the probability of randomly correctly classifying a gesture is equal to 20%, making the chance of incorrect gesture classification equal to 80%. With no learning and using the values of p=0.2 the expected reward can be calculated as:

$$r_{avg} = r_{true} * p + r_{false} * (1 - p) \tag{5.5}$$

Using the values supplied in (5.4) equate to an average reward of $r_{avg} = -0.8$. Calculating the average reward if the cost of observing features is included requires knowledge of how the additional information would impact the classification rate. The new classification rate can be labelled p_n while the cost of feature observation can be labelled as c. For the system with 100 potential features and a cost of -0.001 per feature, the maximum cost of feature observation is calculated at $c_{max} = -0.1$. To determine whether feature observation is mathematically viable, the new classification rate p_n must result in the new expected reward to be greater than the expected reward from complete random guessing:

$$r_{true} * p_n + r_{false} * (1 - p_n) + c_{max} > r_{avg} \tag{5.6}$$

$$0 * p_n - 1 * (1 - p_n) - 0.1 > -0.8$$

$$p_n > 0.3$$

This inequality proves that as long as observing all features increases classification accuracy from 20% to greater than 30% then the expected reward will increase and the RL agent should, in theory, converge to the solution of observing all features rather than relying on random guessing. These results bolster confidence in the current reward function being appropriately balanced for the task at hand. From the results of the CNN classifier it is believed that the classification probability will be much larger than 30% accurate and as such the baseline reward function was retained throughout the remaining experimental trials. For the example mentioned earlier, by adding a positive reward to correct classification, the minimum probability of classification will decrease (a reward of +10 for correct classification will lead to $p_n > 0.2091$ in (5.6)). Although this would simply mean that the system should learn to observe features even for a worse functioning classifier, this is antithetical to the goals of the project, and as such these actions are not taken.

Note that these equations/inequalities do not account for any changes made in reference to the system reobserving features, and instead assumes each feature can only be observed once and for the same price. Any alterations from this behaviour will result in the predicted models above becoming inaccurate.

5.2.2.3. RL Agent Architecture

Due to the nature of the feature selection task being performed, each action is distinct from the adjacent actions in action space and can each be represented by a discrete value within a finite set. This structure is best suited to use a Deep Q-Network (DQN), as Q-learning methods are more sample efficient, yet less stable and more prone to learning divergence (Szepesvári 2010). The DQN network acts as a function approximator for the Q-values of every state-action pair that the agent experiences, with the Q-values produced at the output of the network. The observations from the masked feature vectors act as the input layer to the network, with a total of $2n$ neurons to represent each feature value and its corresponding 'read/unread' state. The number of neurons per hidden layer was set to the mean of number of actions and number of observations. For the experiment with 100 features aiming to classify 5 unique gestures:

- The input layer required 200 neurons to incorporate every feature value and index
- Each hidden layer required 100 neurons
- The output layer required 105 neurons to incorporate every Q-value

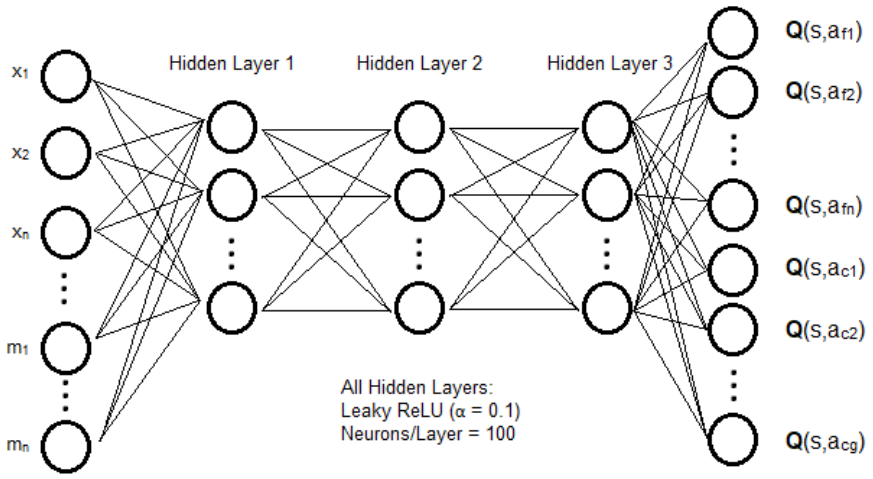

Figure 5.5. Deep Q-Network Architecture for the feature reduction classifier. The Q-values for each action is produced at the output of the network, including both feature observation actions (a_f) and classification actions (a_c). The highest Q-value will determine which action the network performs once trained and acting on-policy

A mathematical function was added to the output of the Q-values to restrict their values. As the reward function is composed entirely of negative components it can be guaranteed that the reward will never exceed zero. As such, the Q-values were clipped to prevent predictions greater than zero. Without this clipping the target Q-network would produce unrealistic expectations and would drastically reduce training performance. The mathematical function was composed of three layers including a scale by -1, a ReLU, and a second scale by -1. As the ReLU function can be written as $f(x) = \max(0, x)$, combining the scale layers results in the function $f(x) = -\max(0, -x) = \min(0, x)$, which effectively cuts out all positive values from occurring and reducing them to zero: the true maximum of the system.

The architecture was predominantly aimed to replicate the original paper, which used 3 hidden layers. Alternative structures were attempted initially, as 3 hidden layers seemed unwarranted for what was believed to be a simple task and excessive neurons can lead to overfitting. Primary tests acted with 1 or 2 hidden layers with all other parameters kept the same. No successful results were generated during these tests and as such the structure shifted to match the paper that had already been proven to function as desired. The observations used also initially did not utilize the mask vector m as it was believed to be redundant data, with index position being inherently contained within the x vector. The chance of a feature being exactly zero was assumed highly unlikely and worries of the negative impacts of increasing the observation space led to the decision to not include the second mask vector. However after multiple failed experiments through hyperparameter tuning it was decided best to simply replicate the existing

paper as best as possible to reduce potential fault variables. Mathematically since m is perfectly correlated to x, although the observation space does increase the number of achievable states remains the same; i.e. it is impossible to reach the state where feature f_i is observed, but the mask status m_i be set to 0. It is therefore theorised that exploration during training will not be negatively impacted by the inclusion, only through the extra computational power required to pass additional values is the system negatively impacted. With this, the inclusion of the secondary mask vector m was justified.

The only conscious difference between the agent proposed by Janisch, Pevný, and Lisý (2019) and the final implemented agent was the use of a leaky ReLU activation function in place of a regular ReLU activation function on each of the hidden layers, which allows for negative values to be propagated through the network without being completely zeroed out. This difference was kept as it is believed that there are no downsides to using a leaky ReLU layer as long as the leaky factor is kept close to zero, while also reducing the chances of the network suffering from the vanishing gradient problem. As a network with more layers is more susceptible to the vanishing gradient problem, the leaky ReLU layers were implemented with a leaky factor of 0.1. Several control experiments were performed to try and identify any meaningful differences in outcome between the two activation functions, but no results presented anything immediately identifiable.

5.2.2.4. Hyperparameter tuning

The hyperparameters of a reinforcement learning agent are simply the variables that are adjusted outside of the training period to best suit the goals of each particular optimisation problem. Determining appropriate hyperparameter values will be context-dependent and proportional to the system as a whole. Although there are advanced approaches to hyperparameter optimisation such as genetic algorithms (Kiran and Ozyildirim 2022), there is no definitive method for selecting the optimal hyperparameters and inevitably relies on the designer's experience. There have been attempts at using reinforcement learning to tune the hyperparameters of reinforcement learning (Neary 2018) but this is well outside of the scope of the project at hand. Majority of the hyperparameters remained unchanged throughout the testing period as many were considered to be unimportant to the system and tuning every hyperparameter would severely limit progression. Several hyperparameters, such as the optimisation algorithm and accompanying momentum, were left either as the MATLAB default or as the value provided by the guiding papers. MATLAB default values are generally well-suited

for many problems and, when mixed with a lack of confidence in understanding these topics, retaining default values is a reasonable approach. The default optimiser algorithm in MATLAB is the adam optimiser, however the root mean square propagation (rmsprop) algorithm was specified in the guiding paper. Early testing stages used adam before this discrepancy was noticed, but once changed there were no apparent differences. The details of the optimisation algorithm were deemed unnecessary for project progression and were simply copied directly from the source paper.

For the hyperparameters that are better understood and believed to need customisation between reinforcement learning projects, small comparison experiments were completed to measure the effects each parameter had on the training results of the system and the overall classification accuracy of the final agent. Due to the large number of hyperparameters and the computational and time-intensive nature of reinforcement learning, these small experiments were shortened to prevent project delay. Although this does reduce the significance and reliability of the experiments the primary goal was to identify any effects these hyperparameters had, rather than the elaborate detail of the effects.

A. Learning rate

Throughout each experimental iteration several hyperparameters were changed to better suit the current reward function, architecture, and training options. One of the most important hyperparameters in reinforcement learning is the learning rate. A larger learning rate will transition towards an optimal solution faster but runs the risk of never finding the exact optimal solution. In extreme cases this may also lead to instability in the training and will prevent convergence altogether. Smaller learning rates can avoid these risks, however convergence time increases and the risk of falling into a local minima. For feature selection and classification, the local minima of incorrect classification would equate to observing no features and guessing gesture classification randomly. This is due to the fact that discovering the correct features leading to correct classification is a very particular state space path that the agent may not discover often during exploration. A common learning rate used in similar problems is within the magnitude of 10^{-3}, which was the default value used for experiments. Some experiments were performed using values of the magnitude 10^{-2} but no meaningful information was produced due to changes in the reward function also being made at the same time. Results did show the increase in learning rate seemed to cause the training to result in a non-optimal

solution more frequently, which matches the theoretical results expected. Majority of the experiments performed were working under the assumption that majority of errors were effects of the environment setup and as such the learning rate was conventionally left constant with a value of 0.001.

As the learning rate determines a percentage of collected reward that will update the network policy it is also important to mention the reward discount factor. The discount factor reduces the values of rewards in the future, however in the case of a purely negative reward the discount factor actually reduces the cost of later actions. As each observation should be equal in cost across an episode the discount factor is set to unity. Tests were performed with a lower discount factor of 0.99 and 0.95 to observe whether this would encourage more exploration or agent longevity, but no changes to the action outputs seemed to be produced from this hyperparameter alone. This may be due to the relatively short lifespan of the agent, with at maximum 100 future steps but with early termination step count being predicted less than 10 steps. Values this low are not enough for the changes to policy parameters to make any clear difference.

B. Target network updating

A second important set of hyperparameters are linked to the target DQN network and determine how fast this network updates. Details of how the DQN network updates towards an optimal solution are discussed in 2.4.1.1. The hyperparameters for target smoothing and target update frequency can have a large impact on what solutions the primary network converges towards. Initially, before Q-clipping was implemented through the network, target Q-values were increasing exponentially for the beginning of the training session before eventually reducing to more reasonable (but still higher than expected) expectation levels. This behaviour was mentioned by both Song and Janisch in their respective papers as an “initial value explosion” and was the justification behind Q-clipping. These target hyperparameters were investigated with the intention to improve the target network to be more in line with the theoretical cut-off value; reducing the target smooth factor from 0.001 to 0.0005 and reducing the update frequency to only copy the primary network every 5 episodes were implemented under the assumption this would decrease the initial exponential growth. The changes showed no noticeable improvement in either the learning convergence or the predicted Q0 values which continued to exponentially grow. These test experiments were only performed on training sessions which lasted 5000 episodes which is likely not enough time for the Q0 value to self-correct. This implies the training times were too short to experience any meaningful differences

from the target network hyperparameter changes. By observing the training progress presented in Figure 5.6 it can be seen that self-correction does not begin until approximately episode 23000 and settle until approximately episode 28000. These tests take several days to complete training and as a consequence the comparison experiments were not performed. Instead the implementation of Q-clipping was favoured.

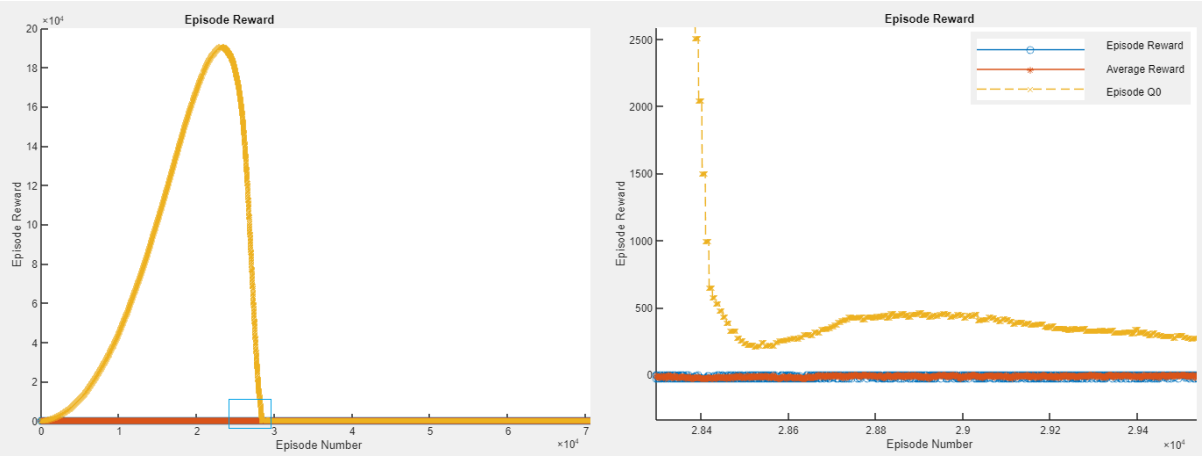

Figure 5.6. Example of Q0 initial value explosion during training (left) and the point of convergence (right) when zoomed in from the blue window shown. Training plot utilised a smoothing factor of 0.001 and an update frequency of 1

Once Q-clipping was implemented it was believed that the target network hyperparameters would be much less impactful on the training results due to the fact that any predicted values greater than 0 would be replaced with the 0 value and all previous experiments showed exclusively positive predictions. Therefore it is likely that the Q0 value will remain a constant 0 throughout the entire training course and updating said network will have effectively no change.

C. Exploration parameters

As a form of Q-learning DQN is an off-policy learning algorithm and will rely on random non-optimal actions being taken to explore the state space. An epsilon-greedy exploration model with epsilon decay was used to encourage state space exploration more in the early stages of training and reliable reward collection in the later stages of training. The ϵ hyperparameter represents the probability the agent will perform a random exploratory action. Beginning at a value of 1, the agent is guaranteed to explore, but as the episode count E increases the value will decay according to the equation (5.7):

$$\epsilon = \max(0.1, \left(1 - \epsilon_{decay}\right)^{E}) \tag{5.7}$$

A minimum exploration value of 0.1 is set to retain exploration 10% of the time even at the end of the training session, as a perfectly greedy agent will learn no new information and training becomes unnecessary. From here it can be seen that the decay parameter must be chosen to allow for exploration for a desired section of training. As it was agreed that exploration should only reach the minimum exploration value after approximately 50% of training, ϵ_{decay} was set appropriately depending on the maximum number of training episodes.

For the long training sessions of 100 000 episodes and minimum exploration value of 1% rather than 10%, the decay factor was set to $6x10^{-5}$ which resulted in exploration occurring only 1% of the time after episode 76 750. Another useful statistic to examine is that the exploration becomes less likely than greedy behaviour ($\epsilon < 0.5$) after only 11 552 episodes; only 12% of the training is predominantly exploration. For the shorter training sessions of 5000 episodes the decay factor was increased to follow a similar proportional exploration rate. A decay factor of $1x10^{-3}$ will result in 50% exploration after episode 692 and 1% exploration after episode 4602. Although the lower threshold comes later in the training (around 90% through) due to the exponential decay there is very little difference between 75% and 90%. An exploration factor ϵ of 0.0235 will remain at episode 3750 which is accurate enough for the purposes of comparison, especially since the 75% threshold was arbitrarily chosen. There is no guarantee either of these decay rates are well suited for the proposed classification problem and whether the agent explores enough must be judged based off the training results. For the shorter training sessions alternate decay rates were tested in $5x10^{-3}$ and $5x10^{-4}$, where the former resulted in very fast convergence to a suboptimal solution and the latter got worse initially before settling on a poor performance. This shows a faster decay rate will negatively impact the training while a slower decay rate may still be beneficial and simply require longer training.

D. Experience buffer

For every action taken by the agent, a tuple is produced containing the state the system was in, the action taken, the reward collected, and the state the system ended in: (s_t, a_t, r_t, s_{t+1}). This tuple is considered the 'experience' the system collects and uses to calculate the loss between the default network and the target network. The experience buffer stores every generated experience and selects entries in batches (known as minibatches) to help average out the learned behaviours to avoid large swings in reward that may cause training to diverge and become unstable. A larger minibatch size will help smooth out the learning and assist in convergence, however will increase the computational costs and will likely increase training time by a significant amount. The size of the experience buffer can also have a large impact on the training success, as only a finite number of experiences can be saved. Depending on the environment and how frequently experiences are accrued the experience buffer must change size to retain a reasonable number of full episode experiences. In the case of this feature observation environment each episode contains a maximum of 101 time steps, meaning each episode will produce at most 101 experiences. For a training session of 100 000 episodes this will produce over 10 million experiences, and an experience buffer with only 10 000 entries would not be able to remember behaviours it experiences early in training. This is often referred to as "catastrophic forgetting" and can be seen within training progress by a sharp decline in performance due to overfitting of new data. In contrast, an experience buffer too large would require massive amounts of memory which may be infeasible to implement as well as reducing the chances of learning from later sourced experiences if the minibatch size is too small. As such, the experience buffer size and the minibatch size are two important hyperparameters that must be considered during experimentation.

The experience buffer was traditionally set to include approximately the most recent 10% of experiences generated for long training sessions (1 000 000 experiences) and 20% for short training sessions (100 000 experiences). These values were again chosen arbitrarily as no reliable sources could be found on buffer size design. As no overfitting seemed to occur during any experiments it is likely that this experience-buffer ratio is large enough to avoid catastrophic forgetting, while also not being too large for the training hardware. The minibatch size varied across many tests, varying from 32 up to 256. No discernible differences were noted other than an increase in training time, and as such the final minibatch size of 128 was settled upon for both long and short training sessions. This value was selected as each episode is able to produce 101 experiences and averaging across an entire episode should in theory guarantee the

experiences being batched together do not have heavy correlation to one another which will improve training stability; 128 was chosen as the minibatch size was traditionally a power of two.

5.3. Results

All experiments were performed on the UTS high performance computing (HPC) cluster using either the Mars or Mercury nodes. The Mars node consists of 26 CPU cores and a NVIDIA Quadro RTX 6000 Passive GPU; the Mercury node consists of 26 CPU cores and a NVIDIA Quadro RTX 5000 GPU. Training utilizes the GPU and both nodes contain enough memory to become a nonfactor. Parallel computing settings are enabled through MATLAB to utilize 10 workers (10 cores) to increase training speed.

5.3.1. Long Short Term Memory Alternative Approach

Using a convolutional neural network is only one approach to classification and for time-series data a recurrent neural network (RNN) may be the better choice. A long short-term memory (LSTM) network is a form of RNN that allows sequence data to act as input to the system. From this fact, the EMG signal data can be imported directly into the RNN and bypasses the need to convert the signal into a time-frequency spectrogram image. The LSTM approach was used to classify the exact EMG data being utilised in the rest of the chapter before it was split into individual motions (Mathworks 2022a). As assurance that the data can still reliably classify gestures after the motions were individually split, an LSTM network was constructed following the MathWorks example to input the newly split time-series data and classify the appropriate gestures. This network utilised all the motion files that were used to generate the spectrogram images (see Table 5.2), meaning any gestures that lasted longer than 10000 ticks or any supination or pronation files were excluded. The architecture and hyperparameters of the network were initially preserved from the provided example but training produced poor performance, predicting the “wrist flexion” gesture for every signal. This behaviour would be expected if the input data had no correlation to classification and wrist flexion was the most common gesture, however it is merely the second most common gesture and other experiments have shown an ability to correctly classify gestures from the EMG signals. As there are approximately equal gesture occurrences between the first and second most common gestures (430 vs 411) it is possible that a random bias in the early stages of training caused the network to begin favouring wrist flexion over all other predictions. This prediction behaviour would be

representative of the agent being stuck in a local minima solution and not properly exploring alternative predictive methods.

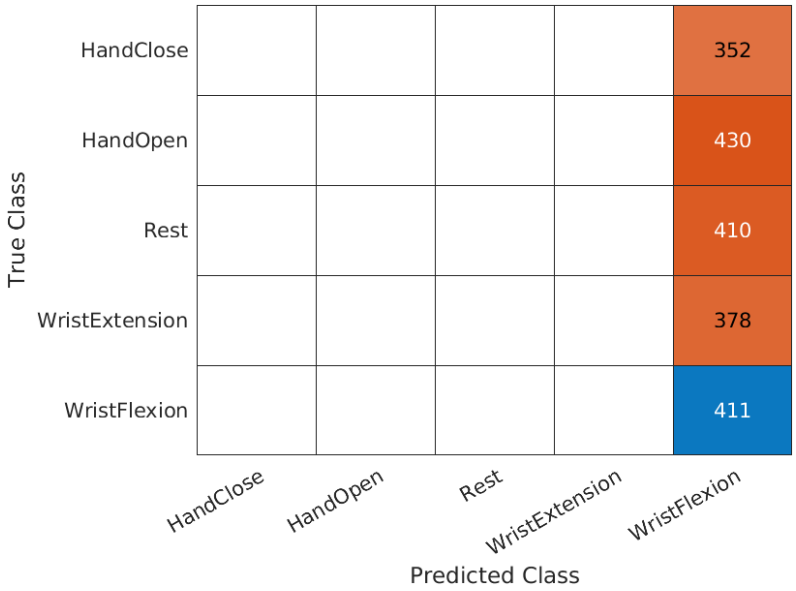

Figure 5.7. Confusion matrix of long short-term memory neural network used for predicting hand gestures from EMG signals. Results show wrist flexion being predicted as the gesture for every EMG signal passed to the system

Attempts were then made to change the hyperparameters and architecture to try and adjust to the differences in input data. The original system has 720 potential input files with lengths in the hundreds of thousands, while this new system has over 9000 files with lengths of only 10000. Systematic experiments were performed on the number of hidden units in the LSTM layer (related to signal length), minibatch size (number of training files), and learning rate (accuracy stability).

The number of hidden units in the LSTM layer was originally set to 80 for the signals that contained 28 gestures per signal. Current understanding of the LSTM structure is that the number of hidden units determine how much can be 'remembered' about the signal and its features. As the signal length was less than the original it was assumed that a lower value would be beneficial to avoid potential overfitting, and that increasing this value would not benefit the system in any way. Tests were performed using values of 80, 40, and 10, with no meaningful changes to the training session or the resulting accuracy. Each result showed predominant favouring for the "wrist flexion" gesture in approximately 85% of cases with approximately 14% predicting "rest" and 1% scattered within the remaining gestures. Decreasing the number of hidden units did result in more predictions for rest overall (shifting percentages to around 75-25% split), but the strong favourability of wrist flexion remained. Signal lengths of 10000 would

likely have enough information embedded within them to justify sharing the original structure of 80 hidden units. With the intuition that decreasing the amount of signal retained in memory would not improve classification, combined with the understanding that this variable was the least understood among the remaining variables in question, this value was kept at 80 for the remainder of the tests.

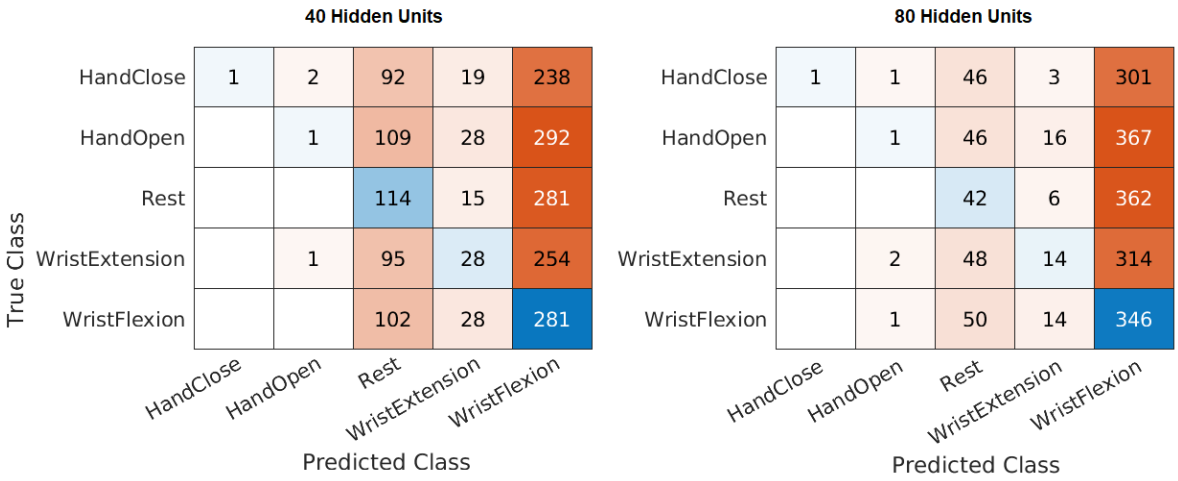

Figure 5.8. Comparison of testing accuracy for a variation in LSTM hidden units (40 on left versus 80 on right)

The minibatch size of the original example was only 32 which must be increased to better correspond to the number of training files being used in the tests. As the training set was only 576 files with the remaining being used for validation, only 18 iterations were required to exhaust the dataset and complete one epoch. To aim for a similar ratio for the new training dataset of size 7923 the minibatch size must be increased to 440. As values equal to powers of two are preferred the minibatch size was tested with values of 128 and 256. A minibatch size of 512 was attempted but exceeded the maximum memory available to the hardware and prohibited training. This led to an attempted 400 minibatch which did run properly but still never learned any substantial classification techniques. As no variation caused improvement the minibatch size was set to 128 for the computational speed improvements that come from smaller minibatches.

The learning rate of the system is often described as the most important hyperparameter as it dictates how much the network will incorporate from any individual result. Although this test used a piecewise learning rate which drops by a factor of 10 near the end of training, the initial learning rate is discussed in this paragraph with no variation in the dropped value. With the default value of 0.001 the most likely reason for poor performance is a small learning rate, being unable to learn from new experiences. As there is no noticeable change in training accuracy at

epoch 40/60 when the learning rate drops it is extremely unlikely that a smaller learning rate would improve performance. Tests were performed for both values 0.01 and 0.1 which did affect the spread of gesture predictions, but instead both caused a complete 100% “wrist flexion” prediction rate, producing the opposite behaviours as desired. As a follow-up the learning rate was reduced, although results were expectedly poor. Learning rate was ruled out as the cause for improper prediction behaviour.

Alternative tests were performed on the training dataset to confirm the current network was capable of learning. The training set was reduced to contain exclusively the hand close gesture to force the network to learn to only predict the appropriate gesture. This did result in the network predicting hand close for 100% accuracy, which means the network is capable of learning and may simply be struggling to distinguish between gestures based exclusively off the single EMG files. This does not answer how the original Mathworks example was able to effectively learn the gestures, nor does it answer how to fix the current network state. As the entire LSTM experiment was exclusively performed to improve faith in the training data it was deemed no additional time could be justified in improving the LSTM network and the subproject was not pursued. Although the tests did not validate the motion-separated EMG signals as intended, they do provide useful negative results stating that the EMG data is not guaranteed to be classifiable in its current state and any future experiments that utilize the same data may run into similar problems. Although disappointing, it is an important consideration to have for the later tests involving reinforcement learning in 5.3.3.

5.3.2. CNN Classification

With the intention to use the CNN to produce the features, the CNN classification rate must be as accurate as possible to maximise the correlation between the produced features and the correct classification. For similar reasons, it is important to review the accuracy of the CNN without the feature layer as this will allow the examination of the feature extraction process effects on accuracy. Results using the 8-channel data were produced at the earliest stages of development which showed very high accuracy results, but was discarded for the preferred data structure of 4-channel data. All training discussion hereon will be referring to 4-channel data for classifying the five hand gestures of “Hand Open, Hand Close, Rest, Wrist Flexion, & Wrist Extension” along with a sixth “Unknown” category. Both CWT and FFT spectrograms were used separately to train their own networks and compare potential accuracies.

For no feature vector layer within the CNN architecture the CWT training set was able to achieve a validation accuracy of 96.08% within 25 epochs while the FFT training set was only able to achieve an 88.11% accuracy in the same time period. This 8% difference in accuracy is not negligible and must be considered before further implementation is continued. This difference is exacerbated when the feature vector layer is added to the architecture: CWT accuracy drops down to 94.10% while the FFT accuracy drops to 75.28%. The FFT accuracy with feature extraction miscategorises 1 in 4 gestures and should be considered bordering on invalid; an accuracy lower than 75% would be too low to reliably use in any real-world applications. The test of changing the relative position of the dropout layer to after the feature vector showed an even further drop in accuracy for the FFT network, down to 68.3%. This result simply confirmed the position of the feature vector within the layer structure and as such the feature vector is retained after the dropout layer. This test was not repeated for the CWT network as it was not expected to provide any important revelations.

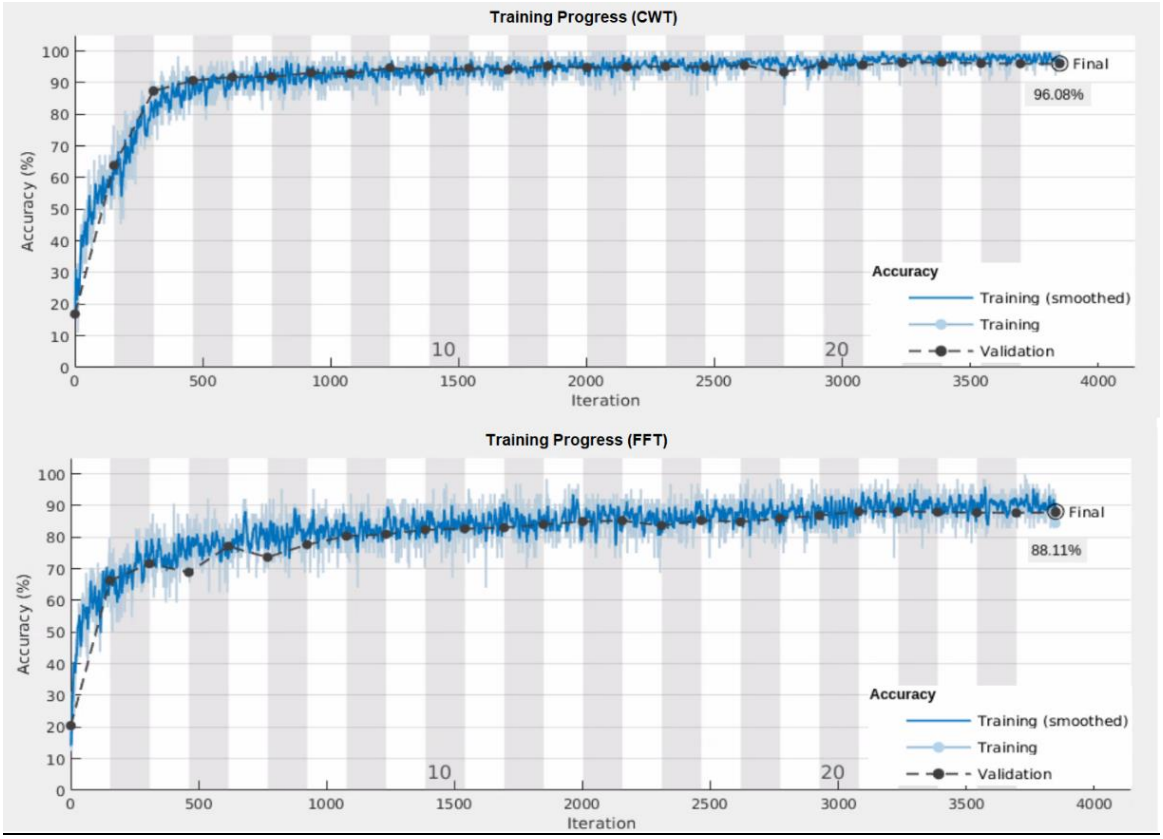

Figure 5.9. Training progress of CNN for both CWT (top) and FFT (bottom) spectrogram images with no feature vector layer

The training progress in Figure 5.9 shows both datatypes learn quite rapidly with both training and validation accuracy reaching over 90% for CWT and 80% for FFT in only 10 epochs. The only significant comparison that can be made is that FFT has higher variation in the training results when compared to the tight smoothed training graph produced by CWT. To extract any meaningful interpretation between the input data it is better to observe the predictive behaviour on the testing data. The total 9904 data samples are subdivided into three categories: training, validation, and testing data. Training data makes up 70% of the total samples (9900 images) and is the data used to produce the plots above. Validation data is 15% of the total samples (2122 images) and is used during the training session at the end of every epoch to validate the training accuracy on a separate dataset. Once training is completed the final testing dataset is used to confirm accuracy on previously unseen data. Testing data makes up 15% of the total samples (2122 images) and can have the predictions of this data presented visually as a confusion matrix to provide insight into each category and its predictive patterns. Results for the CNN structure without the feature vector layer is provided in Figure 5.10 while the CNN structure with the feature vector layer is provided in Figure 5.11.

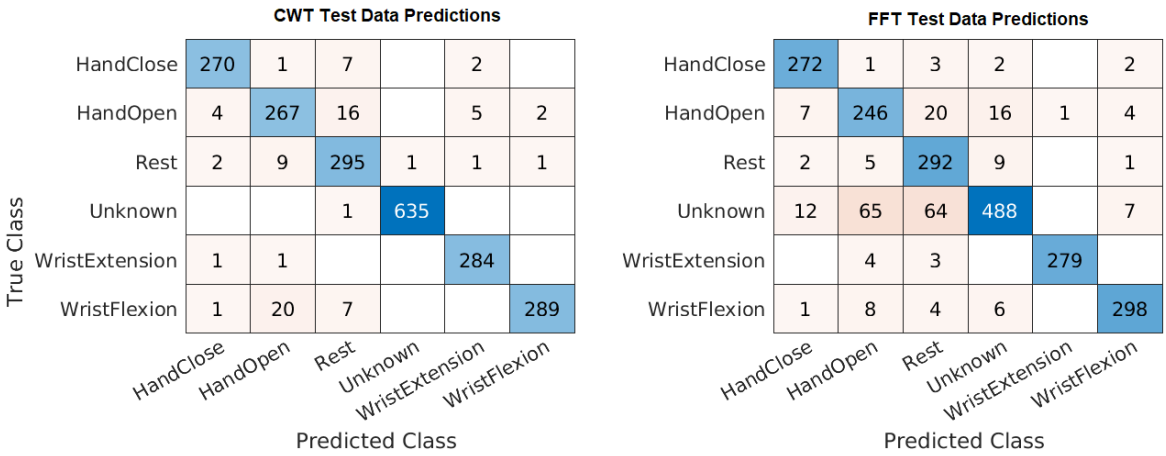

Figure 5.10. Comparison of test data classification accuracy between CWT spectrograms (left) and FFT spectrograms (right) when no feature vector layer is included within the CNN structure

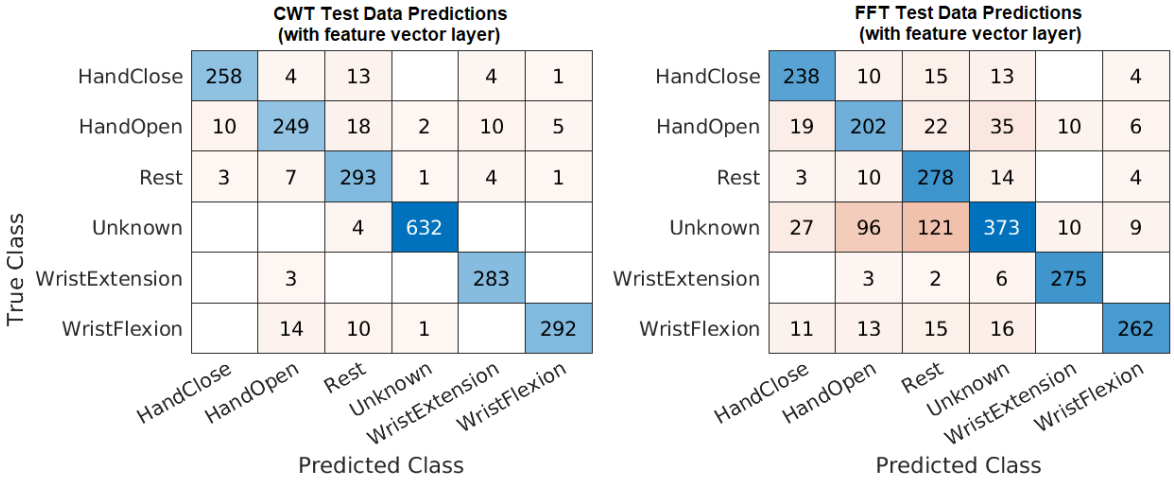

Figure 5.11. Comparison of test data classification accuracy between CWT spectrograms (left) and FFT spectrograms (right) when an additional fully connected layer of 100 neurons is included within the CNN structure for feature vector extraction

The confusion matrices show that CWT and FFT predictions are relatively equal in all categories excluding the correct labelling of unknown data. For the non-feature structure, where CWT only miscategorised 1 unknown data sample, FFT miscategorised 148 of the 636 samples. 129 out of these 148 were predicted to be either the hand open gesture or simply a resting motion; adding the feature vector layer intensifies these issues. This outcome implies that the FFT images that are used for the unknown category are similar enough to these two gestures that it is causing a large number of false positives. As a reminder, the unknown category contains a combination of the supination and pronation gestures which were originally discarded due to the EMG sensor placement not being configured to record the appropriate muscles. This would naturally result in these gestures resembling resting, as the appropriate muscles were not recorded. Hand opening may also be similar to resting as the final position of the hand would likely be the same in both cases. These confusion matrices highlight a potential failure in the training data which explains the drastic drop in accuracy as a by-product of a poorly constructed “unknown” category. By discounting the unknown category both CWT and FFT will have similar accuracy within the testing data. This opens up future experiments to continue using FFT spectrograms rather than considering them as an inferior data set, as long as the unknown data is explicitly considered.

5.3.3. Reinforcement Learning

5.3.3.1. Early stage testing results

Although the final RL agent and environment are detailed in the earlier sections of this chapter there were several previous iterations that varied in some notable form that all had results generated and analysed. Not all results are presented here, however some results have been included to present a larger picture of the design process.

Originally the observations for the agent did not include the index vector m and only the observation vector x was input. The neural network that acted as the Q-value function approximator had a simpler architecture with only one hidden layer and no Q-value clipping, which allowed initial Q0 value explosions. Throughout all the hyperparameter tuning no system was ever able to converge to a reasonable behaviour, although each training session was rarely longer than 5000 episodes which is likely not enough time for proper state space exploration for a system with so many observations. Some experiments lasted 100 000 episodes and similarly poor results were produced and as such it is believed that runtime is not the only factor in the lack of convergence to a reasonable solution. Several iterations did converge to a clearly non-optimal solution; the most common behaviour that the system would converge to is the constant repeating of the same action corresponding to observing one specific feature over and over. This led to the necessity of punishing repeat observation behaviour through the reward function.

When the cost of reobserving a feature was set to twice the cost of a regular observation the system still learned to continue repeating the same observation, despite the clearly superior approach of observing different features every step existing and being possible. This result shows that the agent never found this solution through random exploration, so to discourage repeat observations even further the cost of reobserving was increased dramatically from -0.002 to values ranging from -0.5 to -5 over several tests. The goal of these tests was to cause the Q-values to decrease more severely and produce a larger loss function during training that would cause faster learning and discourage the illegal behaviour more severely. All tests along these lines returned similar training patterns with the final agent favouring the constant repeating observation patterns. Some agents observed several features before repeating observations, but all did eventually repeat their actions for the remainder of the episode. The agent with a smaller relative cost of -0.2 was shown to observe more unique features before collapsing into repeat observations when compared to the agent with the higher cost of -5. Both post-training agent action behaviours are displayed below:

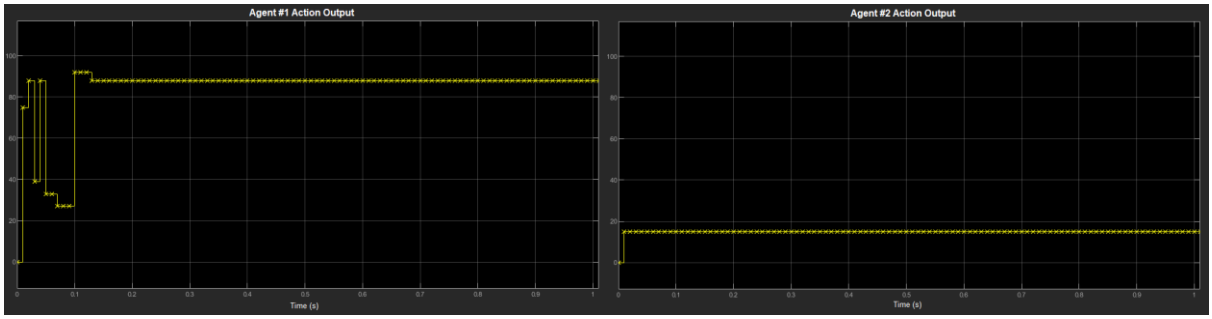

Figure 5.12. Comparison of two failed agent action outputs with different reobservation costs (-0.002 vs -5, respectively) displaying a repeating observation behaviour. Agent #1 observed several different observations until eventually observing one feature exclusively. Agent #2 only ever observed one feature.

A side-experiment was performed by providing a positive reward for unique observations, which should in theory encourage the agent take new actions every time step. This addition to the reward function does break the guaranteed negativity of the Q-value and as such the Q-clipping is no longer mathematically valid, but may still help in training if the positive reward path is not discovered during training. This positive-observation system was trained both with and without Q-clipping and neither result learned to observe all the features available. To test whether the lack of convergence was due to the excessive number of possible observation combinations the number of features was drastically reduced from 100 to 24, including only the features with positive value, and then only 10 of those features to reduce even further. Even with the reduced state space the agent favoured repeated actions for both clipped and unclipped Q-values, although the training progress did show a steady increasing reward per episode which is a trend not seen in the larger state space tests. This suggests that the system is capable of learning but a 5000 mini-training session is not enough time for full convergence. The accumulation of reward in this scenario comes from simple observations and does not provide any insight of the actual value of any of the features, not does it display an ability to correctly predict a gesture from this information. From this experiment it is suggested that the primary limiting factor for the agent learning to observe every feature is simply due to the number of observations and the number of potential states that can be transitions to at any time step. Longer training sessions are one approach to compensate for the large observation space but training times for 100000 episodes were recorded to take over 1 week to complete which is unreasonable for the purposes of this chapter.

Any attempts to shorten training time, such as implementing an early termination from any illegal action such as reobservation, simply force the agent to collapse to the solution of immediate termination which prevents the agent from exploring and learning any meaningful

portion of the observation space. Only a single experiment redirecting illegal actions to legal observations was performed as the results showed similar behaviours of repeated actions. Since they were being redirected to observe an appropriate feature it was clear that this design would not learn this behaviour to be undesirable within a reasonable timeframe. Continued experiments in the future may find value in continuing this approach.

5.3.3.2. Final attempt results

The final structure relied upon an increased cost for features that had been observed previously, along with no early termination condition for illegal actions. Using the CWT feature vectors the training was performed to determine which feature combination would cost the least while also increasing the probability of correct classification. Every illegal action taken was penalised more severely than an incorrect classification to discourage these actions as much as possible. From the baseline reward function presented in (5.4) the cost of a repeated observation was set to -10 while the cost of a unique observation was set to -0.01.

$$r_t = \begin{cases} 0 & a = y \mid a \in A_c \\ -1 & a \neq y \mid a \in A_c \\ -0.01 & a \notin \bar{F} \mid a \in A_f \\ -10 & a \in \bar{F} \mid a \in A_f \end{cases} \tag{5.8}$$

The cost of new observations was increased to -0.01 for this attempt, however this was a mistake and from the inequality (5.6) it can be calculated that for this cost to be viable the probability of correct classification would have to increase to 1.2, violating probability laws. It is believed that this issue can be ignored for the time being for several reasons. The maximum observation cost assumes that the best way to improve prediction chances is to observe every feature, yet this is antithetical to the actual goal of the project of observing as few features as possible. If the agent were to discover a subset of features that increased classification chances then no probability laws would be violated and the reward function would still converge to this solution over the random guessing solution. A secondary reason can be seen in the agent action output in Figure 5.14, which shows unique observations being selected for the first 8 time steps of the simulation, then repeating the same action until termination. These results will be discussed in more detail later, however this behaviour suggests the agent was indeed learning to observe new features but was simply unable to complete the learning pattern within the training parameters. For this reason, the additional cost of the new features is unlikely to have caused any noticeable effect on behaviour.

Due to the required training time (184 hours) and resources required to repeat the test, it has been elected to accept the potential mistake in the reward function and fix the issue in any future works.

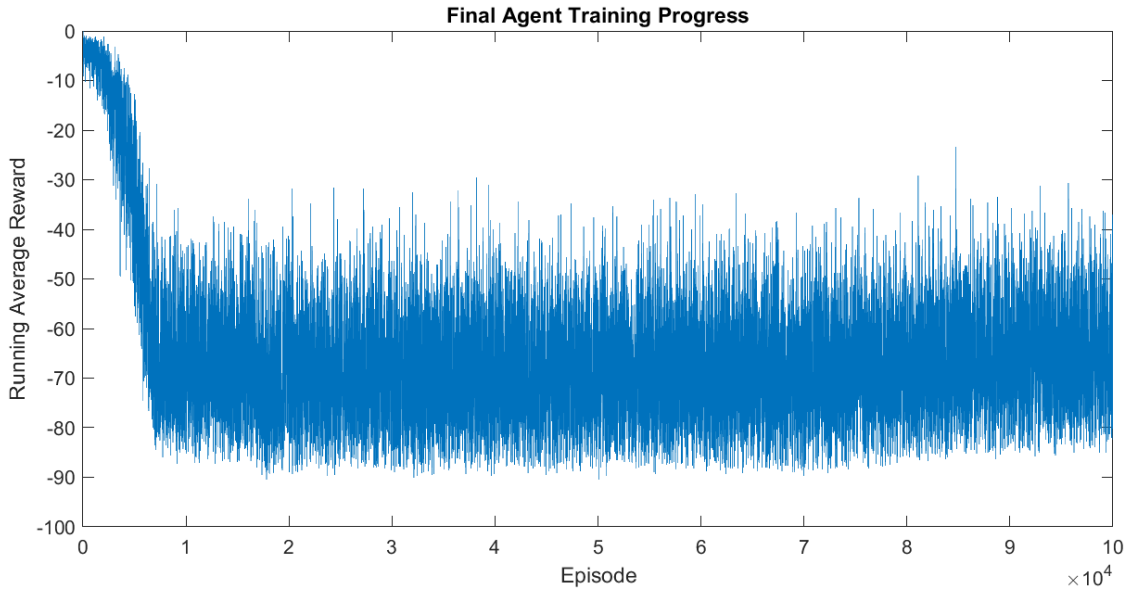

Figure 5.13. Training progress of final agent attempt of feature selection via reinforcement learning. Running average calculated with a window of 10 episodes

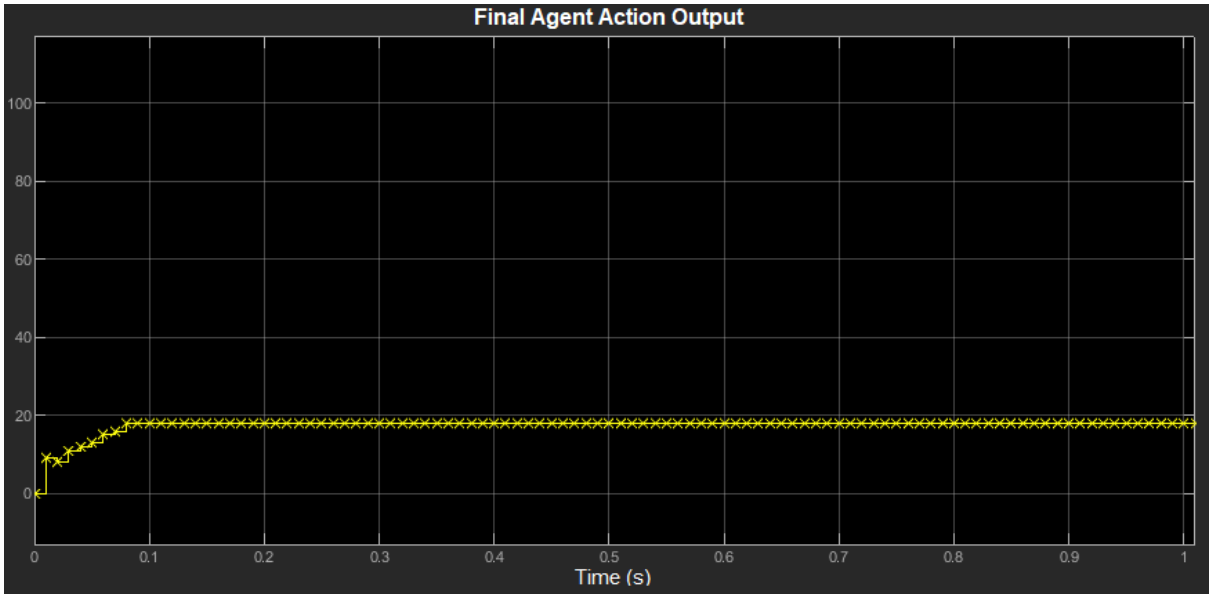

Figure 5.14. Action output of final agent attempt of feature selection via reinforcement learning

5.3.4. Real-time EMG classification

The final stage of the EMG classification project was planned to allow a user to wear four EMG sensors to collect the appropriate channels, pass these time-series signals through a buffer to construct a uniform-size spectrogram, and utilize a CNN to classify the spectrogram image to the appropriate gesture. The reinforcement learning component was intended to help optimize the CNN architecture or act as an alternative classifier to compare accuracies. The real-time classification code created was based on the real-time audio classification code provided in the Mathworks speech recognition example (Mathworks 2022d), but with the method of spectrogram conversion being replaced with the appropriate transform.

Another difference is the number of channels of data that are required for the classification network. The audio network only required a single channel that was imported from the microphone and as such only produced one spectrogram. The EMG classification data requires four channels and must produce four spectrograms all displayed on the same graph. The combination of spectrogram data is not difficult but the production of multiple spectrograms will increase computation requirements and will potentially slow down real-time classification. Tests were performed on both CWT and FFT networks to determine how fast the spectrogram results could be generated and displayed graphically with only a single channel of audio data. Using the continuous wavelet transform showed a very slow update time, with each loop requiring on average 0.269 seconds for a single channel. The fast Fourier transform showed a much faster average update time of 0.054 seconds, much better suited for real-time classification. These times would naturally be increased for additional channels, as calculating and plotting the same data twice increased the CWT time to 0.84 seconds and the FFT time to 0.1016 seconds. A two-channel CWT calculation requiring almost 1 second suggests that this transform method, or at the very least this implementation, would not be feasible for a four-channel input real-time classifier. The FFT method seems to scale linearly and has the potential for real-time applications, but may cause some problems if the calculation times hinder the EMG data collection. All tests were performed on an HP Elitebook 840 G5 with an Intel core i7-8650U CPU and 16GB of memory.

The collection of EMG signals is performed using DELSYS Trigno wireless EMG sensors which can be imported directly into MATLAB with a sample frequency of 2 kHz. Performance has been tested and confirmed working, however as the sensors are restricted to the UTS laboratory most tests were performed using substitute data for proof-of-concept development. Due to the results generated in all previous stages of this chapter project being problematic, along with lab

access being restricted due to COVID, the application of EMG data to the real-time classifier was never completed as it was not considered a high priority. Each component of EMG collection into MATLAB, time-series conversion to spectrogram images, and real-time image classification works individually, and any future work should be able to combine these techniques to create a real-time EMG classifier that functions as effectively as the original CNN classifier.

5.4. Discussion

5.4.1. Comparison of CWT and FFT for rehabilitation purposes

When deciding upon which signal transform method to implement for any practical purposes involving robotics there are two dominant factors that must be considered: accuracy and response time. The results for the CNN classification showed continuous wavelet transform spectrograms generated a higher accuracy than the fast Fourier transform spectrograms when the system was required to also categorise unknown gestures that were similar to existing categories. This suggests that CWT is able to produce a more nuanced data representation that is able to be detected by the CNN. The direct trade-off of this comes in the form of the increased calculation time. This response time increase discussed in 5.3.4 would be extremely detrimental in a robotic device, especially one coupled to a human that relies on the device to move as they expect in a predictable manner. Any delays between user intention and robot activation run the risk of inhibiting motion or harming the user, or violating the guaranteed passivity of the coupled system (Lee and Hogan 2016b). These factors lead to an unfavourable view of CWT applications, yet other papers have discussed the use of CWT for image processing and specific feature detection and suggest fast CWT algorithms are more widely available than what many believe (Antoine 1998). Open source algorithms for fast CWT are available with accuracies that match CWT with drastically improved speeds (Arts and van den Broek 2022). This approach has potential to improve real-time classification and would make a valuable expansion to the project as a whole. This is not employed at this stage due to the lack of robotic device to appropriately implement within, and standard CWT and FFT functioning acceptable for the simulation stages of the project.

On the opposing end, FFT results showed fast response time but lower accuracy. For the two networks that did not require feature extraction the difference in accuracy was 8% which in practice would lead to the robotic system miscategorising an action and providing incorrect assistance for almost 1 in 10 more gestures. As a rehabilitation device would be expected to

handle hundreds of inputs throughout daily living this 8% would be extremely noticeable and have potential negative impacts on the user. These problems would be even worse for the feature extraction CNN, and as such some improvement to the FFT networks must be performed before they become feasible for robotic implementation. With the realisation that the primary cause of inaccuracy comes from the unknown category (as seen in Figure 5.10) it is possible to remove these results to determine the accuracy of a 5-category FFT classifier. Removing the true class unknown files does remove the samples that cause majority of the problems, however there is no way of knowing which prediction class the other samples would be distributed towards. These samples are simply considered incorrect predictions, which does bias the following results in favour of CWT by assuming every gesture would not be incorrectly placed, however the importance of this information is minimal and does provide a close enough approximation for insights of FFT accuracy.

Table 5.3 – Test Data Accuracy Dependence on "Unknown" Category

Transform	With Unknown	Without Unknown
CWT	0.9614	0.9461
FFT	0.8836	0.9334

By removing the unknown files it is revealed that FFT accuracy is on par with CWT. Combined with the improved response time FFT seems to be the favoured method. With each spectrogram being produced in approximately 0.05 seconds, using FFT for control purposes is feasible. CWT as it was implemented caused too much delay in signal sampling and classification to be useful for any real-time applications, however further research into fast CWT algorithms may be able to match FFT speed as well as improve accuracy. Without this additional research FFT is the recommended transformation algorithm to use for rehabilitation robotic purposes.

5.4.2.Potential explanation of non-convergence

At the end of all the experiments and training sessions no agent had been able to achieve the desired behaviour of observing the bare minimum number of features to correctly classify the gesture that generated said features. The ideal training progress was expected to begin with observing a large number of features and slowly reduce the number of observations taken to reduce the negative cost while not negatively impacting the probability of correct classification. This pattern was never recorded as the current setup allowed repeated observations which

caused the number of possible transitions between states to never decrease and led to a very difficult environment to explore. With 100 observations the number of potential non-terminal states to exist in at any given time is 2^{100}, or $1.27x10^{30}$, which is a lot of space to explore. To determine whether an action should be repeated requires the agent to observe the entire state vector and not just an individual value, as these other components may change the scenario.

For a simple analogy, a system observing two-dimensional coordinates [x y] may learn that when in the state [5 0] it is beneficial take the action “move forward”. However the agent must take into consideration both the x and y values and cannot determine behaviour based exclusively off the x value; existing in the state [5 5] and performing the action “move forward” may cause a collision and result in negative reward. For the observation equivalent of this limitation, if after agent has observed the first feature and exists in the state [1 0 0] if it attempts to observe feature 1 again it will be negatively rewarded. However if the agent has observed the first and second observations and is in state [1 1 0] then the previous learned experience is no longer applicable, the appropriate neural network weights to discourage repeat actions would not have been updated, and the agent must explore every possible action from every possible state to achieve a complete understanding of the environment. Learning the relatively simple rule of “do not repeat observations” is surprisingly difficult through reinforcement learning. The inability to prevent repeat observations leads to the problem that every possible state has a potential 100 states to transition towards which means that for any given episode there is a total of 100^{100} possible pathways through state space (as opposed to 100 factorial if the action space prevented repeated actions). These two numbers are both astronomical ($1x10^{200}$ versus $9.33x10^{157}$) and although the difference between these two numbers is extremely large the smaller value would still be infeasible to appropriately explore in any reasonable amount of time. It would also be unwise to stop exploration partially through a poor path that accumulates large amounts of negative reward as there is no way of knowing if a large reward would be experienced at the end to promote specific behaviours.

The original paper by Janisch, Pevný, and Lisý (2019) performs tests on several databases with feature vectors up to 784 features in length (although most are below 54, three tests are 400+) and achieved better results, so it is confirmed possible to perform feature selection via RL. If the action space reduction is not believed to be the main cause of failure the other differences mentioned are what are labelled as “extensions” to the training method. It is described that before training occurs a smaller training session takes place focusing on states that have very few observations taken (states distributed towards the initial state). These generated states are

used to pre-train the Q-values that correspond to classification actions, which due to their terminal nature do not depend on any following states and will not need further updating. This technique seems to depend on the main algorithm having a high classification accuracy to begin with. It is not clear whether this is true for this project, although it should logically follow that an accuracy approximately equal to the CNN with feature extraction ought to be possible. Another procedure employed that may improve training efficiency is the utilisation of a high-performance classifier (HPC) that is able to act as a supervisory body that can predict with high accuracy (but not necessarily 100%) the appropriate label for the current data sample for a relatively high cost. This can be implemented into the reinforcement learning agent fairly easily by adding an additional action that passes the entire data sample to an external classifier rather than iteratively taking individual features from the sample and passing them to the RL classifier. By setting the cost of this action to equal the cost of observing all remaining features the system can benefit from the external classifier setting a default classification rate and biasing the learning parameters to favour the correct labels earlier in the training process. It is explicitly mentioned that HPC improves performance for samples that require many features for classification, which necessitates the sample will have large numbers of features overall. The construction of a non-neural network classifier such as a support vector machine can be completed without much difficulty, but was not done so due to time constraints. The final extensions applied to the base DQN network include the double DQN structure, duelling architecture, and the Retrace algorithm. Double DQN simply utilises a target network that updates at a different rate than the primary network and reduces overestimation of action values (van Hasselt, Guez, and Silver 2015). Double DQN is already employed in the project and therefore no adjustments can be made in this particular case. Duelling architecture merely separates the Q-functions into a value function and an advantage function to help stabilise the resulting values over various states which accelerates and stabilises training. Retrace allows network parameters to update using episode-long experience batches rather than random scattered experiences. Neither duelling architecture nor Retrace are truly understood and due to time constraints research was not expanded into these topics. Instead the implementation of these domain-independent improvements are left for future works.

The experiments that reduced the observation space also failed to achieve predicted action outputs, although once the observation space was cut to only 10 maximum observations it was possible to see unique feature observations being favoured over repeated actions. Only observing the training progress displayed in Figure 5.15 would not suggest a well-trained agent

as the running average reward has decreased significantly throughout training. This drop is caused by an additional component in the reward function that punishes the agent if the simulation reaches the maximum time steps without performing a classification action and instead performs an observation action at every step. This component adds a very large negative reward of -100 to the accumulated reward which pulls the average reward of each episode down while the agent explores the non-terminal states. Based on the action output seen in Figure 5.16 further exploration is needed to avoid the repeated observations in the final 5 steps as well as discover the classification gestures before the final step allow the avoidance of the drastic penalty. What this test does show is that even for a small observation space with only 10 measurements, a training session of 10 000 episodes is not enough to discover optimality. These results put into perspective the monumental task of 100 observations truly is, how a standard DQN network is ill-prepared for this task, and explains the necessity for the additional techniques mentioned above to improve training efficiency.

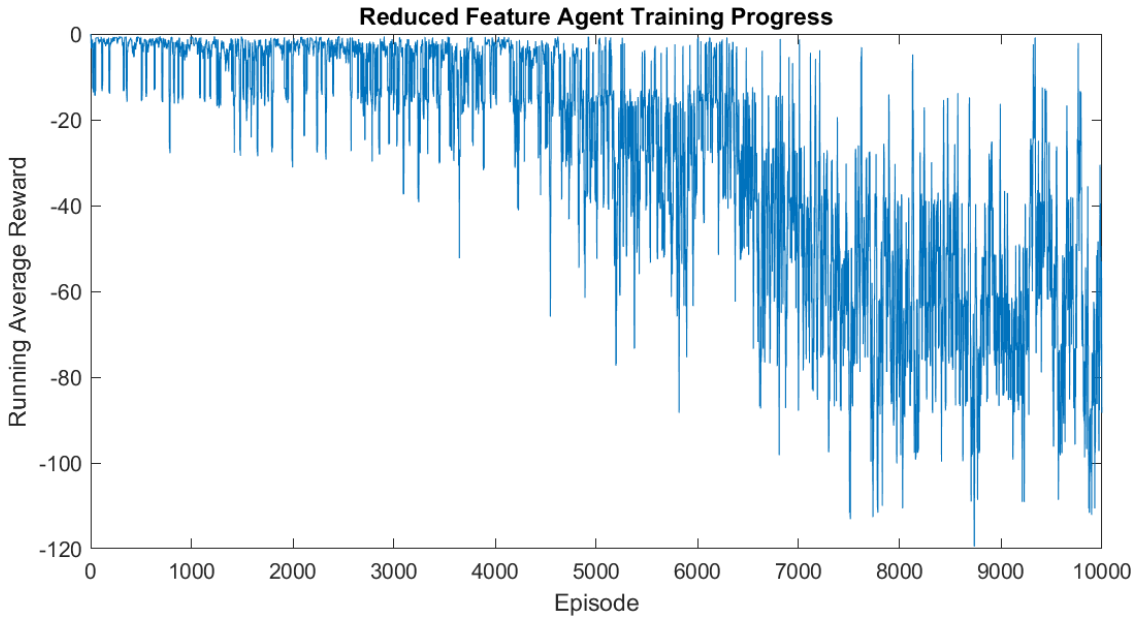

Figure 5.15. Reduced feature agent training progress showing a deterioration of reward despite reasonable action output

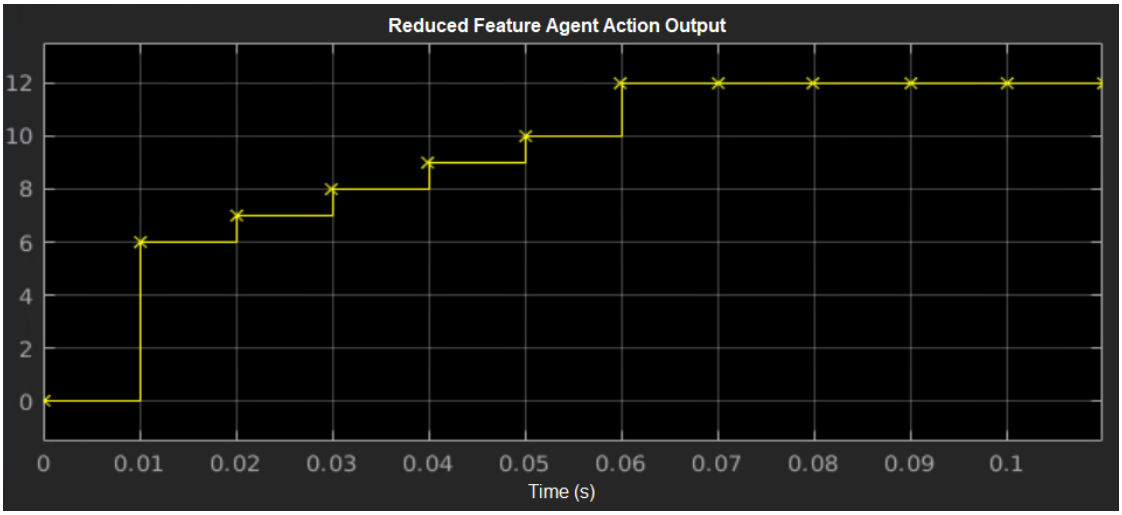

Figure 5.16. Reduced feature agent action output showing a non-repeating action behaviour

From this test it can be seen that no attempts at classification are being made as the agent favours repeated observations and without a reducing action space the classification cannot be guaranteed. As such, the default classification rate is simply equal to the inverse of the number of categories (in this case 20%). No improvement in gesture prediction accuracy seemed to occur from the additional feature observations. The method for achieving the 10-feature vector was simply including the last 10 positive features from the original feature vector as the new feature vector. By ignoring a large proportion of the originally generated features the system is potentially missing out on information that is paramount for accurate predictions. Because of this, along with the agent's hesitancy to actually predict a label, it is impossible to know whether there is enough information encoded within the remaining features to reliably predict the gestures, and the lack of improvement in predictive accuracy cannot be attributed to a failure to learn by the agent and may simply be a product of poor initial feature vectors. This concern, when cascaded with the original lack of confidence in the feature extraction method, severely restricts any validating results that are achieved and makes any conclusions difficult to attribute to any specific cause. The original concern with the feature extraction method is that a fully connected neuron layer at the end of a CNN does not have any inherent meaning or relation to physical measurements. Without these guarantees it is difficult to intuitively claim the produced features should be able to classify a physical gesture. The desired feature extraction method of condensing each filter map (also known as a feature layer) into a scalar value to save would have retained a semblance to physical meaning as each filter map would have derived from the EMG spectrogram image. As this method was not easily implemented the fully connected layer was selected, with the confirmed knowledge that the combination of activation values, when run

through an activation function and appropriate node weighting connections, was guaranteed to be able to result in a classification of at least 75% accuracy. It was this certainty that persuaded the continuance of this approach, however it is possible that due to the nonlinear nature of the relationship between features and classification (which includes a ReLU activation, a dropout layer, and a softmax activation) that it is not possible for an RL agent to extract any meaningful combination of features that result in classification. This is not believed to be the case, as it seems that Song et al. (2018) also followed a similar technique, which quotes “the size of the CNN network output vector is 288x1, then as the feature vector input to the DQN network”. Therefore, although it is not guaranteed to be the correct feature extraction method it is believed that the generated feature vectors are a viable method and as such the failure to learn classification behaviour is unlikely to be caused exclusively by faulty input data.

The final explanation of non-convergence is simply an incorrectly tuned hyperparameter set that results in poor training progress. Due to the Q-clipping for the guaranteed negative Q-values the target network will likely remain at 0 and as such any hyperparameters related to the target network are unlikely to have much effect on overall training. No experiments gave any insight as to whether the learning rate should be adjusted or in which way it should be adjusted. Each reinforcement learning problem is unique and basing a learning rate off existing papers is not a reliable tuning method. In spite of that, all Mathworks DQN examples use a learning rate of 0.001 to generate successful results with a similar hyperparameter combination, and the experiments performed using higher learning rates simply showed a proclivity to fast convergence to a suboptimal solution. Tests with lower learning rates, or learning rates that dropped during training did not show any improvements to feature selection behaviour.

The experience buffer size should be dependent on the environment and the maximum training time, as it must contain a proportion of the total training at any given point during training. With 100 observations each episode can generate a maximum of 101 experiences to store in the buffer. Naturally, the buffer should contain learned experiences from early in the training session to understand the environment, as well as the later stages where behaviours are being discovered and patterns emerge. An experience buffer of 100 000 will be able to hold 1000 episodes of experience, which if training lasts for 100 000 episodes, is only 1% of training. This would likely lead to catastrophic forgetting, where any patterns learned early in training would be forgotten and the agent begins overfitting to a behaviour it has learned but not guaranteed. Guidelines were set to try and keep the experience buffer large enough to contain approximately 10% of all training. This number was chosen arbitrarily and may be adjustable based on the agent

exploration factor (where agents with higher exploration will have more variation in later stages of training and would likely benefit from a longer memory) and hardware memory limitations. The minibatch size will also need to scale appropriately to avoid sample biases. If the minibatch size is too small relative to the amount of generated experience then each training step runs the risk of selecting experiences from only one time frame of the simulation. A simulation with 1000 steps would need a minibatch size large enough that the sample is likely to contain experience from every point in the run. This is especially important if the environment allows early termination (such as this one) which would bias the experiences into favouring early simulation times.

The only direct statement that can be distilled from these experiments is that the maximum number of features has a negative impact on system learnability and a basic implementation of reinforcement learning is not well suited for high dimensionality. More advanced techniques such as the use of a secondary high-performance classifier or Retrace seem to be a necessity in reliably exploring the environment. Other applications of RL have shown the ability to learn much more complex tasks in more expansive environments, such as the OpenAI system trained to play the complex computer game DotA 2 using millions of frames as the observations (OpenAI et al. 2019). This paper mentions one of the challenges being only able to access partially observed states and requiring inferences based on this incomplete data. Should their techniques for solving this problem be implemented into the EMG feature reduction and classifier problem, then large amounts of features would become obsolete and predictions could be made with fewer EMG readings. The paper also mentions high dimensionality in observations spaces as being a challenge to overcome and a requirement for efficient computations arising from this. One explained example of training efficiency was the increase of batch size to increased learning speed, which does require more optimiser GPUs. They discuss the use of a second agent that is trained separately to compare the primary agent performance, however this second agent seems to be designed to account for a changing environment and neural network architecture which is not applicable to this project. Although not the exact same, this implementation is similar to the use of the HPC, and having access to a secondary prediction actor seems to be beneficial to training efficiency.

5.5. Conclusion

For traditional classifiers, the input to the system requires the designer to manually select a number of features to best highlight the differences between samples while reducing the amount of necessary data that is fed to the system. The selection of these features requires expertise and knowledge of the system that may not be readily available for many systems; for a reinforcement learning agent the selection of important observations is one of the most fundamental stages of development. To try and avoid this step, an RL agent was designed to accept a multitude of observations that were generated by a separate classifier with the intention of iteratively observing individual features to learn which features were most important for the purposes of signal classification.

This implementation was performed with the goal of classifying hand gestures based off collected electromyography signals from the forearm. Feature extraction from EMG signals is an important and potentially difficult task due to the variation in humans causing some features to be highly dominant in some cases and non-existent in others. By using a convolutional neural network to generate an excessive number of features automatically, the likelihood of no features being comparable between humans decreases as the number of features increases. A publicly available EMG dataset was manipulated to construct a new dataset of individual EMG signal files that corresponded to one of five hand gestures. Each of these files were then used to develop a feature vector that contained 100 features for every EMG signal through the use of a CNN. The customised CNN was able to classify spectrogram images that were produced from the EMG signals with an accuracy of approximately 93% when no features were extracted with a slight drop when features were extracted from an additional hidden layer in the neural network. The CNN alone proved that the produced feature vector could be converted into a classification probability vector using nonlinear functions, so classification via reinforcement learning should be mathematically possible at the very least. It was also discovered that the time to calculate a spectrogram using the fast Fourier transform was much shorter than the continuous wavelet transform and was therefore more suited for the real-time classification that would be required for a rehabilitation robot.

By following a guiding paper as close as possible, the RL agent and environment was constructed and tested throughout a myriad of permutations. No results showed a true convergence in the training results and many iterations showed similar behaviours of repeatedly observing the same feature throughout one episode regardless of the negative reward accrued from this action. Observing agent output did reveal a slow learned behaviour of selecting alternate

features, however this behaviour was limited to the beginning of the simulation which suggests the agent was indeed optimising behaviour in the correct direction but due to the massive observation space was unable to completely explore the options and discover the preferred trajectory. This suggests that the agent could improve simply with additional training, however the current agent was trained for 100 000 episodes which took almost 8 full days of training using the UTS high powered computing cluster and only learned to observe 8 unique features out of the 100 total. As such it would be much more advantageous to optimise the training method and network structure, as the current structure produced many difficulties during development. Some recommendations of improvements are already available, such as the addition of a high-performance classifier to help guide the agent to correct labels and reduce the number of complex samples the RL agent encounters.

Since the primary goal of this chapter was to replicate the results of another paper, the novel contribution of the chapter is less pronounced than the other chapters. The feature extraction method used in this experiment is novel, however with the negative results and an unsure conviction lead to hesitation in recommending similar future approaches. Using reinforcement learning to prune future AI classifiers is a fairly broad claim that is likely to have been attempted within other fields. For rehabilitation purposes this may be an unprecedented idea, as it was not found during the literature review. In the end, there was very little greater contribution to the overall knowledge base, but key factors were discovered for the purposes of this project, such as the need for classification systems outside of reinforcement learning if the intended goal of the system was real-time classification that could be embedded into a wearable orthosis.

Overall the work completed in this chapter provided deep insight into the development of reinforcement learning agents, specifically discrete action space systems such as Deep Q-Networks. The limitations of RL became clear and suggested other forms of machine learning for tasks such as classification. The explanation for why the system was unable to learn to classify data is not a simple or direct answer, with many components of the system design believed to play a part in the suboptimal results. Network architecture and hyperparameters were essentially chosen based on existing papers and examples, with small alterations to find improved functionality. If the initial selection was invalid then all follow-up tests were unlikely to succeed. The input data had no obvious physical or mathematical foundation due to the black-box nature of neural networks which makes validation difficult to perform. Finally due to the observation space being so large even it becomes difficult to determine if the lack of convergence is derived from a poorly designed system or simply due to the probabilistic nature

of reinforcement learning. These three topics combined make it very challenging to identify any singular point of failure in the experiments. Further research can be performed with almost guaranteed improvements, however this chapter was originally supposed to be a small component of the overall project and any additional information gained from this will not greatly support the overarching goals. For this reason, this chapter must conclude by providing insight into the limitations of reinforcement learning, without offering any novel solutions on how to minimise these limitations.

Chapter 6 - Conclusion

6.1. Summary

6.1.1. Individual Experimental Work

In this thesis a new control configuration selection method was proposed to identify which input-output pairing required the least amount of modification to become a passive system. By using the concept of input feedforward passivity this 'distance to passivity' could be numerically evaluated, with the minimal value calculated through the use of convex optimisation and linear matrix inequalities. These same techniques were implemented to guarantee system passivity through the addition of a diagonal feedforward system with a specific structure that naturally rises from using the controller canonical representation of the original system. This form guarantees the feedforward system is diagonal and contains strictly proper transfer functions. As differences between theoretical models and real-world systems can lead to disastrous consequences, minimising the deviations that must be made during simulations effectively improves the reliability of the generated results, and allows smaller system changes to the physical system once information from the simulation is amalgamated and practical implementation begins.

It was originally believed this newly developed method of passivation could be applied to any MIMO system as long as the number of inputs and number of outputs were equal. However during testing it was discovered that the proposed method for guaranteeing passivity was not applicable to all systems, and the optimisation solver could not generate feasible solutions to all systems during testing. This led to the expansion of the project which aimed to discover the required system characteristics that were necessary to utilise the optimisation passivation technique. After review and testing, it was discovered that the technique could guarantee passivity to systems that met specific criteria that can be roughly described by the diagonal dominance of the system matrix structure. The technique was shown to always succeed when the relative degree of the transfer functions in the diagonal elements of the matrix were greater than or equal to 1, AND the relative degree of the off-diagonal transfer functions were greater than the relative degree of the diagonal elements. Due to how the calculations of the matrix in question are performed, the matrix is guaranteed to be symmetric, and comparison of diagonal and off-diagonal elements can be performed across either rows or columns. These restrictions to which systems could be used with the proposed optimisation method were presented as a

lemma in the results of Chapter 3. By forcing the system into a passive state, the design of the controller becomes less restrictive, and a wider variety of physical options available to perform. As a real-world example, where a previous control system would require hard-coded limitations on assistive forces and velocities which may act as a boundary for performance, a passive-guaranteed system would not require such bounds and could reach closer to an individual's physical limitations as they are inherently included within the coupled system (although additional safety precautions would likely be included regardless).

With the guaranteed stability of the coupled system, system redundancies that existed to prevent specific scenarios can be removed, as those scenarios are no longer mathematically achievable. This may come in the form of a simpler controller algorithm or in the form of less hardware built into the robotic device. Other than economic benefits, user comfort will also become a higher priority if safety is guaranteed through software. Some redundancies that restrict which physical motions are possible under the guise of safety could also be removed, allowing a wider range of physical assistance and generally higher rehabilitation performance by offering a more customised training session.

For the reinforcement learning section of the project, it was recorded that controller tuning could be performed through reinforcement learning when aiming to control a 1 degree-of-freedom robotic model resembling an ankle rehabilitation device. Although not innovative work, adaptive PID controllers and adaptive admittance controllers were both successfully created and trained to track a variety of reference signals in several different environments. The novelty of the experiments resided in the comparison and analysis of various controllers to determine which scenarios required which controller, and whether the more advanced adaptive controllers were always the most effective choice. Essentially, chapter 4 focused on weighing the positives and negatives of AI-based controller design and identifying whether it is justifiable for a simple ankle rehabilitation robot.

Experimental results showed that classical PID controllers with constant gain parameters performed better than the RL-based adaptive controllers when the operating conditions remained identical throughout the tuning process and simulation; alternative forms of adaptive control were not tested for comparison. When operating conditions did not remain constant, which was simulated though a sudden drop in input torque signal to represent motor failure, the adaptive PID controller outperformed the classical PID controller by a wide margin. While the constant controller failed to converge once the motor had failed, the adaptive controller retained its tracking performance. Using adaptive controllers can therefore be seen as a form of

fault-tolerant control as it is able to reject internal system failures. External disturbance rejection was not tested for PID control. These results showed the benefits to RL-based tuning as the trained agent was able to counteract an environmental change it had never experienced during training. This general principle of adapting to unseen environments is paramount for a general-use rehabilitation device, especially if operation in a non-controlled environment is expected and physical obstacles are likely to impede a predicted motion path.

One notable shortcoming in these tests was the extremely simple operating environment simulating motion through free space and never experiencing any external resistances as would be expected during proper gait or activities of daily living. As such, the effects of RL-based adaptive control were expanded into admittance and impedance controllers, with a simulated environment containing physical obstacles. These obstacles were overly simple as well, but were able to show the effects of admittance controllers and their ability to reduce the forces that would be experienced by a human user and avoid physically harmful motions. While the RL-based adaptive admittance controller did outperform the constant admittance controller, a basic switching regime to alternate admittance parameters based on current position seemed to be favourable with respect to response time in changing environments. All controllers still experienced overshoot in the reference tracking, which may cause physical distress to the participant, however this is likely due to the non-adaptive PID controller being used in these tests, so both controllers being adapted simultaneously may result in better operations. As this is very simple implementation for reinforcement learning, it is still a viable option for adaptive admittance control. This work is also contained in Chapter 4.

The final set of experimental work aimed to identify the limitations of reinforcement learning by creating an electromyography hand gesture classifier and compare its efficacy to the well-established forms of classification such as convolutional neural networks. The work originated with the goal of optimising other classifiers by automatically generating signal features and iteratively eliminating the least important features relative to correct classification. Once completed, only the features that provided the highest improvement to classification accuracy were deemed necessary and would remain in the classification network. By simplifying the classifier network, the classification speed would increase and reduce any delays between muscle activation and robotic assistance. This experiment essentially aimed to prune a neural network to be more efficient in real-time classification tasks that would be expected within rehabilitation. Final results of the experiments failed to successfully eliminate all redundant features and correctly classify the EMG data. The limitations of reinforcement learning became

apparent for problems with extremely large state spaces, as the agent was unable to properly explore enough to find an optimal solution. Additional tests were performed on the viability of data manipulation techniques; Fast Fourier Transform and Continuous Wavelet Transform were compared through CNN classifier accuracy and real-time spectrogram generation. CWT was shown to produce spectrograms that could be classified more accurately than the FFT spectrograms, however the generation of said spectrograms was drastically slower and restricted the feasibility of CWT as a real-time classification technique. Overall, reinforcement learning was shown to be ill-equipped for classification purposes when alternate methods function extremely effectively with much easier implementation. Further details are provided in Chapter 5.

As a brief summary of the major contributions of this work, this thesis has produced:

1) An algorithm for guaranteeing a system is passive through the use of input feedforward passivity and optimisation solvers
2) A mathematical lemma dictating the conditions of a system that must be met to utilise the aforementioned passivation algorithm
3) A reinforcement learning agent for a 1 degree-of-freedom ankle rehabilitation robot with statistical analysis to reiterate which scenarios benefit most from RL-based adaptive control
4) Analysis of convolutional neural network comparisons for EMG-signal classification and the conclusion that these well-established technologies are better suited than reinforcement learning for standard classification tasks

6.1.2. Experimental Works Combined

Each chapter of experimental work was quite remote in its exploration, with little crossover between the information being examined. Despite this, the overall project aimed to utilise all collated knowledge to be simultaneously implemented into a rehabilitation robot and unite the experimental works together. For the stages of rehabilitation robot development the following stages must be performed:

1) Determine which physical actions the rehabilitation robot should assist in performing. This initial decision will dictate how the EMG data is collected; placing electrodes on the appropriate muscle activation points will affect both EMG classification and the system's ability to be passivated.

2) Guarantee passivity to guarantee system stability even under uncertain environments or systems with many interacting subsystems (loop interactions for MIMO systems). Guaranteeing passivity of the system guarantees a passive controller is all that is necessary for stability and avoids additional redundancies designed for disturbance rejection or fault-tolerance.
3) Use some form of machine learning to transform EMG data into a predicted joint motion signal to be used as a reference signal. This reference signal will guide the robotic device on how it will assist with motion, so real-time conversion is required with minimal delay between user intention and robotic motion to increase safety and user comfort.
4) Use reinforcement learning to utilise adaptive controllers capable of adjusting robot behaviour to be environment-dependent. Controller design must occur after the plant modelling stage. An adaptive PID controller will dictate the system's reference tracking abilities in free space, allowing characteristics such as response time to be set to human-safe levels. An adaptive admittance controller will dictate the system's response to external forces such as the human force component or environmental stiffness changes.

An additional chapter was originally intended to perform reinforcement learning techniques to validate the control configuration selection method proposed in chapter 3. This chapter would have tied the concepts of reinforcement learning and passivity together by displaying its ability to guarantee passivity in a similar way to the optimisation techniques. The use of neural networks (and other AI such as genetic algorithms and fuzzy logic) for input output pairings has been attempted to determine optimal pairings. By measuring interactions between signals, a neural network will be able to discover the best pairing to reduce crosstalk and achieve decentralised control. The purpose of creating an algorithm that could use optimisation toolboxes to achieve the same goals was to try and avoid the long training times associated with artificial intelligence techniques, instead using semi-definite programming to return a solution much faster. Unfortunately this experimental work was unable to be completed due to a combination of factors. The predominant reason can be summarised as time constraints caused by the unexpected results in chapter 3 requiring further analysis, and several project pivots due to the COVID-19 pandemic restricting laboratory access to the PaddleBot device.

6.2. Future Works & Project Continuation

With each chapter focusing on individual topics and majority of the experimental work being performed through simulations, there is still a lot of progress to be made in the pursuit of improving a real-world rehabilitation robot's measurable performance. With the number of unique engineering fields embedded within rehabilitation robotics, the abilities and effectiveness of these devices is constantly improving and adapting. Changing materials, battery life, and sensory abilities will all need to be considered within the control system to account for the physical changes these developments cause. From this, it should be understood that future works for the control system must always adapt to new hardware technologies.

Some specific follow-up experiments are discussed below.

6.2.1. CCS via RL

Using optimisation techniques to calculate the input-output pairings is extremely beneficial for its guarantee of passivity and mathematical certainty. However as the optimisation solver requires the problem to take the form of a convex optimisation problem it is restricted on which systems can be operated upon. Additionally, a mathematical model of the system must be available for the application to begin, which may be difficult for some advanced robotic devices or systems estimated through function approximators. To work around these issues it is recommended to conduct further research into utilising reinforcement learning for input-output pairing selection. Originally planned for the project, a reinforcement learning agent that is capable of observing system loop interactions and minimising characteristics that may lead to instability (such as non-passivity) has the potential to provide similar results to the proposed optimisation CCS method without the need for a detailed model. The lack of required model would allow further application of passivity-based CCS methods, with the immediate trade-off of requiring long training sessions for each system. An RL agent may be trained to identify loop interactions for any system and its potential configurations, but many unique samples would be required for the training sessions, and some mathematical approach to numerically represent the necessary conditions. As such, it is likely that using the developed optimisation method may assist in developing an RL agent, and the RL agent may be able to validate the results of the optimisation method and expand on the applicable cases and real-world examples.

Before beginning the development, it is important to note that reinforcement learning is best suited for determining sequential actions within an unknown environment, as seen by the

results in Chapter 5. For the purposes of CCS and producing only a single pairing recommendation, RL may not be the most appropriate form of machine learning to implement. The use of neural networks, genetic algorithms, and fuzzy logic have been employed for solving the input-output pairing problem (Khaki-Sedigh and Moaveni 2009: 10). This can act as a good starting point for further development and comparisons of techniques. Performing these tasks would also help unite all experimental work done within this thesis under the banner of reinforcement learning based rehabilitation improvements.

6.2.2. Apply passivity method to robot

All experimental work for the optimisation technique was performed on externally sourced transfer functions that were not related to the rehabilitation robotics the technique was intended for. Utilising 3D robotic models such as the PaddleBot in place of the transfer function matrix will remove a layer of abstractness from the results produced so far.

Once simulations using robotic models are possible the next stage will be to deploy the recommended control systems into functioning MIMO robotic systems such that the real-world effects of the passivity can be measured through physical measurements such as velocity and steady state error. By collecting this data it will become much easier to justify the claims of "improved performance" that the project aimed to achieve.

6.2.3. Improve modelled environment for adaptive tests

The reinforcement learning experimental work showed the benefits of adaptive controllers in changing environments. These environments were extremely simple and not true representations of a real-world environment. For the adaptive PID experiments there was no external environment interactions and the changing environment occurred within the robotic actuators (which is the environment from the perspective of the controller and RL agent). To better simulate real-world conditions the environmental models must be updated. This step is especially important for the adaptive admittance controller, as the interaction forces are dictated by the environment and will directly affect tracking performance. Current changing environments were sudden changes in human stiffness that would be unlikely for actual human response to stimuli. Further research into the stiffness of the human body under different external conditions will allow a more accurate representation of how the human and robot will interact in practice.

Comparisons between the adaptive RL-based controller and more state-of-the-art controllers may assist in further development. As it currently stands, the adaptive RL controllers were compared to their non-adaptive counterparts (adaptive PID vs classical PID, and adaptive admittance vs classical admittance), so the conclusion that the adaptive controller is the best suited controller for robotic purposes does not consider alternative controllers such as model predictive control, which is often used for its ability to function with unmodelled dynamics and disturbances. A large portion of newer control techniques are closer to machine learning in nature, relying on data-collection for model prediction. Comparing reinforcement learning to other machine learning forms is a valid future experiment that must be constantly revisited as new techniques are developed within the field.

Another approach to improving and validating adaptive performance is to collect real motion data to use as the input reference signal for the system. Current setup uses generated signals in the form of sine waves, which may be too smooth for actual gait motions, and step functions, which is not a natural motion within daily activities. Unlikely to affect the results at a large scale, this small improvement to the system will help close the gap between simulation and real-world deployment and potentially reveal unexpected safety issues that may otherwise go overlooked.

6.2.4. Dual-agent controller tuning

Current RL-based adaptive control experiments focused on either the PID controller or the admittance controller individually, and did not try to tune the controllers simultaneously. Experiments were performed with an actor-critic network aiming to tune both a PD controller and an admittance controller simultaneously, however no positive results were ever produced and the agent never learned any interesting or viable reference tracking behaviours. It is possible that changes to the training hyperparameters or reward function could have improved the results, however several different versions were attempted with little success, so alternate approaches may be necessary. With the increase to the number of actions each step, the state-action space becomes much larger and more difficult to comprehensively explore. To produce the best results each controller should be designed with the other in mind – a difficult task if both controllers are adaptive and changing in accordance to the environment. It has been reported that multi-agent reinforcement learning is more efficient at determining environmental dynamics when the agents work cooperatively and share knowledge. Research into such topics as federated reinforcement learning may help develop a new approach to tune both controller simultaneously and produce better results than the current adaptive admittance

controller tests, which relied on a constant PD controller that caused overshoot in cases that would have preferably been avoided.

Experiments involving multi-agent reinforcement learning could be compared to the existing work, as well as additional artificial intelligence techniques for adaptive control. Only comparing the experimental work to basic PID controllers does not effectively highlight the values that RL-based adaptive control brings to operation. Alternative controllers such as linear quadratic Gaussian control or fuzzy control should be tested and compared to both single- and multi-agent reinforcement learning techniques to show the advantages and disadvantages of some of the more modern control techniques.

6.2.5. Multi-agent EMG classification & extensions

For similar reasons discussed above, multi-agent reinforcement learning may be able to alleviate several of the problems that were present within the RL-based EMG classification experiment. Additional agents will allow state space exploration to be performed more efficiently at the cost of computing power. Although the conclusion of the experiment was that reinforcement learning was ill-suited for classification tasks, if the experiment were to be continued then the additional techniques employed by Janisch, Pevný, and Lisý (2019) are recommended extensions. Pre-training Q-values that do not depend on future states can effectively be set before training using supervised learning techniques to reduce the state space requiring first-hand experiences. Including a high-performance classifier (typically non-neural network based) as one possible action will also improve training speed, sample efficiency, and bias the training towards the high-performance classifier results.

As the long-term plan for integrating the EMG classification and previous experimental work together was to use the classifications as the reference signals within the reference tracking control loop, the EMG-to-motion decoder must be adjusted to output numerical data rather than categorical data. This act will require a narrower classification, as the number of categories will invariably be less than the number of angular displacement brackets necessary for the specified precision of rehabilitation. This will be difficult as the muscle activations to swing a limb 20° will be similar to the activation to swing the same limb 50°. Changes may occur depending on specific thresholds crossed or simply due to duration of muscle activation, so feature extraction from the EMG signals will be an important point of analysis. The test to automate feature extraction and use RL to iteratively eliminate all but the most fundamental features may become crucial to achieve this step of the project, so the aforementioned extensions to RL classification may become necessary.

REFERENCES

Al-Quraishi, M. S., I. Elamvazuthi, S. A. Daud, S. Parasuraman, and A. Borboni. 2018. 'EEG-Based Control for Upper and Lower Limb Exoskeletons and Prostheses: A Systematic Review', *Sensors (Basel)*, 18.

Allard, Ulysse Côté, François Nougarou, Cheikh Latyr Fall, Philippe Giguère, Clément Gosselin, François Laviolette, and Benoit Gosselin. 2016. "A convolutional neural network for robotic arm guidance using sEMG based frequency-features." In, 2464-70. IEEE.

Altman, R., E. Asch, D. Bloch, G. Bole, D. Borenstein, K. Brandt, W. Christy, T. D. Cooke, R. Greenwald, M. Hochberg, and et al. 1986. 'Development of criteria for the classification and reporting of osteoarthritis. Classification of osteoarthritis of the knee. Diagnostic and Therapeutic Criteria Committee of the American Rheumatism Association', *Arthritis Rheum*, 29: 1039-49.

Anand, Akhil Sadanandan, Guoping Zhao, Hubert Roth, and Andre Seyfarth. 2019. 'A deep reinforcement learning based approach towards generating human walking behavior with a neuromuscular model'.

Antoine, Jean-Pierre. 1998. 'The continuous wavelet transform in image processing', *CWI Q*, 11: 323-45.

Ao, D., R. Song, and J. Gao. 2017. 'Movement Performance of Human-Robot Cooperation Control Based on EMG-Driven Hill-Type and Proportional Models for an Ankle Power-Assist Exoskeleton Robot', *IEEE Trans Neural Syst Rehabil Eng*, 25: 1125-34.

Araujo, I. L., M. C. Castro, C. Daltro, and M. A. Matos. 2016. 'Quality of Life and Functional Independence in Patients with Osteoarthritis of the Knee', *Knee Surg Relat Res*, 28: 219-24.

Artemiadis, P. K., and K. J. Kyriakopoulos. 2011. 'A switching regime model for the EMG-based control of a robot arm', *IEEE Trans Syst Man Cybern B Cybern*, 41: 53-63.

Artemiadis, Panagiotis K., Pantelis T. Katsiaris, Minas V. Liarokapis, and Kostas J. Kyriakopoulos. 2011. "On the effect of human arm manipulability in 3D force tasks: Towards force-controlled exoskeletons." In *2011 IEEE International Conference on Robotics and Automation*, 3784-89.

Arts, Lukas P. A., and Egon L. van den Broek. 2022. 'The fast continuous wavelet transformation (fCWT) for real-time, high-quality, noise-resistant time–frequency analysis', *Nature Computational Science*, 2: 47-58.

Atashzar, S. Farokh, Hsien-Yung Huang, Fulvia Del Duca, Etienne Burdet, and Dario Farina. 2020. 'Energetic Passivity Decoding of Human Hip Joint for Physical Human-Robot Interaction', *IEEE robotics and automation letters*, 5: 5953-60.

Atashzar, Seyed Farokh, Mahya Shahbazi, Mahdi Tavakoli, and Rajni V. Patel. 2017. 'A grasp-based passivity signature for haptics-enabled human-robot interaction: Application to design of a new safety mechanism for robotic rehabilitation', *The International journal of robotics research*, 36: 778-99.

Au, S., M. Berniker, and H. Herr. 2008. 'Powered ankle-foot prosthesis to assist level-ground and stair-descent gaits', *Neural Netw*, 21: 654-66.

Bao, J. I. E., Peter L. Lee, Fuyang Wangc, Weibiao Zhou, and Yudi Samyudia. 2000. 'A NEW APPROACH TO DECENTRALISED PROCESS CONTROL USING PASSIVITY AND SECTOR STABILITY CONDITIONS', *Chemical engineering communications*, 182: 213-37.

Bao, Jie, Kwong H. Chan, Wen Z. Zhang, and Peter L. Lee. 2007. 'An experimental pairing method for multi-loop control based on passivity', *Journal of Process Control*, 17: 787-98.

Bao, Jie, P. James McLellan, and J. Fraser Forbes. 2002. 'A passivity-based analysis for decentralized integral controllability', *Automatica (Oxford)*, 38: 243-47.

Bao, Jie, and Peter L. Lee. 2007. *Process Control The Passive Systems Approach* (Springer London: London).

Bao, Jie, Wen Z. Zhang, and Peter L. Lee. 2002. 'Passivity-Based Decentralized Failure-Tolerant Control', *Industrial & engineering chemistry research*, 41: 5702-15.

Basmajian, and De Luca. 1985. 'Muscles Alive: Their functions revealed by elecrtomyography'.

Begovic, Haris, Guang-Quan Zhou, Tianjie Li, Yi Wang, and Yong-Ping Zheng. 2014. 'Detection of the electromechanical delay and its components during voluntary isometric contraction of the quadriceps femoris muscle', *Frontiers in physiology*, 5: 494-94.

Bingjing, Guo, Han Jianhai, Li Xiangpan, and Yan Lin. 2019. 'Human–robot interactive control based on reinforcement learning for gait rehabilitation training robot', *International Journal of Advanced Robotic Systems*, 16.

Blaya, J. A., and H. Herr. 2004. 'Adaptive control of a variable-impedance ankle-foot orthosis to assist drop-foot gait', *IEEE Trans Neural Syst Rehabil Eng*, 12: 24-31.

Bressel, Eadric, and Peter J. McNair. 2002. 'The Effect of Prolonged Static and Cyclic Stretching on Ankle Joint Stiffness, Torque Relaxation, and Gait in People With Stroke', *Physical Therapy*, 82: 880-87.

Brockett, Claire L., and Graham J. Chapman. 2016. 'Biomechanics of the ankle', *Orthopaedics and trauma*, 30: 232-38.

Brunnbauer, Axel, Luigi Berducci, Andreas Brandstätter, Mathias Lechner, Ramin Hasani, Daniela Rus, and Radu Grosu. 2021. 'Latent Imagination Facilitates Zero-Shot Transfer in Autonomous Racing'.

Brunton, Steve. 2018. 'Machine Learning Control: Overview'. https://www.youtube.com/watch?v=3yU2k8R9JeU.

Cao, J. 2014. 'Improved delay-dependent stability conditions for MIMO networked control systems with nonlinear perturbations', *ScientificWorldJournal*, 2014: 196927.

Chaffey, Thomas. 2022. 'A rolled-off passivity theorem', *Systems & control letters*, 162: 105198.

Chan, Adrain D. C., and Geoffrey C. Green. 2007. 'Myoelectric control development toolbox', *CMBES Proceedings*, 30.

Chesworth, B. M., and A. A. Vandervoort. 1989. 'Age and passive ankle stiffness in healthy women', *Phys Ther*, 69: 217-24.

Chokroverty, S., and J. Medina. 1978. 'Electrophysiological study of hemiplegia. Motor nerve conduction velocity, brachial plexus latency, and electromyography', *Arch Neurol*, 35: 360-3.

Chowdhury, Rubana H., Mamun B. I. Reaz, Mohd Alauddin Bin Mohd Ali, Ashrif A. A. Bakar, K. Chellappan, and T. G. Chang. 2013. 'Surface electromyography signal processing and classification techniques', *Sensors (Basel, Switzerland)*, 13: 12431-66.

Colgate, J. E., and N. Hogan. 1988. 'Robust control of dynamically interacting systems', *International journal of control*, 48: 65-88.

Colombo, Gery, Matthias Joerg, Reinhard Schreier, and Volker Dietz. 2000. 'Treadmill training of paraplegic patients using a robotic orthosis', *Journal of rehabilitation research and development*, 37: 693-700.

Cornwall, M. W. 1994. 'Effect of temperature on muscle force and rate of muscle force production in men and women', *J Orthop Sports Phys Ther*, 20: 74-80.

Douglas, Brian. 2021. 'Passivity-Based Control to Guarantee Stability'. https://www.youtube.com/watch?v=yFS5PSmlp6E.

Dulac-Arnold, Gabriel, Ludovic Denoyer, Philippe Preux, and Patrick Gallinari. 2011. "Datum-Wise Classification: A Sequential Approach to Sparsity." In, 375-90. Berlin, Heidelberg: Springer Berlin Heidelberg.

Duschau-Wicke, A., J. von Zitzewitz, A. Caprez, L. Lunenburger, and R. Riener. 2010. 'Path control: a method for patient-cooperative robot-aided gait rehabilitation', *IEEE Trans Neural Syst Rehabil Eng*, 18: 38-48.

Erden, Mustafa Suphi, and Bobby Marić. 2011. 'Assisting manual welding with robot', *Robotics and computer-integrated manufacturing*, 27: 818-28.

Fahmy, Rania A., Ragia I. Badr, and Farouk A. Rahman. 2014. 'Adaptive PID Controller Using RLS for SISO Stable and Unstable Systems', *Advances in Power Electronics*, 2014: 1-5.

Fleischer, C., C. Reinicke, and G. Hommel. 2005. "Predicting the intended motion with EMG signals for an exoskeleton orthosis controller." In, 2029-34. IEEE.

Goge, Aarti R., and A. D. C. Chan. 2005. 'Investigating classification parameters for continuous myoelectrically controlled prostheses', *CMBES Proceedings*, 28.

Goodwin, G., S. Graebe, and M. Salgado. 2000. *Control System Design* (Prentice Hall).

Grande, Anthony. 2016. 'Physiotherapy Or Surgery, What Is Best For Me?', Accessed 2023/01/25. https://www.focusphysiotherapy.com/physiotherapy-surgery-best/.

Guan, Zhe, and Toru Yamamoto. 2021. 'Design of a Reinforcement Learning PID Controller', *IEEJ transactions on electrical and electronic engineering*, 16: 1354-60.

Health, Australian Institute of, and Welfare. 2020. "Osteoarthritis." In. Canberra: AIHW.

Heloyse Uliam, Kuriki, Azevedo Fa?bio Mi?colis de, Takahashi Luciana Sanae Ota, Mello Emanuelle Moraes, Filho Ru?ben de Faria Negra?o, and Alves Neri. 2012. "The Relationship Between Electromyography and Muscle Force." In.: Unpublished.

Henrion, Didier, and Michael Šebek. 2009. 'Polynomial and matrix fraction description', *Control Systems, Robotics and Automation*, 7: 211-31.

Hidler, Joseph, Diane Nichols, Marlena Pelliccio, and Kathy Brady. 2005. 'Advances in the Understanding and Treatment of Stroke Impairment Using Robotic Devices', *Topics in stroke rehabilitation*, 12: 22-35.

Hill, Archibald Vivian. 1938. 'The heat of shortening and the dynamic constants of muscle', *Proceedings of the Royal Society of London. Series B - Biological Sciences*, 126: 136-95.

Hogan, N. 1984. 'Adaptive control of mechanical impedance by coactivation of antagonist muscles', *IEEE transactions on automatic control*, 29: 681-90.

Hogan, N., H. I. Krebs, J. Charnnarong, P. Srikrishna, and A. Sharon. 1992. "MIT-MANUS: a workstation for manual therapy and training. ." In *[1992] Proceedings IEEE International Workshop on Robot and Human Communication*, 161-65.

Howell, Mark, and M. Best. 2000. 'On-line PID tuning for engine idle-speed control using continuous action reinforcement learning automata', *Control Engineering Practice*, 8: 147-54.

Huang, Hsiao-Ping, Masahiro Ohshima, and Iori Hashimoto. 1994. 'Dynamic interaction and multiloop control system design', *Journal of Process Control*, 4: 15-27.

Huang, X., F. Naghdy, H. Du, G. Naghdy, and C. Todd. 2015. "Reinforcement learning neural network (RLNN) based adaptive control of fine hand motion rehabilitation robot." In *2015 IEEE Conference on Control Applications (CCA)*, 941-46.

Huang, Y., R. Song, A. Argha, A. V. Savkin, B. G. Celler, and S. W. Su. 2020. 'Continuous Description of Human 3D Motion Intent Through Switching Mechanism', *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, 28: 277-86.

Huang, Yao, Rong Song, Ahmadreza Argha, Branko G. Celler, Andrey V. Savkin, and Steven W. Su. 2021. 'Human Motion Intent Description Based on Bumpless Switching Mechanism for Rehabilitation Robot', *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, 29: 673-82.

Huilcapi, V., X. Blasco, J. M. Herrero, and G. Reynoso-Meza. 2019. 'A Loop Pairing Method for Multivariable Control Systems Under a Multi-Objective Optimization Approach', *IEEE Access*, 7: 81994-2014.

Hunter, Sandra K., Hugo M. Pereira, and Kevin G. Keenan. 2016. 'The aging neuromuscular system and motor performance', *Journal of applied physiology (1985)*, 121: 982-95.

Hyodo, Kashitaro, Tadashi Masuda, Junya Aizawa, Tetsuya Jinno, and Sadao Morita. 2017. 'Hip, knee, and ankle kinematics during activities of daily living: a cross-sectional study', *Brazilian journal of physical therapy*, 21: 159-66.

Hyunglae, Lee, and Neville Hogan. 2013. "Investigation of human ankle mechanical impedance during locomotion using a wearable ankle robot." In, 2651-56. IEEE.

Ibrahim, A. A. S. 2002. "Nonlinear PID controller design using fuzzy logic." In *11th IEEE Mediterranean Electrotechnical Conference (IEEE Cat. No.02CH37379)*, 595-99.

James, B., and A. W. Parker. 1989. 'Active and passive mobility of lower limb joints in elderly men and women', *American journal of physical medicine & rehabilitation*, 68: 162-67.

Jamwal, Prashant K., Sheng Q. Xie, Shahid Hussain, and John G. Parsons. 2014. 'An Adaptive Wearable Parallel Robot for the Treatment of Ankle Injuries', *IEEE/ASME transactions on mechatronics*, 19: 64-75.

Janisch, Jaromír, Tomáš Pevný, and Viliam Lisý. 2019. "Classification with costly features using deep reinforcement learning." In *Proceedings of the AAAI Conference on Artificial Intelligence*, 3959-66.

Jiang, X., M. Gholami, M. Khoshnam, J. J. Eng, and C. Menon. 2019. 'Estimation of Ankle Joint Power during Walking Using Two Inertial Sensors', *Sensors (Basel)*, 19.

Kadhim, Ali M. H., Wolfgang Birk, and Miguel Castano Arranz. 2016. 'Dynamic Relative Gain Array Estimation using Local Polynomial Approximation Approach', *Model, Identificationand Control*, 37: 247-59.

Kailath, Thomas. 1980. *Linear systems* (Prentice-Hall Englewood Cliffs, NJ).

Kelly-Hayes, M. 2010. 'Influence of age and health behaviors on stroke risk: lessons from longitudinal studies', *J Am Geriatr Soc*, 58 Suppl 2: S325-8.

Khaki-Sedigh, A., and B. Moaveni. 2009. *Control Configuration Selection for Multivariable Plants* (Springer Berlin Heidelberg: Berlin, Heidelberg).

Kim, Byungchan, Jooyoung Park, Shinsuk Park, and Sungchul Kang. 2010. 'Impedance Learning for Robotic Contact Tasks Using Natural Actor-Critic Algorithm', *IEEE transactions on systems, man and cybernetics. Part B, Cybernetics*, 40: 433-43.

Kiran, Mariam, and Melis Ozyildirim. 2022. 'Hyperparameter Tuning for Deep Reinforcement Learning Applications', *arXiv preprint arXiv:2201.11182*.

Kobetic, Rudi, Curtis S. To, John R. Schnellenberger, Musa L. Audu, Thomas C. Bulea, Richard Gaudio, Gilles Pinault, Scott Tashman, and Ronald J. Triolo. 2009. 'Development of hybrid orthosis for standing, walking, and stair climbing after spinal cord injury', *Journal of rehabilitation research and development*, 46: 447-62.

Kofinas, Panagiotis, and Anastasios Dounis. 2019. 'Online Tuning of a PID Controller with a Fuzzy Reinforcement Learning MAS for Flow Rate Control of a Desalination Unit', *Electronics*, 8: 231.

Kon, Elizaveta, Giuseppe Filardo, Matej Drobnic, Henning Madry, Mislav Jelic, Niek van Dijk, and Stefano Della Villa. 2012. 'Non-surgical management of early knee osteoarthritis', *Knee Surgery, Sports Traumatology, Arthroscopy*, 20: 436-49.

Lee, Daesoo, Seung Jae Lee, and Solomon C. Yim. 2020. 'Reinforcement learning-based adaptive PID controller for DPS', *Ocean Engineering*, 216.

Lee, H., and N. Hogan. 2016a. 'Energetic Passivity of the Human Ankle Joint', *IEEE Trans Neural Syst Rehabil Eng*, 24: 1416-25.

———. 2016b. "Essential considerations for design and control of human-interactive robots." In *2016 IEEE International Conference on Robotics and Automation (ICRA)*, 3069-74.

Lee, H., H. I. Krebs, and N. Hogan. 2014. 'Multivariable Dynamic Ankle Mechanical Impedance With Active Muscles', *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, 22: 971-81.

Lee, Hyunglae, Elliott J. Rouse, and Hermano Igo Krebs. 2016. 'Summary of Human Ankle Mechanical Impedance During Walking', *IEEE journal of translational engineering in health and medicine*, 4: 2100407-07.

Li, Li, and Lorna L. Ogden. 2012. 'Muscular activity characteristics associated with preparation for gait transition', *Journal of sport and health science*, 1: 27-35.

Liu, Chujun, Andrew G. Lonsberry, Mark J. Nandor, Musa L. Audu, Alexander J. Lonsberry, and Roger D. Quinn. 2019. 'Implementation of Deep Deterministic Policy Gradients for Controlling Dynamic Bipedal Walking', *Biomimetics (Basel, Switzerland)*, 4: 28.

Lofberg, Johan. 2004. "YALMIP: A toolbox for modeling and optimization in MATLAB." In, 284-89. IEEE.

Lotze, M., C. Braun, N. Birbaumer, S. Anders, and L. G. Cohen. 2003. 'Motor learning elicited by voluntary drive', *Brain*, 126: 866-72.

Mathworks. 2022a. 'Classify Arm Motions Using EMG Signals and Deep Learning', Accessed 2022/07/04. https://au.mathworks.com/help/signal/ug/classify-arm-motions-using-emg-signals-and-deep-learning.html.

———. 2022b. 'How PID Autotuning Works', Accessed 2022/04/08. https://au.mathworks.com/help/slcontrol/ug/how-pid-autotuning-works.html.

———. 2022c. 'rlTD3AgentOptions - Noise Models', Accessed 2022/05/11. https://au.mathworks.com/help/releases/R2021a/reinforcement-learning/ref/rltd3agentoptions.html.

———. 2022d. 'Speech Command Recognition Using Deep Learning', Accessed 2022/08/18. https://au.mathworks.com/help/releases/R2021a/deeplearning/ug/deep-learning-speech-recognition.html.

———. 2022e. 'Train Biped Robot to Walk Using Reinforcement Learning Agents', Accessed 2022/03/31. https://au.mathworks.com/help/releases/R2021a/reinforcement-learning/ug/train-biped-robot-to-walk-using-reinforcement-learning-agents.html.

Mc Avoy, Thomas, Yaman Arkun, Rong Chen, Derek Robinson, and P. David Schnelle. 2003. 'A new approach to defining a dynamic relative gain', *Control Engineering Practice*, 11: 907-14.

Meng, Wei, Quan Liu, Zude Zhou, Qingsong Ai, Bo Sheng, and Shengquan Xie. 2015. 'Recent development of mechanisms and control strategies for robot-assisted lower limb rehabilitation', *Mechatronics*, 31: 132-45.

Merlo, A., M. Manca, S. Cavazza, G. Ferraresi, and P. Marchi. 2009. 'Crosstalk in surface EMG signals of the brachioradialis muscle', *Gait & Posture*, 29: e26-e27.

Miller, Ross H. 2018. 'Hill-Based Muscle Modeling.' in Bertram Müller, Sebastian I. Wolf, Gert-Peter Brueggemann, Zhigang Deng, Andrew McIntosh, Freeman Miller and William Scott Selbie (eds.), *Handbook of Human Motion* (Springer International Publishing: Cham).

Moraux, Amélie, Aurélie Canal, Gwenn Ollivier, Isabelle Ledoux, Valérie Doppler, Christine Payan, and Jean-Yves Hogrel. 2013. 'Ankle dorsi- and plantar-flexion torques measured by dynamometry in healthy subjects from 5 to 80 years', *BMC musculoskeletal disorders*, 14: 104-04.

Moritani, T., M. Muro, and A. Nagata. 1986. 'Intramuscular and surface electromyogram changes during muscle fatigue', *Journal of applied physiology (Bethesda, Md. : 1985)*, 60: 1179-85.

Narasimhan, K. Adith. 2021. 'Mean Normalization and Feature Scaling — A simple explanation', Accessed 2022/08/18. https://medium.com/analytics-vidhya/mean-normalization-and-feature-scaling-a-simple-explanation-3b9be7bfd3e8.

Neary, P. 2018. "Automatic Hyperparameter Tuning in Deep Convolutional Neural Networks Using Asynchronous Reinforcement Learning." In *2018 IEEE International Conference on Cognitive Computing (ICCC)*, 73-77.

Nise, Norman S. 2011. *Control systems engineering* (John Wiley & Sons, Inc: Hoboken, NJ).

Nordander, C., J. Willner, G. Å Hansson, B. Larsson, J. Unge, L. Granquist, and S. Skerfving. 2003. 'Influence of the subcutaneous fat layer, as measured by ultrasound, skinfold calipers and BMI, on the EMG amplitude', *European Journal of Applied Physiology*, 89: 514-19.

OpenAI. 2018. 'OpenAI: Spinning Up', Accessed 11/11/2022. https://spinningup.openai.com/en/latest/.

OpenAI, Christopher Berner, Greg Brockman, Brooke Chan, Vicki Cheung, Przemysław Dębiak, Christy Dennison, David Farhi, Quirin Fischer, Shariq Hashme, Chris Hesse, Rafal Józefowicz, Scott

Gray, Catherine Olsson, Jakub Pachocki, Michael Petrov, Henrique P. d O. Pinto, Jonathan Raiman, Tim Salimans, Jeremy Schlatter, Jonas Schneider, Szymon Sidor, Ilya Sutskever, Jie Tang, Filip Wolski, and Susan Zhang. 2019. 'Dota 2 with Large Scale Deep Reinforcement Learning'.

Perry, Thomas A., Alan Silman, David Culliford, Lucy Gates, Nigel Arden, Catherine Bowen, Ian A. Harris, Chelsea Nicole Dyer, Andrew Beischer, Ilana Ackerman, Ove Furnes, Geir Hallan, Keijo T. Mäkelä, Miika Stenholm, Anders Henricson, John McKie, Dawson Muir, and Consortium the International Ankle Arthroplasty Registry. 2022. 'Survival of primary ankle replacements: data from global joint registries', *Journal of Foot and Ankle Research*, 15: 33.

Phinyomark, Angkoon, Pornchai Phukpattaranont, and Chusak Limsakul. 2012. 'Feature reduction and selection for EMG signal classification', *Expert Systems with Applications*, 39: 7420-31.

Pickle, Nathaniel T., Alena M. Grabowski, Arick G. Auyang, and Anne K. Silverman. 2016. 'The functional roles of muscles during sloped walking', *Journal of biomechanics*, 49: 3244-51.

Pilarski, P. M., M. R. Dawson, T. Degris, F. Fahimi, J. P. Carey, and R. S. Sutton. 2011. "Online human training of a myoelectric prosthesis controller via actor-critic reinforcement learning." In *2011 IEEE International Conference on Rehabilitation Robotics*, 1-7.

Qin, Yunxiao, Weiguo Zhang, Jingping Shi, and Jinglong Liu. 2018. "Improve PID controller through reinforcement learning." In *2018 IEEE CSAA Guidance, Navigation and Control Conference (CGNCC)*, 1-6.

Rábago, Christopher A., Jennifer Aldridge Whitehead, and Jason M. Wilken. 2016. 'Evaluation of a Powered Ankle-Foot Prosthesis during Slope Ascent Gait', *PloS one*, 11: e0166815-e15.

Raez, M. B. I., M. S. Hussain, and F. Mohd-Yasin. 2006. 'Techniques of EMG signal analysis: detection, processing, classification and applications', *Biological procedures online*, 8: 11-35.

Ramsey, P. L., and W. Hamilton. 1976. 'Changes in tibiotalar area of contact caused by lateral talar shift', *J Bone Joint Surg Am*, 58: 356-7.

Roboticsbiz, Editorial. 2020. 'Spinal Cord Injury: Pros and Cons of Robotic Exoskeletons', Accessed 2023/01/25. https://roboticsbiz.com/spinal-cord-injury-pros-and-cons-of-robotic-exoskeletons/.

Romeo, Ortega, Romero Jose Guadalupe, Borja Pablo, and Donaire Alejandro. 2021. 'Motivation and Basic Construction of PID Passivity‐Based Control.' in, *PID Passivity-Based Control of Nonlinear Systems with Applications* (IEEE).

Rossini, P. M., and G. Dal Forno. 2004. 'Integrated technology for evaluation of brain function and neural plasticity', *Phys Med Rehabil Clin N Am*, 15: 263-306.

Sam, S. M., and T. S. Angel. 2017. "Performance optimization of PID controllers using fuzzy logic." In *2017 IEEE International Conference on Smart Technologies and Management for Computing, Communication, Controls, Energy and Materials (ICSTM)*, 438-42.

Song, Chengjie , Chunjie Chen, Yanjie Li, and Xinyu Wu. 2018. "Deep Reinforcement Learning Apply in Electromyography Data Classification." In *IEEE International Conference on Cyborg & Biomic Systems*, 505-10. Shenzhen, China: IEEE.

Song, Peng, Yueqing Yu, and Xuping Zhang. 2019. 'A Tutorial Survey and Comparison of Impedance Control on Robotic Manipulation', *Robotica*, 37: 801-36.

Song, Rong, Kai-yu Tong, Xiaoling Hu, and Wei Zhou. 2013. 'Myoelectrically controlled wrist robot for stroke rehabilitation', *Journal of NeuroEngineering and Rehabilitation*, 10: 52.

Steultjens, M. P. M., J. Dekker, M. E. van Baar, R. A. B. Oostendorp, and J. W. J. Bijlsma. 2000. 'Range of joint motion and disability in patients with osteoarthritis of the knee or hip', *Rheumatology*, 39: 955-61.

Stratton-Powell, A. 2018. 'On the Failure of Total Ankle Replacement: A Retrieval Analysis'.

Su, Steven W., Jie Bao, and Peter L. Lee. 2006. 'Conditions on input disturbance suppression for multivariable nonlinear systems on the basis of feedforward passivity', *International journal of systems science*, 37: 225-33.

Sutton, Richard S., and Andrew G. Barto. 2018. *Reinforcement learning : an introduction* (The MIT Press: Cambridge, Massachusetts).

Szepesvári, Csaba. 2010. 'Algorithms for reinforcement learning', *Synthesis lectures on artificial intelligence and machine learning*, 4: 1-103.

Toumi, Hechmi, Georges Poumarat, Thomas M. Best, Alain Martin, John Fairclough, and Mike Benjamin. 2006. 'Fatigue and muscle-tendon stiffness after stretch-shortening cycle and isometric exercise', *Applied physiology, nutrition, and metabolism*, 31: 565-72.

Tucker, M. R., J. Olivier, A. Pagel, H. Bleuler, M. Bouri, O. Lambercy, R. Millan Jdel, R. Riener, H. Vallery, and R. Gassert. 2015. 'Control strategies for active lower extremity prosthetics and orthotics: a review', *J Neuroeng Rehabil*, 12: 1.

Uhlmann, Jeffrey. 2019. 'On the Relative Gain Array (RGA) with singular and rectangular matrices', *Applied Mathematics Letters*, 93: 52-57.

van Hasselt, Hado, Arthur Guez, and David Silver. 2015. 'Deep Reinforcement Learning with Double Q-learning'.

Venayagamoorthy, G. K., R. K. Sharma, P. K. Gautam, and A. Ahmadi. 2016. 'Dynamic Energy Management System for a Smart Microgrid', *IEEE Trans Neural Netw Learn Syst*, 27: 1643-56.

Waldman, Genna, Chung-Yong Yang, Yupeng Ren, Lin Liu, Xin Guo, Richard L. Harvey, Elliot J. Roth, and Li-Qun Zhang. 2013. 'Effects of robot-guided passive stretching and active movement training of ankle and mobility impairments in stroke', *NeuroRehabilitation (Reading, Mass.)*, 32: 625-34.

Wu, Wen, Kate Saul, and He Helen Huang. 2020. 'Using Reinforcement Learning to Estimate Human Joint Moments via EMG Signals or Joint Kinematics: An Alternative Solution to Musculoskeletal-Based Biomechanics', *Journal of biomechanical engineering*.

Yang, Jin, Hang Su, Zhijun Li, Di Ao, and Rong Song. 2016. 'Adaptive control with a fuzzy tuner for cable-based rehabilitation robot', *International Journal of Control, Automation and Systems*, 14: 865-75.

Yao, Shaowei, Yu Zhuang, Zhijun Li, and Rong Song. 2018. 'Adaptive Admittance Control for an Ankle Exoskeleton Using an EMG-Driven Musculoskeletal Model', *Frontiers in Neurorobotics*, 12.

Yousif, Hayder A., Ammar Zakaria, Norasmadi Abdul Rahim, Ahmad Faizal Bin Salleh, Mustafa Mahmood, Khudhur A. Alfarhan, Latifah Munirah Kamarudin, Syed Muhammad Mamduh, Ali Majid Hasan, and Moaid K. Hussain. 2019. 'Assessment of Muscles Fatigue Based on Surface EMG Signals Using Machine Learning and Statistical Approaches: A Review', *IOP conference series. Materials Science and Engineering*, 705: 12010.

Zames, G. 1966. 'On the input-output stability of time-varying nonlinear feedback systems Part one: Conditions derived using concepts of loop gain, conicity, and positivity', *IEEE transactions on automatic control*, 11: 228-38.

Zhang, Chiyuan, Oriol Vinyals, Remi Munos, and Samy Bengio. 2018. 'A Study on Overfitting in Deep Reinforcement Learning'.

Zhang, Feng, Pengfeng Li, Zeng-Guang Hou, Zhen Lu, Yixiong Chen, Qingling Li, and Min Tan. 2012. 'sEMG-based continuous estimation of joint angles of human legs by using BP neural network', *Neurocomputing*, 78: 139-48.

Zhang, Juanjuan, and Chien Chern Cheah. 2015. 'Passivity and Stability of Human-Robot Interaction Control for Upper-Limb Rehabilitation Robots', *IEEE transactions on robotics*, 31: 233-45.

Zhang, Mingming, T. Claire Davies, and Shane Xie. 2013. 'Effectiveness of robot-assisted therapy on ankle rehabilitation--a systematic review', *Journal of NeuroEngineering and Rehabilitation*, 10: 30-30.

Zhang, Mingming, Sheng Q. Xie, Li Xiaolong, Zhu Guoli, Meng Wei, Huang Xiaolin, and Allan J. Veale. 2018. 'Adaptive Patient-Cooperative Control of a Compliant Ankle Rehabilitation Robot (CARR) With Enhanced Training Safety', *IEEE transactions on industrial electronics (1982)*, 65: 1398-407.

Zhang, Wen Z., Jie Bao, and Peter L. Lee. 2002. 'Decentralized Unconditional Stability Conditions Based on the Passivity Theorem for Multi-loop Control Systems', *Industrial & engineering chemistry research*, 41: 1569-78.

Zhang, Z., K. Yang, J. Qian, and L. Zhang. 2019. 'Real-Time Surface EMG Pattern Recognition for Hand Gestures Based on an Artificial Neural Network', *Sensors (Basel)*, 19.

Ziegler, J. G., and N. B. Nichols. 1993. 'Optimum Settings for Automatic Controllers', *Journal of dynamic systems, measurement, and control*, 115: 220-22.