Development, Testing, and Implementation of An Optimized Sliding Control Scheme for Unstable Under-Actuated Mechanical Systems

by

Lucas Allan Zoël Pupek

Bachelor of Science in Mechanical Engineering, UNB, 2016

A THESIS SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF

Master of Science in Mechanical Engineering

In the Graduate Academic Unit of Mechanical Engineering

Supervisor(s):	Rickey Dubay, PhD, Mechanical Engineering/Advanced Controls
Examining Board:	Phil Garland, PhD, Mechanical Engineering/Solid Mechanics Muhammad Afzal, PhD, Mechanical Engineering/Biomass & Biofuels Kaveh Arjomandi, PhD, Civil Engineering/Structural Tiger Jeans, PhD, Mechanical Engineering/Fluid Dynamics, Chair

This thesis is accepted by the Dean of Graduate Studies

THE UNIVERSITY OF NEW BRUNSWICK

October, 2018

Abstract

Many prominent mechanical systems used in today's world, such as robotic manipulators, drones, and unmanned vehicles can be accurately described with nonlinear dynamic models. Sliding Mode Control (SMC) is a powerful tool that can drive uncertain systems towards the desired trajectory while also rejecting disturbances. This is a valuable property when dealing with fast systems that require precise operating points. This thesis details the design, validation, and implementation of a novel sliding control scheme. The sliding surface is made up of proportional, integral, and derivative terms which generate superior performance when compared to other sliding schemes and is easier to design. The controller is then tuned using a custom genetic algorithm in order to achieve optimal performance. A detailed overview of the controller design as well as stability proofs are provided in the following thesis. The new control scheme was implemented on several under-actuated systems to test its performance. The results collected showed that the new Proportional-Integral-Derivative (PID) sliding surface controller produces superior performance when compared to standard sliding and linear schemes.

Acknowledgments

I firstly want to thank my supervisor and mentor, Dr. Rickey Dubay. The hard work contained in these pages was made possible through his constant support, patience, and drive for academic excellence. I would also like to thank Dr. Ricardo Bautista for his theoretical and experimental expertise and advice. Additional thanks go to my colleagues in the Intelligent Controls Laboratory; their friendship and knowledge kept me moving forward.

To my parents, Kathy and Darryl Pupek, words cannot express all they have done for me throughout my academic career. The encouragement they give and example they set have been invaluable to me.

Finally, to my confidant, sounding board, and partner in crime, I thank my girlfriend, Jessica. Her love and support have made the past two years my greatest adventure.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

Control systems have become irreplaceable in modern industry. Several applications exist in all facets of life and today's society depends on them functioning properly. Self driving cars reacting to outside stimuli, robotic manipulators accurately path tracking, and rockets landing themselves are all examples of control systems at work [27]. Today's advances in technology are forcing control theory to produce improved results. The need for better performance is causing a shift from traditional techniques to more complex strategies [26].

When considering control of robotic systems, both precision and stability must be guaranteed. A robotic manipulator that is not precise or accurate is useless to the operator while an unstable exoskeleton could cause severe injuries. It is this dichotomy of accuracy and stability that drives the pro-

gression of mechanical system control schemes; one cannot be sacrificed for the other. The increase in system complexity and operational speed also contribute to the challenge of the control problem. Implemented controllers must be able to preserve operation in a world of automation and under-actuation while producing superior results.

The control scheme developed for this thesis is a Proportional-Integral-Derivative (PID) sliding surface controller. The sliding surface contains proportional, integral, and derivative error terms for both the actuated and under-actuated states. This allows the controller to account for all system states and maintain overall stability. The surface constants are tuned using an evolving population based genetic algorithm in order to obtain the optimal values. This bypasses the need to hand tune the controller and gives valuable insight into the interaction between the sliding surface and the system dynamics.

The following chapters will detail all of the work performed towards the completion of this thesis. Chapter 1 goes over relevant literature and clearly states the objectives and contributions of this thesis. Chapter 2 discusses the necessary control theory. Chapter 3 runs through all details required to design and validate the new control scheme. Chapter 4 contains all of the simulation and implementation tests performed using the new controller as well as other comparison controllers. Finally, Chapter 5 discusses the results collected and provides recommendations and future work.

1.1 Background

Controllers have existed since the dawn of life. An organism regulating its body temperature, eyes tracking movement and limbs grasping objects are all natural examples of controllers at work [27]. While the previously mentioned systems are natural rather than artificial, they demonstrate the link between control systems and life itself. Early controllers were simple but effective designs used to improve their respective processes. An example of one such early controller is the centrifugal flyball governor created by James Watt in 1788 [9]. By using the kinetic energy of the engine, the balls would spin faster and raise up which would limit the amount of fuel to the engine. In doing so, the device was used to control the speed of rotation of the steam engine and is one of the earliest forms of feedback control [9].

The field of control has come a long way since the time of the flyball governor. The many controllers of today are all much more sophisticated in their design and execution. Linear control techniques like proportional-integral-derivative (PID) and model predictive control (MPC) have given way to nonlinear and adaptive schemes. Control structures like sliding mode control, robust adaptive control, nonlinear MPC, and neural network based control are all examples of more advanced controller theories.

These more complex controllers have become necessary due to the rapid

growth of the systems they aim to control. A simple PID can handle a mass-spring-damper system with little issue. However, when trying to stabilize an under-actuated, unstable system, the error based PID struggles unless augmented with supplementary schemes. On the other hand, today's hybrid controllers are able to maintain stability while also tracking setpoints of such a system.

1.2 Literature Review

The following section contains an overview of current and relevant research conducted in the applicable fields. Sliding mode control is the main focus while genetic algorithms and under-actuated systems are also covered. The goal of this section is to show the gap that this thesis is filling. Other basic concepts like mechanical systems, robotics and nonlinear dynamics will be discussed briefly.

1.2.1 Linear versus Nonlinear Dynamics

This section describes what differentiates nonlinear from linear dynamics. The following example illustrates how the two different system types are distinguished. Below is the state representation of a damped oscillator.

$$\dot{x_1} = x_2$$
$$\dot{x_2} = -\frac{b}{m}x_2 - \frac{k}{m}x_1$$

The above system example is said to be linear because all of the x_i states are to the first power only. The states x_1 and x_2 are not involved in any products, powers, or functions such as x_1x_2, x_1^2, or $cos(x_2)$. Any of the previously mentioned dynamics would cause the system to be labeled as nonlinear.

In the context of feedback control, one fundamental assumption considers that a real system can be described with linear and time-invariant differential equations. The mathematical concept of Laplace, which is heavily relied upon in control theory, cannot be used when dealing with nonlinear systems. This simple fact prohibits many linear control strategies from being implemented when considering nonlinear dynamics. Traditionally, when dealing with nonlinearities one would attempt to linearize the system about the operation point [31]. However, this causes two major issues. The first being that it limits the range of operation to the linearized region. If the system exits this region, the control scheme may break down and cause unstable, dangerous behavior. The second issue is linearization inherently reduces the accuracy of the system model. Even if the linearization is very robust, dynamics are lost in the process, which, if they had been maintained, could be

used to achieve superior control.

It may be questioned as to why linear models are used despite the fact that nonlinear models are more accurate. One of the primary reasons is that nonlinear models are difficult to handle analytically. The linear techniques such as normal modes, superposition and Fourier analysis fail when applied to nonlinear systems [26]. It is for this reason that so much research has been spent on linearization schemes. However, it will be shown that many nonlinear control techniques are very successful despite the difficulty nonlinear dynamics present.

Nonlinear dynamics are sometimes volatile and unstable which makes them hazardous. Improper control of these systems can damage the plant or harm operators. A robotic system may have a nonlinearity in its dynamics associated with its current output. If the current goes unstable, it could melt the circuitry and destroy the robot. Things like quad-copters and inverted pendulums easily go unstable during operation. A robust control scheme is needed to simply operate these types of plants.

1.2.2 Robotic Systems

Robotics is a relatively new branch of modern technology that blends several different engineering and science fields. One must have an understanding of mechanical engineering, electrical engineering, systems engineering, com-

puter science and mathematics (to name a few) in order to properly deal with all of the various aspects of a robotic system. In order to illustrate the field where this research will be applied, the current uses of various robotic systems will be discussed.

The word robot comes from the Czech word for work: *robota*. The term has since been used to describe a great many number of systems. Devices such as underwater vehicles, industrial manipulators, and assisted walking apparatuses have all been referred to as robots. With such a variety of systems under the hood of "robot", it becomes useful to talk in terms of system properties and functions rather than labels.

The industrial robotic manipulator has become widespread since the boom in the early 1980's. These systems consist of a computer controlled electromechanical arm. They typically perform tasks such as pick and place, welding, and spray painting [36]. Such duties required high precision with regards to path tracking and quick execution. In other words, the arm must move quickly to exactly to were it is told to be. This introduces the general concepts of trajectory tracking and system error. On the other hand, systems such as inverted pendulums and bipedal robots focus less on path tracking and more on disturbance rejection and robust stability. Tracking errors are acceptable as long as the robots remain upright. This shows the importance of system stability. It is the goal of this research to produce a control scheme

that can handle both demands simultaneously.

Other robots are also used in many industries today. Drones have become key pieces of equipment in fields such as military surveillance [5], videography, and even as a standalone sport [14]. Unmanned underwater vehicles have become more widespread in the fields of exploration, data collection, and military activities [38]. All of these presented robotic systems require high performance controllers in order to properly perform their diverse sets of tasks. Additionally, the rise in under-actuated systems creates the need for even higher controller performance. An under-actuated system reduces weight, cost, and energy consumption [46] which makes them an attractive design choice. However, it becomes difficult to properly stabilize the under-actuated states of the system. The newly designed controller is tailored to handle these types of systems.

This research focused on unstable under-actuated systems. An unstable system can be thought of as a system that does not naturally remain at or near its operating point. For example, an inverted pendulum will not stand up without the aid of an outside influence. Likewise, many nonlinear systems tend away from their desired operating points. The study of such systems is important because many more complex plants like assisted walking robots can be modeled using basic unstable plants. Therefore, control schemes that are successful with basic unstable models can be extended to be applied on

more complex and relevant robotic systems.

1.3 Control Schemes

The following sections will outline relevant control scheme developments to illustrate the current state of the field. Chapter 2 will provide better insight on the theoretical workings of the relevant controllers.

1.3.1 PID Control

The Proportional-Integral-Derivative (PID) is a relatively basic control scheme. This controller utilizes three different gains to process system error into a control action. Figure 1.1 shows the standard block diagram of a PID controller.

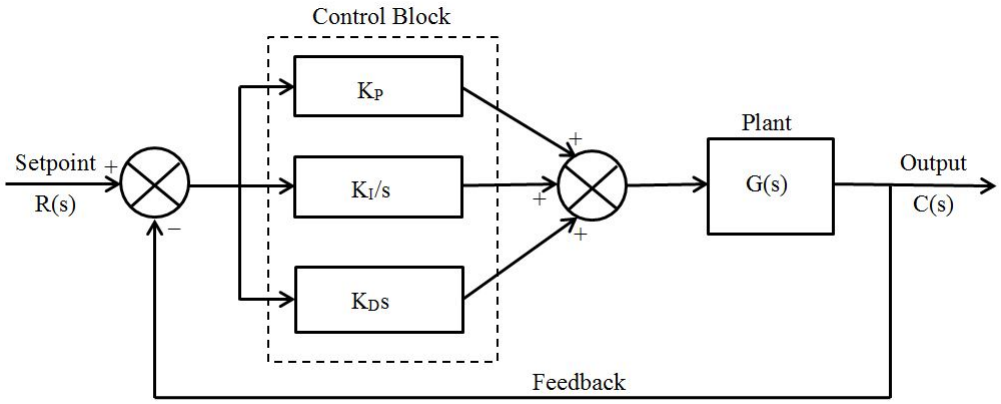

Figure 1.1: PID Controller Block Diagram

The three controller gains work to achieve the desired closed loop response.

The proportional term (K_P) scales the system error, The integral term (K_I) helps reduce steady state error, and the derivative term (K_D) improves the transient response of the closed loop system. Each of the controller gains are typically designed through closed loop specifications. [27]

While PID controllers are basic, they are very forgiving when it comes to control output. Even a simple PID will show some success when used on complex systems. However, when combined with more advanced strategies, the PID can be quite powerful. Wilson et al. implemented a cascade PID/Lead-Lag compensator with predictive feedback to control an inverted pendulum robot [16]. [8] also controlled an inverted pendulum robot but used a fuzzy PID scheme. Nuella et al. detailed the design of adaptive PID controllers for nonlinear systems [28]. Based on the PID's general applicability and effective system response management, a PID sliding surface was created for the new control scheme.

1.3.2 Sliding Mode Control

When modeling a system's dynamics it is very common to either neglect higher order dynamics (unstructured uncertainty) or make simplifications/approximations to lower order dynamics (structured uncertainties). Both lead to uncertainty in the obtained mathematical model which have strongly adverse effects on nonlinear systems. It therefore becomes necessary to explicitly deal with the model uncertainties directly in the controller design.

One of the most widely used techniques for uncertain nonlinear systems is the sliding mode control (SMC) method. A sliding based controller works on the principle that a first order system (nonlinear, uncertain, or otherwise) is much easier to control than systems of higher order. A intermediate sliding variable is introduced which replaces n^{th} order tracking problems with a first order stabilization problem. This system simplification allows for "perfect" performance in the presence of model uncertainty. However, this performance comes at the cost of extremely high control effort and oscillations in the controller output. This robustness has made sliding mode control a popular choice when dealing with a variety of systems.

Sliding mode controllers have applications in mechanical systems, robots and manipulators, and even space applications such as rocket control. Chen et al. designed a combined adaptive sliding mode controller to track the velocity profile of a wheeled mobile robot [3]. An adaptive component was added in order to compensate for system uncertainties and disturbances. Their work shows real results indicating the advantage of combined mixed sliding strategies. The team from [19] implemented a fractional order sliding controller to handle the deployment of tethered satellites with disturbances. Liu, Shan, and Li proposed a quasi-sliding mode control scheme combined with a neural network to land a rocket on an asteroid [24]. These publications show the advanced system applications in which sliding schemes are being currently used.

Yu et al. derived a terminal sliding surface controller for nonlinear systems [47]. This example shows how variations in the sliding surface can be beneficial to performance. Wang et al. showed that a terminal surface and adaptive techniques can be successfully combined and implemented [43]. Uncertain flexible structures were tackled by Nader Jalili and Nejat Olgac [17]. Finally, [42] employed an adaptive, fuzzy logic sliding mode controller for an inverted pendulum robot. The above demonstrate the versatility of the sliding mode control scheme and how it can be blended with a wide range of other strategies in order to improve results. They also show that sliding mode can be used in real world applications rather than just simulation.

However, many advanced sliding schemes come with the drawback of extremely complex computations and coordinate transformations [20]. Din et al. performed a comparative study of several sliding surface designs for under-actuated systems [7], It was found that each sliding surface had it's benefits but each one required the use of complex coordinate transforms in order to design the final controller. Olfati-Saber [29] dedicated a full paper on transforming under-actuated systems into cascade normal forms for control design. This was done through the use of geometric algebra, namely Lie brackets and Lie derivatives. [39] also tackled sliding mode control of under-actuated system using higher order integral sliding mode control. While the team was ultimately successful in controlling a real world ball and beam sys-

tem, the derived controller required 2 separate coordinate transforms and a highly complex control action derivation. The final control scheme also contained multiple constant for which the tuning process was not fully explained. The complex (coordinate transforms) and at times ambiguous (sliding surface tuning) nature of sliding mode control is one of the main drawbacks of the technique. This thesis seeks to both reduce control design complexity and simplify controller tuning. By achieving these two goals, the new sliding scheme can be more readily implemented in real world systems.

1.3.3 Genetic Algorithms

When considering optimization techniques, it is important to keep the specific problem in mind. In the case of this thesis' target systems, the optimization problem involves complex mathematical models and nonlinear terms. These factors make genetic algorithms a perfect fit for the task of tuning the sliding surface variables. This is due to a genetic algorithm's ability to perform a global search of the variable space despite the presence of nonlinearities or constraints. The added benefit of defining custom fitness functions also allows for tailored optimization results.

Genetic algorithms are a popular tuning option in the field of controls [13], [40], [6]. It is especially popular when dealing with artificial neural networks and fuzzy logic controllers [22], [4]. Farias et al. used genetic algorithms to tune a neural network and also a fuzzy logic controller in their 2018 paper [10].

They found that both the NN and the fuzzy controller could be successfully tuned using the genetic algorithm with the fuzzy logic controller requiring fewer generations to tune. This work shows that GA controller tuning can be successful with control schemes that contain many tuning variables. Hershey et al. employed a genetic algorithm to optimally path plan for autonomous vehicles [12]. The genetic algorithm was effective enough to potentially allow a single user to operate multiple autonomous vehicles at once while still achieving all mission targets.

In terms of genetic algorithms being paired with sliding mode controllers, there are a few notable references. Long et al. combined a sliding controller with GA optimization tuning and a cerebellar model articulation controller in order to control a lower limb exoskeleton [25]. It was found that the proposed control structure was more suitable to follow human motion intent under the occurrence of uncertainties in simulation. Dereje et al. applied a genetically tuned integral slinding mode controller to a Stewart Platform manipulator [32]. The goal was to reduce the design difficulty of integral sliding mode control through the use of genetic tuning for the trajectory tracking problem. It was found that the proposed control structure produced superior results compared to other tested controllers. These works show that genetic tuning is an effective technique to tune sliding mode controllers and can help simplify the design procedure.

1.4 Objectives

The objectives of this thesis can be broken into primary and secondary objectives. For the work to be considered a success, the newly developed control scheme should simultaneously meet all of the presented objectives. The following list states each objective in more detail.

- The primary objective of this work is to design a sliding controller that is capable of controlling under-actuated and unstable mechanical systems. The work should detail each step in the controller design process. It should also include the theoretical workings required to prove overall system stability and robustness.

- The secondary objective of this thesis is to show with simulation and experimental work that the new control scheme demonstrates improved performance compared to standard sliding and linear control techniques. Controller performance will be judged based on factors such as settling time, steady state error, and computational complexity/run time.

- The final objective of the work is to show the viability of the new control scheme for real world systems. The new sliding controller should be readily applicable to real systems and not require excessive computation. The sliding controller should also avoid the classic control signal chatter phenomenon commonly associated with sliding mode con-

trollers.

1.5 Contribution

The contributions of this thesis are as follows:

- The first contribution of this thesis is the full design process breakdown of the new sliding controller. This includes the sliding surface design, parameter tuning, control signal derivation and stability analysis.

- The second contribution is the description of the custom genetic algorithm used to tune the sliding surface parameters. This includes information about fitness function selection, variable bounds, and reproduction method. Interpreting the resulting algorithm output is also covered.

- The final contributions are the physical implementations themselves. They demonstrate that this control scheme can be used in the real world and not just in simulation.

Chapter 2

Control Theory

This chapter focuses on the theoretical aspects of various control schemes and tools rather than relevant applications. The intent of each section is to give the reader a high level overview of the topic so as to provide insight into the following research. Sliding mode control will be the focus of the chapter but genetic algorithms will also be discussed. Chapter 3 covers the actual design of the genetically tuned PID sliding surface control scheme.

For additional information, the following reference materials are suggested:

- Sliding Mode Control: [35], [41], [46]
- Genetic Algorithms: [4], [32]

2.1 State Space Modeling

As previously discussed in section 1.2.1, Laplace formulations cannot be used on systems with time variance or nonlinear dynamics. It therefore becomes necessary to analyze these systems using a different approach. State space modeling, like transforms, is a tool used to analyze and design feedback control schemes [27]. The advantage of state space methods is that they can be applied to systems regardless of time variance or nonlinearities (hard or soft). The general form of a state space model can be seen below in equation (2.1).

$$\begin{aligned} \dot{\mathbf{x}} &= \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{u} \\ \mathbf{y} &= \mathbf{C}\mathbf{x} + \mathbf{D}\mathbf{u} \end{aligned} \tag{2.1}$$

The matrix $\mathbf{A}$ is the system matrix which describes the dynamics of the system. The matrices $\mathbf{B}$, $\mathbf{C}$, and $\mathbf{D}$ are the input, output, and feedforward matrices respectively. The vector $\mathbf{x}$ is the state vector and $\dot{\mathbf{x}}$ its derivative; the state vector contains all of the state variables of the system. The vectors $\mathbf{u}$ and $\mathbf{y}$ are the input/control vector and the output vector, respectively [27]. A second general form for state space modeling exists which is more useful when working with nonlinear systems. Companion form or controllability canonical form is shown below in equation (2.2).

$$\mathbf{x}^{(\mathbf{n})} = f(\mathbf{x}) + b(\mathbf{x})\mathbf{u} \tag{2.2}$$

In this form f($\mathbf{x}$) and b($\mathbf{x}$) are nonlinear functions of the states. The vector $\mathbf{x}$ is defined as $\mathbf{x} = [x_1, \dot{x}, ..., x^{(n-1)}]^T$ and u is the scalar input. This form of state space modeling is important as there is no derivative of the input in the formulation. This form is required by many nonlinear control schemes which brought about the need for transformation techniques.

By implementing state space representation, a system of equations with a combined differential order of n can be written as a set of n first order differential equations. This is a valuable property as it greatly simplifies analysis of the system and eases simulation. The Euler-Lagrange method was the primary technique used in creating the state space mathematical models used in this thesis. Due to its ability to handle and simplify more complex systems, all of the following control schemes are described in the state space regime.

2.2 Sliding Mode Control

The backbone of this research is sliding mode control (SMC) theory. Sliding mode is a nonlinear control technique that excels in system stability and trajectory tracking. The main principle of the controller is that it is much easier to control a first order system, be it nonlinear or otherwise, than any system of higher order. Through the use of an intermediate sliding variable, the target system is compressed into a first order sliding manifold. This changes

a multi-order trajectory tracking problem into a series of first order stability problems. The sliding variable is designed in such a way that when it equals zero, system stability is guaranteed. This is the main strength of sliding mode theory, especially when working with unstable nonlinear systems.

However, the natural robustness of the controller comes with drawbacks. In order to assure stability, large magnitude control actions are required. Additionally, these control actions contain high frequency oscillations across the sliding surface. If mishandled, these oscillations or chatter can damage the system by exciting higher order vibration modes. It therefore becomes necessary to work around the control magnitude and chatter in order to properly implement a sliding mode controller. The following subsections will detail the various components of a sliding mode controller and how they are designed.

2.2.1 Sliding Surface

The sliding surface or manifold is arguably the most important piece of a sliding mode controller. The surface must be designed so as to guarantee system convergence to either the desired trajectory or a stable equilibrium state. Generally this means that the surface is defined as a linear combination of system error states. The goal of the controller is to then drive the system to a zero sliding surface value. This causes the state error to then exponentially decay. Figure 2.1 shows a basic sliding surface represented in

state space.

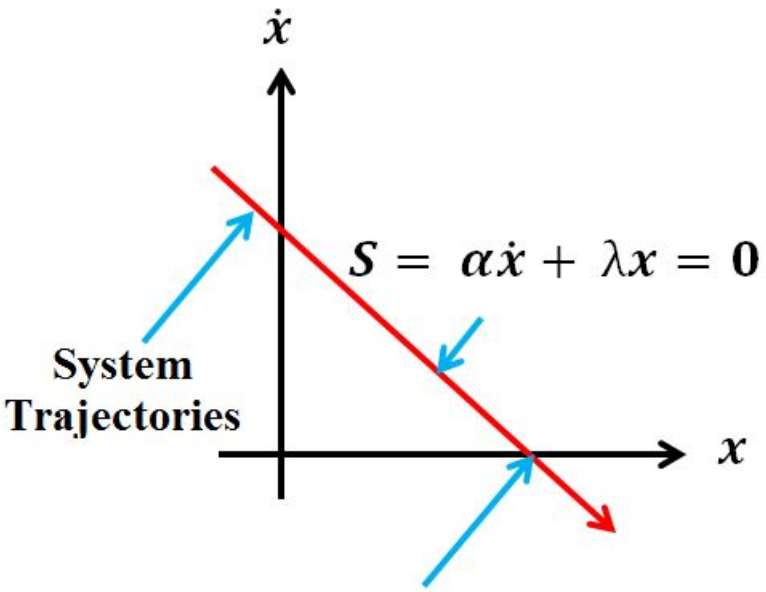

Figure 2.1: Diagram of a Sliding Surface Illustrated in State Space

There are many flavors of sliding surface definition that can be used. First order surfaces are generally linear combinations of the system's position and velocity error states. This allows the controller to directly affect the system. However, this comes at the cost of direct control chatter. A second order sliding surface operates on higher order system states rather than the position and velocity. This allows the controller to suppress the control chatter as it operates in the higher order states but comes at the cost of reduced precision and convergence speed. Integral sliding mode control adds an integrating component into the sliding surface to assist in convergence and

chatter suppression. It should be noted that the more complex surfaces require computation intensive design and validation. This heavy reliance on theory can be viewed as a drawback as it makes the surfaces difficult to work with for widespread applications.

Regardless of the surface definition, each one contains surface constants. These constants are the coefficients of the state error values and only follow loose rules for design. Often times inequalities are derived to give limits to the constants but not concrete values. In many cases, they are arbitrarily set or hand tuned through several iterations. This is both suboptimal and tedious. This research provides an optimal tuning process that eliminates the need for repetitive testing through the use of a custom genetic algorithm.

2.2.2 Control Signal

The control signal is the final product of any controller. It is the signal that is sent to the plant in question in order to achieve the desired performance. Many controller use the system state error values to derive a control signal while others use an ideal model performance. A sliding mode controller, however, generates its control actions through the use of the sliding manifold. The control action is designed in order to maintain the sliding manifold at a value of zero. This then guarantees that the system will exponentially tend towards the desired trajectory.

The first step is to take the derivative of the sliding surface definition. By setting this new expression equal to zero, an equation for the control signal can be found. This, however, is known as an ideal control law. It is the control signal that would work if the system was perfectly known, unchanging, and without disturbances. In other words, this control law alone will not work in real applications. To fix this, a second piece is added called the switching control law. This piece continually pushes the system towards the zero surface state and is the key to sliding mode control. This switching control law comes with the drawback of high frequency chatter in the control signal so further design steps must be taken to counteract its effects.

2.2.3 Control Gain

The control signal of a sliding mode scheme can be broken into two pieces; one piece counter acts the system dynamics while the other forces the system to toward the zero surface. The counter dynamics component is derived using the system mathematical model and desired trajectories. The second component, known as the switching term, is made up of a discontinuous switching function with a controller gain coefficient. This controller gain isn't derived from the dynamics but rather from a stability proof.

Using a Lyapunov-like stability analysis, a control gain can be found that guarantees the sliding surface is attractive and all system trajectories converge towards it. This guarantee of convergence combined with the definition

of the sliding surface mean all system trajectories tend toward the sliding surface and all errors exponentially decay to zero. By adding in considerations for system model uncertainty and outside disturbances, a well designed sliding scheme can reject the previous obstacles and still maintain tracking. Chapter 3 will go into further detail regarding the Lyapunov-like stability analysis.

2.2.4 Boundary Layer

One of the most effective and simplest methods of reducing control signal chatter is to implement a boundary layer around the sliding surface. Figure 2.2 shows a diagram of a sliding surface with a boundary layer around it. Normally the controller operates using a zero value for the sliding surface. Any system state above or below the zero surface receives the full force of the controller in an attempt to push the system back to zero. A boundary layer allows the controller to widen the acceptable margin of the sliding surface. In doing so, the controller no longer forces the system to zero, but to an area around the zero surface. This reduces chatter about zero but also reduces the tracking precision of the controller. Because the system is no longer held at zero, small errors become acceptable. Therefore, a well designed boundary layer effectively reduces chatter while still maintaining high tracking performance.

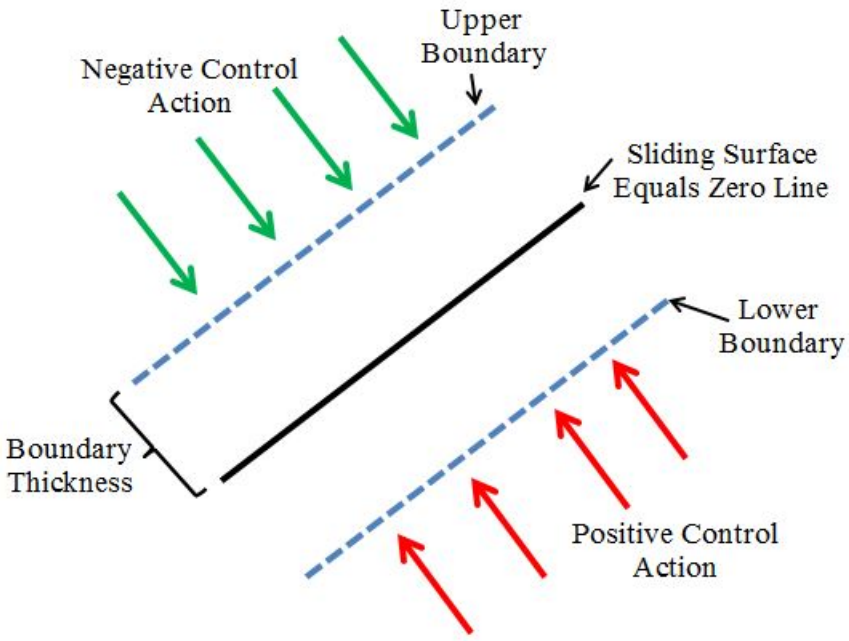

Figure 2.2: Diagram of a Sliding Surface with a Boundary Layer

2.3 Genetic Algorithm

The optimization piece of this thesis comes from the use of a genetic algorithm to tune the controller parameters. A genetic algorithm is an optimization strategy that is based on the principles of natural selection and survival of the fittest. The heuristic strategy uses a population of individual with "genes" that represent the variables to be optimized. These individuals are then evaluated based on a predefined fitness function to asses their strength. The best individuals are then selected to reproduce which generates another generation of individuals. The process is then repeated until an end condi-

tion is met. The biological inspiration for this algorithm can be easily seen.

This type of optimization can be thought of a structured trial and error type of search. The following subsections will go into further detail about the important pieces of a genetic algorithm with focus on how the algorithm used in this thesis works. Figure 2.3 shows a flowchart of a genetic algorithm.

2.3.1 Population and Generations

The first step for a genetic algorithm is to define the initial population size and the number of generations. The population size refers to the number of possible solutions that are evaluated every generation. These solutions are referred to as individuals in keeping with the biological theme. Each individual is made up of genes. The genes are coded to contain the individual's value for each optimization variable. For a one value optimization, a gene contains just that one value. More optimization variables means larger genes.

It is important to design the population size to be large enough to accurately obtain the optimal value but not so large so as to waste computation time. As the number of variables to optimize increases so too should the population number. For the purposes of this thesis, populations of 20, 50, and 100 were used as small, medium and large population sizes respectively.

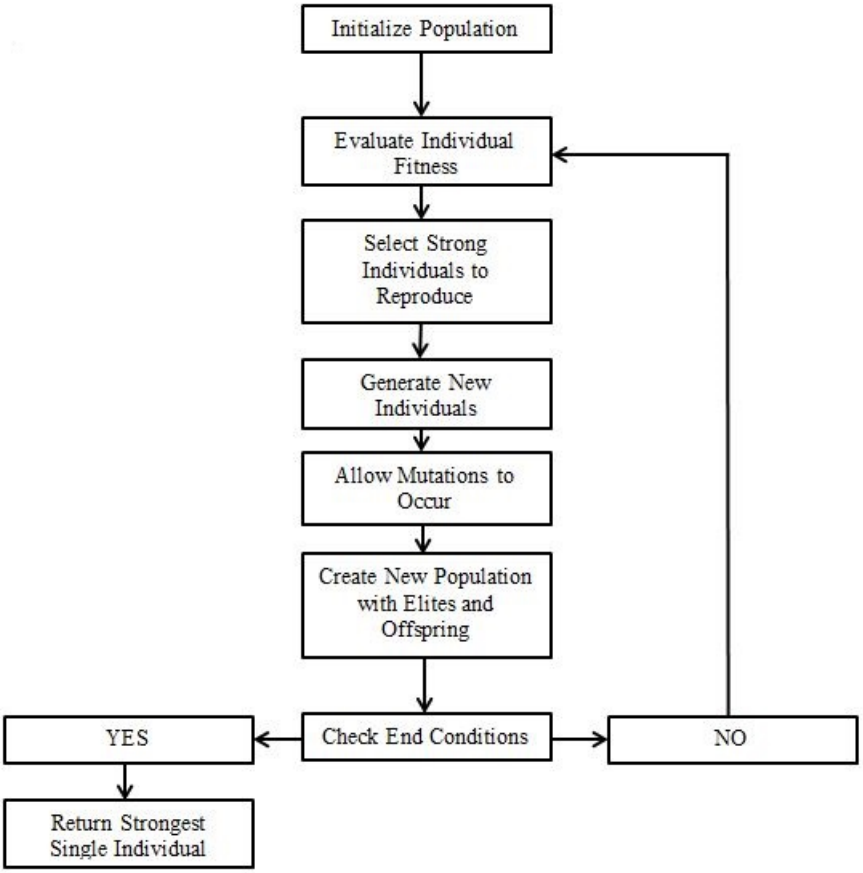

Figure 2.3: Flowchart of a Genetic Algorithm

The second part of initialization is defining the number of evolution generations. This parameters limits how many times the population reproduces and is evaluated which means it is also an end condition. Allowing the population to evolve over more generations means that a greater number of individuals will tend towards the optimal solution. A lower generation number has a faster solution time but may not fully reach the best solution. This thesis

used generation sizes of 10, 30, and 50. When designing a genetic algorithm it is important to properly balance the complexity of the problem with the population and generation sizes. A good strategy is to run smaller size trials to ensure proper function and then run a larger final algorithm to obtain the best solution.

2.3.2 Fitness Evaluation

Arguably the most important part of a genetic algorithm is the fitness function. This is a mathematic formula that outputs a fitness score for each individual based on their gene values. The fitness score can either be a minimum or maximum based on the definition. Individuals that have a better fitness value will be more likely to reproduce and pass on their superior genes. Therefore, it is important to ensure only the best individuals can obtain good fitness scores. Additionally, as each individual is evaluated every generation, the fitness function should not be overly complex to the point of slowing down computation time. In this thesis, the fitness function is a novel blend of state tracking mean squared error values and a controller sign switching counter. This way, fitter individuals will have lower error values and less control chatter.

2.3.3 Selection and Reproduction

Before reproduction happens, the parents must be determined based on the current population. There are a few methods for selecting parents but the most useful and common are line method, roulette wheel, and tournament. The line method lays out a line composed of each individual. The length associated to each individual is directly related to their relative fitness value. A better fitness means a longer line segment. The algorithm then takes uniform steps along this line. Each step selects a parent at that point in the line. Individuals with better fitness will be selected more often meaning stronger gene propagation.

Roulette wheel is similar to the line method but uses a weighted wheel instead of a line. The wheel is separated into sections corresponding to each individual. Stronger individuals get more space on the wheel increasing their chance of being selected. This method is regarded as superior to the line method as it does not limit the number of times a very strong individual can be selected. Tournament selection involves creating small groups of individuals. The fittest member of each group is selected to reproduce. The size of the tournament must be predefined and cannot be less than two individuals.

The actual reproduction mechanism is called crossover. After the all of the parents are selected, their genes are crossed in order to create a new individual. The genes are a series of binary values with each segment representing

a different optimization variable. Optimizing for four variables will create genes that contain four segments of binary joined in series. Reproduction then switches pieces of the binary series between the two parents. If a binary value is 4 bits long (values from 0 to 15), the crossover may take place in the middle. This would switch the third and fourth bits between the two parents creating new unique individuals. Figure 2.4 shows a simple diagram of crossover taking place. This process allows new combinations to be created while still propagating strong genes onwards. The location and number of crossover points can be specifically set of left to be the standard value.

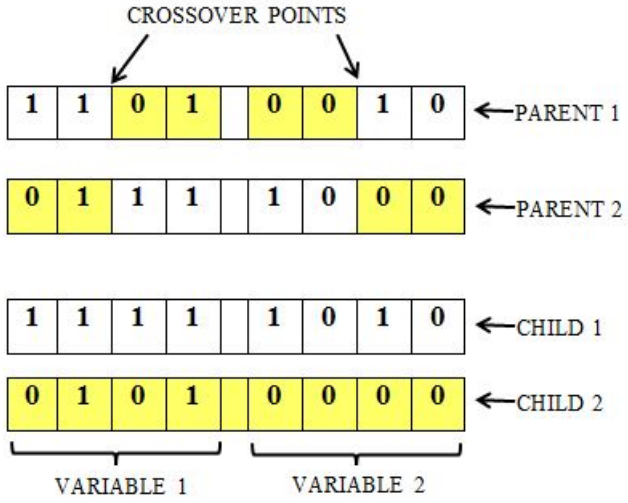

Figure 2.4: Diagram of how Crossover takes place

While crossover is an effective reproduction method, in nature there is another factor at play that affects an offspring's genes. Mutation is a random change in a individuals genes that may improve or hinder that individual's

fitness. Figure 2.5 shows a simple diagram of mutation occurring. By specifying the mutation rate, typically very small, one can ensure that the genetic algorithm explores a sufficient variable space while searching for the optimal combination. A proper mutation rate prevents premature population convergence but a mutation rate that is too high will hinder the algorithm in converging to the best value.

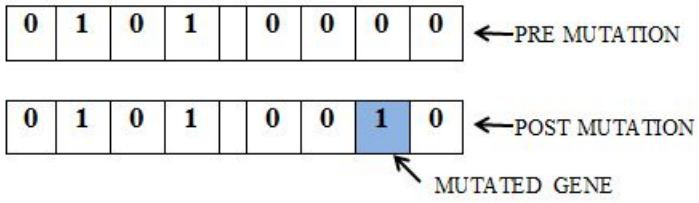

Figure 2.5: Diagram of an Individual Mutating

With the new individuals generated and mutated, a new population is formed consisting of the strongest of the previous generation and the new offspring. The number of individuals from the previous generation that are carried over to the new population is called the elite count. Typically this number is kept to ten percent of the total population size. This ensures that the individuals with the best fitness are not lost but that there is still room for possible improvements.

2.3.4 End Conditions

As with most optimization techniques, genetic algorithms have end conditions that stop the search process once met. There are two main end conditions for genetic algorithms: maximum generation number and mean fitness score. It should be noted that both end conditions can be used simultaneously during the optimization. This prevents one from over-relying on a single stopping criteria. The generation number is as simple as setting how many times the algorithm can create a new population. This sets a hard limit to the number of runs the algorithm performs. If this end condition is hit, the genetic algorithm returns the most fit individual from the last generation as the optimal solution. This was the main stopping criteria triggered in this thesis.

The second end condition for a genetic algorithm is the different between the best fitness score and the mean fitness score; called the function tolerance. The best fitness score of a generation is the individual that gives the most optimal solution to the problem. The mean fitness score is the average fitness of the all the individuals in the current population. When these two values get within a certain range of each other, the algorithm can be considered to be converged to the best solution in the search space. A lower function tolerance means a more tightly converged population set while a higher function tolerance reaches a solution faster but is less precise. Knowledge of the fitness function should be used when trying to determine an appropriate tolerance.

Chapter 3

Controller Validation

The following chapter will detail all of the necessary steps needed to design the new sliding mode controller. Because sliding mode controllers are nonlinear control schemes, they are quite computation intensive and require accurate system math models to function properly. Each of the following sections will go through a separate step in the design process in order to make it as understandable as possible. Combined with chapter 2, the theoretical base, and the chapter 4, the implementations, this chapter should give the reader a solid comprehension of the novel control scheme.

3.1 Dynamic Model Manipulations

The first step for designing a sliding mode controller, specifically the genetically tuned PID sliding surface controller, is to obtain an accurate math model for the target system. This is a very important step because much of the theoretical backing for a sliding controller relies on an accurate dynamic model. If the model is not well derived, the convergence and stability properties of the controller will not hold causing sub-par performance. The math models used in this thesis were derived using the Euler-Lagrange method. Equation (3.1) shows the desired general form of a dynamic model that sliding schemes like to work with.

$$M(q)\ddot{q} = f(q, \dot{q}) + u + d \tag{3.1}$$

This general form is preferred as it separates the components of the dynamic model in the correct way. The mass matrix $M(q)$ contains parameters like system masses and inertial values. Vectors $f(q, \dot{q})$, u, and d are the Coriolis, control, and disturbance vectors, respectively. The disturbance term can be lumped into the f vector in order to reduce the number of variables without loss of robustness. This general form can be further broken down by defining the individual components of each matrix and vector term. Equation (3.2) shows how each term is labeled.

$$\begin{bmatrix} M_{aa} & M_{au} \\ M_{au}^T & M_{uu} \end{bmatrix} \begin{bmatrix} \ddot{q}_a \\ \ddot{q}_u \end{bmatrix} = \begin{bmatrix} f_a + u \\ f_u \end{bmatrix} \tag{3.2}$$

where $M_{aa}(m \times m)$, $M_{uu}(r \times r)$, and $M_{au}(m \times r)$

This thesis deals with unstable, under-actuated systems so it is useful to divide the dynamic model into actuated and unactuated sections. For a system with n degrees of freedom, m actuated state(s), and r= n-m unactuated state(s), a and u denote actuated and unactuated variables respectively. Solving the above for the system accelerations yields the following. (Note that the accent ($^-$) denotes an intermediate variable used to reduce the number of terms in the equations).

$$\begin{bmatrix} \ddot{q}_a \\ \ddot{q}_u \end{bmatrix} = \begin{bmatrix} \overline{M_{aa}^{-1}}(\overline{f_a} + u) \\ \overline{M_{uu}}(\overline{f_u} - M_{au}^T M_{aa}^{-1} u) \end{bmatrix} \tag{3.3}$$

where:

$$\overline{M_{aa}} = M_{aa} - M_{au} M_{uu}^{-1} M_{au}^T$$

$$\overline{f_a} = f_a - M_{au} M_{uu}^{-1} f_u$$

$$\overline{M_{uu}} = M_{uu} - M_{au}^T M_{aa}^{-1} M_{au}$$

$$\overline{f_u} = f_u - M_{au}^T M_{aa}^{-1} f_u$$

Using equation (3.1), the dynamic model can be placed in the proper form.

The model can then be labeled according to (3.2) which distinguishes actuated versus unactuated. Using the new intermediate model variables, the dynamic model can be rearranged to solve for the accelerations using (3.3). The result can be used to begin the next design steps.

3.2 Surface Selection

The definition of the sliding surface is the keystone to a sliding mode controller. It is what ensures setpoint tracking and forces system state error decay. For a standard sliding controller, the surface is made up of a linear combination of the position and velocity state errors. Equation (3.4) shows this type of surface's formal definition. The terms λ and $\tilde{x}$ are a surface constant and a state error value respectively ($\tilde{x} = x - x_d$).

$$s = (\frac{d}{dt} + \lambda)^{n-1}\tilde{x} \tag{3.4}$$

For a second order system, equation(3.4) would result in the following surface definition:

$$s = \dot{\tilde{x}} + \lambda\tilde{x} \tag{3.5}$$

However, because this thesis works with under-actuated systems, a special sliding surface must be used. In order to account for both actuated and unactuated states with only one control action all of the errors must be

combined into a single surface. Equation (3.6) shows a sliding surface for an under-actuated system.

$$s = \alpha_a \dot{\tilde{q}_a} + \lambda_a \tilde{q_a} + \alpha_u \dot{\tilde{q}_u} + \lambda_u \tilde{q_u} \tag{3.6}$$

The sliding surface used in this thesis takes the above definition one step further by adding in an integrating piece. The integral term continuously sums the position error for the system states inside the sliding surface. This help the controller minimize steady state error which helps system stability. The integrator also aids in disturbance rejection due to the error being summed. The same strengths that have made standard PID controllers so prevalent is now compressed into the sliding surface of the new controller. Equation (3.7) shows the surface used in the newly designed controller. (The added integrating terms should not be confused for integral sliding mode control which has a different structure.)

$$s = \alpha_a \dot{\tilde{q}_a} + \alpha_u \dot{\tilde{q}_u} + \lambda_a \tilde{q_a} + \lambda_u \tilde{q_u} + \beta_a \int \tilde{q_a} + \beta_u \int \tilde{q_u} \tag{3.7}$$

While the new integration term in the surface have benefits, they also add two additional surface constants that must be tuned in order to obtain good control performance. This issue is covered by implementing a custom genetic algorithm which solves for the optimal surface constants. This will be discussed in the Genetic Parameter Tuning section of this chapter.

The final step of selecting a sliding surface is to ensure that two conditions are met. The first is to show that if the sliding surface equals zero, the state error will converge to zero. The second is to make sure that the control action appears in the desired derivative of the sliding surface. In this case the first derivative of the surface should contain the control action u. For higher order sliding mode control, the control action appears in higher derivatives of the surface.

From basic inspection it can be seen that both conditions hold true for the surface used in the new controller. When the surface shown in equation (3.7) equals zero, the state errors form a stable system that naturally decays to zero. Additionally, because the surface contains velocity error terms, the control action shows up in the first derivative of the surface. Differentiating the surface yields equation (3.8) which clearly contains the control action u.

$$\dot{s} = M_s u + f_s + \dot{s}_r \tag{3.8}$$

where:

$$\begin{aligned}
M_s &= \alpha_a \overline{M_{aa}^{-1}} - \alpha_u \overline{M_{uu}^{-1}} M_{au}^T M_{aa}^{-1} \\
f_s &= \alpha_a \overline{M_{aa}^{-1} f_a} - \alpha_u \overline{M_{uu}^{-1} f_u} \\
\dot{s}_r &= -\alpha_a \ddot{q}_a^d - \alpha_u \ddot{q}_u^d + \lambda_a \dot{\tilde{q}}_a + \lambda_u \dot{\tilde{q}}_u + \beta_a \tilde{q}_a + \beta_u \tilde{q}_u
\end{aligned}$$

Therefore, should the system reach a zero sliding surface value, the system

errors will decay to zero. Additionally, the appearance of the control action in the first derivative of the sliding surface definition mean the control action can be properly formulated. This completes the validation of the implemented PID sliding surface.

3.3 Control Action Derivation

With the sliding surface selected, the control action can now be designed. The purpose of the control action is to drive the sliding surface to a value of zero. Based on the subsection above, this will force the system to have zero trajectory tracking error. Once the surface is at zero it should remain at zero. Therefore, by setting the derivative of the sliding surface to zero, the control action can be derived. Using equation (3.8), u can easily be isolated to give the following expression.

$$u = -\hat{M}_s^{-1}[\hat{f}_s + \dot{s_r} + Ksat(s/\epsilon)] \tag{3.9}$$

Here the ($\hat{}$) accents indicate estimations. This is the convention due to the dynamic model of the system not being completely known. The term K is a controller gain constant and will be derived in the next section. Note that the saturation function is used instead of the sign function. This implements a boundary layer around the sliding surface of thickness ϵ.

3.4 Controller Gain Derivation & Stability Analysis

At this point, there is a sliding surface which, if forced to zero, will cause the system state errors to exponentially tend towards zero and a control law that keeps the surface at zero. The last piece of the controller design is to define the switching control gain in such a way that the system is guaranteed to be stable and the surface is attractive to all system trajectories. This condition can be met by performing a Lyapunov-like stability analysis [35].

First, we define the Lyapunov like candidate function. Equation (3.10) shows the standard function for sliding mode controllers which is a composed of the sliding surface.

$$V = \frac{1}{2}s^2 \tag{3.10}$$

The s^2 term can be thought of as the squared distance to the zero surface value. By maintaining the change of this value below a strictly negative value, the distance to the zero surface value must decrease along all system trajectories. Additionally, once on the zero surface, all trajectories remain there which completes the final control condition. Satisfying equation (3.10) makes the sliding surface an invariant set. In order to meet this condition, the next step of the Lyapunov-like analysis is performed.

$$\dot{V} = \dot{s}s \leq -\eta|s| \tag{3.11}$$

Where η is a strictly positive constant

Substituting equations (3.8) and (3.9) into the above equation yields the following:

$$K = F_s + \eta \tag{3.12}$$

where $|f_s - \hat{f}_s| \leq F_s$

Thus the control gain K is defined in such a way that validates condition (3.10) and guarantees sliding surface convergence. This completes the controller design and performance validation. The controller tuning parameters $(\alpha_a, \alpha_u, \lambda_a, \lambda_u, \beta_a, \beta_u, \eta, \epsilon)$ should be selected such that the system remains stable while reaching the sliding surface with minimal control chattering. The next section will detail how a genetic algorithm is used to tune all of the sliding surface variables.

3.5 Genetic Parameter Tuning

The purpose of the genetic algorithm in this thesis is to optimally tune the sliding surface and reaching constants. This is done through the use of a

custom and novel fitness function. By setting the fitness function to be the mean squared error of the sliding surface states, the algorithm can select individuals whose surface values give the lowest state error. Essentially the algorithm will eventually return the set of surface constants that produce the lowest tracking error possible for the given simulation conditions. Figure 3.1 shows a plot of a system error response.

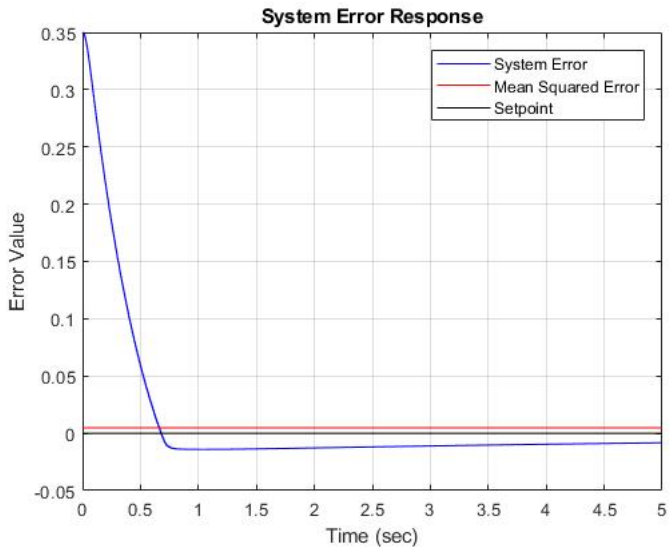

Figure 3.1: System Error Response with Setpoint and Mean Squared Error shown

The above response has a small mean squared error value. This would result in the implemented surface constants receiving a good fitness function score. Figure 3.2 shows a plot of the evolution of the population during optimization. The top section shows how the population's best and mean fitness score converge to a minimum. The bottom section shows the lineage of each

individual in a generation. The blue, red, and black lines represent standard reproduction, mutation, and elite individuals.

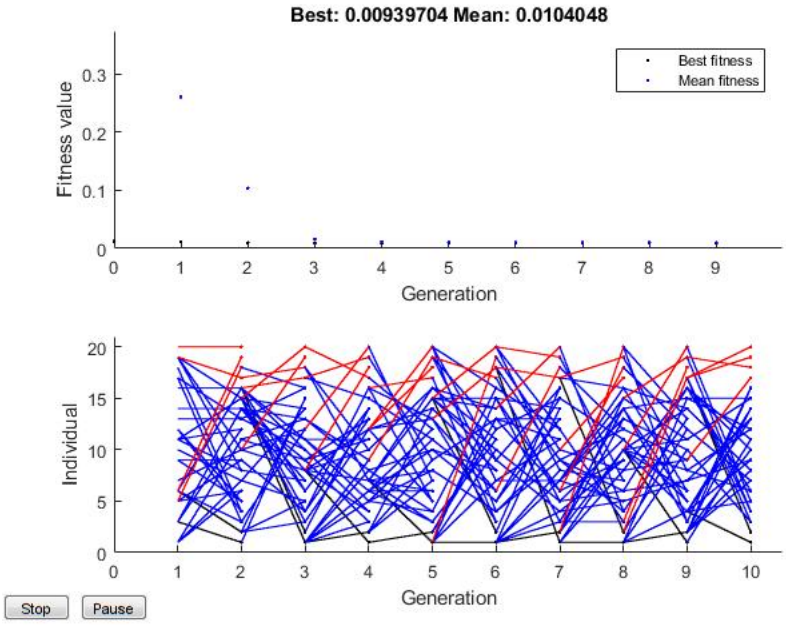

Figure 3.2: Plot of Fitness Scores and Individual Lineage

In addition to the mean square tracking error in the fitness function, there is another term that assists in solving for the best possible surface constants. A controller switching counter was added to the fitness function. This switching counter tracks how many times the sign of the control action changes for the duration of the simulation. Two sets of surface constants may give the same mean squared error tracking value but one may have excessive control chatter. By incorporating the sign change into the fitness function, response like Figure 3.3 can be eliminated.

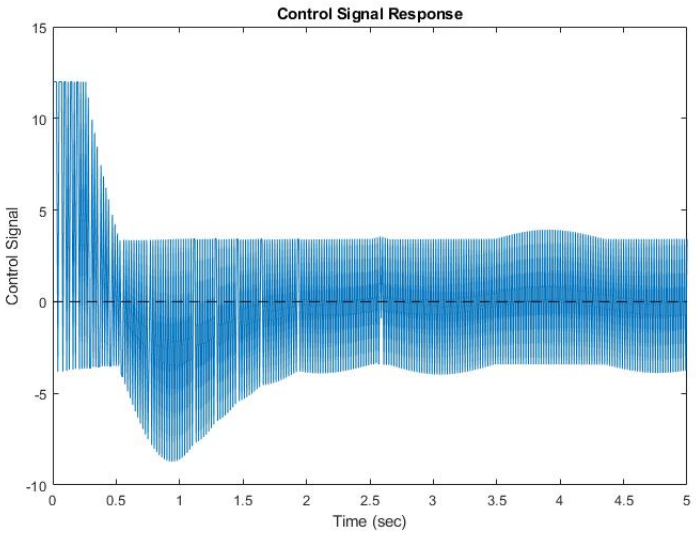

Figure 3.3: Plot of a High Chatter Control Signal

If a set of constants gives a high chatter control signal, then a large deduction is added to their fitness score. This effectively prevents them from reproducing. This also ensures that high fitness candidates have a control switching number that is lower than the designated value. This mix of tracking error and controller switching ensures high performance tracking while also actively reducing control chatter. Figure 3.4 shows how the fitness function is created within the MATLAB code.

The simulations return the value of E to the genetic algorithm as the fitness score. The lower the value of E, the better the individual fitness. The above example focuses only on the theta error with the control signal chatter

```
E1= mean((x-xd).^2);
E2= mean((dx-dxd).^2);
E3= mean((ddx-ddxd).^2);

E4= mean((theta-thetad).^2);
E5= mean((dtheta-dthetad).^2);
E6= mean((ddtheta-ddthetad).^2);

if u_sign <= 50
    u_sign_fitness=0;
else
    u_sign_fitness= 50;
end

E= [E4+u_sign_fitness];
```

Figure 3.4: Snapshot of Fitness Function MATLAB Code

counter.

With the entire control design process defined, the next chapter discusses the case studies performed using the new control scheme.

Chapter 4

Case Studies

The following sections detail all of the work that was done for the respective mechanical systems. The sections are broken up by system and then further into simulation and practical applications when necessary. The dynamic model for each system will be described prior to the control results. The control law design will not be described for each system as it is largely the same process and was detailed in Chapter 3. The dynamic models for each system were found using Euler-Lagrange formulations while the model parameters were calculated experimentally with the help of modeling software. The case studies are as follows: an Inverted Pendulum-Cart, a Double Inverted Pendulum (Pendubot), and a Two Wheeled Robot.

4.1 Inverted Pendulum

The first case study focused on an inverted pendulum system. The test plant is comprised of a pendulum rod mounted to a cart. The cart rests on a rail and is actuated through the use of a belt and DC motor. The goal of the plant is to stabilize the pendulum in an upright position by moving the cart back and forth. Additionally, the cart can be forced to follow other trajectories such as sinusoids to further test a controllers performance. Disturbances can also be tested through the use of manual perturbations. This system is a perfect first case study as it is not an overly complex or aggressive system. The following subsections will outline the simulation and practical tests that were performed.

4.1.1 Dynamic Model

The dynamic model of the inverted pendulum-cart system was developed using Euler-Lagrange formulations. The model is presented using state space representation and is broken into the standard mechanical system model format. Figure 4.1 shows the free body diagram of the inverted pendulum test plant.

The cart and rod states are represented by x and θ respectively. Note that the rod angle originates from the vertical axis. The control action u is a force which is converted to a voltage before being sent to the motor. Table 4.1

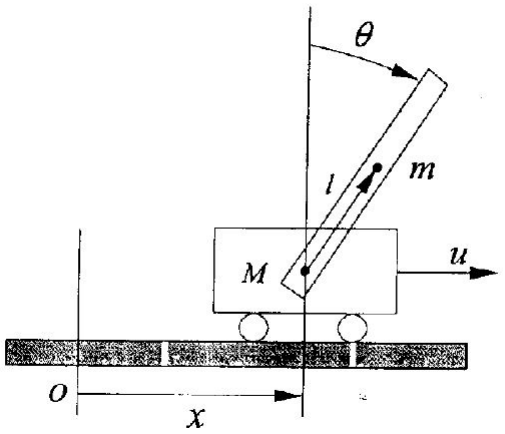

Figure 4.1: Inverted Pendulum Free Body Diagram [41]

details the parameters found in the mathematical model.

Table 4.1: Inverted Pendulum Model Parameters

Parameter	Symbol	Value	Unit
Cart Mass	M	2.4650	kg
Rod Mass	m	0.113	kg
Rod Length	L	0.15	m
Rod Inertia	J	$9.89e^{-7}$	kgm
Gravity Constant	g	9.81	$kg\frac{m}{s^2}$
Coefficient 1 (Voltage Conversion)	C_1	1.2778	$\frac{N}{V}$
Coefficient 2 (Friction)	C_2	8.5	$\text{kg}\frac{m}{s}$

Equation (4.1) shows the final dynamic model of the inverted pendulum in matrix form. It can be seen that the model is symmetrical with regards to the mass matrix which is solely a function of the unactuated rod angle θ. Additionally, the control input Is a scalar value. These two properties are

markers of an under-actuated mechanical system.

$$\begin{bmatrix} M+m & mL\cos(\theta) \\ mL\cos(\theta) & J+mL^2 \end{bmatrix} \begin{bmatrix} \ddot{x} \\ \ddot{\theta} \end{bmatrix} = \begin{bmatrix} mL\dot{\theta}^2\sin(\theta) - C_2\dot{x} \\ mgL\sin(\theta) \end{bmatrix} + \begin{bmatrix} C_1 u_0 \\ 0 \end{bmatrix} \quad (4.1)$$

This is the model that will be used in the following subsections. As sliding mode controllers utilize the system model within their control action formulation, it is important that the model has high fidelity to the actual plant.

4.1.2 Simulation

This subsection focuses on all simulations performed using the inverted pendulum plant. These simulations tackled two main tasks. First, they validated the sliding mode control scheme when dealing with various trajectories and disturbances. Secondly, they were used to implement and validate the novel genetic tuning algorithm. The first set of simulations will show the effectiveness of a iteratively tuned standard sliding mode control scheme. The controller was derived using the methods outlined in Chapter 3 of this document. Steps were taken to maximize the simulations' real world fidelity. The control action was limited to the actual voltage limitations and surface constant were kept to reasonable values to as to not excite higher order vibration dynamics.

Please refer to the end of this section for performance tables. The solutions of the genetic algorithm as well as the script run time will be shown in the first table. The second table will have system response parameters such as settling time and peak displacements.

Figure 4.2 shows the controller stabilizing the system from a non zero initial condition. This is the most basic stability situation as there is no disturbances or trajectory to track. The rod was given a positive initial condition of 20 degrees which is not a trivial magnitude. Many linearizing control techniques would break down at this size of initial condition due to their model linearization no longer being valid.

It can be seen that the iteratively tuned PD sliding mode controller successfully stabilizes the pendulum rod. In under one second the rod is brought back to a near zero position without a huge control action or cart displacement. The rod is not brought back to exactly zero due to the controller trying to bring the cart back to the zero position. In order for the cart to move with purpose, the pendulum must have a non-zero angle. This behavior is what would be expected in a real application and helps validate the dynamic model.

The next iteratively tuned PD surface simulation is for a sinusoidal trajectory with both a non zero initial condition and a pendulum rod disturbance (occurs at the halfway point in the simulation). This simulation encompasses

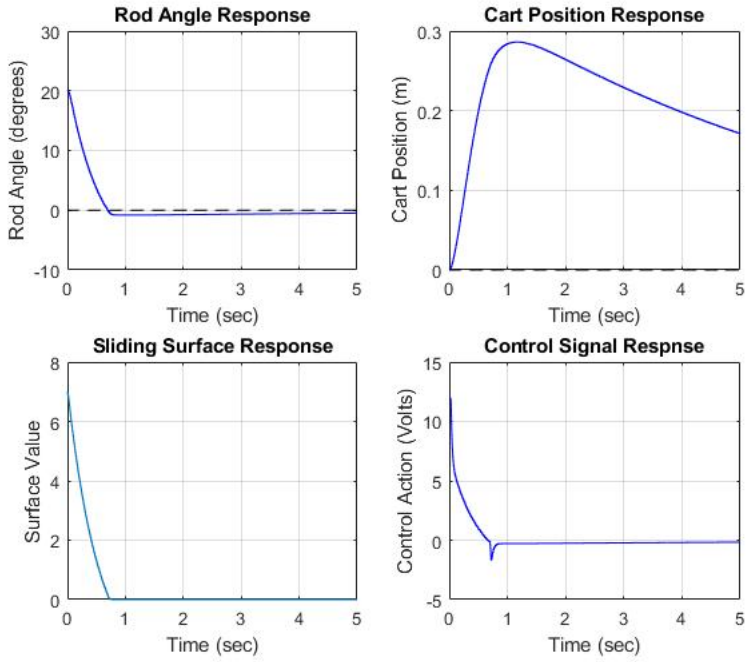

Figure 4.2: Inverted Pendulum: Hand Tuned PD SMC Initial Condition Rejection

the majority of issues a controller would have to deal with during operation. Figure 4.3 shows the system and controller response for this situation. The amplitude and frequency of the sinusoidal setpoint was kept within reasonable limits to prevent excessive cart displacements and voltage signals.

Again, the iteratively tuned sliding mode controller is able to stabilize the system while maintaining good tracking performance. Even when a large rod angle disturbance is added, the controller is capable of first stabilizing the system and then returning to the setpoint.

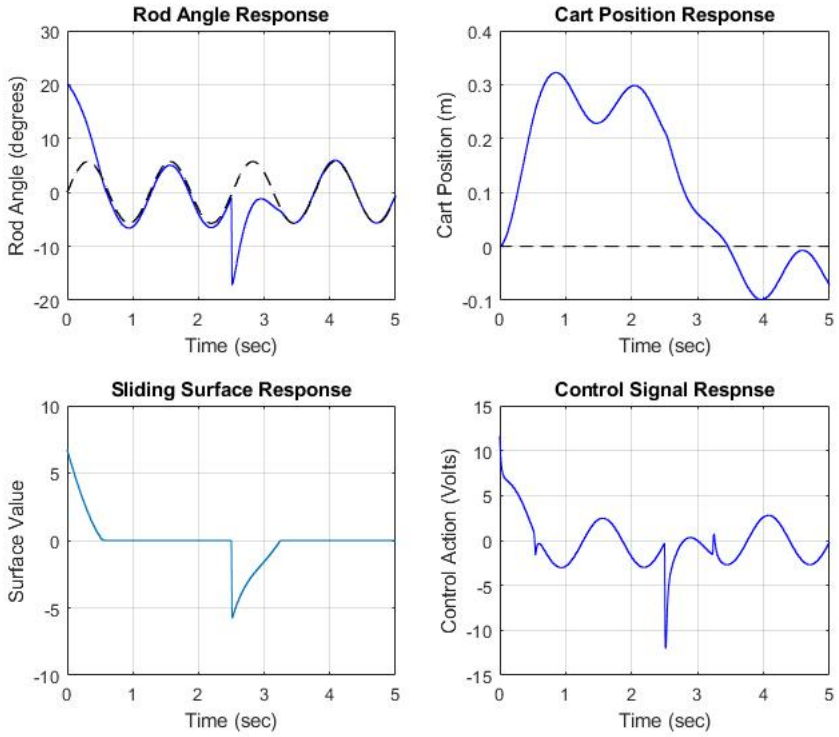

Figure 4.3: Inverted Pendulum: Hand Tuned PD SMC Setpoint Tracking

Given the success of the iteratively tuned PD SMC, the objective of the genetically tuned PD sliding mode controller for this system is to improve settling time and reduce minor tracking errors. Additionally, the genetically tuned PD SMC will remove the need for tedious trial and error tuning. These two benefits (performance and design) should be clearly demonstrated in the following simulations.

The genetically tuned PD sliding surface controller was first tested with only

the non zero initial condition. The same magnitude of pendulum rod angle was used for both sets of simulations to obtain a true comparison. Figure 4.4 and 4.5 show the performance of the genetically tuned controller. The difference between the two is the number of initial population and generations. This was done in an attempt to study the benefit versus additional computation time a high initial population brings.

The added run time resulted in different surface constants. The 20 population test returned values of 0.1856, 0.2081, 0.1034, and 0.8132 $\alpha_a, \alpha_u, \lambda_a, \lambda_u$. The 50 population test had only slightly different results with 0.1979, 0.1038, 0.1016, and 0.9936. The small change came with an increased run time which jumped from 128 seconds to almost 15 minutes.

The two figures show that both of the genetically tuned PD surface controllers outperform the iteratively tuned controller (ITC). The pendulum rod has a slightly better settling time and the cart has a substantially improved response. The ITC was only able to slowly move the cart back to the zero position while the genetic controllers actually brought the cart back to zero in three seconds. The improved cart response can be attributed to the genetic algorithm's novel fitness function definition. Both the mean squared error of the pendulum rod angle and cart position are included in the fitness function. This allows the genetic controller to effectively compensate both system errors while also filtering out surface values that cause control signal

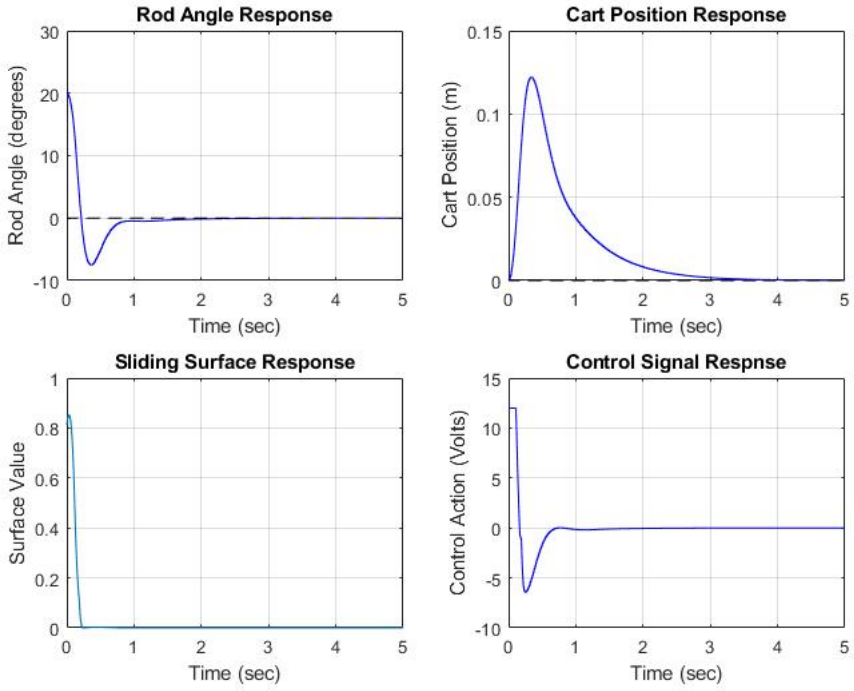

Figure 4.4: Inverted Pendulum: Genetically Tuned PD SMC with POP: 20, GEN: 10

chatter.

The higher population size and generation count controller showed marginally better performance but not enough to warrant the increased computation time. This result can be attributed to the simple nature of the simulation.

Next the genetically tuned PD controller was tested in simulation with the non zero initial condition, sinusoidal trajectory, and pendulum disturbance. Again, multiple simulations were done to study initial population size and

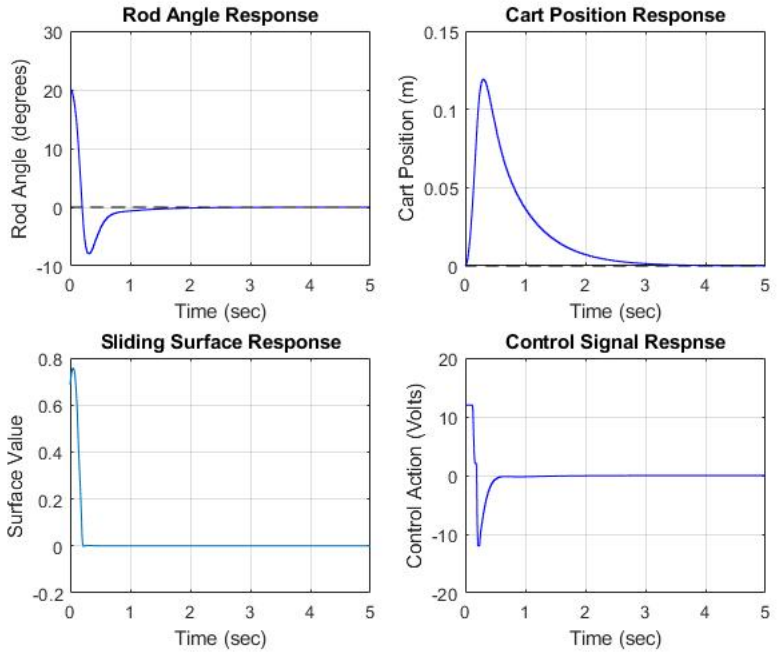

Figure 4.5: Inverted Pendulum: Genetically Tuned PD SMC with POP: 50, GEN: 30

generation number effects. Figure 4.6 and 4.7 show the performance of the genetically tuned controller given the more complex simulation situation.

The above figures show that the genetically tuned PD surface controller shows mixed results. There is a lag in the sinusoid tracking which is not found in the iteratively tuned controller. The cart tracking this time also does not show much improvement. These issues, however, can be fixed by adjusting the fitness function. Because this more complex trajectory and disturbances put more stress on the pendulum, the fitness function can be modified to better counteract this. Figure 4.8 shows the genetically tuned PD surface

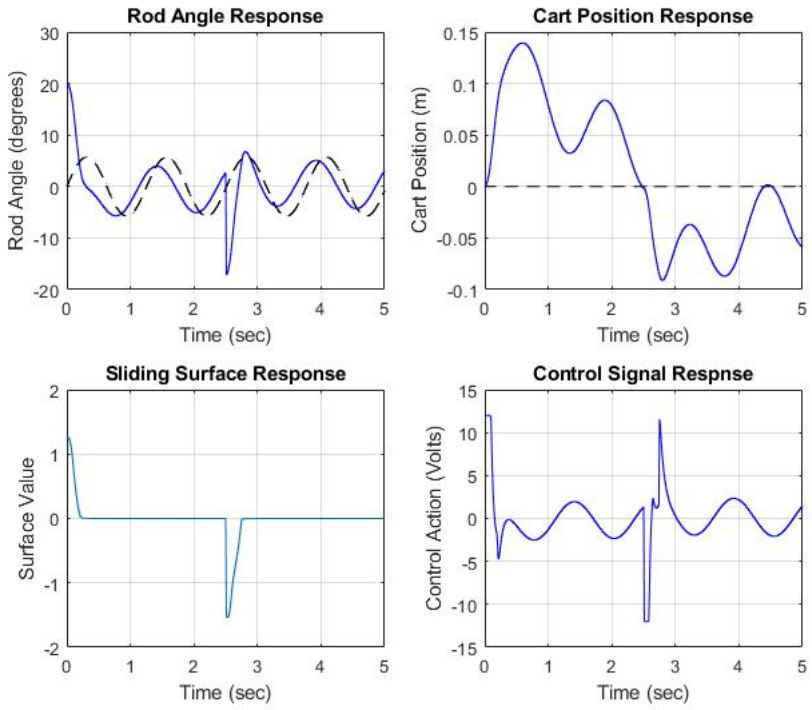

Figure 4.6: Inverted Pendulum: Genetically Tuned PD SMC with POP: 20 , GEN: 10

controller performance when operating with a custom fitness function better tailored to fit the situation. The algorithm was changed to only focus on the pendulum rod angle and controller chatter.

When more intelligently designing the fitness function to fit the expected operating situation, the genetically tuned PD surface controller shows stronger setpoint tracking. Not only does it outperform the previous genetic results, but it also outshines the iteratively tuned controller. The system snaps to the desired trajectory and recovers from the rod angle disturbance faster than all

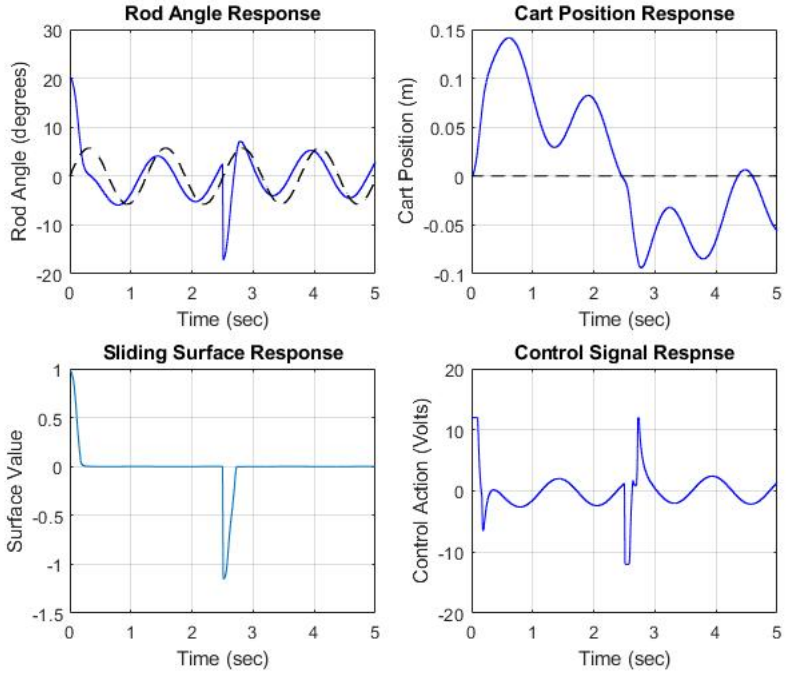

Figure 4.7: Inverted Pendulum: Genetically Tuned PD SMC with POP: 50 , GEN: 30

previous controller iterations. This result shows that proper consideration for the fitness function is a key part of the design for this control scheme. Deciding which errors to focus on will alter the system response in a nontrivial way.

The final set of simulations was performed using the full genetically tuned PID sliding surface controller. By adding an integrating piece into the sliding surface definition, the controller should better handle system tracking error. This adds two additional surface coefficients which makes iterative tuning that much more tedious. However, the genetic algorithm can easily handle

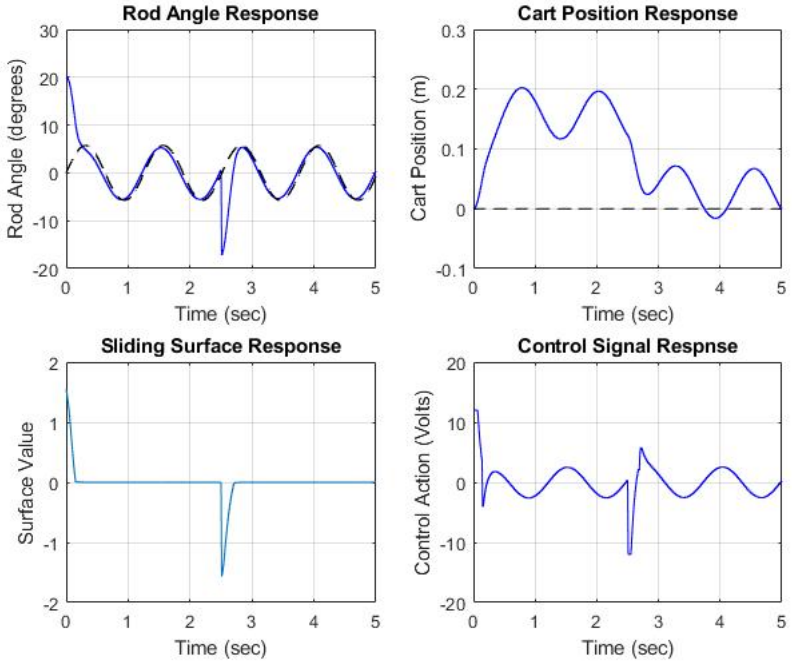

Figure 4.8: Inverted Pendulum: Genetically Tuned PD SMC with POP: 50 , GEN: 30 and a Modified Fitness Function

the additional variables with just a few small adjustments. Figure 4.9 shows the performance of the genetic PID surface controller for the complex simulation.

The genetically tuned PID sliding surface shows the best response yet. The added integrator in the sliding surface helped improve settling time and disturbance rejection. Additionally, despite the fact that the fitness function focuses on the pendulum angle error, the PID surface also keeps the cart position at a reasonable displacement while tracking the desired setpoint.

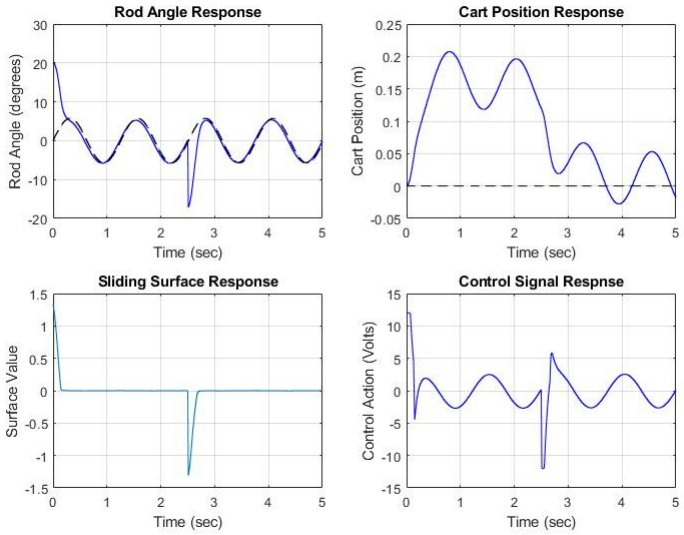

Figure 4.9: Inverted Pendulum: Genetically Tuned PID SMC with POP: 70 , GEN: 50

It should be noted that due to the additional surface constants the PID surface introduces, the genetic algorithm requires additional individuals and evolution generations compared to the other controllers.

The following Tables 4.2 and 4.3 show relevant performance indicators as well as the genetic algorithm results.

Table 4.2: Inverted Pendulum Simulation Results

#POP-#GEN	PD Iterative	PD GA 20-10	PD GA 50-30	PID GA 70-50
Alpha A	0.1	0.1856	0.1979	0.231
Lambda A	1	0.2081	0.1038	0.1074
Beta A	N/A	N/A	N/A	0.01
Alpha U	0.5	0.1034	0.1016	0.2527
Lambda U	20	0.8132	0.9936	4.0747
Beta U	N/A	N/A	N/A	0.01
Run Time	4.3 sec	128.3 sec	14.7 min	36.5 min

Table 4.3: Inverted Pendulum Performance Notes

Figure #	Rod Angle Tracking	Cart Position Tracking
4.2	Under 1 sec	20 cm off after 5 sec
4.3	Slow Convergence after Disturbance	Peaks at 32 cm stays near 0 point
4.4	Under 1 sec	3 sec
4.5	Under 1 sec	3 sec
4.6	Lags behind setpoint Fast Disturbance Rejection	Peaks at 14 cm Stays near 0 point
4.7	Lags behind setpoint Fast Disturbance Rejection	Peaks at 14 cm Stays near 0 point
4.8	Fast Disturbance Rejection Precise Tracking	Peaks at 20 cm Stays near 0 point
4.9	Fast Disturbance Rejection Zero Tracking Error	Peaks at 21 cm Stays near 0 point

4.1.3 Practical Implementation

This subsection details the experimental results collected for the single inverted pendulum test plant. Figure 4.10 and 4.11 show images of the test setup. The system is made up of extruded aluminum supports, one passive pulley, a DC motor, a driver belt, an aluminum pendulum rod, and a 3D printed/metal cart assembly. The system is controlled using an embedded micro-controller that is connected through USB to a computer.

To begin each test, the pendulum was started in the downward position. When ready, the pendulum was set by hand to be straight up. This triggered the controller to begin sending voltages to the system. In order to

Figure 4.10: Fully View of Inverted Pendulum Test Setup with Control Circuit

Figure 4.11: Close Up View of Pendulum-Cart Assembly and Pulleys

protect the plant and its users, a safety limit of 30 centimeters was placed on both the positive and negative cart axes. If the cart exceeded this limit, power to the system was cut off to avoid unstable behavior and having the cart hit the pulley brackets. Data was collected from the embedded controller by storing and then printing the desired variables. This data was then post-processed in Excel.

The first test was performed using an Linear Quadratic Regulator (LQR) control strategy. This was done to obtain a performance benchmark to which further controllers will be compared. An LQR is a popular linear control

technique that minimizes a cost function in order to generate a control action. While an LQR is an effective control scheme, because it linearizes the system, it struggles with larger magnitude initial conditions and disturbances. The test results should show that where the LQR fails in robustness, the developed SMC controllers can still maintain system stability. Figure 4.12 and 4.13 show the experimental results of the LQR scheme.

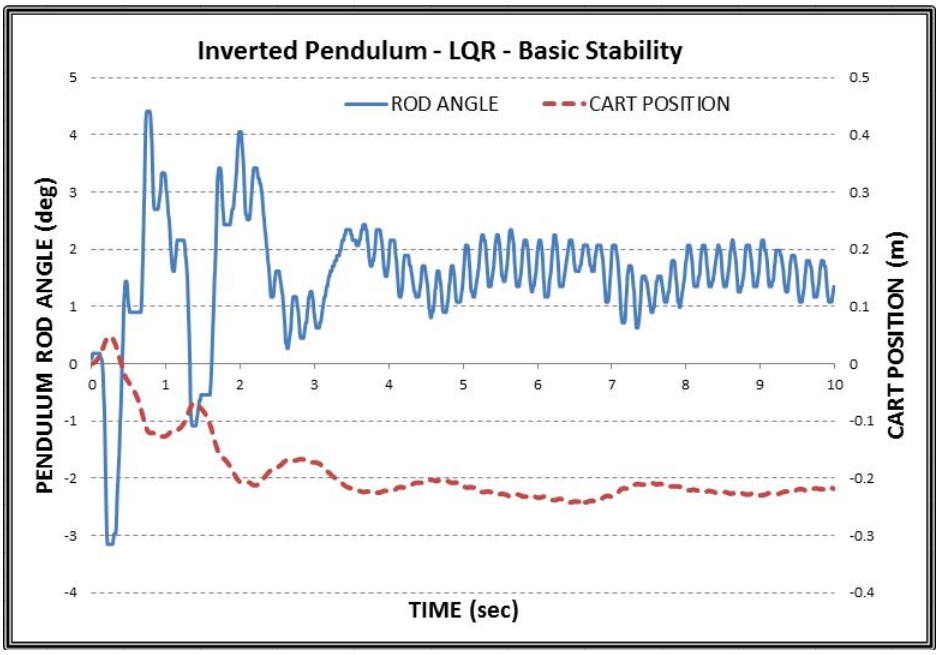

Figure 4.12: Inverted Pendulum: Basic Stability Performance of the LQR Controller

It can be seen that the LQR is able to maintain the pendulum rod at the zero degree equilibrium point. After initially struggling to stabilize, the LQR holds the pendulum between one and two degrees for the rest of the test. Dur-

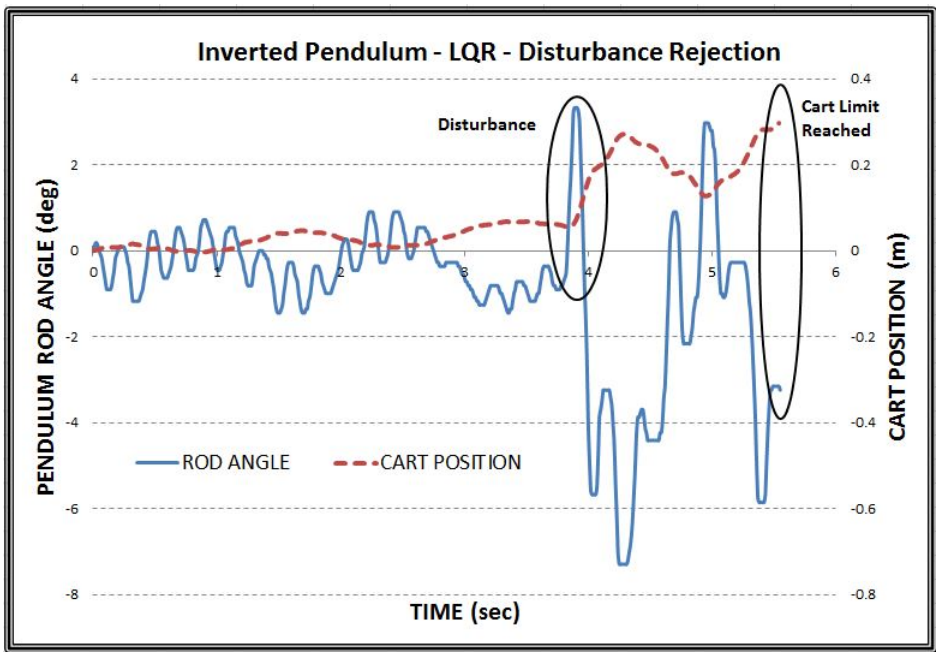

Figure 4.13: Inverted Pendulum: LQR Disturbance Rejection

ing the test the cart stays within the 30 centimeter boundaries but does drift close to failure at over 20 cm of displacement. Due to the relatively high resting rod angle and cart displacement, it can be said that there is clear room for improvement in the LQR controller performance. Note that small cart movements are occurring while the pendulum is oscillating from t= 4 sec to t= 10 sec. Also, the control actions cause the rod to vibrate to a certain degree which may cause further oscillations.

While the LQR had success with the basic stability test, it did not fare as well with the disturbance rejection test. For the test, the system was allowed to stabilize and was then manually perturbed. Ideally the control scheme would

quickly sense the disturbance and take action to prevent instability and return to equilibrium. Looking at the LQR disturbance rejection results, it can be seen that ultimately the controller failed to reject the disturbance. The LQR initially corrects the disturbance, but the resulting system movement causes the cart to exceed the safety limits. A superior controller should be able to reject the disturbance without excessive cart movement. This is the target result for the upcoming sliding mode controllers.

Shifting to sliding control schemes, Figure 4.14 and 4.15 show the basic stability and disturbance rejection of a PD surface sliding mode controller.

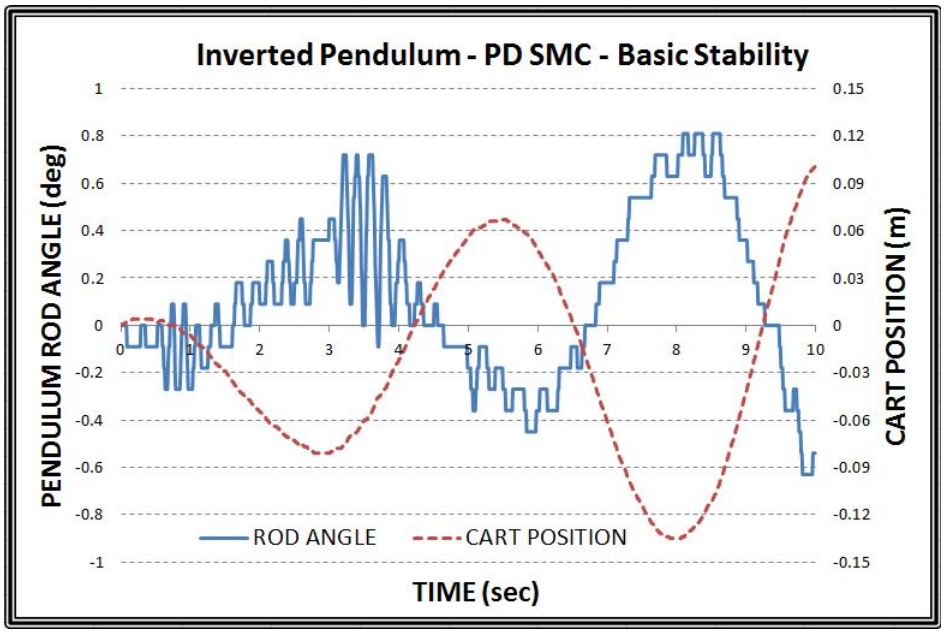

Figure 4.14: Inverted Pendulum: Basic Stability Performance of the PD SMC Controller

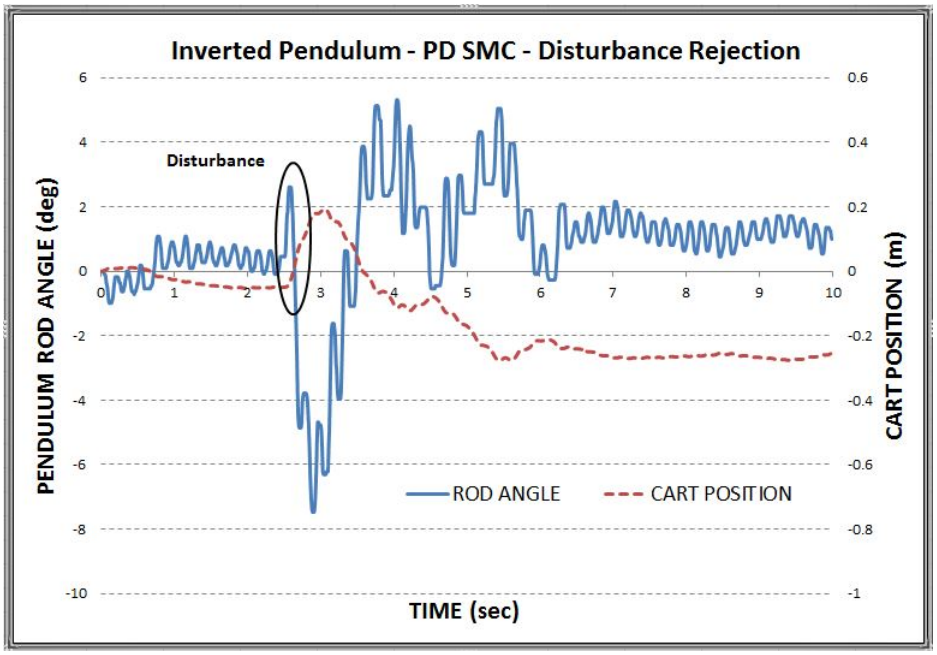

Figure 4.15: Inverted Pendulum: Basic Stability Performance of the PD SMC Controller

For the PD sliding surface controller, both the stability and disturbance tests gave better results than the LQR controller. The PD SMC held the pendulum rod angle magnitude under 1 degree for the entire test. Additionally, during the stability test the cart displacement did not exceed 15 cm and oscillated around the zero marker. This shows that both the pendulum angle and cart position were well controlled and stabilized. For the disturbance rejection, the PD SMC successfully rejected the pendulum rod perturbation and did not breech the cart limits in doing so. Clearly this was a better result than the LQR, which failed the disturbance rejection test.

While the PD sliding mode controller has already demonstrated better results than the LQR, the PID SMC should be able to achieve a system performance that is better than either controller. The addition of the integrating component in the sliding surface should help reduce steady state error and also react faster to unexpected system disturbances. Figure 4.16 and 4.17 show the stability and disturbance tests for the PID sliding mode controller.

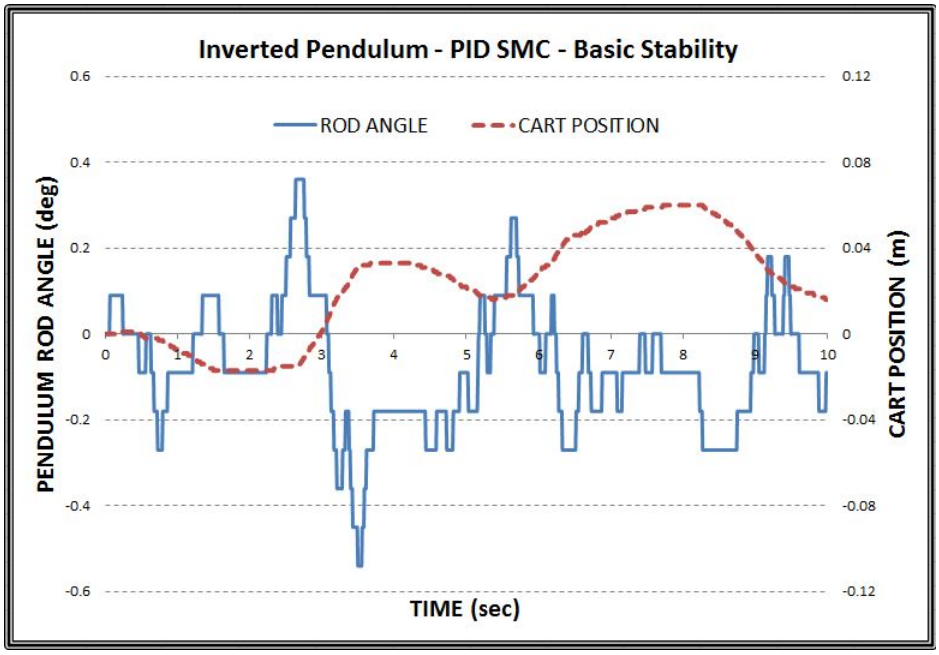

Figure 4.16: Inverted Pendulum: Basic Stability Performance of the PID SMC Controller

As expected, the PID SMC demonstrates the best performance of the three tested controllers. The pendulum rod angle was held to under 0.5 of a degree while the cart never exceeds a displacement of 6 cm during the stability test.

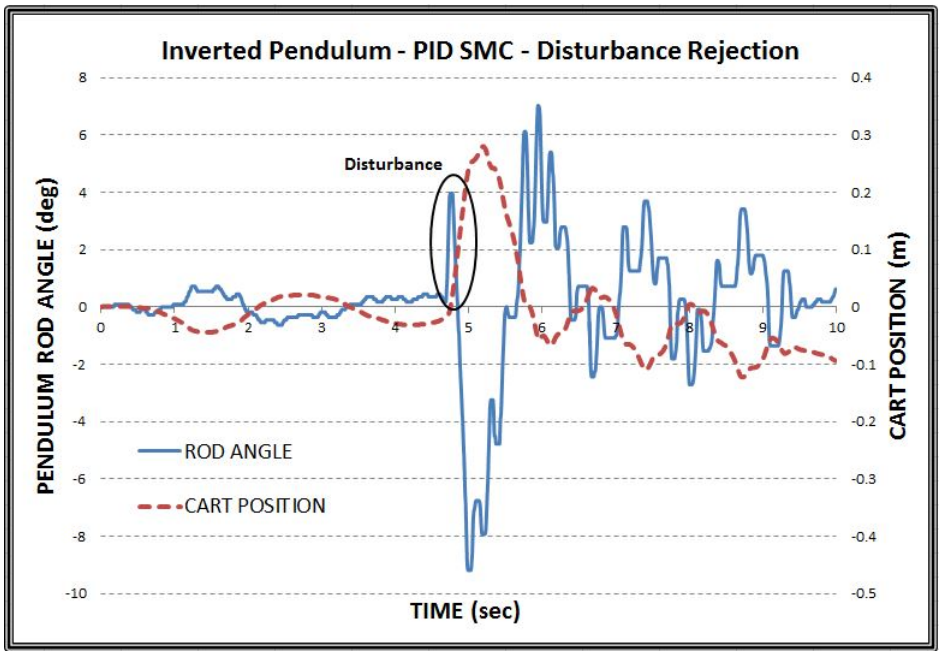

Figure 4.17: Inverted Pendulum: Basic Stability Performance of the PID SMC Controller

This is an improvement on the PD SMC and is much better than the LQR scheme. For the disturbance rejection test, the PID SMC was also able to handle the pendulum angle disturbance and stay within the safety bounds. The magnitude of the PID SMC disturbance test was also larger than either of the previous tests and it still brought the system back to equilibrium.

The PID sliding mode controller was also put through an additional test that the other controllers were not. Setpoint tracking is a common problem in the field of controls but achieving it on under-actuated systems is challenging. Figure 4.18 shows the result of the PID SMC tracking a sinusoidal cart

position setpoint.

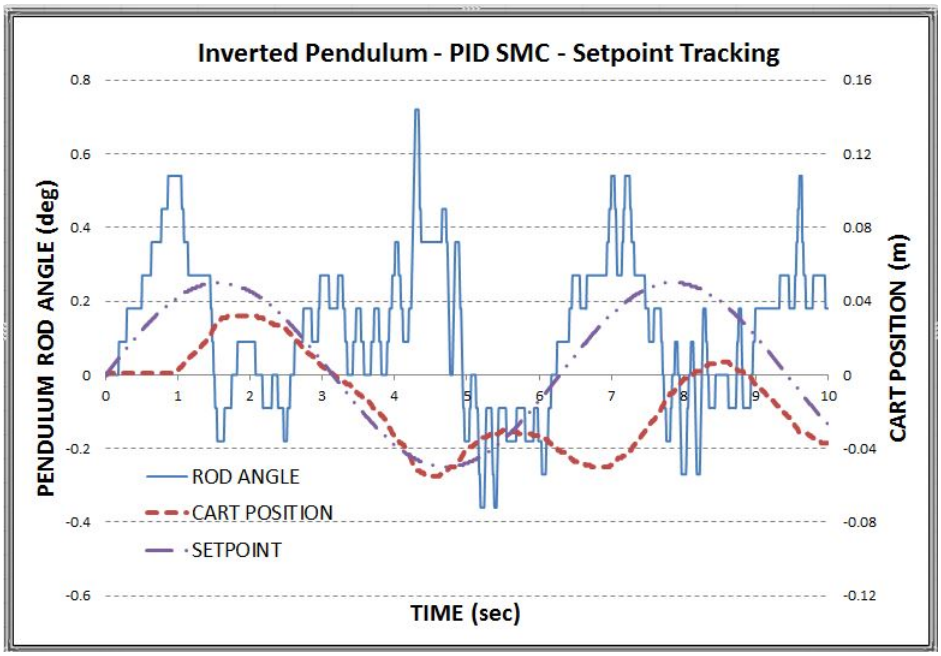

Figure 4.18: Inverted Pendulum: PID SMC Sinusoidal Cart Position Setpoint Tracking

While the overall tracking was not perfect, the PID SMC does manage stay relatively close to the desired setpoint while maintaining stability in the pendulum rod. Figure 4.19 shows a plot of the error during the cart position tracking test.

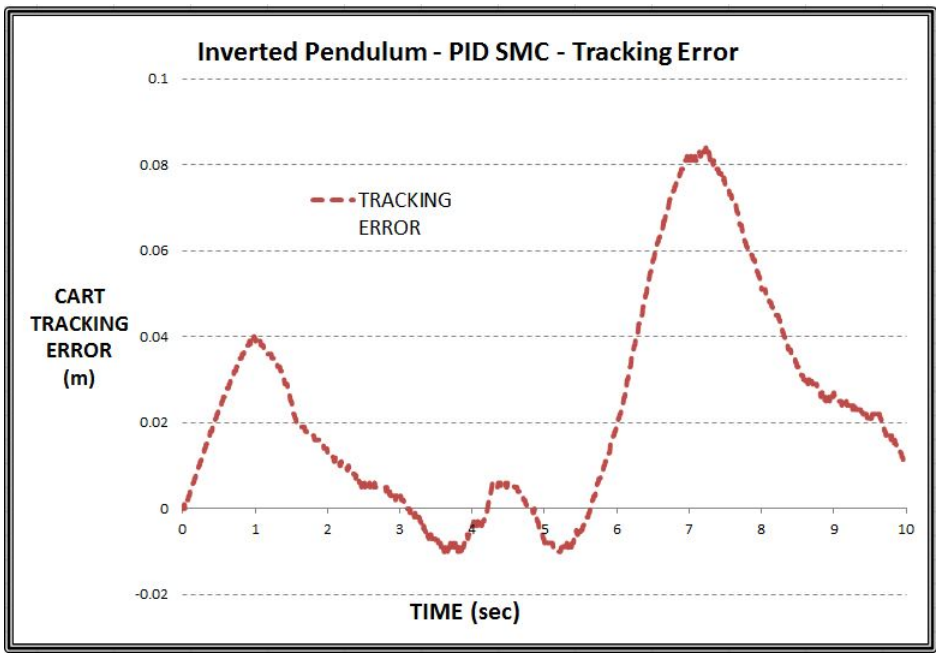

Figure 4.19: Inverted Pendulum: PID SMC Sinusoidal Cart Position Setpoint Tracking Error

4.2 Pendubot Mechanism

This case study focused on a PenduBot system. This system is made up of two rods connected in series. The bottom rod is actuated by a DC motor while the top rod is free swinging. This system is similar to a double inverted pendulum cart system but instead of an actuated cart, the plant is driven using the bottom rod. The goal of this system is to balance both rods along the vertical axis. Small magnitude initial conditions and setpoints were tested. This system is very unstable so complex trajectories are not realistically followable.

4.2.1 Dynamic Model

Like the previous model, the PenduBot model is presented in state space form and was formulated using the Euler-Lagrange method. Figure 4.20 shows the free body diagram of the PenduBot system. The dynamic model of the PenduBot system is shown in equation series 4.2. Table 4.4 contains the system parameters used in the controller design and simulation of the PenduBot mechanism.

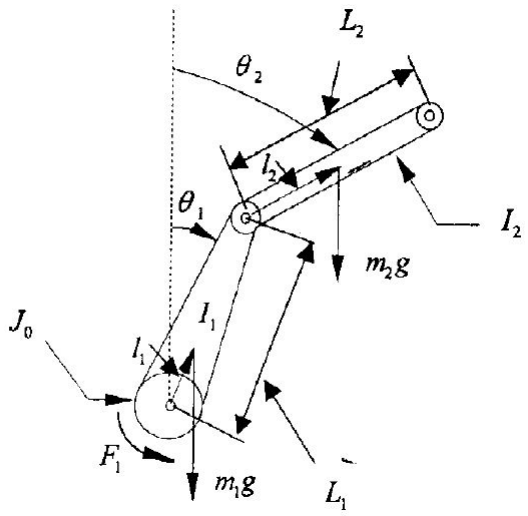

Figure 4.20: Free Body Diagram of the PenduBot System

$$M(q)\ddot{q} = f(q,\dot{q}) + \tau \tag{4.2a}$$

$$M(q) = \begin{bmatrix} J_1 + m_1 L_{C1}^2 + m_2 L_1^2 & m_2 L_1 L_{C2} \cos(q_1 - q_2) \\ m_2 L_1 L_{C2} \cos(q_1 - q_2) & m_2 L_{C2}^2 + J_2 \end{bmatrix} \tag{4.2b}$$

$$f(q,\dot{q}) = \begin{bmatrix} -m_2 L_1 L_{C2} \sin(q_1 - q_2)\dot{q_2}^2 + g(m_1 L_{C1} + m_2 L_1)\sin(q_1) - C_2\dot{q_1} \\ m_2 L_1 L_{C2} \sin(q_1 - q_2)\dot{q_1}^2 + m_2 g L_{C2} \sin(q_2) \end{bmatrix} \tag{4.2c}$$

$$\tau = \begin{bmatrix} C_1 u_0 \\ 0 \end{bmatrix} \tag{4.2d}$$

Table 4.4: PenduBot Model Parameters

Parameter	Symbol	Value	Unit
Link 1 Mass	m_1	0.175	kg
Link 2 Mass	m_2	0.065	kg
Link 1 Length	L_1	0.08	m
Link 2 Length	L_2	0.26	m
Link 1 Center of Mass Length	L_{C1}	0.03	m
Link 2 Center of Mass Length	L_{C2}	0.13	m
Gravity Constant	g	9.81	$kg\frac{m}{s^2}$
Link 1 Inertia Moment	J_1	$1e^{-4}$	kgm^2
Link 2 Inertia Moment	J_2	$5e^{-4}$	kgm^2
Coefficient 1	C_1	1.2778	$\frac{N}{V}$
Coefficient 2	C_2	0.01	$\mathrm{kg}\frac{m}{s}$

4.2.2 Simulation

This subsection focuses on the simulation results collected for the PenduBot mechanism. The main focus of the following simulation tests was to show that the genetic algorithm can find optimal surface constants in situations where iterative tuning is not feasible. Figure 4.21 shows the results collected for the basic iteratively tuned PD sliding mode controller. The controller was tasked with returning the system back the upright position given a small initial condition (5 degrees) for the unactuated second link.

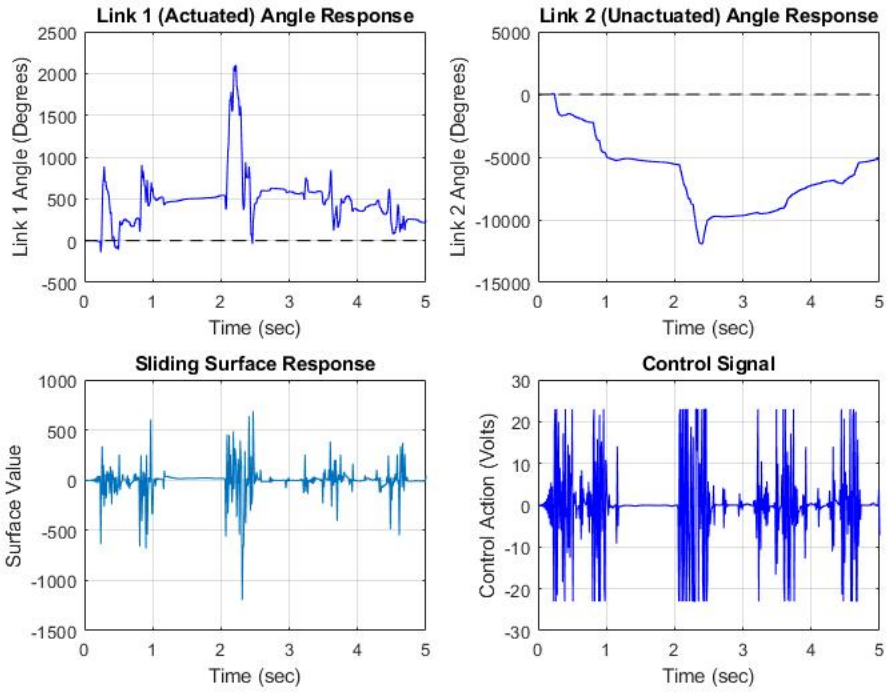

Figure 4.21: PenduBot System Response using PD Sliding Mode Control

It can be easily seen that the PD sliding mode controller failed to stabilize the system in the upright position. The system becomes unstable and demonstrates chaotic behavior as the controller continuously tries to balance the system. It should be stated that eventually the PD SMC could be iteratively tuned to find constants that are capable of controlling the system. However, this is an impractical solution and akin to guessing the combination of a padlock through trial and error.

Given the above result, the genetically tuned PID sliding mode controller should show that not only can it recover from the small initial condition, but that it can also handle higher I.C.'s and disturbances. Figure 4.22 shows the PenduBot system response when controlled by the PID SMC given the same initial condition of 5 degrees on the second link. The controller was tuned with a twenty individual population evolving over ten generations.

While the PID SMC does manage to outperform the traditional SMC, it still was ultimately unable to stabilize the system. The second link of the PenduBot slowly begins to oscillate to higher and higher angles until it finally falls. By increasing both the population size and number of generations to 50 and 30 respectively, the performance of the PID SMC drastically improves. Figure 4.23 shows the PenduBot system response under the same simulation conditions when the PID SMC has 50 individuals evolving over 30 generations.

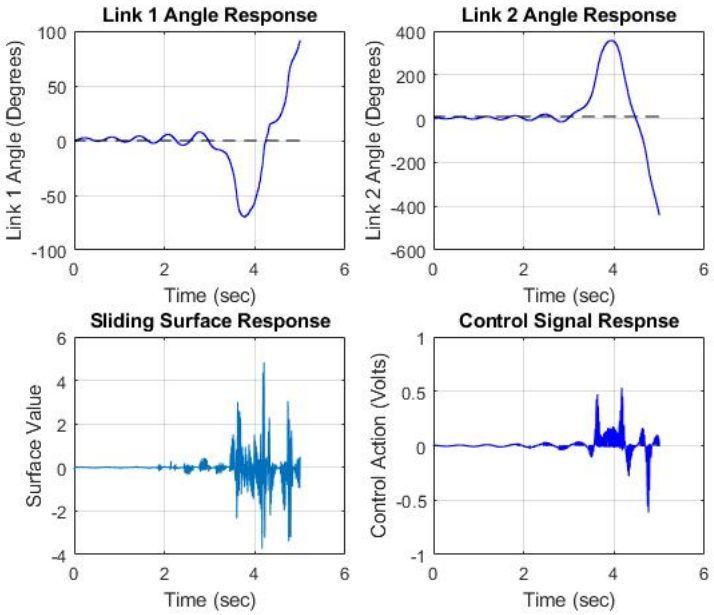

Figure 4.22: PenduBot System Response using PID Sliding Mode Controller POP:20 GEN:10

Given sufficient optimization conditions, the genetic algorithm was successfully able to tune the PID SMC to stabilize the PenduBot mechanism. Now that the PID SMC has rejected the small initial condition, further simulations were performed using larger starting conditions and disturbances. Figure 4.24 shows how the genetically tuned PID SMC deals with the aforementioned challenges.

Again the genetically tuned PID SMC rejects the larger initial conditions (10 degrees) and the added system disturbances (5 degrees). These results shows that not only does genetic tuning greatly improve surface constant se-

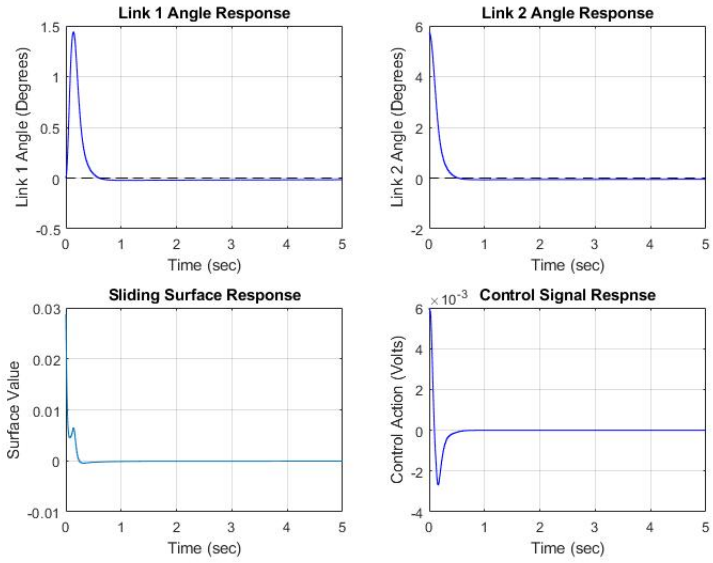

Figure 4.23: PenduBot System Response using PID Sliding Mode Controller POP:50 GEN:30

lection, but that the PID SMC scheme is a superior control technique when compared to the standard sliding mode controller. The following Tables 4.5 and 4.6 provide the GA tuning simulation result and performance notes.

Table 4.5: PenduBot Simulation Results

#POP-#GEN	PD Iterative	PID GA 20-10	PID GA 50-30
Alpha A	0.6778	0.4712	0.3028
Lambda A	9.5	9.9559	9.9973
Beta A	N/A	1.7024	0.1274
Alpha U	0.6283	0.4776	0.4387
Lambda U	0.6391	0.1617	0.1169
Beta U	N/A	0.1799	0.1165
Run Time	1.7 sec	153.5 sec	15.7 min

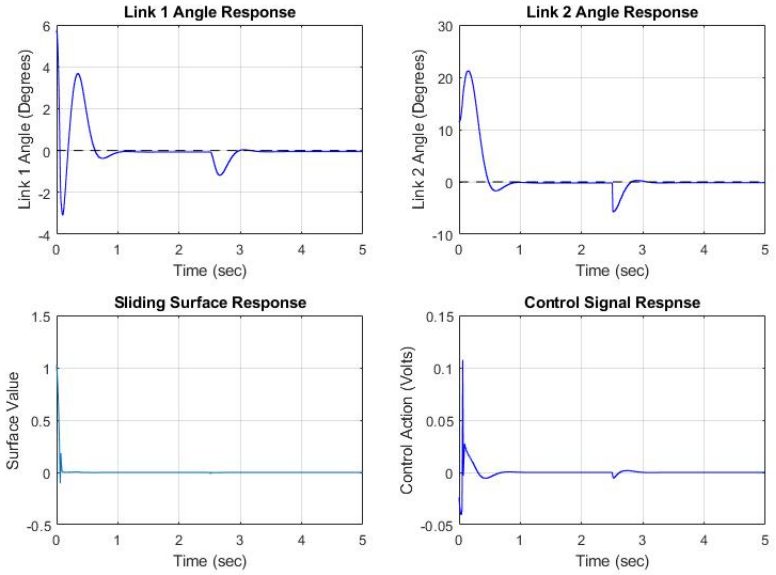

Figure 4.24: PenduBot System Response with larger Initial Conditions and Disturbances using PID Sliding Mode Controller POP:50 GEN:30

Table 4.6: PenduBot Performance Notes

Figure #	Link 1 Tracking	Link 2 Tracking
4.20	Total Failure	Total Failure
4.21	Oscillates to Failure	Oscillates to Failure
4.22	Rejects IC and Disturbance	Rejects IC and Disturbance
4.23	Under 1 sec	3 sec

4.3 Two Wheeled Robot

The following case study focuses on a two wheeled self balancing robot (TWSBR). The system is made up of a main body and two wheels. The body is free spinning with respect to the wheels and the only actuation available is the wheel motor voltages. The goal of the system is to keep the main

body of the robot upright while the robot moves around. Ideally, the robot could tracking specified paths and wheel velocities while staying balanced. This system is unstable and has very aggressive reactions to the control input making it a difficult and popular control problem.

4.3.1 Dynamic Model

Like the previous models, the two wheeled robot model is presented in state space form and was formulated using the Euler-Lagrange method. Figure 4.25 shows the free body diagram of the TWSBR system. The dynamic model of the robot is shown in equation 4.3. Table 4.7 contains the system parameters used in the controller design and simulation of the two wheeled robot.

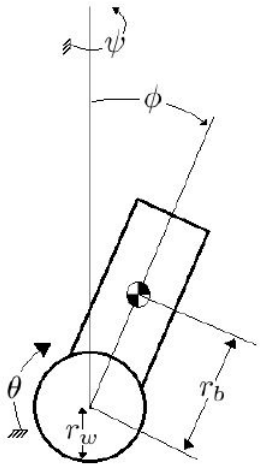

Figure 4.25: Free Body Diagram of Two Wheeled Robot

$$\begin{bmatrix} 3m_w r_w^2 + m_b r_w^2 & m_b r_b r_w \cos(\Phi) \\ m_b r_b r_w \cos(\Phi) & I_{bx} + m_b r_b^2 cos(\Phi) \end{bmatrix} \begin{bmatrix} \ddot{\Theta} \\ \ddot{\Phi} \end{bmatrix} = \begin{bmatrix} m_b r_b r_w sin(\Phi)\dot{\Phi}^2 \\ m_b r_b^2 cos(\Phi) sin(\Phi)\dot{\Phi}^2 - m_b g r_b \sin(\Phi) \end{bmatrix} + \begin{bmatrix} \tau \\ 0 \end{bmatrix} \tag{4.3}$$

Table 4.7: Two Wheeled Robot Model Parameters

Parameter	Value	Unit
Body $\hat{x}$ Inertia, I_{bx}	0.0677	kgm^2
Body Mass, m_b	4.9303	kg
Wheel Mass, m_w	0.2706	kg
Axial to Center of Mass, r_b	0.0417	m
Wheel Radius, r_w	0.0762	m

4.3.2 Simulation

This subsection focuses on the simulation results collected for the two wheeled self balancing robot system. Where as the previous case studies focused on controller comparison and genetic tuning advantages, these simulations tended more towards trajectory tracking. Because the two wheeled robot is a much more mobile system, trajectory tracking is a more likely control requirement. The main state that should follow the desired setpoint is the wheel velocity. This state is both the only point of actuation for the system and the means by which the robot moves. The other states should be kept stable by the controller during operation. Various velocity profiles were

tested such as steps and sinusoids. Initial conditions and disturbances were also included to push the limits of the genetically tuned PID sliding surface controller.

The first simulation is a basic stability test when the system is faced with initial conditions and disturbances. Figure 4.26 shows the two wheeled robot's response when given an initial body angle of approximately 17 degrees. The system was also disturbed half way through the simulation with a sudden body angle change.

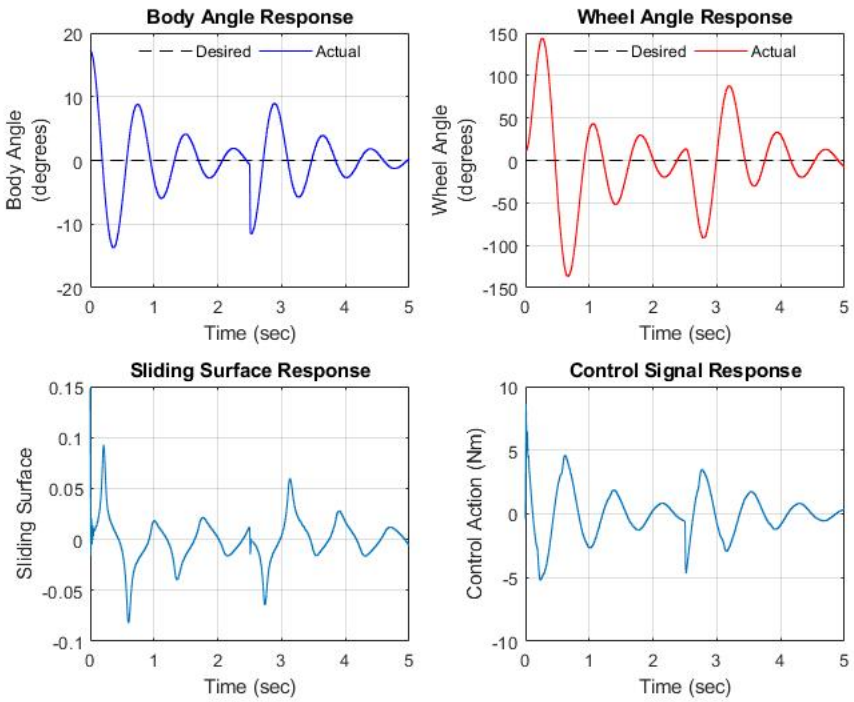

Figure 4.26: Two Wheeled Robot System Stability Response with PID SMC

The genetically tuned PID sliding controller was able to stabilize all of the system states while rejecting the initial conditions and the disturbance. The algorithm returned a solution of 0.9974, 0.9096, 0.1, 0.1, 0.141, and 0.966 for $\alpha_a, \lambda_a, \beta_a, \alpha_u, \lambda_u, and \beta_u$ respectively. The simulation took a total of 37 minutes to run.

Now that the controller has successfully stabilized the system, the next simulations focus on wheel velocity setpoint tracking. Figure 4.27 shows the system response as the controller tracks a positive and negative wheel velocity step setpoint.

The genetically tuned PID sliding controller was able to successfully track the wheel velocity setpoint while still maintaining stability in the other states. The body angle does a fair bit of oscillating but this is due to the wheel velocity and body angle being dynamically connected. The robot cannot move unless there is a non zero body angle. The simulation shows that not only can the controller handle stability tasks, but it can also produce solid tracking results. The final simulation featured a sinusoidal wheel velocity trajectory setpoint. Figure 4.28 shows the two wheel robot response to the sinusoidal velocity tracking.

A lower frequency sine wave was used as the wheel velocity setpoint to avoid sending the system into unstable oscillations. The genetically tuned PID slid-

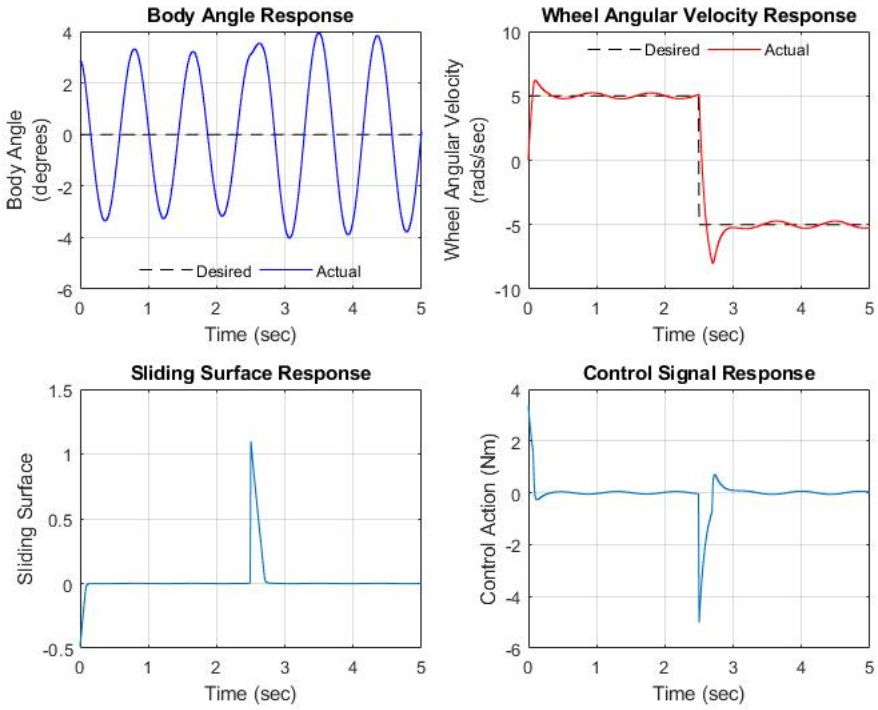

Figure 4.27: Two Wheeled Robot Velocity Step Tracking with PID SMC

ing controller was also able to handle the sinusoidal setpoint. These three simulation results provide validation that the new control scheme can simultaneously handle stability and tracking tasks for each system state.

The following Table 4.8 provides simulation performance notes for the two wheeled robot.

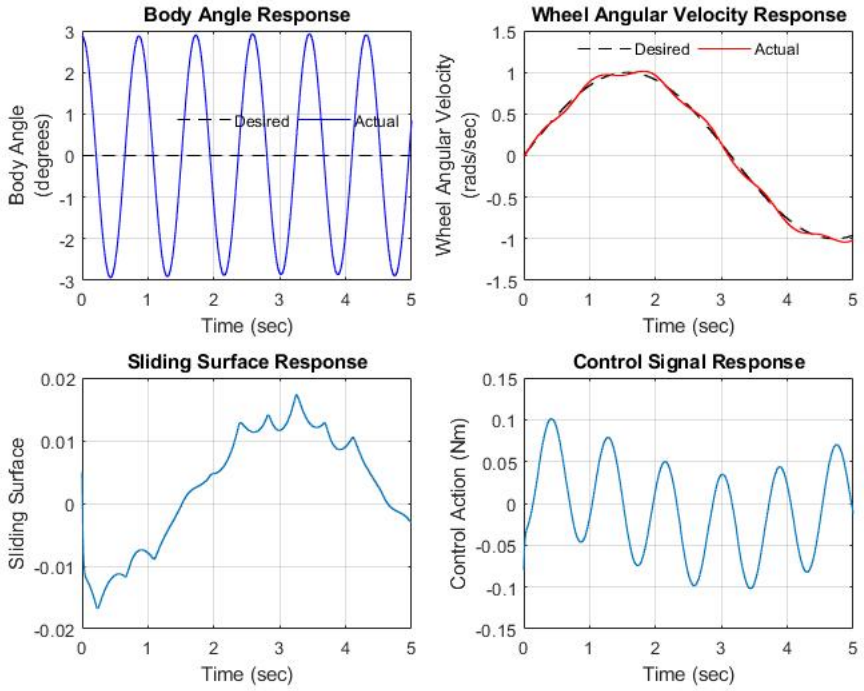

Figure 4.28: Two Wheeled Robot Sinusoidal Velocity Tracking with PID SMC

Table 4.8: Two Wheel Robot Performance Notes

Figure #	Body Angle Tracking	Wheel Angle Tracking
4.25	Stable Oscillations	Oscillates about zero
4.26	Oscillates between +/- 4 degrees	Small overshoot and oscillations
4.27	Oscillates between +/- 3 degrees	Small Oscillations

4.3.3 System Identification

This subsection details the work done to estimate the friction and control action coefficients for the real world testing of the two wheeled robot. Due to the fact that the real system experiences friction and is controlled through voltage signals, extra steps must be taken to ensure an accurate model. This

ensures that any controller applied to the system has the best chance of being successful.

The system identification work consisted of two sets of tests, one of just the robot and one when a known mass was added to the base of the robot. Equation (4.4) shows the first order differential equation that was used to approximate the system.

$$m\dot{w} + C_1 w = C_2 U_0 \tag{4.4}$$

Where m (kg) is the system mass, w (m/s) is the linear speed of the wheels, C_1 (kgm/s) and C_2 (N/V) are the friction and voltage coefficents, and U_0 (V) is the control action.

If a known mass (M= 0.345 kg) is added to the system then the equation becomes:

$$(m + M)\dot{w} + C_1 w = C_2 U_0 \tag{4.5}$$

Rearranging equations (4.4) and (4.5) and taking the Laplace Transform, the following equation set can be derived.

$$H_1(s) = \frac{\frac{C_2}{C_1}}{\frac{m}{C_1}s + 1} = \frac{K}{\tau_1 s + 1} \tag{4.6}$$

$$H_2(s) = \frac{\frac{C_2}{C_1}}{\frac{m+M}{C_1}s + 1} = \frac{K}{\tau_2 s + 1} \tag{4.7}$$

Running the system at a known voltage (2 Volts for this work) and then plotting the speed of the motor generates a time response curve. Repeating this process for the no mass and added mass cases creates two sets of data. For each scenario, three trials were performed. From this data the values of τ_1, τ_2, and K were extrapolated. From there the values of C_1 and C_2 were found.

The results of the system identification tests were that C_1 was found to be 4.22 kgm/s and C_2 was found to be 1.0346 N/V.

4.3.4 Practical Implementation

The following subsection details the experimental results collected for the two wheeled self balancing robot. Figure 4.29 shows an image of the robot used in the tests. The robot is made up of two dc motors with encoders for position feedback, a gyroscope chip, a dual motor driver chip, and an embedded micro-processor. The micro-processor is connected to a PC via USB cable. Plastic support bars were attached to either side of the robot to

prevent damaging the system during testing.

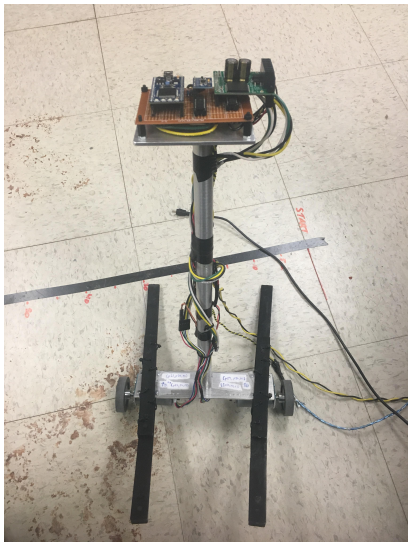

Figure 4.29: Picture of the Two Wheeled Robot

Each test performed on the Two Wheeled Robot (TWR) lasted for ten seconds. The goal of these tests were to balance the robot in the upright equilibrium position. The model of the real TWR was judged to be different than the theoretical model outlined above and more akin to the inverted pendulum model. Therefore, a new genetic algorithm was run in order to tune the controller for the different model. The outcome of the re-tuned surface variables was $\alpha_a = 0.1010, \alpha_u = 0.4201, \lambda_a = 0.1130, \lambda_u = 0.1007, \beta_a = 0.9979,$ and $\beta_u = 0.1408$. These are the surface values that were used for the experimental testing.

A total of three balancing tests were performed after adequately tuning and configuring the robot. Figure 4.30 shows the body angle of the robot during each of the three tests. The horizontal axis of the plot was shortened to five seconds in order to increase the visibility of the data.

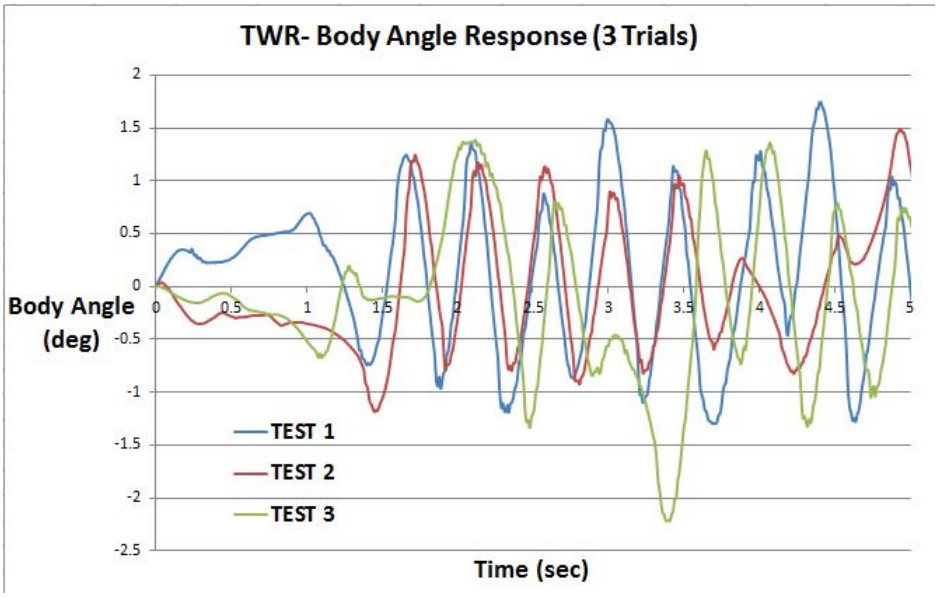

Figure 4.30: Body Angle Response of the Two Wheeled Robot

It can be seen that the genetically tuned PID SMC is successfully able to force the robot to the unstable upright equilibrium point. The robot body angle stays between +/-2 degrees for the duration of each test and only goes higher than that once (test 3 at 3.4 seconds).

The same result can be seen when analyzing the wheel position data for the three trials, shown in Figure 4.31. The controller is able keep the robot within a +/- 10 cm range. This shows that the controller only needs to make minor adjustments to maintain the robot in the upright position. It can be therefore stated that the genetically tuned PID sliding surface controller is successfully able to stabilize both system states: body angle and wheel position.

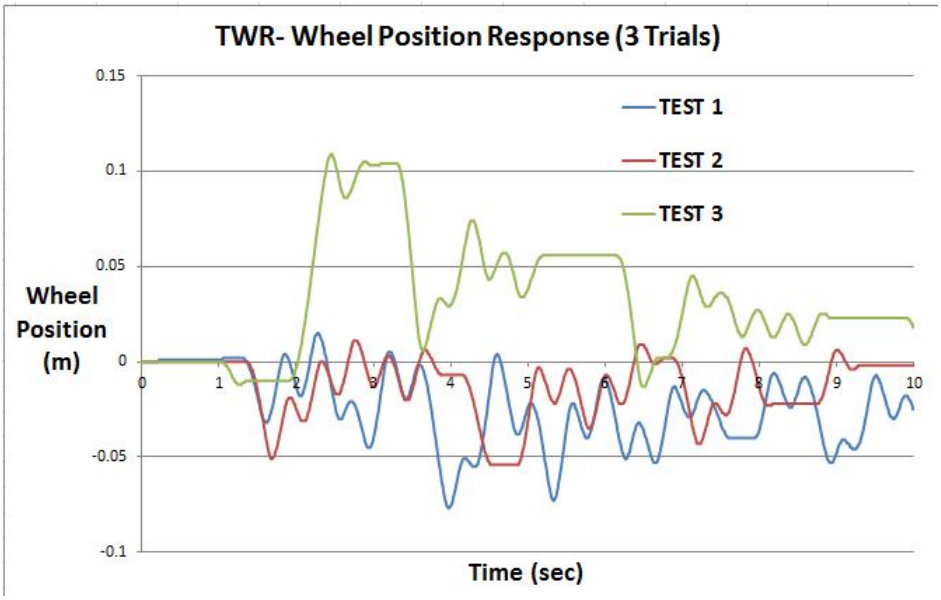

Figure 4.31: Body Angle Response of the Two Wheeled Robot

Figure 4.32 shows the control signal generated by the controller during the third trial. While the signal is slightly chattering, it is not doing so at excessively higher frequencies. This level of chatter would be expected from any control scheme implemented on this system. This is due to the fact that

the two wheeled robot is naturally unstable and will tend to oscillate about the equilibrium point during operation. The added chatter of a sliding mode controller appears to not have a significant impact on the overall control signal.

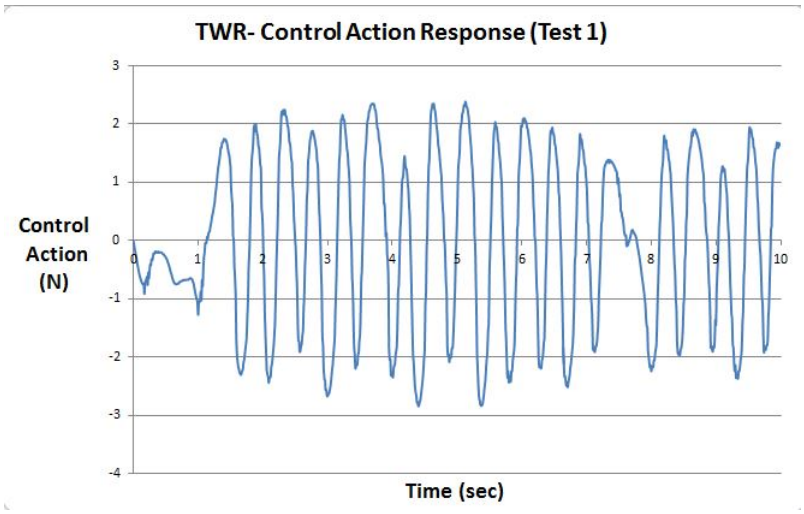

Figure 4.32: Control Action Response of the Two Wheeled Robot

Chapter 5

Discussion and Conclusions

The following chapter will summarize the simulation, implementation, and comparative results concerning the genetically tuned PID sliding surface controller. The overall results of the novel controller were very positive. The simplified design procedure allows for faster controller derivation when compared to more complex sliding controllers. The standard practice is to first perform a coordinate transformation on the math model and then design the controller. The method used in this thesis does not require a transform which are often extremely complex and labor intensive. The exclusion of the transform also permits the math model to remain in easily interpretable dynamics making setpoint design much simpler.

The design of the sliding surface has also been simplified compared to other sliding schemes [7]. Surfaces found in higher order SMC, super twisting

SMC, fast terminal SMC, and integral SMC are all mathematically complex to design. This is compounded when working with under-actuated systems which are already more complex to control than standard systems. The PID surface sliding mode controller uses an augmented first order sliding surface. By adding a basic integration term in the surface, the steady state error and disturbance rejection properties are improved without additional complexities in the controller design. Using a genetic algorithm to tune the sliding surface parameters also eliminates the need to arduously hand tune multiple constants.

Due to the ease of math model application and surface design, the PID sliding controller is very portable between different under-actuated systems. By changing out the dynamic model terms in the controller, one can quickly create an effective and functional control scheme for a new system. Additionally, the design of the tuning genetic algorithm can also be easily altered to fit another system. Therefore, tuning simulations can be promptly run in order to tune the controller. The following sections will go into further detail regarding key aspects of the results of this thesis.

One of the weaknesses of a sliding control scheme is the fact that there is significant chatter in the control signal. This is caused by the controller constantly trying to push the system towards the zero sliding surface value. A boundary layer around the sliding surface was introduced in the switching

control law to combat this chatter effect. The magnitude of the boundary layer is selected based on the relative magnitude of the sliding surface. For the work done in the thesis the boundary layer was kept to 0.1.

The results show that both the standard sliding mode and PID surface controllers produce less chatter than the LQR technique. The LQR showed not only more chatter than the sliding controllers but the chatter had a higher magnitude as well. Because the target of this thesis is under-actuated systems, the chatter effect of sliding controllers is less of an issue. Under-actuated systems are naturally unstable and chatter about their equilibrium points regardless of the implemented control scheme. This allows what was once a weakness of SMC to become a non-issue. This effect is seen in both the inverted pendulum and two wheeled robot systems.

5.1 Performance Comparison

This section concerns the comparison results collected during the real world testing of the inverted pendulum system. Three controllers were run on the system: a LQR, a standard sliding mode controller, and the genetically tuned PID sliding surface controller. Chapter four contains the relevant performance plots for the three implemented control schemes. The first performance comparison is found in Figure 5.1.

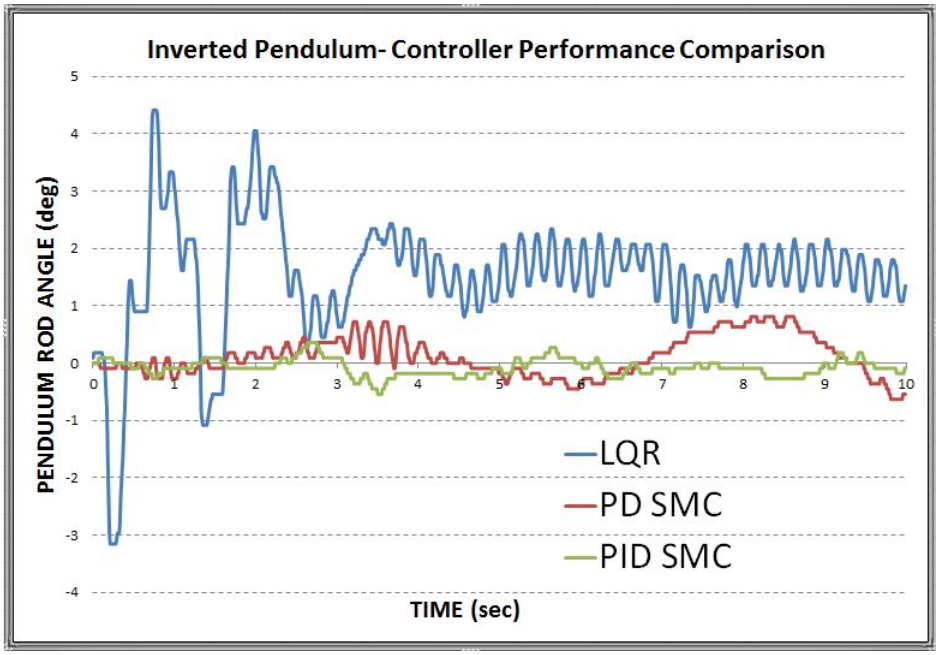

Figure 5.1: Inverted Pendulum: Comparison of the three control schemes

Comparing the pendulum rod angle for each of the three controllers, it can be seen that the PID SMC controller has the best performance. Both sliding controllers outperform the LQR scheme while the PID variant surpasses the standard sliding controller. Not only are the rod angle magnitudes much lower in the PID SMC result, but the angle also experiences less sign chatter compared to the first two control schemes. The PID SMC reduced the angle magnitude of the pendulum rod by 67% and 15% compared to the LQR and standard SMC respectively. Table 5.1 contains relevant performance metrics and comments for the three controllers.

System state response is only one aspect of a controller's overall performance.

Table 5.1: Controller Comparison Metrics

Controller	Peak Magnitude	Settling Time	Chatter	Settling Point
LQR	4.3 degrees	3.5 sec	High Frequency 1 degree oscillations	1-2 degrees
PD SMC	0.7 degrees	Instant	Medium	0 degrees
PID SMC	<0.5 degrees	Instant	Low	0 degrees

Another important indicator is the control effort required to properly control a system. If a controller produces slightly better performance at the cost of significantly higher control effort then it may not be worth it to the operator. Therefore it is important to compare the control signal voltage of the LQR and PID SMC. Figure 5.2 shows the control signal voltage from the stability tests of both controllers.

It can be seen that the PID SMC control voltage is not only of a lower magnitude but also contains less sign switching. Based on this result, it can be said that the PID SMC requires less energy to run when compared to the LQR controller. Better system response and less energy spent mean that the PID SMC controller is an overall superior control technique versus the linear quadratic regulator scheme.

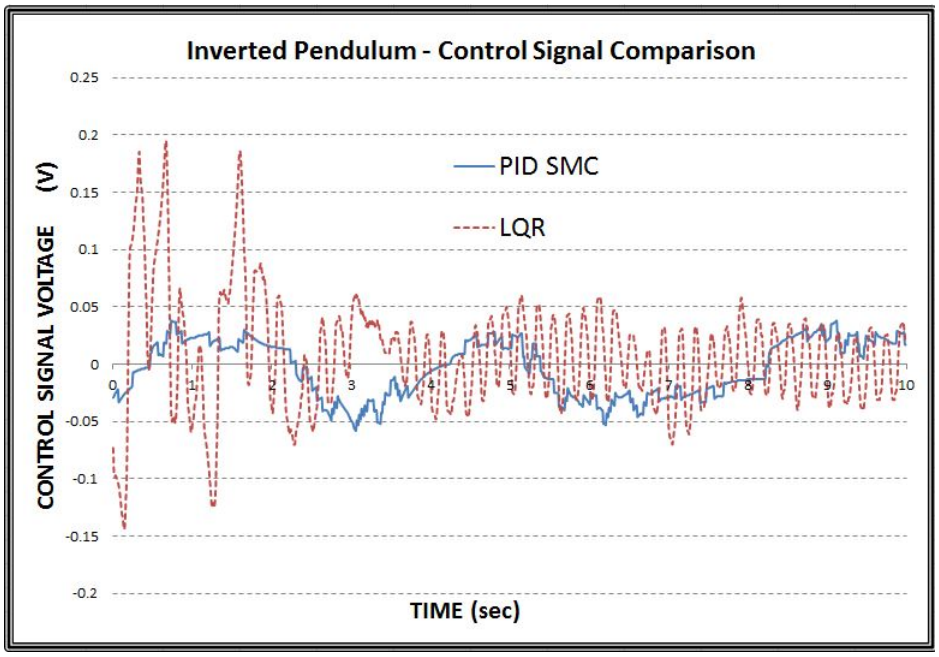

Figure 5.2: Inverted Pendulum Control Signal Voltage Comparison

5.2 Implementation Comments

This section focuses on the physical implementation of the PID SMC technique. The inverted pendulum and two wheeled robot systems were controlled using an embedded micro-controller (mbed LPC1768). The controller code was written in C++ based language on an online complier. To run a test, the code was compiled and then uploaded to the micro-controller via a USB connection.

After working on the implementations in this thesis, there are a few comments regarding best practice that others may find helpful. First of all,

because sliding mode controllers are very dependent on the system states, ensuring the accuracy of state measurements is paramount. Running validation tests will make further work much easier. Additionally, ensure that the controller states match the measured states with respect to positive and negative signs. If the signs of the measured states are inverted then obtaining effective control will be impossible.

The second implementation comment relates to interpreting the results of the genetic tuning algorithm. The algorithm is run in MATLAB using the system model and a ODE solving function. The genetic algorithm runs simulations with many combinations of sliding surface parameters in order to find the set of coefficients that produce the lowest mean square tracking error. Because this process happens in simulation, it is not totally faithful to real world applications. A simulated system does not have higher vibration modes, unmodeled dynamics, or frictional effects. Additionally, the limits selected for the simulation may not be valid for the real plant. This is something that cannot be totally known during the design process.

With this in mind, the results of the genetic algorithm are still valuable. The algorithm gives the control engineer insight into how the coefficients should be relatively portioned. The exact values may not work initially but by maintaining the same relative magnitudes between the surface coefficients while tuning, the most effective values can be found much faster than just itera-

tive tuning alone. The genetic algorithm may reveal that a certain surface value (the actuated state velocity coefficient for example) should be higher than intuition would suggest in order to obtain optimal control. In this way, the genetic tuning provides valuable insight into the interaction between the system's actuated and unactuated states that would otherwise be unknown.

In the case of the inverted pendulum, small changes had to be made to the optimal surface parameters due to excessive system vibration. However, for the two wheeled robot, the genetically tuned surface parameters were found to be satisfactory without any modifications. In summary, the genetic algorithm output should be seen as the best starting point and correct order of importance for the surface constants' weights and not as the final word in surface design.

5.3 Conclusions

The main goal of this thesis was to design a sliding mode control scheme that could effectively control under-actuated, unstable mechanical systems. Based on the theoretical, simulated, and experimental findings, this objective has been met. The PID SMC has been shown to guarantee stability of both the actuated and unactuated states via a Lyapunov-like analysis. The controller was also shown to be robust when dealing with system disturbances thanks

to the switching control law and PID sliding surface. The combination of exponential stability and general robustness makes the PID sliding surface technique an effective controller when dealing with under-actuated, unstable systems.

The secondary objectives were to show improved performance compared to other controllers and to prove that the PID SMC can be effective on actual physical systems. It was shown that the PID SMC is an effective controller for the inverted pendulum and two wheeled robot physical plants. This proves that the technique is actually useful and not only a theoretical controller. Both the simulated and experimental results indicated that the PID SMC produced superior performance both in terms of system stability and control effort. For the inverted pendulum physical tests, the PID surface scheme had the smallest rod angle variations and handled the rod disturbance the best. The PID sliding controller also used the lowest voltage during the tests. These two results complete both of the secondary objectives of this thesis.

The overall result of the work performed indicates that the newly developed genetically tuned PID sliding surface control scheme is an effective under-actuated system controller that outperforms other tested control techniques and is readily applicable to real world systems.

5.4 Recommendations and Future Work

The majority of recommendations relate to the physical implementation of the genetically tuned PID SMC technique. The research showed that making informed decisions regarding controller design is the key to successful tests. The boundary layer should be set to be sufficiently large in order to avoid chatter. In a physical test, the boundary layer should be keep around 0.1 and not lower than 0.05. Secondly, small modifications to surface constants can greatly affect controller performance. It is therefore recommended that only small changes be made and only one variable should be altered per test. This will help understand each constant's effect on the system response and assist zeroing in on the best surface parameters.

The main focus of possible future work utilizing the methods described in this thesis would be on further implementations. Applying the genetically tuned PID SMC technique on multiple physical plants would help strengthen the support for the controller and reveal where improvements could be made. The system that should be next in line to be tested is the PenduBot system. The PenduBot should show that the controller can handle very unstable and sensitive systems. The industrial manipulator and injection molding machine are two good starting points for testing industrial systems.

Bibliography

[1] H Ashrafiuon and Richard Erwin, *Sliding control approach to underactuated multibody systems*, Proceedings of the American Control Conference, vol. 2, 01 2004, pp. 1283 – 1288 vol.2.

[2] Mustafa Sinasi AYAS and Ismail Hakki ALTAS, *Fuzzy logic based adaptive admittance control of a redundantly actuated ankle rehabilitation robot*, Control Engineering Practice **59** (2017), no. Supplement C, 44 – 54.

[3] Chih-Yang Chen, Tzuu-Hseng S. Li, Ying-Chieh Yeh, and Cha-Cheng Chang, *Design and implementation of an adaptive sliding-mode dynamic controller for wheeled mobile robots*, Mechatronics **19** (2009), no. 2, 156 – 166.

[4] T.T. Chow, G.Q. Zhang, Z. Lin, and C.L. Song, *Global optimization of absorption chiller system by genetic algorithm and neural network*, Energy and Buildings **34** (2002), no. 1, 103–109.

[5] Timothy Coffey and John A Montgomery, *The emergence of mini uavs for military applications*, Defense Horizons (2002), no. 22, 1.

[6] Metin Demirtas, *Off-line tuning of a PI speed controller for a permanent magnet brushless DC motor using DSP*, Energy Conversion and Management **52** (2011), no. 1, 264–273.

[7] Sami Ud Din, Qudrat Khan, Fazal-Ur Rehman, and Rini Akmeliawanti, *A comparative experimental study of robust sliding mode control strategies for underactuated systems*, IEEE Access **5** (2017), 10068–10080.

[8] Mohammad El-Bardini and Ahmad M. El-Nagar, *Interval type-2 fuzzy pid controller for uncertain nonlinear inverted pendulum system*, ISA Transactions **53** (2014), no. 3, 732 – 743.

[9] Shlomo Engelberg, *Tutorial 15: control theory, part i*, IEEE Instrumentation & Measurement Magazine **11** (2008), no. 3, 34–40.

[10] Weslley Alves Farias, Eduardo Oliveira Freire, Sidney Nascimento Givigi, Elyson Adan Nunes Carvalho, and Lucas Molina, *Comparison between fuzzy and neural controllers to cross the reality gap in evolutionary robotics*, 2018 Annual IEEE International Systems Conference (SysCon), IEEE, apr 2018.

[11] Jaeyoung Han, Sangseok Yu, and Sun Yi, *Advanced thermal management of automotive fuel cells using a model reference adaptive control*

algorithm, International Journal of Hydrogen Energy **42** (2017), no. 7, 4328 – 4341.

[12] Paul Hershey, Mike Sica, and Mike Lewis, *Common ground control system (CGCS) to support autonomous object observation, collection, and response in multi-domain environments*, 2018 Annual IEEE International Systems Conference (SysCon), IEEE, apr 2018.

[13] John H. Holland, *Adaptation in natural and artificial systems: An introductory analysis with applications to biology, control, and artificial intelligence (complex adaptive systems)*, A Bradford Book, 1992.

[14] N. Horbaczewski, *Drone racing league for speeding into the mainstream.*, Fast Company (2017), no. 213, 90 – 91.

[15] Jian Huang, Zhi-Hong Guan, Takayuki Matsuno, Toshio Fukuda, and Kosuke Sekiyama, *Sliding-mode velocity control of mobile-wheeled inverted-pendulum systems*, IEEE Transactions on Robotics **26** (2010), no. 4, 750–758.

[16] Rickey Dubay Jacob Wilson, Zachary Knopp, *Velocity control through predictive feedback of an inverted pendulum robot*, Proceedings of The Canadian Society for Mechanical Engineering International Congress 2016, 2016.

[17] Nader Jalili and Nejat Olgac, *Time-optimal/sliding mode control implementation for robust tracking of uncertain flexible structures*, Mechatronics **8** (1998), no. 2, 121 – 142.

[18] Fernando Jaramillo-Lopez, Godpromesse Kenne, and Francoise Lamnabhi-Lagarrigue, *Adaptive control for a class of uncertain nonlinear systems: Application to photovoltaic control systems*, IEEE Transactions on Automatic Control **62** (2017), no. 1, 393–398.

[19] Junjie Kang, Zheng H. Zhu, Wei Wang, Aijun Li, and Changqing Wang, *Fractional order sliding mode control for tethered satellite deployment with disturbances*, Advances in Space Research **59** (2017), no. 1, 263 – 273.

[20] Qudrat Khan, Aamer Iqbal Bhatti, Sohail Iqbal, and Mohammad Iqbal, *Dynamic integral sliding mode for MIMO uncertain nonlinear systems*, International Journal of Control, Automation and Systems **9** (2011), no. 1, 151–160.

[21] Sangtae Kim and SangJoo Kwon, *Dynamic modeling of a two-wheeled inverted pendulum balancing mobile robot*, International Journal of Control, Automation and Systems **13** (2015), no. 4, 926–933.

[22] H.K. Lam, S.H. Ling, F.H.F. Leung, and P.K.S. Tam, *Tuning of the structure and parameters of neural network using an improved genetic*

algorithm, 27th Annual Conference of the IEEE Industrial Electronics Society, IEEE, 2002.

[23] Jinkun Liu and Xinhua Wang, *Advanced sliding mode control*, Advanced Sliding Mode Control for Mechanical Systems, Springer Berlin Heidelberg, 2011, pp. 81–96.

[24] Xiaosong Liu, Zebiao Shan, and Yuanchun Li, *Dynamic boundary layer based neural network quasi-sliding mode control for soft touching down on asteroid*, Advances in Space Research **59** (2017), no. 8, 2173 – 2185.

[25] Yi Long, Zhi jiang Du, Wei dong Wang, and Wei Dong, *Robust sliding mode control based on GA optimization and CMAC compensation for lower limb exoskeleton*, Applied Bionics and Biomechanics **2016** (2016), 1–13.

[26] H. Márquez, *Nonlinear control systems: Analysis and design*, Wiley, 2003.

[27] N.S. Nise, *Control systems engineering, 7th edition*, Wiley, 2015.

[28] Imma Nuella, Cheng Cheng, and Min-Sen Chiu, *Adaptive PID controller design for nonlinear systems*, Industrial & Engineering Chemistry Research **48** (2009), no. 10, 4877–4883.

[29] Reza Olfati-Saber, *Cascade normal forms for underactuated mechanical systems*, Proceedings of the 39" IEEE Conference on Decision and Control, 2000.

[30] Samer Riachy, Yuri Orlov, Thierry Floquet, Raul Santiesteban, and Jean-Pierre Richard, *Second-order sliding mode control of underactuated mechanical systems i: Local stabilization with application to an inverted pendulum*, International Journal of Robust and Nonlinear Control **18** (2008), no. 4-5, 529–543.

[31] C.E. Rohrs, J.L. Melsa, and D.G. Schultz, *Linear control systems*, Electrical engineering series, McGraw-Hill, 1993.

[32] Anamika Jain S. Dereje, Mahantesh K. Pattanshetti and R. Mitra, *Genetic algorithm based integral sliding surface design and its application to stewart platform manipulator control*, INTERNATIONAL JOURNAL OF SYSTEMS APPLICATIONS, ENGINEERING & DEVELOPMENT **5** (2011), no. 4.

[33] Raul Santiesteban, Thierry Floquet, Yury Orlov, Samer Riachy, and Jean-Pierre Richard, *Second-order sliding mode control of underactuated mechanical systems II: Orbital stabilization of an inverted pendulum with application to swing up/balancing control*, International Journal of Robust and Nonlinear Control **18** (2008), no. 4-5, 544–556.

[34] Yuri B. Shtessel, Leonid Fridman, and Alan Zinober, *Higher order sliding modes*, International Journal of Robust and Nonlinear Control **18** (2008), no. 4-5, 381–384.

[35] Jean-Jacques E Slotine and Weiping Li, *Applied nonlinear control*, Pearson, Upper Saddle River, NJ, 1991, The book can be consulted by contacting: BE-ABP-CC3: Pfingstner, Juergen.

[36] Mark Spong, Seth Hutchinson, and Mathukumalli Vidyasagar, *Robot modeling and control*, John Wiley and Sons Ltd, 2005.

[37] Steven H. Strogatz, *Nonlinear dynamics and chaos*, The Perseus Books Group, 2000.

[38] Robert Sutton, G. N. Roberts, and Engineers Institution of Electrical, *Advances in unmanned marine vehicles.*, IET Control Engineering Series, no. Vol. 69, The Institution of Engineering and Technology, 2006.

[39] Sami ud Din, Qudrat Khan, Fazal ur Rehman, and Rini Akmeliawati, *Robust control of underactuated systems: Higher order integral sliding mode approach*, Mathematical Problems in Engineering **2016** (2016), 1–11.

[40] S USTUN and M DEMIRTAS, *Optimal tuning of PI coefficients by using fuzzy-genetic for v/f controlled induction motor*, Expert Systems with Applications **34** (2008), no. 4, 2714–2720.

[41] Jingxin Shi Vadim Utkin, Juergen Guldner, *Sliding mode control in electro-mechanical systems*, CRC Press, 2009.

[42] Rong-Jong Wai, Meng-An Kuo, and Jeng-Dao Lee, *Design of cascade adaptive fuzzy sliding-mode control for nonlinear two-axis inverted-*

pendulum servomechanism, IEEE Transactions on Fuzzy Systems **16** (2008), no. 5, 1232–1244.

[43] Hai Wang, Zhihong Man, Huifang Kong, Yong Zhao, Ming Yu, Zhenwei Cao, Jinchuan Zheng, and Manh Tuan Do, *Design and implementation of adaptive terminal sliding-mode control on a steer-by-wire equipped road vehicle*, IEEE Transactions on Industrial Electronics **63** (2016), no. 9, 5774–5785.

[44] Hanlei Wang, *Adaptive control of robot manipulators with uncertain kinematics and dynamics*, IEEE Transactions on Automatic Control **62** (2017), no. 2, 948–954.

[45] Frank Wicks, *Where on earth?.*, Mechanical Engineering **124** (2002), no. 7, 34.

[46] Rong Xu and mit zgner, *Sliding mode control of a class of underactuated systems*, Automatica **44** (2008), no. 1, 233 – 241.

[47] Xinghuo Yu and Man Zhihong, *Fast terminal sliding-mode control design for nonlinear dynamical systems*, IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications **49** (2002), no. 2, 261–264.

Appendix A

A.1 Inverted Pendulum Implementation Code

```
#include "mbed.h"
#include "Motor.h"
#include "QEI.h"
#include < cmath >

#define Vmax 1
#define EncM_ppr 5900
#define EncP_ppr 4000
#define radiopulley 0.035
#define Pi 3.1459
#define Pulses2metres 2*Pi*radiopulley/EncM_ppr
#define Pulses2Rad (2*Pi)/EncP_ppr
#define Volts2PWM 1/25
```

```
#define offsetPWM 0.01


// Cart States (meters)
float x = 0; // Cart Position (m)
float dx = 0; // Cart Velocity (m/s)
float x_prev = 0; // Previous cart position
// Rod States (radians)
float theta = 0; // rod angle (radians)
float dtheta = 0; // rod angular velocity (rads/sec)
float theta_prev = 0; // previous rod angle
// Time
float flag_start = 0; // start signal
float flag_stop = 0; // stop signal
float t = 0; // Time
float t_max = 10; // Maximum Run Time
float Ts = 0.01; // Sampling Time
// Cart Desired States
float xd = 0; // Desired Cart Position
float dxd = 0; // Desired Cart Velocity
// Rod Desired States
float thetad = 0; // Desired Rod Angle
float dthetad = 0; // Desired Rod Velocity
// Cart Tilde Variables
```

```
float xt = 0; // Cart Position Error
float dxt = 0; // Cart Velocity Error
// Rod Tilde Variables
float thetat = 0; // Rod Angle Error
float dthetat = 0;// Rod Ang Velocity Error
// Surface Variables
float s = 0; // Surface Value
float ds = 0; // Surface Derivative Value
float int_x_tilde = 0; // Integrated Cart position Error
float int_theta_tilde = 0; // Integrated Rod angle error
// Surface Constants
float alpha_a = 0.4; // cart velocity
float lambda_a = 0.1031; // cart position
float beta_a= 0.1; // integrated cart position

float alpha_u = 0.1754; // rod velocity
float lambda_u = 0.99; // rod position
float beta_u = 0.1; // integrated rod position
// Controller Parameters
float nu = 3; // Surface Convergence Gain
float epsilon = 0.1; // Surface Boundary Layer
// System Parameters
float M = 2.465; // Cart Mass (kg)
```

```
float m_rod = 0.113; // Rod Mass (kg)
float L = 0.15; // Rod Length (m)
float J = 0.000000989; // Rod Inertia (kg m)
float g = 9.81; // Gravity (kgm/s2̂)
float c1 = 1.2778; // Constant (N/V)
float c2 = 8.5; // Constant (kgm/s)
// System Dynamics
// Model Variables
float Maa;
float Mau;
float Muu;
float fa;
float fu;

float Maa_p;
float fa_p;
float Muu_p;
float fu_p;
float Ms;
float fs;
float d_sr;

float F; // model estimate function
```

```
float k_gain; // switching controller gain

float X_p=0;
float Theta_p=0;

// variables para el sistema de control
float sp_x=0;
float Xv=0;
float Thetav=0;

float alfa=0.95; // velocity smoothing factor

float u=0;
float umax=Vmax;

float tiempo=0;
float cont[1200];
float x_store[1200];
float theta_store[1200];
//float angulo[1200];
int k=0;
int kmax=0;
float Tfinal=10;
```

```
Motor m(p23, p6, p5); // pwm, fwd, rev
DigitalOut Ext_Pin(p12);
DigitalOut myled(LED1);

Serial pc(USBTX, USBRX);
QEI encoder1 (p29, p30, NC, EncM_ppr); // Cart Motor Encoder
QEI encoder2(p27, p28, NC, EncP_ppr); // Pendulum Angle Encoder

Ticker time_up;

float sat(float value)
{
float result;
if (value ≥ 1)
value= 1;
if (value ≤ -1)
value= -1;
result = value;
return result;
}

void iserv() {
```

```
X_p=encoder1.getPulses(); // read cart encoder value
x=-X_p*Pulses2metres; // convert to meters

Theta_p=encoder2.getPulses(); // read rod encoder value
theta=Theta_p*Pulses2Rad; // convert to radians

Xv=(x-x_prev)/Ts; // cart velocity
dx=alfa*dx+(1-alfa)*Xv; // smooth velocity value
x_prev=x; // update previous cart value

Thetav=(theta-theta_prev)/Ts; // rod velocity
dtheta=alfa*dtheta+(1-alfa)*Thetav; // smooth velocity value
theta_prev= theta; // update previous rod value

xd = 0; // Desired cart position
dxd = 0; // Desired cart velocity

thetad = 0; // Desired rod angle
dthetad = 0; // Desired rod ang vel

// Error Vectors
xt = x-xd; // x-tilde
```

```
dxt = dx-dxd; // x-dot tilde
thetat = theta-thetad; // theta-tilde
dthetat = dtheta-dthetad; // theta-dot tilde

int_x_tilde =+ xt*Ts; // cart position integral
int_theta_tilde =+ thetat*Ts; // rod angle integral

// System Dynamics
Maa = (M+m_rod);
Mau = m_rod*L*cos(theta);
Muu = (J+m_rod*pow(L,2));
fa = m_rod*L*(pow(dtheta,2))*sin(theta)-c2*dx;
fu = m_rod*g*L*sin(theta);

// Algebra Manipulations
Maa_p = Maa - Mau * (1.0/Muu) * Mau;
fa_p = fa - (Mau * (1.0/Muu) * fu);
Muu_p = Muu - (Mau*Mau)*(1.0/Maa);
fu_p = fu -(Mau*(1.0/Maa)*fu);
Ms = alpha_a * (1.0/Maa_p) - alpha_u * (1.0/Muu_p) * Mau*(1.0/Maa);
fs = alpha_a * (1.0/Maa_p) * fa_p + alpha_u * (1.0/Muu_p) * fu_p;
d_sr = lambda_a * dxt + lambda_u * dthetat;
// Sliding Surface
```

```
s = alpha_a * dxt + lambda_a * xt + alpha_u * dthetat + lambda_u * thetat
+ beta_a * int_x_tilde + beta_u * int_theta_tilde;
// Gain Bound
F = abs(fs); // model estimation function
// Controller Gain
k_gain = F+nu; // switching controller gain

if(flag_start==1)
{

if(tiempo¡Tfinal)
{
if(x<0.3 && x>-0.3) // cart position safety limits
{
u= (1.0/c1)*(1.0/Ms)*(fs + d_sr + k_gain*sat(s/epsilon)); // Control Action
}else u=0;

u=u*Volts2PWM; // get PWM signal
cont[k]=u; // store control value
x_store[k]=x; // store cart position
theta_store[k]=theta*180/Pi; // store rod angle in degrees
k=k+1; // increase counter
kmax=k;
```

```
if(u¿umax)u=umax; // check controller limits
if(u¡-umax)u=-umax;

tiempo=tiempo+Ts;
}else
{
u=0; // turn off control signal
Ext_Pin=0;
}

if(u¿0)u=u+offsetPWM; // apply voltage offset
if(u¡0)u=u-offsetPWM;

m.speed(u);
myled=!myled;
}
}

int main() {

Ext_Pin=1;
```

```
time_up.attach(&iserv, 0.01);
pc.baud (115200);

wait(0.5);
sp_x=0;

do{
if(Theta_p¿=2000) // start condition
{
encoder2.reset(); // reset rod angle to zero
Theta_p=0;
theta=0;
theta_prev=0;

flag_start=1; // start control code
}
}while(!flag_start);

while(true)
{
if (tiempo¿Tfinal)
{
for (k=0;k¡kmax;k++)
```

{ pc.printf(”wait(1000); } } }

Vita

Candidate's full name:
Lucas Allan Zoël Pupek

University attended:
University of New Brunswick: Bachelor of Science in Engineering, Mechanical (2016)

Publications:
Velocity and position trajectory tracking through sliding mode control of two-wheeled self-balancing mobile robot. 2018 Annual IEEE International Systems Conference (SysCon)

Conference Presentations:
SYSCON 2018: Annual IEEE International Systems Conference. Vancouver, BC, Canada.