Summer 2015

# Discrete-time neural network based state observer with neural network based control formulation for a class of systems with unmatched uncertainties

Jason Michael Stumfoll



Department:

# DISCRETE-TIME NEURAL NETWORK BASED STATE OBSERVER WITH NEURAL NETWORK BASED CONTROL FORMULATION FOR A CLASS OF SYSTEMS WITH UNMATCHED UNCERTAINTIES

by

JASON MICHAEL STUMFOLL

A THESIS

Presented to the Faculty of the Graduate School of the

MISSOURI UNIVERSITY OF SCIENCE AND TECHNOLOGY

In Partial Fulfillment of the Requirements for the Degree

MASTER OF SCIENCE IN AEROSPACE ENGINEERING

2015

Approved by

S. N. Balakrishnan, Advisor
Jagannathan Sarangapani
Joshua Rovey

## ABSTRACT

An observer is a dynamic system that estimates the state variables of another system using noisy measurements, either to estimate unmeasurable states, or to improve the accuracy of the state measurements. The Modified State Observer (MSO) is a technique that uses a standard observer structure modified to include a neural network to estimate system states as well as system uncertainty. It has been used in orbit uncertainty estimation and atmospheric reentry uncertainty estimation problems to correctly estimate unmodeled system dynamics. A form of the MSO has been used to control a nonlinear electrohydraulic system with parameter uncertainty using a simplified linear model. In this paper an extension of the MSO into discrete-time is developed using Lyapunov stability theory. Discrete-time systems are found in all digital hardware implementations, such as that found in a Martian rover, a quadcopter UAV, or digital flight control systems, and have the added benefit of reduced computation time compared to continuous systems. The derived adaptive update law guarantees stability of the error dynamics and boundedness of the neural network weights.

To prove the validity of the discrete-time MSO (DMSO) simulation studies are performed using a two wheeled inverted pendulum (TWIP) robot, an unstable nonlinear system with unmatched uncertainties. Using a linear model with parameter uncertainties, the DMSO is shown to correctly estimate the state of the system as well as the system uncertainty, providing state estimates orders of magnitude more accurate, and in periods of time up to 10 times faster than the Discrete Kalman Filter. The DMSO is implemented on an actual TWIP robot to further validate the performance and demonstrate the applicability to discrete-time systems found in many aerospace applications. Additionally, a new form of neural network control is developed to compensate for the unmatched uncertainties that exist in the TWIP system using a state variable as a virtual control input. It is shown that in all cases the neural network based control assists with the controller effectiveness, resulting in the most effective controller, performing on average 53.1% better than LQR control alone.

## ACKNOWLEDGMENTS

I want to acknowledge my advisor, Dr. S.N. Balakrishnan, for all of his support and guidance throughout this project. Without him this thesis would be nothing. I want to thank my committee, Dr. Jagannathan Sarangapani, and Dr. Joshua Rovey, for their excellent classes and research opportunities that were provided throughout my collegiate career. A big part of this goes to Avimanyu Sahoo, who proved invaluable to me during my difficulties with the math necessary for this thesis, and for that I am eternally grateful.

I also want to thank my parents, for their never ending support through my years in school. I would have never been able to make it to where I have without their guidance and will for me to succeed. Lastly, but not least, to the love of my life, for being there with me every step of the way.

# TABLE OF CONTENTS

# LIST OF ILLUSTRATIONS

# LIST OF TABLES

# 1. INTRODUCTION

## 1.1. TWO WHEELED INVERTED PENDULUM

The inverted pendulum on a cart is a standard problem for controls engineers to study found in many texts on control and nonlinear systems[1, 2]. The system consists of a pendulum at the end of a pole attached to a cart where the pendulum is allowed to move freely. The cart must move to control the momentum of the pendulum to keep it stabilized vertically, think of it like trying to balance a broom on your hand. The system is nonlinear and inherently unstable, and must be controlled properly to keep the pendulum stable, hence it's inclusion in many textbooks. It is a well-rounded problem encompassing several important areas of both nonlinear systems and controls engineering. The two wheeled inverted pendulum (TWIP) robot is a slight modification of this system where the four wheeled cart is reduced to two wheels and the pendulum becomes the body of the robot which must be controlled to stabilize the body vertically. There is little difference in the derivation of the equations of motion, but the two wheeled inverted pendulum has many applications where the four wheeled cart has few.

Two wheeled inverted pendulums gained much attention in 2001 with the announcement of the Segway, a personal transportation vehicle capable of speeds up to 12.5 mph. Two wheeled systems are capable of zero radius turning, and are able to get into many spaces that conventional carts are unable due to their much smaller footprint. There are lesser known TWIP robots marketed towards the medical community. The iBOT is a two wheeled balancing wheelchair that gives the user a higher sitting point to raise them to eye level. The iBOT has a smaller footprint than a traditional wheelchair, and allows the user to enter spaces normally restrictive to wheelchair users. The VGo is a virtual telepresence device that uses the TWIP design incorporating a camera, microphone, and computer screen balanced by two wheels that allows a user to interact with a remote environment. The VGo is marketed to several markets. In healthcare it is marketed as a patient monitoring aid, allowing healthcare staff to interact with patients when not physically in their location. The VGo is marketed towards the education market as well, allowing students with disabilities, or extended illnesses, the ability to still attend school in a physical way. The business market is also considered, the VGo provides an

easy method for remote training, and allows employees and executives to still stay involved and interact with the employees from home, or while away on business trips. Applications in industry are also possible. By placing various tooling on a movable cart a manufacturer may be able to reduce the number of required machines, or reduce the size of the assembly line. Several patents are available in relation to this idea, however the accuracy of moving tooling does not match that of a stationary machine, and is a current open topic of research [3].

A two wheeled inverted pendulum robot is used in this work for several reasons. First is the low cost, a robot can be made for only a few hundred dollars. The simplicity is a part as well. With a controller board, two DC motors, sensors, and associated electronics, the system is not overly complex. As a comparison to a quadcopter, lower cost components can be used as weight is not a huge factor, and the number of controls is greatly reduced. The failure mode of the controller is not catastrophic, as it can be for a quadcopter. If the controller on a quadcopter fails the system will crash, potentially damaging the system. In the case of the TWIP robot, it will simply fall over without harm. While the number of controls is reduced, the problem is still a difficult controls problem. Two wheeled inverted pendulum robots are underactuated, that is to say that they have less controls available than states to control, unstable nonlinear system. Two wheeled inverted pendulums are restricted by nonholonomic constraints and include unmatched uncertainties. This represents a difficult problem to obtain highly accurate control of the system, and is representative of a larger class of systems.

## 1.2. CONTRIBUTION OF THIS WORK

This paper proposes an extension of the Modified State Observer (MSO) into discrete time, called the Discrete Modified State Observer (DMSO). Lyapunov stability analysis is used to ensure boundedness of the neural network weights and state estimation error. Simulation studies are performed using a two wheeled inverted pendulum (TWIP) robot that show under the presence of system uncertainty, the DMSO is capable of highly accurate state estimation, as well as accurate estimation of the system uncertainty. The

DMSO is implemented in a physical system to control a TWIP robot to further show the applicability and success of the technique.

A new neural network based method of control is proposed for a class of systems with unmatched uncertainties, of which the TWIP is an example. Systems with unmatched uncertainties create problems for standard techniques as they cannot accurately compensate for the uncertainties. By using the tilt angle state as a virtual control and using two separate neural networks to estimate the uncertainties in the system, stability of the system is guaranteed in the presence of unmodeled dynamics, parameter uncertainty, and actuator nonlinearities. Lyapunov stability analysis is used to prove stability and boundedness of the tracking errors and neural network estimation errors. Simulation studies are performed to show that the control design is an accurate method of control for this class of systems.

# 2. LITERATURE REVIEW

## 2.1. TWO WHEELED INVERTED PENDULUM

Research on two wheeled inverted pendulums began in the 90's, researchers Ha and Yuta at the University of Tsukuba created the Yamabico Kurara and used a linear quadratic regulator control design for trajectory tracking control [4]. Shiroma et al. created a wheeled inverted pendulum and analyzed the stability in the presence of applied forces, using an observer to estimate the unknown force [5]. Ozaki et al. created a wheeled inverted pendulum system and implemented decoupling control by adding an additional state variable, thereby creating a square system and proposed algorithms to cope with the singularity problem caused by the feedback control [6]. Grasser et al. at the Industrial Electronics Laboratory at the Swiss Federal Institute of Technology created JOE, a mobile inverted pendulum. The researchers decoupled the dynamics of the robot and pole placement control design was used along with a filter to eliminate backlash effects of the DC motor [7]. Ooi at the University of Western Australia built a balancing robot to investigate the use of the Kalman filter for sensor fusion, implementing both a pole placement controller and a linear quadratic regulator controller [8]. Various forms of PID control design has been investigated by a number of different researchers [9-12]. Kim and Kwon developed a feedforward controller based on the State Dependent Riccati Equation (SDRE) [13].

The previous work was all done through linearization of the equations of motion, nonlinear control of two wheeled inverted pendulums is a current topic of research. Kim et al. investigated the exact dynamics of a two wheeled inverted pendulum robot, looking at the stability of the nonlinear system in a situation involving an inclined plane, and in turning motion [14]. Pathak et al. utilized the full nonlinear equations of motion based on the Euler-Lagrange method and implemented partial feedback linearization to control the velocity and position of a wheeled inverted pendulum system [15]. Askari et al. implemented a version of model predictive control and studied the performance under the presence of input disturbances [16]. Shibayama et al. implemented an observer-based robust controller to counteract unknown disturbances, allowing the use of a simplified system model [17]. Ha and Lee implemented sliding-mode control, improving the control

of the inverted pendulum over linear controllers when far from equilibrium [18]. Tsai and Ju implemented a backstepping sliding-mode for trajectory tracking and stabilization by first decoupling the kinematics and dynamics [19]. Do and Seet used partial feedback linearization combined with p-times differentiable saturation and backstepping techniques. The proposed controller has a large domain of attraction and allows simplicity of tuning control gains and implementation [20]. Rudra and Darai developed a robust adaptive backstepping technique that estimates system parameters and allows for stable tracking control. The controller removes the need for any prior knowledge of the system parameters [21]. Durdevic and Yang proposed a hybrid switching controller to overcome the backlash nonlinearity of DC motors and showed the effectiveness in experimental tests [22]. Yue, Wei, and Li developed an adaptive sliding-mode technique based on zero-dynamics theory, allowing for parameter uncertainties to be estimated, and robust control in the presence of nonlinearities [23].

Neural networks have been used by a number of researchers to control the two wheeled inverted pendulum system. Noh, Lee and Jung used radial basis function neural networks to control the unstable system, using the back-propagation learning algorithm to derive an online training law for the neural network [24, 25]. Tsai, Juang, and Lin proposed an adaptive control technique using radial basis function neural networks, the researchers used a backstepping technique and Lyapunov stability analysis to synthesize stable adaptive control laws [26]. Li and Yang formulated an output feedback adaptive neural network controller with a linear dynamic compensator and compared the results to a model based controller, showing that it outperforms the model based control when parametric uncertainties are included [27]. Li, Yang, and Fan wrote a book on the topic of advanced nonlinear control of wheeled inverted pendulum systems, dedicating an entire chapter to the neural network control of the systems using radial basis function neural networks [28]. Optimal control is so far a small area of research for these systems, however Gomez developed an optimal control strategy based on Control Adjointing Cell Mapping Reinforcement Learning (CACM-RL) and used a robot built from LEGO NXT components to test the capabilities of the technique [29].

## 2.2. MODIFIED STATE OBSERVER

Artificial neural networks have recently emerged as a highly capable area of adaptive control, gathering interest from researches across the globe. Modeled after the way neurons work in biological systems, they have proven to be a highly capable method of function approximation [30]. This function approximation capability has been manipulated into many unique methods of adaptive control by using neural networks to estimate nonlinear system uncertainty [31, 32]. Rajagopal et al. developed the Modified State Observer (MSO) concept as a new method for estimating nonlinear uncertainties by using a standard Leunberger observer structure in combination with a neural network [33]. The use of the MSO allows the designer to utilize large adaptive control gains without encountering the high frequency oscillations in the control signal that could excite unmodeled system dynamics.

Where originally applied to control, the MSO has been used simply as a method of state estimation as well. Harl et al. applied the MSO to an orbit uncertainty estimation problem, and also extended the method to include a reduced order formulation, for when the full state cannot be measured [34]. Darling et al. used the MSO and the reduced order MSO in an atmospheric reentry uncertainty estimation problem to estimate the uncertain aerodynamic acceleration of a piece of falling debris entering the atmosphere [35]. Darling et al. then developed the Sigma Point MSO extension, which uses sigma point filtering, similar in idea to the Unscented Kalman Filter, and applied it to the same atmospheric reentry problem [36]. This extension allows the use of nonlinear measurements, and incorporates a variable Kalman observer gain. Yang et al. formulated a model reference adaptive control design based on the MSO to control a nonlinear electrohydraulic system with parameter uncertainty using a simplified linear model [31]. Pappu et al. formulated an adaptive control law design based on the MSO and applied it to the Black Kite micro air vehicle [37].

All of this previous work on the MSO has been done in continuous time. In practice, implementation on digital systems requires extensive computations to integrate the dynamics over time. Discrete time implementations have the advantage of reduced computation time, in addition to being designed specifically for digital systems. Discrete time neural networks have also been a topic of study in adaptive control [32]. To a much

lesser extent they been used specifically as an observer. Salgado and Chairez [38] used recurrent neural networks and a method of offline training based on a least mean square method to create a discrete time neural observer. Alanis et al. [39] proposed a reduced order discrete time neural observer with a method of offline training using the Extended Kalman Filter (EKF) with high order recurrent neural networks. What these works lack is an online training law with guaranteed boundedness. Lewis et al. [40] developed a multi-layer discrete-time neural network observer that utilized online training. This work differs in the weight update law, the structure of the observer, and also does not suffer from having a large number of tunable parameters. Reducing the number of necessary neural networks allows for a simpler observer design and reduced time in tuning the system for the desired performance.

# 3. DISCETE TIME MODIFIED STATE OBSERVER

The Discrete Time Modified State Observer (DMSO) is an extension of the Modified State Observer as first described by Rajagopal et al. [33]. This section outlines the problem statement, the discrete time observer, and the Lyapunov proof of stability.

## 3.1. PROBLEM STATEMENT

Assume the dynamics of the system are given by

$$\mathbf{x}_{k+1} = F\mathbf{x}_k + G\mathbf{u}_k + B_{MSO}\mathbf{f}(\mathbf{x}_k) \tag{1}$$

where $\mathbf{f}(\mathbf{x})$ is an unknown nonlinear uncertainty vector of length *p*, where *p* is the number of states with uncertainty and $\mathbf{x}_k$ is the state vector at time *k*, a column vector of length *n*, where *n* is the number of states of the system. *F* is the discretized system dynamics matrix, or state transition matrix, of dimension $n \times n$. $\mathbf{u}_k$ is the system control vector of length *m*, where *m* is the number of control inputs, and *G* is the control input matrix, of dimension $n \times m$. $B_{MSO}$ is the uncertainty identification matrix of dimension $n \times p$. This matrix contains only zeros and ones, with the ones placed appropriately to identify the uncertain states. Measurements are available of the form

$$\mathbf{y}_k = \mathbf{x}_k \tag{2}$$

The discrete time observer for the system in Eqs. (1) and (2) is chosen as a modified Luenberger observer given by

$$\hat{\mathbf{x}}_{k+1} = F\hat{\mathbf{x}}_k + G\mathbf{u}_k + B_{MSO}\hat{\mathbf{f}}(\mathbf{x}_k) + K_{MSO}(\mathbf{y}_k - \hat{\mathbf{x}}_k) \tag{3}$$

where $\hat{\mathbf{x}}_k$ is the estimate of the system state, *K* is a user-selected constant gain matrix of dimension *n*×*n*, and $\hat{\mathbf{f}}(\mathbf{x}_k)$ is the estimate of the nonlinear system uncertainty $\mathbf{f}(\mathbf{x}_k)$, calculated using the single layer online neural network

$$\hat{\mathbf{f}}(\mathbf{x}_k) = \hat{W}_k^T \phi(\mathbf{x}_k) \tag{4}$$

$\hat{W}_k$ is the estimated neural network weight matrix of dimension *l*×*p* at timestep *k* and $\phi(\mathbf{x}_k)$ is a basis vector of length *l*, where *l* is the number of basis functions in the basis vector, and the number of weights in the neural network weight matrix. The universal function approximation property of neural networks states that any smooth function $\mathbf{f}(\mathbf{x}_k)$ can be approximated to an accuracy of $\epsilon$ using a two layer neural network [32]. Sadegh [30] extended this for a single layer neural network, with the added requirement that $\phi(\mathbf{x}_k)$ is selected as a basis, this can be expressed as the functional link neural network

$$f(\mathbf{x}_k) = W^T \phi(\mathbf{x}_k) + \epsilon_k \tag{5}$$

where *W* is the set of ideal weights and $\epsilon$ is a bounded positive constant that satisfies the condition $\| \epsilon_k \| \leq \epsilon_n$.

A weight update law is chosen to estimate the ideal weights as

$$\hat{W}_{k+1} = \hat{W}_k + \Gamma \phi(\mathbf{x}_k) e_{k+1}^T B_{MSO} \tag{6}$$

where $\Gamma$ is the neural network adaptation rate and $e_{k+1}$ is the state estimation error at timestep *k+1*.

### 3.2. LYAPUNOV STABILITY PROOF

In this section a Lyapunov stability proof is used to show boundedness and stability of the state estimate and the neural network weights.

Define the estimation error as

$$\mathbf{e}_k = \mathbf{x}_k - \hat{\mathbf{x}}_k \tag{7}$$

The estimation error dynamics are given by

$$\mathbf{e}_{k+1} = \mathbf{x}_{k+1} - \hat{\mathbf{x}}_{k+1} \tag{8}$$

Including the system dynamics from Eq. (3), Eq. (8) becomes

$$\mathbf{e}_{k+1} = F\mathbf{x}_k + G\mathbf{u}_k + B_{MSO}\mathbf{f}(\mathbf{x}_k) - (F\hat{\mathbf{x}}_k + G\mathbf{u}_k + B\hat{\mathbf{f}}(\mathbf{x}_k) + K_{MSO}(\mathbf{y}_k - \hat{\mathbf{x}}_k)) \tag{9}$$

Rearranging, including $\mathbf{y}_k = \mathbf{x}_k$, and the definition of $\mathbf{f}(\mathbf{x_k})$, Eq. (9) can be given by

$$\mathbf{e}_{k+1} = F(\mathbf{x}_k - \hat{\mathbf{x}}_k) - K_{MSO}(\mathbf{y}_k - \hat{\mathbf{x}}_k) + B_{MSO}(W^T\phi(\mathbf{x}_k) + \epsilon_k - \hat{W}_k^T\phi(\mathbf{x}_k)) \tag{10}$$

Which can be expressed as

$$\mathbf{e}_{k+1} = A\mathbf{e}_k + B_{MSO}\tilde{W}_k^T\phi(\mathbf{x}_k) + B_{MSO}\epsilon_k \tag{11}$$

where $A = F - K_{MSO}$. If $K_{MSO}$ is chosen to stabilize *A*, along with the boundedness of the neural network weight estimation error, $\tilde{W}_k = W - \hat{W}_k$, and the boundedness of the error of the neural network approximation, $\epsilon_n$, the error dynamics are stable and bounded. This will be confirmed using Lyapunov stability analysis.

Consider the neural network weight estimation error dynamics

$$\tilde{W}_{k+1} = W - \hat{W}_{k+1} \tag{12}$$

Including the weight update law from Eq. (6) and the error dynamics from Eq. (11), Eq. (12) becomes

$$\tilde{W}_{k+1} = \tilde{W}_k - \Gamma \phi(\mathbf{x}_k)\mathbf{e}_k{}^T A^T B_{MSO} - \Gamma \phi(\mathbf{x}_k)\phi^T(\mathbf{x}_k)\tilde{W}_k B_{MSO}^T B_{MSO} - \Gamma \phi(\mathbf{x}_k)\epsilon_k^T B_{MSO}{}^T B_{MSO} \tag{13}$$

Use the fact that BMSO is a semi orthogonal matrix that can be expressed as $B_{MSO}{}^T B_{MSO} = I$ [41], and rearrange to simplify Eq. (13).

$$\tilde{W}_{k+1} = \left[ I - \Gamma \phi(\mathbf{x}_k)\phi(\mathbf{x}_k)^T \right]\tilde{W}_k - \Gamma \phi(\mathbf{x}_k)(\mathbf{e}_k{}^T A^T B_{MSO} + \epsilon_k^T) \tag{14}$$

Now consider the Lyapunov function candidate:

$$V_k = \mathbf{e}_k^T \mathbf{e}_k + \frac{1}{\alpha} Tr(\tilde{W}_k{}^T \tilde{W}_k) \tag{15}$$

The first difference is given by

$$\Delta V = \mathbf{e}_{k+1}^T \mathbf{e}_{k+1} - \mathbf{e}_k^T \mathbf{e}_k + \frac{1}{\alpha} Tr(\tilde{W}_{k+1}{}^T \tilde{W}_{k+1}) - \frac{1}{\alpha} Tr(\tilde{W}_k{}^T \tilde{W}_k) \tag{16}$$

Including the error dynamics in Eq. (11) and the weight update from Eq. (14), Eq. (16) becomes

$$\begin{aligned}\Delta V = &(A\mathbf{e}_k + B_{MSO}\tilde{W}_k{}^T\phi(\mathbf{x}_k) + B_{MSO}\epsilon_k)^T(A\mathbf{e}_k + B_{MSO}\tilde{W}_k{}^T\phi(\mathbf{x}_k) + B_{MSO}\epsilon_k) - \mathbf{e}_k^T\mathbf{e}_k \\ &+ \frac{1}{\alpha}Tr\left\{\left(\left[I - \Gamma\phi(\mathbf{x}_k)\phi(\mathbf{x}_k)^T\right]\tilde{W}_k - \Gamma\phi(\mathbf{x}_k)(\mathbf{e}_k{}^T A^T B_{MSO} + \epsilon_k^T)\right)^T\right. \\ &\left.\times\left(\left[I - \Gamma\phi(\mathbf{x}_k)\phi(\mathbf{x}_k)^T\right]\tilde{W}_k - \Gamma\phi(\mathbf{x}_k)(\mathbf{e}_k{}^T A^T B_{MSO} + \epsilon_k^T)\right)\right\} \\ &- \frac{1}{\alpha}Tr(\tilde{W}_k^T\tilde{W}_k)\end{aligned} \tag{17}$$

By expanding a series of squared and cross product terms are obtained

$$\begin{aligned}\Delta V = &\left(A\mathbf{e}_k\right)^T\left(A\mathbf{e}_k\right) - \mathbf{e}_k^T\mathbf{e}_k + \left(B_{MSO}\tilde{W}_k{}^T\phi(\mathbf{x}_k)\right)^T\left(B_{MSO}\tilde{W}_k{}^T\phi(\mathbf{x}_k)\right) + \left(B_{MSO}\epsilon_k\right)^T\left(B_{MSO}\epsilon_k\right) \\ &+2\left(A\mathbf{e}_k\right)^T\left(B_{MSO}\tilde{W}_k{}^T\phi(\mathbf{x}_k)\right) + 2\left(A\mathbf{e}_k\right)^T\left(B_{MSO}\epsilon_k\right) + 2\left(B_{MSO}\tilde{W}_k{}^T\phi(\mathbf{x}_k)\right)^T\left(B_{MSO}\epsilon_k\right) \\ &+ \frac{1}{\alpha}Tr\left\{\left(\left[I - \Gamma\phi(\mathbf{x}_k)\phi(\mathbf{x}_k)^T\right]\tilde{W}_k\right)^T\left(\left[I - \Gamma\phi(\mathbf{x}_k)\phi(\mathbf{x}_k)^T\right]\tilde{W}_k\right)\right\} \\ &+ \frac{\Gamma^2}{\alpha}Tr\left\{\left(\phi(\mathbf{x}_k)(\mathbf{e}_k{}^T A^T B_{MSO} + \epsilon_k^T)\right)^T\left(\phi(\mathbf{x}_k)(\mathbf{e}_k{}^T A^T B_{MSO} + \epsilon_k^T)\right)\right. \\ &- \frac{2}{\alpha}Tr\left\{\left(\left[I - \Gamma\phi(\mathbf{x}_k)\phi(\mathbf{x}_k)^T\right]\tilde{W}_k\right)^T\left(\Gamma\phi(\mathbf{x}_k)(\mathbf{e}_k{}^T A^T B_{MSO} + \epsilon_k^T)\right)\right\} \\ &- \frac{1}{\alpha}Tr(\tilde{W}_k^T\tilde{W}_k)\end{aligned} \tag{18}$$

Expand further to obtain:

$$
\begin{aligned}
\Delta V = &\left(A\mathbf{e}_k\right)^T\left(A\mathbf{e}_k\right) - \mathbf{e}_k^T\mathbf{e}_k + \left(B_{MSO}\tilde{W}_k{}^T\phi(\mathbf{x}_k)\right)^T\left(B_{MSO}\tilde{W}_k{}^T\phi(\mathbf{x}_k)\right) + \left(B_{MSO}\epsilon_k\right)^T\left(B_{MSO}\epsilon_k\right) \\
&+2\left(A\mathbf{e}_k\right)^T\left(B_{MSO}\tilde{W}_k{}^T\phi(\mathbf{x}_k)\right) + 2\left(A\mathbf{e}_k\right)^T\left(B_{MSO}\epsilon_k\right) + 2\left(B_{MSO}\tilde{W}_k{}^T\phi(\mathbf{x}_k)\right)^T\left(B_{MSO}\epsilon_k\right) \\
&+\frac{1}{\alpha}Tr\left\{\left(\left[I-\Gamma\phi(\mathbf{x}_k)\phi(\mathbf{x}_k)^T\right]\tilde{W}_k\right)^T\left(\left[I-\Gamma\phi(\mathbf{x}_k)\phi(\mathbf{x}_k)^T\right]\tilde{W}_k\right)\right\} \\
&+\frac{\Gamma^2}{\alpha}Tr\left\{\left(\phi(\mathbf{x}_k)(\mathbf{e}_k{}^TA^TB_{MSO})\right)^T\left(\phi(\mathbf{x}_k)(\mathbf{e}_k{}^TA^TB_{MSO})\right)\right\} \\
&+\frac{\Gamma^2}{\alpha}Tr\left\{\left(\phi(\mathbf{x}_k)(\epsilon_k^T)\right)^T\left(\phi(\mathbf{x}_k)(\epsilon_k^T)\right)\right\} \\
&+\frac{2\Gamma^2}{\alpha}Tr\left\{\left(\phi(\mathbf{x}_k)(\mathbf{e}_k{}^TA^TB_{MSO})\right)^T\left(\phi(\mathbf{x}_k)(\epsilon_k^T)\right)\right\} \\
&-\frac{2}{\alpha}Tr\left\{\left(\left[I-\Gamma\phi(\mathbf{x}_k)\phi(\mathbf{x}_k)^T\right]\tilde{W}_k\right)^T\left(\Gamma\phi(\mathbf{x}_k)(\mathbf{e}_k{}^TA^TB_{MSO})\right)\right\} \\
&-\frac{2}{\alpha}Tr\left\{\left(\left[I-\Gamma\phi(\mathbf{x}_k)\phi(\mathbf{x}_k)^T\right]\tilde{W}_k\right)^T\left(\Gamma\phi(\mathbf{x}_k)(\epsilon_k^T)\right)\right\} \\
&-\frac{1}{\alpha}Tr(\tilde{W}_k{}^T\tilde{W}_k)
\end{aligned}
\tag{19}
$$

By using Young's inequality [42], $2a^Tb \le a^Ta + b^Tb$ , on the cross product terms the following expression is obtained.

$$
\begin{aligned}
\Delta V \le &\mathbf{e}_k^T\left(A^TA - I\right)\mathbf{e}_k + 2\left(A\mathbf{e}_k\right)^T\left(A\mathbf{e}_k\right) \\
&+3\left(B_{MSO}\tilde{W}_k{}^T\phi(\mathbf{x}_k)\right)^T\left(B_{MSO}\tilde{W}_k{}^T\phi(\mathbf{x}_k)\right) + 3\left(B_{MSO}\epsilon_k\right)^T\left(B_{MSO}\epsilon_k\right) \\
&+\frac{3}{\alpha}Tr\left\{\left(\left[I-\Gamma\phi(\mathbf{x}_k)\phi(\mathbf{x}_k)^T\right]\tilde{W}_k\right)^T\left(\left[I-\Gamma\phi(\mathbf{x}_k)\phi(\mathbf{x}_k)^T\right]\tilde{W}_k\right)\right\} \\
&+\frac{3\Gamma^2}{\alpha}Tr\left\{\left(\phi(\mathbf{x}_k)(\mathbf{e}_k{}^TA^TB_{MSO})\right)^T\left(\phi(\mathbf{x}_k)(\mathbf{e}_k{}^TA^TB_{MSO})\right)\right\} \\
&+\frac{3\Gamma^2}{\alpha}Tr\left\{\left(\phi(\mathbf{x}_k)(\epsilon_k^T)\right)^T\left(\phi(\mathbf{x}_k)(\epsilon_k^T)\right)\right\} \\
&-\frac{1}{\alpha}Tr(\tilde{W}_k{}^T\tilde{W}_k)
\end{aligned}
\tag{20}
$$

Using the solution to the discrete Lyapunov equation $-Q = A^T A - I$, the definition of the Frobenius norm [32], $Tr(x^T x) = \|x\|^2$, and applying norms to the rest of the terms Eq. (20) becomes

$$\begin{aligned}\Delta V \le & -\lambda_{min}(Q)\|\mathbf{e}_k\|^2 + 2\|A\mathbf{e}_k\|^2 + 3\|B_{MSO}\phi(\mathbf{x}_k)\tilde{W}_k\|^2 + 3\|B_{MSO}\epsilon_k\|^2 \\ & + \frac{3}{\alpha}\|(I - \Gamma\phi(\mathbf{x}_k)\phi(\mathbf{x}_k)^T)\tilde{W}_k\|^2 + \frac{3\Gamma^2}{\alpha}\|\phi(\mathbf{x}_k)(\mathbf{e}_k{}^T A^T B_{MSO})\|^2 \\ & + \frac{3\Gamma^2}{\alpha}\|\phi(\mathbf{x}_k)(\epsilon_k^T)\|^2 - \frac{1}{\alpha}\|\tilde{W}_k\|^2\end{aligned} \tag{21}$$

where $\lambda_{min}(Q)$ is the minimum eigenvalue of *Q*. Using the Triangle inequality [43], $\|x+y\| \le \|x\| + \|y\|$, and the property of the Frobenius norm, $\|AB\| \le \|A\|\|B\|$ to simplify.

$$\begin{aligned}\Delta V \le & -\lambda_{min}(Q)\|\mathbf{e}_k\|^2 + 2\|A\|^2\|\mathbf{e}_k\|^2 + 3\|B_{MSO}\|^2\|\phi(\mathbf{x}_k)\|^2\|\tilde{W}_k\|^2 + 3\|B_{MSO}\|^2\|\epsilon_k\|^2 \\ & + \frac{3}{\alpha}\left(1 - \Gamma\|\phi(\mathbf{x}_k\|^2\right)^2\|\tilde{W}_k\|^2 - \frac{1}{\alpha}\|\tilde{W}_k\|^2 + \frac{3\Gamma^2}{\alpha}\|\phi(\mathbf{x}_k)\|^2\|\epsilon_k\|^2 \\ & + \frac{3\Gamma^2}{\alpha}\|\phi(\mathbf{x}_k)\|^2\|A\|^2\|B_{MSO}\|^2\|\mathbf{e}_k\|^2\end{aligned} \tag{22}$$

This can be further simplified by using $\|B_{MSO}\|^2 = I$, $\|\phi(\mathbf{x}_k)\| \le \phi_{max}$, $\|\epsilon_k\| \le \epsilon_n$, and $\|A\| \le \lambda_{max}(A)$ where $\lambda_{max}(A)$ is the maximum eigenvalue of *A*

$$\begin{aligned}\Delta V \le & -\lambda_{min}(Q)\|\mathbf{e}_k\|^2 + 2\lambda_{max}(A)^2\|\mathbf{e}_k\|^2 + 3\phi_{max}^2\|\tilde{W}_k\|^2 + 3\epsilon_n{}^2 \\ & + \frac{3}{\alpha}\left(1 - 2\Gamma\phi_{max}^2 + \Gamma^2\phi_{max}^4\right)\|\tilde{W}_k\|^2 - \frac{1}{\alpha}\|\tilde{W}_k\|^2 + \frac{3\Gamma^2}{\alpha}\phi_{max}^2\epsilon_n{}^2 + \frac{3\Gamma^2}{\alpha}\phi_{max}^2\lambda_{max}(A)^2\|\mathbf{e}_k\|^2\end{aligned} \tag{23}$$

Rearrange to obtain the following.

$$\Delta V \le \left( -\lambda_{min}(Q) + 2\lambda_{max}(A)^2 + 3\frac{\Gamma^2}{\alpha}\phi_{max}{}^2\lambda_{max}(A)^2 \right) \left\| \mathbf{e}_k \right\|^2 + \left( \frac{2}{\alpha} - 6\frac{\Gamma}{\alpha}\phi_{max}{}^2 + 3\frac{\Gamma^2}{\alpha}\phi_{max}{}^4 + 3\phi_{max}{}^2 \right) \left\| \tilde{W}_k \right\|^2 + \left( 3\frac{\Gamma^2}{\alpha}\phi_{max}{}^2 + 3 \right) \epsilon_n{}^2 \tag{24}$$

With some bound on the estimation error, $\Delta V \le 0$ as long as the following two conditions are met.

$$-\lambda_{min}(Q) + 2\lambda_{max}(A)^2 + 3\frac{\Gamma^2}{\alpha}\phi_{max}{}^2\lambda_{max}(A)^2 \le 0 \tag{25}$$

and

$$\frac{2}{\alpha} - 6\frac{\Gamma}{\alpha}\phi_{max}{}^2 + 3\frac{\Gamma^2}{\alpha}\phi_{max}{}^4 + 3\phi_{max}{}^2 \le 0 \tag{26}$$

Choosing $\alpha = \frac{1}{3\phi_{max}{}^2}$ , $\Delta V \le 0$ as long as

$$\Gamma \le \frac{1}{\phi_{max}{}^2} \tag{27}$$

and

$$\lambda_{min}(Q) \ge 2\lambda_{max}(A)^2 + 9\Gamma^2\phi_{max}{}^4\lambda_{max}(A)^2 \tag{28}$$

The first condition is easily met, the second is a little more troublesome, however, the eigenvalues of both *A* and *Q* are user-selectable, and so this condition is not impossible to meet. This provides a bound on the estimation error as

$$\|\mathbf{e}_k\| \geq \sqrt{\frac{\left(9\Gamma^2 {\phi_{max}}^4 + 3\right){\epsilon_n}^2}{\lambda_{min}(Q) - 2\lambda_{max}(A)^2 - 9\Gamma^2 {\phi_{max}}^4 \lambda_{max}(A)^2}} \tag{29}$$

With these conditions met it is shown that $\Delta V \leq 0$ in a compact set. According to a standard Lyapunov extension theorem [32], this demonstrates that the estimation error is bounded for all $k \geq 0$. The neural network weights remain to be shown to be bounded.

The following definitions will be used to prove stability and boundedness of the neural network weights. Consider the system

$$x_{k+1} = F(k)x_k + G(k)u_k \tag{30}$$

Lemma 1: Define $\psi(k_1, k_0)$ as the state-transition matrix corresponding to $F(k)$ for the system in Eq. (30). If $\|\psi(k_1, k_0)\| \leq 1, \forall k_1, k_0 \geq 0$, the system is stable. For the proof see Sadegh [30].

Lemma 2: If $F(k) = I - \Gamma\phi(\mathbf{x}_k)\phi(\mathbf{x}_k)^T$ where $0 < \Gamma < 2$ and $\phi(\mathbf{x}_k)$ is a vector of basis functions, then $\|\psi(k_1, k_0)\| < 1$ is guaranteed if there is an $L > 0$ such that $\sum_{k=k_0}^{k_1+L-1} \phi(\mathbf{x}_k)\phi(\mathbf{x}_k)^T > 0$ for all *k*, then Lemma 1 guarantees stability of the system. For the proof see Sadegh [30]. Lemma 2 can be guaranteed with the Persistency Exciting (PE) condition [32, 40].

Definition 1: An input sequence $x(k)$ is said to be persistently exciting if there are $\lambda > 0$ and an integer $k_1 \geq 1$ such that

$$\lambda_{min}\left[\sum_{k=k_0}^{k_1+L-1} \phi(\mathbf{x}_k)\phi(\mathbf{x}_k)^T\right] > \lambda, \forall k_0 \geq 0 \tag{31}$$

where $\lambda_{min}(P)$ represents the smallest eigenvalue of *P*.

Now, consider the neural network weight estimation error dynamics

$$\tilde{W}_{k+1} = \left[ I - \Gamma \phi(\mathbf{x}_k)\phi(\mathbf{x}_k)^T \right]\tilde{W}_k - \Gamma \phi(\mathbf{x}_k)(\mathbf{e}_k{}^T A^T B_{MSO} + \epsilon^T) \tag{32}$$

when the Persistency of Excitation condition is met considering that both the estimation error and the neural network approximation error are bounded, the neural network weight estimation error, $\tilde{W}_k$, and equivalently the neural network weight estimate, $\hat{W}_k$, are shown to be stable and bounded [32, 44]. A bound on the neural network weight estimation error can also be found using Eq. (24) as

$$\left\|\tilde{W}_k\right\| \geq \sqrt{\frac{\left(9\Gamma^2 \phi_{max}{}^4 + 3\right)\epsilon_n{}^2}{\left(18\Gamma \phi_{max}{}^4 - 9\Gamma^2 \phi_{max}{}^6 - 9\phi_{max}{}^2\right)}} \tag{33}$$

# 4. TWO WHEELED INVERTED PENDULUM IMPLEMENTATION

This section outlines the two wheeled inverted pendulum robot that was created to test the DMSO in a real system. Parameter uncertainty exists in a real system, as well as sensor noise and unmodeled nonlinearities, so the Modified State Observer is a good choice for state estimation. Limitations in hardware limit the number of available computations, which makes the continuous-time MSO a poor choice due to the integration necessary for the calculation of the system states and weights. The discrete-time implementation is perfect for digital systems.

## 4.1. PHYSICAL DESIGN

Two wheeled inverted pendulum robots have two key aspects, independent motors control two wheels giving the robot a zero turning radius and high maneuverability, and it is designed in such a way that the center of gravity can be held above the wheels using proper control. The robot built for this implementation is shown in Figure 4.1. Two plastic mounting plates are held apart with steel threaded rod. The motors are mounted on the bottom of the bottom plate, with the majority of the electronics on the top of the bottom plate. To raise the center of gravity, the battery, a high discharge 11.1V 20C Lithium Polymer battery, is mounted on the top plate. Using steel threaded rods allows for easy adjustment of the center of gravity. Stiffening rods were added to the bottom mounting plate. The motors are heavy in comparison to the plastic plate's stiffness which resulted in undesirable bending of the mounting plate before the stiffening rods were added.

The mounting of the motors places the wheel axis 17.5mm below the bottom mounting plate. The wheels are plastic Pololu 90mm diameter wheels with silicone traction tires and steel washers attached for added weight to give the wheels a measured mass of .117kg. The two mounting plates are held 22cm apart, and the center of gravity is measured as 7.5cm above the wheel axis. The total mass of the robot is measured as 1.657kg. The moment of inertia of the wheels are calculated by assuming the wheels are solid disks and using the standard formula for moment of inertia of a solid disk.

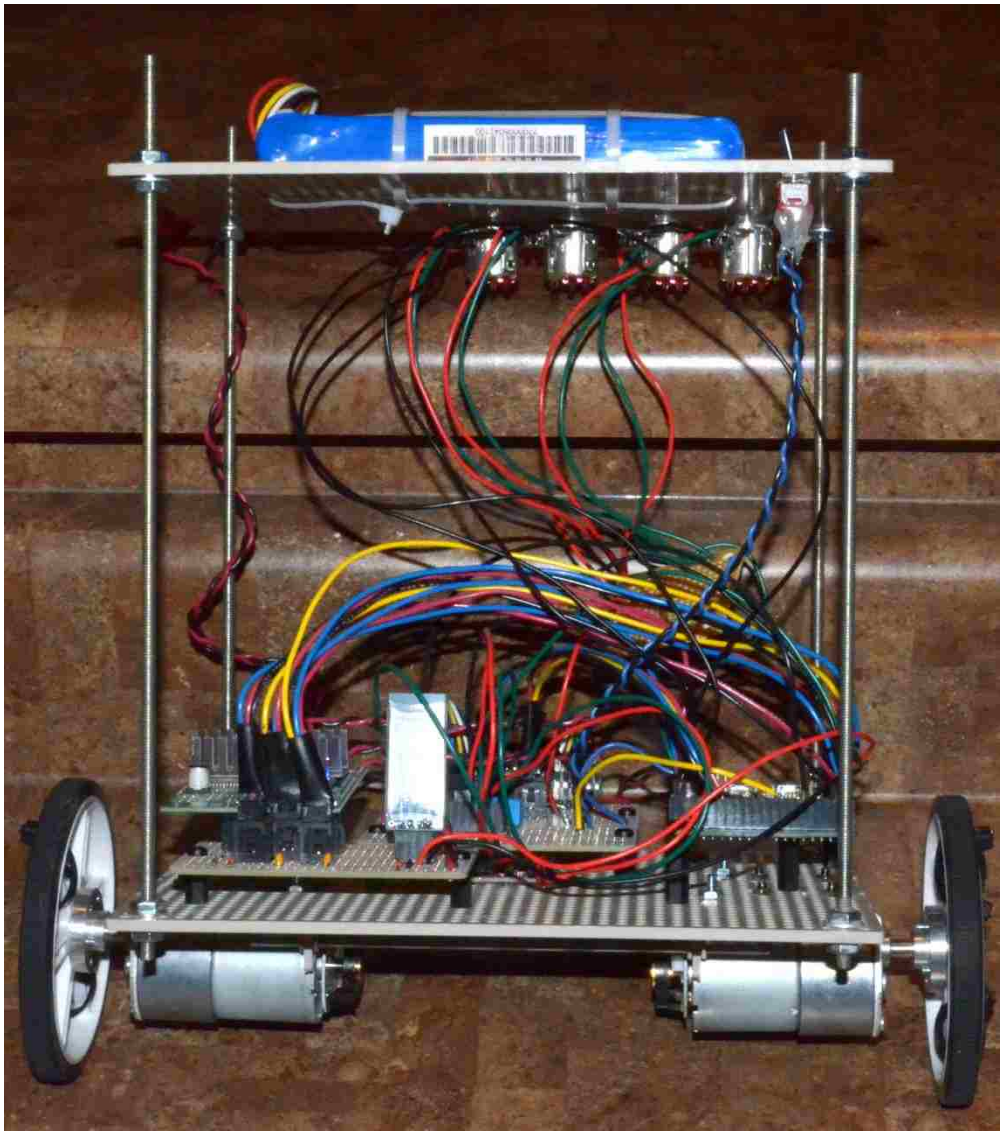

Figure 4.1. Two Wheeled Inverted Pendulum Robot

The moment of inertia of the robot's pendulum about the wheel axis is calculated by measuring the mass of the two plates separately, calculating the moment of inertia of two flat plates, and using the parallel axis theorem to get the moment of inertia of the pendulum about the wheel axis. The robot's physical parameters are summarized in Table 4.1.

Table 4.1. TWIP Robot Physical Parameters

| Parameter | Value | Units |
|---|---|---|
| Pendulum Moment of Inertia | 0.025 | $kgm^2$ |
| Wheel Moment of Inertia | 1.441e-4 | $kgm^2$ |
| Distance to CG from Wheel Axis | 0.075 | m |
| Mass of Pendulum | 1.432 | kg |
| Mass of Wheels | 0.1168 | kg |
| Wheel Radius | 0.045 | m |

## 4.2. ELECTRONIC DESIGN

Electronics are a major component of any robot, and the two wheeled inverted pendulum robot is no exception. The robot consists of a main computer, two motors, various sensors, and power and communication devices.

**4.2.1. Computer.** The robot is controlled using an Arduino Due microcontroller board based on the Atmel SAM3X9E ARM Cortex-M3, a 32-bit, 84MHz CPU. The Due has 96KB of onboard SRAM and 512KB available RAM for programming. The Due has 54 digital input/output channels, 12 of which can be set to output pulse width modulation (PWM) signals, 12 analog inputs, and 2 analog outputs. The Arduino Due supports serial communication over USB. The microcontroller is programmed in C using the Arduino interface. The Arduino is a popular hobbyist electronics platform, and a large range integrated electronic shields can be purchased to work directly with the Arduino platform to expand the capabilities of the Arduino microcontroller. Helpful information can also be obtained online for interfacing most electronics with an Arduino.

**4.2.2. Motors.** Two Pololu 12V 37mm DC motors are used to control the two wheeled inverted pendulum robot. Pololu is a Las Vegas located manufacturer and online retailer of robotic electronics founded by three students of the Massachusetts Institute of Technology. The motors have a 30:1 gear ratio gearbox and 64 counts per revolution (CPR) integrated Hall Effect quadrature encoders to measure the position of the output shaft. The motors are characterized by their free-run speed and current, and stall torque

and current. From these characteristics the torque motor constant and the back electromotive force constant can be obtained. The motor nominal resistance can also be measured by directly measuring the resistance of the output terminals across several positions of the output shaft and averaging the values. The physical motor constants are summarized in Table 4.2. The motors are controlled using a Pololu Dual VNH5019 Motor Driver Arduino Shield. The VNH5019 motor driver chip allows DC motor operation between 5.5 and 24 V at currents up to 12 A. The chips provide current sensing capabilities, and PWM operation at ultrasonic speeds up to 20KHz to allow for quieter motor operation. The Pololu designed Arduino shield is designed specifically to work directly with the Arduino platform, and a provided library allows easy operation of two individual motors through several implemented functions.

Table 4.2. DC Motor Physical Parameters

| Parameter | Value | Units |
|---|---|---|
| Free-Run Speed | 350 | RPM |
| Free-Run Current | 300 | mA |
| Stall Torque | 110 | oz-in |
| Stall Current | 5000 | mA |
| Torque Motor Constant | 0.11541 | Nm/A |
| Back EMF Constant | 0.00361 | Vs/rad |
| Nominal Resistance | 2.5 | Ω |

**4.2.3. Sensors.** Sensors are necessary in any robot in order to measure the state of the system. The state must be known in order to properly control the system. There are many different ways that the states of the system can be obtained. In this system the measurements are obtained by three devices, an integrated inertial measurement unit (IMU), and Hall Effect digital encoders integrated on each motor.

**4.2.3.1 Inertial measurement unit.** Integrated System in Package (SiP) microchips are quickly gaining hold in many areas of consumer electronics due to their smaller size and lower cost of production. The InvenSense MPU-9150 is a low cost, low power SiP motion tracking device marketed for consumer electronics equipment such as smartphones and tablets. The SiP includes a 3-axis gyroscope, a 3-axis accelerometer, a 3-axis magnetometer, and an onboard Digital Motion Processor. The Digital Motion Processor (DMP) is used to offload complex algorithms from the main processor, and is mainly geared towards typical uses of an IMU in consumer electronics.

In the two wheeled inverted pendulum robot the SparkFun MPU-9150 breakout board was used. The breakout board simply places the surface mount MPU-9150 on an easy to use circuit board with the communication pins available to solder directly to wires. The full capabilities of the MPU-9150 including the DMP were not utilized as they were not necessary. Only the direct readings from the accelerometer and gyroscope were used to obtain the necessary states of the robot.

The gyroscope has a digital output of 16 bits and user-selectable ranges of output from ±250 °/s to ±2000 °/s. The lowest range of ±250 °/s is selected for use as it has the highest sensitivity of 131 LSB/°/s. This translates to an accuracy of exactly 1/131 °/s or approximately 0.00763°/s. Noise is given by two parameters, the total RMS noise at a rate of 92Hz is 0.06 °/s-rms, and the noise spectral density at 10Hz is 0.005 °/s/√Hz. The noise has been better characterized by a careful test and analysis of the actual sensor, to be described in Section 7.4. These parameters are summarized in Table 4.3.

The accelerometer also has a digital output of 16 bits, and gives readings in multiples of gravity, or g's. Calibration of the accelerometer was required to obtain a 1g reading at rest. At the highest accuracy, minimum range scale of ±2g, the sensor has a sensitivity of 16,384 LSB/g. This translates to an exact accuracy of 1/16384 g, or approximately 59.4 µg. Noise is given by the same two parameters, but this time at different rates, the total RMS noise at a rate of 100Hz is 4mg-rms, and the power spectral density at a rate of 10Hz is 400 µg/√Hz. These differences in rates and the parameters used for the noise tests necessitates a careful analysis of the actual noise from the sensor used in the implementation to ensure proper simulation. The accelerometer specifications are summarized in Table 4.4.

Table 4.3. Gyroscope Specifications

| **Parameter** | **Value** | **Units** |
|---|---|---|
| Range | ±250 | °/s |
| Sensitivity | 131 | LSB/°/s |
| Accuracy | 0.00763 | °/s |
| Total RMS Noise | 0.06 | °/s-rms |
| Power Spectral Density | 0.005 | °/s/√Hz |

Table 4.4. Accelerometer Specifications

| **Parameter** | **Value** | **Units** |
|---|---|---|
| Range | ±2 | g |
| Sensitivity | 16384 | LSB/g |
| Accuracy | 59.4 | µg |
| Total RMS Noise | 4 | mg-rms |
| Power Spectral Density | 400 | µg/√Hz |

**4.2.3.2 Digital encoders.** Each Pololu metal DC motor has an integrated Hall Effect quadrature encoder used to measure the relative position of the motor output shaft. The motor output shaft is connected to a magnetic disk with 16 magnetic divisions. Two Hall Effect sensors sense the changing magnetic field as the output shaft rotates. By counting both the rising and falling edge of both signals a total resolution of 64 counts per revolution of the motor output shaft is obtained, this is where the term quadrature encoder originates. In combination with the 30:1 gearbox, the output of the gearbox has a measurable accuracy of 1920 counts per revolution. With 90mm diameter wheels, the encoder measures discrete steps equal to 0.000147m.

**4.2.4. Miscellaneous Electronics.** Four potentiometers are connected to the Arduino Due to allow for easy adjustment of parameters, in this case the controller gains, and a switch is used to turn the robot controller on and off. Miscellaneous electronics necessary to convert voltages for communication with the Arduino Due are included, as well as a HC-05 Bluetooth module for wireless transmitting of serial output data. The robot is powered by a single high discharge 11.1V 20C Lithium Polymer battery.

## 4.3. PROGRAMMING

On initialization of the robot the various pins and connections are setup allowing for communication between the sensors, switches, Bluetooth module, and the microcontroller. The motor controller is initialized, and the IMU is enabled. On initialization of the IMU the constant bias is removed from the sensor measurements by averaging the first 100 measurements and removing the calculated value from the sensor reading. As a result the robot must be stationary while initializing or the constant bias value will be inaccurate. Until the on-switch is flipped, the robot sits in a loop that takes measurements and readings from the potentiometers to allow adjustment of the controller gains.

When the on-switch is flipped the robot begins the control loop. First the measurements are obtained, a simple filter is used to correct the tilt angle measurement. Details are provided in the next section, and the DMSO is then used to estimate the full state of the robot. A C based matrix library was implemented to allow simplified matrix calculations on the robot, similar to how Matlab computes. Due to computational limitations, the estimates from the DMSO are used in a linear quadratic regulator (LQR) controller to calculate the output to the motor driver. Important data is then transmitted over serial Bluetooth for later analysis.

**4.3.1. Measurements.** The inverted pendulum tilt angle can be obtained directly using a tiltometer, but this would require having an additional sensor on the robot. The tilt angle can also be obtained by integrating the gyroscope signal, but this suffers from two issues, first is that the initial angle must be known precisely or the integration will have a bias. This is an issue for the unstable inverted pendulum as even a very small

bias causes instability of the robot as it tries to stabilize at a statically stable point. The second issue is called gyroscope drift and describes the tendency of an integrated gyroscope signal to drift from the correct value over time. To overcome these issues the tilt angle is measured using the accelerometer and filtered in combination with the gyroscope measurements to obtain a more accurate measurement of the tilt angle.

The orientation of an accelerometer can be defined by the roll, $\phi$ , pitch, $\theta$, and yaw, $\psi$ , rotations from an initial position. By defining the initial position of the Earth's gravitational field vector in the positive z direction and applying rotation matrices, a set of equations can be obtained transforming the gravitational field vector into roll, pitch, and yaw angles. The aerospace rotation sequence, $\mathbf{R}_{xyz}$, is commonly used in the aerospace industry and results in the following relationship [45]

$$\mathbf{R}_{xyz}\begin{bmatrix}0\\0\\1\end{bmatrix}=\begin{bmatrix}\cos\theta\cos\psi & \cos\theta\sin\psi & -\sin\theta\\ \cos\psi\sin\theta\sin\phi-\cos\phi\sin\psi & \cos\phi\cos\psi+\sin\theta\sin\phi\sin\psi & \cos\theta\sin\phi\\ \cos\phi\cos\psi\sin\theta+\sin\phi\sin\psi & \cos\phi\sin\theta\sin\psi-\cos\psi\sin\phi & \cos\theta\cos\phi\end{bmatrix}\begin{bmatrix}0\\0\\1\end{bmatrix}$$
$$\mathbf{R}_{xyz}\begin{bmatrix}0\\0\\1\end{bmatrix}=\begin{bmatrix}-\sin\theta\\ \cos\theta\sin\phi\\ \cos\theta\cos\phi\end{bmatrix} \tag{34}$$

When rewritten using an arbitrary gravitational field vector, $G_p$, Eq. (34) can be solved for the pitch and roll angles as follows.

$$\frac{G_P}{\|G_P\|}=\frac{1}{\sqrt{a_x^2+a_y^2+a_z^2}}\begin{bmatrix}a_x\\a_y\\a_z\end{bmatrix}=\begin{bmatrix}-\sin\theta\\ \cos\theta\sin\phi\\ \cos\theta\cos\phi\end{bmatrix}$$
$$\tan\phi=\frac{a_y}{a_z} \tag{35}$$
$$\tan\theta=\frac{-a_x}{\sqrt{a_y^2+a_z^2}}$$

The above relationship for the pitch angle is only one possible way to calculate the pitch from the accelerometer. Rotation matrices do not compute, but as only the pitch angle is of concern a different rotation sequence can be utilized to obtain the same pitch angle measurement. Computations on the microcontroller must be minimized wherever possible, so by using the rotation matrix $\mathbf{R}_{yxz}$ in the same manner, an easier to compute relationship for the tilt angle is obtained.

$$\theta_a = \tan^{-1}\frac{-a_x}{a_z} \tag{36}$$

This calculation assumes two things, first is that the initial orientation of the accelerometer is aligned with the gravitation field vector in the z-axis and second is that there are no external accelerations. Clearly when moving the robot will have external accelerations, so filtering of this measurement is necessary. This is easily remedied by using a filter called the complementary filter [46]. The same task can be done with the Kalman filter, but due to the computational limitations on the microcontroller the complementary filter is a good choice. The filter combines measurements from the accelerometer and the gyroscope to obtain a more accurate estimate of the angle using a simplified high pass and low pass filter. The form implemented on the robot is

$$\theta_{k+1} = \alpha(\theta_k + g_y \Delta t) + (1-\alpha)\theta_a \tag{37}$$

where α is a design parameter set to 0.99, $g_y$ is the y-axis gyroscope reading, $\theta_a$ is the accelerometer tilt angle measurement, and $\theta_k$ is the complementary filter angle estimate. The complementary filter is extremely easy to implement and is perfect for this implementation where computations are limited.

The tilt angle is the only measurement that has to be pre-filtered before the DMSO or Kalman filter are used to estimate the full state. The gyroscope measurements are used directly to measure the tilt rate. The motor encoder outputs are processed using hardware interrupts in the code. Each time the Hall Effect sensor signal switches from

positive to negative the interrupt is activated to either add to or subtract from the motor encoder position. The velocity measurements are obtained by taking the derivative of the position readings.

### 4.4. DIFFICULTIES

During the assembly and programming of the robot many difficulties were encountered. Initially the motors were a huge problem, and were eventually replaced. The initial motors chosen were 6V DC motors with a lower torque output. The original motors had an enormous deadzone nonlinearity, and a correspondingly small range of controllable output levels. Simulation studies showed that the best control possible would be large stable oscillations. Noise in the sensors created instability that made this control impossible to achieve. It was during this time that the potentiometers were added in order to quickly adjust the controller gains in an attempt to stabilize the robot, rather than reprogramming the microcontroller each time the gains were to be adjusted.

The motors were replaced with larger 12V motors that do not have the same large deadzone issue, and allowed a larger range of output to be used for control. This created a problem with the plastic mounting plates as the new motors were so heavy the plates twisted under the weight. Stiffening rods had to be added to the underside of the robot to compensate. This also pointed out an initial design flaw; plastic rods were used to separate the two mounting plates. There was a lot of flexibility in the system which allowed vibrations to create a problem in the sensor measurements. Steel threaded rods were used as a replacement, which worked nicely to stiffen the system.

As mentioned previously, the robot is limited in the number of computations available, so wherever possible computations were limited on the robot. The serial output was found to be one of the largest uses of computation power. Serial data, essential for evaluation of the performance, had to be carefully selected and limited in order to keep the loop time at an acceptable level. The minimum loop execution time that could be obtained was 0.01s.

The LQR gains had to be modified from the ideal values calculated from the parameters of the robot. This is indicative of inaccurately measured parameters. The

model is also not entirely accurate to the true robot system. The pendulum is modeled as a point mass attached by a thin beam to the motor axis, however the mounting plates create more complex interactions than this. Once tuned properly stability of the robot was obtained.

# 5. THE BALANCING ROBOT MODEL

This section describes the two wheeled inverted pendulum robot model that is used as the motivating example to show the effectiveness of the Discrete Modified State Observer. There are many forms [7, 8, 13, 14] that these equations can take depending on the method used for derivation and the different models taken into consideration. The model below follows from Newton's method when including a linear DC motor model [8].

## 5.1. MODEL OF A DIRECT CURRENT (DC) MOTOR

The following section describes a DC motor in mathematical terms and places the system in a space-space form. The model derived describes a relationship between the output torque of the motor and the input voltage. Figure 5.1 shows the electromechanical diagram used to describe a linear DC motor when neglecting the higher dynamics associated with the stator, or field, circuit of a DC motor [47].

Figure 5.1. DC Motor Model

When a voltage is applied to a DC motor the motor itself also induces a voltage called the back electromotive force (EMF), $V_e$, which can be described by a linear relationship between the voltage and the output speed of the motor, $\omega$. The constant of

proportionality in this case is called the back EMF constant, $k_e$, with units of $\frac{Vs}{rad}$. This relationship is described by the equation

$$V_e = k_e \omega \tag{38}$$

Using Kirchoff's Voltage Law, which states that the sum of all voltages in a circuit must equal zero, on the circuit in Figure 5.1 and using the relationship in Eq. (38) the following dynamic equation is found

$$V_a - Ri - L\frac{di}{dt} - V_e = 0 \tag{39}$$

Substitute Eq. (38) into Eq. (39) and rearrange to obtain the first equation of motion.

$$\frac{di}{dt} = \frac{V_a}{L} - \frac{R}{L}i - \frac{k_e}{L}\omega_e \tag{40}$$

When a voltage, $V_a$, is applied to the DC motor a current, $i$, is induced in the armature, or rotor, circuit. DC motors produce a torque, $\tau_m$, proportional to the current in the circuit. The constant of proportionality is called the torque constant with units of $\frac{Nm}{A}$, this relationship is defined as

$$\tau_m = k_m i \tag{41}$$

The friction on the shaft of the DC motor can be approximated by a linear relationship between the output speed, $\omega$, and the frictional torque, $\tau_f$, by a frictional torque constant, $k_f$, with units of $\frac{Nm}{rad}$. This is described by the relationship

$$\tau_f = k_f \omega \tag{42}$$

Using Newton's second law and summing moments about the motor output shaft the following dynamic equation is found

$$\sum M = \tau_m - \tau_f - \tau_a = I_m \dot{\omega} \tag{43}$$

where $\tau_a$ is an applied load and $I_m$ is the moment of inertia of the applied load. Substituting Eqs. (41) and (42) into Eq. (43) and rearranging the second equation of motion is found.

$$\frac{d\omega}{dt} = \frac{k_m}{I_m} i - \frac{k_f}{I_m} \omega - \frac{\tau_a}{I_m} \tag{44}$$

Equations (40) and (44) describe the fundamental equations of a DC motor. They describe a set of first order differential equations between the circuit current, motor speed, and applied torque. In the case of a balancing robot, a simplified model provides acceptable performance. Assuming motor inductance, $L$ , and friction, $k_f$ ,are negligible, the model can be simplified to

$$i = \frac{V_a}{R} - \frac{k_e}{R} \omega_e \tag{45}$$

and

$$\frac{d\omega}{dt} = \frac{k_m}{I_m} i - \frac{\tau_a}{I_m} \tag{46}$$

Substituting Eq. (45) into Eq. (46) a model is obtained that does not include the circuit current. This is desirable as the current is not directly controllable in the physical system, and will add no benefit as an additional state in the equations of motion.

$$\frac{d\omega}{dt} = -\frac{k_m k_e}{I_m R}\omega + \frac{k_m}{I_m R}V - \frac{\tau_a}{I_m} \tag{47}$$

By neglecting the inductance of the motor circuit the current will reach a steady state value immediately, as compared to the motor speed which will take a period of time to reach steady state. Eq. (47) can be placed in state-space form as shown below.

$$\frac{d}{dt}\begin{bmatrix}\theta \\ \omega\end{bmatrix} = \begin{bmatrix}0 & 1 \\ 0 & -\frac{k_m k_e}{I_m R}\end{bmatrix}\begin{bmatrix}\theta \\ \omega\end{bmatrix} + \begin{bmatrix}0 & 0 \\ \frac{k_m}{I_m R} & -\frac{1}{I_m}\end{bmatrix}\begin{bmatrix}V_a \\ \tau_a\end{bmatrix} \tag{48}$$

## 5.2. TWO WHEELED INVERTED PENDULUM DYNAMIC MODEL

The two wheeled inverted pendulum (TWIP) shares many similarities to the pendulum on a cart problem. The motion is very similar, but the dynamics are more complex. The two wheeled inverted pendulum analysis begins with the two wheels. The pendulum is analyzed separately, and the equations are combined to synthesize two dynamic equations describing the motion of the system.

The free body diagram of the left wheel is shown in Figure 5.2. The reaction forces between the wheel and the pendulum are $P_L$ and $H_L$. The torque applied on the wheel by the DC motor is $C_L$, and the frictional force between the ground and the wheel is $H_{fL}$.

The equations for the left and right wheel are completely analogous, so only the left wheel will be analyzed. The wheels are assumed to always stay in contact with the ground with no slip. Sum forces in the x direction to obtain the following.

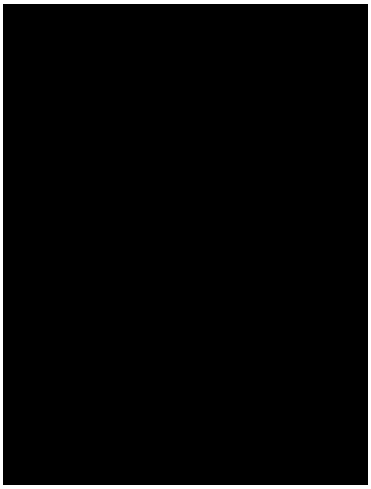

Figure 5.2. Left Wheel Free Body Diagram

$$M_w \ddot{x} = H_{fL} - H_L \tag{49}$$

Summing moments around the center of the wheel yields

$$I_w \ddot{\theta}_w = C_L - H_{fL} r \tag{50}$$

where $I_w$ is the moment of inertia of the wheel and $\ddot{\theta}_w$ is the angular acceleration of the wheel. Start with the DC motor dynamics Eq. (43) and substitute Eq. (47) to obtain

$$C_L = C_R = \tau_m = -\frac{k_m k_e}{R} + \frac{k_m}{R} V_a \tag{51}$$

Use Eq. (51) in Eq. (50) and rearrange to obtain an expression for the friction reaction force.

$$H_{fL} = -\frac{k_m k_e}{Rr} + \frac{k_m}{Rr} V_a - \frac{I_w}{r} \ddot{\theta}_w \tag{52}$$

Substitute Eq. (52) into Eq. (49) to obtain the equation of motion for the left wheel.

$$M_w \ddot{x} = -\frac{k_m k_e}{Rr}\dot{\theta}_w + \frac{k_m}{Rr}V - \frac{I_w}{r}\ddot{\theta}_w - H_L \tag{53}$$

A similar equation can be derived for the right wheel.

$$M_w \ddot{x} = -\frac{k_m k_e}{Rr}\dot{\theta}_w + \frac{k_m}{Rr}V - \frac{I_w}{r}\ddot{\theta}_w - H_R \tag{54}$$

A transformation is used to translate the rotational motion into linear motion.

$$\begin{aligned} \ddot{\theta}_w &= \frac{\ddot{x}}{r} \\ \dot{\theta}_w &= \frac{\dot{x}}{r} \end{aligned} \tag{55}$$

Thus Eq. (53) for the left wheel becomes

$$M_w \ddot{x} = -\frac{k_m k_e}{Rr^2}\dot{x} + \frac{k_m}{Rr}V - \frac{I_w}{r^2}\ddot{x} - H_L \tag{56}$$

and Eq. (54) for the right wheel becomes

$$M_w \ddot{x} = -\frac{k_m k_e}{Rr^2}\dot{x} + \frac{k_m}{Rr}V - \frac{I_w}{r^2}\ddot{x} - H_R \tag{57}$$

Combine Eqs. (56) and (57) to obtain an equation of motion for the combined wheel system.

$$2\left(M_w + \frac{I_w}{r^2}\right)\ddot{x} = -\frac{2k_m k_e}{Rr^2}\dot{x} + \frac{2k_m}{Rr}V - \left(H_L + H_R\right) \tag{58}$$

Figure 5.3 shows the free body diagram of the inverted pendulum. Two reference frames are used in the analysis. An inertial frame is connected to the base of the pendulum, and a translating reference frame at the center of gravity of the pendulum is used to simplify the analysis. $l$ is the distance between the base of the pendulum and the center of gravity, $\alpha, \omega, \theta$ are the angular acceleration of the pendulum, the angular velocity of the pendulum, and the angular position of the pendulum respectively. $x, \dot{x}, \ddot{x}$ are the position, velocity, and acceleration of the base of the pendulum, or the pendulum body, and $x_p, \dot{x}_p, \ddot{x}_p$ are the position, velocity, and acceleration of the pendulum's center of gravity.

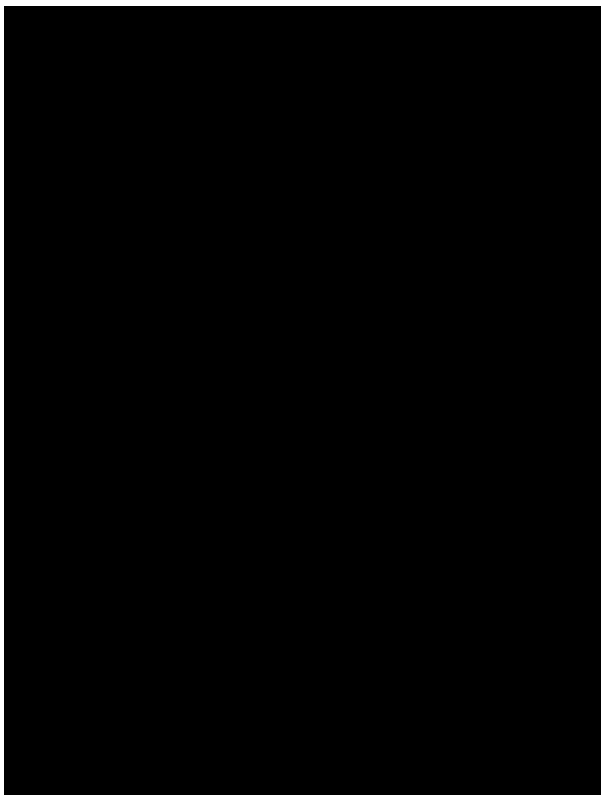

Figure 5.3. Inverted Pendulum Free Body Diagram

Begin by summing forces in the $x_p$ direction

$$\sum F_{xp} = H_L + H_R = M_p \ddot{x}_p \tag{59}$$

and the $y_p$ direction

$$\sum F_{yp} = P_L + P_R - M_p g = M_p \ddot{y}_p \tag{60}$$

The acceleration of a point in a translating coordinate frame can be found by relative motion analysis [48]. The acceleration vector of the pendulum's center of gravity, here called $\mathbf{a}_B$, can be found with

$$\mathbf{a}_B = \mathbf{a}_A + \boldsymbol{\alpha} \times \mathbf{r}_{B/A} + \boldsymbol{\omega} \times (\boldsymbol{\omega} \times \mathbf{r}_{B/A}) \tag{61}$$

where the following definitions are used

$$\mathbf{a}_B = \ddot{x}_p \hat{i} + \ddot{y}_p \hat{j} \tag{62}$$

$$\mathbf{r}_{B/A} = l \sin\theta \hat{i} - l \cos\theta \hat{j} \tag{63}$$

$$\boldsymbol{\omega} = \dot{\theta} \hat{k} \tag{64}$$

$$\boldsymbol{\alpha} = \ddot{\theta} \hat{k} \tag{65}$$

and the acceleration of the pendulum body is expressed as

$$\mathbf{a}_A = \ddot{x} \hat{i} \tag{66}$$

Using Eqs. (63)-(66) in Eq. (61), the acceleration of the pendulum's center of gravity can be found.

$$\mathbf{a}_B = \left(\ddot{x} + \ddot{\theta} l\cos\theta - \dot{\theta}^2 l\sin\theta\right)\hat{i} + \left(\ddot{\theta} l sin\theta + \dot{\theta}^2 l cos\theta\right)\hat{j} \tag{67}$$

Use Eq. (67) in Eq. (59) and Eq. (60) to obtain

$$H_L + H_R = M_p\left(\ddot{x} + \ddot{\theta} l\cos\theta - \dot{\theta}^2 l\sin\theta\right) \tag{68}$$

and

$$P_L + P_R = M_p g + M_p\left(\ddot{\theta} l sin\theta + \dot{\theta}^2 l cos\theta\right) \tag{69}$$

Now, sum moments about the pendulum's center of gravity with the clockwise direction taken as positive.

$$\sum M = -(C_L + C_R) - (H_L + H_R) l\cos\theta - (P_L + P_R) l\sin\theta = I_p\ddot{\theta} \tag{70}$$

Combine Eqs. (51), (68), (69), and (70) to obtain the following.

$$\begin{aligned} -\left(-\frac{2k_m k_e}{R} + \frac{2k_m}{R}V_a\right) - \left(M_p\left(\ddot{x} + \ddot{\theta} l\cos\theta - \dot{\theta}^2 l\sin\theta\right)\right) l\cos\theta \\ -\left(M_p g + M_p\left(\ddot{\theta} l sin\theta + \dot{\theta}^2 l cos\theta\right)\right) l\sin\theta = I_p\ddot{\theta} \end{aligned} \tag{71}$$

Cancel terms, rearrange, and use the trigonometric identity $\sin^2\theta + \cos^2\theta = 1$ to obtain the first nonlinear equation of motion for the two wheeled inverted pendulum

$$\left(I_p + M_p l^2\right)\ddot{\theta} - \frac{2k_m k_e}{Rr}\dot{x} + \frac{2k_m}{R}V + M_p g l sin\theta = -M_p l\ddot{x}cos\theta \tag{72}$$

Use Eq. (68) to remove the body forces from Eq. (58) to obtain the second nonlinear equation of motion

$$\frac{2k_m}{Rr}V = \left(2M_w + \frac{2I_w}{r^2} + M_p\right)\ddot{x} + \frac{2k_m k_e}{Rr^2}\dot{x} + M_p l\ddot{\theta}\cos\theta - M_p l\dot{\theta}^2\sin\theta \qquad (73)$$

Equations (72) and (73) describe the full nonlinear equations of motion for the two wheeled inverted pendulum. The point $\theta = \pi$ is the statically stable vertical point of the inverted pendulum. A set of linearized equations can be obtained by assuming $\theta = \pi + \phi$, where $\phi$ represents a small angle from the vertical position. Using the small angle assumption

$$\cos\theta = -1 \qquad (74)$$

and

$$\sin\theta = -\phi \qquad (75)$$

Also assume that the squared time derivative of the angular position is small.

$$\dot{\theta}^2 = 0 \qquad (76)$$

Using these assumptions the linearized equations of motion can be obtained as

$$\left(I_p + M_p l^2\right)\ddot{\phi} - \frac{2k_m k_e}{Rr}\dot{x} + \frac{2k_m}{R}V - M_p g l\phi = M_p l\ddot{x} \qquad (77)$$

and

$$\frac{2k_m}{Rr}V = \left(2M_w + \frac{2I_w}{r^2} + M_p\right)\ddot{x} + \frac{2k_m k_e}{Rr^2}\dot{x} - M_p l\ddot{\phi}\cos\phi \qquad (78)$$

Rearranging the linearized model the following state-space model is obtained

$$\dot{\mathbf{x}} = A\mathbf{x} + Bu$$

$$\frac{d}{dt}\begin{bmatrix} x \\ \dot{x} \\ \phi \\ \dot{\phi} \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & \dfrac{2k_m k_e \left(M_p l r - I_p - M_p l^2\right)}{Rr^2\alpha} & \dfrac{M_p^2 g l^2}{\alpha} & 0 \\ 0 & 0 & 0 & 1 \\ 0 & \dfrac{2k_m k_e \left(r\beta - M_p l\right)}{Rr^2\alpha} & \dfrac{M_p g l \beta}{\alpha} & 0 \end{bmatrix} \begin{bmatrix} x \\ \dot{x} \\ \phi \\ \dot{\phi} \end{bmatrix} + \begin{bmatrix} 0 \\ \dfrac{2k_m \left(I_p + M_p l^2 - M_p l r\right)}{Rr\alpha} \\ 0 \\ \dfrac{2k_m \left(M_p l - r\beta\right)}{Rr\alpha} \end{bmatrix} V \tag{79}$$

where

$$\beta = \left(2M_w + \frac{2I_w}{r^2} + M_p\right) \tag{80}$$

and

$$\alpha = \left[I_p\beta + 2M_p l^2\left(M_w + \frac{I_w}{r^2}\right)\right] \tag{81}$$

## 5.3. MODEL SIMULATION STUDY

The model should be checked for accuracy and for the expected motion of an inverted pendulum. This is easily done by examining the uncontrolled motion of the nonlinear equations of motion. Any values of the parameters can be used. To check the stability of the plant, choose the starting point $\theta = 180^\circ$, the plant should not deviate from the statically stable point. Figure 5.4 shows the states with this starting point, and as shown the system is stable at this point.

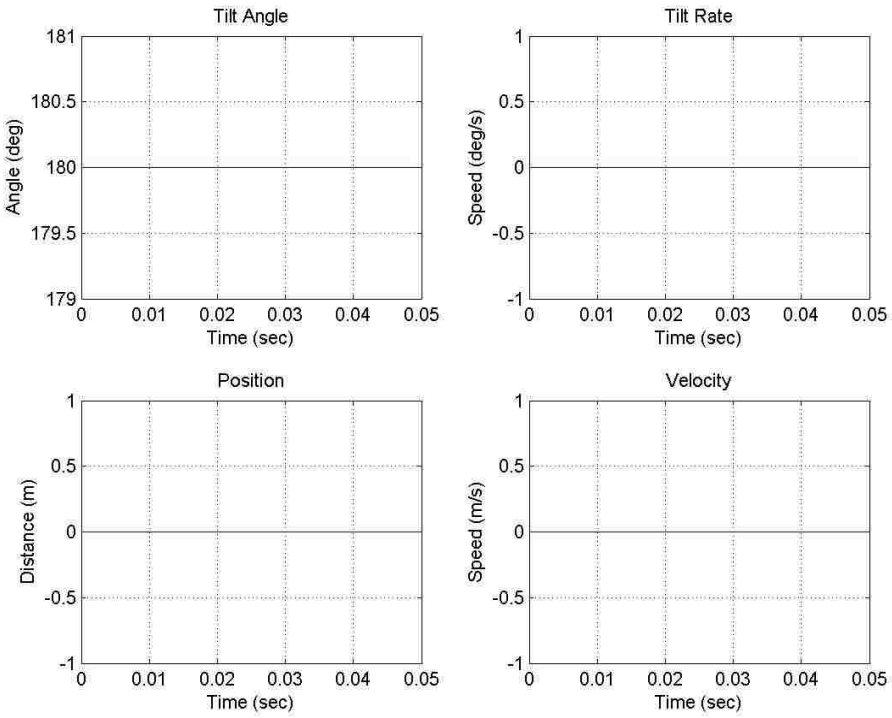


Figure 5.4. Stability Test θ=180°

The stable point $\theta = 0^{\circ}$ is examined by using the starting point $\theta = 10^{\circ}$. The plant is expected to oscillate about $\theta = 0^{\circ}$. With friction modeled the system should have damped oscillations. In this case, while there is not friction modeled, the motor dynamics creates an amount of resistance resulting in a frictional like effect. Looking at the free body diagram of the pendulum in Figure 5.3, with a small positive angle the body of the pendulum should initially move in the positive direction as the tilt angle returns to zero. Figure 5.5 shows the result of this test, and the system behaves as expected. It is interesting to note that the position does not stabilize at zero. This steady state value of the position is in proportion to the starting angle, and is reasonable if the dynamics are considered. The robot will first move in the positive direction, and as the oscillations are damped the pendulum will not return to the starting point on the return swing, resulting in a net positive position. These simple checks show that the nonlinear equations of motion behave as expected and can be trusted.

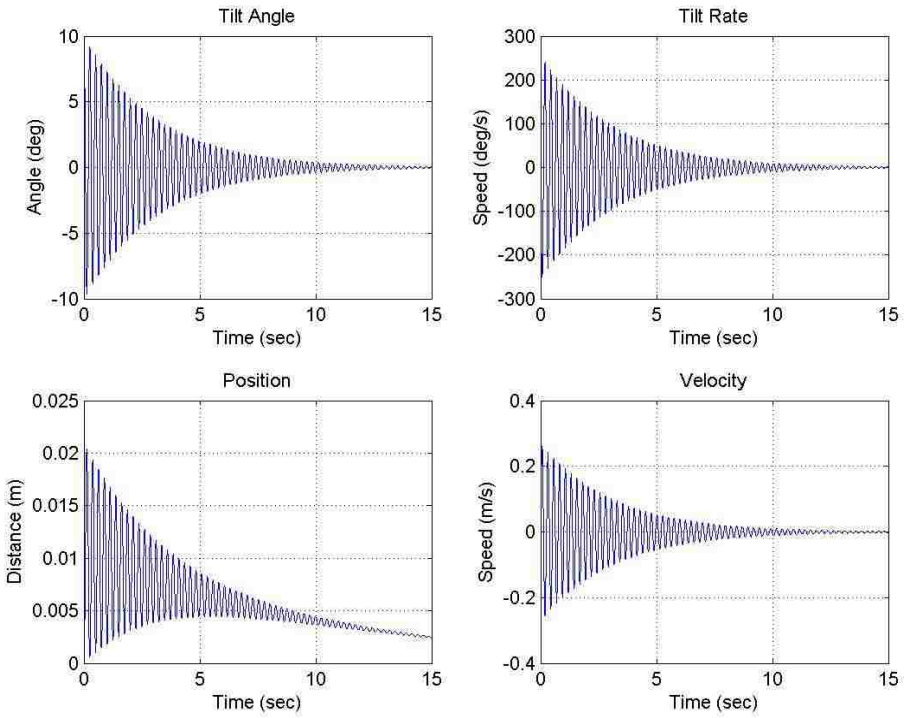


Figure 5.5. Stability Test θ=10°

# 6. CONTROL DESIGN

The two wheeled inverted pendulum is an unstable system and must be actively controlled to stabilize the plant. Many different methods of control exist, and it is a current area of research in nonlinear control [13, 23, 28, 29]. Linear controllers are effective for this nonlinear system when near the operating point and the parameters are accurately known [7, 8]. In this case the linear quadratic regulator (LQR) control design is used as the base control. An extra control signal is formulated to compensate for the unmatched uncertainties present in the two acceleration terms of the nonlinear equations of motion using neural networks.

## 6.1. LQR CONTROL DESIGN

As this controller is meant to be implemented in a digital system, a discrete-time formation is used. The infinite horizon, discrete time linear quadratic regulator is a modern state-space technique for discrete optimal control, where the performance index

$$J = \sum_{k=0}^{\infty} (\mathbf{x}_k{}^T \bar{Q} \mathbf{x}_k + \mathbf{u}_k^T \bar{R} \mathbf{u}_k) \tag{82}$$

is optimized by solving the discrete-time algebraic Riccati equation

$$P = \bar{Q} + F^T (P - PG(\bar{R} + G^T PG)^{-1} G^T P) A \tag{83}$$

where $P$ is the solution to the Riccati equation, $F$ and $G$ come from the discrete-time state-space model, and $\bar{Q}$ and $\bar{R}$ are user-selected positive definite weighting matrices of dimension $n \times n$ and $m \times m$ respectively. They weight the regulator performance and the control effort to obtain the optimal control based on the performance index. The user can weight the matrices to tune the response of the system, or to reduce the level of control.

The linear control law is given by

$$\mathbf{u}_k = -K_{LQR}(\mathbf{x}_k - \mathbf{x}_{des,k}) \tag{84}$$

where $\mathbf{x}_k$ is the state vector at time *k*, $\mathbf{x}_{des,k}$ is the desired state at time *k*, and

$$K_{LQR} = (\bar{R} + G^T P G)^{-1} G^T P F \tag{85}$$

## 6.2. EXTRA CONTROL DESIGN

In systems where the uncertainty is present in the same states as the control, the system is said to have matched uncertainty. In systems with matched uncertainty an estimate of the uncertainty, for example that found using the DMSO, can be used directly in the control to cancel the uncertainty. Unfortunately in the system of the two wheeled inverted pendulum, unmatched uncertainties are present. This section will outline a continuous-time technique to derive an extra control signal using the tilt angle as a virtual control. This method is an extension of a method described by Huang [49] and is used to compensate for the unmatched uncertainties in the system.

The two wheeled inverted pendulum equations of motion can be expressed by

$$\begin{aligned} \ddot{x} &= f_1(\dot{x}, \theta, \dot{\theta}) + g_1(\theta)\,\mathrm{V} \\ \ddot{\theta} &= f_2(\dot{x}, \theta, \dot{\theta}) + g_2(\theta)\,\mathrm{V} \end{aligned} \tag{86}$$

When linearizing and including parameter uncertainty, the equations of motion take the following form

$$\begin{aligned} \dot{x}_1 &= x_2 \\ \dot{x}_2 &= A_1 x_2 + A_2 x_3 + B_1 u + d_1(\mathbf{x}) \\ \dot{x}_3 &= x_4 \\ \dot{x}_4 &= A_3 x_2 + A_4 x_3 + B_2 u + d_2(\mathbf{x}) \end{aligned} \tag{87}$$

where

$$\begin{aligned} x_1 &= x \\ x_2 &= \dot{x} \\ x_3 &= \theta \\ x_4 &= \dot{\theta} \end{aligned} \tag{88}$$

and the coefficients, $A_1, A_2, A_3, A_4, B_1, B_2$, come from Eq. (79). In this formulation the equations of motion contain two separate uncertainties, $d_1(\mathbf{x})$ and $d_2(\mathbf{x})$, with only one control. To be stated another way, the system has unmatched uncertainties. A new control signal is defined as

$$u = u_{nom} + u_e \tag{89}$$

where $u_{nom}$ is a nominal control signal that is derived by other means, in this case LQR control design, and $u_e$ is an extra control signal that will be added to compensate for these uncertainties.

The desired system is defined by the equations of motion with the desired state values, indicated with the subscript *d*, and a desired control signal $u_d$.

$$\begin{aligned} \dot{x}_{1d} &= x_{2d} \\ \dot{x}_{2d} &= A_1 x_{2d} + A_2 x_{3d} + B_1 u_d \\ \dot{x}_{3d} &= x_{4d} \\ \dot{x}_{4d} &= A_3 x_{2d} + A_4 x_{3d} + B_2 u_d \end{aligned} \tag{90}$$

The goal is to be able to control the system's position using $x_1$ and $x_2$ to track a desired trajectory. The errors are defined as deviation from the desired values.

$$\begin{aligned} e_1 &= x_1 - x_{1d} \\ e_2 &= x_2 - x_{2d} \\ e_3 &= x_3 - x_{3d} \\ e_4 &= x_4 - x_{4d} \end{aligned} \tag{91}$$

It is assumed that the system is tracking a smooth trajectory and derivatives up to the second order of $x_{1d}$ are available. The error dynamics of the system are given by the following equations.

$$\begin{aligned}\dot{e}_1 &= x_2 - x_{2d} = e_2 \\ \dot{e}_2 &= A_1 x_2 + A_2 x_3 + B_1 u + d_1(\mathbf{x}) - \dot{x}_{2d} \\ \dot{e}_3 &= x_4 - x_{4d} = e_4 \\ \dot{e}_4 &= A_3 x_2 + A_4 x_3 + B_2 u + d_2(\mathbf{x}) - \dot{x}_{4d}\end{aligned} \tag{92}$$

The state $x_3$ is used as a virtual control to compensate for the uncertainty in $\dot{e}_2$. Therefore, a new state $\bar{x}_3$ will be designed to make $x_2$ and $x_1$ close to the desired values. The following form of $\bar{x}_3$ is used

$$\bar{x}_3 = x_{3d} + A_2^{-1}\left(-k_1 e_2 - e_1 - \hat{F}_1\right) \tag{93}$$

where $\hat{F}_1$ is a single layer neural network estimate. Using Eq. (93) in the equation for $\dot{e}_2$ the following is obtained.

$$\dot{e}_2 = A_1 x_2 + A_2\left(x_{3d} + A_2^{-1}\left(-k_1 e_2 - e_1 - \hat{F}_1\right)\right) + B_1 u + d_1(\mathbf{x}) - \dot{x}_{2d} \tag{94}$$

Performing algebraic multiplication $\dot{e}_2$ can be expressed by the following.

$$\dot{e}_2 = A_1 x_2 + A_2 x_{3d} - k_1 e_2 - e_1 - \hat{F}_1 + B_1 u + d_1(\mathbf{x}) - \dot{x}_{2d} \tag{95}$$

The neural network $\hat{F}_1$ is used to estimate the uncertainty $d_1(\mathbf{x})$ and the other states in the error equation. Ideally the neural network will estimate the following quantity.

$$F_1 = A_1 x_2 + A_2 x_{3d} + B_1 (u_{nom} + u_e) + d_1(\mathbf{x}) - \dot{x}_{2d} \tag{96}$$

Using this definition in Eq. (95) the error dynamics can be expressed as:

$$\dot{e}_2 = F_1 - \hat{F}_1 - k_1 e_2 - e_1 \tag{97}$$

A new error system is defined using the new state $\bar{x}_3$.

$$\begin{aligned} \bar{e}_1 &= x_1 - \bar{x}_1 \\ \bar{e}_2 &= x_2 - \bar{x}_2 \\ \bar{e}_3 &= x_3 - \bar{x}_3 \\ \bar{e}_4 &= x_4 - \bar{x}_4 \end{aligned} \tag{98}$$

As $x_1$ and $x_2$ have not been affected, $\bar{x}_1 = x_{1d}$ and $\bar{x}_2 = x_{2d}$. As $x_4$ is the derivative of $x_3$, the definition $\bar{x}_4 = \dot{\bar{x}}_3$ is used. The error bar dynamics are defined by the following equations.

$$\begin{aligned} \dot{\bar{e}}_1 &= \dot{x}_1 - \dot{\bar{x}}_1 \\ \dot{\bar{e}}_2 &= \dot{x}_2 - \dot{\bar{x}}_2 \\ \dot{\bar{e}}_3 &= \dot{x}_3 - \dot{\bar{x}}_3 \\ \dot{\bar{e}}_4 &= \dot{x}_4 - \dot{\bar{x}}_4 \end{aligned} \tag{99}$$

Including the state dynamics from Eq. (87) in Eq. (99) the error bar dynamics are obtained.

$$\begin{aligned} \dot{\bar{e}}_1 &= \dot{e}_1 = x_2 - x_{2d} = \bar{e}_2 \\ \dot{\bar{e}}_2 &= \dot{e}_2 = F_1 - \hat{F}_1 - k_1 \bar{e}_2 - \bar{e}_1 \\ \dot{\bar{e}}_3 &= x_4 - \bar{x}_4 = \bar{e}_4 \\ \dot{\bar{e}}_4 &= A_3 x_2 + A_4 x_3 + B_2 (u_{nom} + u_e) + d_2(\mathbf{x}) - \dot{\bar{x}}_4 \end{aligned} \tag{100}$$

The extra control signal, $u_e$, will be used to drive $x_4$ to $\bar{x}_4$ and in turn drive $x_3$ to $\bar{x}_3$. The extra control $u_e$ is selected as

$$u_e = B_2^{-1}\left(-k_2\bar{e}_4 - \hat{F}_2 - \bar{e}_3\right) \tag{101}$$

where $\hat{F}_2$ is a second single layer neural network. Using this extra control in the equation for $\dot{\bar{e}}_4$ the following expression for $\dot{\bar{e}}_4$ is obtained.

$$\dot{\bar{e}}_4 = A_3 x_2 + A_4 x_3 + B_2(u_{nom} + B_2^{-1}\left(-k_2\bar{e}_4 - \hat{F}_2 - \bar{e}_3\right)) + d_2(\mathbf{x}) - \dot{\bar{x}}_4 \tag{102}$$

Performing algebraic multiplication the following expression can be obtained.

$$\dot{\bar{e}}_4 = A_3 x_2 + A_4 x_3 + B_2 u_{nom} - k_2\bar{e}_4 - \hat{F}_2 - \bar{e}_3 + d_2(\mathbf{x}) - \dot{\bar{x}}_4 \tag{103}$$

The neural network $\hat{F}_2$ is used to estimate the uncertainty $d_2(\mathbf{x})$ and the other states in the error equation. Ideally the neural network will estimate the following quantity.

$$F_2 = A_3 x_2 + A_4 x_3 + B_2 u_{nom} + d_2(\mathbf{x}) - \dot{\bar{x}}_4 \tag{104}$$

Using this definition in Eq. (103) the dynamics can be simplified to

$$\dot{\bar{e}}_4 = F_2 - \hat{F}_2 + k_2\bar{e}_4 + \bar{e}_3 \tag{105}$$

The error bar dynamics are summarized by

$$\begin{aligned}\dot{\bar{e}}_1 &= \bar{e}_2 \\ \dot{\bar{e}}_2 &= F_1 - \hat{F}_1 - k_1\bar{e}_2 - \bar{e}_1 \\ \dot{\bar{e}}_3 &= \bar{e}_4 \\ \dot{\bar{e}}_4 &= F_2 - \hat{F}_2 - k_2\bar{e}_4 - \bar{e}_3\end{aligned} \tag{106}$$

For the two wheeled inverted pendulum problem a single-layer neural network was found to be incapable of estimating the nonlinear functions $F_1$ and $F_2$, so multi-layer neural networks are used as they provide better approximation capabilities than single layer networks. The functions described by Eqs. (96) and (104) can be expressed by the two-layer neural networks

$$F_1 = W_{12}^T\phi_{12}(W_{11}^T\phi_{11}(\mathrm{V}_1^T\mathrm{P}_1)) + \epsilon_1 \tag{107}$$

and

$$F_2 = W_{21}^T\phi_{21}(W_{22}^T\phi_{22}(\mathrm{V}_2^T\mathrm{P}_2)) + \epsilon_2 \tag{108}$$

where $W_{11}, W_{12}$ and $W_{21}, W_{22}$ are the ideal neural network weights. $V_1$ and $V_2$ are randomly selected input layer neural network weights. $\phi_{11}(\mathbf{x}), \phi_{12}(\mathbf{x})$ and $\phi_{21}(\mathbf{x}), \phi_{22}(\mathbf{x})$ are user selected activation functions and $P_1$ and $P_2$ are inputs to the neural networks. The neural networks provide estimates of the functions $F_1$ and $F_2$ where $\epsilon_1$ and $\epsilon_2$ are the bounded neural network estimation errors which satisfy the conditions $\|\epsilon_1\| \le \epsilon_{1m}$ and $\|\epsilon_2\| \le \epsilon_{2m}$. Estimates of the neural network weights are used to calculate estimates of the functions $F_1$ and $F_2$, defined by

$$\hat{F}_1 = \hat{W}_{12}^T\hat{\phi}_{12}(\hat{W}_{11}^T\hat{\phi}_{11}(\mathrm{V}_1^T\mathrm{P}_1)) \tag{109}$$

and

$$\hat{F}_2 = \hat{W}_{22}^T \hat{\phi}_{22}(\hat{W}_{21}^T \hat{\phi}_{21}(\mathrm{V}_2^T \mathrm{P}_2)) \tag{110}$$

The weight update laws are chosen for the first neural network as

$$\dot{\hat{W}}_{11} = -\gamma_{11}\left(\hat{\phi}_{11}\left[\hat{W}_{11}^T \hat{\phi}_{11} + B_1 \bar{e}_2\right]^T + \sigma_{11}\hat{W}_{11}\right) \tag{111}$$

and

$$\dot{\hat{W}}_{12} = \gamma_{12}\left(\hat{\phi}_{12}\bar{e}_2^T - \sigma_{12}\hat{W}_{12}\right) \tag{112}$$

where $\gamma_{11}, \gamma_{12}, \sigma_{11}$ and $\sigma_{12}$ are neural network adaptation or learning rates and $B_1$ is a coefficient matrix. For the second neural network the update laws are similar, defined as

$$\dot{\hat{W}}_{21} = -\gamma_{21}\left(\hat{\phi}_{21}\left[\hat{W}_{21}^T \hat{\phi}_{21} + B_2 \bar{e}_4\right]^T + \sigma_{21}\hat{W}_{21}\right) \tag{113}$$

and

$$\dot{\hat{W}}_{22} = \gamma_{22}\left(\hat{\phi}_{22}\bar{e}_4^T - \sigma_{22}\hat{W}_{22}\right) \tag{114}$$

Lyapunov stability analysis is used to show the boundedness of the state tracking errors and the neural network weights. Define the Lyapunov function as follows.

$$\begin{aligned} V = {} & \frac{1}{2}\bar{e}_1^T\bar{e}_1 + \frac{1}{2}\bar{e}_2^T\bar{e}_2 + \frac{1}{2}\bar{e}_3^T\bar{e}_3 + \frac{1}{2}\bar{e}_4^T\bar{e}_4 \\ & + \frac{1}{2\gamma_{11}}Tr\left(\tilde{W}_{11}^T\tilde{W}_{11}\right) + \frac{1}{2\gamma_{12}}Tr\left(\tilde{W}_{12}^T\tilde{W}_{12}\right) + \frac{1}{2\gamma_{21}}Tr\left(\tilde{W}_{21}^T\tilde{W}_{21}\right) + \frac{1}{2\gamma_{22}}Tr\left(\tilde{W}_{22}^T\tilde{W}_{22}\right) \end{aligned} \tag{115}$$

The derivative of the Lyapunov function is given by:

$$\dot{V} = \bar{e}_1^T \dot{\bar{e}}_1 + \bar{e}_2^T \dot{\bar{e}}_2 + \bar{e}_3^T \dot{\bar{e}}_3 + \bar{e}_4^T \dot{\bar{e}}_4 + \frac{1}{\gamma_{11}} Tr\left(\tilde{W}_{11}{}^T \dot{\tilde{W}}_{11}\right) + \frac{1}{\gamma_{12}} Tr\left(\tilde{W}_{12}{}^T \dot{\tilde{W}}_{12}\right) + \frac{1}{\gamma_{21}} Tr\left(\tilde{W}_{21}{}^T \dot{\tilde{W}}_{21}\right) + \frac{1}{\gamma_{22}} Tr\left(\tilde{W}_{22}{}^T \dot{\tilde{W}}_{22}\right) \quad (116)$$

where $\tilde{W} = W - \hat{W}$ . The Lyapunov function is split into two parts, the first part consists of the error bar terms.

$$\dot{V}_1 = \bar{e}_1^T \dot{\bar{e}}_1 + \bar{e}_2^T \dot{\bar{e}}_2 + \bar{e}_3^T \dot{\bar{e}}_3 + \bar{e}_4^T \dot{\bar{e}}_4 \quad (117)$$

Including the error bar dynamics from Eq. (106) in Eq. (117) the first half of the Lyapunov function becomes

$$\dot{V}_1 = \bar{e}_1^T \bar{e}_2 + \bar{e}_2^T \left(F_1 - \hat{F}_1 - k_1 \bar{e}_2 - \bar{e}_1\right) + \bar{e}_3^T \bar{e}_4 + \bar{e}_4^T \left(F_2 - \hat{F}_2 - k_2 \bar{e}_4 - \bar{e}_3\right) \quad (118)$$

The neural network estimates $\hat{F}_1$ and $\hat{F}_2$ are subtracted from the ideal neural networks $F_1$ and $F_2$ as follows

$$F_i - \hat{F}_i = W_{i2}^T \phi_{i2}(W_{i1}^T \phi_{i1}(\mathrm{V}_i^T \mathrm{P}_i)) + \epsilon_i - \hat{W}_{i2}^T \hat{\phi}_{12}(\hat{W}_{i1}^T \hat{\phi}_{i1}(\mathrm{V}_i^T \mathrm{P}_i)) \quad (119)$$

for $i = 1, 2$ . Eq. (119) can be expressed as

$$F_i - \hat{F}_i = \tilde{W}_{i2}^T \hat{\phi}_{i2} + W_{i2}^T \tilde{\phi}_{i2} + \epsilon_i \quad (120)$$

where $\tilde{\phi}_{i2} = \phi_{i2}(W_{i1}^T \phi_{i1}(\mathrm{V}_i^T \mathrm{P}_i)) - \hat{\phi}_{i2}(\hat{W}_{i1}^T \hat{\phi}_{i1}(\mathrm{V}_i^T \mathrm{P}_i))$ . Using Eq. (120) in Eq. (118) and distributing the $\bar{e}_2$ and $\bar{e}_4$ terms the following expression is obtained.

$$\begin{aligned}\dot{V}_1 &= \bar{e}_1^T\bar{e}_2 - \bar{e}_1^T\bar{e}_2 - \bar{e}_2^T k_1\bar{e}_2 + \bar{e}_3^T\bar{e}_4 - \bar{e}_4^T k_2\bar{e}_4 - \bar{e}_3^T\bar{e}_4 \\ &+\bar{e}_2^T(\tilde{W}_{12}^T\hat{\phi}_{12} + W_{12}^T\tilde{\phi}_{12} + \epsilon_1) + \bar{e}_4^T(\tilde{W}_{22}^T\hat{\phi}_{22} + W_{22}^T\tilde{\phi}_{22} + \epsilon_2)\end{aligned} \tag{121}$$

Cancel the $\bar{e}_1^T\bar{e}_2$ and $\bar{e}_3^T\bar{e}_4$ terms to obtain

$$\dot{V}_1 = -\bar{e}_2^T k_1\bar{e}_2 - \bar{e}_4^T k_2\bar{e}_4 + \bar{e}_2^T\tilde{W}_{12}^T\hat{\phi}_{12} + \bar{e}_2^T W_{12}^T\tilde{\phi}_{12} + \bar{e}_2^T\epsilon_1 + \bar{e}_4^T\tilde{W}_{22}^T\hat{\phi}_{22} + \bar{e}_4^T W_{22}^T\tilde{\phi}_{22} + \bar{e}_4^T\epsilon_2 \tag{122}$$

The second half of the Lyapunov function is the half consisting of the neural network estimation error terms, expressed by

$$\dot{V}_2 = \frac{1}{\gamma_{11}}Tr\left(\tilde{W}_{11}^T\dot{\tilde{W}}_{11}\right) + \frac{1}{\gamma_{12}}Tr\left(\tilde{W}_{12}^T\dot{\tilde{W}}_{12}\right) + \frac{1}{\gamma_{21}}Tr\left(\tilde{W}_{21}^T\dot{\tilde{W}}_{21}\right) + \frac{1}{\gamma_{22}}Tr\left(\tilde{W}_{22}^T\dot{\tilde{W}}_{22}\right) \tag{123}$$

For the neural network weight update laws to be used in the Lyapunov function, use the definition of $\tilde{W}$ and its derivative, expressed by

$$\begin{aligned}\tilde{W} &= W - \hat{W} \\ \dot{\tilde{W}} &= -\dot{\hat{W}}\end{aligned} \tag{124}$$

Using Eq. (124) the weight update laws can be expressed as

$$\dot{\tilde{W}}_{11} = \gamma_{11}\left(\hat{\phi}_{11}\left[\hat{W}_{11}^T\hat{\phi}_{11} + B_1\bar{e}_2\right]^T + \sigma_{11}\hat{W}_{11}\right) \tag{125}$$

$$\dot{\tilde{W}}_{12} = -\gamma_{12}\left(\hat{\phi}_{12}\bar{e}_2^T - \sigma_{12}\hat{W}_{12}\right) \tag{126}$$

$$\dot{\tilde{W}}_{21} = \gamma_{21}\left(\hat{\phi}_{21}\left[\hat{W}_{21}^T\hat{\phi}_{21} + B_2\bar{e}_4\right]^T + \sigma_{21}\hat{W}_{21}\right) \tag{127}$$

and

$$\dot{\hat{W}}_{22} = -\gamma_{22}\left(\hat{\phi}_{22}\bar{e}_4^T - \sigma_{22}\hat{W}_{22}\right) \tag{128}$$

Using Eqs. (125) and (128) in Eq. (123) the following expression for the second half of the Lyapunov function is obtained.

$$\begin{aligned}\dot{V}_2 = \frac{1}{\gamma_{11}}Tr\left(\tilde{W}_{11}^{\ T}\left[\gamma_{11}\left(\hat{\phi}_{11}\left[\hat{W}_{11}^{\ T}\hat{\phi}_{11} + B_1\bar{e}_2\right]^T + \sigma_{11}\hat{W}_{11}\right)\right]\right)\\ + \frac{1}{\gamma_{12}}Tr\left(\tilde{W}_{12}^{\ T}\left[-\gamma_{12}\left(\hat{\phi}_{12}\bar{e}_2^T - \sigma_{12}\hat{W}_{12}\right)\right]\right)\\ + \frac{1}{\gamma_{21}}Tr\left(\tilde{W}_{21}^{\ T}\left[\gamma_{21}\left(\hat{\phi}_{21}\left[\hat{W}_{21}^{\ T}\hat{\phi}_{21} + B_2\bar{e}_4\right]^T + \sigma_2\hat{W}_{21}\right)\right]\right)\\ + \frac{1}{\gamma_{22}}Tr\left(\tilde{W}_{22}^{\ T}\left[-\gamma_{22}\left(\hat{\phi}_{22}\bar{e}_4^T - \sigma_{22}\hat{W}_{22}\right)\right]\right)\end{aligned} \tag{129}$$

Performing algebraic multiplication the following expression is obtained.

$$\begin{aligned}\dot{V}_2 = Tr\left(\tilde{W}_{11}^{\ T}\hat{\phi}_{11}\left[\hat{W}_{11}^{\ T}\hat{\phi}_{11}\right]^T\right) + Tr\left(\tilde{W}_{11}^{\ T}\hat{\phi}_{11}\left[B_1\bar{e}_2\right]^T\right) + \sigma_{11}Tr\left(\tilde{W}_{11}^{\ T}\hat{W}_{11}\right)\\ -Tr\left(\tilde{W}_{12}^{\ T}\hat{\phi}_{12}\bar{e}_2^T\right) + \sigma_{12}Tr\left(\tilde{W}_{12}^{\ T}\hat{W}_{12}\right)\\ +Tr\left(\tilde{W}_{21}^{\ T}\hat{\phi}_{21}\left[\hat{W}_{21}^{\ T}\hat{\phi}_{21}\right]^T\right) + Tr\left(\tilde{W}_{21}^{\ T}\hat{\phi}_{21}\left[B_2\bar{e}_4\right]^T\right) + \sigma_{21}Tr\left(\tilde{W}_{21}^{\ T}\hat{W}_{21}\right)\\ -Tr\left(\tilde{W}_{22}^{\ T}\hat{\phi}_{22}\bar{e}_4^T\right) + \sigma_{22}Tr\left(\tilde{W}_{22}^{\ T}\hat{W}_{22}\right)\end{aligned} \tag{130}$$

Combining the two parts of the Lyapunov function from Eqs. (122) and (130) the Lyapunov function becomes

$$\begin{aligned}\dot{V} = &-\bar{e}_2^{\,T} k_1 \bar{e}_2 - \bar{e}_4^{\,T} k_2 \bar{e}_4 + \bar{e}_2^T \tilde{W}_{12}^{\,T} \hat{\phi}_{12} + \bar{e}_2^T W_{12}^{\,T} \tilde{\phi}_{12} + \bar{e}_2^T \epsilon_1 + \bar{e}_4^T \tilde{W}_{22}^{\,T} \hat{\phi}_{22} + \bar{e}_4^T W_{22}^{\,T} \tilde{\phi}_{22} + \bar{e}_4^T \epsilon_2 \\ &+ Tr\left(\tilde{W}_{11}^{\,T} \hat{\phi}_{11} \left[\hat{W}_{11}^{\,T} \hat{\phi}_{11}\right]^T\right) + Tr\left(\tilde{W}_{11}^{\,T} \hat{\phi}_{11} \left[B_1 \bar{e}_2\right]^T\right) + \sigma_{11} Tr\left(\tilde{W}_{11}^{\,T} \hat{W}_{11}\right) \\ &- Tr\left(\tilde{W}_{12}^{\,T} \hat{\phi}_{12} \bar{e}_2^T\right) + \sigma_{12} Tr\left(\tilde{W}_{12}^{\,T} \hat{W}_{12}\right) - Tr\left(\tilde{W}_{22}^{\,T} \hat{\phi}_{22} \bar{e}_4^T\right) + \sigma_{22} Tr\left(\tilde{W}_{22}^{\,T} \hat{W}_{22}\right) \\ &+ Tr\left(\tilde{W}_{21}^{\,T} \hat{\phi}_{21} \left[\hat{W}_{21}^{\,T} \hat{\phi}_{21}\right]^T\right) + Tr\left(\tilde{W}_{21}^{\,T} \hat{\phi}_{21} \left[B_2 \bar{e}_4\right]^T\right) + \sigma_{21} Tr\left(\tilde{W}_{21}^{\,T} \hat{W}_{21}\right)\end{aligned} \tag{131}$$

Note that $\bar{e}_2^T \tilde{W}_{12}^{\,T} \hat{\phi}_{12} = Tr\left(\tilde{W}_{12}^{\,T} \hat{\phi}_{12} \bar{e}_2^T\right)$ and $\bar{e}_4^T \tilde{W}_{22}^{\,T} \hat{\phi}_{22} = Tr\left(\tilde{W}_{22}^{\,T} \hat{\phi}_{22} \bar{e}_4^T\right)$. Using this, Eq. (131) becomes

$$\begin{aligned}\dot{V} = &-\bar{e}_2^{\,T} k_1 \bar{e}_2 - \bar{e}_4^{\,T} k_2 \bar{e}_4 + \bar{e}_2^T W_{12}^{\,T} \tilde{\phi}_{12} + \bar{e}_2^T \epsilon_1 + \bar{e}_4^T W_{22}^{\,T} \tilde{\phi}_{22} + \bar{e}_4^T \epsilon_2 \\ &+ Tr\left(\tilde{W}_{11}^{\,T} \hat{\phi}_{11} \left[\hat{W}_{11}^{\,T} \hat{\phi}_{11}\right]^T\right) + Tr\left(\tilde{W}_{11}^{\,T} \hat{\phi}_{11} \left[B_1 \bar{e}_2\right]^T\right) + \sigma_{11} Tr\left(\tilde{W}_{11}^{\,T} \hat{W}_{11}\right) \\ &+ \sigma_{12} Tr\left(\tilde{W}_{12}^{\,T} \hat{W}_{12}\right) \\ &+ Tr\left(\tilde{W}_{21}^{\,T} \hat{\phi}_{21} \left[\hat{W}_{21}^{\,T} \hat{\phi}_{21}\right]^T\right) + Tr\left(\tilde{W}_{21}^{\,T} \hat{\phi}_{21} \left[B_2 \bar{e}_4\right]^T\right) + \sigma_{21} Tr\left(\tilde{W}_{21}^{\,T} \hat{W}_{21}\right) \\ &+ \sigma_{22} Tr\left(\tilde{W}_{22}^{\,T} \hat{W}_{22}\right)\end{aligned} \tag{132}$$

Use $\hat{W} = W - \tilde{W}$ and distribute terms to obtain the following.

$$\begin{aligned}\dot{V} = &-\bar{e}_2^{\,T} k_1 \bar{e}_2 - \bar{e}_4^{\,T} k_2 \bar{e}_4 + \bar{e}_2^T W_{12}^{\,T} \tilde{\phi}_{12} + \bar{e}_2^T \epsilon_1 + \bar{e}_4^T W_{22}^{\,T} \tilde{\phi}_{22} + \bar{e}_4^T \epsilon_2 \\ &+ Tr\left(\tilde{W}_{11}^{\,T} \hat{\phi}_{11} \hat{\phi}_{11}^T W_{11}\right) - Tr\left(\tilde{W}_{11}^{\,T} \hat{\phi}_{11} \hat{\phi}_{11}^T \tilde{W}_{11}\right) + Tr\left(\tilde{W}_{11}^{\,T} \hat{\phi}_{11} \bar{e}_2^T B_1^T\right) \\ &+ \sigma_{11} Tr\left(\tilde{W}_{11}^{\,T} W_{11}\right) - \sigma_{11} Tr\left(\tilde{W}_{11}^{\,T} \tilde{W}_{11}\right) + \sigma_{12} Tr\left(\tilde{W}_{12}^{\,T} W_{12}\right) - \sigma_{12} Tr\left(\tilde{W}_{12}^{\,T} \tilde{W}_{12}\right) \\ &+ Tr\left(\tilde{W}_{21}^{\,T} \hat{\phi}_{21} \hat{\phi}_{21}^T W_{21}\right) - Tr\left(\tilde{W}_{21}^{\,T} \hat{\phi}_{21} \hat{\phi}_{21}^T \tilde{W}_{21}\right) + Tr\left(\tilde{W}_{21}^{\,T} \hat{\phi}_{21} \bar{e}_4^T B_2^T\right) \\ &+ \sigma_{21} Tr\left(\tilde{W}_{21}^{\,T} W_{21}\right) - \sigma_{21} Tr\left(\tilde{W}_{21}^{\,T} \tilde{W}_{21}\right) + \sigma_{22} Tr\left(\tilde{W}_{22}^{\,T} W_{22}\right) - \sigma_{22} Tr\left(\tilde{W}_{22}^{\,T} \tilde{W}_{22}\right)\end{aligned} \tag{133}$$

The activation functions $\phi_{ij}$ for $i = 1, 2$ and $j = 1, 2$ are bounded functions satisfying $\left\|\phi_{ij}\right\| \le \phi_{ijm}$. As long as the activation functions are selected as bounded functions, this same bound will apply to the estimates and the estimate error terms $\hat{\phi}_{ij}$

and $\tilde{\phi}_{ij}$. Applying norms to the neural network weight terms and using the bounds on the neural network approximation errors $\|\epsilon_1\| \le \epsilon_{1m}$ and $\|\epsilon_2\| \le \epsilon_{2m}$, and the bounds on the neural network weights $\|W_{11}\| \le W_{11m}$, $\|W_{12}\| \le W_{12m}$, $\|W_{21}\| \le W_{21m}$, and $\|W_{22}\| \le W_{22m}$ the following can be obtained.

$$\begin{aligned}\dot{V} \le & -\lambda_{\min}(k_1)\|\bar{e}_2\|^2 - \lambda_{\min}(k_2)\|\bar{e}_4\|^2 + \|\bar{e}_2\|W_{12m}\phi_{12m} + \|\bar{e}_2\|\epsilon_{1m} + \|\bar{e}_4\|W_{22m}\phi_{22m} + \|\bar{e}_4\|\epsilon_{2m} \\ & + \left\|\tilde{W}_{11}^{\ T}\hat{\phi}_{11}\right\|\phi_{11m}W_{11m} - \left\|\tilde{W}_{11}^{\ T}\hat{\phi}_{11}\right\|^2 + \left\|\tilde{W}_{11}^{\ T}\hat{\phi}_{11}\right\|\|\bar{e}_2\|\|B_1\| \\ & + \sigma_{11}\left\|\tilde{W}_{11}\right\|W_{11m} - \sigma_{11}\left\|\tilde{W}_{11}\right\|^2 + \sigma_{12}\left\|\tilde{W}_{12}\right\|W_{12m} - \sigma_{12}\left\|\tilde{W}_{12}\right\|^2 \\ & + \left\|\tilde{W}_{21}^{\ T}\hat{\phi}_{21}\right\|\phi_{21m}W_{21m} - \left\|\tilde{W}_{21}^{\ T}\hat{\phi}_{21}\right\|^2 + \left\|\tilde{W}_{21}^{\ T}\hat{\phi}_{21}\right\|\|\bar{e}_4\|\|B_2\| \\ & + \sigma_{21}\left\|\tilde{W}_{21}\right\|W_{21m} - \sigma_{21}\left\|\tilde{W}_{21}\right\|^2 + \sigma_{22}\left\|\tilde{W}_{22}\right\|W_{22m} - \sigma_{22}\left\|\tilde{W}_{22}\right\|^2\end{aligned} \tag{134}$$

Completing the square on the $\left\|\tilde{W}_{11}^{\ T}\hat{\phi}_{11}\right\|\|\bar{e}_2\|\|B_1\|$ and $\left\|\tilde{W}_{21}^{\ T}\hat{\phi}_{21}\right\|\|\bar{e}_4\|\|B_2\|$ terms yields

$$\begin{aligned}\dot{V} \le & -\lambda_{\min}(k_1)\|\bar{e}_2\|^2 - \lambda_{\min}(k_2)\|\bar{e}_4\|^2 + \|\bar{e}_2\|W_{12m}\phi_{12m} + \|\bar{e}_2\|\epsilon_{1m} + \|\bar{e}_4\|W_{22m}\phi_{22m} + \|\bar{e}_4\|\epsilon_{2m} \\ & + \left\|\tilde{W}_{11}^{\ T}\hat{\phi}_{11}\right\|\phi_{11m}W_{11m} - \left\|\tilde{W}_{11}^{\ T}\hat{\phi}_{11}\right\|^2 \\ & + \frac{1}{4}\left\|\tilde{W}_{11}^{\ T}\hat{\phi}_{11}\right\|^2 + \|\bar{e}_2\|^2\|B_1\|^2 - \left(\frac{1}{2}\left\|\tilde{W}_{11}^{\ T}\hat{\phi}_{11}\right\| - \|\bar{e}_2\|\|B_1\|\right)^2 \\ & + \sigma_{11}\left\|\tilde{W}_{11}\right\|W_{11m} - \sigma_{11}\left\|\tilde{W}_{11}\right\|^2 + \sigma_{12}\left\|\tilde{W}_{12}\right\|W_{12m} - \sigma_{12}\left\|\tilde{W}_{12}\right\|^2 \\ & + \left\|\tilde{W}_{21}^{\ T}\hat{\phi}_{21}\right\|\phi_{21m}W_{21m} - \left\|\tilde{W}_{21}^{\ T}\hat{\phi}_{21}\right\|^2 \\ & + \frac{1}{4}\left\|\tilde{W}_{21}^{\ T}\hat{\phi}_{21}\right\|^2 + \|\bar{e}_4\|^2\|B_2\|^2 - \left(\frac{1}{2}\left\|\tilde{W}_{21}^{\ T}\hat{\phi}_{21}\right\| - \|\bar{e}_4\|\|B_2\|\right)^2 \\ & + \sigma_{21}\left\|\tilde{W}_{21}\right\|W_{21m} - \sigma_{21}\left\|\tilde{W}_{21}\right\|^2 + \sigma_{22}\left\|\tilde{W}_{22}\right\|W_{22m} - \sigma_{22}\left\|\tilde{W}_{22}\right\|^2\end{aligned} \tag{135}$$

Combining terms and completing the square in a similar fashion on the $\left\|\tilde{W}_{11}^{\ T}\hat{\phi}_{11}\right\|\phi_{11m}W_{11m}$ and $\left\|\tilde{W}_{21}^{\ T}\hat{\phi}_{21}\right\|\phi_{21m}W_{21m}$ terms leads to the expression

$$\begin{aligned}\dot{V} \le & -\lambda_{\min}(k_1)\left\|\bar{e}_2\right\|^2 - \lambda_{\min}(k_2)\left\|\bar{e}_4\right\|^2 + \left\|\bar{e}_2\right\|W_{12m}\phi_{12m} + \left\|\bar{e}_2\right\|\epsilon_{1m} + \left\|\bar{e}_4\right\|W_{22m}\phi_{22m} + \left\|\bar{e}_4\right\|\epsilon_{2m} \\ & + \phi_{11m}^2 W_{11m}^2 - \left(\frac{1}{2}\left\|\tilde{W}_{11}{}^T\hat{\phi}_{11}\right\| - \phi_{11m}W_{11m}\right)^2 - \frac{1}{2}\left\|\tilde{W}_{11}{}^T\hat{\phi}_{11}\right\|^2 \\ & + \left\|\bar{e}_2\right\|^2\left\|B_1\right\|^2 - \left(\frac{1}{2}\left\|\tilde{W}_{11}{}^T\hat{\phi}_{11}\right\| - \left\|\bar{e}_2\right\|\left\|B_1\right\|\right)^2 \\ & + \sigma_{11}\left\|\tilde{W}_{11}\right\|W_{11m} - \sigma_{11}\left\|\tilde{W}_{11}\right\|^2 + \sigma_{12}\left\|\tilde{W}_{12}\right\|W_{12m} - \sigma_{12}\left\|\tilde{W}_{12}\right\|^2 \\ & + \phi_{21m}^2 W_{21m}^2 - \left(\frac{1}{2}\left\|\tilde{W}_{21}{}^T\hat{\phi}_{21}\right\| - \phi_{21m}W_{21m}\right)^2 - \frac{1}{2}\left\|\tilde{W}_{21}{}^T\hat{\phi}_{21}\right\|^2 \\ & + \left\|\bar{e}_4\right\|^2\left\|B_2\right\|^2 - \left(\frac{1}{2}\left\|\tilde{W}_{21}{}^T\hat{\phi}_{21}\right\| - \left\|\bar{e}_4\right\|\left\|B_2\right\|\right)^2 \\ & + \sigma_{21}\left\|\tilde{W}_{21}\right\|W_{21m} - \sigma_{21}\left\|\tilde{W}_{21}\right\|^2 + \sigma_{22}\left\|\tilde{W}_{22}\right\|W_{22m} - \sigma_{22}\left\|\tilde{W}_{22}\right\|^2\end{aligned} \tag{136}$$

Finally, by completing the square one more time on the $\sigma_{11}\left\|\tilde{W}_{11}\right\|W_{11m}$, $\sigma_{12}\left\|\tilde{W}_{12}\right\|W_{12m}$, $\sigma_{21}\left\|\tilde{W}_{21}\right\|W_{21m}$ and $\sigma_{22}\left\|\tilde{W}_{22}\right\|W_{22m}$ terms and rearranging the following is obtained.

$$\begin{aligned}\dot{V} \le & -\left(\lambda_{\min}(k_1) - \left\|B_1\right\|^2\right)\left\|\bar{e}_2\right\|^2 + \left\|\bar{e}_2\right\|W_{12m}\phi_{12m} + \left\|\bar{e}_2\right\|\epsilon_{1m} \\ & -\left(\lambda_{\min}(k_2) - \left\|B_2\right\|^2\right)\left\|\bar{e}_4\right\|^2 + \left\|\bar{e}_4\right\|W_{22m}\phi_{22m} + \left\|\bar{e}_4\right\|\epsilon_{2m} \\ & -\left(\frac{1}{2}\left\|\tilde{W}_{11}{}^T\hat{\phi}_{11}\right\| - \left\|\bar{e}_2\right\|\left\|B_1\right\|\right)^2 - \left(\frac{1}{2}\left\|\tilde{W}_{21}{}^T\hat{\phi}_{21}\right\| - \left\|\bar{e}_4\right\|\left\|B_2\right\|\right)^2 \\ & -\frac{1}{2}\left\|\tilde{W}_{11}{}^T\hat{\phi}_{11}\right\|^2 - \frac{1}{2}\left\|\tilde{W}_{21}{}^T\hat{\phi}_{21}\right\|^2 - \left(\frac{1}{2}\left\|\tilde{W}_{11}{}^T\hat{\phi}_{11}\right\| - \phi_{11m}W_{11m}\right)^2 - \left(\frac{1}{2}\left\|\tilde{W}_{21}{}^T\hat{\phi}_{21}\right\| - \phi_{21m}W_{21m}\right)^2 \\ & + \phi_{11m}^2 W_{11m}^2 + \frac{\sigma_{11}}{4}W_{11m}^2 + \frac{\sigma_{12}}{4}W_{12m}^2 + \phi_{21m}^2 W_{21m}^2 + \frac{\sigma_{21}}{4}W_{21m}^2 + \frac{\sigma_{22}}{4}W_{22m}^2 \\ & -\sigma_{11}\left(\left\|\tilde{W}_{11}\right\| - \frac{1}{2}W_{11m}\right)^2 - \sigma_{12}\left(\left\|\tilde{W}_{12}\right\| - \frac{1}{2}W_{12m}\right)^2 \\ & -\sigma_{21}\left(\left\|\tilde{W}_{21}\right\| - \frac{1}{2}W_{21m}\right)^2 - \sigma_{22}\left(\left\|\tilde{W}_{22}\right\| - \frac{1}{2}W_{22m}\right)^2\end{aligned} \tag{137}$$

Now, if the following conditions are met,

$$\begin{aligned}\lambda_{\min}(k_1) &\geq \left\|B_1\right\|^2 \\ \lambda_{\min}(k_2) &\geq \left\|B_2\right\|^2\end{aligned} \tag{138}$$

and

$$\begin{aligned}0 \leq -\left(\lambda_{\min}(k_1) - \left\|B_1\right\|^2\right)\left\|\bar{e}_2\right\|^2 + \left\|\bar{e}_2\right\|\left(W_{12m}\phi_{12m} + \epsilon_{1m}\right) + \left\|\bar{e}_4\right\|W_{22m}\phi_{22m} + \left\|\bar{e}_4\right\|\epsilon_{2m} \\ + \phi_{11m}^2 W_{11m}^2 + \frac{\sigma_{11}}{4}W_{11m}^2 + \frac{\sigma_{12}}{4}W_{12m}^2 + \phi_{21m}^2 W_{21m}^2 + \frac{\sigma_{21}}{4}W_{21m}^2 + \frac{\sigma_{22}}{4}W_{22m}^2\end{aligned} \tag{139}$$

or

$$\begin{aligned}0 \leq -\left(\lambda_{\min}(k_2) - \left\|B_2\right\|^2\right)\left\|\bar{e}_4\right\|^2 + \left\|\bar{e}_4\right\|\left(W_{22m}\phi_{22m} + \epsilon_{2m}\right) + \left\|\bar{e}_2\right\|W_{12m}\phi_{12m} + \left\|\bar{e}_2\right\|\epsilon_{1m} \\ + \phi_{11m}^2 W_{11m}^2 + \frac{\sigma_{11}}{4}W_{11m}^2 + \frac{\sigma_{12}}{4}W_{12m}^2 + \phi_{21m}^2 W_{21m}^2 + \frac{\sigma_{21}}{4}W_{21m}^2 + \frac{\sigma_{22}}{4}W_{22m}^2\end{aligned} \tag{140}$$

the Lyapunov function is less than or equal to zero. That is to say that it is bounded. Using the expressions in Eqs. (139) and (140) bounds on the estimation errors can be found as

$$\left\|\bar{e}_2\right\| \geq \frac{\left(W_{12m}\phi_{12m} + \epsilon_{1m}\right)}{2\left(\lambda_{\min}(k_1) - \left\|B_1\right\|^2\right)} + \left[\frac{\left(W_{12m}\phi_{12m} + \epsilon_{1m}\right)^2}{4\left(\lambda_{\min}(k_1) - \left\|B_1\right\|^2\right)^2} + \frac{\left(\left\|\bar{e}_4\right\|\left(W_{22m}\phi_{22m} + \epsilon_{2m}\right) + W_t\right)}{\left(\lambda_{\min}(k_1) - \left\|B_1\right\|^2\right)}\right]^{\frac{1}{2}} \tag{141}$$

and

$$\left\|\bar{e}_4\right\| \geq \frac{\left(W_{22m}\phi_{22m} + \epsilon_{2m}\right)}{2\left(\lambda_{\min}(k_2) - \left\|B_2\right\|^2\right)} + \left[\frac{\left(W_{22m}\phi_{22m} + \epsilon_{2m}\right)^2}{4\left(\lambda_{\min}(k_2) - \left\|B_2\right\|^2\right)^2} + \frac{\left(\left\|\bar{e}_2\right\|\left(W_{12m}\phi_{12m} + \epsilon_{1m}\right) + W_t\right)}{\left(\lambda_{\min}(k_2) - \left\|B_2\right\|^2\right)}\right]^{\frac{1}{2}} \tag{142}$$

where

$$W_t = \phi_{11m}^2 W_{11m}^2 + \frac{\sigma_{11}}{4} W_{11m}^2 + \frac{\sigma_{12}}{4} W_{12m}^2 + \phi_{21m}^2 W_{21m}^2 + \frac{\sigma_{21}}{4} W_{21m}^2 + \frac{\sigma_{22}}{4} W_{22m}^2 \qquad (143)$$

In a similar way, bounds on the neural network weights can be found using Eq. (137). The bounds are found to be

$$\begin{aligned} \left\| \tilde{W}_{11} \right\| &\geq \left( 2\phi_{11m}^{-2} \left[ \left\| \bar{e}_2 \right\| \left( W_{12m}\phi_{12m} + \epsilon_{1m} \right) + \left\| \bar{e}_4 \right\| \left( W_{22m}\phi_{22m} + \epsilon_{2m} \right) + W_t \right] \right)^{\frac{1}{2}} \\ \left\| \tilde{W}_{12} \right\| &\geq \frac{1}{2} W_{12m} + \left[ \sigma_{12}^{-1} \left( \left\| \bar{e}_2 \right\| \left( W_{12m}\phi_{12m} + \epsilon_{1m} \right) + \left\| \bar{e}_4 \right\| \left( W_{22m}\phi_{22m} + \epsilon_{2m} \right) + W_t \right) \right]^{\frac{1}{2}} \end{aligned} \qquad (144)$$

and

$$\begin{aligned} \left\| \tilde{W}_{21} \right\| &\geq \left( 2\phi_{21m}^{-2} \left[ \left\| \bar{e}_2 \right\| \left( W_{12m}\phi_{12m} + \epsilon_{1m} \right) + \left\| \bar{e}_4 \right\| \left( W_{22m}\phi_{22m} + \epsilon_{2m} \right) + W_t \right] \right)^{\frac{1}{2}} \\ \left\| \tilde{W}_{22} \right\| &\geq \frac{1}{2} W_{22m} + \left[ \sigma_{22}^{-1} \left( \left\| \bar{e}_2 \right\| \left( W_{12m}\phi_{12m} + \epsilon_{1m} \right) + \left\| \bar{e}_4 \right\| \left( W_{22m}\phi_{22m} + \epsilon_{2m} \right) + W_t \right) \right]^{\frac{1}{2}} \end{aligned} \qquad (145)$$

Using this proof it is shown that the state tracking errors $\bar{e}_2, \bar{e}_4$ are bounded, and since they are connected to the errors $\bar{e}_1, \bar{e}_3$, all errors are bounded, and the neural network weights estimation errors $\tilde{W}_{11}, \tilde{W}_{12}, \tilde{W}_{21}$, and $\tilde{W}_{22}$ are bounded. This extra control can be computed online to estimate and compensate for the unmatched uncertainties in the TWIP system.

This method can easily be extended to the class of more general nonlinear systems with unmatched uncertainties

$$\begin{aligned} \dot{x}_1 &= x_2 \\ \dot{x}_2 &= f_1(\mathrm{x}_1, x_2, \mathrm{x}_4) + f_2(\mathrm{x}_1, x_2, \mathrm{x}_4) x_3 + g_1(x_1, x_2) u + d_1(x_1, x_2, x_3, x_4) \\ \dot{x}_3 &= x_4 \\ \dot{x}_4 &= f_3(x_1, x_2, x_3, x_4) + g_2(x_1, x_2, x_3, x_4) u + d_2(x_1, x_2, x_3, x_4) \end{aligned} \qquad (146)$$

# 7. TWO WHEELED INVERTED PENDULUM SIMULATION

This section outlines the simulation of a two wheeled inverted pendulum used to show the effectiveness of the Discrete Modified State Observer. This section describes the simulation preliminaries: discretization, control, system uncertainty, noise, nonlinearities, and the discrete-time Kalman filter.

## 7.1. DISCRETIZATION

The Discrete Modified State Observer works for a discrete-time system, so the continuous time equations of motion must first be discretized. This can be easily done using the matrix exponential [50]. To convert the continuous state-space system

$$\dot{\mathbf{x}} = A\mathbf{x} + B\mathbf{u} \tag{147}$$

into the discrete system

$$\mathbf{x}_{dt,k+1} = F\mathbf{x}_{dt,k} + G\mathbf{u}_k \tag{148}$$

the matrix exponential is used where $F = \exp(A\Delta t)$ , $G = F\left[I - \exp(-A\Delta t)\right]A^{-1}B$ , and $\Delta t$ is the discretization time step.

## 7.2. CONTROL

The linear control law used in all simulations is given by

$$\mathbf{u}_k = -K_{LQR}(\hat{\mathbf{x}}_k - \mathbf{x}_{des,k}) \tag{149}$$

where

$$K_{LQR} = (\bar{R} + G^T P G)^{-1} G^T P F \tag{150}$$

Since the majority of the system uncertainty comes from the linearization errors, the uncertainty is greatest when the tilt angle is farthest from zero. Therefore, values used in the weighting matrices are chosen to not drive the system to a stable solution as fast as possible, but to give the system small oscillations so that the system uncertainty to be measured by the DMSO will have some dynamics to it. The control effort is also weighted to limit the maximum control in an effort to simulate the real system, where there is control saturation at a point. The parameters used in all simulations are

$$\bar{R} = 1000 \tag{151}$$

$$\bar{Q} = \begin{bmatrix} 10 & 0 & 0 & 0 \\ 0 & 10 & 0 & 0 \\ 0 & 0 & 1e-5 & 0 \\ 0 & 0 & 0 & 1e-4 \end{bmatrix} \tag{152}$$

## 7.3. UNCERTAINTY

In the following cases the uncertainty that the DMSO is estimating is the error between the true state from the nonlinear equations of motion with the true parameter values, $\mathbf{x}_{true}$, and the discrete linear system output with possibly incorrect values, $\mathbf{x}_{dt}$. This is defined as

$$\mathbf{f}(\mathbf{x}_k) = \mathbf{x}_{true,k} - \mathbf{x}_{dt,k} \tag{153}$$

## 7.4. SIMULATED NOISE

Three sensors are simulated, replicating the physical system: an accelerometer, gyroscope, and motor encoder. Accelerometers and gyroscopes have many sources of

error, but can be modeled with three main sources of error [51, 52]. Both sensors are modeled by

$$m = m_r + c_r + b_r + w_r \tag{154}$$

where $m$ is the sensor reading, $m_r$ is the true measurement, $c_r$ is a constant bias that changes on each initialization of the sensor, $b_r$ is called the walking bias, and $w_r$ is a white noise process with variance $\sigma_r^2$. The constant bias is easily remedied and can be removed on initialization of the sensor with digital logic. The other two remain, and must be considered when simulating the sensors. White noise is the main source of noise in low cost sensors, and can easily be characterized by finding the variance of a long run of data with the sensor stationary. The walking bias is a little more complex, but can be modeled as a first order Markov process with the dynamics

$$\dot{b}_r = -\frac{1}{\tau_r} b_r + w_{b_r} \tag{155}$$

where $\tau_r$ is the time constant of the Markov process and $w_{br}$ is white noise with variance $\sigma_{br}^2$. The walking bias captures the tendency of the sensor bias to walk, or drift over time. The parameters of the walking bias can be obtained by performing an autocorrelation analysis of a long run of filtered data, for more details the reader is referred to Flenniken [52]. Allan variance plots are a typical method of analyzing the accuracy of gyroscopes and accelerometers. Figure 7.1 shows an Allan variance plot of measured gyroscope data to simulated data. The results are not perfect, clearly other sources of error are present, but the results are much more accurate and representative than simply adding white noise to the measurements.

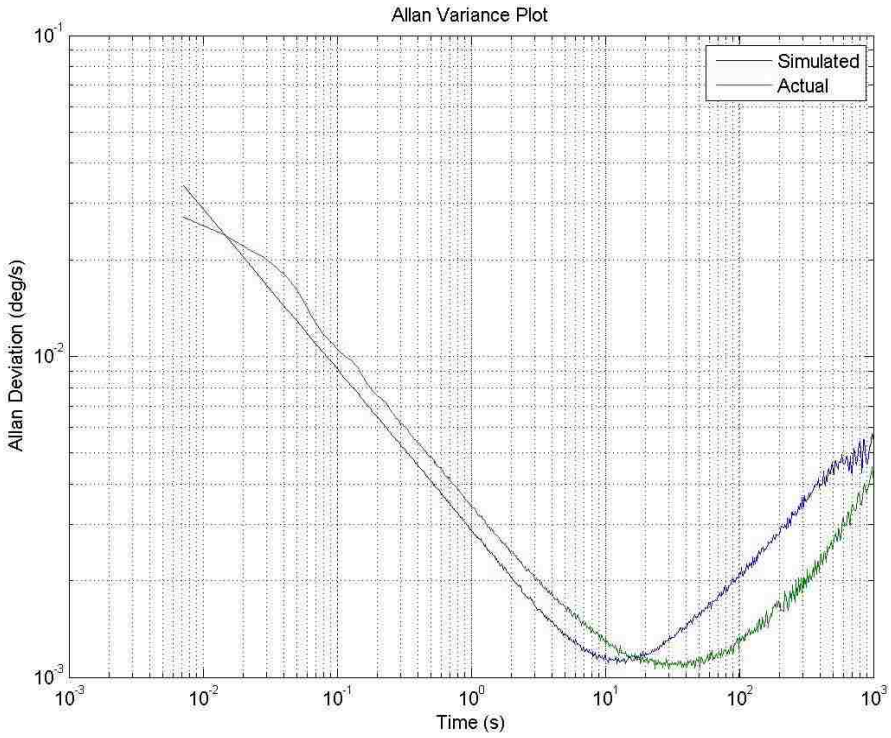


Figure 7.1. Gyroscope Allan Variance Analysis

The motor encoders have a different model of error. They are not plagued with noise, but they are plagued with discretization error. Hall Effect encoders output discrete counts as the motors turn. In combination with a 30:1 gear box, a 64 count per revolution encoder, and 90mm diameter wheels, the motor encoders output discrete steps corresponding to 0.000147m movements. The velocity is obtained by taking a derivative of the position measurements, and is such limited by both the encoder step and the discretization time step. The minimum velocity step is then $0.000147\text{m}/\Delta t$. This discretization error is a problem for the DMSO, as the universal approximation theorem states that it can only estimate a smooth function, because of this the state estimate can do no better than the discretization error of these sensors.

The gyroscope gives directly the tilt rate measurement, but the tilt angle is not directly measureable without a tiltometer. This is remedied by using the accelerometer. The tilt angle can be determined from the direction of the gravity vector by

$$\theta_a = tan^{-1}\left(\frac{-a_x}{a_z}\right) \tag{156}$$

This is a highly inaccurate measurement as all accelerations are measured which corrupt the true measurement of the tilt angle. This is a problem in the physical system, but can easily be remedied as described in Section 4.3.1. This combination of measurements is used to simplify the sensor noise simulation, instead of directly simulating two accelerometers, only the tilt angle is simulated. Parameters used in the noise calculation are summarized in Table 7.1. The values were measured from the sensors used in the TWIP robot implementation.

Table 7.1. Noise Simulation Parameters

| **Parameter** | **Tilt Angle** | **Gyroscope** |
|---|---|---|
| $\sigma_r^2$ | 4.0397e-6 rad | 0.0012 deg/s |
| $\sigma_{br}^2$ | 1.8929e-7 rad/s | 1.0986e-4 deg/s$^2$ |
| $\tau_r$ | 2160s | 1610s |

| **Encoder Measurement** | **Discretization Error** |
|---|---|
| Position | 0.000147 m |
| Velocity | 0.000147/Δt m/s |

## 7.5. NONLINEARITIES

The Pololu DC motors exhibit several obvious nonlinearities that can be accounted for in simulation. Friction is an important nonlinearity to consider, but due to the difficulties of evaluating the forces and obtaining an accurate model, friction was left out of the simulation. The following sections describe the various nonlinearities encountered and simulated.

**7.5.1. Saturation.** DC motors exhibit saturation at the limit of the magnetization of the motor core [47]. Saturation is a standard nonlinearity handled in controls, it is a hard limit on the output after a certain level of input. The input output relationship is shown in Figure 7.2.

Saturation is formally defined as

$$sat(u(k)) = \begin{cases} u_-, & u(k) < d_- \\ u(k), & d_- \le u(k) < d_+ \\ u_+, & u(k) \ge d_+ \end{cases} \tag{157}$$

Figure 7.2. Saturation Nonlinearity

In this case, since the motor is a 12V motor, it is limited by the available power. The robot is powered by a single high discharge 11.1V 20C Lithium Polymer battery. This sets the nominal saturation value at 11.1V. This value will change with use as over time the battery will discharge and the output voltage will change. Measured values of the battery give a fully charged voltage of up to 12.3V, and the battery should not be used when the output is less than 10V to avoid permanent damage to the lithium polymer

battery. In the simulations the nominal value of 11.1V is used as the saturation limit. The DC motor saturation can easily be handled in a simulation by asserting a maximum control signal value. Saturation is important, and actually helpful, in the case of an overzealous control signal. In the optimal sense the best controller would be an infinitely large spike to stabilize the robot in a minimal time. In practice this is not obtainable due to saturation of the control signal and other possible instabilities caused by such a large control effort. In the implemented LQR controller the control signal is weighted heavily to prevent the control signal from reaching the saturation value.

**7.5.2. Deadzone.** Deadzone is another standard nonlinearity and is described by a range of small inputs that do not have an effect on the output signal. The standard deadzone nonlinearity is shown in Figure 7.3, where when the input signal is between $d_-$ and $d_+$ the output is zero.

The deadzone nonlinearity is formally defined by

$$\tau(u(k)) = \begin{cases} m_-(u(k) - d_-), & u(k) < d_- \\ 0, & d_- \le u(k) < d_+ \\ m_+(u(k) - d_+), & u(k) \ge d_+ \end{cases} \tag{158}$$

The output does not necessarily have to have a linear relationship outside of the deadzone, but in the case of the linear DC motor model, the output is considered linear and this is an accurate model. In testing of the DC motors, a slightly different model of the deadzone is necessary to fully describe the nonlinearity. The nonlinearity experienced is a deadzone in combination with jump discontinuities at the edge of the deadzone. This is shown in Figure 7.4.

Figure 7.3. Standard Deadzone Nonlinearity

Figure 7.4. Deadzone with Jump Discontinuities

This jump discontinuity deadzone nonlinearity is defined by

$$\tau_j(u(k)) = \begin{cases} u_+ + m_-(u(k) - d_-), & u(k) < d_- \\ 0, & d_- \le u(k) < d_+ \\ u_- + m_+(u(k) - d_+), & u(k) \ge d_+ \end{cases} \tag{159}$$

The slope of the deadzone output is equal to the torque motor constant, and is worked into the equations of motion. The simplified deadzone model implemented in simulation is applied directly to the control signal and is defined as

$$\tau_s(u(k)) = \begin{cases} u(k), & u(k) < d_- \\ 0, & d_- \le u(k) < d_+ \\ u(k), & u(k) \ge d_+ \end{cases} \tag{160}$$

where $d_-$ and $d_+$ are the initial voltages where the DC motor begins to move. The values used are $d_- = -0.5V$ and $d_+ = 0.5V$.

**7.5.3. Backlash.** Backlash is a common phenomenon found in situations where gears are used. Backlash stems from the spacing of teeth in mechanical gearing systems. If the teeth were machined to mesh completely, the gears would lock up and be unable to move. The amount of play in the gears is called the backlash. In the DC motors a gearbox is attached to the motor output shaft in order to provide a larger torque and slower speed output. These gearboxes give the wheels a small amount of backlash. Figure 7.5 shows the input output relationship for the backlash nonlinearity. Backlash results in a delay in the system motion and it is a first-order velocity-driven dynamical system.

Backlash is formally defined by

$$B(\psi(k), u(k), u(k+1)) = \begin{cases} mu(k) - md_+, & if\ u(k+1) - u(k) > 0\ and\ \psi(k) \le mu(k) - md_+ \\ mu(k) - md_-, & if\ u(k+1) - u(k) < 0\ and\ \psi(k) \ge mu(k) - md_- \\ \psi(k), & otherwise \end{cases} \tag{161}$$

Figure 7.5. Backlash Nonlinearity

Backlash is applied to the system output of the tilt angle in simulation, and the values used are

$$\begin{aligned} m &= 1 \\ d_{+} &= 0.1^{\circ} \\ d_{-} &= -0.1^{\circ} \end{aligned} \tag{162}$$

where $\psi(k)$ is the tilt angle of the system without backlash.

## 7.6. DISCRETE KALMAN FILTER

The Discrete Kalman Filter is used for comparison of the results of the Discrete Modified State Observer. This section provides an outline of the equations and methodology used.

The Kalman filter developed by R. E. Kalman in 1960 provides the statistically optimal state estimate when given a linear system and measurements corrupted by Gaussian, zero-mean, uncorrelated, and white noise [53]. The filter processes

measurements in combination with knowledge of the system dynamics to provide a more accurate estimate of the state. Many versions and extensions of the Kalman filter have been developed since the original theory was described. The version used in this simulation is the discrete-time Kalman filter, derived for optimal state estimation of a discrete-time linear system, and follows from Simon [50].

Assume the dynamic system is given by

$$x_{k+1} = Fx_k + Gu_k + w_k \tag{163}$$

with measurements defined by

$$y_k = Hx_k + v_k \tag{164}$$

where $w_k$ and $v_k$ are zero-mean, uncorrelated, white Gaussian noise with the statistical properties

$$w_k \sim (0, Q) \tag{165}$$

and

$$v_k \sim (0, R) \tag{166}$$

$Q$ is the process noise covariance matrix, and $R$ is the measurement noise covariance matrix. $Q$ and $R$ can be adjusted for optimal performance. States with greater uncertainty in the dynamics can be assigned large values in the process noise covariance matrix which will cause the Kalman filter to trust the measurements more. The measurement noise covariance matrix can be accurately obtained by careful statistical analysis of the available measurements.

The discrete-time Kalman filter is given by the following update equations

$$P_{k+1}^{-} = FP_k^{-}F^T + Q \tag{167}$$

$$K_{k+1} = P_{k+1}^{-}H^T(HP_{k+1}^{-}H^T + R)^{-1} \tag{168}$$

$$\hat{x}_{k+1}^{-} = F\hat{x}_k^{+} + Gu_k \tag{169}$$

$$\hat{x}_{k+1}^{+} = \hat{x}_{k+1}^{-} + K_k(y_k - H\hat{x}_{k+1}^{-}) \tag{170}$$

$$P_{k+1}^{+} = (I - K_kH)P_{k+1}^{-}(I - K_kH)^T + K_kRK_k^T \tag{171}$$

where the – and + superscripts represent a priori and a posteriori quantities. $P$ is the covariance of the estimation error, $K$ is the Kalman gain, and $\hat{x}$ is the Kalman filter state estimate. The update equations are processed sequentially from Eq. (167) to Eq. (171). One thing to note is that the measurements do not have to be obtained at the same rate as the system dynamics, Eq. (170) can be processed at only the time intervals that have available measurements. However, in the simulations the measurements are assumed to be available at the same rate as the system dynamics are processed.

# 8. RESULTS

This section outlines the results of the Discrete Modified State Observer and the neural network controller for unmatched uncertainties. The first section displays the results of several simulation test cases when using LQR control alone. The second section provides simulation results for the extra control formulation described in Section 6.2. The final section displays the experimental results from the two wheeled inverted pendulum robot implementation.

## 8.1. SIMULATION RESULTS

**8.1.1. No Noise, No Parameter Uncertainty.** In this case study there is zero noise in the measurements, and all parameters are known perfectly. Parameter values used are shown in Table 8.1. The saturation nonlinearity was used on the control signal output.

Table 8.1. TWIP Robot Parameters

| Parameter | Value | Units |
|---|---|---|
| $I_p$ | 0.025 | $kgm^2$ |
| $I_w$ | 1.183e-4 | $kgm^2$ |
| $k_e$ | 0.00361 | Vs/rad |
| $k_m$ | 0.11541 | Nm/A |
| l | 0.075 | m |
| $M_p$ | 1.432 | kg |
| $M_w$ | 0.1168 | kg |
| r | 0.045 | m |
| R | 2.5 | Ω |

The DMSO is used as shown in Eq. (3), repeated here for clarity.

$$\hat{\mathbf{x}}_{k+1} = F\hat{\mathbf{x}}_k + G\mathbf{u}_k + B_{MSO}\hat{\mathbf{f}}(\mathbf{x}_k) + K_{MSO}(\mathbf{y}_k - \hat{\mathbf{x}}_k) \tag{172}$$

$F$ and $G$ come from the discretized equations of motion, as described in Section 7.1. The uncertainty is placed in the acceleration terms of the continuous equations of motion, shown in Eq. (79), this is expressed in the $B_{MSO}$ term as

$$B_{MSO} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}^T \tag{173}$$

It is noted that the uncertainty as defined in Eq. (153) has a dimension of *4×1*, but only the second and fourth terms are assigned uncertainty in the DMSO. The first and third terms correspond to states without uncertain dynamics as they are time derivatives. The errors between the nonlinear system and the discrete linearized system are orders of magnitude smaller than the terms with linearized dynamics.

The other selected parameters used in the DMSO are

$$K_{MSO} = I_{4\times 4} \tag{174}$$

$$\Gamma = 0.1 \tag{175}$$

Choosing a larger $K$ matrix will help the decrease the error of the DMSO state estimate while the uncertainty is being estimated. Increasing the adaptation rate, $\Gamma$, will increase the speed of the uncertainty estimation at the expense of state estimation accuracy. The gain values were chosen to replicate the steady state gains of the Kalman filter as a better comparison of the two filters. The adaptation rate was chosen as a balance between speed of uncertainty estimation and accuracy. The basis functions are chosen as

$$\phi(\mathbf{x}_k) = \begin{bmatrix} 1 & \text{tansig}(x_{1,k}) & \text{tansig}(x_{2,k}) & \text{tansig}(x_{3,k}) & \text{tansig}(x_{4,k}) \end{bmatrix}^T$$

where $\text{tansig}(\cdot)$ represents the hyperbolic tangent sigmoid function, a standard choice used in neural network basis functions [32].

The discrete Kalman filter is used for comparison. The following parameters were used in the calculations

$$Q = \begin{bmatrix} 0.0001 & 0 & 0 & 0 \\ 0 & 0.01 & 0 & 0 \\ 0 & 0 & 0.0001 & 0 \\ 0 & 0 & 0 & 0.01 \end{bmatrix} \tag{176}$$

$$R = \begin{bmatrix} 1e-6 & 0 & 0 & 0 \\ 0 & 1e-5 & 0 & 0 \\ 0 & 0 & 4e-6 & 0 \\ 0 & 0 & 0 & 2e-5 \end{bmatrix} \tag{177}$$

$$P_0 = 0.01 I_{4\times 4} \tag{178}$$

where $Q$ is the process noise covariance matrix, $R$ is the measurement noise covariance matrix, and $P_0$ is the state estimate covariance matrix. The values of $Q$ were chosen to weight the uncertainty in the acceleration terms, placing larger values in the second and fourth terms in comparison to the first and third terms. The values of $R$ come from the statistical analysis of the available measurements for the tilt angle and tilt rate. The position and velocity covariance were estimated from the discretization error of the motor encoders, giving a larger covariance to the velocity measurement as the discretization error is greater. While in this simulation there is no noise, there was minimal change by reducing the measurement covariance values, so these values were kept.

The discretization time step is chosen as $\Delta t = 0.01s$, this time step was chosen to match the time step on the physical robot system. The initial estimates for the Kalman filter and MSO are set to zero, and the true initial system states are set as

$$\mathbf{x}_{true,0} = \begin{bmatrix} 1m & 0.3m/s & 10\,\text{deg} & 1\,\text{deg/s} \end{bmatrix}^T \tag{179}$$

The state estimates are shown alongside the true state values in Figure 8.1. Both the DMSO and the Kalman filter are able to accurately estimate the system states. The linear controller is shown to be effective in driving the system states to zero after a period of time, reaching the desired states in 15 seconds. To better compare the state estimates, the first second of the state estimation errors are shown in Figure 8.2. In all cases the DMSO is able to capture the true system state to a higher degree of accuracy in a shorter time period. It is noted that the Kalman filter does very well, but that is to be expected for this case where there is little system uncertainty. What is interesting to see, shown in Figure 8.3 in an error logarithmic plot, is that after the initial DMSO convergence to the true tilt rate, the Kalman filter converges to obtain a better estimate of the state. Finally, the system uncertainties, given by Eq. (153) are shown in Figure 8.4. The uncertainty estimate accurately captures the error between the true nonlinear system and the linearized discrete system. The uncertainty is shown to not entirely accurately obtain the system uncertainty with errors up to 25%. This is attributed to the bound on the estimation error, as there is a bound on the accuracy of the DMSO. Even with this slight inaccuracy, the DMSO is shown to be an effective state estimator in this system.

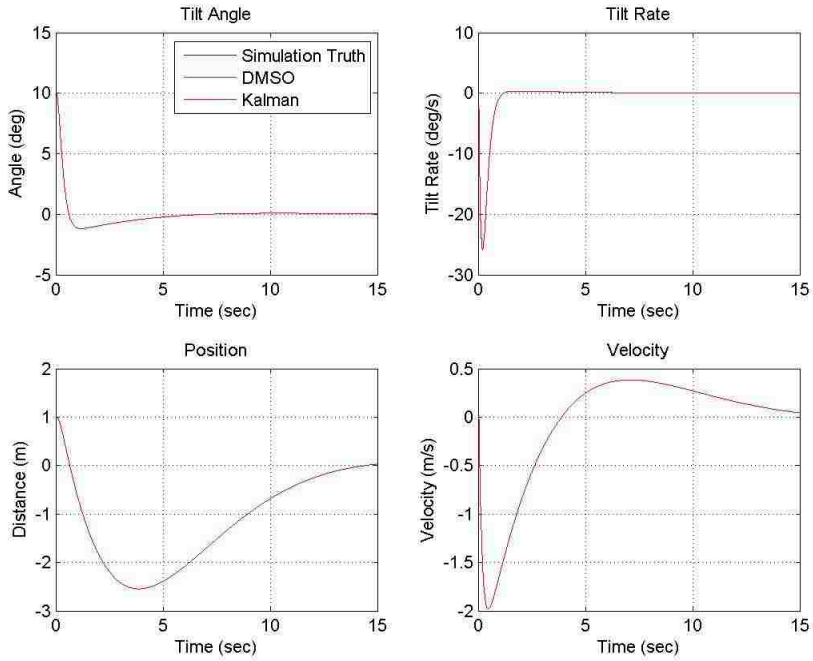


Figure 8.1. No Noise, No Uncertainty TWIP States

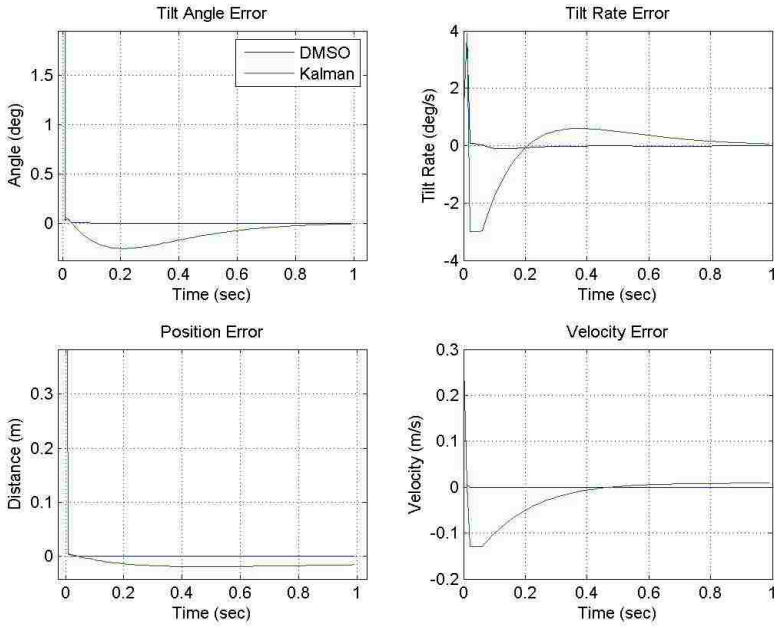


Figure 8.2. No Noise, No Uncertainty TWIP State Estimate Error

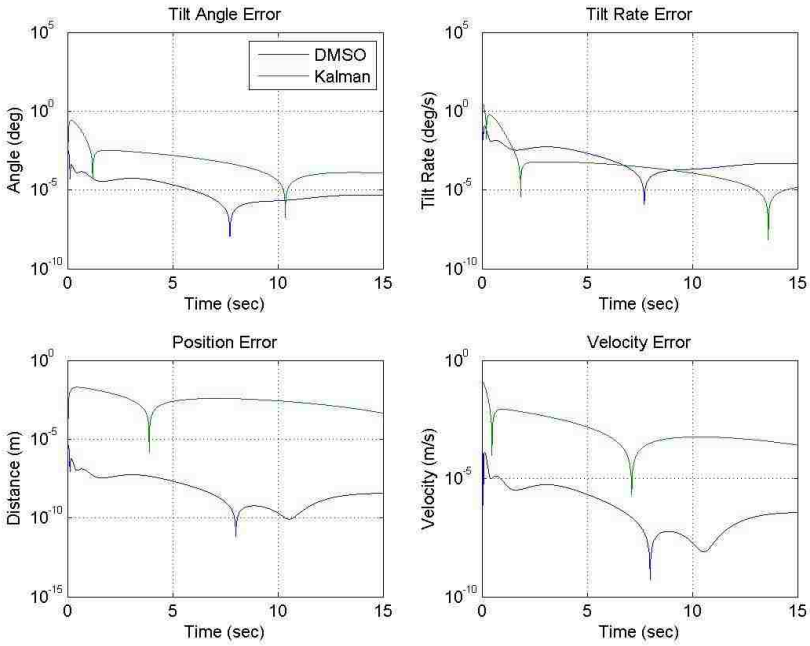


Figure 8.3. No Noise, No Uncertainty TWIP State Estimate Error Logplot

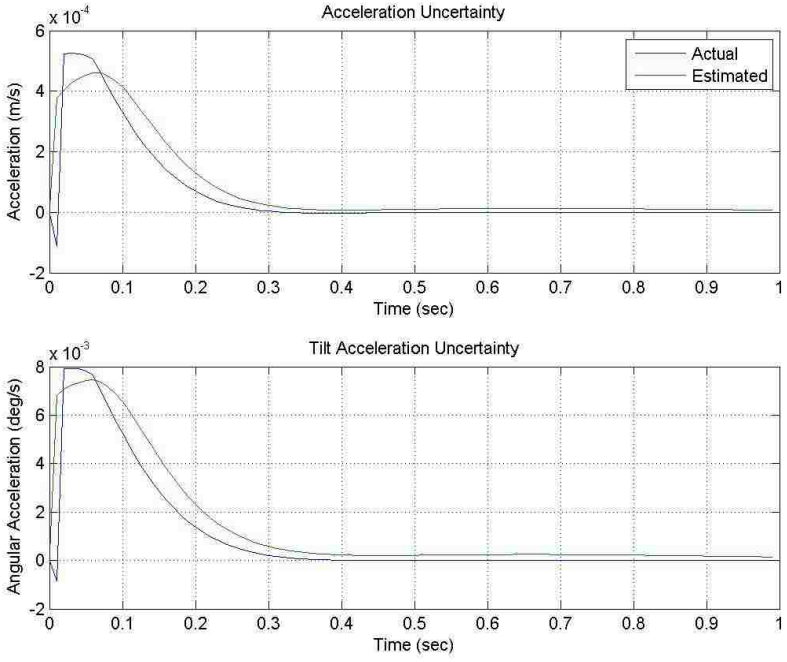


Figure 8.4. No Noise, No Uncertainty DMSO Uncertainty Estimates

**8.1.2. Parameter Uncertainty, No Noise.** In this study all parameters are kept equal from the previous section, with the exception of the robot system parameters, which are varied to create additional system uncertainty. The estimated system parameters, used in calculating the Kalman filter estimate and the DMSO estimates, are the same as in the previous section, shown in Table 8.1. The parameters for the true system, calculated from the nonlinear equations of motion, are adjusted and given in Table 8.2. Figure 8.5 shows the system states where it is shown that the controller is no longer as effective as the first simulation study due to the parameter uncertainty; the LQR controller is now tuned for an incorrect system. The tilt angle overshoots the desired value by 5 degrees and causes oscillations to be present that take longer to stabilize in this case. However, the Kalman filter still does a decent job at estimation, the error is small enough to not be shown in this figure. Figure 8.6 shows the state estimation error, where it is shown again that the DMSO provides better estimates than the Kalman filter in all states, obtaining on average a state estimate that is an order of magnitude more accurate. However, as before, Figure 8.7 shows the error logarithmic plot over a longer time where it is shown that the Kalman filter does obtain a better estimate of the tilt rate after five seconds. Finally, Figure 8.8 shows the system uncertainties. Even with the issue in the tilt angle rate estimate, the system uncertainties are accurately estimated. One thing to note here is the small anomaly on the tilt acceleration uncertainty estimate at 1.5 seconds. This is attributed to the large time step used, as will be shown.

Figure 8.8 shows a good example of the adaptation rate set too low for high accuracy of the uncertainty estimates. Figure 8.9 shows the same simulation run with the adaptation rate set to $\Gamma = 0.5$. As shown, the error in the uncertainty estimates are decreased, and the estimate converges to the true uncertainty in .1 seconds as compared to .25 seconds previously. The issue with the tilt acceleration uncertainty estimate at 1.5 seconds is still present. There is a small period of time where the uncertainty estimate diverges from the true value before returning. However, when decreasing the time step size to $\Delta t = 0.001s$, the results shown in Figure 8.10 are obtained. The system response is similar, but the estimation errors are much smaller for both the Kalman filter and the DMSO, on average one order of magnitude smaller. The uncertainty estimates in Figure 8.11 are much more accurate, on average two orders of magnitude more accurate, and

now without the unwanted errors seen in Figure 8.8 and Figure 8.9. Clearly a smaller time step is beneficial for discrete time systems. A trade study must be done between accuracy and computational expense when selecting the discrete time step and parameters for the DMSO.

Table 8.2. Modified TWIP Robot Parameters

| **Parameter** | **Value** | **Units** |
|---|---|---|
| $I_p$ | 0.025 | $kgm^2$ |
| $I_w$ | 1.774e-4 | $kgm^2$ |
| $k_e$ | 0.0034 | Vs/rad |
| $k_m$ | 0.2885 | Nm/A |
| l | 0.075 | m |
| $M_p$ | 1.5752 | kg |
| $M_w$ | 0.1752 | kg |
| r | 0.045 | m |
| R | 3 | Ω |

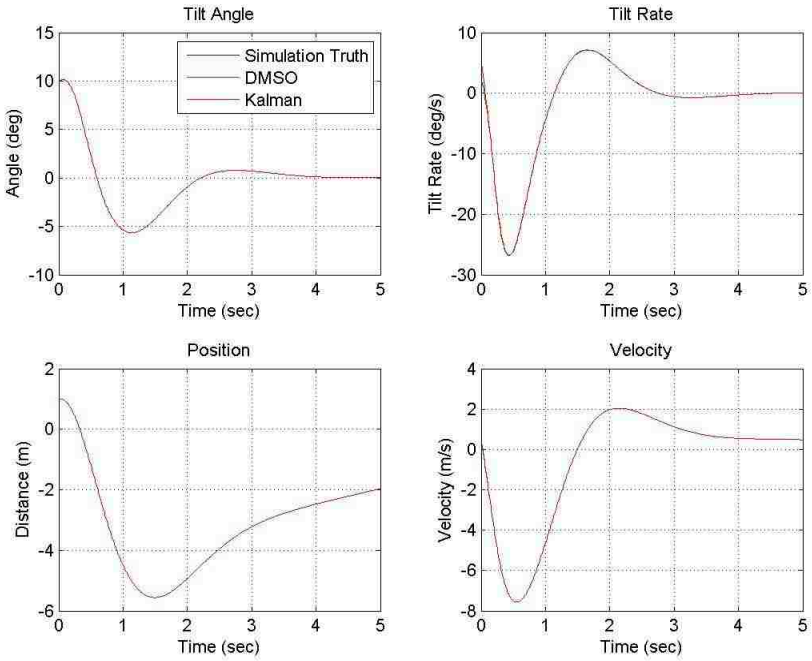


Figure 8.5. No Noise with Uncertainty TWIP States

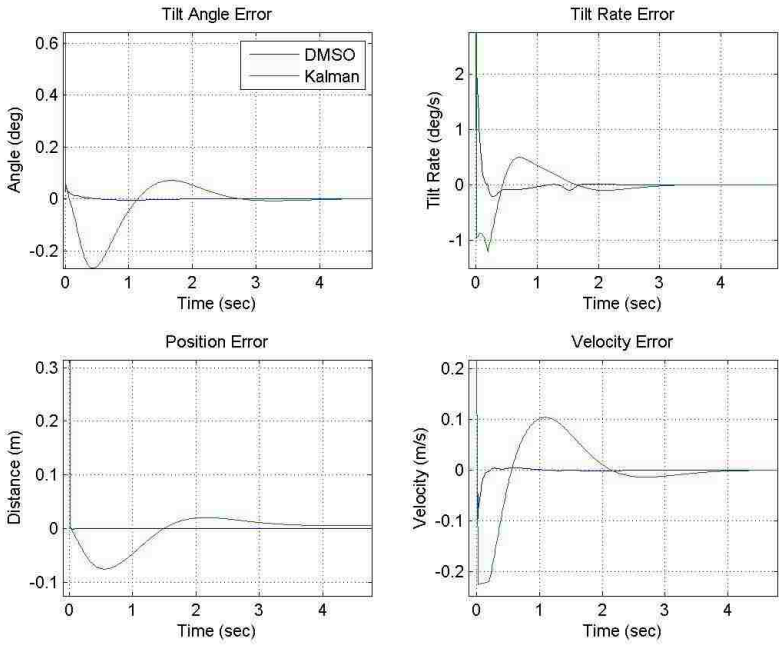


Figure 8.6. No Noise with Uncertainty TWIP State Estimate Error

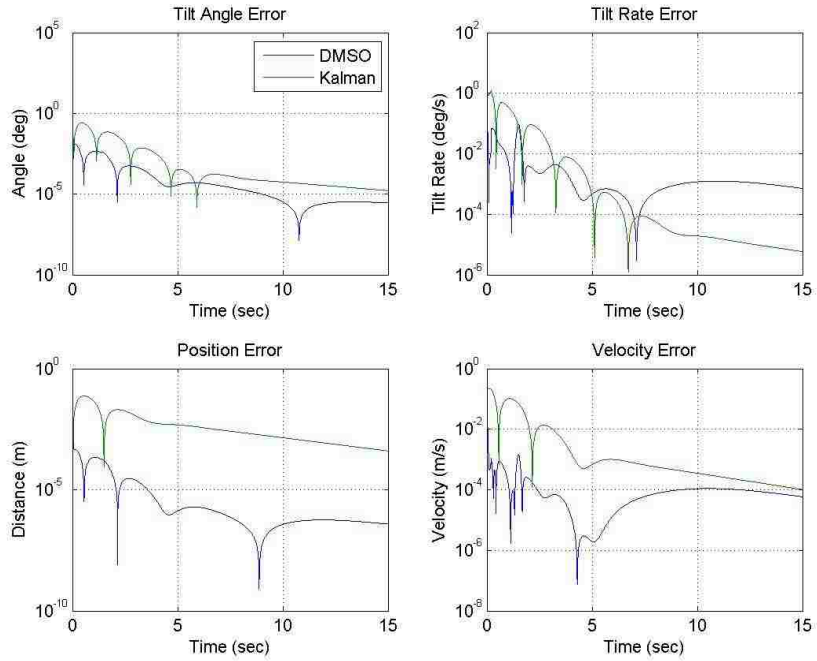


Figure 8.7. No Noise with Uncertainty TWIP State Estimate Error Logplot

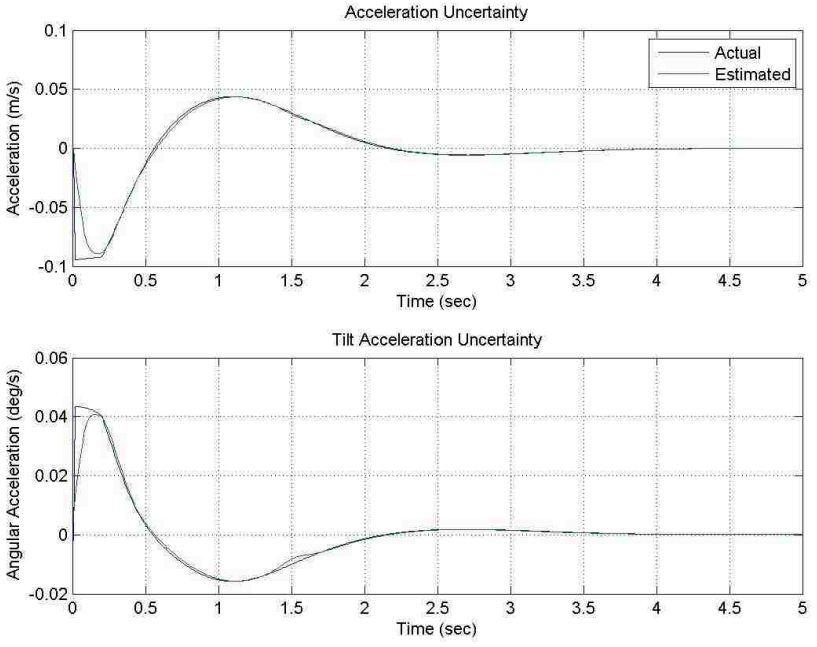


Figure 8.8. No Noise with Uncertainty DMSO Uncertainty Estimates

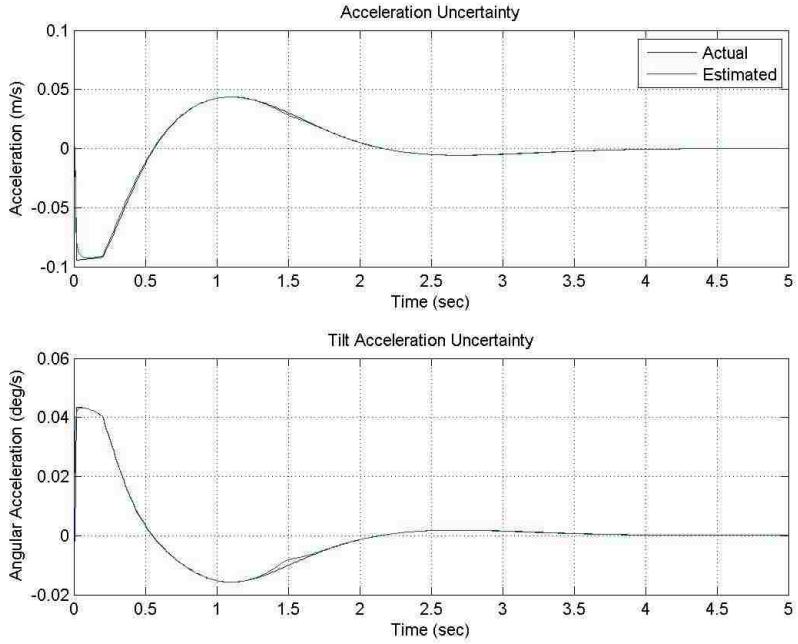


Figure 8.9. No Noise with Uncertainty DMSO Uncertainty Estimates, Γ = .5

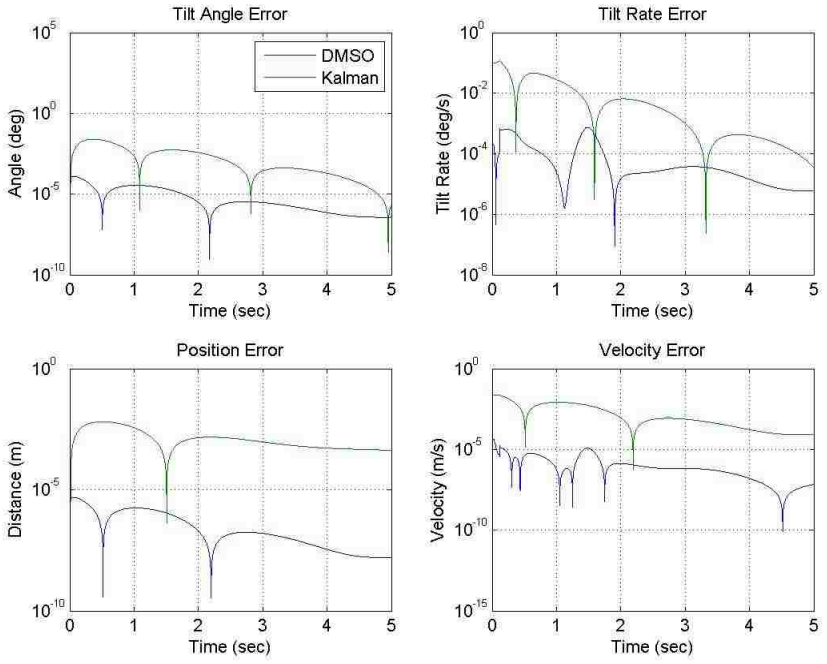


Figure 8.10. No Noise with Uncertainty TWIP State Estimate Error, Δt = .001s

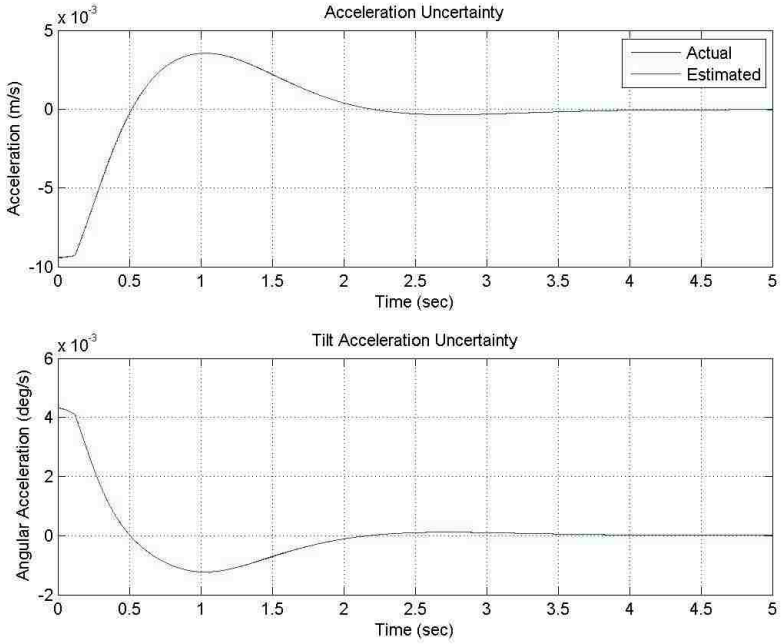


Figure 8.11. No Noise with Uncertainty DMSO Uncertainty Estimates, Δt = .001s

**8.1.3. Noisy Measurements with Parameter Uncertainty.** In this final case study noise is introduced into the simulation. The simulation includes this noise, along with the same parameter uncertainty that was present in the previous section. The time step is set to $\Delta t = 0.01s$, in this case decreasing the time step amplifies the effect of the noise in the error of the state estimates and the system uncertainty. The adaptation rate is set to $\Gamma = 0.1$ this is a balance between the effects of the noise and the errors in the estimation. Just as the Kalman filter must be tuned for each system, the DMSO also needs to be lightly tuned, the benefit here is that there are fewer parameters to tune, and no detailed stochastic analysis has to be performed to obtain accurate parameters, such as that needed for the Kalman filter measurement covariance values. The DMSO has gain values that can be adjusted, but they are less sensitive. Values just need to be found to stabilize the state estimate error while the neural network is adjusting to the uncertainty, as shown in these stimulation the same gain values were used while only the adaptation rate was adjusted.

Figure 8.12 displays the system states, once again the oscillations are present that stem from the LQR controller tuned for an inaccurate system. Noise is also visible in the tilt angle and tilt rate states. The discretization error is present in the position and velocity, but the steps are too small to be seen in this figure. Figure 8.13 shows the state estimation error. The state estimate errors of the DMSO are shown to converge to the true values in a shorter period of time as compared to the Kalman filter, for example in the tilt rate the Kalman filter takes 1.3 seconds to converge to within the bounds of the measurement noise, where the DMSO takes .15 seconds. Both filters do end up with equal accuracy after 4 seconds in all states except the position, where the DMSO provides an estimate error two orders of magnitude smaller than the Kalman filter. What is important to remember is that in addition to these state estimates, the DMSO also estimates the system uncertainty, shown in Figure 8.14. Once again the DMSO is capable of accurately estimating the system uncertainty, even in the presence of measurement noise. The error is on the order of the measurement noise, but the dynamics are still captured. It could be possible to use this uncertainty estimate to back out unmodeled dynamics in the system model, with a proper analysis one may even be able to determine the incorrect parameters. The uncertainty estimate could also be used cancel uncertainty with the controller, this is what was done by Yang et al. [31] in the control of an uncertain nonlinear electrohydraulic system using a simplified uncertain linear model. The main difference here is the extension into discrete-time, which allows simplified and more accurate execution in digital systems.

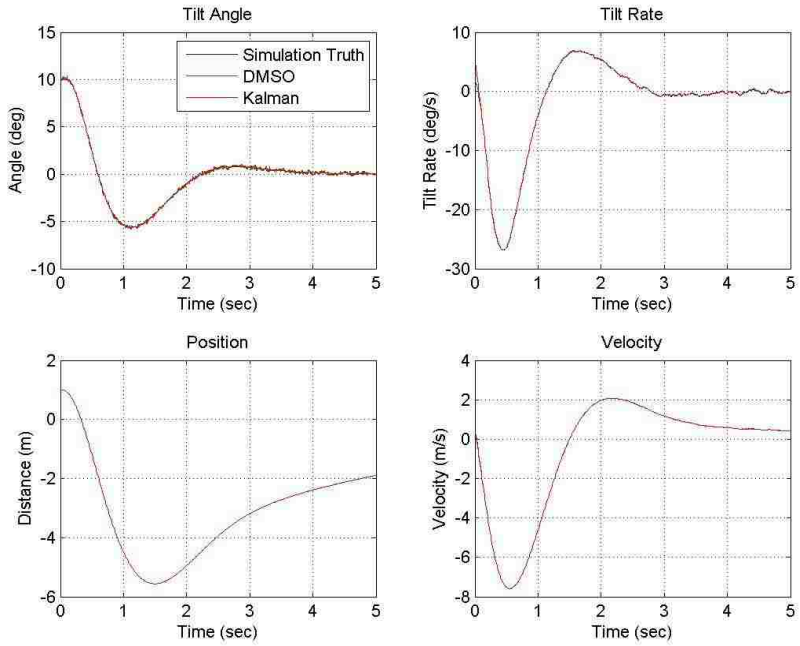


Figure 8.12. Noise and Uncertainty TWIP States

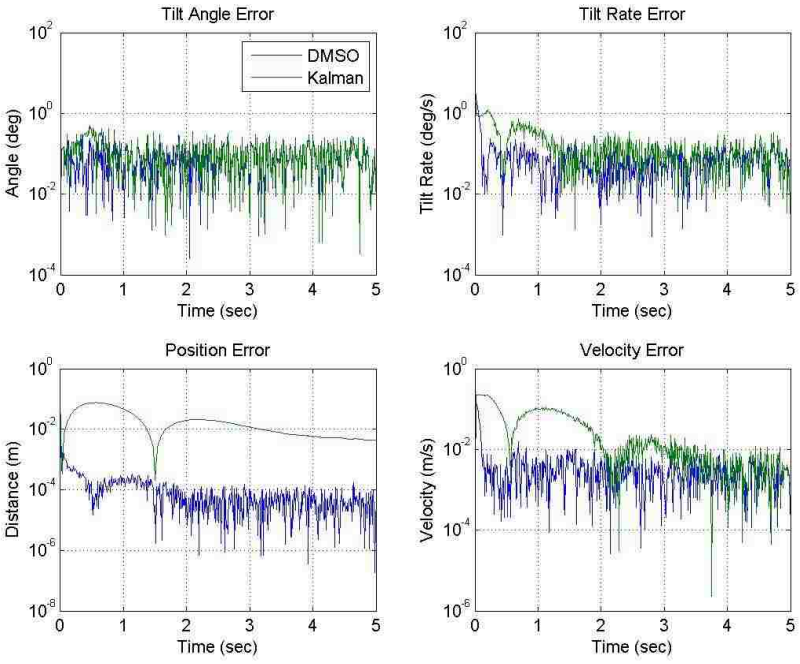


Figure 8.13. Noise and Uncertainty State Estimation Error

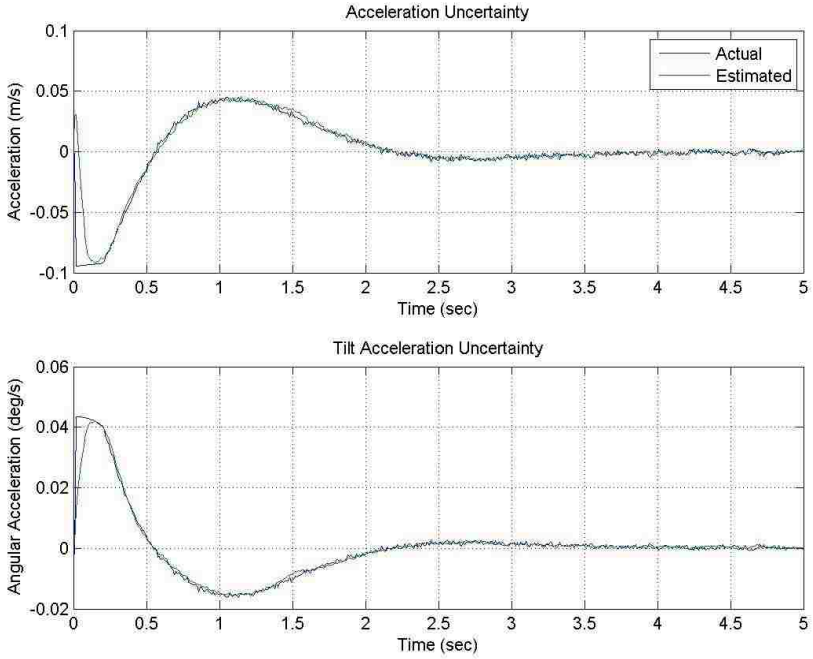


Figure 8.14. Noise and Uncertainty DMSO Uncertainty Estimation

## 8.2. EXTRA CONTROL RESULTS

The controller designed in Section 6.2 is simulated using the TWIP system to show the effectiveness of it in the presence of parameter uncertainty, unmodeled dynamics, and actuator nonlinearities. Several simulations are performed including each of these forms of uncertainties. The base controller used is the same LQR control as in the previous simulations, however the weighting matrices were modified to the following values. These values allow for higher performance of the controller, driving the states to the desired values in a shorter period of time.

$$\bar{R} = 1 \tag{180}$$

$$\bar{Q} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & .1 & 0 \\ 0 & 0 & 0 & .1 \end{bmatrix} \tag{181}$$

The system is commanded to track a desired trajectory. The system is given a desired constant velocity of $0.1m/s$, and the position is derived from this constant value. The desired tilt angle is calculated from the linear system assuming it is of the form

$$\begin{aligned} x_{1d,k+1} &= x_{1d,k+1} + \Delta t x_{1d,k} \\ x_{2d,k} &= 0.1 \\ \dot{x}_{2d,k} &= A_1 x_{2d,k} + A_2 x_{3d,k} + B_1 u_{d,k} = 0 \\ \dot{x}_{3d,k} &= x_{4d,k} = 0 \\ \dot{x}_{4d,k} &= A_3 x_{2d,k} + A_4 x_{3d,k} + B_2 u_{d,k} = 0 \end{aligned} \tag{182}$$

where the coefficients $A_1, A_2, A_3, A_4, B_1, B_2$, come from the linear system in Eq. (79). This allows the desired tilt angle to be calculated using

$$\begin{bmatrix} x_{3d} \\ u_d \end{bmatrix} = -\begin{bmatrix} A_2 & B_1 \\ A_4 & B_2 \end{bmatrix}^{-1} \begin{bmatrix} A_1 \\ A_3 \end{bmatrix} x_{2d} \tag{183}$$

The extra control is calculated according to the method in Section 6.2 using two-layer neural networks with 10 hidden neurons. The first neural network is given the inputs $1, x_1, x_2, x_3, x_4, e_1, e_2, x_{1d}, \mathrm{x}_{2d}$, and $u$. The second neural network is given the inputs $1, x_1, x_2, x_3, x_4, e_1, e_2, \bar{e}_3, \bar{e}_4, x_{1d}, \mathrm{x}_{2d}$, and $u_{nom}$. The parameters used are $k_1 = k_2 = 2.5$, and $B_1 = B_2 = [1]_{10x1}$. The learning rates are selected as $\gamma_{11} = \gamma_{12} = \gamma_{21} = \gamma_{22} = .1$ and $\sigma_{11} = \sigma_{12} = \sigma_{21} = \sigma_{22} = .01$. Tangent sigmoid activation functions are used as in the DMSO with randomly selected input layer weights $V_1$ and $V_2$.

**8.2.1. Extra Control with Parameter Uncertainty.** In the first simulation case parameter uncertainty is used to create uncertainty, along with linearization errors. The initial state is selected as

$$\mathbf{x}_{true,0} = \begin{bmatrix} 1m & 0.3m/s & 10\,\mathrm{deg} & 1\,\mathrm{deg/s} \end{bmatrix}^T \tag{184}$$

Parameters are varied as they were in the DMSO simulations, values used are shown in Table 8.1 and Table 8.2. The true parameters used in the linear system of equations are hereby called the nominal system. LQR control is used to calculate the optimal control for the nominal system. The modified parameters are used in the nonlinear equations of motion, which will be called the perturbed system. To compare the effectiveness of the extra control signal, the nominal system is compared to the perturbed system, both with, and without the extra control added. Desired state values are plotted along with the states. Control saturation is used on all control signals.

Figure 8.15 shows the states of the simulation in the case with only parameter uncertainty. All controllers work effectively to drive the state to the desired trajectory. It is shown that the extra control results in a more effective controller for the system, shown in the trajectory labeled 'Perturbed system with ue.' The states are driven to the desired values in a shorter period of time, even in the presence of parameter uncertainty, as compared to the optimal control calculated for the linear system. This is in part due to the added control gains $k_1$ and $k_2$, but is also assisted by the neural network estimates used in the extra control calculation. Figure 8.16 shows the calculated control values for the system. The optimal control is the LQR based control for the linear system with the true parameters. The nominal control is the LQR based control signal calculated for the perturbed system, and the extra control signal is the calculated extra control. The total control signal is the control used in the perturbed system with extra control, labaled 'Perturbed system with ue' in Figure 8.15.

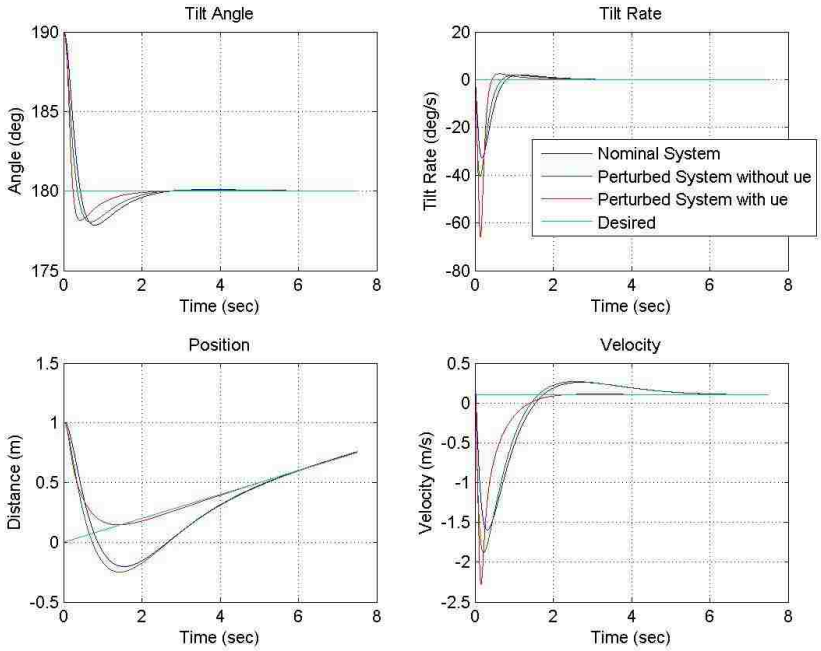


Figure 8.15. Extra Control States with Parameter Uncertainty

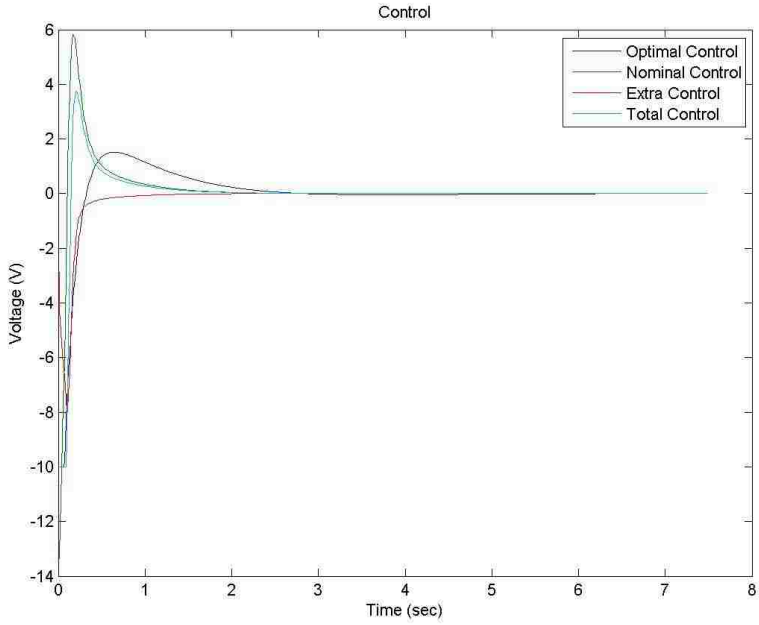


Figure 8.16. Extra Control with Parameter Uncertainty

**8.2.2. Parameter Uncertainty with Unmodeled Dynamics.** This simulation is the same as the previous section with one difference. Unmodeled dynamics are included in the nonlinear equations of motion. The unmodeled dynamics are defined as

$$d_1(\mathbf{x}) = \frac{1}{2}\dot{x}\sin\left(\frac{180\,\mathrm{x}}{\pi}\right)$$
$$d_2(\mathbf{x}) = \frac{1}{4}\dot{x}^2 \tag{185}$$

The states shown in Figure 8.17 are labeled the same as the previous section. In this case the perturbed system is not nearly as effective in driving the system to the desired trajectory. The perturbed system controlled with the extra control tends towards the nominal system, showing that the extra control is accurately estimating the uncertainties in the system and removing them from the dynamics. Due to the unmodeled dynamics, the LQR controller is much less capable of effective control. In fact, it is easy to create a system that is unstable using only LQR control. By increasing the gains on the unmodeled dynamics in Eq. (185) from $\frac{1}{2}$ to $\frac{3}{4}$ and from $\frac{1}{4}$ to $\frac{2}{5}$ the results in Figure 8.18 are obtained. It is shown that while the LQR controller is unstable when attempting to control the perturbed system, the neural network controller is capable of estimating and compensating for the unmodeled dynamics. The control values are shown in Figure 8.19. It is shown that the control signals are stable, and that over time the extra control signal for the perturbed system tends towards the nominal control value, indicating that the unmodeled dynamics have been compensated for and removed from the system dynamics.

**8.2.3. Deadzone Nonlinearity.** This simulation study removes the unmodeled dynamics and parameter uncertainty, and instead includes control deadzone as defined in Section 7.5.2. Control saturation is also included, but is of little effect. Figure 8.20 shows the states of the system. It is shown that with control deadzone highly accurate control of the system is impossible. The optimal controller is no longer optimal due to the unaccounted for nonlinearity. However, it is shown that the neural network controller proposed is capable of partially eliminating the effects of the control deadzone. The extra

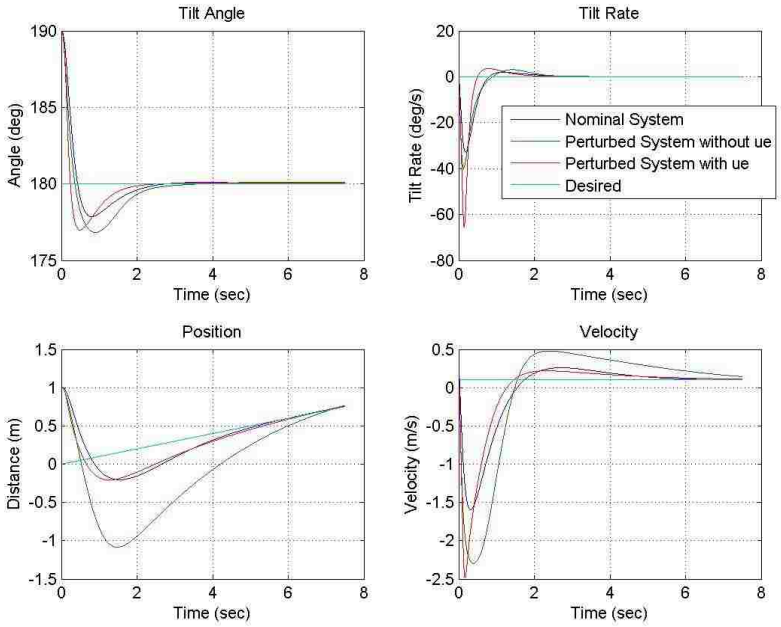


Figure 8.17. Extra Control States with Unmodeled Dynamics

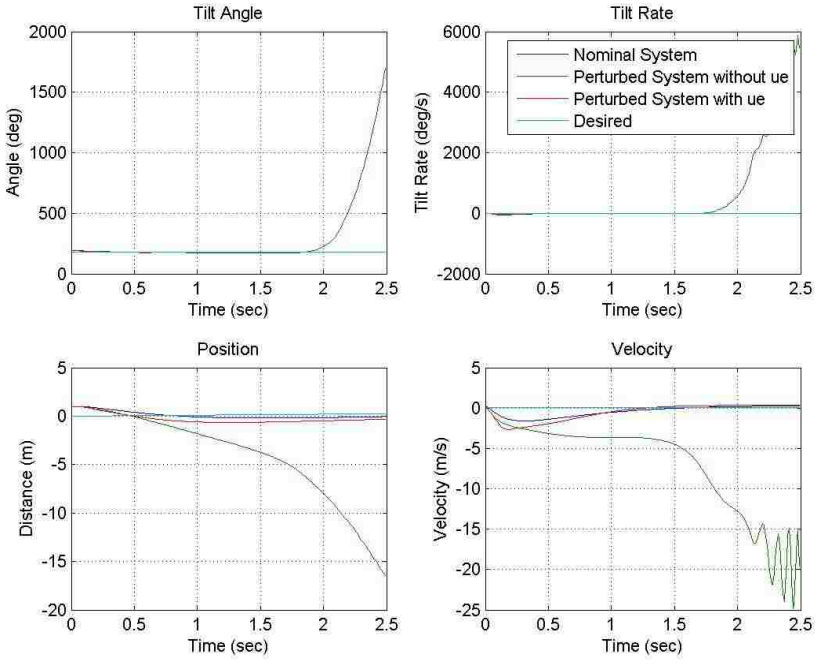


Figure 8.18. Extra Control States with Unmodeled Dynamics - Unstable

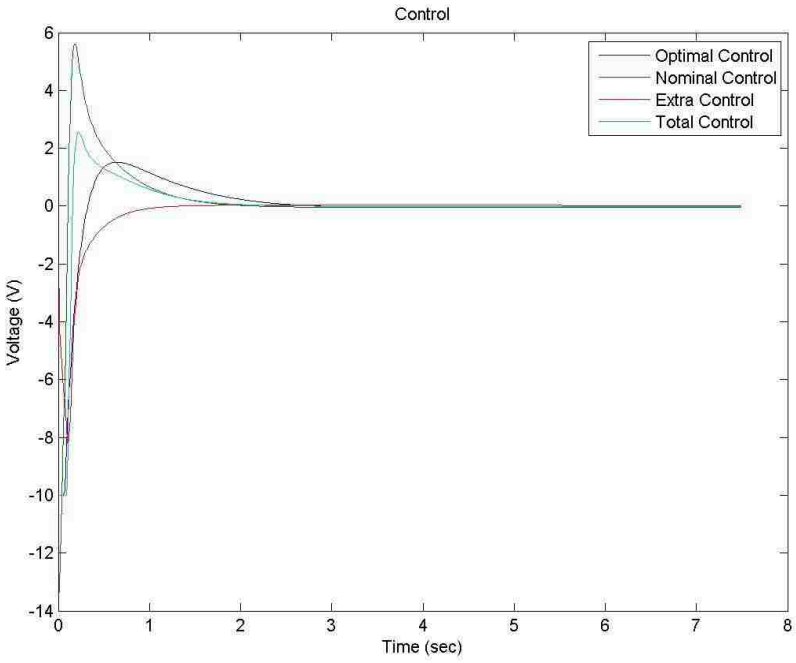


Figure 8.19. Extra Control with Unmodeled Dynamics

control drives the system closer to the desired trajectory than the optimal control for the nominal system. The control signals shown in Figure 8.21 display some interesting dynamics. The deadzone prevents any control under 0.5 from having an effect on the system, this is done by zeroing the control signal in the simulation. This causes the control graph to have the high frequency spikes shown. The control signal is above 0.5 for single time steps. At these steps the control drives the system closer to the desired trajectory, which on the next time step reduces the control signal to below 0.5. This occurs until the combination of state errors causes the LQR controller to switch directions. The end result are small stable oscillations about the desired trajectory. The extra control signal adds an additional signal to the LQR control, which assists with overcoming the controller deadzone.

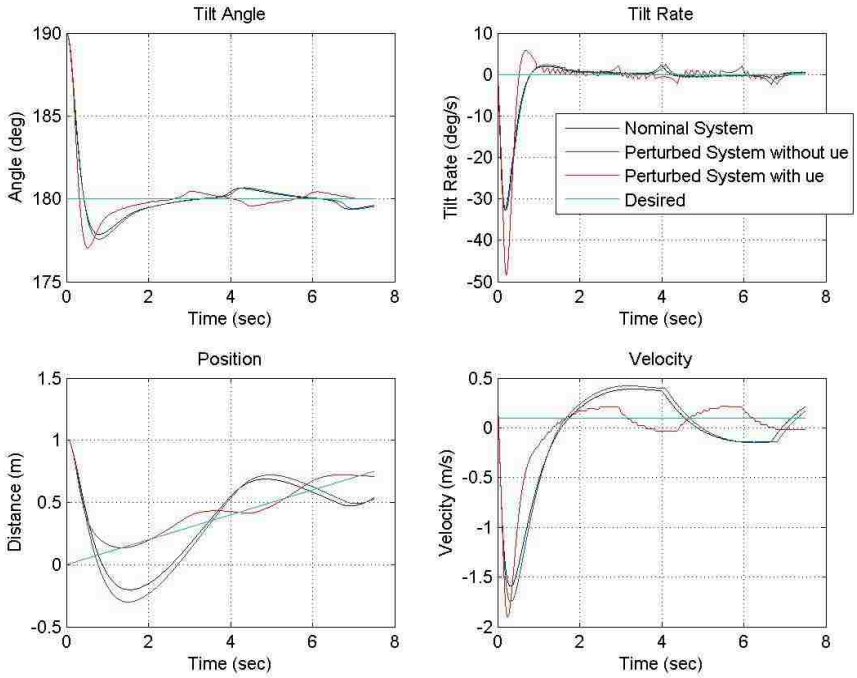


Figure 8.20. Extra Control States with Deadzone

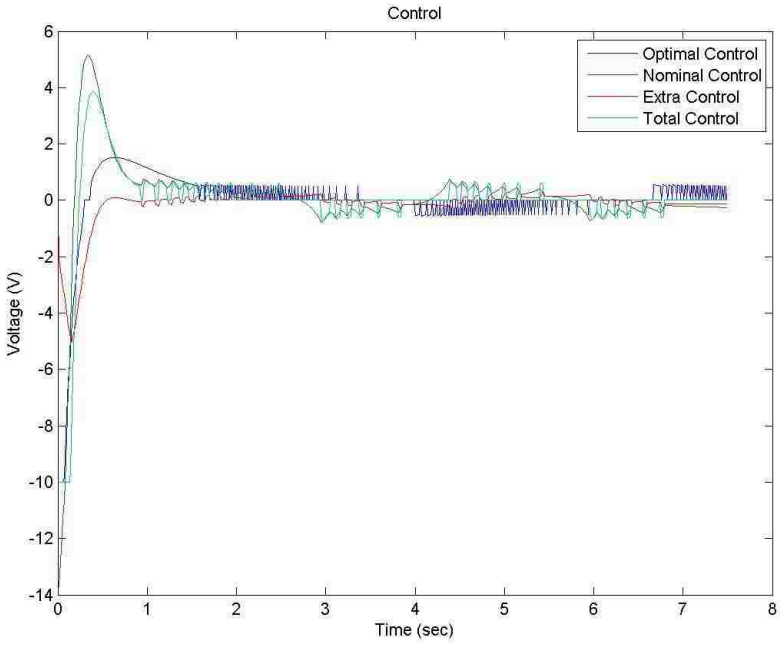


Figure 8.21. Extra Control with Deadzone

**8.2.4. Deadzone and Backlash Nonlinearities.** As in the previous study, this final simulation study includes control deadzone, but additionally adds output backlash to the tilt angle as described in Section 7.5.3. The states are shown in Figure 8.22. Once again, the optimal control for the linear system is ineffective in driving the system with nonlinearities to the desired trajectory. With compounding nonlinearities the control is even less effective than in the previous study. However, it is shown that the neural network based extra control is helpful in driving the system states closer to the desired trajectory. What is important to note is that the extra control formulation is not intended to compensate for these nonlinearities, they are not smooth continuous functions as is the case for the unmodeled dynamics and parameter uncertainty. Even so, the extra control signal is helpful in the presence of these input and output nonlinearities. The control signals are shown in Figure 8.23.

As a better comparison of the control under these various uncertainties a table is created. The controller effectiveness is quantified by summing the norms of the difference of the four system states and the desired trajectory. Lower scores indicate more effective controllers, as the system states are driven closer to the desired values. Table 8.3 shows these control scores. It is shown that in all cases the extra control assists with the controller effectiveness, resulting in the most effective controller, performing on average 53.1% better than LQR control, and even 37.3% better than the optimal LQR controller for the linear, or nominal, system.

Table 8.3. Extra Control Effectiveness

| | **Perturbed System LQR Control** | **Perturbed System LQR + Extra Control** | **Nominal System Optimal Control** |
|---|---|---|---|
| **Parameter Uncertainty** | 248.92 | 123.07 | 240.72 |
| **Unmodeled Dynamics** | 587.04 | 235.35 | 240.72 |
| **Deadzone** | 358.42 | 181.53 | 326.10 |
| **Deadzone and Backlash** | 1451.9 | 688.17 | 1434.5 |

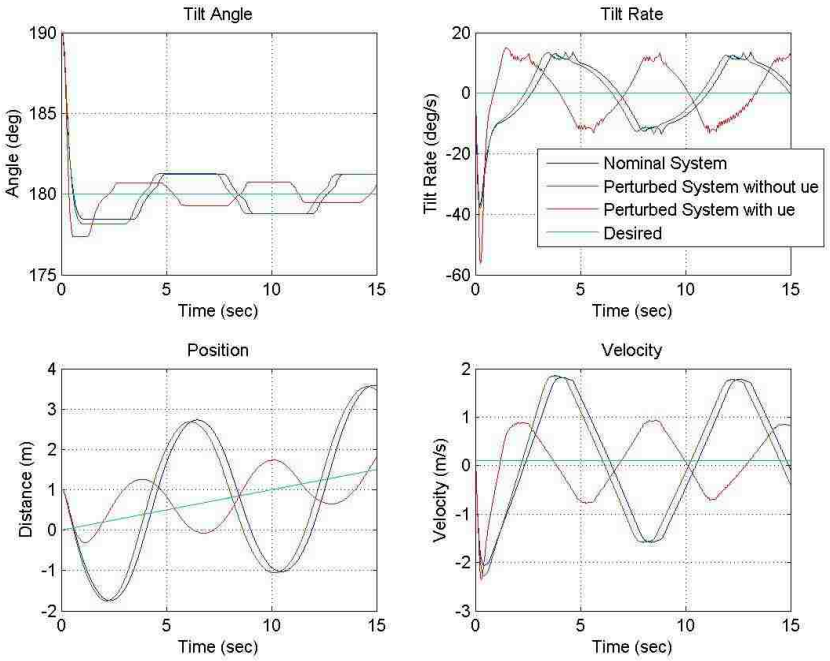


Figure 8.22. Extra Control States with Deadzone and Backlash

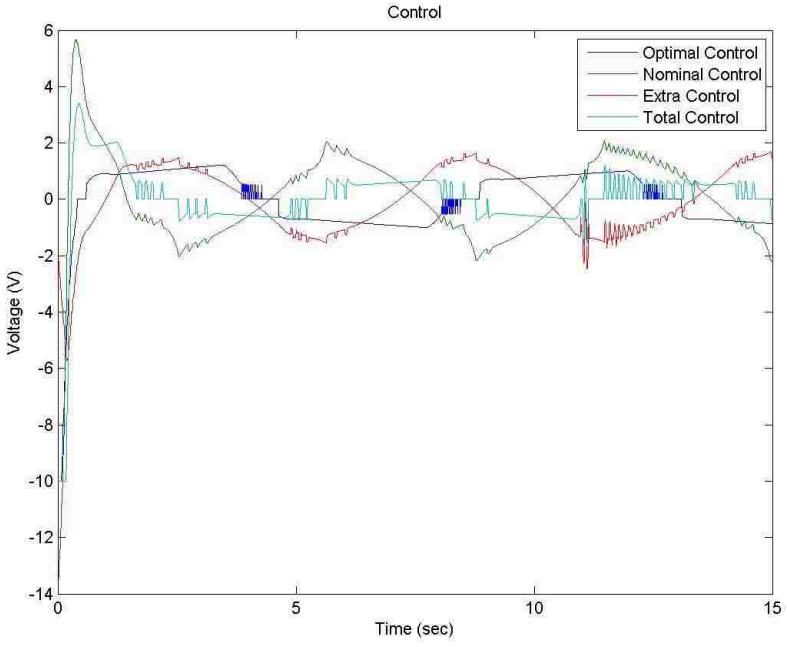


Figure 8.23. Extra Control with Deadzone and Backlash

## 8.3. IMPLEMENTATION RESULTS

The DMSO implemented on the robot used the following parameters.

$$\Gamma = 0.05 \qquad K = .5I_{4\times 4}$$

The DMSO is compared to the discrete Kalman filter, just as before in the simulations. The Kalman filter parameters are kept the same as before. The results are shown in Figure 8.24 along with the raw measurements. The tilt angle measurement is the output of the complementary filter. As before the Kalman filter and the DMSO both give very similar estimates. Figure 8.25 shows the states again zoomed in to see the differences between the Kalman filter and the DMSO. There are slight differences in the tilt angle between the two filters, but both closely match the output of the complementary filter measurement. The tilt rate is nearly identical between the two filters and the gyroscope measurements. In the position there is a greater difference, the Kalman filter has high frequency oscillations around the measurements, while the DMSO closely follows the measurements from the motor encoders. The velocity estimates are actually the opposite, the Kalman filter more closely matches the measurements from the encoder, while at points the DMSO has some larger oscillations than the Kalman filter.

Figure 8.26 shows the reason for using the complementary filter. The raw accelerometer tilt angle measurements are displayed along with the three filters estimates. In this case the Kalman filter and the DMSO are using the raw accelerometer readings for the estimate, instead of using the output of the complementary filter. As is shown the accelerometer tilt angle measurement is extremely inaccurate, resulting in measurements over 1000% from the actual value, and actually causes both the Kalman filter and the DMSO to provide very noisy, inaccurate measurements of the tilt angle. Interestingly the complementary filter performs better than both filters that have knowledge of the system dynamics. This is due to the complementary filters sensor fusion of the raw accelerometer angle measurement, along with the integrated gyroscope measurements to obtain a more accurate estimate of the tilt angle.

Lastly, the uncertainty estimates are shown in Figure 8.27. The uncertainty estimate oscillates just like it did in the simulation studies before, and they are of similar

magnitude. It is noted that before the uncertainty estimate was trending towards zero over time, where in the physical system the oscillations are statically stable. This is easily attributed to the difference in control between the simulations and the implemented system. In the simulations the controller is working perfectly to reduce the regulation error to zero over time, where in the physical system small oscillations around 1 degree from desired are obtained as the stable point. The motors used in this implementation, while better than the original, still have a deadzone. There is also a small amount of backlash in the gears, identified to be less than 1 degree. These facts in combination with the sensor noise, result in the stable oscillations shown. One may also note that the controller is also not working well in that the regulation error is not being driven to zero. Looking at Figure 8.24 at the position estimate, the controller stabilized the system around -0.05m. This is attributed to a small bias found in the tilt angle measurement, which is compensated by the negative position error in the LQR controller. Further work can definitely be done on the controller in the system, but it is not necessary to see the point of the DMSO. The state estimates of the DMSO are as accurate as the Kalman filter, with the added benefit of seeing the calculated uncertainty in the system.

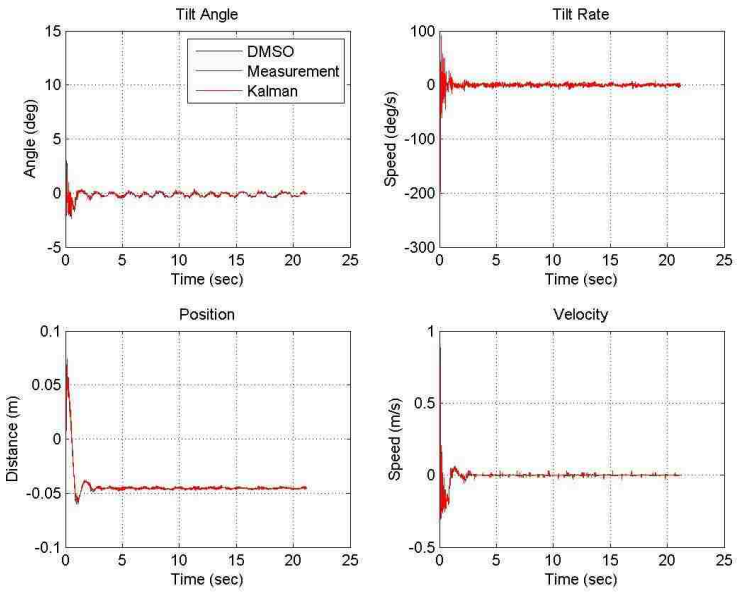


Figure 8.24. Implementation Test State Variables

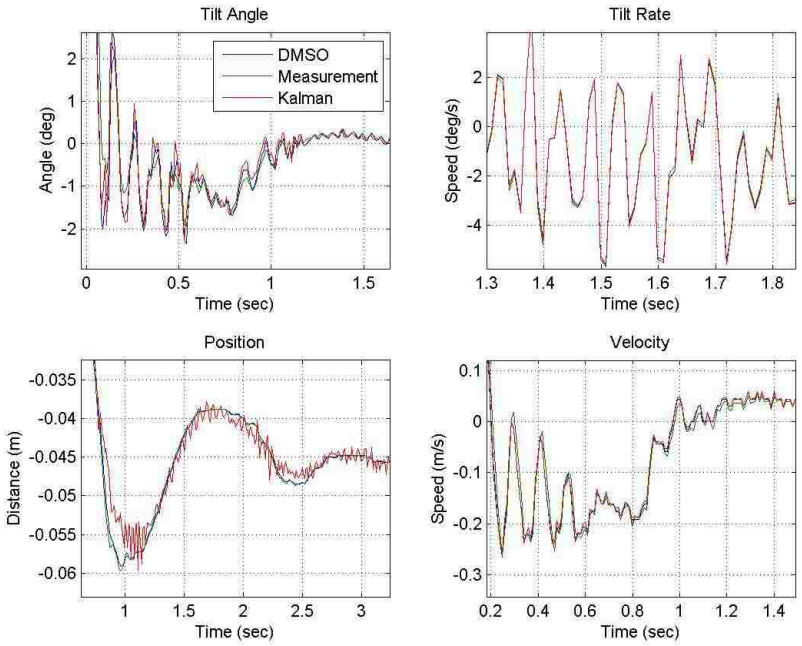


Figure 8.25. Implementation Test State Variables Close Up

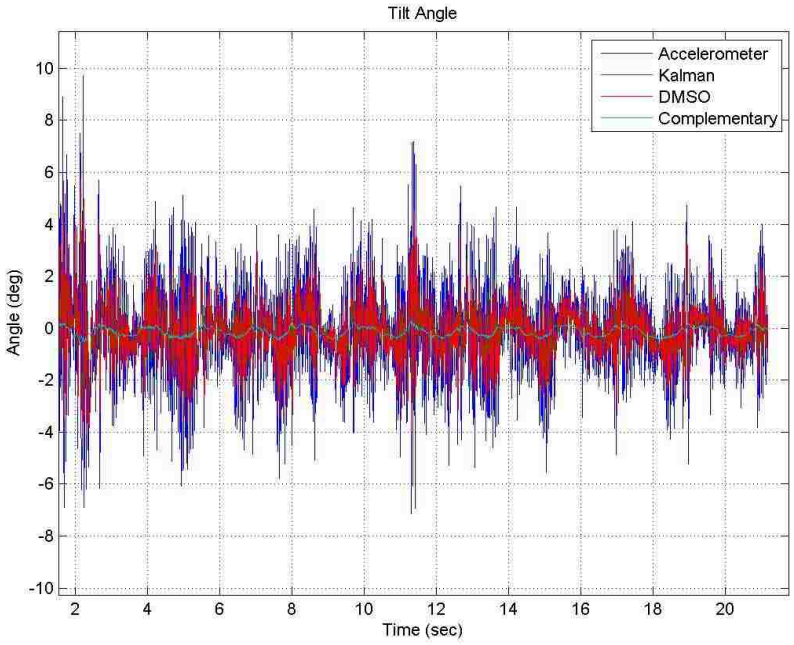


Figure 8.26. Complementary Filter Comparison

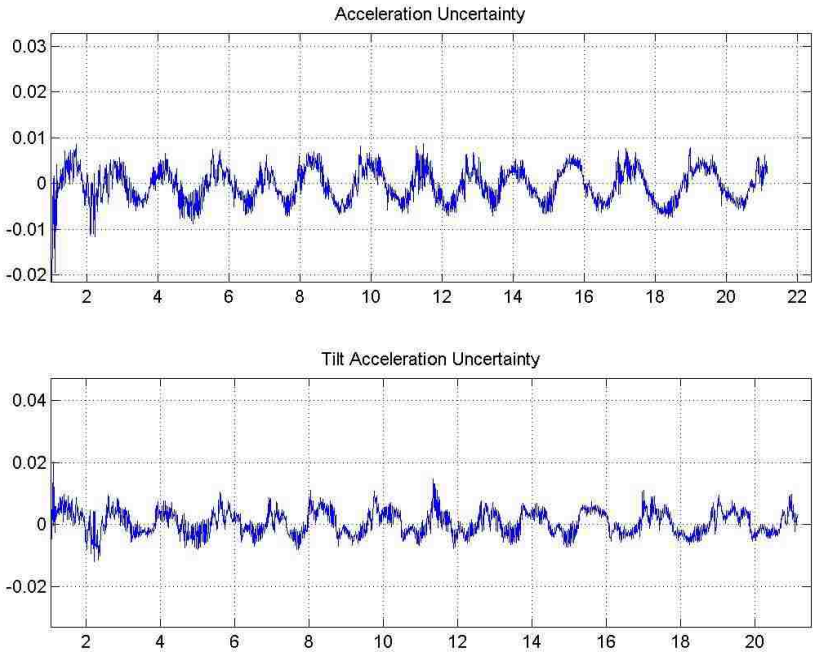


Figure 8.27. Implementation Test Results Estimated Uncertainties

# 9. CONCLUSIONS

This thesis presented the Discrete-time Modified State Observer (DMSO), a discrete-time neural network based observer for state and uncertainty estimation. The DMSO was derived for a class of general systems, and is widely applicable to many different problems in the aerospace community. In this thesis the DMSO was applied to a two wheeled inverted pendulum (TWIP) robot and shown using simulation studies to be effective in estimating the states of the system in the presence of parameter uncertainty, linearization and discretization errors, and measurement noise. Additionally the DMSO was implemented on a digital system and used to accurately estimate the system states in a physical TWIP robot. These tests using a simple system show the power of the technique, and allow for its use in more complex aerospace applications.

A neural network based controller for a class of systems with unmatched uncertainties was proposed as well. Lyapunov stability analysis was used to ensure the stability and boundedness of the state tracking errors and the neural network estimation errors. Simulation studies were used to show the effectiveness of the neural network controller in the presence of parameter uncertainty, unmodeled dynamics, and actuator nonlinearities. Once again, these studies on a simple system show the effectiveness of the technique, and allow for further study on a number of more difficult problems.

Future work should include implementing the neural network based controller in a physical system with unmatched uncertainties to test the effectiveness in a real system. The TWIP used in this thesis would be a good place to start. These simulations and tests with a TWIP show the effectiveness of both of the proposed techniques in a simple problem. Further work should include testing and implementation on a number of additional problems to show the robustness of the techniques.

# BIBLIOGRAPHY


1. Khalil, H.K., *Nonlinear Systems*. 2002, Upper Saddle River, NJ: Prentice Hall. 750.

2. Slotine, J.-J.E. and W. Li, *Applied Nonlinear Control*. 1991, Upper Saddle River, New Jersey: Pearson Education. 461.

3. Neugebauer, R., et al., *Mobile Systems for Machining Large Work Pieces*, in *Enabling Manufacturing Competitiveness and Economic Sustainability*, H.A. ElMaraghy, Editor. 2012, Springer Berlin Heidelberg. p. 135-140.

4. Ha, Y. and S.i. Yuta, *Trajectory tracking control for navigation of self-contained mobile inverse pendulum*, in *IEEE/RSJ/GI International Conference on Intelligent Robots and Systems*. 1994, IEEE: Munich. p. 1875-1882.

5. Shiroma, N., et al., *Cooperative behavior of a wheeled inverted pendulum for object transportation*, in *IEEE/RSJ International Conference on Intelligent Robots and Systems*. 1996, IEEE: Osaka. p. 396-401.

6. Ozaki, H., et al., *Position and Orientation Control of a Wheeled Inverted Pendulum.* JSME International Journal Series C Mechanical Systems, Machine Elements and Manufacturing, 2001. **44**(1): p. 188-195.

7. Grasser, F., et al., *JOE: A Mobile, Inverted Pendulum.* IEEE Transactions on Industrial Electronics, 2002. **49**(1): p. 8.

8. Ooi, R.C., *Balancing a Two-Wheeled Autonomous Robot*, in *School of Mechanical Engineering*. 2003, The University of Western Australia: Crawley, Western Australia. p. 72.

9. Solis, J. and A. Takanishi, *Development of the Waseda Wheeled robot No. 2 Refined II and Pilot Experiments with Undergraduate Students*, in *13th World Congress in Mechanism and Machine Science*. 2011: Guanajuanto, Mexico.

10. Sun, L. and J. Gan, *Researching Of Two-Wheeled Self-Balancing Robot Base On LQR Combined With PID*, in *2nd International Workshop on Intelligent Systems and Applications*. 2010, IEEE: Wujan.

11. Guerrero, G.C. and J.J.R. Pasaye, *Real-time Control to Hold Upright Equilibrium of a Two-Wheeled Auto-balancing Vehicle*, in *IEEE International Autumn Meeting on Power, Electronics and Computing*. 2013, IEEE: Mexico city.

12. Miranda, J.L.C., *Application of Kalman Filtering and PID Control for direct Inverted Pendulum control*, in *Electrical and Computer Engineering*. 2009, California State University, Chico.

13. Kim, S. and S. Kwon. *Nonlinear Control Design for a Two-wheeled Balancing Robot*. in *10th International Conference on Ubiquitous Robots and Ambient Intelligence*. 2013. Jeju, Korea.

14. Kim, Y., S.H. Kim, and Y.K. Kwak, *Dynamic Analysis of a Nonholonomic Two-Wheeled Inverted Pendulum Robot.* Journal of Intelligent and Robotic Systems, 2005. **44**: p. 22.

15. Pathak, K., J. Franch, and S.K. Agrawal, *Velocity and Position Control of a Wheeled Inverted Pendulum by Partial Feedback Linearization.* IEEE Transactions on Robotics, 2005. **21**(3): p. 505-513.

16. Askari, M., et al., *Model Predictive Control of an Inverted Pendulum*, in *International Conference for Technical Postgraduates*. 2009: Kuala Lumpur, Malaysia.

17. Shibayama, K., V. Kroumov, and A. Inoue, *Robust Control of Underactuated Inverted Pendulum System in Presence of Unkown Disturbances*, in *International Conference on Modelling, Identification and Control*. 2010: Okayama, Japan.

18. Ja, J.-S. and J.-J. Lee, *Position Control of Mobile Two Wheeled Inverted Pendulum Robot by Sliding Mode Control*, in *International Conference on Control, Automation and Systems*. 2012: Jeju Island, Korea.

19. Tsai, C.-C. and S.-Y. Ju, *Trajectory Tracking and Regulation of a Self-Balancing Two-Wheeled robot: A Backstepping Sliding-Mode Control Approach*, in *SICE Annual Conference*. 2010, SICE: Taipei, Taiwan.

20. Do, K.D. and G. Seet, *Motion Control of a Two-Wheeled Mobile Vehicle with an Inverted Pendulum.* Journal of Intelligent and Robotic Systems, 2010. **60**(3-4): p. 577-605.

21. Rudra, S. and R.K. Barai, *Robust Adaptive Backstepping Control of Inverted Pendulum on Cart System.* International Journal of Control and Automation, 2012. **5**(1): p. 13-26.

22. Durdevic, P. and Z. Yang, *Hybrid Control of a Two-Wheeled Automatic-Balancing Robot with Backlash Feature*, in *IEEE International Symposium on Safety, Security, and Rescue Robotics*. 2013, IEEE: Linkoping.

23. Yue, M., X. Wei, and Z. Li, *Adaptive Sliding-Mode Control for Two-Wheeled Inverted Pendulum Vehicle Based on Zero-Dynamics Theory.* Nonlinear Dynamics, 2014. **76**: p. 459-471.

24. Noh, J.S., G.H. Lee, and S. Jung, *Position control of a mobile inverted pendulum system using radial basis function network*, in *IEEE International Joint Conference on Neural Networks*. 2008, IEEE: Hong Kong. p. 370-376.

25. Jung, S. and S.S. Kim, *Control Experiment of a Wheel-Driven Mobile Inverted Pendulum Using Neural Network.* IEEE Transactions on Control Systems Technology, 2008. **16**(2): p. 297-303.

26. Tsai, C.-C., H.-C. Huang, and S.-C. Lin, *Adaptive Neural Network control of a Self-Balancing Two-Wheeled Scooter.* IEEE Transactions on Industrial Electronics, 2010. **57**(4): p. 1420-1428.

27. Li, Z. and C. Yang, *Neural-Adaptive Output Feedback Control of a class of Tranportation Vehicles Based on Wheeled Inverted Pendulum Models.* IEEE Transactions on Control Systems Technology, 2012. **20**(6): p. 1583-91.

28. Li, Z., C. Yang, and L. Fan, *Advanced Control of Wheeled Inverted Pendulum Systems*. 2013, London: Springer.

29. Gomez, M., T. Arribas, and S. Sanchez, *Optimal Control Based on CACM-RL in a Two-Wheeled Inverted Pendulum.* International Journal of Advanced Robotic Systems, 2012. **8**(235).

30. Sadegh, N., *A Perceptron Network for Functional Identification and Control of Nonlinear Systems.* IEEE Transactions on Neural Networks, 1993. **4**(6): p. 7.

31. Yang, Y., et al., *Electrohydraulic Control Using Neural MRAC Based on a Modified State Observer.* IEEE/ASME Transactions on Mechatronics, 2013. **18**(3): p. 11.

32. Sarangapani, J., *Neural Network Control of Nonlinear Discrete-Time Systems*. Control Engineering, ed. F.L. Lewis. 2006, Boca Raton, FL: CRC Press Taylor & Francis Group. 602.

33. Rajagopal, K., et al., *Neuroadaptive Model Following Controller Design for Non-affine Non-square Aircraft System*, in *AIAA Guidance, Navigation, and Control Conference*. 2009, AIAA: Chicago, Illinois. p. 21.

34. Harl, N., K. Rajagopal, and S.N. Balakrishnan, *Modified State Observer for Orbit Uncertainty Estimation*, in *AIAA Guidance, Navigation, and Control Conference*. 2011: Portland, Oregon. p. 25.

35. Jacob Darling, J.S., S.N. Balakrishnan, *Modified State Observer for Atomospheric Reentry Uncertainty Estimation*, in *AIAA Guidance, Navigation, and Control Conference*. 2012, AIAA: Minneapolis, Minnesota. p. 16.

36. Darling, J.E., S.N. Balakrishnan, and C. D'Souza, *Sigma Point Modified State Observer for Nonlinear Uncertainty Estimation*, in *AIAA Guidance, Navigation, and Control Conference*. 2013: Boston, MA. p. 10.

37. Pappu, V.S.R., et al., *Modified State Observer Based Adaptive Control Law Design for the Black Kite MAV*, in *American Control Conference*. 2014: Portland, Oregon.

38. Salgado, I. and I. Chairez, *Nonlinear Discrete Time Neural Network Observer.* Neurocomputing, 2013. **101**: p. 9.

39. Alma Y. Alanis, E.N.S., Luis J. Ricalde, *Discrete-Time Reduced Order Neural Observers for Uncertain Nonlinear Systems.* International Journal of Neural Systems, 2010. **20**(1): p. 29-38.

40. Lewis, F.L., J. Sarangapani, and A. Yesildirek, *Neural Network Control of Robot Manipulators and Nonlinear Systems*. 1999, Padstow, UK: Taylor & Francis.

41. Abadir, K.M. and J.R. Magnus, *Matrix Algebra*. 2005, Cambridge University Press.

42. Young, W.H., *On Classes of Summable Functions and their Fourier Series*. Vol. 87. 1912. 225-229.

43. Kress, R., *Numerical Analysis*. 1998: Springer New York.

44. Ioannou, P. and P. Kokotovic, *Adaptive Systems with Reduced Models*. 1983, Springer-Verlag, New York.

45. Kuipers, J.B., *Quaternions and Rotation Sequences*. 1999, Princeton, New Jersey: Princeton University Press.

46. Walter T. Higgins, J., *A Comparison of Complementary and Kalman Filtering.* IEEE Transactions on Aerospace and Electronic Systems, 1975. **11**(3): p. 5.

47. Silva, C.W.d., *Mechatronics: An Integrated Approach*. 1st ed. 2004: CRC Press. 1344.

48. Hibbeler, R.C., *Engineering Mechanics: Dynamics*. 12th Edition ed. 2010, Upper Saddle River, New Jersey: Pearson Prentice Hall.

49. Huang, Z., *Lyapunov Function Based Neurocontrollers for a Class of Deterministic and Stochastic Problems*, in *Mechanical Engineering*. 2002, University of Missouri-Rolla: Rolla, MO.

50. Simon, D., *Optimal State Estimation*. 2006, Hoboken, New Jersey: John Wiley & Sons. 526.

51. Gebre-Egziabher, D., *Design and Performance Analysis of a Low-Cost Aided Dead Reckoning Navigator*, in *Department of Aeronautics and Astronautics*. 2004, Stanford University. p. 285.

52. Warren S. Flenniken, I., *Modeling Inertial Measurement Units and Analyzing the Effect of their Errors in Navigation Applications*, in *Mechanical Engineering*. 2005, Auburn University. p. 165.

53. Kalman, R.E., *A New Approach to Linear Filtering and Prediction Problems.* Transactions of the ASME - Journal of Basic Engineering, 1960. **82**: p. 35-45.

# VITA

Jason Michael Stumfoll was born in Springfield, Missouri on October 23, 1990 to Leo and Lisa Stumfoll. He graduated from Parkview High School in May of 2008. In 2007 he was selected to participate in the Missouri Scholars Academy at the University of Missouri. He graduated with his Bachelor of Science degree in Aerospace Engineering in May of 2012 from the Missouri University of Science and Technology. He spent time working with the Advanced Aero Vehicle Group and the Missouri Satellite Team, holding leadership positions in both. He worked for IST-Rolla as an engineer during the year of 2013 had an internship with NASA Marshall Space Flight Center in Spring of 2014. In August 2015 he received a Master's of Science degree in Aerospace Engineering from the Missouri University of Science and Technology.