Underslung Payload Tension Control from an Autonomous Unmanned Helicopter

Brian J. McCabe

Thesis submitted to the Faculty of the Virginia Polytechnic Institute and State University in partial fulfillment of the requirements for the degree of

Master of Science in

Mechanical Engineering

Kevin B. Kochersberger, Chair

Alfred L. Wicks

Steve C. Southward

May 2, 2012

Blacksburg, Virginia

Keywords: Tension Control, Tethered Payloads, Unmanned Systems

Copyright 2012, Brian J. McCabe
A tension control algorithm for the deployment of a unmanned ground vehicle from an autonomous helicopter is designed and tested in this thesis. The physical hardware which the controller will run on is detailed. The plant model and underlying controllers are derived and modeled. The tension controller algorithm is selected, derived, and modeled. The parameters of the tension controller are chosen and simulations are run with the chosen parameters. The tension control algorithm is run on the physical hardware, successfully demonstrating tension control on a ground vehicle. Robustness simulations are run for a change in the radius of the spool and the length of the tether. Lastly, Future work is outlined on several paths to move forward with the tension controller.
Contents

Abstract ii

List of Figures vi

List of Tables x

1 Introduction 1

1.1 Motivation of work 2

1.2 Thesis Organization 4

2 Literature Review 7

2.1 Unmanned Systems Lab Literature 8

2.1.1 Indirectly Related Work 8

2.1.2 Directly Related Work 10

2.1.3 Key Differences from Directly Related Work 18

2.2 Other Literature 18

2.2.1 Tethered Unmanned Operations 19

2.2.2 Underslung Payload Sway Control 21

3 Winch Pod Hardware and Software 24
<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.1</td>
<td>Structure hardware</td>
<td>27</td>
</tr>
<tr>
<td>3.2</td>
<td>Winch hardware</td>
<td>29</td>
</tr>
<tr>
<td>3.3</td>
<td>Electronics hardware</td>
<td>34</td>
</tr>
<tr>
<td>3.4</td>
<td>Software</td>
<td>36</td>
</tr>
<tr>
<td>4</td>
<td>Closed Loop Speed Control System Simulation</td>
<td>40</td>
</tr>
<tr>
<td>4.1</td>
<td>Plant Model</td>
<td>44</td>
</tr>
<tr>
<td>4.2</td>
<td>ELMO Current Controller</td>
<td>58</td>
</tr>
<tr>
<td>4.3</td>
<td>ELMO Velocity Controller</td>
<td>63</td>
</tr>
<tr>
<td>4.4</td>
<td>Test Results</td>
<td>69</td>
</tr>
<tr>
<td>4.5</td>
<td>Speed controller bandwidth</td>
<td>74</td>
</tr>
<tr>
<td>5</td>
<td>Tension Control Model</td>
<td>77</td>
</tr>
<tr>
<td>5.1</td>
<td>Time-Step Selection</td>
<td>80</td>
</tr>
<tr>
<td>5.2</td>
<td>Continuous-Time PID Model</td>
<td>87</td>
</tr>
<tr>
<td>5.3</td>
<td>Finite Difference Approximation</td>
<td>90</td>
</tr>
<tr>
<td>5.4</td>
<td>Tustin’s Method Approximation</td>
<td>94</td>
</tr>
<tr>
<td>5.5</td>
<td>Comparison of Methods</td>
<td>97</td>
</tr>
<tr>
<td>5.6</td>
<td>Implemented PID Controller</td>
<td>101</td>
</tr>
<tr>
<td>5.7</td>
<td>PID Gain Selection</td>
<td>104</td>
</tr>
<tr>
<td>5.8</td>
<td>Tension Controller Validation</td>
<td>110</td>
</tr>
<tr>
<td>6</td>
<td>Validation of the Design</td>
<td>112</td>
</tr>
<tr>
<td>6.1</td>
<td>Simulation Test Results</td>
<td>113</td>
</tr>
<tr>
<td>6.2</td>
<td>Physical Test Results</td>
<td>121</td>
</tr>
</tbody>
</table>
# List of Figures

1.1 The UGV being deployed from an RMAX autonomous helicopter ............... 3

3.1 The USL’s Yamaha RMAX helicopter UAV ................................. 25
3.2 The USL’s custom built UGV outfitted with a scoop ......................... 25
3.3 Winch pod hardware .................................................................. 26
3.4 Winch pod structure ................................................................... 28
3.5 Winch pod motor and gearhead ................................................... 29
3.6 Aluminum spool .......................................................................... 31
3.7 Experimental stiffness data collected at 9 meters ............................. 32
3.8 ELMO Solo-Whistle motor controller ........................................... 33
3.9 KEIL MCB2300 evaluation board ................................................ 34
3.10 Custom designed power and sensor distribution board ................. 35
3.11 Ground control station software front panel .................................. 37

4.1 Speed control block diagram ...................................................... 41
4.2 Schematic of the speed control system plant ................................ 45
4.3 Spool, tether, and UGV configuration ......................................... 51
4.4 Simulink model for the speed control system plant ....................... 53
5.10 Finite difference approximation of PID controller for time step of 0.02 seconds
5.11 Tustin’s approximation of PID controller for time step of 0.02 seconds
5.12 Simulink model of implemented tension controller
5.13 LabVIEW Embedded implementation of PID controller
5.14 Metrics for step input of the system with a tether length of 15.8 meters
5.15 Metrics for step input of the system with a tether length of 8.2 meters
5.16 Metrics for step input of the system with a tether length of 4.9 meters
5.17 Step inputs of the tension controller using the gains in Table 5.3
5.18 Simulated versus experimental data of the tension controller for varying tether lengths
6.1 Step input simulations for varying tether lengths
6.2 Box plots of 100 samples of UGV lift off time for varying $\theta$, $\alpha$, and tether lengths
6.3 Simulated tension and actual UAV velocities for simulated mission
6.4 Tension controller simulation for actual UAV velocities
6.5 Step inputs for spring tests simulating various tether lengths
6.6 Tension test with 9.6 m tether length at 0.3 m/s and $\theta < 8.5^\circ$
6.7 Tension test with 5.8 m tether length at 0.3 m/s and $\theta < 13.5^\circ$
6.8 Tension test with 5.8 m tether length at 0.15 m/s and $\theta < 3.5^\circ$
6.9 Tension controller simulations of various spool radii at 5 meters of tether
6.10 Metrics for simulations with varying radii at 5 meters of tether
6.11 Tension controller simulations of various spool radii at 15 meters of tether
6.12 Metrics for simulations with varying radii at 15 meters of tether
6.13 Tension controller simulations of various spool radii at 25 meters of tether
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>6.14 Metrics for simulations with varying radii at 25 meters of tether</td>
<td>134</td>
</tr>
<tr>
<td>6.15 Tension controller simulations of varying tether lengths assuming a tether length of 5 m</td>
<td>136</td>
</tr>
<tr>
<td>6.16 Metrics for simulations with varying tether lengths assuming a tether length of 5 m</td>
<td>137</td>
</tr>
<tr>
<td>6.17 Tension controller simulations of varying tether lengths assuming a tether length of 15 m</td>
<td>138</td>
</tr>
<tr>
<td>6.18 Metrics for simulations with varying tether lengths assuming a tether length of 15 m</td>
<td>139</td>
</tr>
<tr>
<td>6.19 Tension control simulations of varying tether lengths assuming a tether length of 25 m</td>
<td>140</td>
</tr>
<tr>
<td>6.20 Metrics for simulations with varying tether lengths assuming a tether length of 25 m</td>
<td>141</td>
</tr>
</tbody>
</table>
# List of Tables

<table>
<thead>
<tr>
<th>Table</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1</td>
<td>Metrics for models developed in May's Thesis</td>
<td>14</td>
</tr>
<tr>
<td>3.1</td>
<td>Tether stiffness test data</td>
<td>31</td>
</tr>
<tr>
<td>4.1</td>
<td>Metrics for closed loop speed control system developed in May's Thesis</td>
<td>42</td>
</tr>
<tr>
<td>4.2</td>
<td>Comparison of metrics for the closed loop speed control system simulation</td>
<td>73</td>
</tr>
<tr>
<td>5.1</td>
<td>Robot Performance Data from Rose's Thesis</td>
<td>81</td>
</tr>
<tr>
<td>5.2</td>
<td>Line velocity based on helicopter and UGV velocity data</td>
<td>83</td>
</tr>
<tr>
<td>5.3</td>
<td>PI controller gains based on tether length</td>
<td>104</td>
</tr>
<tr>
<td>A.1</td>
<td>Gain selector values</td>
<td>149</td>
</tr>
<tr>
<td>A.1</td>
<td>Gain selector values</td>
<td>150</td>
</tr>
<tr>
<td>A.1</td>
<td>Gain selector values</td>
<td>151</td>
</tr>
<tr>
<td>A.1</td>
<td>Gain selector values</td>
<td>152</td>
</tr>
<tr>
<td>A.2</td>
<td>Values used for the constants of the speed control model</td>
<td>153</td>
</tr>
</tbody>
</table>
Chapter 1

Introduction

This thesis presents the work done in the design of a gain scheduled tension controller for a UGV tethered to an autonomous helicopter. Along with the design of the tension controller itself, the physical components of a winch system are modeled to allow for meaningful simulations of the tension controller to be conducted. The tension controller is designed as a proportional-integral-derivative controller and implemented as a proportional-integral controller. It is shown to work successfully from a UAV at heights greater than 15 meters with some limitations or 25 meters with no limitations, and from a stationary platform at heights greater than 5 meters.
1.1 Motivation of work

There are many scenarios where an unmanned ground vehicle (UGV) needs to be operated from a distance to ensure the safety of the operators while maintaining the ability to complete its given assignment. Examples include the defusing of improvised explosive devices, the mapping of active volcanoes, and the investigation of radioactive environments. A small UGV is also the perfect platform for activities in constrained environments. The ability of a small UGV to move between larger objects or under overhangs gives it the advantage over larger UGVs in the same environment. However, a smaller UGV tends to have limitations on the range and speed in which it can travel. It is sometimes necessary, especially with small UGVs, to deploy the UGV from so far out that it is impractical, or impossible, to guide it to the location of interest in a reasonable amount of time. In these cases the use of a tele-operated UGV tethered to an unmanned helicopter allows the UGV to be lowered close to the location of interest while ensuring the safety of the operator. The Unmanned Systems Lab (USL) at Virginia Tech is currently working on this particular set of problems. Figure 1.1 shows a test flight of a tethered UGV from an autonomous helicopter in hover.

In the deployment of a UGV from an autonomous helicopter the problem that must be considered is how to keep the UGV on the ground while still being tethered to the helicopter. In this thesis, this problem is accomplished through the use of a tension controller which maintains the tension on the tether, never allowing too much line, or too little, to be let out. If too much line has been spooled out, the tether could catch on ground objects or on the
Figure 1.1: The UGV being deployed from an RMAX autonomous helicopter.
treads of the UGV itself. If too little line is spooled out, the UGV could be pulled off the ground by the movement of the autonomous helicopter. Either case is undesirable, possibly resulting in the premature ending of the mission or the loss of the UGV.

1.2 Thesis Organization

Chapter 2 consists of the literature review done for this thesis. The first section details the work done in the Unmanned System Lab on the deployment of a UGV from an autonomous helicopter. The second section goes over the current state of the art for tethered deployment of UGVs from other vehicles and discusses the state of the art in the realm of sway control from autonomous unmanned helicopters.

Chapter 3 consists of the discussion of the winch pod hardware and software. This chapter goes over the structural and electro-mechanical hardware, both of which were not specifically designed for this thesis. It also details the electronics hardware and software which were designed for this thesis.

Chapter 4 consists of the modeling and verification of the speed control model. The first section in the chapter begins by deriving the equations of motion for the plant of the system: the motor, gearhead, spool, and tether. This section also describes in detail the Simulink model developed from the equations of motion. The second and third sections in this chapter deal with the digital servo driver. The second section models the current controller, while the
third section models the velocity controller, both of these models are in Simulink. The fourth section presents the test results, comparing the simulation against actual data collected from the winch. The fifth section documents the bandwidth of the speed controller.

Chapter 5 consists of the modeling and implementation of the tension controller. The first section details the selection of the time-step for the discrete-time controller. The next three sections describe the modeling of three different, proportional-integral-derivative (PID) controller implementations: a continuous-time PID controller, a finite difference approximation of the time-domain PID controller, and a Tustin’s approximation of the continuous-time PID controller. The fifth section compares the three models, and describes why the finite difference approximation was selected for the tension controller. The sixth section begins by describing the Simulink model of the final implementation of the PID controller. This section then presents the LabVIEW Embedded program which runs the PID controller on the MCB2300 board. The seventh section outlines the development of the PID gain equations. These equations allow the controller to be gain scheduled based on the length of the tether. The eighth, and final section, displays test results validating the tension control model.

Chapter 6 consists of the simulation, physical, and robustness tests. The first section describes the simulation tests run. These tests show that the gain equations work over a large range of tether lengths and the tension controller works for simulated and real helicopter and UGV velocities. The second section shows the physical test results. The first set of physical test results show the output of the tension controller to a step input with springs
simulating different tether lengths. The second set of physical test results show the tension controller working from a height of 9.6 meters and 5.8 meters. The third, and final, section describes the robustness simulation tests. These tests show the behavior of the controller for unmeasured changes in the radius of the spool and the length of the tether.

Chapter 7 consists of the conclusions and future work of this thesis.
Chapter 2

Literature Review

There are many applications for tethered unmanned vehicles, tethered robotics, and sway control, not limited to a UGV tethered to a UAV. Along with the tethered deployment of a UGV from a UAV, tethered robotics include remotely operated vehicles (ROVs), as well as UGVs and UAVs tethered to a command station. The tethers are sometimes used to limit the movement of a vehicle, and often pass signals and power to and from the vehicle. The first section of this chapter deals with the current research into the tethered deployment of a UGV being conducted by the Unmanned Systems Lab. The second section is a survey of tethered robotics of all kinds in addition to automatic sway control of tethered payloads.
2.1 Unmanned Systems Lab Literature

The Unmanned Systems Lab (USL) at Virginia Tech has developed a ground sampling UGV. Krawiec et al. [1] and Rose et al. [2] describe the mission and system architecture during which the UGV is meant to be deployed from an autonomous helicopter and sample a hazardous environment. While this thesis deals with the deployment of the UGV much work has been done on the UGV itself.

2.1.1 Indirectly Related Work

Rose, in his thesis [3], outlines the design and construction of the first generation ground sampling UGV. He designs his UGV with two main factors in mind. The first is the size and weight of the UGV, so that it will fit on the Yamaha RMAX helicopter, and the second is the scenario into which the UGV will be placed. In order to fit onto the RMAX helicopter, Rose determines that the UGV must be sized less then $540 \times 510 \times 200$ mm and weigh less then 8 kg. Additionally, he determines that the UGV must be able to traverse up a $45^\circ$ degree slope and across a $35^\circ$ degree slope while maintaining a speed of 0.25 m/s to move around in the environment. To accomplish this, Rose designs a differential drive, tracked UGV made mostly of carbon fiber. The design includes a front tray area where a sample collection system will be placed in future designs. The UGV was tested thoroughly and found to meet all of the design requirements.
In Rose’s thesis he also designs a path planning algorithm based on pre-existing three dimensional terrain data. Given this data, the algorithm develops a path to the location of interest using A* search and a custom cost function based on the capabilities of the UGV. Rose develops this algorithm to be used as either a stand alone automatic waypoint navigator or as an aid for remote control of the UGV.

Rose [4] also looks into the effectiveness of different user interfaces and camera angles for the teleoperation of the UGV. In order to accomplish this, he develops a computer simulation of a mission to travel from a starting location to a location of interest and tests the effectiveness through human subject testing. This is done by first designing a course in the real world, dividing the course into a $12 \times 12$ grid, and taking picture at every grid location to show what the UGV would see at that location. Then an overhead picture was taken to provide that camera angle and a stereo image was taken to develop a 3D map of the area. Testing was done to determine the effectives of four different user interfaces and camera angles: an in-situ monovision view from the robot, an external monovision view from the air, a combined in-situ and external monovision, and a 3D map with the location of the UGV displayed. The four user interfaces were tested on several metrics including total simulation time and number of invalid moves. Rose et al. finds that the combined monovision system and the 3D map were the most effective interfaces and that the other two (single monovision systems) were significantly less effective. Kroeger et al. [5] adds to this by testing user interfaces with a small scale hardware UGV. In addition to the different vision systems, Kroeger et al. provides the user with Rose’s path planning algorithm and a color overlay of valid locations
for the UGV. The users then moved the UGV from the starting location to the location of interest and similar metrics to the ones used by Rose et al. were collected. Kroeger et al. finds that the combination of in-situ monovision, external monovision, and 3D maps as well as the ability to modify the data as necessary significantly improves the effective use of the UGV.

2.1.2 Directly Related Work

May’s thesis [6] develops several theoretical and practical technologies for the control of a tethered payload from an autonomous helicopter. His theses covers four main points: the hardware necessary to deploy the payload, a model of the underlying speed and position controllers for the winch system, a practical tension controller, and a theoretical sway controller. May proposes a five step mission which utilizes the research described in his thesis.

1. The helicopter maneuvers to a location of interest

2. A Ground Sampling Robot is winched down while the helicopter hovers

3. The winch system maintains tension on the tether while the Ground Sampling Robot performs its mission

4. The winch system raises the Ground Sampling Robot while controlling for sway

5. The helicopter returns and lands
This theoretical mission described by May allows for the autonomous sampling of an extreme environment. May points out that the cooperation between two robots eliminates many of the tradeoffs which would be necessary for this mission otherwise and therefore his thesis deals with technologies for cooperative robotics.

May dedicates a chapter to describing the physical hardware he uses as a test platform for his control systems. He details a system which is adaptable to any aircraft and capable of remote operation, given a communication link, and automatic control. May proceeds to outline the winch components, including a DC brushed motor, its attached gearhead, a brake mechanism, and the spool attached to a braided Spectra fiber tether. Based on the speed of the ground vehicle, May develops several requirements for the winch components, specifically the motor, including a final output speed of 200 rpm. Following the design requirements developed, May selects the Maxon RE40 DC brushed motor with a 23:1 gearhead attached. This arrangement has a max speed of 522 rpm and satisfies the design requirements of the system.

May continues this section by specifying the electrical components including the motor controller, microcontroller, power supply, power regulation, and tension feedback hardware. He develops the power requirements for the theoretical 45-minute mission to be 743 mAh at 22.2 V. He recommends a 7 cell, 3300 mAh, LiPo battery to maintain a voltage of greater than 25 volts for the duration of a typical 45 minute mission. May designs a custom microcontroller board built around a NXP LPC2378 microprocessor to be used as a system controller.
The LPC2378 was chosen due to its ability to run LabVIEW Embedded, number of usable UARTS, availability of CAN networking, and relatively large flash memory. May’s design includes pin outs for an Ethernet port, PWM channels, analog to digital converter, and general purpose digital I/O ports. His design enables the board to be stacked onto another board, in this case the power regulation board, to reduce wiring and failure modes. He recommends all future designs to incorporate this design feature. The motor controller May chooses is the ELMO Whistle controller which features tunable position, velocity, and current loops for control of a DC motor and receives reference commands via RS232. He explains the process used to tune the position and velocity loops so that they would settle within the time it took for the commanding control loop to send it another reference command. The tension controller will command the position loop at 10 Hz and therefore the gains of the position controller were tuned so that a settling time of 0.1 seconds was achieved. Similarly, the sway controller will command the velocity loop at 50 Hz and therefore the gains of the speed controller were tuned so that a settling time of 0.02 seconds was achieved.

The tether tension feedback circuitry consists of a load cell and a custom built instrumentation board. May chose the Omega LC103 0-25 lb load cell as the sensor for the tension controller. This load cell was placed in line with the tether near the ground vehicle. The instrumentation board provides power distribution, filtering, and amplification for the output of the load cell. The onboard analog to digital converter outputs the amplified load cell signal to an Arduino board which in turn sends the signal to the microcontroller via a 2.4GHz radio. May finishes the first section of his thesis with two recommendations for a future
design. The first recommendation is to focus on safety in future designs. He recommends a tether release for emergency situations with built in redundancy for all failure modes. His second recommendation is to move the tension feedback hardware from the ground vehicle to the rest of the system in order to consolidate the components; removing a failure mode and improving the adaptability of the design.

The second section of May’s thesis describes the position and speed controller models derived to simulate the low-level controllers used for his other work. He uses MATLAB Simulink and the MATLAB Simscape physical modeling toolbox to model both controllers and simulate their outputs. He then verifies the output using the ELMO Whistle motor controller and Maxon Motor. For the position controller he uses a four step process to identify the bandwidth of the controller.

1. Tune the position control loop
2. Model and simulate the position control loop
3. Use system identification techniques on the simulated control loop to find a closed loop model
4. Identify the bandwidth of the model from the frequency response of the closed loop model

Table 2.1 shows the values May supplies in his thesis for the peak time, peak output, steady state output, and percent overshoot. The units I have added the percent error column as a
reference. Since all of the simulated values are larger than the actual values May deems the controller conservative and acceptable for simulating future results. Finally, the bandwidth of the position controller is calculated to be slightly greater than 200 rad/sec which is greater than the 62.8 rad/sec (10 Hz) necessary for the position controller to reach steady state within one sample of the outer loop tension controller. The speed controller is treated in the same way following the same steps described above. Table 1 also shows the actual and simulated values for the speed controller. Again, since the values for the simulated controller are larger than the real one, the model is deemed conservative and acceptable for simulation. The bandwidth of the speed controller is found to be greater than 1000 rad/sec which is greater than the 314 rad/sec (50 Hz) necessary for the speed controller to reach steady state within one sample of the sway controller.

Table 2.1: Metrics for models developed in May’s Thesis

<table>
<thead>
<tr>
<th>Metric</th>
<th>Position Controller</th>
<th>Speed Controller</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Actual Values</td>
<td>Simulated Values</td>
</tr>
<tr>
<td>$T_{Peak}$</td>
<td>0.0442 s</td>
<td>0.0506 s</td>
</tr>
<tr>
<td>$Y_{Peak}$</td>
<td>0.0143 m</td>
<td>0.0164 m</td>
</tr>
<tr>
<td>$Y_{ss}$</td>
<td>0.0141 m</td>
<td>0.015 m</td>
</tr>
<tr>
<td>P.O.</td>
<td>1.4 %</td>
<td>8.0 %</td>
</tr>
</tbody>
</table>

May describes in the third section the practical tension controller designed for his thesis. His controller will lower the robot to the ground then maintain tension on the line so that the robot can perform its mission without interruption. Because of the inherent properties of the tether he uses a piecewise model to describe the tension in the line with three modes.
The first is if there is no tension on the line (as the tether cannot support compression), the second is when the line is within the elastic region, and the final mode is when the robot is fully suspended by the line. May proposes a proportional outer loop controller giving commands to the inner loop position controller modeled previously. In his thesis response time is not important and so the controller is tuned for stability only. May tunes the controller with distinct gains for two different cases. The first is lowering the robot from a suspended position to the ground and maintaining a tension of 3.3 lbs. The second case starts with the robot on the ground, and slack in the line, and reeling in the slack until a tension of 3.3 lbs is maintained. The gains were tuned by a trial and error process until stability is achieved. To test the system the hardware was suspended off the ground on a scissors lift and data was collected for both cases. In each case the desired tension was achieved and stability was maintained for the duration of the test.

May’s fourth section develops a theoretical sway controller using only the vertical actuation of the winch. He develops this to deal with the problem of excessive sway of the ground robot while it is being winched up to the helicopter. The excessive sway could affect the stability of the helicopter and cause the system to crash; therefore, the sway must be controlled effectively. He uses only vertical actuation to allow the system to be independent of the helicopter it is used on and to control sway when the tether length is small, a difficult task for helicopter flight controllers. To develop his model May assumes the following:

1. The tether can be approximated as a rigid, massless rod
2. Aerodynamic effects can be ignored

3. The payload can be modeled as a point mass

4. A two dimensional model is sufficient

He then develops the equations of motion of a two dimensional, variable length pendulum using the Lagrangian method. With these equations May creates a model of the system in MATLAB’s Simulink software and verifies that an uncontrolled winching up of the robot would cause dangerous sway to occur, particularly as the length of the line approaches zero. May proposes using autoparametric resonance to remove energy from the swaying system by using the winch to simulate a visco-elastic pendulum in resonance with the sway. He then develops the equations of motion of a single degree of freedom visco-elastic pendulum and couples it with the variable length pendulum by developing a controller to imitate the visco-elastic pendulum’s response. The controller gains are tuned so the frequency of the visco-elastic pendulum is in 2:1 resonance with the pendulum frequency and so energy is taken out of the pendulum. The controller is a second order controller with adaptive parameters such that at each time step the parameters of the controller are found based on the length and velocity of the line. In order to optimize the energy transfer, May varied the damping ratio between zero and 0.5 with 1000 samples. He found an optimal damping ratio which is used in the calculation of the adaptive parameters. May determines that the autoparametric resonance controller is flawed in that it depends on the small angle approximation for the adaptive parameters and therefore a more practical controller is needed. He goes on to
describe a proportional controller which alters the tether velocity based on the difference in tension from a baseline value. May claims that when this occurs the dynamics of the pendulum are altered in such a way that the sway is reduced. He then develops a model of the system and controller in Simulink using the nonlinear pendulum equations of motion. The model is simulated for varying initial angles and the results demonstrate the reduction of sway as the robot is winched up. May goes on to describe how the tension and tether speed are opposite at each point in time resulting in dissipative work. Finally, he tests the system with added noise, both an added sine wave and random noise. Again he simulates the system, with the noise, and finds that despite the noise the controller still reduces the sway in the system. May provides several recommendations to implement this controller in a practical sense. He recommends developing a failsafe break switch system, an emergency tether release system, and, finally, the development of the controller in LabVIEW for easy deployment onto a microcontroller.

May concludes his thesis by giving several general recommendations for future designs. First, he recommends moving the tension sensor from the robot to the pod in order to eliminate noise and communication lag which is present in the system. Second, he recommends consolidating all of the onboard electronics reducing the possibility of error. Finally, he recommends installing an emergency tether release system to avoid dangerous situations with excessive sway.
2.1.3 Key Differences from Directly Related Work

This thesis is directly related to and builds off of May’s thesis. For clarification as to the unique aspects of this work, a comparison of the two theses is presented here. There are four major differences between the two theses. First, new hardware is designed using the recommendations in May’s thesis. The second difference is new speed controller and plant models which are explicitly derived from the physics of the system. The third difference is a new tension controller which uses the speed controller as the inner loop. In my thesis the tension controller is tested both in lab and on an autonomous helicopter. Finally, the fourth difference is that my thesis does not go into sway control as May’s thesis does. Additionally, May’s thesis will be used as a metric for the performance of the controllers found in my thesis.

2.2 Other Literature

Significant work has been done in the realm of tethered operations in helicopters and robotics. In general, tethers attached to a ground control station provide stability, communications, and power transfer while tethered payloads allow for placement of sensors into hazardous environments or multi-robot cooperation. As the field is quite large several examples are given for the three main areas of robotics—ground operations, air operations, and sea operations—as well as the problem of sway control for an underslung payload.
2.2.1 Tethered Unmanned Operations

Some of the first uses of tethered robotics are in the realm of remote operated vehicles (ROVs). These are unmanned, remotely operated submersible vehicles which are intended to be piloted from the ship to which they are tethered. The tether is often necessary for underwater communication as wireless communication is extremely slow in this environment. Nomoto and Hattori [7] present an ROV capable of descending to 3300 meters to survey the ocean floor. This ROV is tethered to the ship and transmits video and sonar from the ROV, and power to the ROV. This vehicle was developed to identify areas where a manned submersible could explore and to explore depths which the only the ROV could sustain. Rife and Rock [8] present a paper on a jellyfish tracking ROV. The ROV is tethered to the ship which again provides power to the ROV and receives sensor data from the ROV. This ROV is capable of autonomously tracking jellyfish, however, the control mechanism is onboard the ship, and therefore must send the commands to the ROV via the tether.

UGVs have also been tethered to deal with a variety of issues. Krishna et al. [9] present a tethered UGV for the exploration of active volcanoes. In this case the tether allows the UGV to overcome several significant problems of this environment. Since the UGV is descending into a volcanic crater, wireless communication with the UGV could be degraded. Therefore, the UGV is tethered to a base station on the rim of the crater which is in constant satellite communication with the operator. The base station then relays the commands to the UGV via the tether. The UGV also needs to descend the potentially steep sides of the crater.
The base station acts as a solid point which the UGV is tethered to, allowing the UGV to maintain tension on the tether and receive support for steep descents and ascents. Finally, because of the issue with steep ascents, the weight of the UGV is minimized by moving the power generation from the UGV to the base station. The base station then supplies the UGV with power through the tether.

Another interesting use of tethers in the UGV world comes from a paper by Ahn et al. describing the use of a tether as a guiding mechanism. The UGV is a service robot which can be guided by sensors attached to a tether used to determine velocity and direction. The output angle of the tether determines the desired rotation of the UGV while the length of tether determines the forward velocity.

Tethered operations for UAVs come in two forms. The first is the UAV carrying a tethered object and the second is the UAV tethered to a relatively stationary object, such as a ship. Ratner and McKerrow present a system where a UAV flies a suspended robotic system in a search and rescue situation. The robotic system was developed to allow the search and rescue operations to occur with minimal disturbances, such as noise and wind, from the UAV. Unlike the ROVs and UGVs presented before, most UAV systems rely on wireless communications. In this case the robotic system is completely self sufficient and communicates with the UAV over wireless with the tether only acting as a support. McKerrow then presents a paper on the second form of a tethered UAV. This UAV is tethered to the environment and moves through the use of ducted fans. An interesting spin on the tethered...
UAV point is that this system is meant to be tethered to a stationary point higher than itself. Therefore, this UAV can move around on the edge of a sphere the length of the tether but it does not use the energy needed to maintain flight that most UAVs require. A third UAV is presented by Muttin [13] for the detection of oil pollution from a ship. The tethered UAV is launched from the ship which provides the UAV with power through the tether. The UAV can then ascend to look for oil at a higher location than the ship would otherwise allow. The use of the tether allows the UAV to maintain its search for a longer period of time and land on the rolling and pitching surface of the ship deck.

2.2.2 Underslung Payload Sway Control

The modeling of an under-slung payload on a helicopter was undertaken by Bisgaard et al. [14] including the model of line slackening and tightening due to payload collision with the ground. This model also took into account multi-lift systems consisting of two or more helicopters and aerodynamic effects of the rotor downwash on the payload. Simulation of the model was completed and verified via flight testing. In a later publication, Bisgaard et al. [15] use an unscented Kalman filter (UKF) to predict the states of the payload with a vision system as the sensor input. This paper also explains a method of determining the length of the line using a dedicated sine estimator. Both the UKF and dedicated sine estimator are flight tested and verified with good results.

The control of a UAV with an under-slung payload has only been minimally researched. Most
research in the field of under-slung payloads deals with stability of manned helicopters and control of cranes. Bernard et al. [16] presents a two-step control of a UAV for under-slung payloads lifted by one or more helicopters. The first controller is an orientation controller intended to be robust against the swaying of the payload. Robustness is accomplished through the use of a torque compensator which takes the force from the line, measured by a force sensor in series with the line, and subtracts the calculated torques from the torques generated by the orientation controller. The second controller is a translational controller which approximates the system as a point mass pendulum with a PI-state-feedback controller and serves as an outer loop to the orientation controller inner loop. This controller scheme is verified using flight data for a single helicopter with an under-slung load and with three helicopters lifting a single payload between them. Another control method is presented in Bisgaard et al. [17] which describes a delayed feedback controller. This controller takes into account the fact that the system is sinusoidal and that the UAV controls have a, sometime substantial, delay. Therefore, the delayed feedback controller takes advantage of the system delay and the sinusoidal nature of the system to dampen the sway of the under-slung payload. This controller was simulated and verified through flight testing. A third controller is presented by Omar [18] expanding upon the work presented in Bisgaard et al. [17] to implement a fuzzy version of the delayed feedback controller. This controller is then simulated and the data is presented. A fourth controller is presented in Bisgaard et al. [19] which again expands on Bisgaard et al. [17] but adds an input shaping feed forward controller. This combination feed forward controller and feedback controller is then simulated
and verified through flight testing.

While all of the previously described controllers assume a rigid line between the helicopter and the payload, Bernard and Kondak [20] propose a flexible rope. This study uses a pivot point at the helicopter and encoders to determine the sway of the payload. On top of the relatively low frequency of the pendulum, Bernard and Kondak found a higher frequency on the rope due to the rope fundamental eigenfrequency. This higher frequency causes the encoders to misrepresent the sway of the line. Therefore an observer is proposed, assuming a rigid representation of the line. This observer is simulated and flight tested, with both a single under-slung payload and three helicopters lifting a single payload.
Chapter 3

Winch Pod Hardware and Software

The winch pod hardware and software allow the user, and the tension controller, to raise and lower a payload and monitor the tension of the tether. The winch pod is the culmination of work done by several members of the USL. This second generation winch pod was designed, specifically, to fit on the USL’s Yamaha RMAX helicopter UAV shown in Figure 3.1. Notice that the payload area of the UAV is between the landing gear where, in the figure, a maroon trapezoidal payload is currently placed. The RMAX can carry a payload with a maximum weight of 28 kg and maximum dimensions of $540 \times 510 \times 260$ mm. Therefore, the winch pod must be light weight while still being able to carry the UGV shown in Figure 3.2. This is the UGV discussed in Rose’s thesis [3]. The final design of the winch pod is shown in Figure 3.3 showing all of hardware necessary to run the tension controller. This design is meant to be non-specific to the RMAX and can be used on any similarly sized UAV with the appropriate
The rest of this chapter details the hardware and software of the winch pod. The first section deals with the structural hardware of the winch pod. The second section outlines the winch hardware, including the winch motor and motor controller. The third section details the electronics hardware, including the power and sensor distribution board as well as the MCB2300 ARM7 development board. The fourth, and final, section deals with the software on both the MCB2300 board and the ground control station.
Figure 3.3: Winch pod hardware
3.1 Structure hardware

The structural hardware of the winch pod, which was designed and fabricated by another student in the Unmanned Systems Lab, includes the aluminum frame, which encompasses the UGV while in flight, and a carbon fiber plate holding the rest of the hardware. Figures 3.4a and 3.4b show the aluminum tube and carbon fiber plate structure. Notice that the aluminum frame is made of bent circular aluminum tubing which has been welded together. The flat plate on top is made from carbon fiber, chosen for its high stiffness and light weight. This structure was designed to be able to hold 50 lbs in the center of the carbon fiber plate, in the same way that a tethered payload would affect the system. Since the weight of the UGV is only 10 lbs, this gives a factor of safety to the structural hardware. Figure 3.4c shows the winch pod mounted in the landing gear with the UGV embedded in it. Figure 3.4d shows the feet of the winch pod. These feet allow the pod to be attached to the UAV with quick release tabs. Therefore, this system can quickly, and easily, be transferred between UAVs as necessary.
Figure 3.4: Winch pod structure
3.2 Winch hardware

The winch hardware centers around the winch motor, a Maxon 24 V DC motor, shown in Figure 3.5b. Attached directly to the motor are the gear head, shown in Figure 3.5a, as well as a brake and an encoder. The requirements and choice of the winch motor are described by May in his thesis [6]. The encoder output and the motor input are attached directly to the motor controller while the brake system is actuated separately by the MCB2300 board.

The gear head is a 26:1 ratio gear head, decreasing the speed of the motor and increasing the torque. The encoder measures 2000 counts per revolution of the motor shaft and outputs that data directly to the motor controller. The brake system is a 24 V active off system, so that the brake is only lifted when 24 volts are passed through the brake wires. These four components make up the central part of the winch system.

Attached to the winch system is the custom designed aluminum spool with the tether wrapped around it. Figure 3.6 shows the design of the aluminum spool. The diameter of the spooling area slopes down toward the center to passively keep the tether wrapped
around the center of the spool. The tether is a braided spectra 150 lb test fishing line. In order to determine the spring constant of the tether, a test rig was set up to measure the displacement of the tether due to varying forces, using weights and gravity, at two different points along the line, approximately 9 meters apart. The two points were chosen to not be at the ends of the tether so as to eliminate displacement caused by the settling of the knots. Table 3.1 shows the data collected while figure 3.7 shows a plot of the force versus displacement of the tether. This data gives a spring constant for 9 meters and, assuming the spring constant is linear with the length of the line, the spring constant of the tether, given its length, is given in the following equation.

$$k(l) = \frac{15476 \, Nm}{l \, m}$$  \hspace{1cm} (3.1)

where \( l \) is the length of the tether and \( k(l) \) is the spring constant given the tether length.

In order to measure the tension of the tether, after the tether leaves the spool horizontally, it travels over a pulley type system which transfers the tension force in the line to a load cell to be measured. Since the tether passes from the spool over the pulley, a spring loaded emergency cut-off mechanism is implemented in this location. In the event of the tether becoming snagged in the environment, it would be necessary to leave the UGV at that location in order to bring the UAV back. Therefore, the spring loaded emergency cut-off mechanism allows the user to cut the tether and save the UAV.

The last component of the winch hardware is the motor controller. The ELMO Solo Whistle
Figure 3.6: Aluminum spool

Table 3.1: Tether stiffness test data

<table>
<thead>
<tr>
<th>Weight (lb)</th>
<th>$d_1$ (in)</th>
<th>$d_2$ in</th>
<th>$d_2 - d_1$ (in)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.375</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>3.25</td>
<td>0.113</td>
<td>0.402</td>
<td>0.289</td>
</tr>
<tr>
<td>5.81</td>
<td>0.158</td>
<td>0.700</td>
<td>0.542</td>
</tr>
<tr>
<td>8.25</td>
<td>0.225</td>
<td>1.032</td>
<td>0.807</td>
</tr>
<tr>
<td>10.44</td>
<td>0.303</td>
<td>1.317</td>
<td>1.014</td>
</tr>
<tr>
<td>13.13</td>
<td>0.352</td>
<td>1.622</td>
<td>1.270</td>
</tr>
<tr>
<td>16.31</td>
<td>0.444</td>
<td>1.942</td>
<td>1.498</td>
</tr>
<tr>
<td>17.81</td>
<td>0.444</td>
<td>2.345</td>
<td>1.901</td>
</tr>
</tbody>
</table>
Figure 3.7: Experimental stiffness data collected at 9 meters
motor controller is a development system for the ELMO Whistle motor controller. This allows the system to be used in a stand alone fashion and connected to the MCB2300 by serial. The Solo Whistle takes the encoder measurements and determines the angular position, velocity, and acceleration of the motor. The motor controller can also take input, over serial or CAN, in the form of position commands, velocity commands, and current commands. The controller then outputs voltage to the motor, driving the system.

Figure 3.8: ELMO Solo-Whistle motor controller (Elmo Motion Control, used with permission)
3.3 Electronics hardware

The electronics hardware chosen for this thesis consists of the MCB2300 microcontroller board, an Omega load cell, a power and signal distribution board, and a LiPo battery. The MCB2300 microcontroller board is a development board for the ARM7 microprocessor and is shown in Figure 3.9. The board provides all of the electronics to run the ARM7 processor as well as pinning out much of the functionality. This includes two serial ports, two CAN ports, a Ethernet port, as well as eight digital I/O lines, eight analog inputs, and one analog output. This ARM7 processor can be programmed using the LabVIEW programming software which speeds up the development time and doesn’t necessitate the learning of C. The MCB2300 can take as inputs all of the signals from the winch pod components and communicate, over serial, with the motor controller. Simultaneously, the MCB2300 board can be communicating with the ground control station via a second serial port connected to a radio. For these reasons the MCB2300 board was chosen to act as the controller board for the winch pod.

Figure 3.9: KEIL MCB2300 evaluation board (Keil, Tools by ARM; used with permission)
The power and signal distribution board routes the battery power and sensor signals from the various components of the winch pod to their final destinations. The board design is shown in Figure 3.10. This board takes the sensor outputs from the Omega load cell and the emergency cut-off mechanism’s linear actuator, amplifies the signals, and sends them to the MCB2300 board for processing. The MCB2300 board sends digital signals to power and signal distribution board which control the actuation of the linear actuator and the brake component of the winch. Finally the board regulates and distributes the power to all of the components on the winch, with the exception of the motor controller which regulates its own voltage internally. The battery which was chosen to provide power for the winch pod is a 6 cell 22.2 volt, 5000 mAh LiPo battery.
3.4 Software

The software for the winch pod was designed for this thesis and is split into two distinct programs. The first is the ground control station (GCS) program, which allows the user to interact with the winch pod. This program is written in LabVIEW, a graphical programming language, and the front panel, what the user sees, is shown in Figure 3.11. This program performs three main tasks. The first is to send commands to the winch pod. These commands are either motor controller commands, such as commanding directly the speed of the motor; microcontroller commands, such as turning on the tension controller or the brake; or a watchdog to ensure continued communication. The second task is to receive, parse, and display data from the winch pod. The final task is to record and timestamp the tension in the line and the commands of the tension controller.

The second software program is the embedded software on the MCB2300 board. This program is written in LabVIEW embedded, a special module of the LabVIEW programming language which converts the program to C code and loads it onto the microcontroller. The embedded program has no front panel as there is no way for a user to see the output of the program, except through the peripherals of the microcontroller itself. This program is in constant communication with the GCS program through the heartbeat mechanism. If at any point in time the heartbeat is not received, the program stops the motor and waits for communication to be restored. Otherwise the program receives commands from the GCS and, either sends them to the motor controller or carries out the necessary tasks. This pro-
Figure 3.11: Ground control station software front panel
gram also has the implementation of the tension controller on board and outputs velocity commands to the motor controller based on the load cell input when the tension controller is turned on. The embedded program is constantly receiving information on the tension in the line and sending that information back to the GCS to be viewed by the user. Finally, the embedded program receives information back from the motor controller and, when the tension controller is not running, sends that information back to the GCS.

The combination of the hardware and the software presented in this chapter allow the tension controller to perform its task. The hardware ensures that output of the tension controller will successfully maintain tension on the tether while the software gives feedback to the user. Together these components are the physical hardware on which the tension controller is deployed.
pac
Chapter 4

Closed Loop Speed Control System

Simulation

This chapter details the simulation of the closed loop speed control system which acts as the inner-loop for the closed loop tension control system described later in this thesis. The closed loop speed control system consists of the DC motor plant, the inner-loop ELMO current controller, and the inner-loop ELMO velocity controller as shown in Figure 4.1. The plant is composed of a brushed DC motor, the Maxon motor; a gearhead; a spool; and a spring connected to a weight to simulate a tethered UGV. The ELMO velocity controller and the ELMO current controller are two of the controllers on the ELMO Solo Whistle digital servo drive. Using the physical properties of the plant and the documentation provided from the ELMO Motion Control company about the ELMO Solo Whistle, this chapter derives
a parametric model of the plant and a more accurate closed loop speed control system simulation than previously developed.

![Figure 4.1: Speed control block diagram](image_url)

The importance of an accurate, parametric closed loop speed control system simulation is two fold. First, the accuracy of the closed loop tension control system simulation is dependent on the accuracy of the closed loop speed control system simulation, as the tension controller will command the closed loop speed control system. As the closed loop speed control system simulation becomes less accurate, the closed loop tension control system simulation performance will suffer as well causing it to become less relevant to the real world. Second, the plant model of the closed loop speed control system must be parametric, as opposed to derived from system ID techniques, in order to allow it to be used for different tether spring constants, as the length of the tether varies, and, for future iterations of the physical design, if any physical component changes. In May’s thesis [6], he describes both the position controller and the velocity controller found in the ELMO Solo Whistle. For my thesis only the velocity controller is simulated, as the velocity controller is the inner-loop to the position controller and, therefore, the velocity controller has a larger bandwidth.
Note that figure 4.1 does not show the position controller. The position controller must also complete a move before executing a second command while the velocity controller will execute a new command immediately, making the velocity controller more ideally suited for the inner loop controller to the tension controller.

The main objective for the closed loop speed control system simulation is to improve the accuracy of the output of the simulation from the one found in May’s thesis. Table 4.1 shows the percent error between experimental and simulated values of the output of the closed loop speed control system developed in May’s thesis for a step input in reference speed. This table shows the peak time, $T_{peak}$; peak value, $Y_{peak}$; and the steady state value, $Y_{ss}$ of the output of the closed loop speed control system. Included in this table is the experimental versus simulated percent overshoot, P.O., as well. Therefore, the closed loop speed control system simulation developed in this thesis should more closely reflect the actual closed loop speed control system then the previous one by reducing the percent error between the experimental and simulated outputs for the metrics described in Table 4.1.

Table 4.1: Metrics for closed loop speed control system developed in May’s Thesis

<table>
<thead>
<tr>
<th>Metric</th>
<th>Experimental Values</th>
<th>Simulated Values</th>
<th>Percent Error</th>
</tr>
</thead>
<tbody>
<tr>
<td>$T_{peak}$</td>
<td>0.0037 s</td>
<td>0.0049 s</td>
<td>32.4 %</td>
</tr>
<tr>
<td>$Y_{peak}$</td>
<td>0.142 m/s</td>
<td>0.169 m/s</td>
<td>19.0 %</td>
</tr>
<tr>
<td>$Y_{ss}$</td>
<td>0.118 m/s</td>
<td>0.153 m/s</td>
<td>29.7 %</td>
</tr>
<tr>
<td>P.O.</td>
<td>20.3 %</td>
<td>10.4 %</td>
<td></td>
</tr>
</tbody>
</table>

The rest of this chapter goes into the specifics of the modeling for the plant and the simu-
lation of the speed control system. The first section goes into detail on the modeling of the plant. This includes the derivation of the equations of motion for the DC motor configuration as well as the Simulink modeling of the plant. The second section discusses the simulation of the ELMO current controller, specifically the Simulink model of the controller. Similar to the second section, the third section develops the Simulink model of the ELMO velocity controller. The fourth section details the combination of the three Simulink models developed and displays the test results and verification of the closed loop speed control system simulation. The final section displays the bandwidth and peak times of the speed control system model.
4.1 Plant Model

The derivation of the equations of motion for a DC motor has been provided in many textbooks, such as Ogata [21] and Bolton [22]. The winch motor model takes into account the dynamics of the brushed DC motor itself, a Maxon Motor RE-40 DC motor, as well as the components attached to it. Figure 4.2 shows a schematic of the DC motor, gearhead, and spool, including their moments of inertia, \( J_1, J_2, \) and \( J_3 \) respectively. The spool component of the figure includes, internally, the torque due to the force of the tension present on the tether, \( T_R \), and the rotational position of the spool, \( \theta_3 \). The motor component of the figure shows the electronics of the armature of the DC motor, including the resistance, \( R \), and the inductance, \( L \), due to the length and winding of the armature wire, as well as the voltage, \( V \), and the current, \( i \), driven by the power source. The motor provides a back emf, \( E \), to the electrical circuit and a torque to the rest of the figure, \( T_m \). The torque and rotational position of the motor shaft, \( \theta_1 \), is then passed through the motor shaft into the gearhead component, amplifying torque and reducing the angular velocity of the shaft of the gearhead by the ratio of the size of the gears, \( \frac{n_1}{n_2} \). The gearhead shaft position, \( \theta_2 \), directly changes the position of the spool to which the tether and UGV are attached. The UGV then provides a torque to the system by applying a force through the tether at the radius of the spool. The following equation is developed by applying Kirchoff’s current law to the circuit portion of the motor component in Figure 4.2.
\[ V - iR - iL - E = 0 \]  \hspace{1cm} (4.1)

where \( V \) is the voltage generated by a voltage source, \( i \) is the current through the circuit, \( R \) is the resistance in the windings of the armature of the motor, \( \dot{i} \) is the rate of change of the current with respect to time through the circuit, \( L \) is the inductance generated by the winding of the wire in the armature of the motor, and \( E \) is the back-emf of the motor. This equation defines how the voltage, and therefore the current, interacts with the windings and drives the motor. According to Lenz’s law when the motor is rotating, the magnetic field of the rotating shaft produces a back-emf proportional to the speed. Therefore, the equation \( E = K_b \dot{\theta}_1 \) holds where \( K_b \) is the back-emf constant of the motor and \( \dot{\theta}_1 \) is the angular velocity of the motor shaft. \( K_b \dot{\theta}_1 \) can then be substituted for the back-emf in Equation 4.1.
and solving for the derivative of $i$ yields the following equation.

$$V - iR - iL - K_b \dot{\theta}_1 = 0$$

$$\dot{i}L = V - iR - K_b \dot{\theta}_1$$

$$i = \frac{1}{L} V - \frac{R}{L} i - \frac{K_b}{L} \dot{\theta}_1$$ \hspace{1cm} (4.2)

Equation 4.2 shows that the current, voltage, and angular velocity of the motor are related. This equation describes the driving of the motor with an applied voltage and a resultant current through the windings. A second equation is required to represent the mechanical dynamics. This second equation is derived from the dynamics of the spinning motor shaft and shaft load. The dynamic equations of motion for the right half of Figure 4.2 are found through multiple applications of Newton’s second law for rotation. Newton’s second law for rotation states that the sum of the external torques is equal to the angular acceleration times the moment of inertia of the rotating body; $\sum T = J \ddot{\theta}$ where $T$ is an external torque, $J$ is the moment of inertia of the rotating body, and $\ddot{\theta}$ is the angular acceleration of the rotating body. Applying Newton’s second law to the shaft between the motor and the gearhead yields the following equation.
\[ J_1 \ddot{\theta}_1 = T_m - b_1 \dot{\theta}_1 - T_1 \]

\[ T_m = J_1 \ddot{\theta}_1 + b_1 \dot{\theta}_1 + T_1 \quad (4.3) \]

where \( J_1 \) is the moment of inertia of the motor shaft, \( \ddot{\theta}_1 \) is the angular acceleration of the motor shaft, \( b_1 \) is the viscous friction coefficient of the first gear, \( T_1 \) is the torque due to the second part of the gearhead, and \( T_m \) is the torque generated by the motor. Applying Newton’s second law to the shaft of the gearhead yields the following equation.

\[ J_2 \ddot{\theta}_2 = T_2 - b_2 \dot{\theta}_2 - T_L \]

\[ T_2 = J_2 \ddot{\theta}_2 + b_2 \dot{\theta}_2 + T_L \quad (4.4) \]

where \( J_2 \) is the moment of inertia of the gearhead; \( \ddot{\theta}_2 \) is the angular acceleration of the shaft of the gearhead; \( b_2 \) is the viscous friction coefficient of the second gear; \( \dot{\theta}_2 \) is the angular velocity of the shaft of the gearhead; \( T_L \) is the torque of the load, in this case the torque generated by the UGV and the moment of inertia of the spool; and \( T_2 \) is the torque due to the first part of the gearhead. In order to combine Equations 4.3 and 4.4 a constraint
equation is required. Since $T_1$ and $T_2$ are acting through the gearhead and assuming the power, $P = T\dot{\theta}$, is the same on either end of the gearhead, the following equation holds true.

$$T_1\dot{\theta}_1 = T_2\dot{\theta}_2$$  \hfill (4.5)

Equation (4.5) can then be rewritten in the following form.

$$T_1 = T_2 \frac{\dot{\theta}_2}{\dot{\theta}_1} = T_2 \frac{\dot{\theta}_2}{\dot{\theta}_1} = T_2 \frac{n_1}{n_2}$$  \hfill (4.6)

where $\theta_1$ is the angular position of the motor shaft, $\theta_2$ is the angular position of the gearhead shaft, and $\frac{n_1}{n_2}$ is the gear ratio of the gearhead. Substituting Equations (4.4) and (4.6) into Equation (4.3) yields the following equation.

$$T_m = J_1\ddot{\theta}_1 + b_1\dot{\theta}_1 + \frac{n_1}{n_2} (J_2\ddot{\theta}_2 + b_2\dot{\theta}_2 + T_L)$$  \hfill (4.7)

Since the load torque on the shaft of the gearhead, $T_L$, is generated by the moment of inertia of the spool and the torque generated by the tension of the UGV on the tether at the radius of the spool, we can say that $T_L = J_3\ddot{\theta}_3 + T_R$, where $J_3$ is the moment of inertia of the spool, $\ddot{\theta}_3$ is the angular acceleration of the spool, and $T_R$ is the torque due to the line tension caused by lifting the UGV. Substituting this equation into Equation (4.7) yields the following
equation.

\[ T_m = J_1 \ddot{\theta}_1 + b_1 \dot{\theta}_1 + \frac{n_1}{n_2} (J_2 \ddot{\theta}_2 + b_2 \dot{\theta}_2 + J_3 \ddot{\theta}_3 + T_R) \]  

(4.8)

However, since the spool and the gearhead are rigidly attached to each other, we can assume that \( \ddot{\theta}_2 = \ddot{\theta}_3 \). Therefore, substituting \( \ddot{\theta}_2 \) for \( \ddot{\theta}_3 \) in Equation 4.8 and simplifying yields the following equation.

\[ T_m = J_1 \ddot{\theta}_1 + b_1 \dot{\theta}_1 + \frac{n_1}{n_2} [(J_2 + J_3) \ddot{\theta}_2 + b_2 \dot{\theta}_2 + T_R] \]

(4.9)

From Equation 4.6 it is known that \( \dot{\theta}_2 = \frac{n_1}{n_2} \dot{\theta}_1 \) and \( \ddot{\theta}_2 = \frac{n_1}{n_2} \ddot{\theta}_1 \). Therefore, substituting into Equation 4.9 and simplifying yields the following equation.

\[ T_m = J_1 \ddot{\theta}_1 + b_1 \dot{\theta}_1 + \frac{n_1}{n_2} [(J_2 + J_3) \dot{\theta}_1 + b_2 \dot{\theta}_2 + T_R] \]

\[ T_m = [J_1 + \left( \frac{n_1}{n_2} \right)^2 (J_2 + J_3)] \dot{\theta}_1 + [b_1 + \left( \frac{n_1}{n_2} \right)^2 b_2] \dot{\theta}_2 + \frac{n_1}{n_2} T_R \]  

(4.10)
The torque generated by the motor is proportional to the current driven through the motor armature wires as described by the following equation.

\[ T_m = K_i \]  \hspace{1cm} (4.11)

where \( K \) is the motor-torque constant. Combining Equations 4.10 and 4.11 yields the following equation.

\[ K_i = \left( J_1 + \left( \frac{n_1}{n_2} \right)^2 (J_2 + J_3) \right) \dot{\theta}_1 + \left[ b_1 + \left( \frac{n_1}{n_2} \right)^2 b_2 \right] \dot{\theta}_1 + \frac{n_1}{n_2} T_R \]  \hspace{1cm} (4.12)

Figure 4.3 shows the UGV tethered to the spool. Since the UGV is tethered to the system at the radius of the spool, it generates a torque which is given by

\[ T_R = r_s F_R \]  \hspace{1cm} (4.13)

where \( T_R \) is the torque due to the tension on the line, \( r_s \) is the radius of the spool, and \( F_R \) is the tension on the line due to the spring and UGV combination.

In the general case for a spring, the force due to the spring is given by \( F_s = K_s x_s \) where \( F_s \) is the force due to the spring, \( K_s \) is the spring constant, and \( x_s \) is the displacement of the spring away from equilibrium. Since the tether, and therefore the spring modeling the tether,
is securely attached to the spool and assuming no slipping at the location where the tether and spool meet, we know that \( x_s = 2\pi r_s \frac{\theta_3}{2\pi} = r_s \theta_3 \). Two assumptions are made here. First, the UGV is on the ground and not moving, and second, the helicopter is at a fixed height and also not moving. Therefore, the force at the spool due to the tether is \( F_R = K_s r_s \theta_3 \).

However, since the tether does not carry compression, only tension, and since the UGV can be lifted off the ground, the tension is described by the following equation.

\[
F_R = \begin{cases} 
0 & \text{if } x_s \leq x_{min} \\
K_s r_s (\theta_3 - \theta_{3,off}) & \text{if } x_{min} < x_s < x_{max} \\
F_{mg} + F_{Dyn} & \text{if } x_s \geq x_{max}
\end{cases}
\]  

(4.14)

where \( K_s \) is the spring constant, \( F_{mg} \) is the weight of the UGV, \( F_{Dyn} \) is the force due to dynamic loading, \( x_{min} \) is the displacement of the tether at which slack forms in the line, \( x_{max} \) is the displacement of the tether at which the UGV is lifted off the ground, and \( \theta_{3,off} \)
is the offset of the spool angular position to deal with the height of the helicopter above the ground. However, similar to the acceleration of the spool, the position of the spool is the same as the position of the gearhead. Therefore, substituting $\theta_2$ for $\theta_3$, as in Equation 4.16 into Equation 4.14 yields the following equation.

\[
F_R = \begin{cases} 
0 & \text{if } x_s \leq x_{min} \\
K_s r_s \left( \frac{n_1}{n_2} \theta_1 - \theta_{3,off} \right) & \text{if } x_{min} < x_s < x_{max} \\
F_{mg} + F_{Dyn} & \text{if } x_s \geq x_{max}
\end{cases}
\]

(4.15)

The torque equation, Equation 4.13 is now substituted into Equation 4.12 yielding the following equation.

\[
K i = \left[ J_1 + \left( \frac{n_1}{n_2} \right)^2 (J_2 + J_3) \right] \ddot{\theta}_1 + \left[ b_1 + \left( \frac{n_1}{n_2} \right)^2 b_2 \right] \dot{\theta}_1 + \frac{n_1}{n_2} r_s F_R
\]

(4.16)

Solving for the motor acceleration in Equation 4.16 yields the following equation.

\[
\ddot{\theta}_1 = -\frac{b_1 + \left( \frac{n_1}{n_2} \right)^2 b_2}{J_1 + \left( \frac{n_1}{n_2} \right)^2 (J_2 + J_3)} \dot{\theta}_1 - \frac{n_1}{n_2} r_s F_R + \frac{K}{J_1 + \left( \frac{n_1}{n_2} \right)^2 (J_2 + J_3)} \dot{i}
\]

(4.17)

Figure 4.4 shows the Simulink model derived from Equations 4.2 and 4.17. Beginning with the right half of the Simulink model, the cyan colored blocks represent Equation 4.17 specifically. The cyan summing junction in the diagram outputs $\ddot{\theta}_1$ and, therefore, the inputs to
Figure 4.4: Simulink model for the speed control system plant
the summing junction are the three terms described on the right side of Equation 4.17. The gains displayed in the Simulink model by the triangular shaped blocks all follow the same basic equation.

\[ y = K_G x \]  

(4.18)

where \( y \) is the output of the block, \( K_G \) is the gain of the block, and \( x \) is the input to the block. The input to the cyan block with label K1 is the current provided by Integrator2. The current passes through the gain block labeled K1, which has the following gain value.

\[
K_{G,K1} = \frac{K}{J_1 + \left( \frac{n_1}{n_2} \right)^2 (J_2 + J_3)}
\]  

(4.19)

The output of the K1 block enters the summing junction which, in turn, outputs the angular acceleration of the motor shaft, \( \ddot{\theta}_1 \). The angular acceleration passes through a continuous-time Integrator block with an initial condition of 0. The output of the Integrator block is the angular velocity of the motor shaft, \( \dot{\theta}_1 \), which feeds into block K2, block K5, and the magenta blocks. The K2 block takes the angular velocity inputs and outputs the following gain.

\[
K_{G,K2} = -\frac{b_1 + \left( \frac{n_1}{n_2} \right)^2 b_2}{J_1 + \left( \frac{n_1}{n_2} \right)^2 (J_2 + J_3)}
\]  

(4.20)
The output of K2 then goes back into the summing junction. The K5 block also takes the angular velocity inputs and converts the angular velocity from radians per second to counts per second, the internal units of the speed controller. The counts come from the total number of counts on the position encoder with 2000 counts per revolution, yielding the following equation.

\[
K_{G,K5} = \frac{2000 \text{ counts/sec}}{2\pi \text{ radians/sec}} = 318.31 \frac{\text{counts/sec}}{\text{radians/sec}}
\]  

(4.21)

The output of the K5 gain block, in counts per second, is passed into the gain block K6 and, simultaneously, passed out of the plant as one of the two outputs of the system. The gain block K6 takes the angular velocity in counts per second and converts it back into radians per second, yielding the following equation.

\[
K_{G,K6} = \frac{1}{K5} = \frac{2\pi \text{ radians/sec}}{2000 \text{ counts/sec}} = \frac{1 \text{ radians/sec}}{318.31 \text{ counts/sec}}
\]  

(4.22)

The converted angular velocity is passed through the Integrator1 block yielding angular position, \(\theta_1\). The angular position is passed into the gain block K3 which outputs the force due to the tension on the line, \(F_R\), with the following gain.

\[
K_{G,K3} = \frac{n_1}{n_2} K_s r_s
\]  

(4.23)
The output of the K3 gain block is passed through the Saturation block which saturates $F_R$ to $0 \leq F_R \leq F_{mg} + F_{Dyn}$. However, this thesis artificially saturates tension in the line at a maximum of $F_R$ assuming that the dynamic loads are zero. The saturated tension force is passed out of the system as another output and into the gain block K4. The K4 block multiplies the saturated tension force by the following gain.

$$K_{G,K4} = -\frac{\left(\frac{n_1}{n_2}\right) r_s}{J_1 + \left(\frac{n_1}{n_2}\right)^2 \left(J_2 + J_3\right)}$$  \hspace{1cm} (4.24)$$

The output of the K4 block is passed into the summing junction completing Equation 4.17.

Similar to the cyan blocks, the magenta blocks represent Equation 4.2. Here the summing junction outputs the derivative of current, the left side of Equation 4.2, and takes as input the three components on the right side of the equation. The input to the system, the voltage, first passes through the block labeled Gain which has the following gain value.

$$K_{G,Gain} = \frac{1}{L}$$  \hspace{1cm} (4.25)$$

The output of this block then enters the summing junction becoming its first input. The output of the summing junction, the time derivative of current, passes into the Integrator2 block which, in turn, outputs current. The current is then passed into the cyan blocks, the block labeled Gain1, and out of the system as an output. Gain1 has the following gain value.
\[ K_{G,Gain1} = \frac{R}{L} \]  \hspace{1cm} (4.26)

The output of Gain1 is then passed into the summing junction becoming the second input. Finally, the last input to the summing junction comes from the block labeled Gain2. This block takes as input the angular velocity, output from the cyan blocks, and multiplies it by the following gain.

\[ K_{G,Gain2} = \frac{K}{L} \]  \hspace{1cm} (4.27)

Here the Simulink model deviates slightly, though inconsequentially, from Equation 4.2. Equation 4.2 dictates that the angular velocity should be multiplied by \( K_b \), the back-emf constant of the motor, rather than \( K \), the motor-torque constant. However, since the back-emf constant is in units of \( \frac{V}{\text{rad/s}} \) and the motor-torque constant is in units of \( \frac{Nm}{A} \), then the value of \( K_b \) is equal to \( K \). Therefore, a single constant is used, \( K \), rather than two constants.

Both the cyan and magenta blocks combine to output the current, tension, and angular velocity of the plant given a voltage input. The current is then fed into the ELMO current controller and the angular velocity is fed into the ELMO velocity controller representing the inner loop controllers of the ELMO motor controller. The tension is used later to provide feedback for the tension controller. This Simulink model provides a complete brushed-DC motor model including torque from a tethered UGV.
4.2 ELMO Current Controller

The ELMO current controller is the inner most controller of the ELMO Solo Whistle digital servo drive. Figure 4.5 shows the block diagram which is provided as part of the ELMO Motion Control SimplIQ Software Manual [23]. The block diagram shows a picture of the most generic form of the current controller. Because the ELMO Solo Whistle is capable of controlling brushless, three-input DC motors, it must be able to source and sink current between the three leads of the motor so that the total current into the motor is equal to the total current out. However, since the motor used in this thesis is a brushed, two-input DC motor I do not need to deal with the intricacies inherent in the three leads. Quite simply, the current going into one lead is the current coming out of the other and if the currents are reversed then the motor rotational direction is reversed. So, the ELMO current controller condenses into three main components not including the power bridge and motor. These three components are represented by the following blocks: the Peak/Continuous limit saturation block, the Pre-filter block, and the Q P+I controller block. In this case, the coordinate transform blocks and the D P+I controller block either pass through the signal or pass no signal depending on the block. Finally, the Power bridge & motor block has been explained in Section 4.1 as the plant model. Given the above, Figure 4.5 can be simplified into Figure 4.6 showing the block diagram of the simplified ELMO current controller.

The input to this controller is a torque command, labeled the same, and is present on the left most side of Figure 4.5. The torque command, however, is actually a current command as the
Figure 4.5: Block diagram of the ELMO current controller (Elmo Motion Control, used with permission)

Figure 4.6: Simplified block diagram of the ELMO current controller
units are in amps. The torque command is then limited based on the maximum amperage allowed by the motor and filtered by the Pre-filter, a low pass filter, before entering the summation block. The summation block takes the limited and filtered current reference value and subtracts the actual current value to find the error which is passed to the Q P+I controller block. This block, as the name suggests, is a PI controller outputting voltage to the motor through a PWM signal. Since the PWM signal has a clock rate of 40 MHz and since the low pass filter bandwidth of PWM circuit is unknown, I assume that the voltage change happens immediately in the Simulink model.

The Simulink model shown in Figure 4.7 is very similar to the condensed current controller block diagram. On the far left are the inputs, Current and Current Feedback, enter into the controller. The current then enters the block labeled Saturation1 which saturates the current to ±5 amps, a programmatical limit to the current. The saturated current then enters the block labeled Rate Limiter, which limits the rate of change of the saturated current to the following equation.

\[- \frac{XP[5]}{21000} \frac{MC}{TS} \leq \dot{i} \leq \frac{XP[5]}{21000} \frac{MC}{TS}\]  \hspace{1cm} (4.28)

where XP[5], MC, and TS are all changeable variables internal to the ELMO Solo Whistle. XP[5] is the step limiter for torque command filter, MC is the largest available drive current, and TS is the current controller sampling time. The limited current then passes into the block labeled Transfer Fcn which is a second order low pass filter with the cutoff frequency
given by the following equation.

\[ f = -\frac{\log(1 - \frac{XP[6]}{32767})}{2\pi \times TS \times 10^{-6}} H\ddot{z} = 1505 Hz \]  \hspace{1cm} (4.29)

where XP[6] is an ELMO internal variable for the time constant for the torque command filter. The output of the filter then enters the first summation block along with the Current Feedback. The output of the summation block, the error between the filtered current and the current feedback, is then passed into the integrator block and the block labeled Gain. The Gain block multiplies the error by the proportional gain, KP[1], and outputs to the second summation block. The integrator block outputs the integral of the error to the block labeled Gain1 which multiplies the integral of the error by the integral gain, KI[1], and outputs to the second summation block. The second summation block outputs voltage which is passed to the block labeled Saturation. The saturation block ensures the voltage stays with the minimum and maximum limits for the motor, in this case the voltage must be between \(-24 \leq V \leq 24\) volts. The output from this block is then passed to the output of the controller as the voltage to be applied to the motor.
Figure 4.7: Simulink model of the ELMO current controller
4.3 ELMO Velocity Controller

The digital servo driver chosen was the ELMO Motion Control Solo Whistle. This driver utilizes an internal angular velocity controller outputting a current command to the current controller. The angular velocity controller uses, at its core, a PI controller architecture with a gain-scheduler found through auto-tuning software provided by ELMO Motion Control. The angular velocity controller block diagram shown in Figure 4.8 consists of five components: a feed-forward controller, a gain-scheduler, a PI controller, a speed estimator for feedback, and a high-order filter.

The feed-forward controller, labeled FF[1] in Figure 4.8 provides control based on the reference velocity alone and does not take into account the actual velocity of the motor. The implementation of the angular velocity controller in this thesis does not use the feed-forward controller and it has been disabled in the motor driver. The gain-scheduler, labeled Automatic Controller Selector in Figure 4.8 chooses the gains for the PI controller based on the reference velocity. The feedback controller, labeled Speed controller: KP, KI in Figure 4.8 uses the PI controller architecture to command current based on the error between the reference speed and the actual speed. The speed estimator, labeled Speed Estimator in Figure 4.8 takes the encoder measurements of the position of the motor and outputs the speed based on these measurements. The last component of the angular velocity controller is the filter labeled High Order Filter in Figure 4.8. This filter takes the output of the PI controller and filters the current before it enters the plant. The High Order Filter was not used in
Figure 4.8: Block diagram of the angular velocity controller (Elmo Motion Control, used with permission)
the Simulink model because it was disabled in the motor driver as it added unnecessary functionality and adds phase lag, limiting the achievable performance of the controller. The simplified ELMO velocity controller implemented in this thesis is shown in Figure 4.9.

Figure 4.9: Simplified block diagram of the angular velocity controller

Figure 4.10 shows the Simulink model derived from the block diagram of the angular velocity controller in Figure 4.8. Of the five components in Figure 4.8, only the Automatic Controller Selector and the Speed Controller exist in the Simulink model as shown in Figure 4.10. The feedforward and filter components have been turned off in the motor driver as they provided unnecessary functionality and the speed estimator is assumed to be perfect in the Simulink model. The inputs to the controller are shown on the left and consist of Speed Command and Speed Feedback. The speed command signal enters the Automatic Controller Selector and the Rate Limiter blocks. The Rate Limiter block takes the commanded speed and limits
Figure 4.10: Simulink model of the angular velocity controller

Figure 4.11: Simulink model of the Automatic Controller Selector
the acceleration of the motor to the user defined bounds. The Automatic Controller Selector block is the subsystem shown in Figure 4.11. The speed command enters the Automatic Controller Selector block and is used as the input to both of the gain scheduling blocks. The gain scheduling blocks, labeled Ki GS and Kp GS, are lookup tables with the input being the speed command and output being the integral gain, for the Ki GS block, and the proportional gain, for the Kp GS block. The gain scheduling parameters used for the automatic controller are found in Appendix A. In Figure 4.10, the speed feedback input also enters the summation block providing the error between the rate limited speed command and the speed feedback. The error, as well as the gains Ki and Kp, enter the Speed Controller block. The current output equation for the Speed Controller is given by the following equation found in the ELMO Motion Control SimplIQ Software Manual [23].

\[
I(t) = K_{\text{speed}} \left( \int_0^t K_I e_{\text{speed}}(\tau) d\tau + K_P e_{\text{speed}}(t)p(t) \right) \tag{4.30}
\]

where \(I(t)\) is the current output of the controller, \(K_{\text{speed}}\) is the conversion factor from the internal scale of the motor driver to current in amperes, \(K_I\) is the integral gain, \(e_{\text{speed}}\) is the difference between the speed commanded and the speed feedback, \(K_P\) is the proportional gain, and \(p(t)\) is the low speed stabilizer. Since perfect speed estimation is assumed, the low speed stabilizer becomes \(p(t) = 1\) and Equation 4.30 can be simplified to yield the following equation.
\[ I(t) = K_{\text{Speed}} \left( \int_0^t K_I e_{\text{Speed}}(\tau) d\tau + K_P e_{\text{Speed}}(t) \right) \] (4.31)

Figure 4.12 shows the Simulink model of the speed controller subsystem found in Figure 4.10 and described by Equation 4.31. The error, integral gain, and proportional gain serve as inputs to the system. The error passes into the product block labeled Proportional Product and simultaneously into the Integrator block. The Integrator block saturates the integral of the error to \(-7.5 \leq \int e_{\text{Speed}} \leq 7.5\) counts, as an anti-windup mechanism. The proportional gain and integral gains enter their respective product blocks. The output of the product blocks are passed through the summing junction, indicated by the round block in Figure 4.12. Finally, the gain block, labeled KO in Figure 4.12 multiplies the output of the summing junction by the gain \(K_{\text{Speed}}\), making the current, in amperes, the output of the controller.

Figure 4.12: Simulink model of the speed controller for the motor driver
4.4 Test Results

After the Simulink models for the speed controller, current controller and plant were designed, they were combined and the output was compared against experimental data for verification. Figure 4.13 shows the Simulink model of the speed control model including the Simulink models of the plant, found in Figure 4.4, the ELMO velocity controller, found in Figure 4.10, and the ELMO current controller, found in Figure 4.7. Comparing the Simulink closed loop speed control system model in Figure 4.13 with the speed control system block diagram presented at the beginning of this chapter, Figure 4.1, the final Simulink model closely matches, at least in block diagram form, the original diagram of the closed loop speed control system. However, testing was also conducted to verify that the Simulink model was a good match to the actual closed loop speed control system.

![Simulink model of the closed loop speed control system model](image)

In order to collect repeatable data from the Winch Pod, a test stand was developed at the USL. The Winch Pod was bolted to two 80/20 beams and suspended off the edge of a table. The beams were clamped to the table providing a solid support for the Winch Pod. Springs, of known stiffnesses, were attached in series with the tether to simulate much longer tether
lengths than the height of the table would allow. Each spring was then attached to a 10 lb weight simulating the weight of the UGV. The values used for the constants throughout this chapter are presented in Appendix A. Finally, the speed controller was allowed to run and collect data for three trials each of three spring constants at seven reference inputs. Each of these trials were run to look at the transient response of the plant, as the steady state dynamics of the closed loop speed control system are trivial. Figure 4.14 shows the experimental versus simulated data for one trial with seven reference values at a spring constant of 981 N/m, or approximately 15.8 m of tether.

Figure 4.14: Experimental versus simulated velocity data for spring constant of 981 N/m

The experimental data shown in Figure 4.14 were collected from the Solo Whistle motor
driver itself and, as seen from this figure, the simulated velocities match closely, in the qualitative sense, with the experimental velocities in rise time, overshoot, and general appearance. However, as discussed in the introduction to this chapter, the goal of this closed loop speed control system simulation is to improve upon the previous simulation by decreasing the percent error for the three metrics described in May’s Thesis: peak time, peak value, and steady state value. After collecting 63 sets of experimental data with varying reference velocities and spring constants, and taking the percent error between the experimental and simulated values for each of the three metrics, Figure 4.15 shows the box plots of the percent errors. The box plots show the median value, the red line; the upper and lower quartiles, which make up the blue box and 50 percent of the data; and the largest and smallest values, which are shown by the black dashed lines extending from the blue box. The box plot also shows outliers of the data which are indicated by the red plus signs.

The box plots show several key things. First of all, the percent errors, for the three metrics, are less than 10% except for the two outliers in the peak time metric. The outliers occurred in the 10,000 cts/s reference case and, therefore, the peak time occurs at less than 0.01 seconds. Due to the noise in the experimental velocity measurements and the relatively small values within which the peak time occurs in, a larger percent error occurred in this case. However, this occurred in only two of 63 simulations and, so, is not significant to the closed loop speed control system as a whole. Secondly, the median cases for the percent error of the three metrics occur at less than 4%. Since the maximum percent error is less than 10%, except for the two outliers, and the median error is less than 4%, this closed loop speed control system...
Figure 4.15: Box plot of percent error for various metrics of simulated versus experimental data
simulation is verified against experimental data. However, the closed loop speed control system simulation must also significantly improve upon the previous one, therefore, Table 4.2 shows a comparison of May’s simulation to the current simulation, both median percent error and maximum percent error, for the three metrics. This table demonstrates that the current closed loop speed control system simulation improves upon the previous simulation significantly, reducing the maximum percent error in the peak time by 14.2 percentage points in the maximum case. The percentage error of the peak value is similarly reduced by 14.1 percentage points in the maximum case while the percentage error of the steady state value was reduced by 26.7 percentage points in the maximum case. In each case the current closed loop speed control system simulation improves upon the previous simulation by a significant amount, realizing the goal for recreating this simulation.

Table 4.2: Comparison of metrics for the closed loop speed control system simulation

<table>
<thead>
<tr>
<th>Metric</th>
<th>May’s Thesis Simulation</th>
<th>Current Simulation</th>
</tr>
</thead>
<tbody>
<tr>
<td>$T_{\text{Peak}}$ (s)</td>
<td>32.4 %</td>
<td>3.75 %</td>
</tr>
<tr>
<td>$Y_{\text{Peak}}$ (cts)</td>
<td>19.0 %</td>
<td>3.23 %</td>
</tr>
<tr>
<td>$Y_{\text{ss}}$ (cts)</td>
<td>29.7 %</td>
<td>0.52 %</td>
</tr>
</tbody>
</table>
4.5 Speed controller bandwidth

Figure 4.16 shows the magnitude of the closed-loop transfer function for the actual speed control system. This data was generated by the ELMO Composer software from the physical winch motor from actuation and sensing of the motor. The red line indicates the -3dB line which the gain crosses at approximately 75 Hz making the bandwidth of the speed control system the same, 75 Hz. Figure 4.17 shows the peak time of the simulated speed control system based on the reference input for the different tether lengths. As expected, as the length of the tether increases and the reference speed increases, the peak time of the speed output increases to approximately 0.077 seconds. These two plots will need to be taken into account later when determining the time-step and operating conditions of the tension controller.

This chapter has described the creation of a Simulink model for the closed loop speed control system for an ELMO motor driver and a Maxon motor. The plant equations of motion were developed mathematically and a Simulink model of the plant was presented. The ELMO current and velocity controllers of the motor driver were then modeled in Simulink. The three Simulink models were combined and tested against data collected from the experimental setup to verify the closed loop speed control system simulation. The closed loop speed control system simulation was successfully verified and found to be an improvement upon the previous simulation. Finally, the bandwidth of the actual closed loop speed control system was presented.
Figure 4.16: Magnitude of the gain of the closed loop speed control system
<table>
<thead>
<tr>
<th>Time (s)</th>
<th>Reference Velocity (counts/sec)</th>
<th>Peak Time for varying reference velocities</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.01</td>
<td>0.5 x 10^5</td>
<td></td>
</tr>
<tr>
<td>0.02</td>
<td>1.0 x 10^5</td>
<td></td>
</tr>
<tr>
<td>0.03</td>
<td>1.5 x 10^5</td>
<td></td>
</tr>
<tr>
<td>0.04</td>
<td>2.0 x 10^5</td>
<td></td>
</tr>
<tr>
<td>0.05</td>
<td>2.5 x 10^5</td>
<td></td>
</tr>
</tbody>
</table>

Figure 4.17: Peak time of the output of the closed loop speed control system simulation
Chapter 5

Tension Control Model

The tension control model is important as it simulates the tension controller, which maintains tension on the tether of the unmanned ground vehicle (UGV), allowing it to carry out its task without having to deal with unforeseen movement of the unmanned aerial vehicle (UAV). If too much line is let out, the tether could drag on the ground and get caught on the environment, or tangled on the UGV itself, effectively ending the current mission. If not enough line is let out, whenever the UAV moves upwards, the UGV will be lifted off the ground and suspended until the UAV descends. Either scenario is detrimental to the success of the UGV, and so, a tension controller is designed to react to the movements of the UAV and the UGV, facilitating the successful use of the UGV. The tension controller outputs a velocity reference to the speed control system, the Simulink model of which was developed in Chapter 4. Figure 5.1 shows the interaction between the tension controller and the speed.
control system; the error, $e_{FT}$, between the tension reference, $F_{T, \text{ref}}$, and the measured tension output of the plant, $F_T$, is the input to the tension controller, with the tension controller output being the reference speed, $\dot{\theta}_{\text{ref}}$, which, in turn, becomes the input to the speed control system. Note that the speed control system includes the ELMO velocity controller, ELMO current controller, and the electro-mechanical plant. This means that the velocity feedback loop and current feedback loop are internal to the speed control system as well as the interaction of the tension with the plant itself (part of the dynamics of the plant).

![Figure 5.1: Tension controller block diagram](image)

There are two basic criteria which the tension controller should be able to meet. The first is that the tension controller should attempt to maintain a constant tension on the line without pulling the UGV off of the ground while minimizing the amount of time there is slack in the line. Secondly, the tension controller should maintain a small enough amount of tension on the tether so that the movement of the UGV is not negatively affected.

The rest of this chapter will cover the design of the tension controller. The first section of this chapter looks into the selection of the time-step of the controller and gives some
operating conditions given the movement of the helicopter and the UGV. The second section will take the time-domain PID controller and convert it into an s-domain (continuous-time) PID controller. The third section takes the continuous-time PID controller and develops a finite difference approximation of the controller in discrete-time. Similar to the third section, the fourth section takes the continuous-time PID controller and develops a Tustin’s method approximation in discrete-time. The fifth section takes the two discrete-time controllers and compares them, finding the finite difference approximation to be the better model for this thesis. The sixth section shows the model of the discrete-time PID controller in Simulink and the implementation of the Simulink model in LabVIEW. The seventh, and last, section details the PID gain selection.
5.1 Time-Step Selection

In order to determine the time step which is appropriate for the controller, the movement of the helicopter and the UGV are studied to find how they affect the tension in the line. Since the tension is proportional to the length of the tether, by Hooke’s law, the rate of change of the tether length due to the movement of the helicopter and the UGV needs to be counteracted by the velocity of the tether due to the winch. This will maintain the tension on the line and allow the UGV to complete its mission. Once the velocity of the tether, due to the helicopter and UGV, is calculated, it will be compared to the maximum possible velocity of the tether due to the winch and the appropriate restrictions on the scenario of operation will be found. Once the velocity is found, the amount of time it takes for the UGV to be lifted off the ground from an equilibrium tension will be calculated and the time step will be derived from this information.

In order to find the tether velocity due to the helicopter and the UGV, the horizontal and vertical velocities of the two vehicles must first be explored. The drift velocity of the helicopter is taken at a hover, the operating condition, on a day with two mph average wind velocity. This data is from the Unmanned System Lab’s Yamaha RMAX helicopter UAV from a flight with no external forces on the UAV except the wind. The position of the UAV was measured, at 50 Hz, and the average drift velocity of the helicopter over one second periods was taken. This was accomplished by taking the position measurements and splitting them into one second segments, then taking the difference in the position of the
helicopter from the start of the period until the end and dividing by the one second time step to get the average velocity over that period. This resulted in 236 samples of both the horizontal drift velocity, the magnitude of the vector addition of the north coordinate and east coordinate, and the vertical drift velocity. The one second average velocities were taken in order to view the drift due of the helicopter due to wind effects and filter out any higher frequency velocity data. Figure 5.2 shows a box plot of the absolute values of the horizontal and vertical velocities. The horizontal velocity has a maximum of 0.42 m/s and a median of 0.077 m/s, while the vertical velocity has a maximum of 0.2 m/s and median of 0.04 m/s. The UGV has a more deterministic velocity as, in his thesis [3], Rose calculates the maximum velocity and maximum scalable slope for the UGV as shown in Table 5.1. With both the velocity of the helicopter, at a hover, and the velocity of the UGV, the velocity of the tether connecting these vehicles can be calculated.

Table 5.1: Robot Performance Data from Rose’s Thesis

<table>
<thead>
<tr>
<th>Performance Data</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Max. Traversable Slope (longitudinal)</td>
<td>50°</td>
</tr>
<tr>
<td>Max. Velocity 50° slope</td>
<td>0.25 m/s</td>
</tr>
<tr>
<td>Max. Velocity 0° slope</td>
<td>0.35 m/s</td>
</tr>
<tr>
<td>Max. Traversable Slope (lateral)</td>
<td>37.5°</td>
</tr>
</tbody>
</table>

Figure 5.3 shows a diagram of how the helicopter and the UGV move in relation to each other and to the velocity of the tether, \( \dot{l} \). This shows a 2-dimensional case which only takes into account the magnitude of the horizontal velocity, as opposed to the vector addition, and assumes the worst case scenario: when all of the velocities are in the same plane. Given this
Figure 5.2: Box plot of helicopter drift velocity during hover
diagram, the velocity of the tether can be calculated with the following equation.

\[ \dot{l} = \dot{y}_h \cos \theta + \dot{x}_h \sin \theta + \dot{x}_u \cos \theta \sin \alpha + \dot{x}_u \sin \theta \cos \alpha \]  

(5.1)

where \( \dot{l} \) is the velocity of the tether, \( \dot{y}_h \) is the vertical velocity of the helicopter, \( \dot{x}_h \) is the horizontal velocity of the helicopter, \( \dot{x}_u \) is the velocity of the UGV, \( \theta \) is the angle of the tether away from the vertical axis, and \( \alpha \) is the angle of the slope the UGV is traveling on.

The maximum angle of \( \theta \) is assumed to be 10°, therefore, using the median and maximum velocities of the helicopter as well as the velocity of the UGV, the velocity of the tether is shown in Table 5.2.

Table 5.2: Line velocity based on helicopter and UGV velocity data

<table>
<thead>
<tr>
<th>( \theta )</th>
<th>( \alpha )</th>
<th>Median Velocity</th>
<th>Max. Velocity</th>
</tr>
</thead>
<tbody>
<tr>
<td>0°</td>
<td>0°</td>
<td>0.0400 m/s</td>
<td>0.2000 m/s</td>
</tr>
<tr>
<td>10°</td>
<td>0°</td>
<td>0.1135 m/s</td>
<td>0.3308 m/s</td>
</tr>
<tr>
<td>0°</td>
<td>37.5°</td>
<td>0.1922 m/s</td>
<td>0.3522 m/s</td>
</tr>
<tr>
<td>10°</td>
<td>37.5°</td>
<td>0.2370 m/s</td>
<td>0.4544 m/s</td>
</tr>
</tbody>
</table>

An important thing to notice here is that the maximum velocity of the tether due to the winch is 0.3835 m/s for the smallest radius of the spool. Therefore, the case where \( \theta = 10° \) and \( \alpha = 37.5° \) is too fast to overcome. This limitation means that, when the UGV is climbing a maximum angled slope, the helicopter must be directly overhead, \( (\theta = 0°) \), otherwise slack will form in the line. Figure 5.4 shows the time it takes for the UGV to go from the reference tension, 3.5 lbs, until it is pulled off the ground, at 10 lbs of tension, given the velocity, from
Figure 5.3: Diagram showing the relative movements of the UGV and UAV
Table 5.2 and length of the tether. The scenario where the winch is not able to match the speed of the tether is shown for reference. Only the situation where the UGV is pulled off the ground is studied as the other situation, letting out too much line, is not as important as the momentary slack in the line will simply be picked up by the movement of the UGV and the actuation of the winch. The figure shows that as the velocity of the tether increase or the length of the line decreases, the time which the UGV takes to be pulled off the ground decreases.

![Graph showing time until lift off based on line velocity at different line lengths](image)

Figure 5.4: Lift off time based on the tether velocities, shown in Table 5.2 and lengths

From Chapter 4, the speed controller has an, approximately, 0.08 second rise time to reach the fastest velocities, 230,000 counts per second. Therefore, understanding that bandwidth
of the speed controller is 75 Hz, and taking into account the speed of the microprocessor, a step time of 0.02 seconds was chosen. This means that the controller works at 50 Hz, sending signals to the speed controller at that frequency. Since the bandwidth of the speed controller is 75 Hz, it should be able to take commands at the lower frequency. This frequency allows for the use of the controller at three different operating lengths. The first is in the static condition, when the helicopter is actually a fixed point and the UGV moving at very small displacements directly below the winch pod (maximum tether velocity of less than 0.04 m/s). In this case, the controller can be operated at a tether length of 5 m or greater, essentially any tether length. The second case is when the winch pod is actually attached to a UAV and the UGV is on level ground directly below the helicopter (maximum tether velocity of 0.2 m/s). In this case, the controller can be operated at 15 m or greater safely, as the amount of time it takes for the UGV to be lifted off the ground is greater than 2.2 seconds. Finally, the third case is when the UGV is attached to the helicopter but the UGV is operating on a slope directly under the helicopter, or the UGV is operating on level ground offset from the helicopter by 10° (maximum tether velocity of 0.3522 m/s). In both of these cases, operating at 25 m or greater will allow for the safe use of the controller.
5.2 Continuous-Time PID Model

The proportional-integral-derivative (PID) controller was chosen for the tension controller. The PID controller is described by the following equation.

\[ u(t) = K_p e(t) + K_i \int_0^t e(\tau)d\tau + K_d \frac{de(t)}{dt} \]  

(5.2)

where \( u(t) \) is the output of the PID controller, \( K_p \) is the proportional gain, \( K_i \) is the integral gain, \( K_d \) is the derivative gain, \( e(t) \) is the time-domain error, and \( t \) is the time variable.

The Laplace transform is exploited to create the continuous-time PID model. The Laplace transform is defined as the following equation.

\[ F(s) = \mathcal{L}[f(t)] = \int_0^\infty e^{-st} f(t)dt \]  

(5.3)

where \( s \) is the Laplace variable, \( F(s) \) is a function in the Laplace domain, \( \mathcal{L} \) is the Laplace operator, \( f(t) \) is a function in the time domain, and \( e \) in this case is the mathematical constant. Therefore, taking the Laplace transform of the time-domain PID model, Equation (5.2) yields the following equation.

\[ \mathcal{L}[u(t)] = \mathcal{L} \left[ K_p e(t) + K_d \frac{de(t)}{dt} + K_i \int_0^t e(\tau)d\tau \right] \]
\[ U(s) = K_p E(s) + K_d (sE(s) - e(0)) + K_i \frac{1}{s} E(s) \]  

(5.4)

where \( U(s) \) is the output of the controller in the Laplace domain and \( E(s) \) is the error, the input to the controller, in the Laplace domain. The initial error in the time domain, \( e(0) \), is assumed to be equal to zero since the reference input is assumed to be changed at time \( t = 0^+ \). Therefore, Equation 5.4 can then be simplified to the following equation.

\[ U(s) = K_p E(s) + K_d sE(s) + K_i \frac{1}{s} E(s) \]  

(5.5)

Equation 5.5 can then be simplified to yield the following equation.

\[ U(s) = K_p \frac{s}{s} E(s) + K_d \frac{s^2}{s} E(s) + K_i \frac{1}{s} E(s) \]

\[ U(s) = \frac{K_d s^2 + K_p s + K_i}{s} E(s) \]  

(5.6)

The ratio of the output, \( U(s) \), to the input, \( E(s) \), is known as the transfer function of the controller and is designated \( H(s) \). The continuous time transfer function for the PID controller can be rewritten from Equation 5.6 as follows.
\[
H(s) = \frac{U(s)}{E(s)} = \frac{K_d s^2 + K_p s + K_i}{s}
\]  

(5.7)

Figure 5.5 shows the Simulink model for the continuous-time PID controller represented by Equation 5.5. This model takes as input Reference Tension and Actual Tension, passing both signals into the summing junction. The output of the summing junction is the error between the two signals, which then enters the Transfer Fcn block with equation 5.7 in it. However, since this is an improper transfer function an extra pole is added as shown in the figure. The output of the Transfer Fcn is the Speed commanded by the PID controller.

Figure 5.5: Simulink model of continuous-time PID controller
5.3 Finite Difference Approximation

In order to implement the PID controller algorithm on a microcontroller, a discrete form of the PID controller must be developed. The finite difference approach is one method of approximating the time-domain PID controller in discrete form. The finite difference approximation takes the three equations; Equations 5.2, 5.4, and 5.5, and approximates them in discrete time using the time step, \( j \), and the sample time \( \Delta T \).

The proportional controller is approximated with the following equation.

\[
K_p e(t) \approx K_p e_j
\]

(5.8)

where \( e_j \) is the error at the \( j \)-th time step. Equation 5.8 simply takes the current error and multiplies it by the proportional gain. This is similar to the time-domain proportional controller except that it only occurs at the discrete time steps. The integral controller is approximated as the following equation.

\[
K_i \int_0^t e(\tau) d\tau \approx K_i \sum_{k=1}^{j} e_k \Delta T
\]

(5.9)

Equation 5.9 takes the rectangle approximation of the integral by summing the product of the discrete time step error values and the sample time. This approximation of the integral
is then multiplied by the integral gain approximating the integral controller. The derivative controller is approximated as the following equation.

\[ K_d \frac{d}{dt} e(t) \approx K_d \frac{e_j - e_{j-1}}{\Delta T} \] (5.10)

Equation 5.10 approximates the derivative as the slope of the line between two consecutive time steps, j and j-1. This approximation of the derivative is multiplied by the derivative gain approximating the derivative controller. Summing Equations 5.8, 5.9 and 5.10, similar to Equation 5.2, yields the following equation.

\[ u_j = K_p e_j + K_i \sum_{k=1}^{j} e_k \Delta T + K_d \frac{e_j - e_{j-1}}{\Delta T} \] (5.11)

Equation 5.11 can be rewritten in terms of the error multiplied by a coefficient, yielding the following equation.

\[ u_j = \left( K_p + K_i \Delta T + \frac{K_d}{\Delta T} \right) e_j + \left( K_i \Delta T - \frac{K_d}{\Delta T} \right) e_{j-1} + K_d \Delta T \sum_{k=1}^{j-2} e_k \] (5.12)

Either Equation 5.11 or Equation 5.12 can be implemented on a microprocessor with relative ease due to their discrete nature.

Figure 5.6 shows the Simulink model of the finite difference approximation of the PID con-
Rather than collapse the model into a discrete-time transfer function, Equation 5.11 is used directly to create the Simulink model. The inputs to the controller are the reference tension, labeled Tension Ref, and actual tension, labeled Tension. These two signals pass into the summing junction where the output of the junction is the error between them. The top third of the Simulink model shows the integral controller from Equation 5.9. The error passes into the block labeled Gain3 which multiplies the signal by the step time, $\Delta T$, shown as $T$ in the Simulink model. The output of Gain3 is passed into an addition block, labeled Add1. The output of Add1 enters the unit delay block, labeled Unit Delay1, which, in turn, outputs the previous value of the signal passed into it. The output of Unit Delay1 is then passed back into the Add1 block. The combination of the addition and unit delay blocks, in recursion, model a Riemann sum of the error multiplied by the step time. The output of the Riemann sum passes into the Gain block, multiplying the signal by $K_i$, and passes into the summing junction. The middle part of Figure 5.6 shows the proportional controller. The error signal enters the Gain1 block, multiplying the input by $K_p$ and is passed into the summing junction. The bottom third of Figure 5.6 shows the derivative controller. The error signal passes into both the Unit Delay block and the Add block. The Unit Delay block outputs the previous time step value which is also passed into the Add block to be subtracted. The output of the Add block enters the Gain4 block where it is divided by the time step, $\Delta T$, again shown as $T$ in the model, and passed through the Gain2 block which multiplies the output of the first by $K_d$. This signal passes into the summing junction which outputs the sum of the three controllers as described by Equation 5.11.
Figure 5.6: Simulink model of finite difference approximation PID controller
5.4 Tustin’s Method Approximation

Another method of creating a discrete-time PID controller is to take the continuous-time equation, given by Equation 5.7, and use the bilinear transform or Tustin’s method to approximate it. This method converts the Laplace domain into the z-domain, another form of the discrete-domain, through the use of the following equation as described by Ogata [24].

\[ s = \frac{1}{T} \ln(z) \approx \left( \frac{2}{T} \right) \frac{z - 1}{z + 1} \]  

(5.13)

where \( z \) is the z-domain variable, and \( T \) is the discrete time step. Substituting Equation 5.13 into Equation 5.5 and simplifying yields the following equation.

\[
U(z) = K_p E(z) + K_d \left( \frac{2}{T} \right) \left( \frac{z - 1}{z + 1} \right) E(z) + K_i \left( \frac{1}{\frac{z - 1}{z + 1}} \right) E(z)
\]

\[
U(z) = K_p E(z) + K_d \left( \frac{2}{T} \right) \left( \frac{z - 1}{z + 1} \right) E(z) + K_i \left( \frac{T}{2} \right) \left( \frac{z + 1}{z - 1} \right) E(z) \quad (5.14)
\]

Multiplying and dividing the individual terms of Equation 5.14 to get a common denominator of \( 2T(z + 1)(z - 1) \) yields the following equation.
\[ U(z) = \frac{2T(z+1)(z-1)}{2T(z+1)(z-1)} K_p E(z) + \frac{2(z-1)}{2(z-1)} K_d \left( \frac{2}{T} \right) \left( \frac{z-1}{z+1} \right) E(z) + \frac{T(z+1)}{T(z+1)} K_i \left( \frac{T}{2} \right) \left( \frac{z+1}{z-1} \right) E(z) \]

Finally, multiplying out the terms containing the \( z \)-domain variable and simplifying yields the following equation.

\[ U(z) = \frac{K_p 2T(z+1)(z-1) E(z) + K_d 4(z-1)^2 E(z) + K_i T^2(z+1)^2 E(z)}{2T(z-1)(z+1)} \]  \hspace{1cm} (5.15)

The ratio of the output, \( U(z) \), to the input, \( E(z) \), is the \( z \)-domain transfer function \( H(z) \) and is found by dividing both sides of Equation 5.16 by \( E(z) \) resulting in the following equation.
\[ H(z) = \frac{U(z)}{E(z)} = \frac{(2TK_p + 4K_d + T^2K_i)z^2 + (2T^2K_i - 8K_d)z + (4K_d + T^2K_i - 2TK_p)}{(2T)z^2 + (2T)} \]  

(5.17)

An interesting thing to note about this design for the controller is that the poles are \( \pm i \).

Figure 5.7 shows the z-domain PID controller model in Simulink. The inputs to the controller are the reference tension and actual tension, both of which enter the summing junction. The output of the summing junction is the error between the two signals and enters the Discrete Transfer Fcn block which contains the z-domain transfer function found in Equation 5.17. The output of the transfer function is the speed commanded to the speed control plant.

![Figure 5.7: Simulink model of z-domain PID controller](image-url)
5.5 Comparison of Methods

In order to compare the different methods used to derive the discrete-time PID controllers, simulations of the two controllers, with the same PID gains, were run. Figure 5.8 shows the Simulink model used to run the simulations. The speed control model is the simulink model developed in Chapter 4. The same model was simulated with the different controllers described in the previous sections in the Tension Controller block while the speed control model was the speed control system model developed in Chapter 4. The gains were chosen to be $K_p = 2000$, $K_i = 100$, and $K_d = 10$ with the only criteria being that the output tension is not saturated. Finally, the simulation was run for each controller at the time step of 0.02 seconds.

Figure 5.8: Overall Tension Control Model

Figure 5.9 shows the tension output of the continuous-time controller and speed control system model. The reference input of 3.5 lbs of tension was found through observation of the UGV to minimize the hindrance on the movement of the UGV. This figure shows the best
that the two other, discrete-time, controllers will be able to accomplish for the given gains as each of the other two are approximations of this controller. Therefore, the two, discrete-time controllers are simulated to compare the output with the continuous-time output, shown in Figure 5.9, for the time step described earlier in this chapter. The two models will then be compared to find which one simulates the tension closest to the continuous-time model output.

Figure 5.9: Output of the simulink model of the continuous-time PID controller and speed control model

Figure 5.10 shows the output (tension) of the finite difference approximation of the PID controller and speed control system. Notice that the output closely follows the continuous time output shown on the same figure.

Figure 5.11 shows the output (tension) of the Tustin’s method approximation of the PID controller and speed control system.
Figure 5.10: Finite difference approximation of PID controller for time step of 0.02 seconds controller. This output has oscillation in it and does not closely follow the continuous time output as the finite difference simulation did. Therefore, with the data collected in this section, the finite difference approximation of the time-domain PID controller was chosen to be implemented.
Figure 5.11: Tustin’s approximation of PID controller for time step of 0.02 seconds
5.6 Implemented PID Controller

The final model of the finite difference approximation of the time-domain PID model is shown in Figure 5.12. This model is an extension of the model shown in Figure 5.10 with the addition of two saturation blocks and a change in the input to the derivative controller. The Saturation block on the right side of the model saturates the output of the PID controller to the maximum speed command that the Solo Whistle will accept. The saturation of the output was done on the MCB2300 board as the Solo Whistle drive will produce an error if the speed command is out of range. The second saturation block labeled Saturation 1, in the integral section (top) of the diagram, has been added as an anti-windup mechanism for the controller. It is necessary because the saturation block on the output, limits the output of the controller while the integral part of the controller continues to wind up causing a significant overshoot if unchecked. Finally, the input to the derivative section of the PID controller (bottom) is altered to be the actual tension, as opposed to the error between the reference tension and the actual tension. This mitigates the effect of a change in reference tension causing a large spike in the derivative controller and adversely affecting the output of the PID controller. All the gains for this controller will be designed in the next section.

The final implementation of the Tension controller is shown in Figure 5.13. This figure shows the LabVIEW Embedded implementation uploaded to the MCB2300 board. Similar to the model shown in Figure 5.12, the inputs to the controller, the tension reference labeled R and the actual tension labeled Tension, are on the left side of the figure. The error between the
actual and reference tensions are input into both the proportional and integral parts of the controller while only the actual tension is input to the derivative controller. Each controller works on its respective input multiplying either the error, derivative of the measurement, or the integral of the error by their respective gains. The outputs of the three controllers are then summed, saturated, and passed out of the controller to be sent to the speed controller.
Figure 5.13: LabVIEW Embedded implementation of PID controller
5.7 PID Gain Selection

The PID gains were chosen by brute forcing the gains and simulating a step input with an initial tension of zero pounds and a reference of 3.5 pounds for each of three spring constants corresponding to three different lengths of tether. The derivative gain is set to zero since the amount of error in the experimental tension signal is enough to significantly disrupt the derivative and therefore only the proportional and integral gains are iterated over. After the three PID gains are found, equations relating the gains to the length of the tether are derived. These final equations are the gains for the final gain scheduled PID controller.

Figures 5.14, 5.15, and 5.16 show the 3D plots of the four metrics; rise time, settling time, percent overshoot, and peak time; for the three spring constants; 5.6 lbs/in, 10.8 lbs/in, and 18.0 lbs/in corresponding to tether lengths of 15.8 meters, 8.2 meters, and 4.9 meters. The goal in selecting the gains is to minimize the four metrics, placing particular emphasis on the rise time of the controller. Therefore, the gains chosen for the three controllers are shown in Table 5.3. Notice that while 4.9 meter tether length has a integral gain of 0, this will never be achieved in reality as the lowest operating condition of the tension controller is 5 meters.

Table 5.3: PI controller gains based on tether length

<table>
<thead>
<tr>
<th>Tether length (m)</th>
<th>Spring Constant (lbs/in)</th>
<th>$K_p$</th>
<th>$K_i$</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.9</td>
<td>18.0</td>
<td>5000</td>
<td>0</td>
</tr>
<tr>
<td>8.2</td>
<td>10.8</td>
<td>7000</td>
<td>2000</td>
</tr>
<tr>
<td>15.8</td>
<td>5.6</td>
<td>12000</td>
<td>3000</td>
</tr>
</tbody>
</table>

Each of the gains from Table 5.3 were picked such that the settling time and percent overshoot
Figure 5.14: Metrics for step input of the system with a tether length of 15.8 meters
Figure 5.15: Metrics for step input of the system with a tether length of 8.2 meters
Figure 5.16: Metrics for step input of the system with a tether length of 4.9 meters
were minimized. These two metrics are the most important in this case, as a large percent
overshoot could pull the UGV off the ground and a large settling time will cause unnecessary
oscillations in the system. In order to ensure that each of the gains were adequate, and not
just a fluke of the dynamics of the system, a simulation was run for each of the three lengths
and shown in Figure 5.17. As seen in this figure, each of the controllers are stable and have
a significantly small settling time and percent overshoot. Note that this could be improved
further in the s-domain or through the use of the derivative controller in future iterations.
Therefore, the proportional and integral gains chosen for this controller produce the following
equations, equating proportional gain, $K_p$, and integral gain, $K_i$, to the length of the tether, $l$.

$$K_p(l) = 645l + 1787 \quad (5.18)$$

$$K_i(l) = 250l - 743 \quad (5.19)$$

These gain schedule equations are simulated in the next chapter over the realizable values
of tether length to confirm stability for all cases.
Figure 5.17: Step inputs of the tension controller using the gains in Table 5.3
5.8 Tension Controller Validation

In order to validate the tension control model, three experimental trials each of the three tether lengths were run and compared against the simulated data. To run the tests in lab, springs were attached in series with the tether to simulate the different lengths. The gains used for each tether length are the ones shown in Table 5.3. Due to the limitations of the winch pod, particularly since the winch pod drops data packets occasionally, due to the communication with the ground control station and not in the loop of the controller itself, some of the data runs were offset by a multiple of the step time. Figure 5.18a shows the simulated versus experimental data for the 4.9 meter case. As can be seen in this figure the data closely follows the simulated output and maintains 3.5 lbs of tension. The 8.2 meter case, shown in Figure 5.18b, shows a similar trend to the 4.9 meter case, with the experimental data closely following the simulated data. In Figure 5.18c the 15.8 meter case, the simulation output again follows closely with the experimental data. Therefore, this data validates the tension control model as a good model for the actual tension controller.

This chapter has completed the design and implementation of a PI tension controller. A comparison between a finite difference discrete-time controller and a Tustin’s method discrete-time controller was accomplished, resulting in the choice of the finite difference controller. The final simulation and implementation of the controller was presented along with the selection of the time step and PI gains given the length of the tether. The next chapter will present both simulation and lab test results for the controller designed.
Figure 5.18: Simulated versus experimental data of the tension controller for varying tether lengths
Chapter 6

Validation of the Design

This chapter displays the test results, both simulated and experimental, for the tension controller described in Chapter 5. While physical testing was done to ensure that the tension controller performed adequately, simulations were also run for situations which could not be physically tested. Some simulations were run using data from the USL’s Yamaha RMAX helicopter UAV to determine the effectiveness of the controller in these conditions.

This chapter is split into three distinct sections. The first section shows simulation results using the tension controller for the different operating conditions. The gain schedule equations for the proportional and integral gains were tested over several tether lengths to validate these equations. Different combinations of UAV and UGV velocities were simulated to determine the amount of time that the UGV was lifted of the ground. Finally, the first section ends with three simulations, at three different tether lengths, using actual position
and velocity data from a UAV at hover.

The second section of this chapter shows the physical test results. This section begins by showing step input tests with springs which simulate different tether lengths. The step inputs were shown to be stable for five different simulated tether lengths. The section then describes a testing rig capable of raising the winch pod to approximately 10 meters in height. Finally, the tension controller was tested from two different heights at two different speeds to determine how well the tension controller works.

The third section shows the results of a robustness analysis for the tension controller. There are two variables which cannot be measured. They are the radius of the spool, which changes as the tether is wrapped around the spool, and the length of the tether. The effects of varying the radius of the spool at different tether lengths are demonstrated. Similarly, the effects of varying the length of the tether and assuming a fixed tether length are demonstrated. Robust ranges of the radius and the tether length are found.

6.1 Simulation Test Results

In order to verify the gain schedule equations, simulations were run for every 10 meters of tether length between 10 meters and 120 meters. Since, the gain schedule equations used tether lengths with a maximum length of 15.8 meters, it was necessary to check longer tether lengths for stability with gains from the gain schedule equations. Figure 6.1 shows
the tension output of the gain scheduled tension controller at the tether lengths indicated. In all cases the tension controller is stable, implying that as the length of the tether continues to increase, the tension controller will continue to work. A couple of interesting things occur as well. The first is that the slope of the transient decreases as the length of the tether increases. This causes the rise and settling times of the system to increase. Secondly, the percent overshoot increases with the length of the tether. However, as seen from Figure 6.1 the system is not negatively affected by the increases in the rise time, settling time, and percent overshoot. Therefore, the tension controller should work over the range simulated, up to a 120 meter tether length.

The next set of simulations were designed to determine the amount of time the UGV would be pulled off the ground given a random set of UAV velocities and a fixed UGV velocity. In order to accomplish this the two-dimensional model described in Figure 5.3 was used where the UAV could move vertically, $\dot{y}_h$, and horizontally, $\dot{x}_h$ while the UGV could move with a fixed velocity, $\dot{x}_u$ at an angle, $\alpha$ from the horizontal plane. The UGV is also assumed to be at a fixed angle $\theta$ away from the vertical projection of the center of the helicopter. For each ten second simulation, the helicopter motion was determined by ten velocities, of one second each, drawn from uniform random numbers of the helicopter drift velocities found in Chapter 5. That is, ten one-second helicopter velocities, in two dimensions, were chosen with the following parameters: $-0.42m/s \leq \dot{x}_h \leq 0.42m/s$ and $-0.20m/s \leq \dot{y}_h \leq 0.20m/s$. The UGV velocity was chosen such that if $\alpha = 0^\circ$ then $\dot{x}_u = 0.35m/s$ and if $\alpha = 37.5^\circ$ then $\dot{x}_u = 0.25m/s$. Then 100 simulations were run for each combination of the following
Figure 6.1: Step input simulations for varying tether lengths
variables, $\theta = 0^\circ, 5^\circ, 10^\circ$, $\alpha = 0^\circ, 37.5^\circ$, and $l = 5m, 15m, 25m$. Finally, the amount of time the UGV was pulled off the ground for each simulation was calculated, based on the tension of the line, as shown by the box plots in Figure 6.2. Figure 6.2a shows the 5 meter tether length case. As seen in this figure, whenever the UGV is climbing a slope, the UGV is very likely to be pulled off the ground. In Chapter 5, the operating conditions for a 5 meter tether length were described as being only when the UAV is stationary and the UGV is moving along the horizontal axis. This simulation validates the operating conditions, showing that even at small angles of $\theta$ the UGV has a chance of being pulled off the ground. Figure 6.2b shows the 15 meter tether length case. Unlike the 5 meter case, in this case the UGV is never pulled off the ground when the UGV is on flat terrain. Again this validates the operating conditions described in Chapter 5 for the 15 meter case. Finally, Figure 6.2c shows the 25 meter tether length case. In this case the UGV was never pulled off the ground for any period of time, at any slope and any angle out from the helicopter. This shows that the tension controller can be run at a 25 meter tether length, or longer, for any scenario.

The last simulation of the tension controller represents a UGV mission from three different tether lengths using actual UAV hover velocities. For the sake of the mission, the UGV was rendered stationary (to perform a delicate task) while the UAV was moving around at the velocities shown in Figure 6.3a. These velocities are taken directly from the wePilot autopilot system on the USL Yamaha RMAX helicopter during a hover at 50 Hz. Therefore, the maximum frequency in the data is 25 Hz, which is good for the tension controller since it has a sample frequency of 50 Hz. The horizontal velocities (north and east) stay roughly
Figure 6.2: Box plots of 100 samples of UGV lift off time for varying $\theta$, $\alpha$, and tether lengths
between ±0.5 m/s while the down velocity stays roughly between ±0.2 m/s. The UAV velocities can then be correlated to tether velocities based on the location of the UAV to the UGV. For the 25 meter tether length case, the tether velocities are shown in Figure 6.3b. Figure 6.4 shows how the tension controller reacts to the movement of the helicopter for the three tether lengths. As expected, the 5 meter tether length case, shown in Figure 6.4a, fails to keep the UGV on the ground, as indicated by the several points where the tension is maxed out at 10 lbs, and allows slack to develop in the line, as shown by the points where the tension is 0 lbs. The 15 meter case does significantly better, never pulling the UGV off of the ground or allowing slack to develop in the tether. Again this is expected as the 15 meter tether length operating conditions fall under this case. Finally, the 25 meter tether length case improves upon the 15 meter case, decreasing the range of tension which the tether experiences.
(a) North, east, and down UAV velocities with 50 Hz sample rate

(b) Simulated tether velocity given actual velocities at 25 meter tether length

Figure 6.3: Simulated tension and actual UAV velocities for simulated mission
Figure 6.4: Tension controller simulation for actual UAV velocities

(a) 5 meters tether length

(b) 15 meter tether length

(c) 25 meter tether length
6.2 Physical Test Results

Along with the simulation results presented in the previous section, two different physical tests were performed. The first was a step input using a spring to simulate different lengths of line. Figure 6.5 shows the tests at each of five different simulated tether lengths. For each of the tether lengths, a spring of known stiffness was put in series with the tether, to simulate the different tether lengths without having to deploy the full length of the tether. The gain scheduled tension controller was then run in each case, taking the tension from the minimum tension to the reference tension, approximately 3.5 lbs. For each case, the controller successfully maintained a tension on the tether quickly and stably. All of the plots show settling times of less than 0.2 seconds with very minimal overshoot despite the noise in the tension signal.

The second physical test was performed with the winch pod securely attached to the side of a scissors lift. The lift was then raised to two different heights and the weight was moved back and forth at a steady velocity to simulate the movement of the UGV. Figure 6.6 shows the three tension controller trials at a height of 9.6 meters. The weight was moved back and forth with a velocity of 0.3 m/s across positions of $\pm 1.2$ meters. This equates to a $\theta$ value in the range of $-8.5^\circ \leq \theta \leq 8.5^\circ$. At this height, in these conditions, the tension controller successfully maintains the tension, with the weight never being pulled off the ground or slack forming in the tether. Figure 6.7 shows the tension controller working in the exact same conditions as Figure 6.6 except at a height of 5.8 meters. This means that the $\theta$ value is
Figure 6.5: Step inputs for spring tests simulating various tether lengths
in the range of $-13.5^\circ \leq \theta \leq 13.5^\circ$, a significantly greater angle than the previous height. This is outside of the $\theta = 10^\circ$ operating condition and, as seen from Figure 6.7 while the weight is never pulled off of the ground, significant slack forms in the line. Notice that there is an, approximately, 0.7 lb offset on the output of the load cell. This offset, due to the signal conditioning of the load cell, is always present, but only visible when slack forms in the line. The final physical tension controller test was performed with a velocity of 0.15 m/s across positions of ±0.3 meters. Figure 6.8 shows the output of the tension controller under these conditions. In this case the controller successfully maintains tension on the tether, never pulling the weights off the ground or allowing slack to form in the line. For all three of these tests the controller was stable and, in the first and third test, successfully maintained tension on the line.
Figure 6.6: Tension test with 9.6 m tether length at 0.3 m/s and $\theta < 8.5^\circ$
Figure 6.7: Tension test with 5.8 m tether length at 0.3 m/s and $\theta < 13.5^\circ$
Tether tension test with 5.8 m tether and UGV velocity of 0.15 m/s

(a) Trial 1  
(b) Trial 2  
(c) Trial 3

Figure 6.8: Tension test with 5.8 m tether length at 0.15 m/s and \( \theta < 3.5^\circ \)
6.3 Robustness Analysis

The final set of test results are a robustness analysis of the tension controller in simulation. Up until this point, the radius of the spool was assumed to be the minimum spool diameter and the length of the tether was assumed to be perfectly known. However, since the spool has no guides for the tether, the tether can spool randomly. Therefore, the radius of the spool is unknown, as the tether can spool onto open space or loops of the tether. Since the radius of the spool is unknown, this means that the velocity of the tether is unknown, and, therefore, the length of the tether is unknown. So, a robustness analysis of the radius of the spool and the length of the tether was performed.

The robustness analysis for the radius of the spool is performed for the three operating conditions, tether lengths of 5 meters, 15 meters, and 25 meters. For each operating condition, the radius of the spool is iterated from 0.55 inches to 1.5 inches, the maximum possible radius of the spool, in 0.01 inch increments. Everything else is kept the same, including the gains which assume the minimum spool radius. Figure 6.9 shows five outputs of the tension controller for different spool radii at a tether length of 5 meters. As the radius of the spool increases, the overshoot increases and, therefore, the settling time increases. The linear velocity of the tether is proportional to the angular velocity of the winch by $\dot{l} = r\dot{\theta}$, where $\dot{l}$ is the linear velocity of the tether, $r$ is the radius of the spool, and $\dot{\theta}$ is the angular velocity of the winch. Since this is the case, as the radius of the spool increases, a proportional increase in the linear velocity of the tether, given the same angular velocity of
the winch, can be expected. This increased linear velocity should overestimate the angular velocity needed to reach the reference tension position, causing a decrease in the rise time and an increase in overshoot and settling time. Indeed, Figure 6.10 shows that while the rise time and peak time decrease, the overshoot and settling time metrics increase. In fact, the overshoot increases almost linearly with the radius of the spool. Figures 6.11 and 6.13 show that the 15 meter and 25 meter tether length cases are very similar to the 5 meter case, with Figures 6.12 and 6.14 showing that the metrics follow course. From the metrics plots we can say that the tension controller is robust to changes in the radius of the spool up to 0.65 inches in radius.

The second robustness analysis looks at the change in the length of the tether assuming a fixed length. Again, the three operating conditions are inspected, assuming tether lengths of 5 meters, 15 meters, and 25 meters. Then, with the gains found from the assumed tether lengths the actual tether lengths were varied from $l - 4.9$ meters to $l + 5$ meters in increment of 0.1 meters. The 5 meter operating condition showed the most extreme variations, as the minimum tether length was 0.1 meters. Figure 6.15 shows the effect of the variation of the tether length on the output of the tension controller assuming a tether length of 5 meters. The first case shows the 0.1 meter case which saturates the tension almost all of the time, as would be expected from such an extremely short tether length. However, as the tether length increase closer to the assumed length, the controller stabilizes and the overshoot decreases. As the tether length increases from the assumed tether length to the maximum, the output becomes slower but stays stable throughout. Figure 6.16 shows the settling time, rise time,
Figure 6.9: Tension controller simulations of various spool radii at 5 meters of tether
Figure 6.10: Metrics for simulations with varying radii at 5 meters of tether
Figure 6.11: Tension controller simulations of various spool radii at 15 meters of tether
(a) Settling time, rise time, and peak time

(b) Percent overshoot

Figure 6.12: Metrics for simulations with varying radii at 15 meters of tether
Figure 6.13: Tension controller simulations of various spool radii at 25 meters of tether
Figure 6.14: Metrics for simulations with varying radii at 25 meters of tether
peak time, and percent overshoot based on the length of tether for the 5 meter case. Figures 6.17 and 6.19 show the output for the 15 meter and 25 meter assumed tether cases. Each of these cases shows significantly less overshoot than the 5 meter case and are generally stable with small overshoot. Figures 6.18 and 6.19 show the metrics for the two cases. From these three cases we see that as the length of the tether approaches the assumed tether length, the settling time decreases. However, the rise time and peak times always increase with the length of the tether, while the percent overshoot decreases until the assumed tether length is reached and then maintains a very small overshoot. Therefore, this simulated system is robust to unknown changes in the length of the tether of up to ±5 meters for every assumed tether length above 15 meters.

From the results of the simulation, physical, and robustness tests, this tension controller has been shown to perform the task of keeping a UGV attached to a UAV without hindering the mission of either vehicle. The simulation and physical tests have shown the tension controller working in a variety of situations, as well as the limitations of the controller. The robustness tests have shown how the controller reacts to two of the, currently, unmeasurable variables in the system, showing that, while not ideal, the tension controller would continue to work regardless.
Figure 6.15: Tension controller simulations of varying tether lengths assuming a tether length of 5 m
Figure 6.16: Metrics for simulations with varying tether lengths assuming a tether length of 5 m
Figure 6.17: Tension controller simulations of varying tether lengths assuming a tether length of 15 m.

(a) 10.1 m tether length
(b) 12.5 m tether length
(c) 15.0 m tether length
(d) 17.5 m tether length
(e) 20.0 m tether length
Figure 6.18: Metrics for simulations with varying tether lengths assuming a tether length of 15 m
Tension controller simulation with tether length of 20.1 m, assuming tether length 25 m.

(a) 20.1 m tether length

Tension controller simulation with tether length of 22.5 m, assuming tether length 25 m.

(b) 22.5 m tether length

Tension controller simulation with tether length of 25.0 m, assuming tether length 25 m.

(c) 25.0 m tether length

Tension controller simulation with tether length of 27.5 m, assuming tether length 25 m.

(d) 27.5 m tether length

Tension controller simulation with tether length of 30.0 m, assuming tether length 25 m.

(e) 30.0 m tether length

Figure 6.19: Tension control simulations of varying tether lengths assuming a tether length of 25 m
Figure 6.20: Metrics for simulations with varying tether lengths assuming a tether length of 25 m
Chapter 7

Conclusion and Future Work

This thesis presented a gain scheduled proportional-integral (PI) tension controller for the deployment of an unmanned ground vehicle from an autonomous, unmanned helicopter. The final tension controller was found to work in all scenarios at a height of 25 meters. Lower heights could be attained, if needed, with limitations on the movement of the UGV. The thesis first presents the winch pod hardware which was outlined to show what the tension controller would be operating on and what components would need to be modeled.

The speed controller of the ELMO motor controller and Maxon motor was modeled. First, the plant of the speed controller; the Maxon DC motor, gearhead, spool, and tether; was modeled. The equations of motion for the plant were developed and a Simulink model of the system was presented. The inner-loop controller, the current controller modeled next. This controller was modeled in Simulink from the information provided by the ELMO Motion
Control company. The speed controller was modeled last, again from the information provided by the ELMO Motion Control company. The speed control model was then validated with data collected from the winch pod.

The tension controller was developed using the proportional-integral-derivative (PID) controller architecture. Two discrete-time approximations were developed and compared. The finite difference approximation was chosen, a Simulink model was developed, and LabVIEW implementation was programmed. The step time was chosen based on the velocity of the helicopter, and the PI gain equations were developed with respect to the length of the tether. The tension controller model was then validated based on data collected from the winch pod.

Finally, tests were run to determine the effectiveness of the tension controller in several different conditions. First, simulations were run to validate the effectiveness of the gain equations over a large range of tether lengths. Simulations were then run using random and actual velocities of the UAV and the UGV to determine the effectiveness of the tension controller in an actual mission scenario. Secondly, physical tests were run on the winch pod. Spring tests were run to see how the winch pod reacted to different tether lengths. A tension control test was run from top of a scissors lift, showing that the tension controller would effectively handle the movement of the UGV. Lastly, robustness simulation tests were run to see how the system would react to a change in the radius of the spool and a change in the length of the tether.

There are three main areas of future work for this thesis. The first involves expanding the PI
controller into a PID controller. The second is a way to improve the modeling of the speed controller. Finally, the third is the development of a new, and robust, tension controller.

The first area of future work centers on the filtering of the load cell signal to eliminate the noise and allow for the use of the derivative term of the PID controller. A series of tests should be conducted to determine the best trade off between the smoothness of the load cell signal and the added delay of using a low-pass filter.

The second area of future work begins with approximating the speed controller and the current controller of the ELMO Solo Whistle with a linear system by means of system identification. Once a linearized system is found, the plant can be linearized in the range of the possible tension values and a complete linear approximation to the system can be found. A full state feedback controller can then be designed for this approximation and optimal control theory can then be used to find the gain values.

The final area of future work is to look into the design of a robust controller to the radius of the spool and the length of the tether. A large change in the radius of the spool particularly poses a problem for the controller and, therefore, a robust controller would be able to react, either passively or actively, to this change without measuring it.
Chapter 8

Bibliography


## Appendix A

### Speed Controller Model Parameters

Table A.1: Gain selector values

<table>
<thead>
<tr>
<th>Velocity (Counts/second)</th>
<th>$K_p$</th>
<th>$K_i$</th>
</tr>
</thead>
<tbody>
<tr>
<td>173</td>
<td>50.827</td>
<td>6970.109</td>
</tr>
<tr>
<td>179</td>
<td>51.502</td>
<td>7145.006</td>
</tr>
<tr>
<td>184</td>
<td>52.192</td>
<td>7323.705</td>
</tr>
<tr>
<td>190</td>
<td>52.896</td>
<td>7506.290</td>
</tr>
<tr>
<td>196</td>
<td>53.565</td>
<td>7707.187</td>
</tr>
<tr>
<td>201</td>
<td>54.245</td>
<td>7911.578</td>
</tr>
<tr>
<td>207</td>
<td>54.938</td>
<td>8119.526</td>
</tr>
<tr>
<td>212</td>
<td>55.642</td>
<td>8331.091</td>
</tr>
<tr>
<td>218</td>
<td>56.960</td>
<td>8522.041</td>
</tr>
</tbody>
</table>
Table A.1: Gain selector values

<table>
<thead>
<tr>
<th>Velocity (Counts/second)</th>
<th>$K_p$</th>
<th>$K_i$</th>
</tr>
</thead>
<tbody>
<tr>
<td>224</td>
<td>58.303</td>
<td>8716.562</td>
</tr>
<tr>
<td>229</td>
<td>59.671</td>
<td>8914.720</td>
</tr>
<tr>
<td>235</td>
<td>61.065</td>
<td>9116.584</td>
</tr>
<tr>
<td>240</td>
<td>61.680</td>
<td>9325.655</td>
</tr>
<tr>
<td>246</td>
<td>62.309</td>
<td>9538.952</td>
</tr>
<tr>
<td>252</td>
<td>62.950</td>
<td>9756.560</td>
</tr>
<tr>
<td>257</td>
<td>63.604</td>
<td>9978.566</td>
</tr>
<tr>
<td>263</td>
<td>63.658</td>
<td>10222.700</td>
</tr>
<tr>
<td>268</td>
<td>63.714</td>
<td>10472.190</td>
</tr>
<tr>
<td>274</td>
<td>63.771</td>
<td>10727.180</td>
</tr>
<tr>
<td>280</td>
<td>63.829</td>
<td>10987.770</td>
</tr>
<tr>
<td>285</td>
<td>64.712</td>
<td>11361.950</td>
</tr>
<tr>
<td>291</td>
<td>65.617</td>
<td>11745.160</td>
</tr>
<tr>
<td>296</td>
<td>66.544</td>
<td>12137.610</td>
</tr>
<tr>
<td>302</td>
<td>67.493</td>
<td>12539.520</td>
</tr>
<tr>
<td>308</td>
<td>69.867</td>
<td>12874.950</td>
</tr>
<tr>
<td>313</td>
<td>72.305</td>
<td>13219.340</td>
</tr>
<tr>
<td>319</td>
<td>74.808</td>
<td>13572.920</td>
</tr>
<tr>
<td>324</td>
<td>77.378</td>
<td>13935.940</td>
</tr>
</tbody>
</table>
### Table A.1: Gain selector values

<table>
<thead>
<tr>
<th>Velocity (Counts/second)</th>
<th>$K_p$</th>
<th>$K_i$</th>
</tr>
</thead>
<tbody>
<tr>
<td>330</td>
<td>77.471</td>
<td>14471.570</td>
</tr>
<tr>
<td>336</td>
<td>77.566</td>
<td>15023.200</td>
</tr>
<tr>
<td>341</td>
<td>77.665</td>
<td>15591.320</td>
</tr>
<tr>
<td>347</td>
<td>77.767</td>
<td>16176.420</td>
</tr>
<tr>
<td>363</td>
<td>79.448</td>
<td>16557.869</td>
</tr>
<tr>
<td>378</td>
<td>81.186</td>
<td>16952.260</td>
</tr>
<tr>
<td>394</td>
<td>82.984</td>
<td>17360.039</td>
</tr>
<tr>
<td>410</td>
<td>84.842</td>
<td>17781.660</td>
</tr>
<tr>
<td>426</td>
<td>85.770</td>
<td>18563.301</td>
</tr>
<tr>
<td>441</td>
<td>86.733</td>
<td>19375.660</td>
</tr>
<tr>
<td>457</td>
<td>87.735</td>
<td>20219.930</td>
</tr>
<tr>
<td>473</td>
<td>88.776</td>
<td>21097.369</td>
</tr>
<tr>
<td>489</td>
<td>91.867</td>
<td>21958.670</td>
</tr>
<tr>
<td>505</td>
<td>95.103</td>
<td>22860.150</td>
</tr>
<tr>
<td>520</td>
<td>98.489</td>
<td>23803.660</td>
</tr>
<tr>
<td>555</td>
<td>102.033</td>
<td>24791.180</td>
</tr>
<tr>
<td>590</td>
<td>102.531</td>
<td>25571.750</td>
</tr>
<tr>
<td>625</td>
<td>103.057</td>
<td>26397.109</td>
</tr>
<tr>
<td>659</td>
<td>103.613</td>
<td>27269.820</td>
</tr>
</tbody>
</table>
Table A.1: Gain selector values

<table>
<thead>
<tr>
<th>Velocity (Counts/second)</th>
<th>$K_p$</th>
<th>$K_i$</th>
</tr>
</thead>
<tbody>
<tr>
<td>694</td>
<td>104.201</td>
<td>28192.590</td>
</tr>
<tr>
<td>752</td>
<td>107.107</td>
<td>29243.740</td>
</tr>
<tr>
<td>810</td>
<td>110.230</td>
<td>30373.279</td>
</tr>
<tr>
<td>868</td>
<td>113.586</td>
<td>31587.039</td>
</tr>
<tr>
<td>954</td>
<td>117.192</td>
<td>32891.320</td>
</tr>
<tr>
<td>1041</td>
<td>118.873</td>
<td>34255.281</td>
</tr>
<tr>
<td>1128</td>
<td>120.734</td>
<td>35764.762</td>
</tr>
<tr>
<td>1215</td>
<td>122.793</td>
<td>37435.270</td>
</tr>
<tr>
<td>1388</td>
<td>125.072</td>
<td>39283.988</td>
</tr>
<tr>
<td>1562</td>
<td>126.854</td>
<td>40170.461</td>
</tr>
<tr>
<td>1909</td>
<td>128.975</td>
<td>41224.672</td>
</tr>
<tr>
<td>2256</td>
<td>131.496</td>
<td>42478.328</td>
</tr>
<tr>
<td>2777</td>
<td>134.495</td>
<td>43969.199</td>
</tr>
<tr>
<td>3645</td>
<td>134.902</td>
<td>43969.199</td>
</tr>
<tr>
<td>5555</td>
<td>135.415</td>
<td>43969.199</td>
</tr>
<tr>
<td>10763</td>
<td>136.061</td>
<td>43969.199</td>
</tr>
</tbody>
</table>
Table A.2: Values used for the constants of the speed control model

<table>
<thead>
<tr>
<th>Constant</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>R</td>
<td>0.317Ω</td>
</tr>
<tr>
<td>L</td>
<td>0.0823 × 10^{-3} H</td>
</tr>
<tr>
<td>K</td>
<td>30.2 × 10^{-3} Nm</td>
</tr>
<tr>
<td>K_b</td>
<td>30.2 × 10^{-3} \frac{A}{rad/s}</td>
</tr>
<tr>
<td>n_1</td>
<td>1</td>
</tr>
<tr>
<td>n_2</td>
<td>26</td>
</tr>
<tr>
<td>J_1</td>
<td>1.73 × 10^{-5} kgm^2</td>
</tr>
<tr>
<td>J_2</td>
<td>9.1 × 10^{-7} kgm^2</td>
</tr>
<tr>
<td>J_3</td>
<td>2.73 × 10^{-5} kgm^2</td>
</tr>
<tr>
<td>b_1</td>
<td>0</td>
</tr>
<tr>
<td>b_2</td>
<td>0</td>
</tr>
<tr>
<td>K_s</td>
<td>200 \frac{N}{m}</td>
</tr>
<tr>
<td>r_s</td>
<td>0.014 m</td>
</tr>
<tr>
<td>K_{Speed}</td>
<td>\frac{MC_VALUE_BITS}{MC_VALUE_BITS} 0.00345</td>
</tr>
<tr>
<td>MC</td>
<td>30A</td>
</tr>
<tr>
<td>MC_VALUE_BITS</td>
<td>14000</td>
</tr>
<tr>
<td>AC</td>
<td>\frac{counts}{s^2} 2575854</td>
</tr>
<tr>
<td>DC</td>
<td>\frac{counts}{s^2} 2575854</td>
</tr>
</tbody>
</table>