A QUANTITATIVE APPROACH FOR TUNING A MOUNTAIN BIKE SUSPENSION

A Thesis

presented to

the Faculty of California Polytechnic State University,

San Luis Obispo

In Partial Fulfillment

of the Requirements for the Degree

Master of Science in Mechanical Engineering

by

Steven R. Waal

November 2020
COMMITTEE MEMBERSHIP

TITLE: A Quantitative Approach for Tuning a Mountain Bike Suspension

AUTHOR: Steven R. Waal

DATE SUBMITTED: November 2020

COMMITTEE CHAIR: Joseph D. Mello, Ph.D.
Professor of Mechanical Engineering

COMMITTEE MEMBER: William R. Murray, Ph.D.
Professor of Mechanical Engineering

COMMITTEE MEMBER: John R. Ridgely, Ph.D.
Professor of Mechanical Engineering
A method for tuning the spring rate and damping rate of a mountain bike suspension based on a data-driven procedure is presented. The design and development of a custom data acquisition system, known as the “MTB DAQ,” capable of measuring acceleration data at the front and rear axles of a bike are discussed. These data are input into a model that is used to calculate the vertical acceleration and pitching angular acceleration response of the bike and rider. All geometric and dynamic properties of the bike and rider system are measured and built into the model. The model is tested and validated using image processing techniques. A genetic algorithm is implemented with the model and used to calculate the best spring rate and damping rate of the mountain bike suspension such that the vertical and pitching accelerations of the bike and rider are minimized for a given trail. Testing is done on a variety of different courses and the performance of the bike when tuned to the results of the genetic algorithm is discussed. While more fine tuning of the model is possible, the results show that the genetic algorithm and model accurately predict the best suspension settings for each course necessary to minimize the vertical and pitching accelerations of the bike and rider.
ACKNOWLEDGMENTS

Even though this thesis was completed at the end of my graduate studies at Cal Poly, the skills and knowledge required to complete it came from experiences from my entire life. As such, I have more people to thank than can properly be mentioned here. Nevertheless, I have attempted to include as many as possible and hope that these few words will be able to give a glimpse into just how much these people have influenced who I have become.

First and foremost, I would like to thank my committee chair, Joseph D. Mello, for presenting me with the idea and supporting me through the longest and most technically challenging project that I have worked on so far. He never ceased to provide excitement and passion, and has a special gift for generating the same excitement from all those around him. I especially want to thank him for the few “testing” sessions we did riding our bikes on a Saturday afternoon in Poly Canyon. I would also like to extend a special thanks to John R. Ridgely, for the amazing technical support and feedback through developing the MTB DAQ hardware and software. Professor Ridgely is a true “Renaissance man” in the world of mechanical engineering, for there is not one question, no matter what subject, that I have asked him that he has not been able to answer! He was always there to help me, answering emails and fielding questions any time of the day. I would further like to thank William R. Murray, for his contribution to my thesis committee, and for giving me most of the theoretical background needed for the dynamic models. Professor Murray has a gift for teaching, being able to contrive eloquent lectures on the spot on any subject.

Beyond my thesis committee, I would like to extend my sincerest thanks to:

- Charlie Refvem, for the amazing patience and number of hours spent helping me troubleshoot problems with the first version of the MTB DAQ.
• John Larson, for planting the seed to pursue a master’s degree in the first place and teaching me what it means to handle the responsibility that comes with being an engineer.

• Hans Mayer, for pushing me to perform my best and setting a new standard for quality and performance.

• John Gerrity, for teaching me the true meaning of “learn by doing.”

• Marshall L. McFarland, for always supporting me and counseling me in my professional development.

• Mike Huntsman, for cultivating my love of engineering from an early age and pushing my creativity to go further.

• Rodger Johnson, for introducing me to the amazing and beautiful world of physics.

• Stan Hitomi, for instilling the values of leadership and humility that are necessary for any large project.

• Andrew Guenther, for uploading the Cal Poly \LaTeX template [1].

• Jesse Barnett, for staying up late on multiple occasions to help me troubleshoot my model when things were not working out.

• Samuel Lee, for being an amazing role model in my life by teaching me how to work hard and get things done.

• Carina Bender, for the constant words of encouragement and support for this thesis and in all areas of life.

• My mom and dad, for supporting me through the adventure that is called life no matter where it has taken me and providing the best example of how to live.
The work presented here is dedicated to my family and my puppy dog, Heidi Theta Waal, who taught me more about life and happiness than any other living creature.
# TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>LIST OF TABLES</th>
<th>xii</th>
</tr>
</thead>
<tbody>
<tr>
<td>LIST OF FIGURES</td>
<td>xiv</td>
</tr>
</tbody>
</table>

## CHAPTER

1. **Introduction**
   - 1.1 A Brief History of the Bicycle .......................... 1
   - 1.2 Modern Mountain Bike Shocks .............................. 2
   - 1.3 Purpose .................................................. 7

2. **Literature Review** ........................................... 9
   - 2.1 The Role of Suspension ................................. 9
   - 2.2 Suspension Optimization ................................ 10
   - 2.3 Road Spectral Analysis .................................. 14
   - 2.4 ShockWiz™ ................................................ 15
   - 2.5 Mountain Bike Ride Comfort ................................ 15
   - 2.6 Measuring Accelerations of a Bike Frame ............... 16

3. **Project Outline** ............................................ 19

4. **Data Acquisition System** ................................. 21
   - 4.1 Design Requirements ..................................... 21
   - 4.2 Conceptual Design ......................................... 23
   - 4.3 Component Selection ...................................... 24
   - 4.4 Hardware Design .......................................... 27
     - 4.4.1 Main Unit Design .................................... 28
     - 4.4.2 Front and Rear Accelerometer Unit Design ......... 32
<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.5</td>
<td>Code Design</td>
<td>33</td>
</tr>
<tr>
<td>4.6</td>
<td>MTB DAQ Prototype</td>
<td>35</td>
</tr>
<tr>
<td>4.6.1</td>
<td>MTB DAQ Technical Specifications</td>
<td>35</td>
</tr>
<tr>
<td>4.6.2</td>
<td>Prototype Testing</td>
<td>37</td>
</tr>
<tr>
<td>4.7</td>
<td>User Manual</td>
<td>40</td>
</tr>
<tr>
<td>4.7.1</td>
<td>Flashing the Board</td>
<td>41</td>
</tr>
<tr>
<td>4.7.2</td>
<td>MTB DAQ Operation</td>
<td>42</td>
</tr>
<tr>
<td>4.8</td>
<td>GitHub Repository</td>
<td>44</td>
</tr>
<tr>
<td>5</td>
<td>Rider and Testing Bike Technical Specifications</td>
<td>45</td>
</tr>
<tr>
<td>5.1</td>
<td>Rider Physical Characteristics</td>
<td>45</td>
</tr>
<tr>
<td>5.2</td>
<td>Geometry and Components</td>
<td>46</td>
</tr>
<tr>
<td>5.3</td>
<td>Dynamic Properties</td>
<td>46</td>
</tr>
<tr>
<td>5.3.1</td>
<td>Center of Gravity</td>
<td>47</td>
</tr>
<tr>
<td>5.3.2</td>
<td>Pitching Moment of Inertia</td>
<td>50</td>
</tr>
<tr>
<td>6</td>
<td>Suspension Characterization</td>
<td>54</td>
</tr>
<tr>
<td>6.1</td>
<td>Wheel to Shock Adjustments</td>
<td>56</td>
</tr>
<tr>
<td>6.1.1</td>
<td>Front Wheel to Fork</td>
<td>57</td>
</tr>
<tr>
<td>6.1.2</td>
<td>Rear Wheel to Shock</td>
<td>58</td>
</tr>
<tr>
<td>6.1.3</td>
<td>Summary</td>
<td>65</td>
</tr>
<tr>
<td>6.2</td>
<td>Air Spring Characterization</td>
<td>65</td>
</tr>
<tr>
<td>6.2.1</td>
<td>Air Shock Theory</td>
<td>66</td>
</tr>
<tr>
<td>6.2.2</td>
<td>Air Spring Curve Measurements</td>
<td>68</td>
</tr>
<tr>
<td>6.3</td>
<td>Damping Rate Characterization</td>
<td>73</td>
</tr>
<tr>
<td>7</td>
<td>MATLAB © Model Development</td>
<td>79</td>
</tr>
<tr>
<td>7.1</td>
<td>1 DOF Model</td>
<td>79</td>
</tr>
<tr>
<td>Section</td>
<td>Page</td>
<td></td>
</tr>
<tr>
<td>--------------------------</td>
<td>------</td>
<td></td>
</tr>
<tr>
<td>B.3 ADXL375 Board</td>
<td>129</td>
<td></td>
</tr>
<tr>
<td>B.4 MTB DAQ Bill of Materials</td>
<td>132</td>
<td></td>
</tr>
<tr>
<td>C MTB DAQ State Transition Diagram</td>
<td>134</td>
<td></td>
</tr>
<tr>
<td>D MTB DAQ Image Gallery</td>
<td>135</td>
<td></td>
</tr>
<tr>
<td>E Air Spring Curve Measurement Data</td>
<td>143</td>
<td></td>
</tr>
<tr>
<td>F Model Validation Image Processing Details</td>
<td>145</td>
<td></td>
</tr>
<tr>
<td>G Testing Documentation</td>
<td>147</td>
<td></td>
</tr>
<tr>
<td>H Testing Results</td>
<td>149</td>
<td></td>
</tr>
<tr>
<td>I Notes on the Jackalope</td>
<td>156</td>
<td></td>
</tr>
</tbody>
</table>
# List of Tables

<table>
<thead>
<tr>
<th>Table</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.1</td>
<td>Core components of the PyBoard version 1.1.</td>
<td>25</td>
</tr>
<tr>
<td>4.2</td>
<td>ADXL375BCCZ technical specifications.</td>
<td>26</td>
</tr>
<tr>
<td>4.3</td>
<td>MTB DAQ Prototype tested technical specifications compared to design requirements.</td>
<td>36</td>
</tr>
<tr>
<td>5.1</td>
<td>Rider physical characteristics important to the dynamic model.</td>
<td>46</td>
</tr>
<tr>
<td>5.2</td>
<td>2017 Ghost Kato FS 3 Components</td>
<td>46</td>
</tr>
<tr>
<td>5.3</td>
<td>2017 Ghost Kato FS 3 Frame Geometry and Mass</td>
<td>47</td>
</tr>
<tr>
<td>5.4</td>
<td>Measurements used to calculate the fore/aft position of the CG of the bike and rider.</td>
<td>49</td>
</tr>
<tr>
<td>5.5</td>
<td>Measurements used to calculate the vertical position of the CG of the bike and rider.</td>
<td>51</td>
</tr>
<tr>
<td>5.6</td>
<td>Measurements used to calculate the pitching moment of inertia of the bike and rider.</td>
<td>53</td>
</tr>
<tr>
<td>6.1</td>
<td>Bounding values on the spring rate, damping rate, and travel of the fork and rear shock of the 2017 Ghost Kato FS 3.</td>
<td>56</td>
</tr>
<tr>
<td>6.2</td>
<td>Values for the dimension shown in Figure 6.4.</td>
<td>60</td>
</tr>
<tr>
<td>6.3</td>
<td>Force scale factors and equivalent travel lengths for the front and rear wheels.</td>
<td>65</td>
</tr>
<tr>
<td>6.4</td>
<td>Calculated values for $L_1$, $L_2$, $A_1$, $A_2$, and $C_1$ for each shock.</td>
<td>71</td>
</tr>
<tr>
<td>6.5</td>
<td>Recommended static air pressure settings based on rider weight for the Rockshox Recon TK Silver 27.5&quot; fork.</td>
<td>74</td>
</tr>
<tr>
<td>7.1</td>
<td>Conditions used for the 1 DOF model drop test. These values are for testing only and do not correspond to any particular units.</td>
<td>82</td>
</tr>
<tr>
<td>7.2</td>
<td>Values for the dimensions of the model depicted in Figure 7.5.</td>
<td>86</td>
</tr>
</tbody>
</table>
7.3 Output data from running GeneticAlgorithm.m with various settings for \( ga \) on the data from test 1 as defined in Figure G.1. 

8.1 Genetic algorithm testing results.

B.1 MTB DAQ Bill of Materials, page 1/2.

B.2 MTB DAQ Bill of Materials, page 2/2.

E.1 Raw and processed data from the air spring curve measurement test for the X-Fusion O2 Pro RL shock.

E.2 Raw and processed data from the air spring curve measurement test for the Rockshox Recon TK Silver 27.5” fork.

F.1 Image processing steps used to determine the position of the markers on the bike.
## LIST OF FIGURES

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.1</td>
<td>Important bicycles from history. <em>(Figures taken from [22]</em>)</td>
<td>2</td>
</tr>
<tr>
<td>1.2</td>
<td>Diagram of a Rockshox Recon TK Silver 27.5” depicting common nomenclature.</td>
<td>3</td>
</tr>
<tr>
<td>1.3</td>
<td>Diagram of an X-Fusion O2 Pro RL rear shock depicting common nomenclature.</td>
<td>4</td>
</tr>
<tr>
<td>1.4</td>
<td>Cross section of a typical air-sprung rear shock. <em>(Figure taken from [9]</em>)</td>
<td>5</td>
</tr>
<tr>
<td>1.5</td>
<td>Plot depicting the positive spring curve, negative spring curve, and combined spring curve (positive - negative) for a typical air-sprung rear shock.</td>
<td>6</td>
</tr>
<tr>
<td>1.6</td>
<td>The needle is moved in or out of the internal nozzle by means of an adjusting knob, thus changing the effective damping rate.</td>
<td>7</td>
</tr>
<tr>
<td>2.1</td>
<td>Suspension designs used on bikes in the late 1800s. <em>(Figure taken from [13]</em>)</td>
<td>9</td>
</tr>
<tr>
<td>2.2</td>
<td>Base excited linear suspension model used in the RMS optimization in [15]. It is important to note that this model cannot leave the ground. <em>(Figure taken from [15]</em>)</td>
<td>11</td>
</tr>
<tr>
<td>2.3</td>
<td>Optimal curve in the $RMS(\ddot{x}) - RMS(x_r)$ plane. <em>(Figure taken from [6]</em>)</td>
<td>13</td>
</tr>
<tr>
<td>2.4</td>
<td>Comparison of the different sampling frequencies for data taken from an accelerometer placed on the front axle during a test of riding a bike directly into a curb at 10-12 km/h. <em>(Figure taken from [16]</em>)</td>
<td>18</td>
</tr>
<tr>
<td>4.1</td>
<td>Data acquisition system concept design. The concept calls for the front and rear accelerometers to be mounted so that their axes are perpendicular to the ground and parallel to the direction of travel.</td>
<td>24</td>
</tr>
<tr>
<td>4.2</td>
<td>Main unit component of the MTB DAQ system.</td>
<td>27</td>
</tr>
<tr>
<td>4.3</td>
<td>Accelerometer units of the MTB DAQ system.</td>
<td>28</td>
</tr>
<tr>
<td>Section</td>
<td>Description</td>
<td>Page</td>
</tr>
<tr>
<td>---------</td>
<td>-----------------------------------------------------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>4.4</td>
<td>Main unit features of the MTB DAQ system.</td>
<td>29</td>
</tr>
<tr>
<td>4.5</td>
<td>Main unit circuit boards.</td>
<td>30</td>
</tr>
<tr>
<td>4.6</td>
<td>Side view of the main unit depicting the installed circuit boards and battery mounting locations.</td>
<td>30</td>
</tr>
<tr>
<td>4.7</td>
<td>ADXL375 board used on both the front and rear accelerometer units.</td>
<td>33</td>
</tr>
<tr>
<td>4.8</td>
<td>Main unit of the first constructed prototype of the MTB DAQ mounted on the 2017 Ghost Kato FS 3.</td>
<td>36</td>
</tr>
<tr>
<td>4.9</td>
<td>Vertical acceleration data (after compensating for zero-g offset) and corresponding velocity and position found using the MATLAB® cumtrapz function.</td>
<td>38</td>
</tr>
<tr>
<td>4.10</td>
<td>Data from Figure 4.9 after implementing a high pass filter with a cutoff frequency of 1 Hz on the velocity data before integrating to find position.</td>
<td>40</td>
</tr>
<tr>
<td>4.11</td>
<td>Close up of jumper used to enable DFU mode. (a) depicts the jumper in the STORE position and (b) depicts the jumper in the DFU position.</td>
<td>42</td>
</tr>
<tr>
<td>5.1</td>
<td>2017 Ghost Kato FS 3 equipped with the MTB DAQ.</td>
<td>45</td>
</tr>
<tr>
<td>5.2</td>
<td>Schematic for the setup used to calculate the fore/aft position of the CG of the bike and rider.</td>
<td>48</td>
</tr>
<tr>
<td>5.3</td>
<td>Schematic for the setup used to calculate the vertical position of the CG of the bike and rider.</td>
<td>50</td>
</tr>
<tr>
<td>5.4</td>
<td>Diagram of an arbitrary body rotating about an arbitrary fixed axis, $A$.</td>
<td>52</td>
</tr>
<tr>
<td>5.5</td>
<td>Testing setup used to measure the pitching moment of inertia of the bike and rider.</td>
<td>53</td>
</tr>
<tr>
<td>6.1</td>
<td>“Clicks from tortoise” references on the Rockshox Recon TK Silver 27.5” fork and the X-Fusion O2 Pro RL shock.</td>
<td>55</td>
</tr>
<tr>
<td>6.2</td>
<td>Free body diagram of the front wheel and fork.</td>
<td>57</td>
</tr>
<tr>
<td>6.3</td>
<td>Schematic used for the four-bar linkage analysis overlaid on a picture of the rear linkage.</td>
<td>59</td>
</tr>
</tbody>
</table>
6.4 Schematic denoting how each dimension is defined for the schematic shown in Figure 6.3. ........................................... 59

6.5 Autodesk® Fusion 360® model of the rear linkage used to determine the maximum travel. ........................................... 61

6.6 Comparison of the input angle $\theta$ to the output angle $\beta$ of the rear linkage. The relationship between these two angles shows that the rear linkage is linear. ........................................... 61

6.7 Free body diagrams of Links 1, 2, and 3. ............................... 62

6.8 Plot of $F_R$ and $F_S$ (with $F_R$ set to a constant value of 1 N) over the entire range of travel of the rear linkage ($0 \leq \theta \leq 13.6^\circ$). .............. 64

6.9 Diagram of a typical air shock. ........................................... 66

6.10 Theoretical air spring force curve as described by Equation 6.24. 68

6.11 Air spring curve measurements testing setup. ......................... 69

6.12 Air spring curves for the front and rear shocks for a variety of different static pressures. ........................................... 70

6.13 Overlay of general spring curves given by Equation 6.27 and the plots from Figure 6.12. ........................................... 71

6.14 General air spring curve for each shock over the entire range of recommended pressures. ........................................... 72

6.15 General air spring curves and their “equivalent” linear spring curves for the selected pressure values given in Table 6.5. .............. 76

6.16 Estimated damping rate as a function of clicks from tortoise for the front of the bike. ........................................... 77

6.17 Estimated damping rate as a function of clicks from tortoise for the rear of the bike. ........................................... 78

7.1 1 DOF model set-up subject to a base excitation. ................. 80

7.2 Response of the 1 DOF MATLAB® model subject to the conditions given in Table 7.1. ........................................... 83

7.3 Comparison of the effects of solver time step on the bottoming out feature of the 1 DOF model. ........................................... 84
7.4 Response of the 1 DOF MATLAB® model subject to the conditions
given in Table 7.4 as well as step input of magnitude 2 at time 30
seconds. .......................................................... 85

7.5 2 DOF model geometry. ........................................ 86

7.6 2 DOF model subject to a base excitation. ................. 88

7.7 Free body diagram (FBD) and mass acceleration diagram (MAD)
for the 2 DOF model. .............................................. 89

7.8 Locations of the markers attached to the bike in order to measure
the response of the bike and rider using graphical methods. ...... 93

7.9 Testing setup used to measure the response of the bike and rider in
order to validate the model. ........................................ 94

7.10 Single frame from the video file output of the Model_Validation.m
MATLAB® script depicting the traced out position of the front and
rear markers as well as the bike and rider CG. .................... 95

7.11 Overlay of the measured response and the calculated response from
the model. The shaded region represents ±0.01 m above and below
the measured response. ............................................. 96

7.12 Overlay of the measured pitching angle and the calculated pitching
angle from the model. The shaded region represents ±2° above and
below the measured pitching angle. .............................. 97

7.13 Input position data as measured by the accelerometers for the front
and rear of the bike during the validation testing session. The input
data here corresponds with the responses shown in Figures 7.11 and
7.12. ......................................................................... 98

7.14 Various responses of the vertical position of the CG from different
 suspension settings (pressure and clicks from tortoise) for the same
 input conditions as computed by the model. ...................... 100

7.15 Various responses of the pitching angle of the CG from different
 suspension settings (pressure and clicks from tortoise) for the same
 input conditions as computed by the model. ...................... 101

7.16 Typical console output from the ga algorithm. .............. 105

8.1 Various courses designed to emulate different features common to
most mountain bike trails. ........................................... 110
8.2 Comparison of the vertical position response and pitching angle response from the bike when riding with the original settings and the genetic algorithm recommended settings with the data from test 4.

8.3 Zoomed in and normalized version of the plot from Figure 8.2 (b) depicting the vertical position responses with the data from test 4.

A.1 Schematic of the PyBoard v1.1 [11].

B.1 Top of main board rendering.
B.2 Bottom of main board rendering.
B.3 Side of main board rendering.
B.4 Main board EAGLE schematic, page 1/3.
B.5 Main board EAGLE schematic, page 2/3.
B.6 Main board EAGLE schematic, page 3/3.
B.7 Top of UI board rendering.
B.8 Bottom of UI board rendering.
B.9 Side of UI board rendering.
B.10 UI board EAGLE schematic, page 1/1.
B.11 ADXL375 top of board rendering.
B.12 ADXL375 bottom of board rendering.
B.13 ADXL375 side of board rendering.
B.14 ADXL375 board EAGLE schematic, page 1/1.

C.1 MTB DAQ state transition diagram.

D.1 Main unit of the first MTB DAQ prototype mounted on frame with power on and test number 20 on display.
D.2 Top view of main unit.
D.3 Bottom view of main unit.
1.1 A Brief History of the Bicycle

The first known record of a “bicycle-like” device is credited to a German man named Baron von Drais, and that device was patented in France in 1818. Drais called the machine “Laufmaschine” (which loosely translates to “running machine”) and invented the device for use in his garden [22]. Figure 1.1 (a) depicts a drawing of this machine. Note that it is also referred to as the “Dandy Horse.” According to the account given in Archibald Sharp’s book *Bicycles and Tricycles*, riders could achieve speeds of up to 10 miles per hour fairly efficiently on moderately smooth and level ground [22]. In 1879, Mr. H. J. Lawson invented the rear driving safety-bike, the first bike to have the same diameter wheels in the front and the back and a chain drive system [22]. A similar variation of this bike, created by Messrs. Starley and Sutton eventually gained popularity and formed the basis for the modern bicycle. Figure 1.1 (b) depicts one of these bikes.

Most of the bikes manufactured at the time were mainly designed for road use. While there were many dirt roads at the time, there were no major design considerations that were made specifically for rough terrain. Despite this fact, there are some early accounts of people using bikes for long and rugged off-road journeys. Most notably is the use of road bikes in 1896 by the 25th Infantry Bicycle Corps, an all black infantry division that modified bikes to carry equipment and rode them from Missoula, Montana, to Yellowstone National Park and back. Their goal was to test out bicycles for military use in mountain terrain [2]. Mountain biking as it is known today,
Figure 1.1: Important bicycles from history. *(Figures taken from [22])*

however, is generally credited to starting on Mt. Tamalpais in Marin County in the 1960s and 1970s. A group of teenagers, known as the “The Larkspur Canyon Gang,” started by riding modified road-bikes down the trails and fire roads. Eventually they started a downhill race series that came to be known as “Repak,” since the coaster brakes would need to be re-packed with grease after each race. In time, the event gained popularity, and ultimately led to the creation of the formal sport of mountain biking as it is known today [2].

1.2 Modern Mountain Bike Shocks

Modern mountain bike shocks come in many shapes and sizes and can be customized for a wide variety of riding styles. Almost every mountain bike has a shock built into the fork of the bike, providing relief from shocks that strike the front wheel. A mountain bike that only contains a shock in the fork is commonly referred to as a “hard tail.” Figure 1.2 depicts the common nomenclature used to describe the
different parts of the fork of a mountain bike. Most forks are constructed with an
air spring located in one stanchion and a damper located in the other. The pressure
in the air spring can be adjusted via a Schrader valve located on the top of the air
spring stanchion, and the rebound damping rate can be adjusted with a small dial on
the bottom of the damping stanchion.

As the name implies, full-suspension mountain bikes include a shock in the rear as
well as in the fork. Rear shocks are composed of either a coil spring or air spring in
combination with a damper. Figure 1.3 depicts a common rear shock. Most modern
shocks (both front (fork) and rear) include ways to tune different parameters of the
shock. One such feature common to both air-sprung and coil-sprung shocks is the

Figure 1.2: Diagram of a Rockshox Recon TK Silver 27.5”
depicting common nomenclature.
ability to independently control the compression and extension damping rates. Air-sprung shocks are also capable of tuning the spring rate curve by adjusting air pressure and the volume of the air chamber. Figure 1.4 depicts a cross-section of a common rear mountain bike shock. As can be seen in Figure 1.4, air-sprung mountain bike shocks include a secondary air chamber that is referred to as a “negative air spring.” Without a negative air spring, the positive air spring initially requires a force equal to the internal pressure times the cross-sectional area of the chamber in order to start compressing. On a mountain bike shock, this leads to poor performance when travelling over obstacles that result in small forces, for these small forces are not great enough to overcome this initial force and result in a stiff and bumpy ride [7]. By adding a negative air spring, this initial force is essentially cancelled and makes it much easier for the shock to be compressed from small forces. Figure 1.5 depicts the spring curve from the positive air chamber only, the negative air chamber only, and the combined result. It is important to note that Figure 1.5 was generated using the theory developed in Section 6.2.1. The negative air spring results in a more linear spring curve similar to a coil spring. At the far end of the travel, however, the
spring rate becomes more progressive. This feature of air springs helps ensure that they do not “bottom out,” or reach the limit of their travel, thus resulting in a hard shock to the rider. By adjusting the volume of both the positive and negative air spring chambers, as well as the pressure, the characteristics of the spring curve can be adjusted for optimal performance.

The damping mechanism in a shock contains a labyrinth of chambers and ports filled with fluid. When the shock is compressed, it moves the fluid and forces it through small passageways and gaps. The friction that results from pumping the fluid through this complex system dissipates the energy absorbed by the shock. By utilizing one-way valves the effective compression and extension damping rates can be adjusted independently through the use of needle valves. Figure 1.6 depicts the mechanism used to make this adjustment. Needle valves are valves that work by driving a needle in or out of an internal nozzle, thus creating a smaller or larger ring for fluid to flow through and effectively changing the damping rate. By advancing the needle further
in the nozzle, fluid flow is restricted and the effective damping rate increases. Pulling the needle out makes the fluid flow easier and lowers the effective damping rate.

It is important to note that not all mountain bike shocks contain the features mentioned here. Some shocks only allow the rebound damping rate to be controlled, others provide independent control for high and low speed damping. The world of mountain bike shocks is complex. The overall principles, however, apply to the vast majority of shocks and are presented here to provide a good understanding of the internal workings of the shocks used.
1.3 Purpose

Mountain bike design has largely been driven by user feedback. A bike would be designed, built, and tested, with suggestions for the next bike largely derived from human feedback on performance and function. Originating in Marin County, mountain biking began with a group of people that decided to simply take a bike off road and hold on tight. When compared to the present day, where one can purchase a mountain bike with electronic shifting and a carbon fiber frame, it becomes clear that mountain bikes are being engineered for improved performance now more than ever.

The suspension arguably has the largest effect on the performance of a mountain bike. While the suspension primarily functions to isolate the rider from large bumps and discontinuities in the road, it is also a key component for maintaining traction [17]. In short, the suspension plays a large role in defining the quality of the ride and how the user must respond.
In order to get the best possible performance out of a given suspension system, it must be tuned properly to match both the skill level and physical characteristics of the rider. An improperly tuned suspension can make an amazing bike feel horrible, and a properly tuned suspension can make a mediocre bike feel awesome [19]. Suspension performance is directly related to rider comfort and performance.

With few exceptions, current methods of suspension tuning rely heavily on “rules of thumb” and “what feels right.” Andrew Richardson, author for the mountain biking magazine *Enduro*, outlines a comprehensive method for dialing in a mountain bike suspension. He points to tried and true values for starting points and then recommends iterating over different settings until the bike reaches a specific feel [19]. While humans are good at sensing their environment, they have a hard time being consistent and are susceptible to a large range of factors such as what they ate or the mood that they happen to be in that day [17].

The bicycle manufacturer Specialized highlighted the importance of rider comfort with the release of the Future Shock, a head tube suspension system designed to increase the ride comfort and smoothness of gravel bikes. Released in 2016, the Future Shock was a direct result of the mentality “smoother *is* faster.” By partnering with the high-performance car manufacturer McLaren and implementing the “McLaren Rolling Efficiency Model,” Specialized was able to quantify “smoothness” and show that it results in an increase in performance.

Accordingly, a quantitative method for tuning mountain bike suspensions, independent of user feedback, would be a valuable asset to the mountain biking industry. By creating a way to allow riders to make data-driven suspension tuning decisions, it would be possible to reliably ensure the best riding experience possible.
2.1 The Role of Suspension

Bicycle suspension systems have been around since early on in the history of the bike. Figure 2.1 depicts some suspension designs that were used on bikes in the late 1800s. In the simplest form, a bike suspension system is composed of a spring and a damper. A spring is a device that deforms elastically under the application of an applied force and a damper is a device that removes energy from a system. On bikes, these two devices are usually combined together into a component known as a shock. The spring serves to isolate the rider from bumps by absorbing the energy, and the damper both partially absorbs and dissipates this energy. Without a damper, the bike would oscillate many times after impacting a bump since there is no easy way for the

Figure 2.1: Suspension designs used on bikes in the late 1800s. (*Figure taken from [13]*)
energy to be dissipated. Without a spring, a damper would not be able to support the weight of the bike. The two work together to isolate the rider from undesired bumps.

Good and McPhee write that a mountain bike suspension essentially has four functions [13]. They are as follows:

1. Isolate the rider from the roughness of the road.
2. Absorb energy and shocks that comes from hitting large obstacles.
3. Keep the wheels on the ground to maintain control and traction.
4. Avoid adding undesirable characteristics to the bicycle.

As mentioned previously, the performance of this system is directly related to the overall performance of the bike and the ability of the rider. A properly functioning suspension system dramatically increases the ability and comfort of the rider.

2.2 Suspension Optimization

Jazar introduces a method for optimizing a linear, one degree-of-freedom, base excited suspension model using the root mean square (RMS) optimization method [15]. Figure 2.2 depicts the system setup, with spring stiffness $k$, damping ratio $c$, and sprung mass $m$. It is important to note that in this model the wheel cannot leave the ground. By defining the relative displacement, $z$, as $z = x - y$, the equation of motion is then given by

$$m\ddot{z} + c\dot{z} + kz = -m\ddot{y}$$  \hspace{1cm} (2.1)
By applying a harmonic excitation $y = Y \sin(\omega t)$ and solving Equation 2.1, an expression for the relative displacement amplitude, $Z$, and the absolute displacement, $X$, can be found. These values can be used to define the following quantities,

$$S_Z = \frac{|Z|}{Y}$$  \hspace{1cm} (2.2)$$

$$G_Z = \frac{|\ddot{X}|}{Y \omega_n^2}$$  \hspace{1cm} (2.3)$$

where $S_Z$ is the relative displacement and $G_Z$ is the absolute acceleration. These parameters, known as frequency responses, are important in characterizing the performance of a suspension. Taking the root mean square (RMS) on these parameters gives $S_Z = RMS(S_Z)$ and $S_{\dot{X}} = RMS(S_{\dot{X}})$. Finally, the optimal spring stiffness, $k$, and damping ratio, $c$, can then be found by minimizing $S_{\dot{X}}$ with respect to $S_Z$. Mathematically, this is done by solving the following minimization problem

$$\min_{S_Z} S_{\dot{X}}$$
\[
\frac{\partial S_x}{\partial S_z} = 0
\] (2.4)

\[
\frac{\partial^2 S_x}{\partial S_z^2} > 0
\] (2.5)

While there are many ways to solve this problem, R Alkhatib, N. G. Jazar, and M.F Golnaraghi outline a solution procedure through the use of a genetic algorithm [6]. In short, a genetic algorithm aims to solve an optimization problem by mimicking how species evolve in nature, as described by Charles Darwin’s theory of evolution by natural selection. The algorithm mathematically simulates this behavior by doing the following. Potential solutions are randomly generated and selected. These solutions are compared to a so-called objective function, or a function that is made up to evaluate the “fitness” of a solution (how well it solves the problem). Once each solution has been evaluated for fitness, only the most “fit” solutions are allowed to “mate” and form the next generation of test solutions. In other words, the most “fit” solutions (“most-fit” being defined by some criteria in the algorithm) are paired up and then parts from each are combined together to form “children” solutions. These solutions are then evaluated based on the objective function, and the cycle continues. By iterating over many generations of possible solutions, the optimal solution can be found. In order to avoid converging to local maximum or minimum values, mutations are introduced into each generation. During each generation, a certain percentage of the population of solutions (the percentage of populations being a characteristic of the genetic algorithm) is exposed to a mutation, where part of the solution is randomly changed. This creates a large genetic diversity and ensures that the absolute maximum or minimum value is reached. By using this method to iterate over a range of input frequencies from \(0 < f_n < 20\) Hz, Alkhatib et al. present a plot of the optimal suspension design curve, as shown in Figure 2.3 [6]. It is important to note
that constraints need to be applied during the optimization problem in order to avoid
converging to a trivial solution. This is due to the fact that it is not possible to
minimize both the RMS of absolute acceleration and the RMS of relative position.
Refer to [6] for a more complete description.

Recall that this plot was generated by solving the following optimization problem:
minimizing the RMS of absolute acceleration, $S_{\ddot{X}}$, with respect to the RMS of rel-
ative position, $S_{Z}$. It follows that this is not the only optimization problem that
can be solved; there are other criteria with which one can try to optimize a sus-
pension to. Moreover, this problem was solved by assuming a base excitation input
given by $y = Y \sin(\omega t)$.

In more general terms, the optimization of a suspension system through the use of a
genetic algorithm consists of three steps:

1. Choose what parameter should be considered in the optimization to best suit
the desired behavior. This is what is used to create the objective function.
2. Select which parameters are allowed to vary during the solution procedure.

3. Define constraints to avoid trivial solutions.

These steps are important, for they outline what the “optimal” system is. While there is no one universally accepted method, the following parameters are important in any suspension design.

1. Absolute acceleration - The absolute acceleration directly defines how much force is going through the system. As a result, this parameter is very important to consider in the objective function.

2. Relative displacement - The relative displacement characterizes how far the suspension needs to be able to travel in order to work properly. If not considered properly, the suspension could bottom out, or reach the limit of its travel, thus shocking the system and possibly causing damage. The relative displacement needs to be considered in the objective function to ensure a properly functioning system.

The methods used by Alkhatib et al. that are discussed here demonstrate a simple suspension optimization solution and provide a basis for a quantitative method for tuning a mountain bike suspension.

2.3 Road Spectral Analysis

Gillespie and Dixon discuss the spectral analysis of a road, as described by the power spectrum density (PSD) function [12] [10]. This method involves plotting the amplitude versus the frequency and can be used to characterize different roads/driving conditions. This can be used to describe the roughness of the road, or the deviation
in elevation that is seen in the vehicle as it moves along the road. Rui, Saleem, and Zhou outline a method of calculating road loads by applying a PSD function for a given road to a simple dynamic model of a vehicle, and then comparing those results to those measured on an actual vehicle when subjected to the same PSD function on a test stand [25]. While the PSD function is not used in the tuning algorithm that is developed later, the works discussed here have been left in as a reference of what could be done in the future.

2.4 ShockWiz™

The ShockWiz™ is a commercially available mountain bike suspension tuning system for air-sprung suspension systems [3]. ShockWiz™ started out as a Kickstarter project and was acquired by SRAM in 2016, where it was placed under their Quarq product line. The ShockWiz™ uses a pressure sensor to sense the changes in the pressure of the suspension while riding. As such, it will only work on mountain bike suspensions that are air-sprung and have a positive chamber with a single volume. The data are sent to a smartphone and processed to suggest ways to tune the suspension for a better ride. According to their website, a single ShockWiz™ can be used to tune a full-suspension mountain bike by alternating the device between the front and rear shocks [3]. The ShockWiz™ is an example of one method for quantitatively tuning a mountain bike suspension. It is referenced here to provide more information on other techniques people have used to try and quantify the suspension tuning process.

2.5 Mountain Bike Ride Comfort

Nordstrom provides a method for quantitatively measuring the ride comfort of a mountain bike [17]. This method involves using unweighted RMS acceleration data
taken from a bike seat post that is normalized against the acceleration of the rear axle. Only vertical (up and down with respect to a bike traveling along the ground) accelerations are considered. The resulting quantity, called the discomfort transmission, $T_D$, is given by

$$
T_D = \frac{S_{RMS}}{A_{RMS}} = \sqrt{\frac{1}{m} \sum_{j=1}^{m} S_j^2} \sqrt{\frac{1}{m} \sum_{j=1}^{m} A_j^2}
$$

where $S_{RMS}$ is the seat ride discomfort (root-mean-square of acceleration data), $A_{RMS}$ is the axle ride discomfort (root-mean-square of axle acceleration data), $S_j$ represents the individual acceleration samples at the seat, $A_j$ represents the individual acceleration samples at the axle, $j$ is the sample number, and $m$ is the total number of samples. It is important to note that Nordstrom discusses how the sensitivity of humans to vibrations changes non-linearly at different frequencies (i.e. humans are good at perceiving large changes in vibrations versus small ones) and at different accelerations. He goes on to discuss that different weighting methods have been developed to try and account for these differences. In the end, however, an unweighted RMS technique proved to work quite well. The data was normalized against the rear axle acceleration in order to account for the fact that trail profiles are incredibly irregular and constantly changing. Ideally, the trail profile would be measured at the contact patch between the tire and the trail. To keep measurements simple, however, the profile can be estimated by measuring the accelerations of the rear axle. This leads to an effective method for evaluating the performance of a suspension system.

### 2.6 Measuring Accelerations of a Bike Frame

Koellner, Cameron, and Battley present a method for conducting bicycle field test studies to measure both strain and accelerations of a bike in use [16]. The system
incorporates 24 sensors measuring strain and 4 sensors measuring acceleration that are connected to a central unit that sits behind the seat post. Koellner et al. describe their initial research to determine the optimal sampling frequency. They started using a BMX bike, chosen for its high stiffness, low weight, and low volume high-pressure tires, providing a good testing platform with minimal damping. A data-acquisition system consisting of four piezo-electric accelerometers mounted on the front and rear axle, the seat clamp, and under the saddle was installed on the bike. In order to characterize normal riding conditions various tests were performed to capture continuous excitation events (such as riding along a road) and single-event scenarios (such as bumping up on a curb). The various tests were conducted at a data rate of 10,000 Hz. These data were then down-sampled using MATLAB® code to represent data rates of 100 Hz, 200 Hz, 500 Hz, 1,000 Hz and 2,000 Hz. Figure 2.4 depicts a comparison of the acceleration data from these different rates from the test of riding the bike directly into a curb at 10-12 km/h. From this preliminary research, Koellner et al. conclude that a sampling frequency of 1000 Hz is the optimal rate. As can be seen in Figure 2.4, frequencies below this rate tend to clip peaks in acceleration and frequencies above this rate do not provide any additional information. It is important to note that the maximum acceleration from this test peaked around 35 times larger than g, the acceleration due to gravity. A separate test conducted by Koellner et al., consisting of jumping the BMX bike off of a curve, saw similar peak acceleration values.
Figure 2.4: Comparison of the different sampling frequencies for data taken from an accelerometer placed on the front axle during a test of riding a bike directly into a curb at 10-12 km/h. *(Figure taken from [16]*)
Chapter 3

PROJECT OUTLINE

The goal of this thesis is to find a quantitative method for tuning a mountain bike suspension. From a high-level perspective, this method involves taking some sort of data while riding the bike, feeding this data into a model, and developing an algorithm that will determine the optimal suspension parameters. Based on the findings outlined in Section 2, the overall outline of the project can be summarized as follows:

1. Develop a data acquisition system that will collect acceleration data at the front and rear axles while riding the bike. Acceleration data is easy to collect using accelerometers and can be installed on any bike. Further, the data is fairly straightforward to feed into a dynamic model of the bike. Section 4 will cover the development of the data acquisition system.

2. Characterize a mountain bike and rider system by measuring the following physical properties:

   - location of the bike and rider CG
   - pitching moment of inertia of the bike and rider about the CG
   - front and rear shock spring rates and damping rates as a function of user adjustable parameters
   - effective spring rates and damping rates acting directly on the front and rear axles from the shocks

These parameters are needed in order to define a dynamic model of the bike. Section 5 will go over the methods used to measure these quantities.
3. Develop and validate a model of a full suspension mountain bike and implement a method for finding the optimal suspension parameters. Section 7 will cover the development of this model.

Since the mountain bike model is heavily dependent on what data can be obtained, the data acquisition was developed first.
Data acquisition systems capable of recording data from multiple accelerometers with a large measurement range and at a high frequency are incredibly expensive and usually very bulky. A quote from the data logging company MicroDAQ listed the price for a three axis data logger with two external $\pm 200$ g accelerometers as approximately $2,000$. In order to gather the necessary data to feed into the models developed in Section 7, it was necessary to develop a low-cost system custom tailored for the application. The following sections present the design of the mountain bike data acquisition system, or “MTB DAQ.”

4.1 Design Requirements

In order to gather useful data, certain design requirements needed to be met. The following list outlines the design requirements that were used when designing the data acquisition system.

1. **The system must be capable of measuring a maximum acceleration of 100 g’s.**

As mentioned in Section 2.6, the maximum acceleration that is expected to be measured is around 35 g’s [16]. Steve Hanly, author for Midé Technology Corporation, a company that sells accelerometer data-logging systems, recommends in his article *Accelerometer Specifications: Deciphering an Accelerometer’s Datasheet* that an accelerometer should ideally be selected such that the maximum acceleration that is predicted is 20% of the maximum acceleration
that can be measured by the accelerometer [14]. As a result, the system will be
designed to measure a maximum acceleration of 100 g’s.

2. **The system must operate at a sample frequency of at least 1,000 Hz.**
   As indicated previously in Section 2.6, Koellner et al. conclude that the opti-
mal sampling frequency for collecting acceleration data while riding a bike is
1,000 Hz [16]. Therefore, the system will be designed to operate at a frequency
of at least 1,000 Hz.

3. **The main board must mount to the water bottle bosses on a frame
   and be no larger than a water bottle in size.**
   In order to minimize the footprint of the system on the bike and the riding
experience, as well as to ensure easy installation, the system will be designed
to mount on standardized hardware that exists on most mountain bikes.

4. **The system must run on batteries with a life of at least 1 hour.**
   The system will need to be battery powered in order to allow for proper testing.
   A minimum battery life of 1 hour was selected in order to ensure plenty of time
to conduct tests during a testing session.

5. **The system must store the data as a *.CSV file.**
   The data will be saved in this file format for easy use with the MATLAB®
model as discussed in Section 7.

The system was designed to use the Micropython programming language. Micropy-
thon is an implementation of the Python 3 programming language that is designed
to run on microcontrollers. It was created by Damien P. George in 2013 and has
been supported by a large online community ever since. The original version of the
language is designed to run on the ST Microelectronics STM32F405RTG6 chip. Tra-
ditionally, the C programming language is used to program microcontrollers. While
programming in C allows for more efficient code and better portability between microcontroller chip sets, the barrier to entry is incredibly high. An in-depth knowledge of the microcontroller is required and considerable effort is necessary to development the proper tool-chain for programming. Similar to Arduino, Micropython eliminates this large barrier to entry by allowing for quick and easy programming of the microcontroller without the support and knowledge of setting up a tool-chain. As a result, any programmer who is sufficiently trained in Python can relatively easily write code for embedded systems. Therefore, developing the system for Micropython proved to be the best option. The ease of use speeds up the development process and the low barrier to entry means the entire data acquisition system could easily be adapted for other academic uses.

4.2 Conceptual Design

As mentioned in Section 3, the MTB DAQ is designed to record acceleration data at the front and rear axles of a bike being ridden. Figure 4.1 depicts the conceptual design for a system capable of doing this. As shown in Figure 4.1, the concept calls for a central unit that is mounted on the water bottle bosses of the frame and contains the microcontroller, batteries, and user interface. Accelerometers, connected to the central unit via cables, are mounted on the front and rear axles of the bike. This configuration allows acceleration data to simultaneously be taken and stored from multiple locations on the bike. The minimal requirements of this system mean it could very easily be installed on almost any bike with almost no modification necessary.

The accelerometers are mounted such that the axes are parallel to the direction of travel and perpendicular to the ground as shown in Figure 4.1. As will be shown in Section 7, the model only calculates the vertical accelerations and pitching angular
accelerations of the bike and rider. Therefore, the front and rear suspension components are modeled as spring and damper pairs acting vertically above the front and rear axles. This simplification of the suspension components required that the acceleration data be collected in the orientation shown.

4.3 Component Selection

Micropython is user friendly because it was originally designed to work on one chip (the ST Microelectronics STM32F405RGT6) mounted on a custom board known as the PyBoard. By locking in the hardware, Micropython comes with all of the hardware-specific code already defined and ready for use. Consequently, when designing a custom board for use with Micropython it is necessary to match the core components in order to ensure compatibility.
The MTB DAQ was designed to work specifically with the firmware developed for the PyBoard version 1.1 (PYBv1.1). This choice was made for two main reasons:

1. At the time of this writing, this was the most updated version of the PyBoard. Consequently, the firmware is constantly being improved and is readily available for download on the Micropython website [11].

2. More importantly, the firmware built for the PYBv1.1 included the incredibly fast SDIO protocol for interfacing with SD cards. While this subject will be discussed in further detail, it is sufficient to say that this was necessary in order to be able to achieve a sampling frequency of at least 1,000 Hz as specified in Section 4.1.

The core components of the PYBv1.1 are the microcontroller, the microcontroller clock crystal oscillators, and the real time clock crystal oscillator. Table 4.1 summarizes these core components. With the core components of the PyBoard identified, the

Table 4.1: Core components of the PyBoard version 1.1.

<table>
<thead>
<tr>
<th>Component</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Microcontroller</td>
<td>STM32F405RGT6</td>
</tr>
<tr>
<td>Microcontroller Clock Crystal Oscillator</td>
<td>12 MHz</td>
</tr>
<tr>
<td>Real Time Clock Crystal Oscillator</td>
<td>32.768 kHz</td>
</tr>
</tbody>
</table>

remaining key component to be chosen was the accelerometer. There are two main types of accelerometers: analog and digital. In general, analog accelerometers can take data at higher frequencies and over a larger range, but they are very expensive. Digital accelerometers are incredibly inexpensive and are only getting faster as technology progresses. In order to keep the system as low cost as possible, it was necessary to find a digital accelerometer with the proper performance characteristics. Based on the requirements laid out in Section 4.1, the ADXL375BCCZ digital accelerometer
from Analog Devices, Inc. was chosen. Table 4.2 summarizes the key specifications of this accelerometer. As shown in Table 4.2, the ADXL375BCCZ can measure up to 200 g’s at a maximum rate of 3,200 Hz. Further, it has a relatively low 0 g offset and noise specification. As noted in Table 4.2 the sensitivity is 20.5 LSB\(^1\)/g or 0.0488 g/LSB for an output data rate less than or equal to 800 Hz. According to the data sheet, the least significant bit is always zero when the data rate is greater than 800 Hz. Without knowing more about the inner workings of the accelerometer when configured for these fast output data rates, it is safe to assume that this effectively means that the sensitivity is cut in half. This apparent reduction in sensitivity is due to the fact that the accelerometer is a binary device. Loosing the least significant bit means that the smallest increment that can be counted is double the increment of the least significant bit. Therefore, for an output data rate greater than or equal to 800 Hz, the sensitivity is assumed to be 10.25 LSB/g, or 0.0976 g/LSB, still much finer than is needed by the MTB DAQ. It is important to note, however, that in practice when decoding the data the same scaling factor is used, no matter what the output data rate selected is. Even though the least significant bit is always zero when the output data rate is greater than 800 Hz, it is still present, and therefore must be accounted for with the same scaling factor of 0.0488 g/LSB. The overall sensitivity does decrease with the higher output data rates, but the calculation used to convert the data remains the same. Due to these specifications the ADXL375BCCZ was a

Table 4.2: ADXL375BCCZ technical specifications.

<table>
<thead>
<tr>
<th>Specification</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Acceleration Range</td>
<td>± 200 g</td>
</tr>
<tr>
<td>Output Data Rate (ODR) Over SPI</td>
<td>0.10 – 3.200 Hz</td>
</tr>
<tr>
<td>0 g Offset (each axis, typ.)</td>
<td>± 400 mg</td>
</tr>
<tr>
<td>Noise (each axis, typ.)</td>
<td>5 mg/RMS(Hz)</td>
</tr>
<tr>
<td>Sensitivity (each axis, typ., ODR ≤ 800 Hz)(^*)</td>
<td>20.5 LSB/g</td>
</tr>
</tbody>
</table>

\(^*\) The sensitivity is only given for an output data rate (ODR) ≤ 800 Hz

\(^1\) LSB = least significant bit
good choice to hit the design requirements. Once the key components were selected, the board design process could begin.

4.4 Hardware Design

The MTB DAQ system is composed of three units: the main unit, the front accelerometer unit, and the rear accelerometer unit. Figures 4.2 and 4.3 depict these three units. As shown in Figure 4.1, a complete system is composed of one main

![Main unit component of the MTB DAQ system.](image)

unit mounted on the water bottle bosses of the frame and two accelerometer units mounted on the front and rear axles. The accelerometer units are connected to the main unit via standard Ethernet cables. Ethernet cables were chosen because they are a common cable, meaning tooling and parts to make custom length cables is widely available and very low cost. Further, the RJ45 connector (the connector used on an Ethernet cable) includes a locking mechanism that ensures that the cables do not come loose while riding.
4.4.1 Main Unit Design

The main unit was designed to work around the Hammond Mfg. part no. 1455J1201 extruded aluminum electronics enclosure. The enclosure is part of a series of electronics enclosures that are designed to hold electronic circuit boards. This specific enclosure was chosen because it is the smallest enclosure that had a removable side panel while still being just tall enough to fit an RJ45 port. The main unit features two RJ45 ports for plugging in the accelerometers, three status indicator LEDs (power, charging, and recording), a record button, a 4-character 14-segment display, a Micro SD card slot, and a USB Mini B port. There are two M5X0.8 screws that run through the unit allowing for easy attachment on to water bottle bosses of a bike frame. Figure 4.4 depicts these features of the main unit.

The main unit is composed of two 2-layer circuit boards housed inside of the enclosure: the UI board and the main board. The UI board connects to all of the devices that make up the user interface (the three status indicator LEDs, the display, and the record button). A ribbon cable connects the UI board to the main board. The
main board contains all of the other electronics, such as the microcontroller, crystal oscillators, power filtering circuit, battery charging circuit, Micro SD slot, RJ45 ports, and USB port. A two board design was used strictly for clearance/mounting purposes. In other words, it was necessary in order to be able to assemble the panel-mounted user interface components. With this configuration, assembly and disassembly of the unit is straightforward, allowing for easy access to the electronics to assist with troubleshooting. Figure 4.5 depicts these two boards separately and Figure 4.6 depicts the boards installed in the enclosure. As shown in Figure 4.6, the main unit includes Lithium Polymer batteries located under the main board. The unit includes a total of five 3.7 volt batteries, four 320 mAh batteries and one 1,200 mAh battery, hooked up in parallel to make the equivalent of one 2,480 mAh battery, glued to the bottom of the main board.

The main board and UI board were both designed using Autodesk® Eagle™. The circuit board designs were linked with a 3D model of the main unit made in Autodesk® Fusion 360® in order to create a detailed 3D model of the entire system. With these
powerful design tools, many aspects of the design were worked out in CAD allowing for a very compact system. Design of the UI board was fairly straightforward, for its’ only function is to provide an electrical connection between the user interface components and the ribbon cable connecting to the main board. Therefore, any further discussion on the design of this board is not needed. The main board, however, was more complex. Recall in Section 4.1 that the use of the Micropython programming language greatly restricts the hardware in order to ensure compatibility. Therefore, when designing the core functionality of the main board, it was necessary to exactly
replicate the electrical connections on that of the PYBv1.1. This replication was accomplished by referencing the PYBv1.1 schematics, available on the Micropython website [11]. The layout of the individual components on the main board was done following good PCB design practices. Carmine Noviello outlines the basics of PCB design in his book *Mastering STM32* [18]. The most effective way to design a PCB is to lay out the components in a certain hierarchy, with the placement of components at the top being more critical than those at the bottom. The following list summarizes the order in which the components were placed.

1. Microcontroller – The microcontroller has the most connections of any of the components on the board, many of which are very critical and need to follow specific design constraints. Therefore, this component was placed first directly in the middle of the board, leaving plenty of room for connecting to the surrounding components.

2. Crystal oscillators – It is important that the traces going to the crystal oscillators be as short as possible. Further, the traces going to each end of an oscillator should ideally be the same length.

3. Decoupling capacitors – Decoupling capacitors provide important power filtering to ensure a smooth supply of power to the microcontroller. In order for them to work properly, it is important that each power pin on the microcontroller has a corresponding decoupling capacitor placed as close as possible.

4. Space/functionally constrained components – These are any components that had to be in a certain location due to space and/or functionality constraints. These components include the Micro SD slot, the two RJ45 ports, the USB Mini B port, the ribbon cable connector (for the UI board), the status LEDs, and the JST connectors for the batteries.
5. Remaining components – The remaining components were placed wherever space remained.

Once the components had been spaced out on the board, the traces were drawn to make all of the necessary electrical connections. The connections were made following the same hierarchy used to layout the components with the most critical connections being made first. Wherever possible, a minimum of 0.34 mm trace width was used to route signals, and a minimum of 1 mm trace width was used to route power. The top and bottom planes were designated ground planes and were connected together with vias spaced apart all over the board.

Refer to Appendix A for a copy of the original PyBoard version 1.1 schematic. For a complete accounting of the schematics, boards, and bill of materials that make up the MTB DAQ, refer to Appendix B.

4.4.2 Front and Rear Accelerometer Unit Design

As discussed in Section 4.3, the front and rear accelerometer units were designed around the Analog Devices, Inc. ADXL375BCCZ digital accelerometer. The front and rear accelerometer units utilize the same board design, complete with proper power filtering components and an RJ45 port. The RJ45 port is connected to give easy access to the ADXL375 power and ground pins, SPI pins, and interrupt pins. Similar to the main board and the UI board, the ADXL375 board was designed in Autodesk® Eagle™ and Fusion 360®. Figure 4.7 depicts a rendering of the ADXL375 board. The front and rear accelerometer units were designed to firmly secure the accelerometers to the bike. The units are composed of 3D printed adapter blocks and covers and are secured onto the bike using rubber cushioned U-bolts. The geometry of the adapter blocks was created such that the $x$-axis of the ADXL375 unit is perpendicular to the
Figure 4.7: ADXL375 board used on both the front and rear accelerometer units.

ground and the $y$-axis is parallel to the direction of travel. While this convention is different from the convention stated in Section 4.2, the necessary adjustments can be made in software.

Refer to Appendix B for more detail on the schematics, boards, and bill of materials of the ADXL375 board.

4.5 Code Design

The MTB DAQ code is composed of various files. They are as follows:

1. `main.py` – This file contains the code that actually runs the MTB DAQ.
2. `boot.py` – This file contains code that runs whenever the system is rebooted.
3. `helperFunctions.py` – This file defines various functions that are used in `main.py`
4. **ADXL375_driver.py** – This file defines a driver for the ADXL375 accelerometer. It contains various functions that make the accelerometer configuration process more user friendly.

5. **ht16k33_seg.py** and **ht16k33_matrix.py** – These files together define a driver for the Adafruit 4-character 14-segment display. The code was sourced from the “micropython-adafruit-ht16k33” GitHub repository [24].

6. **SKIPS** – This is a completely blank file. It is stored on the board to tell Micropython to boot from the internal flash instead of the Micro SD card if a Micro SD card is present upon boot.

The files listed above work together to define a finite state machine. The finite state machine can be represented as a state transition diagram showing each state of the program and what conditions must exist in order to proceed to the next state. Refer to Appendix C for a state transition diagram of the MTB DAQ. Even though all of the files are necessary to make the finite state machine work, the actual implementation is handled in **main.py**.

While not every detail can be discussed, there are some important details of the code that should be noted. The ADXL375 accelerometers are configured for an output data rate of 1,600 Hz using the SPI protocol. This rate is the closest option greater than or equal to the desired specification of 1,000 Hz and this fast data rate is only possible over SPI. The acceleration values for each axis are 16-bit signed values, broken up into two bytes. These data are configured to be right justified with the least significant bit always zero due to the high output data rate. The accelerometers are configured to store the values to a first-in-first-out (FIFO) buffer. Once 20 values have been stored, an interrupt is generated that signals the main code to read the data. The data are then read and stored on the Micro SD card in binary format. At the output
data rate of 1,600 Hz, this means that a data point is taken every 0.625 ms, and that an interrupt is generated every \((0.625 \text{ ms})(20) = 12.5 \text{ ms}\). In order to be able to read all 20 \(x\), \(y\), and \(z\) data points from both accelerometers within 12.5 ms, it was necessary to utilize the SDIO protocol for reading and writing to SD cards. During early experimentation with the MTB DAQ, an attempt was made to use a Micro SD card breakout board connected to a PyBoard via the standard SPI pins. When using the fastest baud rate supported by the board, the microcontroller was simply not able to write to the SD card fast enough. As a result using the SDIO protocol was necessary and this was only available with the PYBv1.1 version 1.12 firmware.

### 4.6 MTB DAQ Prototype

An extensive compilation of images of the MTB DAQ prototype, including images of the system mounted on the bike as well as close ups of the internal parts and assembly procedure, can be found in Appendix D.

#### 4.6.1 MTB DAQ Technical Specifications

Figure 4.8 depicts a close up of the main unit of the first constructed prototype of the MTB DAQ installed on the 2017 Ghost Kato FS 3. Figure 5.1 depicts the entire bike with both the front and rear accelerometer units mounted. The components for the MTB DAQ were sourced from Digikey and McMaster Carr, and the boards were purchased from the PCB manufacturer OSH Park. The accelerometer mounts were 3D printed. In total, the price for the complete MTB DAQ prototype came in to be around $270, much less than the approximately $2,000 quote mentioned earlier. This price is assuming a minimum of three units are made, for the PCBs can only be purchased as a minimum of three from OSH Park.
Figure 4.8: Main unit of the first constructed prototype of the MTB DAQ mounted on the 2017 Ghost Kato FS 3.

Testing of the prototype proved that it was not able to hit all of the design requirements listed in Section 4.1. Table 4.3 lists all of the tested values corresponding to the design requirements from Section 4.1. As shown in Table 4.3, the prototype is not able to store the data as a time-stamped CSV file. This failed for two reasons. First, the timing necessary to record the data at 1,600 Hz is already fairly close to the limit of the microcontroller. Therefore, there simply was not enough time to properly convert the data to a human-readable form and write it to a text file. As a result, it was necessary that the data be stored in binary format. Various attempts were made to create a function that would read through the data after it had been recorded and

<table>
<thead>
<tr>
<th>Specification</th>
<th>Design Requirement</th>
<th>Prototype Value</th>
<th>Pass?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Measurement Range</td>
<td>±100 g</td>
<td>±200 g</td>
<td>Yes</td>
</tr>
<tr>
<td>Output Data Rage</td>
<td>&gt;1,000 Hz</td>
<td>1,600 Hz</td>
<td>Yes</td>
</tr>
<tr>
<td>Mount to Water Bottle Bosses</td>
<td>-</td>
<td>-</td>
<td>Yes</td>
</tr>
<tr>
<td>Battery Life</td>
<td>&gt;1 hour</td>
<td>&gt;8 hours</td>
<td>Yes</td>
</tr>
<tr>
<td>Store data as CSV file</td>
<td>-</td>
<td>-</td>
<td>No</td>
</tr>
</tbody>
</table>
convert it to a text file. It was quickly found, however, that converting hundreds of thousands of data points on the microcontroller simply took way too much time. Therefore, while this design requirement could not be met, it did not prove to be a problem. Rather, a MATLAB® script was written that would read the binary files and convert the data to the proper format. Testing with this method proved to work well, taking no more than a second to convert 5 seconds worth of data versus an unknown amount of time on the microcontroller.

Further testing found that the prototype main unit has a mass of 326 grams. Moreover, a file containing approximately 15 seconds of data is about 340 KB, meaning an 8 GB Micro SD card can store hours of data.

4.6.2 Prototype Testing

Upon completion of the prototype, initial tests were performed to verify the system would work properly when used in the field. One of the initial tests involved riding the bike over a piece of wood 8.26 cm tall by 14 cm wide placed on a flat driveway with no more than 10 cm in elevation change. The vertical acceleration data was recorded using the MTB DAQ and then normalized to account for the zero-g offset and integrated using the MATLAB® cumtrapz function to get the velocity and position. Figure 4.9 depicts the data from the front axle of one of the test trials. As can be seen in Figure 4.9, the position plot shows the effect of the piece of wood by showing a small bump at around 6 seconds. The bump is about 0.08 m tall, indicating that the accelerometer measured the effects of the piece of wood correctly. The overall position plot, however, is not correct, for it shows that by the end of the test the front axle had climbed roughly 3.5 m! This phenomenon is due to small errors in the accelerometer measurements. If the accelerometer values do not cancel out perfectly, then the “leftover” acceleration results in a velocity that in turn results in a change
in position. The effect is amplified through the double integral and only gets worse with time. This is directly shown, for all three plots show consistent values within the first 3 seconds and then show values that start to drift after 3 seconds. The drift gets even worse after the wheel went over the piece of wood. The measured accelerations did not perfectly cancel out and resulted in an elevated velocity that can be seen just after 6 seconds. This velocity, in turn, caused an increase in position over time.

The small acceleration errors shown here are not particular to the ADXL375 accelerometer; rather, they arise with all accelerometers. Consequently, tracking an object via direct integration of acceleration data alone is inherently susceptible to large errors over time. The additional sensors found in an inertial measurement unit
(IMU) do help a little but are still prone to the same errors over long periods of time. The only way to accurately track an object is to fuse the data from an accelerometer or IMU with other independent sensors that can directly measure the desired quantities. For tracking the motion of a mountain bike, the position and velocity could directly be measured with the addition of a GPS unit. This data could then be fused together using an advanced position tracking technique known as a Kalman Filter [21].

Instead of modifying the MTB DAQ with additional sensors and attempting to implement a Kalman Filter, a simpler approach was taken. The errors that cause the drifts in position and velocity occur over a large time scale. Therefore, they can be filtered out using a simple high pass filter. Figure 4.10 depicts the same data from Figure 4.9 after passing the velocity data through a high pass filter with a cutoff frequency of 1 Hz before integrating to find position. The position and velocity in Figure 4.10 provide a better representation of the actual course compared to the position and velocity shown in Figure 4.9. The velocity is now zero after the wheel rides over the piece of wood at a time of about 6 seconds. The effect of the piece of wood is still visible as a small bump about 0.08 m tall at a time of 6 seconds. Moreover, the front axle is not recording a climb to a height of 3.5 m. Rather, it is properly reflecting the level driveway that was ridden on.

The insights gathered here result in important constraints on the capabilities of the MTB DAQ to collect data that is used to track the position and velocity of the front and rear axles. First, the testing sessions must be kept “short.” While no further work was done to quantify an exact length, the phenomenon was noted and kept in mind while interpreting the data from other testing sessions. Second, the tests need to be carried out on relatively level ground in order to reduce errors from a non-constant zero-g offset value. Many features of a trail, such as jumps and high frequency disturbances, can easily be recreated on flat ground. Since the overall goal
Figure 4.10: Data from Figure 4.9 after implementing a high pass filter with a cutoff frequency of 1 Hz on the velocity data before integrating to find position.

is to tune a mountain bike suspension, and not accurately measure the geometry of a trail, this constraint does not pose an issue. Finally, any data interpretation software needs to be able to apply a high pass filter with a cutoff frequency of 1 Hz. As a result of this preliminary testing, the MTB DAQ prototype proved ready for use.

4.7 User Manual

The following section outlines a brief user manual for working with the MTB DAQ.
4.7.1 Flashing the Board

The main board was designed based on the PYBv1.1 schematic. As a result, the firmware and flashing instructions are the same as those for the PyBoard. The main board utilizes the device firmware update (DFU) protocol that comes embedded with each STM32 microcontroller. DFU mode allows for a simple way to update the firmware of an STM32 without requiring specialized hardware. It was mainly designed for updating the firmware remotely on devices that have already been released.

To flash firmware to the board, use the following steps:

1. Make sure that a DFU utility program is installed on the computer that will be used to flash the firmware. “dfu-util” is a free DFU utility program than runs in terminal. Install this program via the package manager.

2. With the power to the board turned off, move the jumper on port JP1 from “JMP STORE” position to the “DFU” position. This will tie the DFU pin of the microcontroller to 3.3V. Figure 4.11 depicts these positions. When the board is powered on, the microcontroller will enter DFU mode upon boot.

3. Connect the board to the computer via USB.

4. Use the dfu-util commands to flash the firmware to the board. For more details on using this software to flash the board, refer to [8].

5. Once the firmware has been loaded on, power off the board and return the jumper on JP1 to the ”JMP STORE” position.

The main board runs off of the standard released PYBv1.1 DFU firmware files available on the Micropython website [11]. At the time of this writing, the most current version that worked with the main board was *pybv11-20191220-v1.12.dfu*. Once the
Figure 4.11: Close up of jumper used to enable DFU mode. (a) depicts the jumper in the STORE position and (b) depicts the jumper in the DFU position.

proper firmware has been loaded onto the board, the main board will appear as a standard USB device when connected to the computer. At this point, the Micropython files outlined in Section 4.5 can be loaded on to the board using a standard method for transferring files. Note that when first loading on the files, make sure that there is no Micro SD card loaded in, as this will appear instead of the USB device representing the microcontroller internal flash memory. It is important to load the files onto the internal flash memory and not the Micro SD card. Once this has been done, the MTB DAQ is ready for operation.

4.7.2 MTB DAQ Operation

For a summary of terms used to describe different components of the MTB DAQ, refer to Figure 4.4.

Before powering on the MTB DAQ, make sure both ADXL375 accelerometers are plugged in to the main unit. The main unit configures the accelerometers upon startup and is not able to re-configure after the power has been turned on. If the accelerometers are not plugged in before the power is turned on, turn off the main
unit, plug them in, and turn it back on. The MTB DAQ is powered on and off via the power switch. Power status is indicated by the power LED (green). Upon power up, the main unit will check for the presence of a Micro SD card. If no Micro SD card is inserted, “SD” will flash continuously on the display. Once a Micro SD card is inserted, “SD” will flash three more times until the main unit detects the card. Once the card has successfully been detected, the display will show the number of the current data file stored on the Micro SD card and the record LED (red) will turn off. If the Micro SD card has just been formatted, the display will show “0” indicating that no data has been logged. The MTB DAQ is now in standby mode and ready to record data.

Data recording is started by pressing down on the record button. After the first press, the record LED (red) will turn on and the display will increment the displayed number indicating that the main unit is recording data. The number that is displayed during and after recording indicates the number of the data file associated with that recording session. Pressing the record button a second time will make the record LED (red) turn off indicating that the main unit is done recording. The display will continue to show the number from that most recent recording session. The main unit is immediately available for another recording session. When testing is finished, it is best to turn the main unit off before removing the Micro SD card.

The main unit is powered by five lithium polymer batteries connected in parallel to make the equivalent of one 2,480 mAh battery. The battery is charged by plugging in the USB port to any powered USB hub. The charge indicator LED (yellow) will turn on indicating that the batteries are charging. Once the batteries are fully charged, the charge indicator LED (yellow) will turn off.

The Micro SD card needs to formatted according to the SD card association. In order to remove old data, it is important to completely erase both the “log” and “count”
folders and all of the contents in them. These folders will be remade if they don’t already exist and the proper files will be generated upon the next power up of the main unit. If only the “log” file is deleted, the system will continue counting based off of the “count.txt” file. The data will still be saved and the proper number will be displayed on the display of the system, but the numbering will not start over as desired. Reformatting the Micro SD card is a good way to ensure that it is completely erased and ready for a new testing session.

4.8 GitHub Repository

In order to assist with further development of the work done here, all of the Micropython files and Autodesk® Eagle™ schematic and board files can be found on the following GitHub repository: https://github.com/swaal/MTB-DAQ.
Chapter 5

RIDER AND TESTING BIKE TECHNICAL SPECIFICATIONS

The bike used for testing is a 2017 Ghost Kato FS 3. The bike was chosen for the simple reason that it is a full suspension bike that was already in the author’s possession. Figure 5.1 depicts this bike with the MTB DAQ prototype installed.

Figure 5.1: 2017 Ghost Kato FS 3 equipped with the MTB DAQ.

5.1 Rider Physical Characteristics

The physical characteristics of the rider (the author) are as important to the dynamic model as those of the bike, for the height and weight of the rider greatly affect the location of the center of gravity (CG) and the pitching moment of inertia of the bike and rider system. Therefore, the relevant characteristics of the rider are summarized in Table 5.1. While the values for rider height and rider inseam length are not directly
Table 5.1: Rider physical characteristics important to the dynamic model.

<table>
<thead>
<tr>
<th>Characteristic</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Height</td>
<td>192 cm</td>
</tr>
<tr>
<td>Inseam length</td>
<td>95 cm</td>
</tr>
<tr>
<td>Mass</td>
<td>84.4 kg</td>
</tr>
</tbody>
</table>

used, they are provided to give the reader a better idea of the build of the rider for ease of comparison with other riders.

5.2 Geometry and Components

The 2017 Ghost Kato FS 3 is an entry level bike in the world of full-suspension mountain bikes. As such, it has decent components and fairly standard frame geometry. Table 5.2 lists the main components of the bike and Table 5.3 lists the geometry and mass of the bike.

Table 5.2: 2017 Ghost Kato FS 3 Components

<table>
<thead>
<tr>
<th>Component</th>
<th>Specification</th>
</tr>
</thead>
<tbody>
<tr>
<td>Front Fork</td>
<td>Rockshox Recon TK Silver 27.5&quot;</td>
</tr>
<tr>
<td>Rear Shock</td>
<td>X-Fusion O2 Pro RL</td>
</tr>
<tr>
<td>Drivetrain</td>
<td>Shimano XT</td>
</tr>
<tr>
<td>Brakes</td>
<td>Shimano 506 Disc</td>
</tr>
</tbody>
</table>

5.3 Dynamic Properties

As can be seen in Section 7, the location of the CG as well as the pitching moment of inertia of the bike and rider are important parameters for the model. Consequently, various methods were used to measure these quantities.
Table 5.3: 2017 Ghost Kato FS 3 Frame Geometry and Mass

<table>
<thead>
<tr>
<th>Dimension</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Seat tube length</td>
<td>546 mm</td>
</tr>
<tr>
<td>Top tube length (horizontal)</td>
<td>648 mm</td>
</tr>
<tr>
<td>Head tube length</td>
<td>143 mm</td>
</tr>
<tr>
<td>Head tube angle</td>
<td>67°</td>
</tr>
<tr>
<td>Seat tube angle (effective)</td>
<td>73°</td>
</tr>
<tr>
<td>Chain stay length (horizontal)</td>
<td>432 mm</td>
</tr>
<tr>
<td>Wheelbase</td>
<td>1,168 mm</td>
</tr>
<tr>
<td>Stack</td>
<td>610 mm</td>
</tr>
<tr>
<td>Reach</td>
<td>495 mm</td>
</tr>
<tr>
<td>Bottom bracket offset</td>
<td>25 mm</td>
</tr>
<tr>
<td>Rear-center</td>
<td>432 mm</td>
</tr>
<tr>
<td>Front-center</td>
<td>737 mm</td>
</tr>
<tr>
<td>Rim diameter</td>
<td>584 mm</td>
</tr>
<tr>
<td>Wheel diameter</td>
<td>699 mm</td>
</tr>
<tr>
<td>Wheel radius</td>
<td>349 mm</td>
</tr>
<tr>
<td>Bottom bracket height</td>
<td>330 mm</td>
</tr>
<tr>
<td>Crank length</td>
<td>171 mm</td>
</tr>
<tr>
<td>Stem length</td>
<td>86 mm</td>
</tr>
<tr>
<td>Fork offset</td>
<td>35 mm</td>
</tr>
<tr>
<td>Mechanical trail</td>
<td>118 mm</td>
</tr>
<tr>
<td>Trail</td>
<td>105 mm</td>
</tr>
<tr>
<td>Mass</td>
<td>15 kg</td>
</tr>
</tbody>
</table>

5.3.1 Center of Gravity

Measurements for the CG of the bike and rider were carried out according to the procedure outlined in the Cal Poly course ME 441: Single Track Vehicle Design. In order to calculate the fore/aft position of the CG, the bike and rider are placed on two scales set on the ground (one under each tire of the bike). Figure 5.2 depicts this set up. In Figure 5.2, note that $\mathbf{F}_R$ and $\mathbf{F}_F$ are simply the values reported by the front and rear scales, respectively. $\mathbf{F}_{mg}$ is the total weight of the bike and rider, $L_1$ is the distance of the CG from the rear axle, and $L_2$ is the wheelbase. Summing
moments about the contact point between the rear wheel and the scale gives

\[(F_F)(L_2) - (F_{mg})(L_1) = 0 \quad (5.1)\]

Solving for \(L_1\) gives

\[L_1 = \frac{(F_F)(L_2)}{F_{mg}} \quad (5.2)\]

Thus Equation 5.2 gives the horizontal distance of the CG from the rear axle. Table 5.4 summarizes the results.

In order to calculate the vertical distance of the CG, a similar method was used. This time, however, the front scale was raised a slight distance off the ground. Figure 5.3 depicts the testing setup. Similar to Figure 5.2, \(\vec{F}_R\) and \(\vec{F}_F\) are the values reported by the front and rear scales, respectively. \(\vec{F}_{mg}\) is the total weight of the bike and
rider and $L_1$ is the distance of the CG from the rear axle. $X$ is the distance between the scales (note that it is now shorter than the wheelbase), $Y$ is the height of the front scale off of the ground, $H$ is the vertical height of the CG above the rear axle, $\theta$ is the the angle of the bike and rider with respect to the ground, and $D$ is the diameter of the wheels. Since the Ghost Kato FS 3 has the same diameter wheels in the front and the back, the analysis was carried out with this assumption. For bikes with different sized front and rear wheels, slight modifications to the formulas would need to be made. From geometry it can be shown that

$$\theta = \tan^{-1}\left(\frac{Y}{X}\right)$$

(5.3)

Summing moments about the contact point of the rear wheel and the scale gives

$$(F_F)(X) - (F_{mg})[L_1 \cos(\theta) - H \sin(\theta)] = 0$$

(5.4)

Solving for $H$ gives

$$H = \frac{(F_{mg})(L_1) \cos(\theta) - (F_F)(X)}{F_{mg} \sin(\theta)}$$

(5.5)

Recall that $H$ is the vertical height of the CG above the rear axle. To find the height of the CG from the ground, $H_{CG}$, simply add half of the wheel diameter:

$$H_{CG} = H + \frac{D}{2}$$

(5.6)

A total of five trials were conducted, with the height of the front wheel being changed between each trial, and the results were averaged. Table 5.5 summarizes the results.
Figure 5.3: Schematic for the setup used to calculate the vertical position of the CG of the bike and rider.

5.3.2 Pitching Moment of Inertia

The moment of inertia of a body about a desired axis can be found by swinging the body about any axis parallel to the desired axis. The period of oscillations can then be correlated to the moment of inertia of the body. Figure 5.4 depicts the diagram for an arbitrary body swinging about an arbitrary fixed axis, $A$. Summing moments about axis $A$ gives

$$\Sigma \vec{M} = I_A \ddot{\theta} \quad (5.7)$$

$$\left( I_A \right) \ddot{\theta} + (F_{mg})(r) \sin(\theta) = 0 \quad (5.8)$$

Assuming $\theta$ is small, then $\sin(\theta) \approx \theta$. Therefore,

$$\left( I_A \right) \ddot{\theta} + (F_{mg})(r)\theta = 0 \quad (5.9)$$
Table 5.5: Measurements used to calculate the vertical position of the CG of the bike and rider.

<table>
<thead>
<tr>
<th>Constants</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Total Weight of Bike and Rider, $F_{mg}$</td>
<td>974 N</td>
</tr>
<tr>
<td>Distance of CG From Rear Axle, $L_1$</td>
<td>43.2 cm</td>
</tr>
<tr>
<td>Wheel Diameter, $D$</td>
<td>69.9 cm</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Trial</th>
<th>Height of Front Scale</th>
<th>Distance Between Scales</th>
<th>Angle</th>
<th>Front Scale Reading</th>
<th>Vertical height of CG above rear axle</th>
<th>Vertical height of CG above ground</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$Y$ [cm]</td>
<td>$X$ [cm]</td>
<td>$\theta$ [deg.]</td>
<td>$F_F$ [N]</td>
<td>$H$ [cm]</td>
<td>$H_{CG}$ [cm]</td>
</tr>
<tr>
<td>1</td>
<td>33.5</td>
<td>111.9</td>
<td>16.7</td>
<td>146.8</td>
<td>85.5</td>
<td>120.4</td>
</tr>
<tr>
<td>2</td>
<td>22.5</td>
<td>114.7</td>
<td>11.1</td>
<td>204.6</td>
<td>95.1</td>
<td>130.0</td>
</tr>
<tr>
<td>3</td>
<td>19</td>
<td>115.3</td>
<td>9.4</td>
<td>235.7</td>
<td>90.5</td>
<td>125.5</td>
</tr>
<tr>
<td>4</td>
<td>39</td>
<td>110.1</td>
<td>19.5</td>
<td>111.2</td>
<td>84.3</td>
<td>119.3</td>
</tr>
<tr>
<td>5</td>
<td>45.5</td>
<td>107.6</td>
<td>22.9</td>
<td>80.1</td>
<td>79.5</td>
<td>114.4</td>
</tr>
</tbody>
</table>

Average, $H_{CG}$: 121.9

The solution to this second order, linear differential equation is

$$\theta(t) = C_1 \sin \left( \sqrt{\frac{F_{mg}r}{I_A}} t \right) + C_2 \cos \left( \sqrt{\frac{F_{mg}r}{I_A}} t \right)$$

(5.10)

Thus the system oscillates with a natural frequency, $\omega_n$ of

$$\omega_n = \sqrt{\frac{F_{mg}r}{I_A}}$$

(5.11)

Substituting in $\omega_n = \frac{2\pi}{T}$, where $T$ is the period of oscillation, and solving for $I_A$ gives

$$I_A = \frac{T^2 F_{mg}r}{4\pi^2}$$

(5.12)

Substitute in $F_{mg} = mg$ to simplify

$$I_A = \frac{T^2 mgr}{4\pi^2}$$

(5.13)

Recall that this solves for the moment of inertia of the body about the fixed axis $A$.

In order to get the moment of inertia of the body about the CG, use the parallel axis theorem to get

$$I_{CG} = I_A + mr^2$$

(5.14)
Equation 5.15 provides a method for calculating the moment of inertia about the CG of a body when swung from a fixed axis parallel to the desired axis.

Using this theory, the pitching moment of inertia of the bike and rider was measured. The bike and rider were hung from a large beam and lightly pushed. The time to complete ten oscillations was measured and recorded. From this, the period of oscillation was calculated and then used to calculate the pitching moment of inertia. Figure 5.5 depicts the testing setup that was used. A total of five trials were conducted and the results were averaged. Table 5.6 summarizes these results.
Figure 5.5: Testing setup used to measure the pitching moment of inertia of the bike and rider.

Table 5.6: Measurements used to calculate the pitching moment of inertia of the bike and rider.

<table>
<thead>
<tr>
<th>Trial</th>
<th>Number of Oscillations</th>
<th>Total Time</th>
<th>Period</th>
<th>Moment of Inertia About Rotation Point</th>
<th>Moment of Inertia About CG of Bike/Rider</th>
</tr>
</thead>
<tbody>
<tr>
<td>[-]</td>
<td>n  [-]</td>
<td>t [sec]</td>
<td>T [sec]</td>
<td>I_A [kg*m^2]</td>
<td>I_CG [kg*m^2]</td>
</tr>
<tr>
<td>1</td>
<td>10</td>
<td>23.9</td>
<td>2.39</td>
<td>152</td>
<td>36</td>
</tr>
<tr>
<td>2</td>
<td>10</td>
<td>23.88</td>
<td>2.39</td>
<td>152</td>
<td>36</td>
</tr>
<tr>
<td>3</td>
<td>10</td>
<td>23.91</td>
<td>2.39</td>
<td>152</td>
<td>36</td>
</tr>
<tr>
<td>4</td>
<td>10</td>
<td>23.78</td>
<td>2.38</td>
<td>151</td>
<td>35</td>
</tr>
<tr>
<td>5</td>
<td>10</td>
<td>23.93</td>
<td>2.39</td>
<td>153</td>
<td>37</td>
</tr>
</tbody>
</table>

Average, $I_{CG}$ 36
The Ghost Kato FS 3 is equipped with a Rockshox Recon TK Silver 27.5” fork and an X-Fusion O2 Pro RL rear shock. The spring rate on each component can be adjusted by increasing or decreasing the static air pressure in the shock with a shock pump via a small Schrader valve. The damping rate can be adjusted via a small dial. Note that it is common to hear mountain bike shock manufacturers refer to the damping adjustment as the “rebound adjustment.” The dials are retained in place by a small detent and offer a set number of fixed positions. Adjustment from one setting to the next results in small “clicks” that can easily be felt by the user. As a result, adjustments made on the dampers of each shock will be quantitatively described as a number of “clicks” referenced from the slowest rebound setting (largest damping rate) and going to the fastest. Most shock manufacturers display these settings by depicting a tortoise and a jackalope\(^1\) on the shock. Moving the dial towards the tortoise slows down the rebound rate and moving it towards the jackalope speeds it up. Therefore, the term “clicks from tortoise” will be used to quickly state the sign convention for adjusting the damping rate from here forward. Figure 6.1 depicts the damping rate adjustment dials on each shock and their corresponding “clicks from tortoise” reference points. Note that the damping rate adjustment dial on the X-Fusion O2 Pro RL is the small red circular dial (the blue lever-arm in front controls the shock lock-out).

\(^1\)Refer to Appendix I for more information on the jackalope.
of each component is bounded. This information is important, for it bounds the possible solutions that can be found by the genetic algorithm as described in Section 7. Table 6.1 summarizes the bounds on each adjustment of each shock component, listed in terms of the user-adjustable parameters of static air pressure and “clicks from tortoise,” as well as the corresponding maximum travel of each shock. In order to be of any use to the model, it was necessary to relate these user-adjustable parameters to actual spring rates and damping rates. In other words, it was necessary to define the spring curve as a function of static air pressure and damping rate as a function of “clicks from tortoise.” It is important to note that initial characterization of the shocks was carried out by using an Instron Model 1331 available in the Cal Poly composites lab. Custom fixturing was built to hold the shocks on the machine and force versus displacement data was recorded. By using the various settings of the machine, different parameters such as displacement increment and displacement velocity could be controlled. By choosing the right settings, the data recorded from this machine could be used to compute both the spring rate and the damping rate of the shocks. While validating the model discussed in Section 7.2, however, it was found that the
Table 6.1: Bounding values on the spring rate, damping rate, and travel of the fork and rear shock of the 2017 Ghost Kato FS 3.

<table>
<thead>
<tr>
<th>Component</th>
<th>Recommended Shock Static Air Pressure [psi]</th>
<th>Damping Rate Adjustment Dial [clicks from tortoise]</th>
<th>Travel [mm]</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rockshox Recon TK Silver 27.5&quot; Fork</td>
<td>Min. 50</td>
<td>Max. 120</td>
<td>Min. 0</td>
</tr>
<tr>
<td>X-Fusion O2 Pro RL Shock</td>
<td>Min. 60</td>
<td>Max. 300</td>
<td>Min. 0</td>
</tr>
</tbody>
</table>

calculated spring rates were too small and the predicted damping rates were too large. The spring rate discrepancy was largely a result of not enough data being collected over the full travel of the shock. The damping rate discrepancy, however, was more subtle, and the best guess as the source of error is the relatively slow displacement velocities used to compress the shocks. As a result, other methods were required to more accurately characterize the shocks. The following sections outline how this was accomplished.

6.1 Wheel to Shock Adjustments

As will be shown in Section 7, the development of a “conversion” between the force and displacement in the front and rear wheels and the equivalent force and displacement at the front and rear shocks was important to develop. The development of these conversions factors is summarized in the following sections. The derivation will show that the conversion factors for both force and displacement are the same for their respective wheel and shock combination. In other words, the conversion factor for the travel of the rear wheel to the travel of the rear shock is the same as the
conversion factor for the force in the rear wheel to the force in the rear shock. This is the same for the front wheel and shock. Consequently, these conversion, or scale factors will be denoted $SF_F$ and $SF_R$ for front and rear, respectively, and will be used to convert either force/displacement at the wheel to force/displacement at the corresponding shock.

6.1.1 Front Wheel to Fork

The adjustment between the vertical force acting on the wheel and the force in the fork is a simple matter of geometry. Figure 6.2 depicts the free body diagram of the front wheel and the fork. Note that since the wheel is attached to the fork with an axle, no moments can be translated. Summing the forces in the $y$ direction gives

$$F_1 = F_2 \sin(\theta) \quad (6.1)$$

where $\theta$ is the head tube angle. This gives a conversion between the vertical force at the wheel, $F_1$ and the force in the fork, $F_2$. It is important to note that if the force
acting on the wheel was coming from a different direction (for example, if the front wheel ran into a large rock) then the force would scale differently between the wheel and the fork. To keep the model simple, however, only the vertical components of the force are considered. Looking up the value for the head tube angle in Table 5.3, the scale factor for the Rockshox Recon TK Silver 27.5” fork, $SF_F$, can be calculated as

$$SF_F = \sin(67^\circ) \approx 0.9205 \quad (6.2)$$

Note that the vertical travel of the front wheel is related to the travel of the fork through the same scale factor and is constant throughout the entire travel. Recalling the maximum travel of the fork to be 130 mm (refer to Table 6.1), the equivalent maximum travel of the front wheel, $L_F$, is given by

$$L_F = (130 \text{ mm})(SF_F) = 119.7 \text{ mm} \quad (6.3)$$

### 6.1.2 Rear Wheel to Shock

The rear shock is connected to the rear wheel via a four-bar “Horst-link” linkage system [23]. In order to properly characterize the motion ratio and force transmission between the rear wheel and the rear shock throughout the entire travel of the rear wheel, an in-depth analysis of the linkage was carried out. Figure 6.3 depicts the schematic used for the analysis overlaid on a picture of the rear linkage. Note in Figure 6.3 that the links are numbered. The angle of each link ($\theta$, $\alpha$, and $\beta$) with respect to the horizontal is also defined. $\vec{F}_S$ denotes the force of the shock acting on the linkage and $\vec{F}_R$ denotes the force of the rear axle acting on the linkage. Note that for simplicity both forces are assumed to always be applied perfectly vertical. The dimensions of each link were then defined according to Figure 6.4 and the values are summarized in Table 6.2. The linkage was then modeled in Autodesk® Fusion 360® in order to obtain an accurate measurement of the maximum range of motion possible.
Figure 6.3: Schematic used for the four-bar linkage analysis overlaid on a picture of the rear linkage.

Figure 6.4: Schematic denoting how each dimension is defined for the schematic shown in Figure 6.3.
Table 6.2: Values for the dimension shown in Figure 6.4.

<table>
<thead>
<tr>
<th>Dimension</th>
<th>Value [cm]</th>
</tr>
</thead>
<tbody>
<tr>
<td>$L_1$</td>
<td>38.5</td>
</tr>
<tr>
<td>$L_2$</td>
<td>34.0</td>
</tr>
<tr>
<td>$L_3$</td>
<td>16.5</td>
</tr>
<tr>
<td>$L_4$</td>
<td>18.5</td>
</tr>
<tr>
<td>$A$</td>
<td>11.0</td>
</tr>
<tr>
<td>$B$</td>
<td>6.5</td>
</tr>
<tr>
<td>$D$</td>
<td>4.2</td>
</tr>
<tr>
<td>$E$</td>
<td>36.5</td>
</tr>
</tbody>
</table>

Figure 6.5 depicts the model and the corresponding measurements. Note that the maximum range of motion of the rear linkage is limited by the maximum travel of the shock. As shown in Table 6.1, the maximum travel for the X-Fusion O2 Pro RL is 40.5 mm. This value was inputted into the model and the corresponding motion of the rear axle and angle of Link 1 were measured. As can be seen in Figure 6.5, the corresponding motion of the rear axle is 95.2 mm and the corresponding maximum angle of Link 1 ($\theta$) is $13.6^\circ$. Therefore, the equivalent maximum travel of the rear wheel, $L_R$, is given by

$$L_R = 95.2 \text{ mm} \quad (6.4)$$

With this, the precise motion of the linkage was known over the entire range of travel.

With the geometry of the linkage defined, the MATLAB® script `RearTriangleGeometry_Mastermind.m` was written to calculate the motion. The equations for the linkage are given by

$$L_1 \cos(\theta) - L_2 \cos(\alpha) - A \cos(\beta) = 0 \quad (6.5)$$

$$L_1 \sin(\theta) + L_2 \sin(\alpha) - A \sin(\beta) - L_4 = 0 \quad (6.6)$$

These equations were solved with the MATLAB® `fsolve` non-linear solver for $\alpha$ and $\beta$ on the interval $0 \leq \theta \leq 13.6^\circ$. Figure 6.6 depicts a plot of the angle $\beta$ of Link 3 compared to the angle of the input angle $\theta$ of Link 1. As can be seen in Figure 6.6,
Figure 6.5: Autodesk® Fusion 360® model of the rear linkage used to determine the maximum travel.

Figure 6.6: Comparison of the input angle $\theta$ to the output angle $\beta$ of the rear linkage. The relationship between these two angles shows that the rear linkage is linear.
the relationship is linear, meaning that the motion ratio between the rear axle and the shock is constant throughout the entire travel. In general, mountain bike linkages are designed to either be progressive, linear, or regressive. In terms of Figure 6.6, a progressive linkage design would be seen by a concave up increasing curve. A regressive linkage design would be seen by a concave down increasing curve. Since the linkage is linear, the motion ratio throughout the entire travel is equal to the ratio of the maximum travel of the rear axle to the maximum travel of the shock, which is given by

$$\frac{95.2\text{mm}}{40.5\text{mm}} = 2.35 \quad \text{(6.7)}$$

Once the motion of the linkage was completely defined, the force transmission between the rear wheel and the shock through the linkage could be solved for. A free body diagram for each link (Links 1, 2, and 3) was drawn and the equilibrium equations were written out in terms of the linkage dimensions and position. Figure 6.7 depicts the free body diagrams. The equations of equilibrium can be written out as the following system of equations

![Free body diagrams of Links 1, 2, and 3.](image)

Figure 6.7: Free body diagrams of Links 1, 2, and 3.
\[
\begin{bmatrix}
F_R & F_S & F_1 & F_2 & F_3 & F_4 & F_5 & F_6 & F_7 & F_8 \\
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & -1 & 0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & -1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & C_1 & -C_2 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & -1 & 0 & 1 & 0 & 0 & 0 \\
1 & 0 & 0 & 0 & -1 & 0 & -1 & 0 & 0 & 0 \\
C_3 & 0 & 0 & 0 & 0 & 0 & C_4 & -C_5 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & -1 & 0 \\
0 & -C_8 & 0 & 0 & 0 & 0 & C_6 & C_7 & 0 & 0 \\
\end{bmatrix} = \begin{bmatrix} F_A \end{bmatrix}
\]

where \( F_A \) is a placeholder for the applied force at the rear axle, \( F_R \), and the coefficients \( C_1 \) through \( C_8 \) are defined as follows:

\[
C_1 = L_1 \cos(\theta) \quad \text{(6.8)}
\]

\[
C_2 = L_1 \sin(\theta) \quad \text{(6.9)}
\]

\[
C_3 = D \cos(\delta) \quad \text{(6.10)}
\]

\[
C_4 = L_2 \cos(\alpha) \quad \text{(6.11)}
\]

\[
C_5 = L_2 \sin(\alpha) \quad \text{(6.12)}
\]

\[
C_6 = A \cos(\beta) \quad \text{(6.13)}
\]

\[
C_7 = A \sin(\beta) \quad \text{(6.14)}
\]

\[
C_8 = B \cos(\phi) \quad \text{(6.15)}
\]

The system of equations was solved in \textit{RearTriangleGeometry}\_\textit{Mastermind.m} for the entire range of motion of the linkage. The script set \( F_A = 1 \text{ N} \). Figure 6.8 depicts...
a plot of $F_R$ and $F_S$ over the entire range of motion of the linkage ($0 \leq \theta \leq 13.6^\circ$). As can be seen in Figure 6.8, a constant force of 1 N at the rear wheel results in a constant equivalent force of 2.4121 N at the rear shock over the entire range of travel. As mentioned previously, this confirms that the linkage is indeed linear (instead of progressive or regressive). Further, this means that the scale factor to convert from the force at the rear wheel to the force at the rear shock, $SF_R$, is given by

$$SF_R = 2.4121 \quad (6.16)$$

It is important to note that most bicycle manufacturers estimate this scale factor by simply taking the ratio of the maximum travel of the shock to that of the rear
axle. This estimate was given in Equation 6.7 and differs from the value given in Equation 6.16 by about 9%. Since the value given in Equation 6.16 was found using a more rigorous method, it is the value that is used in the model.

6.1.3 Summary

The force scale factors and equivalent travel lengths found in this section for the front and rear wheels are very important to the development of the model in Section 7. Therefore, the results from this section are summarized in Table 6.3.

Table 6.3: Force scale factors and equivalent travel lengths for the front and rear wheels.

<table>
<thead>
<tr>
<th>Variable</th>
<th>Maximum Travel</th>
<th>Scale Factor (from wheel to shock)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Value [mm]</td>
<td></td>
</tr>
<tr>
<td>Front Wheel</td>
<td>$L_F$</td>
<td>$S_{F_F}$ 0.9205</td>
</tr>
<tr>
<td>Rear Wheel</td>
<td>$L_R$</td>
<td>$S_{F_R}$ 2.4121</td>
</tr>
</tbody>
</table>

6.2 Air Spring Characterization

The air spring characterization involved deriving a theoretical equation that describes the force versus displacement curve of an air shock, collecting force versus displacement data for each shock, and then curve fitting the theoretical model to the data. Section 6.2.1 describes the derivation of the theoretical equation and Section 6.2.2 describes the data collection.
6.2.1 Air Shock Theory

Figure 6.9 depicts a diagram of a typical air shock. As discussed in Section 1.2, the shock in the diagram has a positive air chamber with length $L_1$ and initial static pressure $P_1$ as well as a negative air chamber with length $L_2$ and initial static pressure $P_2$. Assuming the shock is static, summing the forces on the piston gives

\[ F_3 = F_1 - F_2 \]  

(6.17)

By defining the pressure in the positive and negative air chambers after the piston has moved a distance $x$ from the initial position as $P_{1x}$ and $P_{2x}$, respectively, Equation 6.17 can be written as

\[ F_3 = (P_{1x}A_1) - (P_{2x}A_2) \]  

(6.18)

where $A_1$ is the effective area of the piston on the positive air chamber side and $A_2$ is the effective area of the piston on the negative air chamber side. Assuming the gas can be modeled as an ideal gas and that the process is isentropic, it is possible to calculate the pressure in each chamber after the piston moves a distance $x$ beyond
the initial position as a function of the initial pressures $P_1$ and $P_2$ and $x$. For an isentropic process in a closed system, the ideal gas law gives

$$P_A V_A = P_B V_B$$

(6.19)

where $P_A$ and $V_A$ are the initial pressure and volume and $P_B$ and $V_B$ are the final pressure and volume. Thus the pressure in the positive air chamber after moving the piston a distance $x$ from rest is given by

$$P_{1x} = P_1 \left( \frac{V_1}{V_{1x}} \right)$$

(6.20)

where $V_{1x}$ is the volume in the positive air chamber corresponding to the distance traveled. This can be written as

$$P_{1x} = P_1 \left( \frac{A_1 L_1}{A_1 (L_1 - x)} \right)$$

(6.21)

$$P_{1x} = P_1 \left( \frac{L_1}{L_1 - x} \right)$$

(6.22)

Applying the same analysis to the negative air chamber gives

$$P_{2x} = P_2 \left( \frac{L_2}{L_2 + x} \right)$$

(6.23)

Note that the effect of the moving the piston in positive $x$ makes the positive air chamber smaller ($L_1 - x$) and the negative air chamber bigger ($L_2 + x$). Plugging in Equations 6.22 and 6.23 into Equation 6.18 gives

$$F_3 = \left[ P_1 \left( \frac{L_1}{L_1 - x} \right) \right] A_1 - \left[ P_2 \left( \frac{L_2}{L_2 + x} \right) \right] A_2$$

(6.24)

The first term in Equation 6.24 captures the effect of the positive air spring and the second term captures the effect of the negative air spring. Figure 6.10 depicts a plot of Equation 6.24, showing the effect of the positive air spring, negative air spring, and the combined result. It can then be shown that Equation 6.24 can be written as

$$F_3 = \left[ \frac{P_1 L_1 A_1 + P_2 L_2 A_2}{x^2 + (L_1 - L_2)x + L_2 L_1} \right] x + \left[ \frac{L_1 L_2 (P_1 A_1 - P_2 A_2)}{x^2 + (L_1 - L_2)x + L_2 L_1} \right]$$

(6.25)

While not strictly necessary, Equation 6.25 was used to define the custom curve fit used in MATLAB®.
Figure 6.10: Theoretical air spring force curve as described by Equation 6.24.

6.2.2 Air Spring Curve Measurements

Measuring the actual air spring curve for the front and rear shock involved creating a testing apparatus capable of loading the bike to several hundred pounds. Similar to the MTB DAQ, the main goal was to keep the testing apparatus simple and low cost. The testing setup consisted of the following. A small homemade bike stand held the front or rear wheel and a rope tied to the handlebars kept the front straight. Two ratchet-straps tied in parallel to either the bottom bracket or head tube of the bike were used for the loading and a bathroom scale placed under the front or rear wheel was used to record the applied force. Figure 6.11 depicts the testing setup. The shock to be tested was pumped up to the desired pressure and
the initial scale reading (with the ratchet straps loose) was recorded. The testing procedure then involved tightening the ratchet straps until the scale reading increased by about 20 lbf. The compression of the shock was measured using dial calipers and this value as well as the scale reading were recorded. This was repeated until either the maximum travel was used up or the maximum scale loading was reached. The force at the scale was then converted to the force at the shock using the scale factors summarized in Table 6.3. The fork and the rear shock were each tested for a variety of pressure settings. Table E.1 and E.2 in Appendix E contain the raw and processed data for each test. The data from Tables E.1 and E.2 were plotted and curve fit using two MATLAB® scripts, RockshoxReconTKSilver_ShockCurve.m and X_Fusion_O2_Pro_RL_ShockCurve.m. The scripts computed the curve fit using a custom fit as described by Equation 6.25. Figure 6.12 depicts these plots. Recall that Equation 6.25 defines the air spring curve based on the initial pressures in the positive and negative air spring chambers ($P_1$ and $P_2$) as well as the geometry of the shock itself ($L_1$, $L_2$, $A_1$, and $A_2$). This means that the curve fits for the data taken at different initial static pressures ($P_1$) as shown in Figure 6.12 should theoretically predict the same values for $L_1$, $L_2$, $A_1$, and $A_2$. In other words, the various curves should only differ by the values of $P_1$ and $P_2$. Further, due to the geometry of the shock, the value of $P_2$ should be dependent on $P_1$. Consequently, it becomes clear...
that for both shocks, it should be possible to define the air spring curve based on a modified version of Equation 6.25, where $L_1$, $L_2$, $A_1$, and $A_2$ are predefined, constant values that represent physical dimensions of the shock, $P_2$ is a function of $P_1$, and $P_1$ is given. The two MATLAB® scripts, RockshoxReconTKSilver_ShockCurve.m and X_Fusion_O2_Pro_RL_ShockCurve.m, were adapted to find this modified version of Equation 6.25. To start, the values for $L_1$, $L_2$, $A_1$, and $A_2$ from each individual curve fit were averaged together to estimate the true, constant values for each shock. Next, the relationship between $P_1$ and $P_2$ was assumed to be given by

$$P_1 = C_1 P_2$$

(6.26)

Similar to before, the value for $C_1$ was calculated from each individual curve fit and averaged together for each shock. Substituting the relationship between $P_1$ and $P_2$ given in Equation 6.26 into Equation 6.25 gives

$$F_3 = \left[ P_1 L_1 A_1 + \left( \frac{P_1}{C_1} \right) L_2 A_2 \right] x + \left[ L_1 L_2 (P_1 A_1 - \left( \frac{P_1}{C_1} \right) A_2) \right] - x^2 + (L_1 - L_2)x + L_2 L_1$$

(6.27)
Table 6.4 summarizes the calculated values for $L_1$, $L_2$, $A_1$, $A_2$, and $C_1$ for each shock. By substituting the values from Table 6.4 into Equation 6.27, the air spring curve for each shock for any static pressure $P_1$ can be found. Figure 6.13 depicts the general air spring curves given by Equation 6.27 overlaid on the plots in Figure 6.12 for each static pressure.

Table 6.4: Calculated values for $L_1$, $L_2$, $A_1$, $A_2$, and $C_1$ for each shock.

<table>
<thead>
<tr>
<th>Variable</th>
<th>Units</th>
<th>Rockshox Recon TK Silver 27.5&quot;</th>
<th>X-Fusion O2 Pro RL</th>
</tr>
</thead>
<tbody>
<tr>
<td>$L_1$</td>
<td>[m]</td>
<td>0.1552</td>
<td>0.0689</td>
</tr>
<tr>
<td>$L_2$</td>
<td>[m]</td>
<td>0.0183</td>
<td>0.0041</td>
</tr>
<tr>
<td>$A_1$</td>
<td>[m$^2$]</td>
<td>6.7569E-4</td>
<td>0.0016</td>
</tr>
<tr>
<td>$A_2$</td>
<td>[m$^2$]</td>
<td>1.5987E-4</td>
<td>4.1414E-4</td>
</tr>
<tr>
<td>$C_1$</td>
<td>[-]</td>
<td>0.3005</td>
<td>0.3222</td>
</tr>
</tbody>
</table>

Figure 6.13: Overlay of general spring curves given by Equation 6.27 and the plots from Figure 6.12.
The shaded region in the plots indicate an error band around the original curve fit in order to see how well the general curves match. As can be seen in each plot, the general curve overestimates the higher pressures, underestimates the lower pressures, and almost exactly predicts the middle pressures. This is likely a result of the averaging used when computing the values given in Table 6.4. From this, the general air spring curves were plotted for each shock in set increments from the lowest to highest recommended pressures as given in Table 6.1. Figure 6.14 depicts these plots.

(a) Rockshox Recon TK Silver 27.5”  
(b) X-Fusion O2 Pro RL

Figure 6.14: General air spring curve for each shock over the entire range of recommended pressures.

The plots in Figure 6.14 give a complete description of the behavior of the shocks for any static pressure $P_1$, providing an easy way to interface with the model.
6.3 Damping Rate Characterization

As mentioned previously, an initial attempt to characterize the damping rate of
the shocks was made through the data collected from the Instron machine testing.
Since the results obtained from this test were found to be unsatisfactory, a differ-
ent more theoretical methodology was employed. While the results from this section
are based on large assumptions, the relatively good results of the model discussed in
Sections 7 and 8 indicate that they are valid for the intended purposes.

The equation of motion for an unforced damped harmonic oscillator with mass $m$,
damping rate $c$, and spring constant $k$, is given by

$$m\ddot{x} + c\dot{x} + kx = 0 \quad (6.28)$$

with $m > 0$, $c \geq 0$, and $k > 0$ [4]. The characteristic equation is then given by

$$ms^2 + cs + k = 0 \quad (6.29)$$

with characteristic roots

$$s = \frac{-c \pm \sqrt{c^2 - 4mk}}{2m} \quad (6.30)$$

The system is said to be critically damped when the damping rate $c$, is chosen such
that $c^2 = 4mk$. Of all the possible damping rates, this value for $c$ results in the fastest
decay [4]. Thus the damping rate for a critically damped system is given by

$$c = \sqrt{4mk} \quad (6.31)$$

By assuming that the damping rate settings for the shocks are designed to achieve
critical damping for a variety of different rider weights, Equation 6.31 can be used
to calculate the damping rate as a function of clicks from tortoise. Consequently,
the following methodology was used. There is a sticker on the Rockshox Recon TK Silver 27.5” fork that gives recommended static pressure values based on rider mass.
The sticker groups the possible rider mass values into five categories and gives corresponding static air pressure settings. The columns labeled “Rockshox Recon sticker recommendations” of Table 6.5 summarize these categories. Recall from Table 6.1 that the Rockshox Recon TK Silver 27.5” fork has a total of five damping rate adjustments (0 - 4 clicks from tortoise). Therefore, it was assumed that each damping rate position provides the critically damped condition for a corresponding rider mass and static air pressure (spring rate) as described by Equation 6.31. The damping rate estimation calculations were carried out in the MATLAB® script dampingRateEstimate.m. The script chose a single rider mass and pressure from the table by dividing up the difference between the maximum and minimum weight (as well as pressure) into equal parts. The column labeled “Single Value” in Table 6.5 depicts the mass value selections. Note that there is a slight discrepancy in the mass values of this column compared to the maximum and minimum values due to the fact that the calculations were done in lbf and then converted to kg. These mass values were then adjusted according to the following equation

\[ m = (m_{\text{single value}} + m_{\text{bike}}) \left( \frac{b}{a+b} \right) \]  

(6.32)
where $b$ is the horizontal distance of the CG from the rear axle and $a$ is the horizontal distance of the CG from the front axle. Equation 6.32 gives the total mass of the bike and rider acting on the front axle at static equilibrium on level ground. These values were calculated for each rider mass category and are given in the column labeled “Adjusted” in Table 6.5. A similar adjustment was carried out for the pressures. The single pressure values (shown in the column labeled “Pressure” of Table 6.5) are simply an even division of pressures between the minimum and maximum recommended pressures. The air spring curve corresponding to each of these pressures was plotted using the results from Section 6.2.2 and Equation 6.27. An “equivalent” linear spring constant was computed for each air spring curve by taking the average of the derivative of the air spring curve. Figure 6.15 depicts a comparison of the general air spring curves and their equivalent linear spring curves. It is important to note that these equivalent linear spring rates are only used to calculate the estimated damping rates; the model developed in Section 7.2 indeed uses the non-linear spring curves to calculate the spring forces. The values for each linear spring constant is given in the column labeled “Equivalent Spring Rate” in Table 6.5. Using the “Adjusted” mass values and the “Equivalent Spring Rates,” Equation 6.31 was solved to find the possible damping rates. The damping rates were related to the clicks from tortoise settings with the largest damping rate corresponding to 0 clicks from tortoise and the smallest damping rate corresponding to 4 clicks from tortoise. Figure 6.16 depicts the damping rates as a function of clicks from tortoise for the front of the bike. The results shown in this plot were used in the model developed in Section 7.2 to determine the damping rate from the user settings.

The same methods that were used to estimate the damping rates for the front of the bike were done for the rear of the bike as well. While the details of the calculations are not summarized here, they can be found in the MATLAB® script `dampingRatesEstimage.m`. Figure 6.17 depicts the damping rates as a function of clicks from tortoise.
Figure 6.15: General air spring curves and their “equivalent” linear spring curves for the selected pressure values given in Table 6.5.

for the rear of the bike. Again, the values shown in this plot were used in Section 7.2 to determine the damping rate from the user settings for the rear of the bike.
Figure 6.16: Estimated damping rate as a function of clicks from tortoise for the front of the bike.
Figure 6.17: Estimated damping rate as a function of clicks from tortoise for the rear of the bike.
The development of a mathematical model of the mountain bike is a necessary component in order to be able to tune the suspension. As a result, a 2 degree of freedom (DOF) model was developed in MATLAB®. In order to ensure accurate results, a 1 DOF model was developed first. This 1 DOF model was used to test out various functions that were critical to the 2 DOF model but in a much simpler environment. Once the 2 DOF model was verified, the optimal suspension parameters were found using a genetic algorithm.

7.1 1 DOF Model

7.1.1 1 DOF Model Set-up

The 1 DOF model (also known as the quarter-car model in vehicle dynamics) models a simple spring-mass-damper system with a base excitation input. Figure 7.1 depicts the model set-up. The equation of motion is given by

\[ m \ddot{y} + c \dot{y} + k m = mg + c \dot{x} + k (x + l) \]  

(7.1)

where \( l \) is the length of the un-stretched spring.

7.1.2 1 DOF Model Code Development

Since the ultimate goal of this model is to be able to tune a given mountain bike suspension, the model was created from the ground up with data input in mind.
As mentioned in Section 3, the outline for the data acquisition system calls for the use of accelerometers on the front and rear axle that will be used to determine the road profile. As a result, the model was developed from the beginning to use this as an input. The following MATLAB® scripts were written in the development of the model:

1. *TestingAccelerationData.m* – This file creates an array of acceleration data designed to mimic the data that will be recorded from the data acquisition system.

2. *EOM.m* – This file is a MATLAB® function that computes the acceleration of the mass as given in Equation 7.1.

3. *simulink_mastermind.slx* – This is a MATLAB® Simulink® file that solves the equation of motion.

4. *mastermind.m* – This file imports the acceleration data and runs the necessary files to solve it. It then plots the results and shows a simple animation.

Along with computing the acceleration of the mass, the function *EOM.m* also models the bottoming out of the suspension, as well as the situation when the bike jumps
and leaves the ground. This is accomplished as follows. The relative displacement of the mass from the road profile, \( z \), is given by:

\[
z = y - x
\]  

(7.2)

Before solving for the acceleration of the mass, \( EOM.m \) first checks two conditions:

1. If \( z \) is less than some lower limit, called the bottoming out limit, then the suspension has bottomed out. As a result, the spring stiffness is temporarily set arbitrarily high (at least a few orders of magnitude greater than the normal value) to mimic an impact. When the condition no longer holds, the spring rate is restored to the normal value.

2. If \( z \) is greater than the unsprung suspension length, denoted \( l \), then the spring rate and damping rate are set equal to zero, effectively “turning them off.” This allows the model to leave the ground. When the condition no longer holds, the spring and damping rates are restored to their normal values.

The outputs from the model are plotted and a simple animation is shown to help verify accuracy.

### 7.1.3 Testing

Various test cases were created in order to ensure accuracy and stability over a range of inputs. Initially, the simulation was run with the parameters shown in Table 7.1 with no added base excitation. Note that the values correspond with no particular unit system as they were simply used for testing. The first test done was called the “drop test” and involves dropping the model from an initial height and analyzing the response. This test is a very common test done for verifying suspension models. Figure 7.2 shows the response from this test.
Table 7.1: Conditions used for the 1 DOF model drop test.

These values are for testing only and do not correspond to any particular units.

<table>
<thead>
<tr>
<th>Model Parameters</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mass, ( m )</td>
<td>1</td>
</tr>
<tr>
<td>Damping Constant, ( c )</td>
<td>0.8</td>
</tr>
<tr>
<td>Spring Constant, ( k )</td>
<td>1</td>
</tr>
<tr>
<td>Gravity, ( g )</td>
<td>-0.2</td>
</tr>
<tr>
<td>Unsprung Length, ( l )</td>
<td>0.7</td>
</tr>
<tr>
<td>Bottoming-Out Limit, ( bol )</td>
<td>0.1</td>
</tr>
<tr>
<td>Initial Height, ( y_0 )</td>
<td>5</td>
</tr>
<tr>
<td>Initial Velocity, ( \dot{y}_0 )</td>
<td>0</td>
</tr>
<tr>
<td>Solver Time Step</td>
<td>Auto</td>
</tr>
</tbody>
</table>

Note that the response shows a sharp discontinuity at a time of about 8 seconds corresponding to a height of about 0.1. As shown in Table 7.1, this corresponds with the bottoming out limit, \( bol \), height of 0.1. This means that the suspension bottomed-out and indicates that the feature is working.

When using any numerical differential equation solver it is important to check for stability and consistency in order to ensure that the solver is yielding good results. A solver is said to be stable if the error between the numerical solution and the true solution decays with time; it is consistent if the numerical solution approximates the true solution. MATLAB®’s Simulink® environment automatically chooses between more than 100 different solvers as well as automatically adjusts the time step in order to ensure good results. When initially testing the bottoming out feature of the model, a few limitations were found relating to solver time. While this does not directly relate to the stability of the solver itself, the effect is similar. Figure 7.3 depicts the results of using a fixed solver time step on the solution when the suspension bottoms-out. Note that the response in Figure 7.3 (a) shows the mass drop and then shoot way high at about 8 seconds. Similarly, Figure 7.3 (b) shows the mass drop and then shoot higher (although not as high as in (a)). Finally, the response in Figure 7.3 (c) matches
Figure 7.2: Response of the 1 DOF MATLAB® model subject to the conditions given in Table 7.1.

that of Figure 7.2 where the automatic time step was used. This phenomenon results from the way the bottoming out feature is modeled. As mentioned before, when the bottom-out condition is satisfied, the spring constant is temporarily set arbitrarily high in order to model the impact. When a coarse time step such as 0.1 seconds is used, the model does not predict the bottoming out until it has already passed the limit. This extra distance, combined with the large spring constant, has the effect of “sling shooting” the mass up. When the time step is “fine enough” as shown in Figure 7.3 (c), then the phenomenon is avoided and the response looks reasonable. While the MATLAB® solver does a good job of automatically adjusting the time step to account for this (as shown in Figure 7.2), it is still important to be aware of this phenomenon. Consider the response shown in Figure 7.4. Figure 7.4 depicts the response of the model subject to the conditions given in Table 7.4 as well as a step input of magnitude 2 at time 30 seconds. Note that the response has the same “sling shot” effect from the step input at 30 seconds (and not the bottoming out at
Figure 7.3: Comparison of the effects of solver time step on the bottoming out feature of the 1 DOF model.

about 8 seconds) that was seen in Figure 7.3 (a) and (b) even with the solver using an automatic time step. Theoretically if the model was run with a small enough time step it should be able to compute a solution, although at the expense of a lot of computing time.

7.1.4 1 DOF Model Limitations

As a result of the testing completed in Section 7.1.3, the model is limited to accepting “smooth” road profiles as inputs. The definition of a “smooth” road profile is left as a loose definition, for no further work has been done to characterize the exact requirements that make an input “smooth” enough. Rather, it is important to note the phenomenon in case similar behavior is seen again later when testing with actual data. Since the actual data will be recorded from an actual mountain bike, it is unlikely that a profile will be generated that is too extreme for the model to simulate.

Using the 1 DOF model and results as a base, the 2 DOF model was developed.
Figure 7.4: Response of the 1 DOF MATLAB® model subject to the conditions given in Table 7.4 as well as step input of magnitude 2 at time 30 seconds.

7.2 2 DOF Model

7.2.1 2 DOF Model Geometry

The measurements of the geometry of the bike and rider that were found in Section 5 were used to develop a 2D model. Figure 7.5 depicts this model and Table 7.2 gives the values of the various dimensions and mass properties of the model. As can be seen in Figure 7.5, the bike and rider are simplified to a single rigid body. The effects of the tires is not modeled since the acceleration data is collected at the front and rear axles. As explained in Section 5, the model simplifies the fork and rear shock to an equivalent “effective” spring rate and damping rate acting vertically on the front and rear wheel axles. In order to build up the model in this way, it was necessary to figure out the equivalent travel for these “imaginary” shocks and the corresponding
imaginary mounting points on the frame. Finally, the location of the bike and rider CG with respect to these imaginary mounting points needed to be found. This was accomplished by doing the following. The equivalent total travel of the imaginary front and rear shocks is given by $L_F$ and $L_R$, respectively, and the values for each are given in Table 6.3. Thus the imaginary mounting points for the equivalent shocks are located a distance $L_F$ and $L_R$ directly above the front and rear axle, respectively. Next, the location of the bike and rider CG with respect to these imaginary mounting points was found. Since the imaginary mounting points are directly above the front
and rear axle, the horizontal position of the bike and rider CG is the same as that measured in Section 5. These values are given by \( a \) and \( b \). Finally, the vertical position of the CG above these mounting points was calculated. Recall in Section 5 that the vertical position of the CG above the ground was measured to be 1.219 m. This measurement, however, was taken when the rider was sitting on the bike on the ground. Therefore, the front and rear suspension components were compressed. The development of this model requires that the position of the CG be found with respect to the imaginary suspension mounting points, not with respect to the ground. Assuming that the suspension was set to 30\% sag during the measurements (meaning that the suspension was compressed to 70\% of the total travel), the values for \( L_U \) and \( L_L \) can be calculated as follows

\[
L_U = H_{CG/30\%} - (0.7)(L_F) - r_w \tag{7.3}
\]

\[
L_L = H_{CG/30\%} - (0.7)(L_R) - r_w - L_U \tag{7.4}
\]

where \( H_{CG/30\%} \) is the height of the CG found in Table 5.5 of Section 5 (1.219 m) and \( r_w \) is the radius of the front and rear wheels, given in Table 5.3 (0.3493 m). To complete the model, the values for the total mass of the bike and rider as well as the pitching moment of inertia about the CG is given by \( m_{CG} \) and \( I_{CG} \), respectively.

\subsection{7.2.2 2 DOF Model Set-up}

With the geometry of the model defined, the equations of motion could be derived. Figure 7.6 depicts the model set-up. As shown in Figure 7.6, \( y, \dot{y}, \) and \( \ddot{y} \) denotes the vertical position, vertical velocity, and vertical acceleration of the CG of the bike and rider and \( \theta, \dot{\theta}, \) and \( \ddot{\theta} \) denotes the pitching angle, pitching angular velocity, and pitching angular acceleration of the bike and rider. \( x_F, \dot{x}_F, \) and \( \ddot{x}_F \) denotes the vertical position, vertical velocity, and vertical acceleration of the front axle and \( x_R, \dot{x}_R, \) and
\( \ddot{x}_R \) denotes the vertical position, vertical velocity, and vertical acceleration of the rear axle. The values for the vertical acceleration of the front and rear axles are measured using the accelerometers. The data is then used to calculate the corresponding vertical velocity and vertical position. With these values, the relative displacements and velocities of the front suspension mounting point \((z_F, \dot{z}_F)\) and rear suspension mounting point \((z_R, \dot{z}_R)\) can be defined as

\[
\begin{align*}
    z_F &= y - L_U \sin \left( \frac{\pi}{2} - \theta \right) + a \sin(\theta) - x_F \\
    \dot{z}_F &= \dot{y} + \left[ L_U \cos \left( \frac{\pi}{2} - \theta \right) + a \cos(\theta) \right] \dot{\theta} - \dot{x}_F \\
    z_R &= y - (L_U + L_L) \sin \left( \frac{\pi}{2} - \theta \right) - b \sin(\theta) - x_R \\
    \dot{z}_R &= \dot{y} + \left[ (L_U + L_L) \cos \left( \frac{\pi}{2} - \theta \right) - b \cos(\theta) \right] \dot{\theta} - \dot{x}_R
\end{align*}
\]

Figure 7.7 depicts the free body diagram (FBD) and mass acceleration diagram (MAD) for the model. From this, the equations of motion for the vertical acceleration and pitching angular acceleration of the CG of the bike and rider can then be written as

\[
\ddot{y} = \left( \frac{1}{m} \right) \left[ F_{c_F} + F_{k_F} + F_{c_R} + F_{k_R} - F_{mg} \right] 
\]
\[ \ddot{\theta} = \left( \frac{1}{I_{CG}} \right) \left[ (F_{cF} + F_{kF}) \left[ a \cos(\theta) + L_U \cos\left(\frac{\pi}{2} - \theta\right) \right] 
- (F_{cR} + F_{kR}) \left[ b \cos(\theta) - (L_U + L_L) \cos\left(\frac{\pi}{2} - \theta\right) \right] \right] \] (7.10)

where

\[ F_{mg} = m_{CG}g \] (7.11)
\[ F_{cF} = -(\dot{z}_F)(c_F) \] (7.12)
\[ F_{cR} = -(\dot{z}_R)(c_R) \] (7.13)

\[ F_{kF} = \text{frontWheelForce}(P_{1\text{front}}, (L_F - z_F)) \] (7.14)

\[ F_{kR} = \text{rearWheelForce}(P_{1\text{rear}}, (L_R - z_R)) \] (7.15)

Note that the negative sign in Equations 7.12 and 7.13 are there to capture the fact that the damping force always opposes the motion of the shock.

The functions \text{frontWheelForce()} and \text{rearWheelForce()} in Equations 7.14 and 7.15, respectively, are functions that calculate the force acting on the front and rear wheels.
due to the front and rear shocks. The functions use the static pressure of the shock \((P_{1\text{front}} \text{ and } P_{1\text{rear}})\) to calculate the spring curve for the corresponding shock (either the Rockshox Recon TK Silver 27.5" for the front wheel or the X-Fusion O2 Pro RL for the rear wheel) as discussed in Section 6.2.2. They then take in the displacement of the wheel normalized to the uncompressed state \((L_F - z_F)\) and \((L_R - z_R))\), convert it to the equivalent displacement of the corresponding shock, calculate the force in the shock, and then convert the force on the shock back to the force on the wheel. Refer to Section 6 for details on how the displacements and forces are converted between the wheel and the corresponding shock.

Similar to the 1 DOF model discussed in Section 7.1, the 2 DOF model was designed to simulate when either the front or rear wheel jumps off of the ground or when the front or rear suspension bottoms out. Before solving Equations 7.9 and 7.10, the relative displacements \(z_F\) and \(z_R\) are calculated according to Equations 7.5 and 7.7. If either \(z_F\) or \(z_R\) is greater than the total travel of the equivalent front or rear shocks (given by \(L_F\) and \(L_R\)), then the corresponding wheel is considered to have left the ground. When this happens, the corresponding damping force and spring force calculated from Equations 7.12 – 7.15 are set to zero. If either \(z_F\) or \(z_R\) is less than a lower limit, defined as the bottoming out limit in Section 7.1.1, then the corresponding suspension component is considered to have bottomed out. When this happens, the spring rate for the corresponding shock is set to \(1 \times 10^9\) N/m, a number found to be sufficiently high to model the impact from testing. In order to try and maintain continuity in the shock force curve, this spring rate is added on top of the maximum force that can be outputted from the shock for a given pressure setting.
7.2.3 2 DOF Model Code Development

The 2 DOF model was constructed in a similar fashion to the 1 DOF model, with the following framework of MATLAB\textsuperscript{®} files:

1. \textit{mastermind\_2DOF.m} – This file calls all of the other functions to run the model.

2. \textit{Convert\_ADXL375\_Data.m} – This function reads the binary data file outputted from the MTB DAQ and outputs an array containing the time and corresponding accelerations from all three axes for the front and rear accelerometers (outputted in g’s).

3. \textit{Filter\_Data.m} – This function takes the data from \textit{Convert\_ADXL375\_Data.m} and converts it to m/s\textsuperscript{2}. It then calculates a zero-g offset for each axis based on the average value of the data during the start of the testing session and up until a user-defined parameter. The user defined parameter denotes how long the accelerometers were held still at the beginning of the testing session. This is done specifically to calculate the zero-g offset. Once this is done, the function integrates the data using the trapezoidal rule and filters the data using a high-pass filter with a 1 Hz cutoff frequency to eliminate some of the accelerometer drift. For more details on why the data is filtered, refer to Section 4.6.2.

4. \textit{EOM\_2DOF.m} – This function contains the equations of motion for the bike and rider system expressed as a series of coupled first order differential equations. This function solves the equations of motion using the built-in solver \textit{ode45}, as well as adds the bottoming out and jumping characteristics to the model as discussed in Section 7.2.2. The function file also contains the following other “utility” functions:
• *ode_fun* – The function used by the solver *ode45*. This function contains the equations given in Section 7.2.2.

• *frontWheelForce* – This function is used to calculate the “effective” spring force acting vertically on the front wheel as a result of the fork. It computes the general spring curve for the fork based on the static input pressure. It then takes the vertical displacement of the front wheel and calculates the equivalent displacement of the fork. This is used to calculate the force in the fork which is then converted back to the equivalent vertical force acting on the front wheel.

• *rearWheelForce* – This function is used to calculate the “effective” spring force acting vertically on the rear wheel as a result of the rear shock. It computes the general spring curve for the rear shock based on the static input pressure. It then takes the vertical displacement of the rear wheel and calculates the equivalent displacement of the rear shock. This is used to calculate the force in the rear shock which is then converted back to the equivalent vertical force acting on the rear wheel.

5. *Model_Validation.m* – This file takes in a video recording of the bike and the corresponding acceleration data from the MTB DAQ. It then processes the video file frame by frame in order to measure the response of the bike. This measured response is then overlaid with the calculated response found using *EOM_2DOF.m* for comparison.

### 7.2.4 2 DOF Model Testing and Validation

While constructing the 2 DOF model, various tests were performed using slightly modified versions of the testing data generator functions described for the 1 DOF model. In order to properly validate the model, however, it was necessary to test
using data acquired with the MTB DAQ and system parameters that reflected the actual bike and rider system. The measured values for mass, moment of inertia, location of CG, etc. of the bike and rider found in Section 5 were inputted into the 2 DOF model. Even though the model was configured with the proper values, it was still necessary to validate the calculated response by comparing it to the actual response of the bike and rider.

To measure the actual response of the bike, a graphical method was used. Two circular markers were attached to the bike, one on the head tube and one on the seat tube. The markers were made of concentric black and white circles (cut out from construction paper), with the black circle roughly twice the diameter of the white circle. This was done to ensure that the white circle always remained the same shape and size from the perspective of the camera. If, for example, only a white circle was used, and the bike went in front of a very bright background, then there is a possibility that during the image processing the white circle may be lost. Figure 7.8 depicts the bike with these markers. To further reduce the possibility of losing the markers during

![Figure 7.8: Locations of the markers attached to the bike in order to measure the response of the bike and rider using graphical methods.](image-url)
image processing, the testing was done in the author’s garage. A large blue plastic tarp was hung from one wall to provide a uniform dark background. While it would have been preferable to use a material that is less reflective than the plastic tarp, the tarp turned out to work just fine. A single piece of wood, approximately 8.26 cm tall by 14 cm wide, was placed on the ground. A Sony Alpha 6300 camera, set to record at 120 frames/second, was set up on a tripod on the other side of the garage. The garage doors were closed and the lights turned off. A flashlight was then placed, facing up towards the ceiling, behind the camera to provide uniform lighting from behind the camera. This was done in an effort to illuminate the markers as brightly as possible while keeping everything else dark. Figure 7.9 depicts the testing setup. The author wore black pants, black socks, a black long sleeve shirt, and dark shoes as well to make it as easy as possible to pick out the markers.

Various trials were done, with each one recorded by the camera as well as with the MTB DAQ. The video file from the best trial was then processed using the
Model\_Validation.m MATLAB® script. The script traced the positions of the two markers throughout the entire test and then used them to compute the position and pitching angle of the CG of the bike and rider throughout the entire test. A video file was outputted with the traces overlaid for verification that the marker tracing was working. Figure 7.10 depicts a frame from the outputted video file. For more detail on the Model\_Validation.m MATLAB® script, refer to Appendix F.

Figure 7.10: Single frame from the video file output of the Model\_Validation.m MATLAB® script depicting the traced out position of the front and rear markers as well as the bike and rider CG.

The response shown in Figure 7.10 is in units of pixels. In order to correlate the pixels to meters, a reference was needed. To do this, the diameter of the rear wheel was measured in pixels. Thus by dividing the actual diameter of the rear wheel in meters by the diameter in pixels, a correlation of approximately 0.0017 m/pixel was found. Using this correlation the measured response was converted into meters. This was overlaid on a plot of the calculated response from the model. Figure 7.11 depicts the vertical motion overlay plot and Figure 7.12 depicts the pitching angle overlay plot. In both plots, the model response was normalized to the measured response such that
Figure 7.11: Overlay of the measured response and the calculated response from the model. The shaded region represents ±0.01 m above and below the measured response.

the first data point of the model response equaled the first data point of the measured response. This was done to account for the drift in the accelerometers (as discussed in Section 4.6.2) between starting the recording and riding over the piece of wood. As can be seen in Figure 7.11, the calculated response from the model matches the measured response to within ±0.01 m for the first 0.4 seconds. After that, the calculated response drifts below the measured response. The calculated pitching angle, as shown in Figure 7.12, shows a similar behavior, agreeing with the measured response to within ±2° for the first 0.4 seconds before drifting above. Figure 7.13 depicts the input position data as measured by the accelerometers corresponding to the responses shown in Figures 7.11 and 7.12. Recall that the test consisted of riding the bike over
Figure 7.12: Overlay of the measured pitching angle and the calculated pitching angle from the model. The shaded region represents $\pm 2^\circ$ above and below the measured pitching angle.

a flat and level garage floor and over a piece of wood approximately 8.26 cm tall by 14 cm wide. The input position data should reflect this. As can be seen, however, the data drifts, showing that the front wheel of the bike started approximately 0.02 m below the ground at 0 seconds and eventually drifted up to 0 m at 1.4 seconds. Moreover, the rear wheel started at approximately 0 m at 0 seconds and drifted up and down before ending at a approximately 0 m at 1.4 seconds. Despite the drifting in the accelerometer, however, the important information is still maintained. The input position data correctly captured the proper height of the piece of wood that was ridden over, for the height of the each peak (when measured from the bottom to
Figure 7.13: Input position data as measured by the accelerometers for the front and rear of the bike during the validation testing session. The input data here corresponds with the responses shown in Figures 7.11 and 7.12. The distance (from the top) is approximately 8 cm, differing from the true height of the piece of wood by no more than 3%. This information is properly translated to the calculated response. As can be seen in Figures 7.11 and 7.12, while the calculated responses do drift over time, they maintain the same characteristics of the measured response throughout the entire test. In other words, the concavity and relative amplitude of the calculated response matches that of the measured response for all time. Since the goal of the model is ultimately to tune a suspension system, it is not necessary that the calculated response exactly matches the true response. Rather, it is only important that the overall characteristics of the response are properly captured, exactly what is seen
It is important to note, however, that this analysis is only valid when the bike is ridden for “short” distances on level ground. The high-pass filter implemented in the *Filter_Data.m* MATLAB® script is able to eliminate most of the accelerometer drift but not all of it. Since the accelerometer drift only gets worse as the testing session time increases, it is important to keep tests “short.” While no further rigorous analysis was conducted to exactly quantify the maximum length of a valid testing session, this phenomenon is known and will be taken into consideration when analyzing the results.

Further verification of the model was done with the testing data to ensure the model behavior looked correct when the suspension settings were changed. Figure 7.14 depicts the calculated vertical position of the CG and Figure 7.15 depicts the calculated pitching angle of the CG for a variety of suspension settings for the same input conditions as described above. As can be seen in Figure 7.14, as the pressure in the front and rear shocks increases, so does the overall height of the response. This indicates that bike sits higher under the weight of the rider when static (less “sag”), exactly what is expected. Further examination of the heights of the peaks (found by measuring the value of the response just before the peak and subtracting it from the highest point of the peak) revealed information about the model as well. Measuring all of the peaks and comparing revealed that the highest peak, generated from the highest spring rates, was taller than the shortest peak, generated from the lowest spring rates, by a maximum of 2.2 cm. This makes sense, for riding over a large piece of wood with a stiff suspension should result in a taller peak value of the bike and rider CG. Looking at the difference in damping rates leads to a similar conclusion. For a given spring rate, the smaller clicks from tortoise value (larger damping rate) lead to a higher response than for a larger clicks from tortoise value. This is due to the fact that a larger clicks from tortoise value means the damping rate is less, so the damping force is less. When under motion, this allows the suspension to sag more, as
Figure 7.14: Various responses of the vertical position of the CG from different suspension settings (pressure and clicks from tortoise) for the same input conditions as computed by the model.

is shown in the plots. Figure 7.15 depicts the same information. As the spring rate increases, so does the pitching angle, and vice versa.

Comparing the modeled response to the measured response and checking the model behavior for a range of suspension settings created good confidence in the validity of the model. Therefore, for the intents and purposes of this thesis, the 2 DOF model was considered to be valid.
7.3 Genetic Algorithm Implementation

In order to find the optimal suspension settings, a genetic algorithm was implemented. While there are many methods available for solving optimization problems, a genetic algorithm was used based on the work done by R. Alkhhatib, G. N. Jazar, and M. Golnaraghi [6]. As mentioned in Section 2.2, a genetic algorithm is a robust optimization solver that does a good job of finding the true optimal value.
7.3.1 Objective Function

The MATLAB® Global Optimization Toolbox includes a built in genetic algorithm function, `ga`. This algorithm was implemented in a file called `GeneticAlgorithm.m`. The `GeneticAlgorithm.m` script takes in the system information (such as bike and rider properties, measured input values, etc.) and passes it on to the genetic algorithm function `ga`. Included in `GeneticAlgorithm.m` is the objective function, or the function that the genetic algorithm is trying to minimize. The objective function is given by

\[ objFun = (C_1)(a_{op}) + (C_2)(\alpha_{op}) + (C_3)(B_F) + (C_4)(B_R) \]  \hspace{1cm} (7.16)

where \( C_1, C_2, C_3, \) and \( C_4 \) are constants that can be used to fine tune the function and \( a_{op}, \alpha_{op}, B_F, \) and \( B_R \) are the optimization criteria. It is important to note that \( C_1, C_2, C_3, \) and \( C_4 \) were all set to 1 during all tests. However, they are left in as a reminder that further fine tuning is possible. \( a_{op} \) is the root-mean-square of the vertical accelerations of the CG of the bike and rider normalized to the root-mean-square of the vertical accelerations of the rear wheel. Recall from Section 2.5 that this is known as the discomfort transmission and is a valid metric for the performance of the suspension system [17]. \( a_{op} \) is given by

\[ a_{op} = \frac{\text{rms}(a_{CG})}{\text{rms}(a_{rear})} \]  \hspace{1cm} (7.17)

where \( a_{CG} \) is an array of the vertical accelerations of the bike and rider CG calculated by the model and \( a_{rear} \) is an array of the vertical accelerations measured at the rear axle. \( \alpha_{op} \) is the root-mean-square of the pitching accelerations of the CG of the bike and rider normalized to the root-mean-square of the pitching accelerations measured between the front and rear axle. \( \alpha_{op} \) is given by

\[ \alpha_{op} = \frac{\text{rms}(\alpha_{CG})}{\text{rms}(\alpha_{input})} \]  \hspace{1cm} (7.18)

where \( \alpha_{CG} \) is an array of the pitching accelerations of the the bike and rider CG calculated by the model and \( \alpha_{input} \) is an array of the pitching accelerations between
the front and rear axle. In other words, $\alpha_{input}$ quantifies how much the trail pitches based on the wheelbase and speed of the bike that is traversing it. $\alpha_{input}$ is estimated using

$$\alpha_{input} = \frac{a_{front} - a_{rear}}{wb}$$

(7.19)

where $a_{front}$ is an array of the vertical accelerations measured at the front axle, $a_{rear}$ is an array of the vertical accelerations measured at the rear axle, and $wb$ is the wheel base of the bike. $B_F$ and $B_R$ are simply counters that record how many steps during the $ode45$ solver that the model was bottoming out in the front and rear, respectively. They are included to guarantee that the optimal suspension parameters found from the genetic algorithm do not allow the suspension to bottom out.

While the objective function serves to provide the genetic algorithm with a measure of how well each possible solution solves the problem, it also defines the variables that can be modified to try and achieve a better solution. It is through the objective function definition in MATLAB® that the algorithm is “notified” that the front and rear static air pressure and clicks from tortoise variables are the values that can be modified to try and achieve a more optimal response.

### 7.3.2 Solution Constraints

Along with the objective function, $GeneticAlgorithm.m$ also includes constraints on the possible solutions. These constraints are based on the following. The maximum and minimum settings possible for static air pressure and clicks from tortoise for the front and rear shock (as summarized in Table 6.1) provide natural upper and lower bounds for possible solutions. The possible solutions can be further refined based on what adjustments are actually physically possible to make on the shock. The damping rate, for example, is changed by turning a dial to one of a finite number
of fixed positions. Therefore, the clicks from tortoise settings can be restricted to integer values. Adjustments made to the shock pressure are done with a shock pump. These pumps typically have needle gauges with markings every 10 psi, meaning that they are only accurate to about 5 psi. Therefore, the possible settings for static air pressure can be restricted to increments of 5 psi.

Without utilizing custom solutions, the $ga$ function can only restrict the possible solutions to integers within a certain range. Therefore, in order to restrict the static pressure settings to increments of 5 psi, the following approach was taken. Consider the maximum and minimum values for static air pressure of the Rockshox Recon TK Silver 27.5” fork. The difference between the maximum and minimum values can be found using the values in Table 6.1. Dividing the difference by 5 psi gives

$$\frac{120 \text{ psi} - 50 \text{ psi}}{5 \text{ psi}} = 14$$

This means that there are 15 possible values for the pressure in the fork (one additional value to include zero). In other words, the front static pressure can be bounded to be an integer in between 0 and 14. To convert to pressure, the following equation is used

$$\text{Front Pressure} = n \text{ (psi)} + 50 \text{ psi}$$

where $n$ is an integer on the interval $0 \leq n \leq 14$. Using the same method, the rear pressure for the X-Fusion O2 Pro RL shock can be found using

$$\text{Rear Pressure} = n \text{ (psi)} + 60 \text{ psi}$$

where $n$ is an integer on the interval $0 \leq n \leq 48$. In practice, the solutions were constrained in $GeneticAlgorithm.m$ by restricting each variable to be an integer with the following upper and lower bounds:

$$0 \leq \text{Front Clicks From Tortoise} \leq 4$$

$$0 \leq \text{Front Pressure} \leq 14$$
Since the variables are constrained in this method, Equations 7.21 and 7.22 are incorporated into the objective function calculation and the returned values of *GeneticAlgorithm.m* in order to make sure the proper values are being used. Even after constraining the variables, there are still \((5)(15)(12)(49) = 44,100\) possible solutions!

### 7.3.3 Genetic Algorithm Tuning

There are many different parameters that can be adjusted in the *ga* algorithm. Settings such as population size, crossover options, or the maximum survival rate, to name a few, can all be manipulated and fine tuned to make the genetic algorithm work properly. Extensive testing was done to determine which parameters needed to be changed in order to ensure the absolute minimum, as opposed to a local minimum, was being found within the shortest amount of time possible. In order to determine what needed to be changed, the data outputted to the console from the genetic algorithm was consulted. Figure 7.16 depicts the output from the *ga* function after interpreting the data from a testing session. As can be seen in Figure 7.16,

![Figure 7.16: Typical console output from the *ga* algorithm.](image-url)

the algorithm outputs the current generation number and its corresponding data.

\[
0 \leq \text{Rear Clicks From Tortoise} \leq 11
\]

\[
0 \leq \text{Rear Pressure} \leq 48
\]
The value under “Func-count” indicates the cumulative number of times the objective function has been called up to the generation indicated. The value under “Best Penalty” indicates the best value of the objective function that was calculated from any single member of the population of that generation. The value “Mean Penalty” indicates the mean of all of the values of the objective function from each member of the population of that generation. The value under “Stall Generations” indicates how many generations have been solved with no change in the value for “Best Penalty.” Finally, the reason for terminating the genetic algorithm is listed at the bottom. It is important to note that when the possible solutions are restricted to integers, the \textit{ga} function solves a slightly modified version of the objective function that allows it to account for solutions that are out of bounds of the solution space. Since the mechanics of the penalty function are beyond the scope of this discussion, suffice it to say that the penalty scores are, for all intents and purposes, the values of the objective function.

All testing for fine tuning the genetic algorithm was conducted using the data from test 1 according to Figure G.1 (refer to Section 8.1 for a description of the testing). The testing consisted of running \textit{GeneticAlgorithm.m} to completion and comparing the results with those from previous tests. Table 7.3 depicts the results from running \textit{GeneticAlgorithm.m} with different settings used for \textit{ga}. All of the settings referenced in Table 7.3 are named according to the naming convention given in the MATLAB® documentation [5]. Initially attempts were made to limit the population size, maximum generations, and maximum stall generations in order to speed up the algorithm. The default maximum number of generations is 100 times the number of variables, meaning if left unmodified the algorithm would run for 400 generations! This number was immediately reduced to 20 in order to facilitate faster testing sessions. Trials 1–6 depict the results from adjusting these parameters. As can be seen, however, the recommended suspension parameters (and, by extension, the corresponding calculated
Table 7.3: Output data from running *GeneticAlgorithm.m* with various settings for *ga* on the data from test 1 as defined in Figure G.1.

<table>
<thead>
<tr>
<th>Trial</th>
<th>Front</th>
<th>Rear</th>
<th>Best Penalty</th>
<th>Modified Genetic Algorithm Setting(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>From</td>
<td>From</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Tortoise</td>
<td>Pressure</td>
<td></td>
<td>PopulationSize: 20</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>MaxGenerations: 20</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>MaxStallGenerations: 10</td>
</tr>
<tr>
<td>1</td>
<td>3</td>
<td>6</td>
<td>112</td>
<td>141</td>
</tr>
<tr>
<td>2</td>
<td>4</td>
<td>9</td>
<td>114</td>
<td>184</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td>10</td>
<td>115</td>
<td>155</td>
</tr>
<tr>
<td>4</td>
<td>4</td>
<td>9</td>
<td>115</td>
<td>180</td>
</tr>
<tr>
<td>5</td>
<td>3</td>
<td>10</td>
<td>55</td>
<td>255</td>
</tr>
<tr>
<td>6</td>
<td>4</td>
<td>10</td>
<td>120</td>
<td>170</td>
</tr>
<tr>
<td>7</td>
<td>4</td>
<td>9</td>
<td>115</td>
<td>180</td>
</tr>
<tr>
<td>8</td>
<td>4</td>
<td>9</td>
<td>115</td>
<td>180</td>
</tr>
</tbody>
</table>

"Best Penalty") were different for each of these trials. This meant that the genetic algorithm was finding local minimums, and the particular one that was found was up to chance. In order to make sure the genetic algorithm would find the absolute minimum, the genetic diversity was increased. This was done by doubling the population size and changing the “EliteCount” and “CrossoverFraction” for trials 7 and 8 as shown. The “EliteCount” parameter determines how many individuals of a population are guaranteed to survive to the next generation [5]. The default value is 50% of the population. This number was lowered such that only 20 of the total 80 individuals would survive (instead of 40), meaning there was more opportunity for genetic diversity. The “CrossoverFraction” specifies the fraction of the next generation that are produced by crossover (a mating between two parent solutions) as opposed
to those that are randomly generated (mutated). Lowering the “CrossoverFraction” ensures that there is more genetic diversity. The results for trials 7 and 8, as shown in Table 7.3, indicate that increasing the genetic diversity led to the algorithm finding the absolute minimum solution. The results for trials 7 and 8 exactly match those of trial 4, indicating that in trial 4 the absolute minimum was stumbled upon by chance. Having reproduced the same result after three different trials, the settings used in trials 7 and 8 were considered sufficient and the algorithm was considered to be properly tuned.

With the 2 DOF model validated and the genetic algorithm implemented, testing could begin. Refer to Section 8 for a complete summary of the testing that was conducted.
Various tests were performed to evaluate the effectiveness of tuning a mountain bike suspension using the MATLAB® scripts outlined in Section 7. Due to the limitations in testing duration and trail slope of the MTB DAQ discussed in Section 7.2.4, the tests were designed to be “short” (no more than roughly 10 seconds of time actually riding the bike) and on relatively level ground. The tests and results are summarized in the following sections.

8.1 Testing Setup

A total of five different courses were developed in order to simulate different scenarios likely to be found on a typical mountain bike trail. The courses were designed to capture different features common to most trails, such as high frequency disturbances, small and large jumps, and large blunt obstacles. Figure 8.1 (a)–(e) depicts the various testing courses. Each course was ridden a minimum of three times and the data recorded as separate tests. Care was given to try and emulate the bike speed and path as precisely as possible in order to provide a valid comparison between different data sets for a given course. The various test settings (such as front and rear suspension settings, front and rear tire pressure, etc.) as well as technical information about the course (such as height and length of the obstacles) was recorded in a testing log. Refer to Appendix G for a copy of the filled-in testing log used for these tests.
Figure 8.1: Various courses designed to emulate different features common to most mountain bike trails.

8.2 Results

The data from each test was run through the genetic algorithm and the results were recorded. Various plots were created comparing the modeled response with the test settings compared to those with the recommended genetic algorithm set-
tings. Figure 8.2 depicts these plots with the data from test 4 when riding over the high frequency course. The same plots as those shown in Figure 8.2 were made for every test and can be found in Appendix H. The CG response with the GA settings in Figure 8.2 (b) was “normalized” to the CG response with the test settings by doing the following. The local minima were found for each response using the MATLAB® function \textit{islocalmin}. A minimum prominence value was set in the function such that only the local minima in the “deepest” valleys would be found. The difference between corresponding local minima from each response was found and averaged. The CG response with the GA settings was then shifted up by this average amount. This effectively made each response occur at the same ride height and allowed for a better visual comparison of the relative size of the peaks from each response. Figure 8.3 depicts a zoomed in version of the plot in Figure 8.2 (b) with the data normalized according to the procedure discussed above. As can be seen in Figure 8.3, the local maxima were also found for each peak using a similar method to that used to find the local minima. While the two responses look similar, there are differences that

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{figure8.2.png}
\caption{Comparison of the vertical position response and pitching angle response from the bike when riding with the original settings and the genetic algorithm recommended settings with the data from test 4.}
\end{figure}

(a) Vertical position response  (b) Vertical position response  (c) Pitching angle response with inputs
make the genetic algorithm recommended settings more desirable. The average peak heights of the vertical response from the genetic algorithm recommended settings, for example, are on average 20% smaller than the corresponding peaks from the response with the test settings. The pitching angle response shows a similar result, with the average peak difference measuring around 14%.

Table 8.1 summarizes the genetic algorithm recommended settings from each test. The genetic algorithm results of the tests that were performed on the same course were averaged together with the pressure values rounded to the nearest 5 psi. This is shown in the gray colored rows. It is important to note that the pressure values recommended by the genetic algorithm are much lower than those recommended by
the shock manufactures. Further, the damping rates for both the front and the rear are as low as possible. This is a direct result of the objective function given in Equation 7.16. In order to minimize the objective function, it is necessary to use as much suspension travel as possible and to allow the suspension to travel quickly. A long travel suspension that can quickly react to obstacles on the trail is more effective at absorbing the shock from those obstacles and preventing them from reaching the rider. Consequently, it becomes clear that for each trial, the genetic algorithm is essentially finding the lowest pressure possible that will just barely prevent the shock from bottoming out. This, in turn, ensures that the maximum travel of each shock is used for every course. These observations were confirmed by doing the following. The bike suspension was tuned according to the average settings in Table 8.1 and each course was run through a couple of times in order to get a feel for how the bike performed. The rider feedback is summarized below.

<table>
<thead>
<tr>
<th>Trial</th>
<th>Course Name</th>
<th>Front</th>
<th>Rear</th>
<th>Best Penalty</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Clicks From Tortoise</td>
<td>Pressure</td>
<td>Clicks From Tortoise</td>
</tr>
<tr>
<td>1</td>
<td>Small</td>
<td>4</td>
<td>85</td>
<td>11</td>
</tr>
<tr>
<td>2</td>
<td>wooden</td>
<td>4</td>
<td>100</td>
<td>11</td>
</tr>
<tr>
<td>3</td>
<td>jump</td>
<td>4</td>
<td>55</td>
<td>11</td>
</tr>
<tr>
<td></td>
<td>Average:</td>
<td>4</td>
<td>80</td>
<td>11</td>
</tr>
<tr>
<td>4</td>
<td>High</td>
<td>4</td>
<td>55</td>
<td>11</td>
</tr>
<tr>
<td>5</td>
<td>frequency</td>
<td>4</td>
<td>95</td>
<td>11</td>
</tr>
<tr>
<td>6</td>
<td>course</td>
<td>4</td>
<td>55</td>
<td>11</td>
</tr>
<tr>
<td></td>
<td>Average:</td>
<td>4</td>
<td>70</td>
<td>11</td>
</tr>
<tr>
<td>7</td>
<td>Large</td>
<td>4</td>
<td>85</td>
<td>11</td>
</tr>
<tr>
<td>8</td>
<td>blunt</td>
<td>4</td>
<td>85</td>
<td>11</td>
</tr>
<tr>
<td>9</td>
<td>obstacle</td>
<td>4</td>
<td>75</td>
<td>11</td>
</tr>
<tr>
<td></td>
<td>Average:</td>
<td>4</td>
<td>80</td>
<td>11</td>
</tr>
<tr>
<td>10</td>
<td>Large</td>
<td>4</td>
<td>50</td>
<td>11</td>
</tr>
<tr>
<td>11</td>
<td>rolling</td>
<td>4</td>
<td>50</td>
<td>11</td>
</tr>
<tr>
<td>12</td>
<td>bump</td>
<td>4</td>
<td>50</td>
<td>11</td>
</tr>
<tr>
<td></td>
<td>Average:</td>
<td>4</td>
<td>50</td>
<td>11</td>
</tr>
<tr>
<td>13</td>
<td>Dirty</td>
<td>4</td>
<td>55</td>
<td>11</td>
</tr>
<tr>
<td>14</td>
<td>jump</td>
<td>4</td>
<td>50</td>
<td>11</td>
</tr>
<tr>
<td>15</td>
<td></td>
<td>4</td>
<td>60</td>
<td>11</td>
</tr>
<tr>
<td>16</td>
<td></td>
<td>4</td>
<td>50</td>
<td>11</td>
</tr>
<tr>
<td></td>
<td>Average:</td>
<td>4</td>
<td>50</td>
<td>11</td>
</tr>
</tbody>
</table>
• Small wooden jump – Initially the suspension felt too soft, especially in the rear. Riding over the small jump, however, proved that it was not. Neither the front nor the rear bottomed out and the overall ride felt smooth and comfortable.

• High frequency course – The front felt stiff through the course. It seemed that the front tire was absorbing all the bumps and that the fork was barely being compressed. The rear felt soft and made the course feel smooth.

• Large blunt obstacle – Overall the ride felt smooth, although the rear shock did bottom out. The bottoming out was barely noticeable but was repeatable over the few testing runs through the course.

• Large rolling bump – The rear shock was too soft. Repeatably the rear shock bottomed out. It was not bad, but definitely noticeable. Consequently, the rear felt too soft and made the bike feel unbalanced when compared to the front.

• Dirt jump – The suspension felt really good. The settings made the jump feel nice and soft and the bike felt balanced. Neither suspension component bottomed out. A stiffer suspension would most likely be better for overall control of the bike, but for this case the settings recommended were very good.

As summarized by the comments above, the observations made about the recommended values from the genetic algorithm are fairly accurate. Of the five total courses, the rear suspension was detected bottoming out on two of them. Of these two, one was just barely noticeable. These results confirm that the genetic algorithm is indeed finding the suspension settings that allow the bike to traverse the course while just barely avoiding bottoming out. The fact that the suspension did bottom out, however, does indicated that more fine tuning of both the model and the algorithm are required in order to get more accurate results. On the whole, however, the results are promising.
9.1 Summary

The work presented in this thesis outlines a quantitative method for tuning a mountain bike suspension. Section 4 discusses the design, development, and testing of the custom data acquisition system known as the MTB DAQ, used to gather acceleration data for input into the model. Section 5 discusses how the geometric and dynamic properties of the bike and rider were measured. Section 6 outlines the methods and procedures used to characterize the front and rear shocks of the bike and discusses how “equivalent” front and rear shocks were found for use in the model. Section 7 reviews the dynamic model of the bike and rider and discusses the details of the genetic algorithm. Finally, Section 8 discusses the testing that was done and summarizes the results. Reviewing the data and testing the bike with the genetic algorithm recommended settings proved that the genetic algorithm is working as designed.

9.2 Future Work

There is plenty of opportunity to fine tune and expand on the work done here. The MTB DAQ, for example, could be further refined to fit in a smaller form factor and possibly made wireless. Additional sensors could be added, such as a gyro and GPS unit, that would allow for the implementation of a Kalman filter. Implementing a Kalman filter would allow the MTB DAQ to precisely reconstruct the path of the front and rear axles for any type of trail no matter how long or steep. When contrasted
to the restrictions that the testing sessions be kept “short” and on level ground as discussed in Section 7.2.4, this drastically increases the capability of the system.

Further work could be done to refine the model. As discussed in Section 2.3, adapting the model to work with PSD functions could possibly lead to a more powerful and less computationally intense model. Moreover, additional detail could be added to model the damping rates of the shocks. Even though the shocks used here only had simple rebound rate adjustments, many higher performing shocks can independently adjust the high and low speed rebound and compression damping rates as discussed in Section 1.2. Adding this functionality to the model would allow it to predict the behavior of a wide variety of shocks with greater accuracy. Moreover, properly measuring the damping rates of the shocks, instead of estimating their values as discussed in Section 6.3, would be very beneficial to the model.

The objective function used here was written to achieve the smoothest possible ride. As was seen, this resulted in settings that give the softest spring rate possible just before bottoming out for each course. Modifying the objective function, consequently, would drastically change what settings the genetic algorithm recommends. Building in commonly used practices, such as a balanced feel between the front and rear spring rate, for example, could prove to increase the performance of the objective function.
BIBLIOGRAPHY


*Ford Motor Co.*
APPENDICES

Appendix A

PYBOARD V1.1 SCHEMATIC

Figure A.1: Schematic of the PyBoard v1.1 [11].
Appendix B

MTB DAQ SCHEMATICS, BOARDS, AND BILL OF MATERIALS

B.1 Main Board

Figure B.1: Top of main board rendering.
Figure B.2: Bottom of main board rendering.

Figure B.3: Side of main board rendering.
Figure B.4: Main board EAGLE schematic, page 1/3.
Figure B.5: Main board EAGLE schematic, page 2/3.
Figure B.6: Main board EAGLE schematic, page 3/3.
B.2 UI Board

Figure B.7: Top of UI board rendering.

Figure B.8: Bottom of UI board rendering.
Figure B.9: Side of UI board rendering.
Figure B.10: UI board EAGLE schematic, page 1/1.
B.3 ADXL375 Board

Figure B.11: ADXL375 top of board rendering.

Figure B.12: ADXL375 bottom of board rendering.
Figure B.13: ADXL375 side of board rendering.
Figure B.14: ADXL375 board EAGLE schematic, page 1/1.
<table>
<thead>
<tr>
<th>ASSEMBLY</th>
<th>SUB ASSEMBLY</th>
<th>PART</th>
<th>DESCRIPTION</th>
<th>AMOUNT</th>
<th>UNIT</th>
</tr>
</thead>
<tbody>
<tr>
<td>MAIN UNIT</td>
<td>-</td>
<td>JUMPER</td>
<td>JUMPER</td>
<td>1</td>
<td>PCSTR</td>
</tr>
<tr>
<td>MAIN UNIT</td>
<td>-</td>
<td>MEMS ZIGBEE LONG REX CAP SCR</td>
<td>MEMS ZIGBEE LONG REX CAP SCR</td>
<td>1</td>
<td>CH</td>
</tr>
<tr>
<td>MAIN UNIT</td>
<td>-</td>
<td>POWER SWITCH</td>
<td>POWER SWITCH</td>
<td>1</td>
<td>PCB</td>
</tr>
<tr>
<td>MAIN UNIT</td>
<td>-</td>
<td>QUICK CONNECT CABLE Dip Switch</td>
<td>QUICK CONNECT CABLE Dip Switch</td>
<td>1</td>
<td>PCB</td>
</tr>
<tr>
<td>MAIN UNIT</td>
<td>-</td>
<td>RIBBON CABLE</td>
<td>RIBBON CABLE</td>
<td>1</td>
<td>PCB</td>
</tr>
<tr>
<td>MAIN UNIT</td>
<td>-</td>
<td>THREAD LOCKER</td>
<td>THREAD LOCKER</td>
<td>1</td>
<td>HS</td>
</tr>
<tr>
<td>MAIN BOARD</td>
<td>-</td>
<td>SIDE PANEL 1</td>
<td>SIDE PANEL 1</td>
<td>1</td>
<td>PCB</td>
</tr>
<tr>
<td>MAIN BOARD</td>
<td>-</td>
<td>SIDE PANEL 2</td>
<td>SIDE PANEL 2</td>
<td>1</td>
<td>PCB</td>
</tr>
<tr>
<td>MAIN BOARD</td>
<td>-</td>
<td>MAIN BOARD PCB</td>
<td>MAIN BOARD PCB</td>
<td>1</td>
<td>PCB</td>
</tr>
<tr>
<td>MAIN BOARD</td>
<td>-</td>
<td>LITHIUM POLYMER BATTERY</td>
<td>LITHIUM POLYMER BATTERY</td>
<td>1</td>
<td>PCB</td>
</tr>
</tbody>
</table>

**SUPPLIERS**

- ADAFRUIT INDUSTRIES LLC
- DIODES INCORPORATED
- STMICROELECTRONICS
- TE CONNECTIVITY
- JAUCH QUARTZ
- SAMTEC INC.
- ECS INC.
- KEMET
- MOLEX
- MCM STEER
- DIGIKEY
- METROSTECH
- ESWITCH
- 3D PRINTED

**QTY/UNIT**

1 | 1 | 1 | 5 | 5 | 1 | 1 | 2 | 1 | 1 | 1

**DESCRIPTION**

- MAIN UNIT - EXTRUSION BOX ALUM 4 X 1 INCH LENGTH 5 INCH
- MAIN BOARD PRINTED CIRCUIT BOARD 8 X 8 INCH
- LITHIUM POLYMER BATTERY 3.7V 2.0AH
- ELECTRONIC DISPLAY 2 X 16 CHARACTER 16X2 CHARACTER

**MFG. PART NO.**

- 0090251241
- 0900002557
- 04921576
- 04071710
- 04071710
- 04071710
- 04071710
- 04071710
- 04071710
- 04071710
- 04071710

**SUPPLIER PART NO.**

- DS-01-10-000-005
- WM24066CT-ND
- A129757CT-ND
- A126415CT-ND
- 732-4985-1-ND
- CBR08C209BAGAC
- CBR08C240JAGAC
- PMEG2010ER, 115
- TL3301FF160QG
- 150080SS75000
- PV0H240SS
- 1911
- 92985A815

**QTY/UNIT**

1 | 1 | 1 | 5 | 5 | 1 | 1 | 2 | 1 | 1 | 1

**MFG. PART NO.**

- 0090251241
- 0900002557
- 04921576
- 04071710

**SUPPLIER PART NO.**

- DS-01-10-000-005
- WM24066CT-ND
- A129757CT-ND
- A126415CT-ND
- 732-4985-1-ND
- CBR08C209BAGAC
- CBR08C240JAGAC
- PMEG2010ER, 115
- TL3301FF160QG
- 150080SS75000
- PV0H240SS
- 1911
- 92985A815
<table>
<thead>
<tr>
<th>ASSEMBLY</th>
<th>SUB ASSEMBLY</th>
<th>PART</th>
<th>DESCRIPTION</th>
<th>MFG</th>
<th>MFG. PART NO.</th>
<th>SUPPLIER</th>
<th>SUPPLIER PART NO.</th>
<th>QTY/UNIT</th>
</tr>
</thead>
<tbody>
<tr>
<td>MAIN UNIT</td>
<td>UE BOARD</td>
<td>-</td>
<td>FRONT MOUNT BLOCK</td>
<td>3D PRINTED</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MAIN UNIT</td>
<td>UE BOARD</td>
<td>-</td>
<td>FRONT COVER</td>
<td>3D PRINTED</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FRONT ACCELEROMETER UNIT</td>
<td>ADXL375 BOARD</td>
<td>U1</td>
<td>IC</td>
<td>TEXAS INSTRUMENTS</td>
<td>SN74LVCG122DKR</td>
<td>Digikey</td>
<td>296-9084-1-ND</td>
<td>1</td>
</tr>
<tr>
<td>FRONT ACCELEROMETER UNIT</td>
<td>ADXL375 BOARD</td>
<td>RJ45</td>
<td>M1205 brass inserts</td>
<td>AMPHENOL (COMMERCIAL PRODUCTS)</td>
<td>R-0157600101</td>
<td>Digikey</td>
<td>R-0157600101CT-ND</td>
<td>1</td>
</tr>
<tr>
<td>FRONT ACCELEROMETER UNIT</td>
<td>ADXL375 BOARD</td>
<td>C3, C1</td>
<td>CAP CER 0.1UF 0V XTR 0065</td>
<td>KEMET</td>
<td>CON0C0016036BU777ACUL</td>
<td>Digikey</td>
<td>999-1104-1-ND</td>
<td>2</td>
</tr>
<tr>
<td>FRONT ACCELEROMETER UNIT</td>
<td>ADXL375 BOARD</td>
<td>C2</td>
<td>CAP CER 0.1UF 0V XTR 0065</td>
<td>KEMET</td>
<td>CON0C0016036BU777ACUL</td>
<td>Digikey</td>
<td>999-1105-1-ND</td>
<td>1</td>
</tr>
<tr>
<td>FRONT ACCELEROMETER UNIT</td>
<td>ADXL375 BOARD</td>
<td>ADXL375</td>
<td>ACCELEROMETER 200 DC/ SPI/ LG</td>
<td>ANALOG DEVICES</td>
<td>ADXL375DC</td>
<td>Digikey</td>
<td>AD-XL375DC/25</td>
<td>1</td>
</tr>
<tr>
<td>FRONT ACCELEROMETER UNIT</td>
<td>-</td>
<td>-</td>
<td>FRONT MOUNT BLOCK</td>
<td>3D PRINTED</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FRONT ACCELEROMETER UNIT</td>
<td>-</td>
<td>-</td>
<td>FRONT COVER</td>
<td>3D PRINTED</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FRONT ACCELEROMETER UNIT</td>
<td>-</td>
<td>-</td>
<td>L- RIGU BOLT</td>
<td>STAINLESS STEEL, 3 0.16 THREAD SIZE, 1.25</td>
<td>3</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FRONT ACCELEROMETER UNIT</td>
<td>-</td>
<td>-</td>
<td>M1205 brass inserts</td>
<td>AMPHENOL (COMMERCIAL PRODUCTS)</td>
<td>R-0157600101</td>
<td>Digikey</td>
<td>R-0157600101CT-ND</td>
<td>1</td>
</tr>
<tr>
<td>FRONT ACCELEROMETER UNIT</td>
<td>-</td>
<td>-</td>
<td>SCREW PHILLIPS, M6X0.56</td>
<td>STAINLESS STEEL WASHER; FOR 1/4&quot; SCREW SIZE, 0.281&quot; ID, 0.625&quot; OD</td>
<td>MCMASTER</td>
<td>90107A029</td>
<td>4</td>
<td></td>
</tr>
<tr>
<td>REAR ACCELEROMETER UNIT</td>
<td>ADXL375 BOARD</td>
<td>RJ45</td>
<td>M1205 brass inserts</td>
<td>AMPHENOL (COMMERCIAL PRODUCTS)</td>
<td>R-0157600101</td>
<td>Digikey</td>
<td>R-0157600101CT-ND</td>
<td>1</td>
</tr>
<tr>
<td>REAR ACCELEROMETER UNIT</td>
<td>ADXL375 BOARD</td>
<td>C3, C1</td>
<td>CAP CER 0.1UF 0V XTR 0065</td>
<td>KEMET</td>
<td>CON0C0016036BU777ACUL</td>
<td>Digikey</td>
<td>999-1104-1-ND</td>
<td>2</td>
</tr>
<tr>
<td>REAR ACCELEROMETER UNIT</td>
<td>ADXL375 BOARD</td>
<td>C2</td>
<td>CAP CER 0.1UF 0V XTR 0065</td>
<td>KEMET</td>
<td>CON0C0016036BU777ACUL</td>
<td>Digikey</td>
<td>999-1105-1-ND</td>
<td>1</td>
</tr>
<tr>
<td>REAR ACCELEROMETER UNIT</td>
<td>ADXL375 BOARD</td>
<td>ADXL375</td>
<td>ACCELEROMETER 200 DC/ SPI/ LG</td>
<td>ANALOG DEVICES</td>
<td>ADXL375DC</td>
<td>Digikey</td>
<td>AD-XL375DC/25</td>
<td>1</td>
</tr>
<tr>
<td>REAR ACCELEROMETER UNIT</td>
<td>-</td>
<td>-</td>
<td>REAR MOUNT BLOCK</td>
<td>3D PRINTED</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>REAR ACCELEROMETER UNIT</td>
<td>-</td>
<td>-</td>
<td>REAR COVER</td>
<td>3D PRINTED</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>REAR ACCELEROMETER UNIT</td>
<td>-</td>
<td>-</td>
<td>REAR MOUNT BLOCK</td>
<td>3D PRINTED</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>REAR ACCELEROMETER UNIT</td>
<td>-</td>
<td>-</td>
<td>REAR COVER</td>
<td>3D PRINTED</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>REAR ACCELEROMETER UNIT</td>
<td>-</td>
<td>-</td>
<td>M1205 brass inserts</td>
<td>AMPHENOL (COMMERCIAL PRODUCTS)</td>
<td>R-0157600101</td>
<td>Digikey</td>
<td>R-0157600101CT-ND</td>
<td>1</td>
</tr>
<tr>
<td>REAR ACCELEROMETER UNIT</td>
<td>-</td>
<td>-</td>
<td>SCREW PHILLIPS, M6X0.56</td>
<td>STAINLESS STEEL WASHER; FOR 1/4&quot; SCREW SIZE, 0.281&quot; ID, 0.625&quot; OD</td>
<td>MCMASTER</td>
<td>90107A029</td>
<td>4</td>
<td></td>
</tr>
<tr>
<td>REAR ACCELEROMETER UNIT</td>
<td>-</td>
<td>-</td>
<td>1/4&quot; WASHER</td>
<td>STAINLESS STEEL WASHER; FOR 1/4&quot; SCREW SIZE, 0.281&quot; ID, 0.625&quot; OD</td>
<td>MCMASTER</td>
<td>90107A029</td>
<td>4</td>
<td></td>
</tr>
</tbody>
</table>
Appendix C

MTB DAQ STATE TRANSITION DIAGRAM

Figure C.1: MTB DAQ state transition diagram.
Figure D.1: Main unit of the first MTB DAQ prototype mounted on frame with power on and test number 20 on display.
Figure D.2: Top view of main unit.

Figure D.3: Bottom view of main unit.
Figure D.4: Side view of the rear accelerometer unit.

Figure D.5: Rear view of the rear accelerometer unit.
Figure D.6: Side view of the front accelerometer unit.

Figure D.7: Main unit.
Figure D.8: Main board.

Figure D.9: Microcontroller and supporting circuitry of main board.
Figure D.10: Bottom of main board.

Figure D.11: Assembly of the main unit.
Figure D.12: Close up of the assembly of the main unit. Clearance is clearance!

Figure D.13: UI board assembly.
Figure D.14: Accelerometer board.
Table E.1: Raw and processed data from the air spring curve measurement test for the X-Fusion O2 Pro RL shock.

<table>
<thead>
<tr>
<th>Static Pressure [psi]</th>
<th>Measured Values</th>
<th>Calculated Values</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Displacement [in]</td>
<td>Force at Rear Wheel (Scale Reading) [lb]</td>
</tr>
<tr>
<td>60</td>
<td>0.000</td>
<td>18.2</td>
</tr>
<tr>
<td></td>
<td>0.110</td>
<td>36.4</td>
</tr>
<tr>
<td></td>
<td>0.404</td>
<td>59.6</td>
</tr>
<tr>
<td></td>
<td>0.789</td>
<td>89.0</td>
</tr>
<tr>
<td></td>
<td>1.256</td>
<td>112.8</td>
</tr>
<tr>
<td></td>
<td>1.400</td>
<td>134.8</td>
</tr>
<tr>
<td>95</td>
<td>0.000</td>
<td>18.6</td>
</tr>
<tr>
<td></td>
<td>0.137</td>
<td>59.8</td>
</tr>
<tr>
<td></td>
<td>0.445</td>
<td>93.4</td>
</tr>
<tr>
<td></td>
<td>0.676</td>
<td>119.4</td>
</tr>
<tr>
<td></td>
<td>1.075</td>
<td>146.6</td>
</tr>
<tr>
<td></td>
<td>1.434</td>
<td>194.2</td>
</tr>
<tr>
<td>140</td>
<td>0.000</td>
<td>18.8</td>
</tr>
<tr>
<td></td>
<td>0.037</td>
<td>54.4</td>
</tr>
<tr>
<td></td>
<td>0.162</td>
<td>96.0</td>
</tr>
<tr>
<td></td>
<td>0.316</td>
<td>115.8</td>
</tr>
<tr>
<td></td>
<td>0.514</td>
<td>132.2</td>
</tr>
<tr>
<td></td>
<td>0.666</td>
<td>151.0</td>
</tr>
<tr>
<td></td>
<td>0.966</td>
<td>187.6</td>
</tr>
<tr>
<td></td>
<td>1.217</td>
<td>222.0</td>
</tr>
<tr>
<td></td>
<td>1.357</td>
<td>243.8</td>
</tr>
</tbody>
</table>
Table E.2: Raw and processed data from the air spring curve measurement test for the Rockshox Recon TK Silver 27.5” fork.

<table>
<thead>
<tr>
<th>Static Pressure [psi]</th>
<th>Measured Values</th>
<th>Calculated Values</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Displacement</td>
<td>Force at Rear Wheel (Scale Reading)</td>
</tr>
<tr>
<td>50</td>
<td>0.000</td>
<td>20.0</td>
</tr>
<tr>
<td></td>
<td>0.743</td>
<td>36.0</td>
</tr>
<tr>
<td></td>
<td>1.431</td>
<td>49.6</td>
</tr>
<tr>
<td></td>
<td>1.993</td>
<td>68.4</td>
</tr>
<tr>
<td></td>
<td>2.243</td>
<td>85.4</td>
</tr>
<tr>
<td></td>
<td>2.743</td>
<td>105.8</td>
</tr>
<tr>
<td></td>
<td>3.181</td>
<td>134.2</td>
</tr>
<tr>
<td></td>
<td>3.493</td>
<td>143.4</td>
</tr>
<tr>
<td></td>
<td>3.743</td>
<td>151.4</td>
</tr>
<tr>
<td></td>
<td>3.993</td>
<td>171.0</td>
</tr>
<tr>
<td></td>
<td>4.243</td>
<td>188.2</td>
</tr>
<tr>
<td>80</td>
<td>0.000</td>
<td>19.0</td>
</tr>
<tr>
<td></td>
<td>0.431</td>
<td>47.8</td>
</tr>
<tr>
<td></td>
<td>0.868</td>
<td>61.8</td>
</tr>
<tr>
<td></td>
<td>0.931</td>
<td>74.8</td>
</tr>
<tr>
<td></td>
<td>1.243</td>
<td>94.6</td>
</tr>
<tr>
<td></td>
<td>1.931</td>
<td>107.0</td>
</tr>
<tr>
<td></td>
<td>2.243</td>
<td>130.6</td>
</tr>
<tr>
<td></td>
<td>2.431</td>
<td>139.2</td>
</tr>
<tr>
<td></td>
<td>2.931</td>
<td>153.6</td>
</tr>
<tr>
<td></td>
<td>3.118</td>
<td>170.8</td>
</tr>
<tr>
<td></td>
<td>3.431</td>
<td>186.8</td>
</tr>
<tr>
<td></td>
<td>3.618</td>
<td>202.6</td>
</tr>
<tr>
<td></td>
<td>3.743</td>
<td>225.4</td>
</tr>
<tr>
<td></td>
<td>3.993</td>
<td>234.2</td>
</tr>
<tr>
<td>112</td>
<td>0.000</td>
<td>19.2</td>
</tr>
<tr>
<td></td>
<td>0.431</td>
<td>65.2</td>
</tr>
<tr>
<td></td>
<td>0.493</td>
<td>72.6</td>
</tr>
<tr>
<td></td>
<td>0.806</td>
<td>96.6</td>
</tr>
<tr>
<td></td>
<td>1.118</td>
<td>120.4</td>
</tr>
<tr>
<td></td>
<td>2.181</td>
<td>157.8</td>
</tr>
<tr>
<td></td>
<td>2.368</td>
<td>190.2</td>
</tr>
<tr>
<td></td>
<td>2.868</td>
<td>215.6</td>
</tr>
<tr>
<td></td>
<td>3.243</td>
<td>220.0</td>
</tr>
<tr>
<td></td>
<td>3.431</td>
<td>244.4</td>
</tr>
<tr>
<td></td>
<td>3.556</td>
<td>269.6</td>
</tr>
</tbody>
</table>
The MATLAB® script Model_VALIDation.m uses functions from the MATLAB® Image Processing Toolbox™ in order to track the position of the markers on the bike. The script starts by reading the first frame of the video and saving it as a separate image. The image is then converted into grayscale, the haze is reduced, and the image is sharpened. The processed grayscale image is then converted into a binary image based on a threshold value. As the name implies, a binary image only contains completely white or completely black pixels. This is done in order to isolate the markers as much as possible from the other parts of the image. After this, excess noise is reduced by eliminating all areas of white below a certain value, only leaving the larger areas of white (including the markers) left in the image. Table F.1 depicts the images at various steps during the processing and the corresponding MATLAB® functions used to process them. Finally, the centroids of the remaining white areas are found. The centroids for the two markers are saved separately. These values, along with the original video file, are passed to a point tracker MATLAB® function. This function then steps through the video and tracks the markers through each frame. Once the markers have been properly located, they can be used to calculate the position and pitching angle of the CG of the bike and rider using geometry.
Table F.1: Image processing steps used to determine the position of the markers on the bike.

<table>
<thead>
<tr>
<th>Step</th>
<th>MATLAB® Function</th>
<th>Corresponding Image</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td><code>readFrame</code></td>
<td><img src="image1.jpg" alt="Step 1" /></td>
</tr>
<tr>
<td>2</td>
<td><code>rgb2gray</code></td>
<td><img src="image2.jpg" alt="Step 2" /></td>
</tr>
<tr>
<td>3</td>
<td><code>imadjust</code>, <code>imreducehaze</code>, <code>imsharpen</code>, <code>graythresh</code>, <code>imbinarize</code></td>
<td><img src="image3.jpg" alt="Step 3" /></td>
</tr>
<tr>
<td>4</td>
<td><code>bwareaopen</code></td>
<td><img src="image4.jpg" alt="Step 4" /></td>
</tr>
</tbody>
</table>
## Appendix G

### TESTING DOCUMENTATION

<table>
<thead>
<tr>
<th>Test Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Date</td>
<td>2020.07.19</td>
</tr>
<tr>
<td>Time</td>
<td>11:47</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Test No.</th>
<th>Front Pressure</th>
<th>Rear Pressure</th>
<th>X-Fusion G2 Pro Pressure</th>
<th>Rockshox TK Silver Pressure</th>
<th>C hỗn</th>
<th>Clicks</th>
<th>Offset</th>
<th>Course Name</th>
<th>Notes</th>
<th>Log</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>data1</td>
<td>30</td>
<td>100</td>
<td>150</td>
<td>5</td>
<td>4</td>
<td>[psi]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>data2</td>
<td>30</td>
<td>100</td>
<td>150</td>
<td>5</td>
<td>4</td>
<td>[psi]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>data3</td>
<td>30</td>
<td>100</td>
<td>150</td>
<td>5</td>
<td>4</td>
<td>[psi]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>data4</td>
<td>30</td>
<td>100</td>
<td>150</td>
<td>5</td>
<td>4</td>
<td>[psi]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>data5</td>
<td>30</td>
<td>100</td>
<td>150</td>
<td>5</td>
<td>4</td>
<td>[psi]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>data6</td>
<td>30</td>
<td>100</td>
<td>150</td>
<td>5</td>
<td>4</td>
<td>[psi]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>data7</td>
<td>30</td>
<td>100</td>
<td>150</td>
<td>5</td>
<td>4</td>
<td>[psi]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>data8</td>
<td>30</td>
<td>100</td>
<td>150</td>
<td>5</td>
<td>4</td>
<td>[psi]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>data9</td>
<td>30</td>
<td>100</td>
<td>150</td>
<td>5</td>
<td>4</td>
<td>[psi]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>data10</td>
<td>30</td>
<td>100</td>
<td>150</td>
<td>5</td>
<td>4</td>
<td>[psi]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>data11</td>
<td>30</td>
<td>100</td>
<td>150</td>
<td>5</td>
<td>4</td>
<td>[psi]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>data12</td>
<td>30</td>
<td>100</td>
<td>150</td>
<td>5</td>
<td>4</td>
<td>[psi]</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Figure G.1: Filled-in testing log, page 1/2.
<table>
<thead>
<tr>
<th>Test No.</th>
<th>File-Name</th>
<th>Tire Pressure</th>
<th>Rockshox TK Silver 27.5&quot; Fork</th>
<th>X-Fusion O2 Pro RL Rear Shock</th>
<th>Zero g Offset Period</th>
<th>Course Name</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>[-]</td>
<td>[psi]</td>
<td>[psi]</td>
<td>[psi]</td>
<td>[-]</td>
<td>[-]</td>
<td>[-]</td>
</tr>
<tr>
<td>13</td>
<td>data13</td>
<td>30</td>
<td>30</td>
<td>100</td>
<td>2</td>
<td>150</td>
<td>5</td>
</tr>
<tr>
<td>14</td>
<td>data14</td>
<td>30</td>
<td>30</td>
<td>100</td>
<td>2</td>
<td>150</td>
<td>5</td>
</tr>
<tr>
<td>15</td>
<td>data15</td>
<td>30</td>
<td>30</td>
<td>100</td>
<td>2</td>
<td>150</td>
<td>5</td>
</tr>
<tr>
<td>16</td>
<td>data16</td>
<td>30</td>
<td>30</td>
<td>100</td>
<td>2</td>
<td>150</td>
<td>5</td>
</tr>
</tbody>
</table>
Appendix H

TESTING RESULTS

The following section contains plots comparing the modeled response of the bike and rider with the test settings initially used on the course and those recommended by the genetic algorithm. Each trial has three plots. The first plot shows the modeled vertical response of the CG of the bike and rider over time as well as the input positions of the front and rear axles over time. The second plot shows the same data but without the input positions. This plot is essentially a “zoomed in” view of the first and is included to give better insight into what the behavior of the response looks like. The final plot is a plot of the pitching angle of the CG of the bike and rider with respect to the horizontal for all time. Each figure corresponds to a different course pictured in Figure 8.1 with corresponding test data summarized in the testing logs shown in Figures G.1 and G.2. They are as follows.

- Figure H.1 - Small Wooden Jump Course
- Figure H.2 - High Frequency Course
- Figure H.3 - Large Blunt Obstacle Course
- Figure H.4 - Large Rolling Bump Course
- Figures H.5 and H.6 - Dirt Jump Course
Figure H.1: Modeled responses of the data taken from the small wooden jump course, tests 1–3.
Figure H.2: Modeled responses of the data taken from the high frequency course, tests 4–6.
Figure H.3: Modeled responses of the data taken from the large blunt obstacle course, tests 7–9.
Figure H.4: Modeled responses of the data taken from the large rolling bump course, tests 10–12.
Figure H.5: Modeled responses of the data taken from the dirt jump course, tests 13–15.
Figure H.6: Modeled responses of the data taken from the dirt jump course, test 16.
NOTES ON THE JACKALOPE

Similar to the Diamond-Toothed-Rock-Burrowing-Gopher, the American Jackalope (*Lepus cornutus americanum*) is a small ferocious critter that has evaded the grasps of American biologists for centuries [20]. While one has never been captured alive in the wild, eye-witness accounts best describe the varmit as a jackrabbit with the horns of an antelope. Thus the “jackalope” is a portmanteau of the two animal names. Like other horned creatures, jackalope bucks use their antlers for sparring with other breeding males to establish dominance [20]. Figure I.1 contains excerpts from the leading study on the species, *The Field Guide to the North American Jackalope* by Andy Robbins.

![American Jackalope (*Lepus cornutus americanum*)](image)

The most well-known of the jackalope subspecies, the American Jackalope is found from the forests of Saskatchewan to the central grasslands of the United States, as far east as Illinois and as far west as California. Measures 22-26 inches in length, with a white tail several inches long. Color varies from dark brown to greyish-brown with a pale grey underside. Weight varies depending on latitude, from 5 to 10 pounds, and up to 15 pounds during record wild berry seasons. Males annually grow a pair of branched antlers, beginning in spring and lasting until midwinter, when they are shed.

Geographical Distribution

(a) Notes on the American Jackalope.  
(b) Two bucks sparring.

Figure I.1: Excerpts from *The Field Guide to the North American Jackalope* by Andy Robbins [20].