Cyclist Injury Prevention Challenge: Final Design Report

Submitted to:

The Chad William Young Foundation
Newmarket, NH 03857

ATTN: Kevin and Louis Young

Submitted by:

S18-13
Team Cruise Control
College of Engineering and Computational Sciences
Colorado School of Mines
Golden, Colorado 80401

Capstone Design@Mines

Final Design Report

Team Members:
Canaan Forslund (ctforslund@mines.edu)
James Frazar (jfrazar@mines.edu)
Kevin Miller, Team Leader (kevinmiller@mines.edu)
Hunter Nelson (hnelson@mines)
Thomas Staver (tstaver@mines.edu)
Faculty Adviser: Emily Sievers (sievers@mines.edu)

November 30th, 2018

Acknowledgments

Team Cruise Control would like to express their deep gratitude to The Chad William Young Foundation, our client, for enabling us to work on this project. Their guidance, encouragement, and useful critiques throughout the development process were instrumental to our success.

We would also like to thank Mrs. Emily Sievers for her thoughtful input and patients when working with students. Without you all, this project would not have succeeded.

Furthermore, we would like to thank everyone in the cycling community who volunteered their time to provide feedback for this project and make it the best it could be.

Lastly, to all the professors who taught us the skills required to think critically and tackle challenges, this project is a reflection of your work with us. We hope you find it as remarkable as we do.

Table of Contents

Acknowledgments **1**

Executive Summary **6**

1 Introduction **7**

1.1 The Chad William Young Foundation 7

1.2 The Team 7

1.3 Project Background and Problem Statement 7

1.4 Scope 9

1.5 Report Overview 10

2 Product Concept Creation **11**

2.1 Data Collection Methods 11

2.2 Data Boundaries & Characteristics 12

2.3 Department of Transportation Data 12

2.4 Manual Data Collection 13

2.5 Automatic Strava Data Collection 15

2.6 Cyclist Outreach 16

3 Product Operation **23**

3.1 Inputs 23

3.2 Automatic Algorithm Execution 25

3.2.1 Turning Radius Calculations 26

3.2.2 Velocity Calculations 29

3.2.3 Acceleration Calculations 31

3.2.4 Grade Calculations 33

3.3 Manual Algorithm Execution 35

3.4 Determination of Hazard Multipliers 35

3.4.1 Automatic Hazard Multipliers 36

3.4.2 Manual Hazard Multipliers 37

3.5 Code Walkthrough 38

4 Product Outputs 42

4.1 Hazard Profile 42

4.2 Hazard Profile Importance 43

4.3 Instruction on how to use Outputs 44

5 Testing 45

5.1 Testing Methods 45

5.2 Testing Results 46

6 Project Management 47

6.1 Work Breakdown Structure 47

6.2 Project Schedule 48

6.3 Sprint Architecture 49

6.4 Budget 50

6.5 Unfinished Items 50

7 Recommendations for Future Development 51

7.1 Algorithm Interface 51

7.2 Data Collection Automation 51

7.3 Real-time Feedback 51

7.4 Additional Dataset Incorporation 53

7.5 Recommended Skill Sets 53

8 Lessons Learned 54

9 Conclusion 57

Appendix 58

References 75

List of Figures

Figure 1: Example Strava Page for the Lookout Mountain Segment [8] 16

Figure 2: Causes of Cycling Crashes 17

Figure 3: Information Cyclists Want Before Riding 18

Figure 4: Road Conditions That Are Hazardous to Cyclists 19

Figure 5: Methods to Improve Cycling Event Safety 20

Figure 6: Methods to Improve Recreational Cycling Safety 20

Figure 7: Research That Cyclists Do Before Riding 21

Figure 8: Sample of GET Request Data From Strava 24

Figure 9: Latitude and Longitude plot for Lookout Mountain 29

Figure 10: 3-Point Average Velocity Data For Lookout Mountain 31

Figure 11: Acceleration Vs. Distance Plot for Lookout Mountain 32

Figure 12: Percent Grade of Lookout Mountain Ride 34

Figure 13: Elevation Profile Example of Lookout Mountain Ride 34

Figure 14: Product Operation Flow Chart 38

Figure 15: Latitude and Longitude Hazard Profile for Lookout Mountain 42

Figure 16: Altitude vs Distance Hazard Profile for Lookout Mountain 43

Figure 17: Lookout Mountain Test Figure 46

Figure 18: Possible Methods of Real-Time Feedback 52

List of Tables

Table 1: Client Needs Table 9

Table 2: Description of Numerical Weight Scale 36

List of Appendix Items

Figure A1: Updated Work Breakdown Structure Sections 1 & 2 **58**

Figure A2: Updated Work Breakdown Structure Sections 3 & 4 **59**

Figure A3: Updated Work Breakdown Structure Sections 5 & 6 **60**

Figure A4: Updated Project Schedule Table **61**

Figure A5: Updated Project Schedule Gantt Chart **62**

Figure A6: Sprint Layout 1 & 2 Tasks **63**

Figure A7: Sprint Layout 3 & 4 Tasks **63**

Figure A8: Sprint Layout 5 & 6 Tasks **64**

Figure A9: Sprint Layout 7 Tasks **64**

Figure A10: Code Overview - Data Input and Initialization **64**

Figure A11: Code Overview - Multiplier Assignment and Initialization **65**

Figure A12: Code Overview - Creating Latitude and Longitude Arrays **66**

Figure A13: Calculating Velocity and Acceleration Arrays **67**

Figure A14: Calculating Grade Array **67**

Figure A15: Calculating Turning Radius Array Part 1 of 2 **68**

Figure A16: Calculating Turning Radius Array Part 2 of 2 **68**

Figure A17: Assigning Automatic Multiplier Values Part 1 of 3 **69**

Figure A18: Assigning Automatic Multiplier Values Part 2 of 3 **70**

Figure A19: Assigning Automatic Multiplier Values Part 3 of 3 **71**

Figure A20: Assigning Manual Multiplier Values **72**

Figure A21: Creation of Colormap **73**

Figure A22: Plotting of Hazard Profile **73**

Figure A23: Plotting Various Route and Rider Characteristics **74**

Executive Summary

Cycling is a sport enjoyed all around the world by recreationalists and professionals alike. Cycling involves everything from basic transportation purposes, to some of the world's largest and longest sporting events. Unfortunately, even for how popular cycling is, it has inherent dangers. Tragically, every year in the United States there are thousands of crashes involving cyclists, hundreds of which are fatal accidents [1]. It is clear that better safety devices are needed to prevent such crashes. However, it is also clear that traditional physical safety devices such as helmets are not always effective at preventing fatal injuries sustained during crashes. Therefore, the best way to truly prevent serious or fatal injuries is to prevent cycling crashes altogether.

For the Chad William Young Foundation, this is the goal. Chad Young was a professional cyclist who sustained fatal injuries during the final stage of the Tour of the Gila race in 2017. Therefore, Chad's family established the Foundation in part with this goal in mind, to create a product that would focus on preventing crashes altogether, rather than simply mitigating the damage of the crashes. To help in this pursuit, the Foundation set up a project for the Senior Design Capstone program at the Colorado School of Mines in the fall and spring of 2018. Subsequently, Team Cruise Control decided that it would tackle this project, with the goal of designing a product that could alert cyclists to dangerous portions of their route before they start their ride.

After taking time to determine the scope of the project, the first obstacle that Team Cruise Control faced was collecting the proper data needed to determine how dangerous a route is. While the Team considered multiple methods of data collection, it was decided that online applications that allow cyclists to publicly post their own data would provide the perfect source. After the Team had cyclist data to work with, the next task was to determine methods of analyzing this data for potential hazards the cyclists could face. During this process, the Team reached out to many cyclists to determine their thoughts on what obstacles are most hazardous to cyclists, and what they would like to be warned about ahead of time. The Team also went through an extensive process involving research and physics analysis to use the collected data to determine such hazards. Once these hazards were determined, they needed to be weighted by their severity among the various different types of hazards that were predicted. The Team decided that using a system of multipliers which increased depending on the severity of the hazard would provide the most accurate results. After the total, final hazard value was obtained at each discrete point of data along the route, the Team constructed plots which showed the route colorized depending on the severity of the hazard along each segment. Several of these plots were assembled into a route packet which can clearly and concisely convey to cyclists dangerous sections of their route. Knowing this information ahead of time should allow cyclists to be more cognizant of these hazards while they are riding, which should prevent crashes and the resulting injuries, thereby making cycling more safe for everyone who enjoys the sport.

1 Introduction

1.1 The Chad William Young Foundation

The Chad William Young Foundation was established by Chad's father, mother, and brother, Kevin, Lois, and Evan Young. The Foundation was created after Chad William Young passed due to a traumatic brain injury sustained during the Tour of the Gila professional cycling race. The Foundation was formed in part to help provide support, direction, and funding for developments centering on preventing injuries to cyclists, especially traumatic brain injuries. In particular, the Foundation is interested in developing a technological solution to cyclist safety that aims to prevent crashes, rather than lessen their effects. This is important, because many of the most severe cyclist injuries, including Chad's, are not preventable with physical safety devices like helmets. Therefore, the Foundation created a project for the Senior Design Capstone program at the Colorado School of Mines so that a few of Chad's fellow students could create such a technological solution. The Foundation is the client for this project [2].

1.2 The Team

The Senior Design Team that was selected for this project consists of five mechanical engineering seniors at the Colorado School of Mines. These students are Canaan Forslund, James Frazar, Kevin Miller, Hunter Nelson, and Thomas Staver. While each student on the team works on all aspects of the project equally, each person has a different role that was decided on by the Team. Kevin Miller is the project manager, who is responsible for the high-level project planning and keeping the Team on track. James Frazar is the administrative lead, who is responsible for organizing the Team's documentation and booking any necessary resources for the Team. Canaan Forslund is the communication lead, who is responsible for all external communication between the Team and the faculty advisor or client. Hunter Nelson is the coding technical lead, who is responsible for overseeing the coding of the algorithm used within the technological solution created by the Team. Lastly, Thomas Staver is the research technical lead, who is responsible for researching external information needed by the Team. Together, these five students make up Team Cruise Control, who have been tasked by the Foundation to create a technical solution to increase cyclist safety.

1.3 Project Background and Problem Statement

Evidence throughout the cycling world has exposed an inadequate level of safety surrounding riders, for both professional and recreational riders. Various hazardous route conditions, especially at the speeds reached during cycling races, have contributed to cyclist accidents resulting in injury and death. Sometimes these route conditions are deceivingly dangerous,

making such hazards hard to determine ahead of time for riders. This means that even the best cyclists can get into dangerous situations when riding these routes.

In the case of Chad Young, his was a situation where the hazards were not immediately evident to the perception of riders and race organizers. In an article describing the crash, Laura Weisly said the crash “happened on a section of the course that seemed relatively innocuous, rather than the steep, tight switchback from Pinos Altos that has taken out so many riders in the past.” Another rider’s manager who was involved in the crash said "The crash itself was really no different or remarkable to any other, a small bunch of riders came into the corner, it was a bit deceiving, they realised they'd misjudged it and in the correction, lost control and crashed” [3].

Chad is not the first. Other riders have suffered similar fates due to improperly conveyed hazards on their race courses. It seems as though certain route segments are more dangerous than what can be ascertained from a snap-judgment or traditional route manual. Traditional race manuals describe the course with simple geographical descriptions. They make references to sharp, tight corners, or fast downhill descents. However, cyclists have encountered many such situations in the past, and it can be hard to tell what makes deceptively dangerous sections different from all of the rest.

This is why the Foundation has enlisted the help of Team Cruise Control in order to create a different kind of solution to the traditional race manual. This solution is based on mathematics, physics, and other cyclist’s data and prior experiences. It takes geographical road conditions and average rider data for a given cycling route, and determines where there are segments of the route that are unsafe. To accomplish this, Team Cruise Control has coded an algorithm that can take the aforementioned data and run it through a series of equations based on the dynamics and physics of a cyclist. The algorithm determines whether or not there are dangerous segments throughout a route by first determining the characteristics of the rider throughout the route, and then analyzing these characteristics to determine whether or not they are safe for the given route conditions. The dangerousness of a route can then be ranked on a 1-5 scale with a 5 being the most hazardous situation a rider could face. This information is easily communicated to the rider in the form of a route packet and hazard profiles that show the dangerous sections and their severity in an easily readable, colorized map.

The goal of addressing rider safety in this manner was to create a safety system that was based in mathematics and physics, and not just observations or prior history. As long as the appropriate data is available, this system can be used for professional and recreational riders alike on routes all over the world. The Chad Young Foundation and Team Cruise Control hope that this system will help improve cyclist safety by creating the basis for a system that can be built on and improved over time.

1.4 Scope

As previously discussed, the Foundation have made it evident that a technological, rather than physical, safeguard solution is necessary. This is because while physical safeguards are still very important for general protection, many times they are not able to prevent fatal injuries in serious crashes. Therefore, this project will be constrained to just a technological, preventive solution. While it was acceptable for the Team to make recommendations for the hardware cyclists use to collect data or monitor their status, the design of such hardware was not within the initial scope of this project. Instead, this project was limited to collecting existing cyclist data, creating a numeric scale with which to rate the dangerousness of a route, and performing hazard analysis to understand which situations are dangerous for cyclists. Then creating an algorithm to autonomously perform this hazard analysis, and finally, create a route packet can easily convey these hazards to cyclists before they ride a route.

The deliverables for this project very closely follow this scope. Almost every major development during the project is listed as a deliverable for the client. Therefore, the goals listed above for the scope of the project are outlined in Table 1 below.

Table 1: Client Needs Table

Need #	***Need Statement (Deliverable)***
1	Research available data on road and rider metrics that may be useful in the determining the danger of a give route and preventing cyclist injuries.
2	Develop a theoretical method of evaluating risk at discrete locations based on rider and road characteristics using dynamics.
3	Expand the risk evaluation method to evaluate risk along an entire route using an algorithm to process different routes and conditions easily and quickly.
4	Use risk evaluation method to inform race organizers of hazards and risks along a specified race route.
5	Adapt race organizers hazard report for use by professional and recreational cyclists which details dangerous sections of a route and their severity.
6	Final design report of the project's progress and status after two semesters.

As such, the development of the project was constrained in the following ways:

- The product was limited to a technical solution, not a physical one.
- The output of the final product was a risk assessment for a discrete route.
- The project utilized pre-existing data where possible rather than focus on developing new data collection methods.
- The technological solution was implementable across several different and varying cycling routes.
- Early development took into consideration future plans to integrate with a real-time feedback device.
- Development of physical hardware was limited to devices in which the main function is to provide cyclists with feedback.
- All development complied with the rules listed in both the 2017 USA Cycling Rulebook [4] and the UCI rules and regulations [5] such that cyclists can use the technology during road races.
- There were no other federal/state laws or regulations that needed to be complied with.

The only external constraints imposed on this project were those imposed by various cycling associations so that the final product can be used with professional cyclists during races. Otherwise, the only constraints on the project were those imposed by the client while defining the scope. Additionally, research did not show any patents that the Team needed to be cognizant of or work around in this area. Since the output of this project focused on information and technology, the Team did not incur any expenses. Therefore, there were no budgetary constraints to worry about for this project. Overall, these were the limitations which Team Cruise Control worked within to develop the technological solution for the Foundation.

1.5 Report Overview

The purpose of this final design report is to give an overview of the entire project from start to finish. In doing so, this report will restate some of the information that has been presented previously in the concept portfolio and intermediate design report. Overall, the goal of this report is to be a single document that completely describes the product and its operation. As such, this report will describe the external research the Team conducted for this project, including different sources of data collection and outreach to cyclists. Then this report will go through the product itself, describing how it works and the outputs that it produces. This report will then touch on the testing regime that was conducted in order to validate the product, and the creation of the route packet that conveys this information to the cyclists. Finally, the report will conclude with the discussion of a few different topics including project management items, future development of this product, and lessons learned while working through the project. This should give the reader a complete and thorough understanding of the product and how it operates.

2 Product Concept Creation

2.1 Data Collection Methods

There are several methods that can be used to help prevent cyclists from crashing. Improved safety gear is one that has already been thoroughly explored. Top of the line helmets have been developed to protect riders from brain damage. Other padding such as elbow and knee pads also serve to protect the rider from crashes. These methods however, have their limits and cannot prevent all injuries from occurring. Additionally, they become cumbersome to the rider, especially those trying to achieve aerodynamic advantages in a race. As such, those who wear the least amount of protective gear are racers who are in the most danger of suffering an injury, which is why additional safety measures such as our product are necessary.

There is information available for most professional cycling races that outlines routes and provides safety information. Yet, it has been determined that the information provided in these race brochures is vague and non-specific. For example, in the 2017 “Tour of the Gila” race, the hazard information provided to riders ahead of the event was very short and did not provide much specific information. The packet largely warned of cattle guards, railroad crossings, areas of two-way traffic, and large descents [6]. However it only listed what segments of the race these hazards appeared in rather than specifically calling out the location of each one. This is not sufficient enough to properly convey the various safety hazards that compound when a rider encounters, for example, a steep descent mixed with a tight turn on a road with high camber.

Overall, even though safety gear and race packets have improved over recent years, the issue of serious and/or fatal accidents in cycling is still a persistent problem. This can be seen by the fact that cyclist fatalities in the United States have increased by 6% from 2006 to 2015 [7]. While these statistics do include riders that are hit by cars, something that this algorithm will not be able to protect against, these incidents only account for 29% of the overall number of fatalities [7]. The other 71% of fatalities are caused by road conditions or rider error, and this number is increasing. Therefore, it is obvious that safety gear alone is not solving the issue of cyclist safety and that the only way to completely prevent injury is to prevent a crash from occurring at all.

As such, the Team’s initial research was devoted to finding ways of collecting data that can provide cyclists with specific hazard information. Without better data collection, it would be impossible to give any more detailed information than what cyclists are already receiving before races. Therefore, the Team strove to find a primary source of data collection that would provide enough information about the route characteristics to predict all possible hazards. Three different methods were researched which included collecting data with departments of transportation, manually, or from an online cyclist tracking application. This research and these methods are discussed in the following sections.

2.2 Data Boundaries & Characteristics

Instead of mitigating damages after a crash occurs, preventing crashes from occurring at all is the goal of this project. The main tool to this end, rather than protective hardware or route safety, is information and data processing. Using an in depth and robust systematic analysis of route segments, safety can be achieved through the realm of prior planning. The three methods of data collection which were considered are quite different, yet their characteristics and how they fit into the overall project were all similar. Specifically, what the inputs and outputs of project should be.

2.3 Department of Transportation Data

In order to develop an algorithm that could be used to determine whether or not a particular cycling route poses any dangers to cyclists, the road data for the route must first be collected. Road characteristics such as road surface type, turn radius, elevation gain, as well as other properties are all important characteristics that the Team sought for in a data collection method in order to assess the dangers of a route. When it comes to collecting this data, the first method the Team explored was collecting data from departments of transportations.

State departments of transportation are the authority when it comes to most roadwork projects, and as such were thought to be the most likely place to find specific data on road specifications in their respective states. In addition, local departments of transportation are often responsible for determining dangerous sections of road. For example, they are responsible for placing warning signs and mitigating dangers with guardrails and other devices.

This concept for data collection was intriguing because it took advantage of data that had already been gathered by other organizations, which in this case, is the government. However, since every state has a department of transportation, this data was not centrally located in the United States. In addition, while many foreign countries likely have similar systems where it could be possible to gather data, there was no feasible plan for how a system like this would be implemented by a small team working on a new product. Furthermore, the largest issue with this method of data collection was that DOT's gather data in different ways using different metrics and mediums. For example, some DOT's handle assessing danger on a road by road basis, and do not collect or store actual road data anywhere within their system. The following is a short list of the pros and cons considered in relation to this method of data collection.

Pros:

- Determination of risk is more accurate when based upon specific road data
- Information collected by DOT's, as public agencies, allows for the public acquisition of data for the algorithm if the data is in an accessible format

- Ensures up-to-date information of specific trails without the need for additional or extraneous inspections of said routes
- DOT's are an authority on road conditions and safety; therefore, any information or data that comes from DOT's should be relevant and accurate

Cons:

- Guardrail/safety equipment placement may not be relatable to cycling analysis as it may have depend on different speeds, opposing traffic location, and what the conditions are surrounding the roadway
- Different DOT's across the country may not collect the same data, nor present their data in the same format which would make implementation of their data nationwide difficult
- DOT's may not have a large database of stored road data, and if they do, it may not be in a very accessible format
- In addition, collecting data from DOT's would require the Team to get in contact with all 50 DOT's (only one would be required for a proof of concept, but all 50 would be required for implementation)
- DOT data becomes obsolete over time and requires regular updating

Overall, as these pros and cons show, while it may have been beneficial to work with a department of transportation because they are knowledgeable on this subject, it was also hard to use this method as a legitimate data collection source. The ultimate barrier to the Team was establishing solid lines of contact within a department of transportation. However, while this method was ruled out for the scope of this project, it still possesses potential to be used in future development of the product.

2.4 Manual Data Collection

The algorithm developed for this project is designed to process and utilize the copious amounts of data that is already available to the public. However, it was thought that sometimes it is useful to broaden the scope of focus for prototyping and testing concepts. While data collection and extraction can be a cumbersome task in and of itself, there is no reason why a small sample of physical data can't be manually collected to further the process of building the algorithm.

With the use of a microcontroller and sensors, data such as road surface conditions, road camber, and potholes can be gathered in addition to the speed and elevation data that is collected with Strava. An Arduino microcontroller is a modular device that makes it relatively simple to collect several different strings of data at once. For example, a gyroscope and accelerometer can be connected to an Arduino that will identify the layout of different curves and road characteristics. Variable impact sensors are available and can be attached to the head tube of a bike that can

trigger when a specified impact force is encountered. However, this data collection can also be done by hand by recording a specific hazard, its severity, and gps coordinates.

The advantage of gathering data manually is that it allows the collection of prudent data which cannot be obtained by other means. While gathering data from a tracking application or through a department of transportation requires using whatever data is given, collecting data manually allows the Team to collect exactly what data is needed. This is beneficial because some road characteristics like camber or road surface are not tracked within Strava. In addition, when data is collected manually, the operators of the algorithm have full rights to use the data, and have it in a format that is easily accessible and usable. Therefore, while collecting data manually is more tedious and time consuming, it is a good way to expand, verify, and test the algorithm because all necessary data is collected and readily usable.

Simply receiving data en masse from Strava or other sources is ideal because it would allow the algorithm to be implemented across sections of road nationwide and automatically. Unfortunately, this information is difficult to find. With the manual data collection method, enough data can be collected from a single route to fully develop a safety algorithm that ensures the capability of the algorithm. However, extending this same level of detail to routes all over the nation will be time consuming and difficult. The following is a list of pros and cons of collecting data manually.

Pros:

- Any data that is deemed necessary to determine the danger of a cycling route can be collected
- Data is owned by the collector and is in an easily accessible format
- has the potential to produce a robust database of road condition data that does not currently exist
- Data can be gathered in variable sections, which is preferable for individualizing routes
- Provide very accurate inputs into the algorithm

Cons:

- Data has to be collected for every route in the world manually, this will be a long and tedious process
- Users that collect data will likely need to have access to some special equipment and sensors
- The algorithm cannot be implemented to areas that manual data has not been collected

Manual data collection is not ideal for ensuring the usability of the product anywhere in the world, however it has the potential to provide the most valuable data of any of the methods.

2.5 Automatic Strava Data Collection

The leading concept for data collection initially pointed towards using an application programing interface (API) to pull public data from cycling applications such as Strava or Training Peaks. These applications are used by millions of cyclists all over the world. Cyclists collect data themselves while riding, and the data is then uploaded to these various applications where the data can be viewed publicly. However, upon further investigation, it was found that performing a GET request for specific segments in the Strava database proved more effective at providing useful data. The GET request allows the data from any individual cyclist who rode a specific route to be pulled from the internet. Using the GET request is the primary method of data collection in the algorithm because it allows data to be pulled for almost any route instantly. This is the ture advantage of collecting data from sites like Strava, there is a nearly unlimited wealth of data from riders of all skill levels for routes from all over the world. This data is easy and fast to collect, and it comes in a relatively easy to use format. At this time there is simply not an alternative that provides the same wealth of data and easability of use that Strava does. For these reasons, the Team has selected it as the primary means of data collection for this project.

Figure 1 shows an example Strava page from the Lookout Mountain Hill Climb. This figure shows the data that Strava has the potential to store, which is the same data that the GET request provides if it is available. In particular, every second, Strava always tracks the riders location, elevation, time, and distance. Some data such as power, heart rate, and temperature are available from riders who have the equipment to track such things. For more details on how the GET request is used within the algorithm, see section 3.1.

Collecting data from Strava is unique because it is the only data collection method that collects data from cycling routes all over the world from riders of every skill level. This provides the opportunity to not only have one place where data can be gathered for a segment anywhere in the world, but also to compare data between riders of all skill levels. Yet, there are still various pros and cons of working with Strava. These pros and cons are listed below.

Pros:

- Can gather data from Strava practically instantaneously at any time
- Data will be repeatedly updated with time, and can be continually gathered for the most current data
- Rider data from routes all over the world can be gathered
- Data from different riders of varying skill from recreational to professional can be gathered
- Strava is a public website with free access

Cons:

- Strava's support has been unresponsive which makes asking questions about the GET request or further development with them difficult
- The data available is organized by individual riders making it more intensive to gather enough data to get a good rider distribution.

The overwhelming number of pros for using the Strava GET request is the reason why it was selected as the primary source for gathering data. However, the listed cons are not to be overlooked.

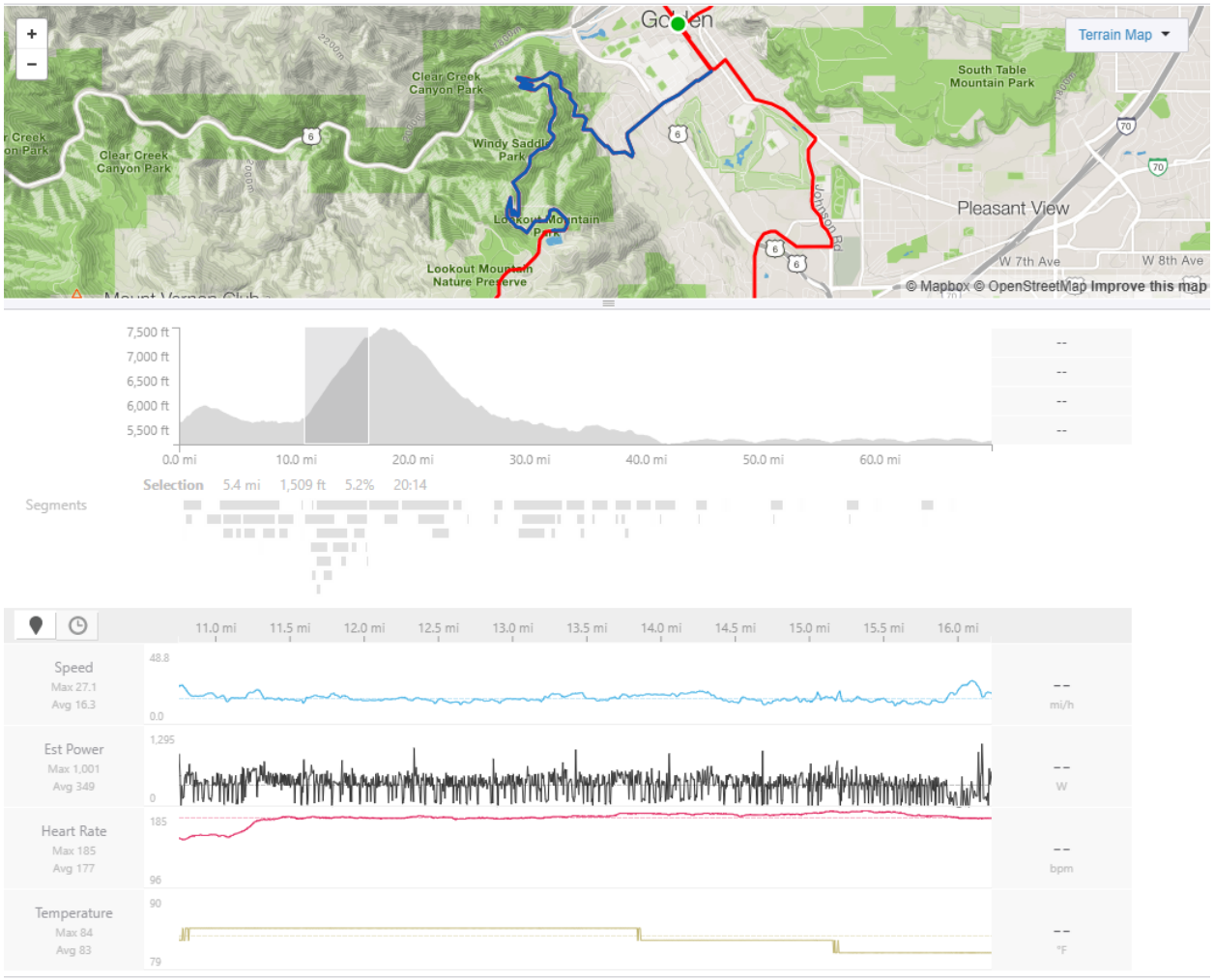

Figure 1: Example Strava Page for the Lookout Mountain Segment [8]

2.6 Cyclist Outreach

Upon reviewing the different options for data collection, it became obvious to the Team that more information was needed to determine what hazards were the most important to identify. As cyclists are the main stakeholders for the project, it was decided that reaching out towards cyclists directly would be the most effective way of gaining this information. As such, a survey

for cyclists was created in an effort to gain insight on their perspective of the dangers in the sport of cycling. The Team attended the Chad Young Memorial Time Trial on April 8th, 2018 and distributed the survey to as many cyclists at the event as possible. Presented here is the data from surveys taken by 96 cyclists of varying skill level. While the survey was conducted anonymously, the vast majority of the respondents likely participated in the time trial that day.

Beginning with skill level, just over half of the participants labelled themselves as recreational or hobbyist cyclists. In addition, 13.5% of them identified as collegiate cyclists, 14% of them identified as professional cyclists, and 7% of them were masters or elite (other than pro) riders. The rest of the participants fell into various smaller categories.

Participants were asked to describe accidents they've either been involved in or personally seen. The results are shown below in Figure 2. As can be seen, vehicle/cyclist interactions account for the biggest chunk of accidents. "Not Route Specific" indicates accidents that occurred as a result of anomalies such as dogs running in front of the cyclist or rider error. Combining several of the factors (descent, terrain, detours), about 20% of the accidents were caused by route specific conditions. Another roughly 13% were caused by riding in a group. For the full breakdown of the results, see the following figure.

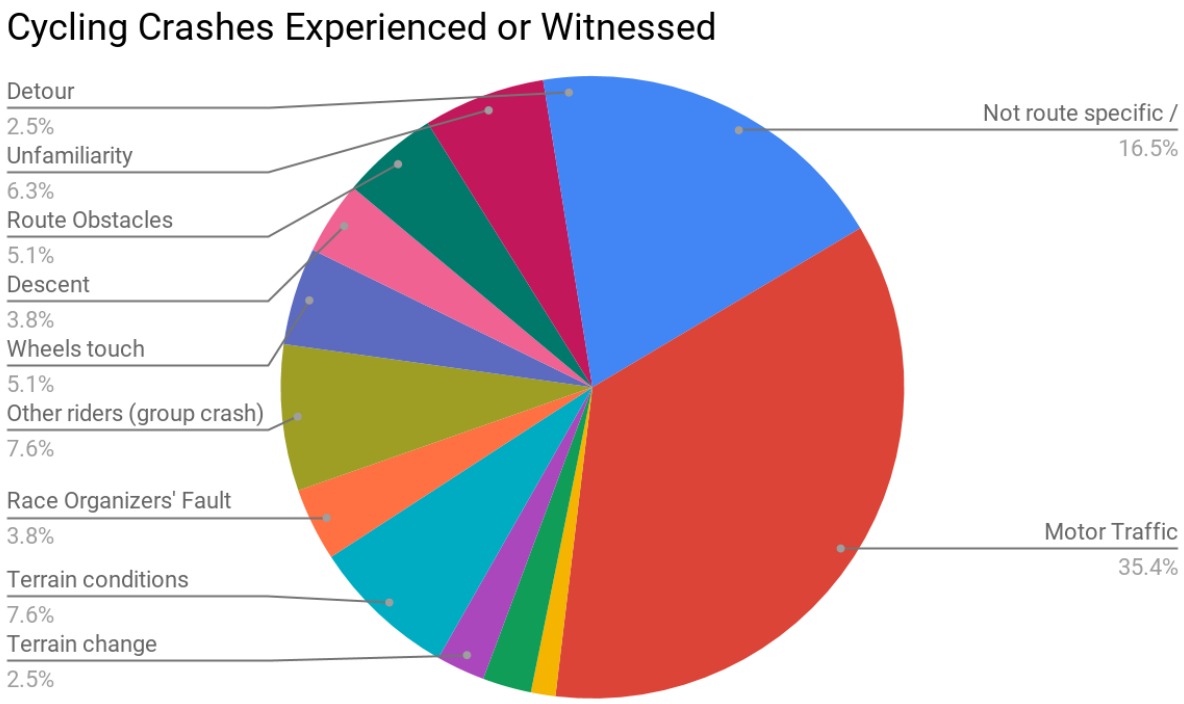

Figure 2: Causes of Cycling Crashes

Figure 3 shows what riders would like to know before embarking on a ride. Here, 15% of riders want information about when paths narrow. In addition, many cyclists also indicated that they want to know about road surface conditions and the presence of sharp turns. This was one of the most informative questions on the survey for feedback for the Team. Since the goal of the product is to warn cyclists about hazards they will face ahead of time, it is important to know and understand what it is that they want to be warned about. Therefore, the results shown in Figure 3 directly impacted the hazards that the Team decided it would be important to track in the algorithm.

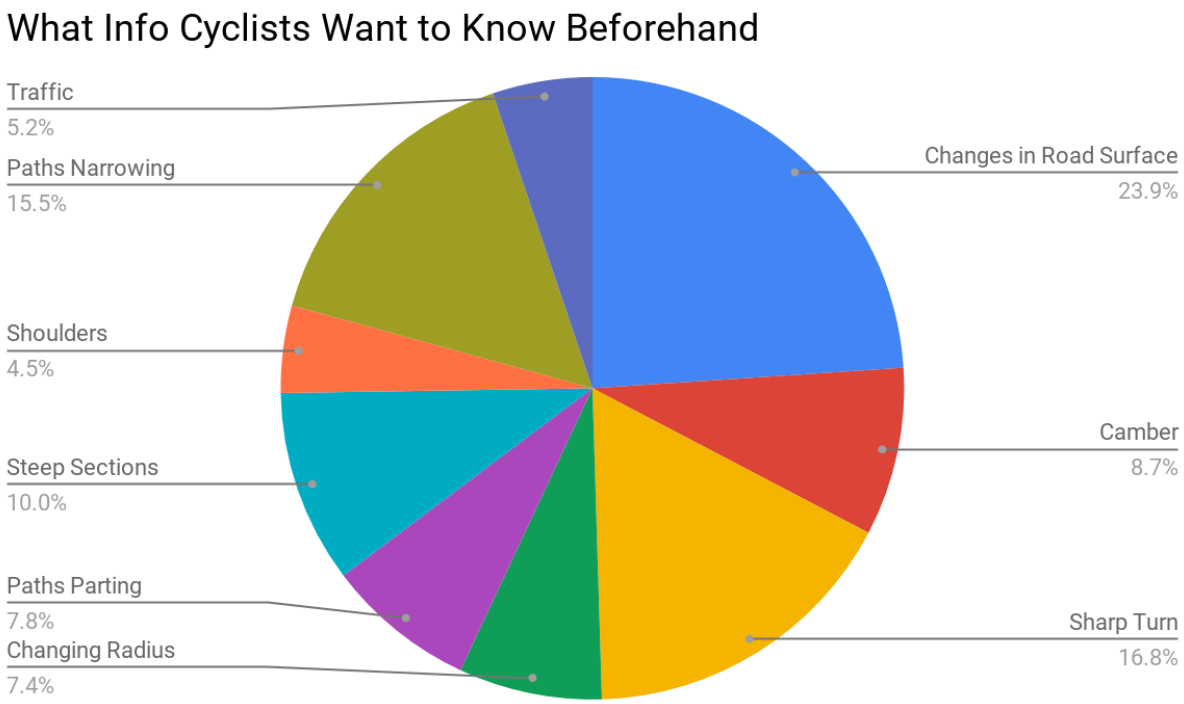

Figure 3: Information Cyclists Want Before Riding

Participants in the survey were also asked to describe in more detail what they thought some of the more specific hazards are in cycling. These results are presented in Figure 4. As expected, the largest chunk of results is due to traffic, but very close behind is the presence of unmarked obstacles and group riding. These conditions create a perfect storm for catastrophic crashes. Anecdotes given in the survey have detailed situations where such hazard pop up and take the cyclists by surprise. This is especially an issue when cyclists are riding in large groups and may not be able to perfectly see what is coming up ahead of them. Again, this provided the Team with good feedback on what sorts of hazard the algorithm should track and warn cyclists about.

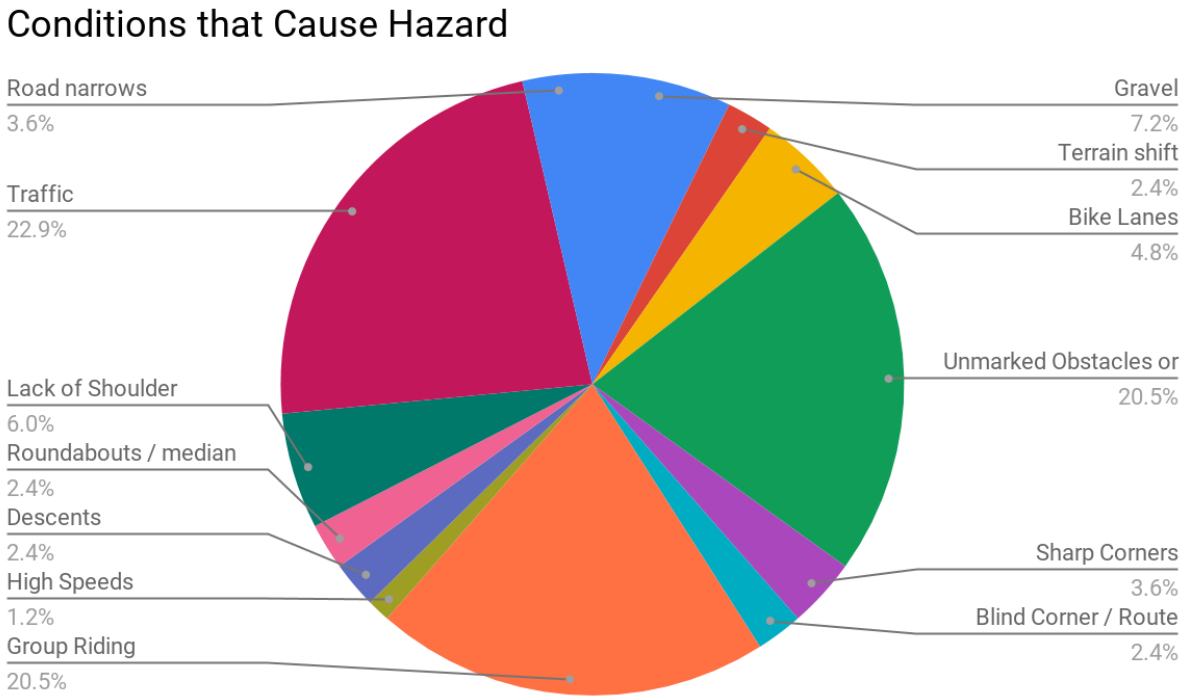

Figure 4: Road Conditions That Are Hazardous to Cyclists

Figure 5 shows various suggestions that the cyclists proposed that would help improve cycling safety. Improvements relating to vehicle traffic once again reveals itself to the number one concern of cyclists. There are several conditions more specific to the actual route and race that cyclists see necessary to address. Three main areas of focus have emerged from the data. First, cyclists indicated that better signage and more robust lane lines could help a lot. Following that is the presence of more officials and getting them better training. Finally, approximately 9% of responses called for a mandatory “briefing” and the availability of a route map.

Figure 6 gives the responses to a similar question, but this time focuses on improvements that could be made to increase recreational cycling safety. As with many of the previous survey responses, improvements involving traffic was once again the most popular response. Aside from traffic, many of the responses called for better shoulder and bike lane situations. Often, the cyclists cite routes that do not have bike lanes or shoulders, or bike lanes that suddenly stop leading to dangerous route conditions.

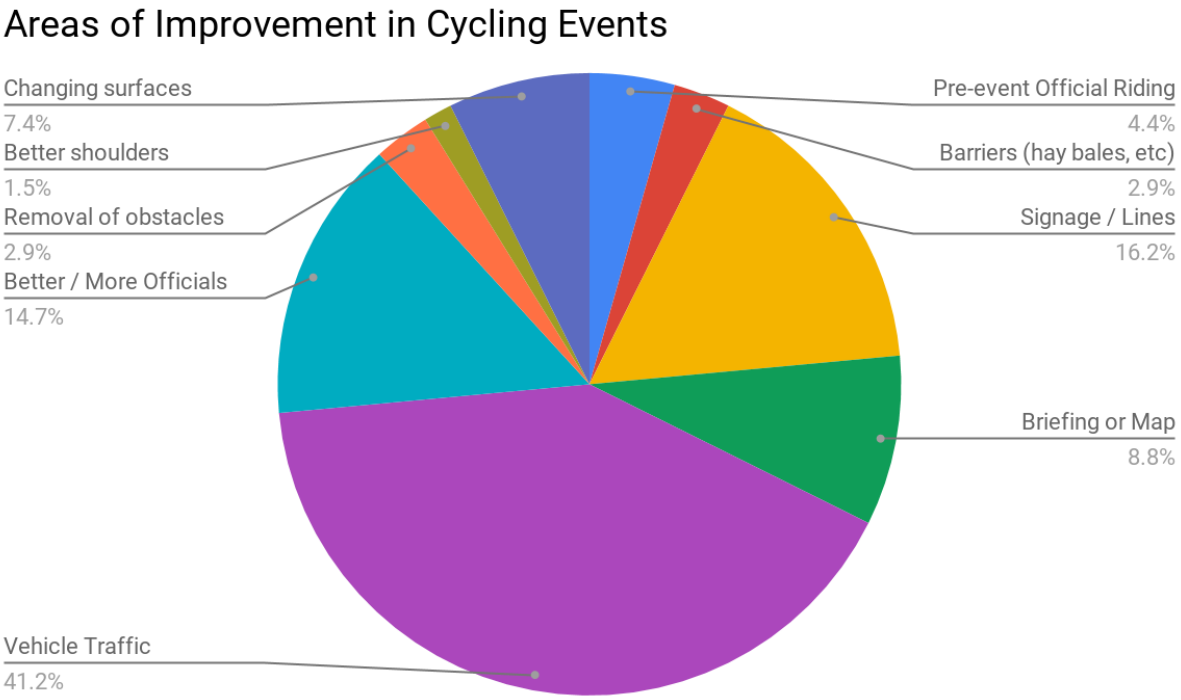

Figure 5: Methods to Improve Cycling Event Safety

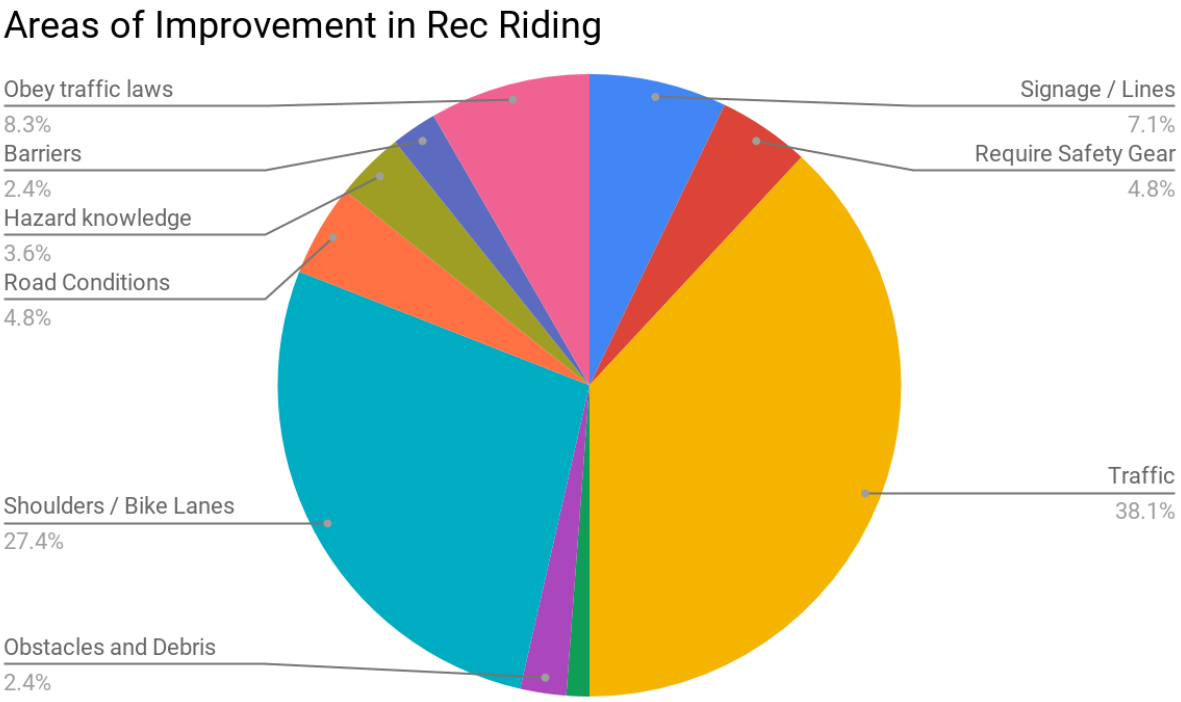

Figure 6: Methods to Improve Recreational Cycling Safety

Figure 7 indicates what cyclists typically do in terms of research before they go for a ride. About half of the cyclists who responded consult a map service such as Strava or Google Maps. They look mostly for vehicle traffic density, road surface conditions, elevation and steepness profile, and the existence of a shoulder or bike lane. However, most of the responses indicated that many cyclists do not do in depth research before they ride. This was important feedback for the Team because it shows the importance of keeping the route packet produced for this project simple and concise. If the route packet is too long or wordy, many cyclists will not take the time to read through all of it before leaving to ride.

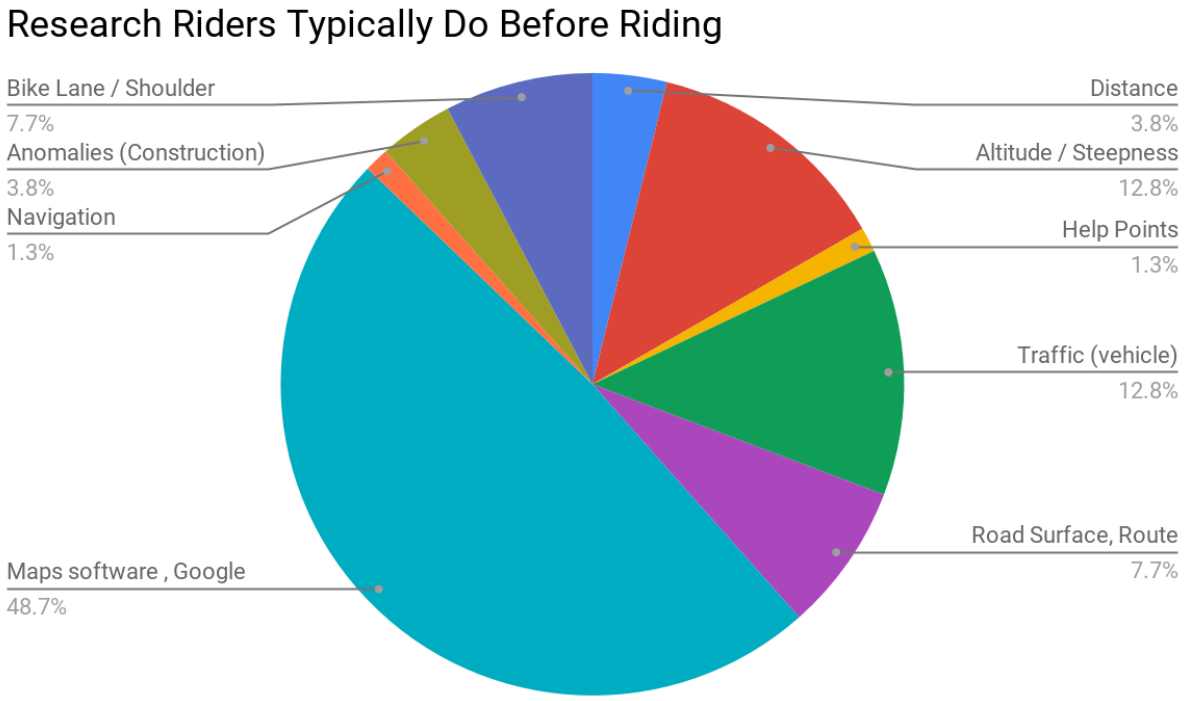

Figure 7: Research That Cyclists Do Before Riding

Overall, the survey responses gave the Team a great insight into the mind of a cyclist when it comes to safety. As expected, motor vehicles are the biggest hazard to cyclists. Many of the comments cite the aggression that vehicles and cyclists often have for each other.

Putting the vehicle info aside, this data also points out that road and surface conditions are a major concern for riders. Many of them specifically cited gravel as being a culprit. Also, shoulders and bike lanes are things that riders have on their minds often. Cyclists want to know where bike lanes are and if they end abruptly. It's also clear that riding in groups amplifies other conditions such as areas where routes narrow, curves are sharp, and obstructions exist.

The results of the Survey lead us to settle on a two prong approach for collecting data. The algorithm is split into two sections, and automatic portion which pulls data from Strava for a route, and a manual portion in which an individual enters types of hazards identified, their severity, and the gps coordinates of each one. The reason behind this is because the Team learned from the survey, that only using data from Strava would not adequately predict many of the hazards cyclists are concerned about. As such, the manual data collection was necessary to cover the hazards which the Strava data would otherwise miss.

3 Product Operation

3.1 Inputs

For this project, the primary method of data collection is gathering public data through Strava. This is done using what is called a GET request. A GET request is a method of gathering the data that is used to make the graphs that Strava displays on a rider's page when they post their data from a route. In essence, this is an automated method for gathering each data point that is used to make those graphs. The first step in this process is to find the desired route. Once the route is found, the next step is to select the rider that the data will be gathered from. For this project, the fastest rider for that route has always been selected. Our reasoning behind this is that the fastest rider is pushing the course to its limits, which means their data will most accurately reveal where there are potential hazards in the route. For example, where the fastest cyclist decides it is safe to go fast versus where they decide to break reveals where the dangerous sections of a route are. While the experience of cyclists of different skill levels riding the route will vary, using data from the fastest cyclist doesn't just show where hazards are for a skilled cyclist, it shows where hazards are in general based on the characteristics of the route. If a corner is dangerous for a professional cyclist, the corner is likely still going to be dangerous for a recreational cyclist, even if they are going slower. Therefore, the Team believes that the best and most accurate results are achieved by collecting data from the fastest cyclist for any given route.

Once the route and cyclist are known, the rider data can be collected using the GET request. The first step in this process is to find the numeric ID that is used to identify the specific ride for the cyclist and route in question. This numeric ID is found in the URL when on the webpage of the specific route that the data is being collected for. For example, a URL is shown below where the numeric ID is located at the end after the final slash:

https://www.Strava.com/activities/<NUMERICID>

Once the numeric ID is found it simply needs to be inserted into the following URL in the place of NUMERICID. The necessary URL for the GET request is shown below:

https://www.Strava.com/stream/<NUMERICID>?streams%5B%5D=latlng&streams%5B%5D=distance&streams%5B%5D=altitude&streams%5B%5D=time

This URL can be entered into any internet browser search bar with the desired numeric ID, and it will return a tab with the raw rider data from Strava. When using the GET request, the four different types of data are given in giant vectors. The entire set of data cannot be shown here as most routes have thousands of data points spanning several pages. However, figure 8 shows an example of the first few numbers for each set of data that is gathered. In this example, the data is from the Lookout Mountain Hill Climb. While the data for a given route may vary depending on

the rider, it appears that for most routes the data sets have a sampling rate of 1 data point per second. While it would always be nice to have data that is of higher resolution, one data point every second does provide enough resolution to make any required calculations. The only thing that must be done to the GET request data before it can be used within the algorithm, is that the commas in between each data point must be removed. While this can be somewhat tedious, it can easily be accomplished using find and replace tools inside of most coding programs. Once the data sets have been cleaned up by removing all unnecessary elements, the data can be used within the algorithm. Figure 8 also shows the data after it has been cleaned up and made ready for use within the algorithm. The following sections will explain how this data is used to accomplish the Team's objective of predicting hazardous cycling conditions.

```
LatLong = [39.741978 -105.22821 39.741985 -105.228228 39.74199 -10

time = [0 1 2 3 4 5 6 7 8 9 31 32 33 34 35 36 37 38 39 40 41 42 43

distance = [0.0 1.7 3.4 5.1 6.5 6.5 6.5 6.5 6.5 6.5 6.5 10.1 12.6

altitude = [1838.0 1838.0 1838.0 1838.0 1838.0 1838.0 1838.0 1838.
```

Figure 8: Sample of GET Request Data From Strava

While Strava is by far the primary method of data collection for this product, the Team is also maintaining the capability for manually collected hazard data to contribute to the final hazard profile presented in the route packet. This is because the data gathered with Strava will not be able to predict all the hazards that a cyclist will face on their route. In order to facilitate this data collection process, the algorithm will has a number of different types of hazard that can be input manually. Once these types of hazards are known, anyone surveying a cyclist route can make note of these hazards and either the distance or the latitude & longitude point the hazard is located at. These hazards can then be used in the algorithm. Initially, this will likely only be done with race organizers, as they are the only ones who are likely to survey an entire route before riding it. However, in the future it should be possible to create a phone application that allows any cyclist to report certain hazards along a route. The phone can then record the hazard type input by the cyclist and the GPS coordinates of the hazard. This will then update a database for future use when analyzing routes. Together, these two methods of collecting data allow the algorithm to properly and accurately calculate and predict many risks that a rider will face along their route.

3.2 Automatic Algorithm Execution

As previously mentioned, the decision was made to break up the algorithm into two different parts. Both parts come from the same algorithm, but have different inputs and purposes. These two parts are the automatic and manual portions of the algorithm. The automatic algorithm refers to the portion of the algorithm that will calculate hazards based on the data gathered from Strava. That is, it will calculate all applicable hazards that can be determined given a rider's time, distance, elevation, and latitude & longitude. The manual portion of the algorithm will use data that is collected manually by race organizers or other riders on a case by case basis. Manual hazard data includes things such as road conditions, road material, bottlenecks, roundabouts, usual traffic, and whatever else is deemed applicable for the particular route in question. The purpose of this portion of the algorithm is to capture any hazards that are impossible to predict with the data gathered from Strava.

Gathering data with Strava is a huge advantage for this project because the Team can quickly gather data for routes of varying skill levels from all over the world. It was chosen because it has the unparalleled opportunity to support a product that aims to be used by all cyclists globally. This is why, even though it can not be used to predict every hazard a rider could face along their route, it has been chosen as the data source for the primary algorithm.

The automatic algorithm is named as such because given the same types of data from Strava, it will always be able to calculate the base risk level of a route purely in terms of its physical properties. The automatic algorithm will do this by calculating risks based on the turn radius, rider velocity, rider acceleration, and percent grade of a route. These hazard types were chosen because data from the cyclist survey indicated those hazards to be important. Additionally, they could all be calculated from the given Strava data. As such, these hazards are based on the physical layout of the route, and form the base risk level for every route. All other applicable hazards required some sort of input data that cannot, at this time, be gathered from Strava. The automatic algorithm assigns each of these four hazards a hazard multiplier which are then multiplied together to receive the overall risk level for every point along a route. The overall hazard equation used to calculate the risk at every point along the route can be seen in its generic form in Equation 1.

$$H = h_1 \cdot h_2 \cdot \ldots \cdot h_{n-1} \cdot h_n \qquad \text{(Eq. 1)}$$

Where: H = The total quatitized hazard at a given point

h = An individual multiplier for a specific hazard (acceleration, turn radius, etc.)

n = The number of hazards which are relevant at that point

For each of these four hazards, equations can be written that interpret the physical data from Strava and determines how hazardous each section is. This is why it is called the automatic algorithm, because all that is needed is the basic input data from Strava in order to calculate the base risk level at each point of the route. This algorithm will remain the same for every route analysis, which allows for widespread use of the algorithm without any additional work. The risk profile the algorithm generates can then be augmented later on with data collected manually if applicable or necessary. However, the automatic algorithm will always form the basic risk profile of the route. The remainder of section 3 will go into detail on how the risk level is calculated for each of the four hazards: turn radius, rider velocity, rider acceleration, and percent grade.

3.2.1 Turning Radius Calculations

The first hazard the algorithm calculates is the relative turn radius at every point along the route. This information is valuable as it can be used to identify where turns are occuring along the route, and whether the radius changes throughout the turn. Identification of all the turns in a given route also identifies areas of increased risk. This is because crashing becomes more likely when the bike is turning as it is harder to control. Understanding how the radius changes through a turn helps to identify which turns are more dangerous than others. Specifically, turns with a decreasing radius can be especially hazardous to riders as they are often more difficult to gauge and approach safely while riding. Two methods of calculating turn radius were pursued. The first was a geometric vector math approach, and the second used interpolation of the data points and a second order differential equation.

The simpler of the two approaches is the geometric vector math method. This analyzes three consecutive location data points to calculate the turn radius at the center point. Equation 2 is used to determine the radius of curvature through three individual points.

$$r = \frac{L_{1-2} \cdot L_{2-3} \cdot L_{1-3}}{4 * A} \qquad \text{(Eq. 2) [9]}$$

Where: r = Radius of curvature [m]

L_{1-2} = Linear distance between the first and second points [m]

L_{2-3} = Linear distance between second and third points [m]

L_{1-3} = Linear distance between first and third points [m]

A = Area of the triangle formed by all three points [m^2]

The distance variables L_{j-i} in Equation 2 can be found using the pythagorean theorem for two points in a cartesian plane, such as in Equation 3 below.

$$L_{i-j} = \sqrt{(x_j - x_i)^2 + (y - y_i)^2} \quad \text{(Eq. 3) [9]}$$

Where: L_{i-j} = Distance between points i and j [m]

x_i = The x coordinate of the first point (i) [m]

x_j = The x coordinate of the second point (j) [m]

y_i = The y coordinate of the first point (i) [m]

y_j = The y coordinate of the first point (j) [m]

When converting the position data into a cartesian plane as is required for Equation 3, x would represent the position along a straight line, such as the line formed by going east to west, and y would represent the position along a perpendicular line such as the line formed by going north to south. The area of a triangle variable, A, in Equation 2 can be calculated using vector math as seen in Equation 4 below. Note that the || and × functions are used to find the magnitude of a vector and cross product of the vectors respectively.

$$A_{i,j,k} = \frac{1}{2} \cdot \left| \vec{L_{i-j}} \times \vec{L_{k-j}} \right| \quad \text{(Eq. 4) [9]}$$

Where: $A_{i,j,k}$ = Area of the triangle formed with location data points i, j, and k [m^2]

$\vec{L_{i-j}}$ = Vector formed between points i and j [m]

$\vec{L_{k-j}}$ = Vector formed between points j and k [m]

The second method for calculating turn radius requires solving the curvature equation seen in Equation 5. This is a second order differential equation which requires the location data be expressed as an explicit function of $y = f(x)$.

$$r = \frac{\left(1+[f'(x)]^2\right)^{\frac{3}{2}}}{|f''(x)|} \quad \text{(Eq. 5) [10]}$$

Where: r = Radius of curvature at point x [m]

f'(x) = First derivative of the position function f(x) [-]

f''(x) = Second derivative of the position function f(x) [m^{-1}]

In order to use Equation 5, the discrete tabulated position data must be interpolated to yield an explicit function. There a several methods of interpolation used in scientific computing today. For the purpose of this project, interpolation using Lagrange Polynomials given in Equation 6 would be sufficient.

$$f(x) = \sum_{k=M}^{N} f(x_k) \prod_{i=M\ (i\neq k)}^{N} \frac{x-x_i}{x_k-x_i} \qquad \text{(Eq. 6) [11]}$$

Where: f(x) = Explicit position function for points data points M through N [m]

M = First point being interpolated [-]

N = Last point being interpolated [-]

k = index value between M and N for summation series [-]

$f(x_k)$ = y coordinate of point k = (y_k) [m]

i = index value between M and N for product series [-]

x = x coordinate variable (used to find f(x)) [m]

x_i = x coordinate at point i [m]

x_k = x coordinate at point k [m]

This interpolation allows Equation 5 to be solved at any position along the curve, not just the positions for which there is a discrete data point. However, the explicit position function is not an input for calculating any of the hazards. Rather, in Equation 5, only the first and second derivatives of the explicit function are required. The method used for determining the derivatives in Equation 6 is the finite difference method.

Both the geometric vector and the interpolation approaches each have their own pros and cons. The geometric vector approach is a fast and easy calculation, however it can only be used to calculate the radius at the discrete data points taken from Strava. The interpolation method is a longer and more computationally intensive process, however it has the potential to produce more accurate results and can be used to calculate the radius of a turn in between the discrete position data points. The final algorithm presented to the Foundation uses the geometrical vector math approach because it was found to provide a high enough resolution of data while being computationally inexpensive.

Once the turn radii along a route have been calculated, they can be used to assign a risk or hazard multiplier for every discrete point in the overall hazard equation (equation 1). Turns with tighter or smaller radii are more hazardous than turns with larger radii, thus the assigned multiplier will be higher. The same is true for turns which are found to have a decreasing radius than for turns with an increasing radius. Figure 9 is a graphical representation of the Latitude and Longitude data for a rider on Lookout Mountain. This segment is being used for preliminary testing of the algorithm due to high amount of tight turns throughout the route.

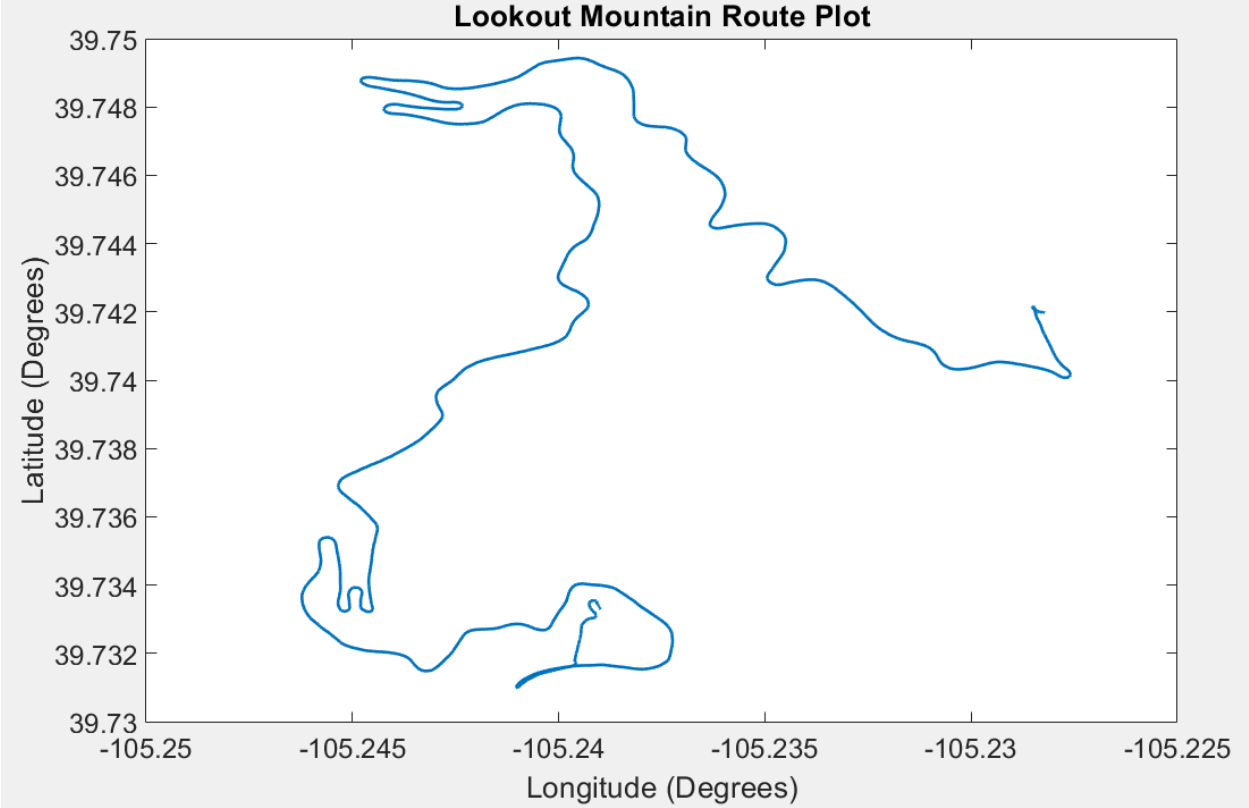

Figure 9: Latitude and Longitude plot for Lookout Mountain

3.2.2 Velocity Calculations

The next hazard the algorithm calculates is the velocity of the rider(s). It is important to note that the rider(s) in this instance are those who have recorded their data in Strava, not those who will potentially be using the algorithm. Cycling at higher velocities presents an inherent danger to any rider as it decreases their reaction time, increases the difficulty in controlling the bike, and amplifies the consequences to the rider if a failure occurs. For example, if a tire becomes flat while traveling five miles per hour, it is easier for the rider to maintain control of the bike and stop safely than it is if they are traveling at forty miles per hour. By examining how fast previous riders have gone on a given route, the fastest parts of a route can be identified, which proves useful in determining the risk level at those points.

The actual calculation of the velocity is fairly straightforward. The distance traveled between each of the Strava data points is divided by the time interval between those data points. This calculation is shown in Equation 7.

$$V_i = \frac{dist_i - dist_{i-1}}{time_i - time_{i-1}} \quad \text{(Eq. 7)}$$

Where: V_i = Velocity at time index i [m/s]

$dist_i$ = distance value at time index i [m]

$dist_{i-1}$ = distance value immediately prior to time index i [m]

$time_i$ = time value at time index i [s]

$time_{i-1}$ = time value immediately prior to time index i [s]

The velocity values calculated here are then used to calculate a three point running average. This velocity data can pose issues when attempting to apply it to future riders. For instance, if the rider stopped along the route to take pictures or drink water, then that is not a good representation of how a different rider may approach the route. However, by comparing how multiple riders have completed the route, it is possible to overlay and average all of their data sets together. This will yield a good representation of how the average rider might approach the route as it will show places where most riders are traveling quickly or traveling slowly.

Once the velocities from the rider are calculated, every point along the route can be assigned a hazard multiplier for the velocity. Areas where the velocities are higher will receive a higher hazard multiplier for the reasons discussed previously. It will also allow for identification of likely places for riders to stop. This could indicate a stop light, rest area, or even a common flat tire spot. While it is difficult to decipher which of these it could be, it is still useful information to riders if they understand beforehand where other riders are likely to be stopped. Figure 10 shows a graphical representation of the three point running average for velocity of a rider on the lookout mountain hill climb.

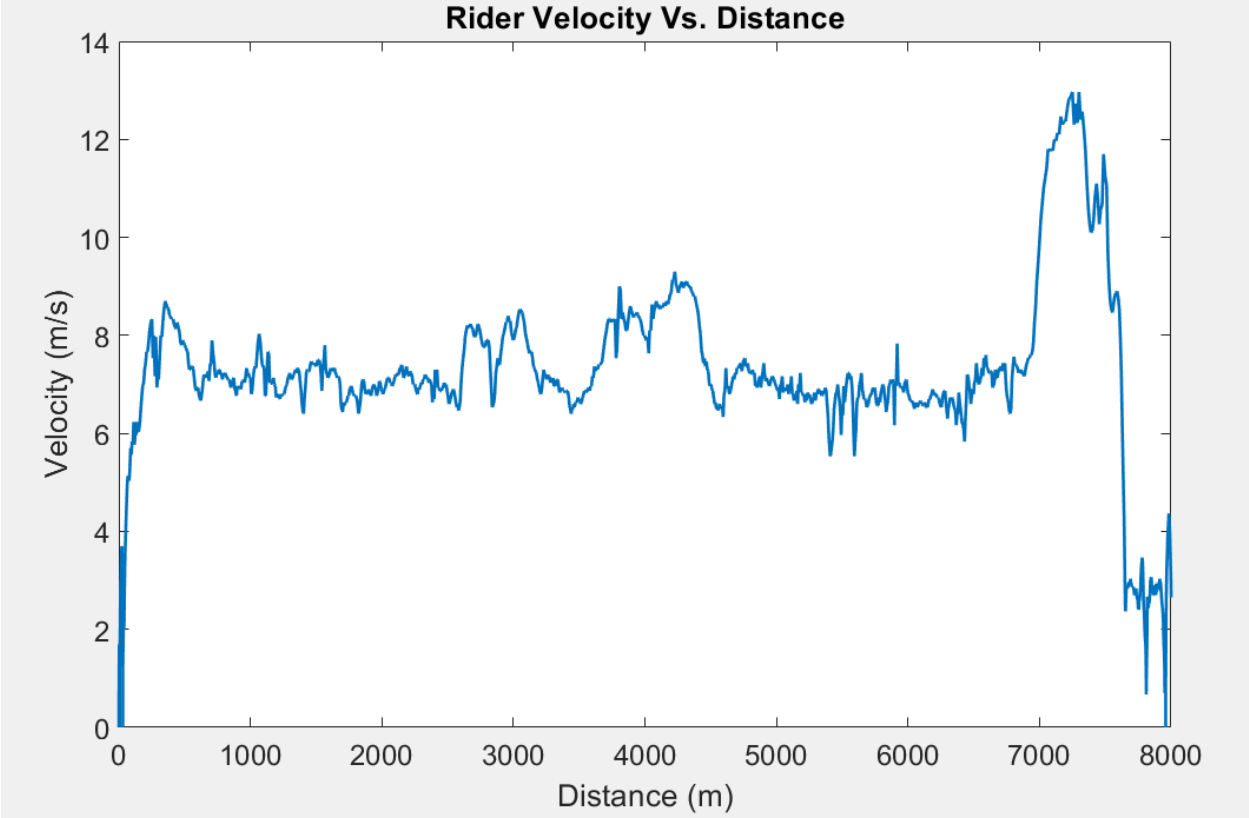

Figure 10: 3-Point Average Velocity Data For Lookout Mountain

3.2.3 Acceleration Calculations

The acceleration hazard is very similar in calculation and implementation to the velocity hazard. As with the velocity hazard, it is important to note that the acceleration calculated in the algorithm is of the rider(s) who have recorded their data in Strava, not those who will potentially be using the algorithm. Areas where cyclists rapidly accelerate or decelerate can be identified as hazardous. This is because the bike becomes harder to control, and for decelerations in particular, the chances of tires losing traction and sliding are increased. Additionally, if the rider is decelerating and an additional unforeseen hazard appears, the ability for the rider to decelerate even more is decreased.

The calculation of the acceleration is again reminiscent to the velocity data calculation. The change in velocity between each of the data points is divided by the time interval between those data points. However, the velocity data set used in this calculation is the three point running average. The reason for this is to prevent the requirement of an additional running average calculation in the acceleration. The acceleration calculation is shown in equation 8.

$$Acc_i = \frac{V_{avg,\,i} - V_{avg,\,i-1}}{time_i - time_{i-1}} \qquad \text{(Eq. 8)}$$

Where: Acc_i = Acceleration at time index i [m/s^2]

$V_{avg,\,i}$ = Average velocity value at time index i [m/s]

$V_{avg,\,i-1}$ = Average velocity value immediately prior to time index i [m/s]

$time_i$ = time value at time index i [s]

$time_{i-1}$ = time value immediately prior to time index [s]

The information gathered from this calculation is also not necessarily representative of how a rider using the algorithm may approach the route. However if information from multiple riders is compiled, it can offer a better representation of how the average rider approaches the route. Areas with larger decelerations will have the largest multipliers as the risk is greatly increased. Areas of high acceleration will not be treated as dangerously as with large decelerations, but it will most likely warrant a slight increase in the risk multiplier. Figure 11 below shows an example of the acceleration data calculated for a rider on Lookout Mountain.

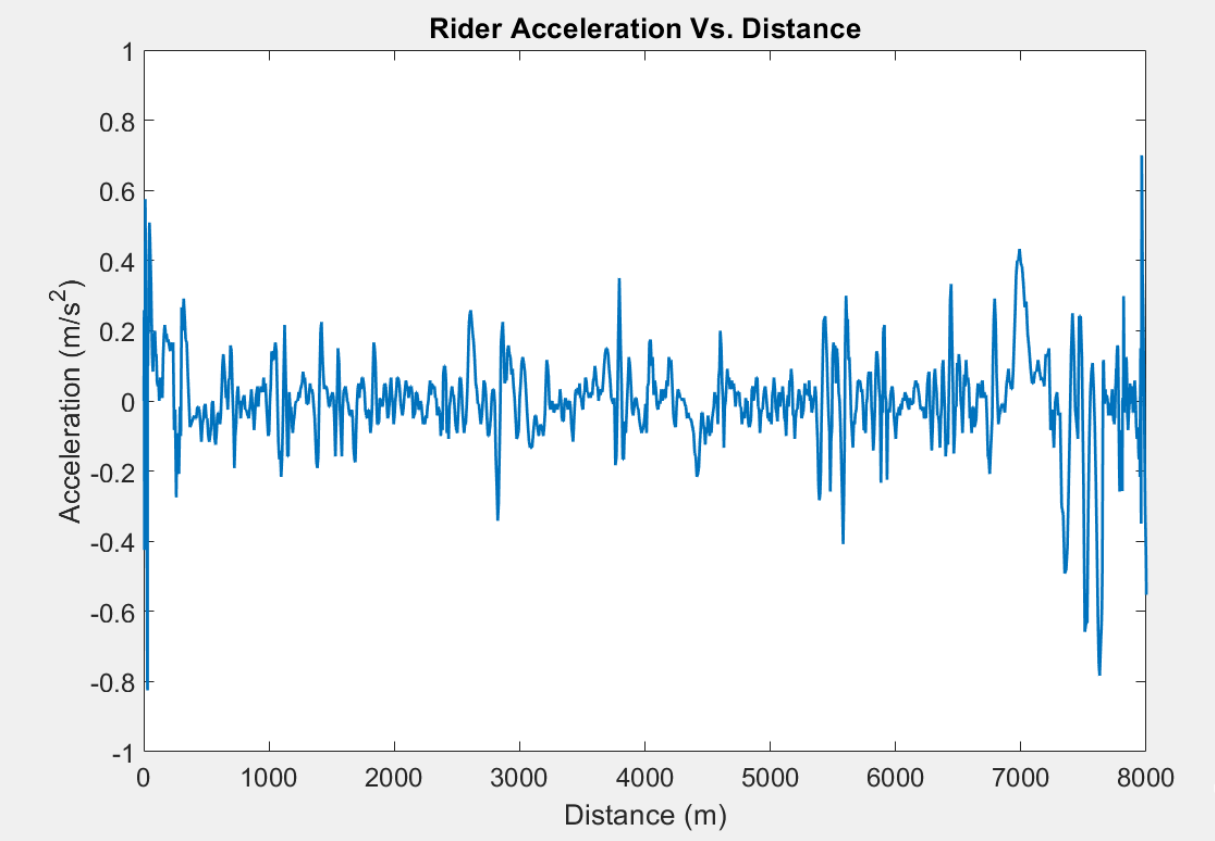

Figure 11: Acceleration Vs. Distance Plot for Lookout Mountain

3.2.4 Grade Calculations

The last hazard being calculated in the automatic portion of the algorithm is the percent grade. This information is important because it is one of the most important factors in determining how a rider plans their execution of a route. Long uphills have drastic effects on rider stamina, while prolonged downhills generate increased momentum and wear on equipment such as overheated brakes. Either way, large positive or negative grades represent areas where the risk to the rider is greater than if the grade is zero.

Calculating the percent grade proved to be more challenging than anticipated. The reason for this is the low resolution of the altitude data yielded from Strava. The altitude data is rounded to the nearest meter. Therefore, the change in altitude, even for steep inclines, is not always great enough to be seen from one data point to the next. In other words, if the change in altitude from one second to another is one foot, the data from Strava will show no change in altitude over that time period. As such, the calculation for percent grade must incorporate the change in elevation over several data points in order for the change in altitude to be large enough to overcome the rounding process employed in the data received from Strava. Equation 9 below uses five data points to calculate the percent grade.

$$G_i = \frac{alt_{i+2} - alt_{i-2}}{dist_{i+2} - dist_{i-2}} \cdot 100\% \qquad \text{(Eq. 9)}$$

Where: G_i = Percent grade at time index i

alt_{i+2} = Altitude value at two time periods after time index i

alt_{i-2} = Altitude value at two time periods before time index i

$dist_{i+2}$ = Distance value at two time periods after time index i

$dist_{i-2}$ = Distance value at two time periods before time index i

This method of calculating grade is essentially the same as the running average performed on the velocity data, except with a little wider range of the data points included. However, this time the purpose is not to reduce noise in the data, but to produce grade data that accurately matches the route. For the reasons described above, portions of the route with a steep downslope grade will receive higher values for the risk multiplier, as opposed to sections with zero percent grade which will get a multiplier of 1. In addition, portions of the route that have a steep uphill grade do not get an increase in the risk multiplier. This is because going uphill does not always add risk for a rider. It is important to note that while the algorithm will not track portions of sustained downhill grade, it will independently track grade and velocity. For sustained downhill segments, even if the grade stays the same, the rider's velocity will be increasing, which will cause the

overall risk factor to increase. Figure 12 shows the percent grade vs distance plot of Lookout Mountain, while Figure 13 shows the elevation profile when traveling uphill.

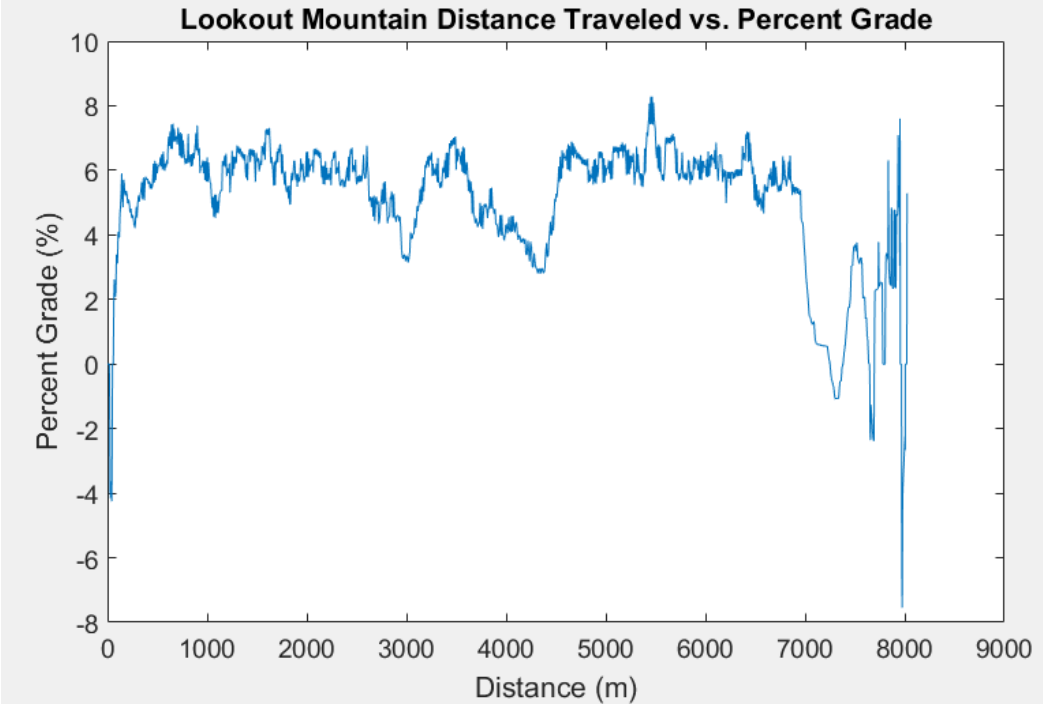

Figure 12: Percent Grade of Lookout Mountain Ride

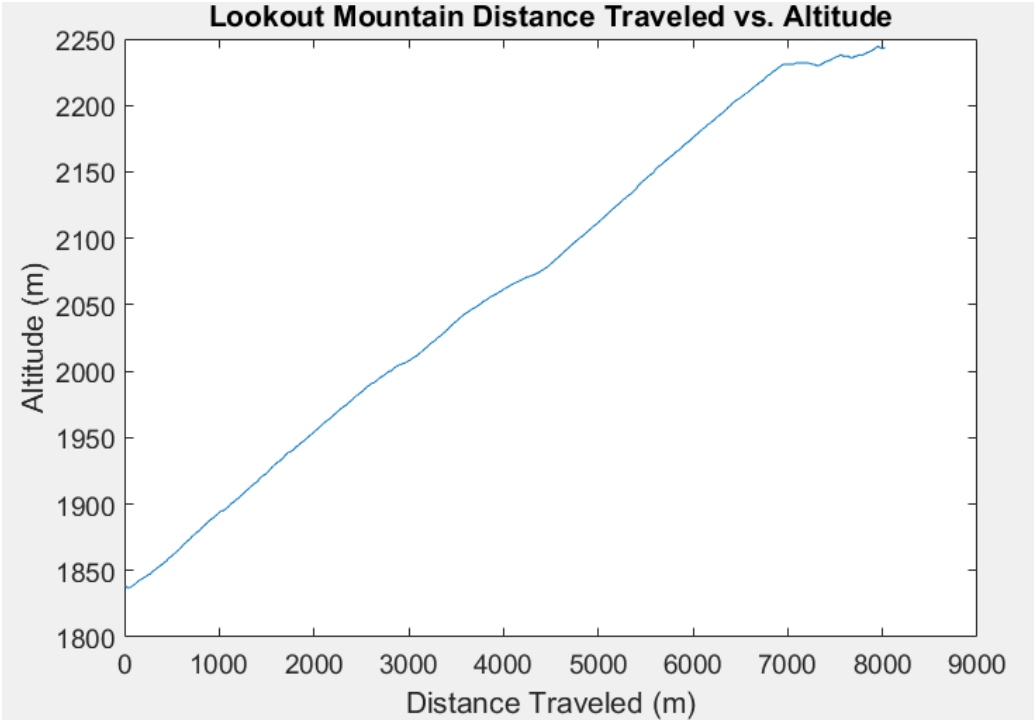

Figure 13: Elevation Profile Example of Lookout Mountain Ride

3.3 Manual Algorithm Execution

Based on outreach data collected from the Mines Hill Climb event, cyclists are often concerned with various hazards that cannot be identified via Strava data. The largest concerns of safety the riders shared included; potholes, poor road conditions, tight bottlenecks, high traffic zones, intersections, and blind corners or turns. Due to the nature of several of these hazards, location data is not readily available for them. Thus, in order to account for them and incorporate them into the algorithm's analysis, additional data on a route must be collected manually. However, aside from this, the manual algorithm is very similar to the automatic algorithm. Each manually identified hazard has a risk multiplier attached to it which multiplies by the base hazard modifier.

Initially, the manual data is intended to be collected by race coordinators or officials by physically traveling the desired course prior to a race and tagging the GPS coordinates of a hazard. The type of hazard and its GPS coordinates are then incorporated into the algorithm to further increase the quality of the analysis and create a safer race environment for the participants. In order for this to work, the algorithm has set types of manual hazards for which risk multipliers have already been determined. Currently this list is small, but over time the list of manual hazards will grow into a database that covers many different possible hazards. By designing the algorithm to accept manually inputted data, it also allows for consideration of temporary or seasonal hazards such as construction or ice along the route. Furthermore, if future data becomes available on other hazards, only the GPS data will be needed to incorporate it into the algorithm. Some adjustments to the balance of the overall hazard equation may be needed if new data becomes available, however the principles used within the algorithm are adaptable enough to allow for future development.

When common and specific hazards such as potholes, large road cambers, and blind corners are entered into the algorithm, the overall hazard calculation, Equation 1, is modified to include an additional hazard multiplier. Depending on the type of hazard, the multiplier can be implemented at one or multiple points around the GPS coordinates. However, because the algorithm cannot account for every type of hazard which may be in existence along a route, there will be some hazards which are identified which will not have a predetermined hazard multiplier. In the event that a hazard is identified which the algorithm was not built to consider, the individual inputting the data will be asked to enter a brief phrase which describes the hazard. This phrase and the location of the hazard along the route will be identified graphically and be available to the riders to see along with the other algorithm outputs. Ultimately, the manual algorithm is just a way to provide for a more robust hazard analysis.

3.4 Determination of Hazard Multipliers

Hazards on a cycling route are tracked using hazard multipliers. A hazard multiplier is a number, usually between 1 and 2 for every hazard that alters the baseline score of the overall risk

assessment for that route. Occasionally, in circumstances where conditions actually provide a safer ride such as safety barriers, bicycle lanes, or race officials, the multiplier may be less than 1. The baseline score of a route will be the number 1, which indicates a route that is smooth, wide, flat, has no traffic, and no obstacles that can cause an accident. If conditions such as curves, slopes, or obstacles are added to a route, they increase the score to indicate the change in safety. This is accomplished through Equation 1. Any hazard multiplier above 1 that multiplies with the base value of 1 must raise the overall risk value for the route. Table 2 below shows the 1 through 5 hazard ranking system.

Table 2: Description of Numerical Weight Scale

Hazard Score	Description
1	Little to no route conditions that would contribute to an accident. Flat grade, no curves, no obstacles, light to no traffic.
2	Conditions require some attention, but generally not hazardous. Slight grades, slight curves, or some obstacles on an otherwise hazard-free route.
3	Speed reduction and attention will be needed to navigate the route conditions. Multiple hazard factors are likely present.
4	Drastic speed reduction and careful attention to the route will be necessary.
5	Existing conditions create a “perfect storm” of hazards that can easily catch any cyclist off-guard.

3.4.1 Automatic Hazard Multipliers

The multipliers that define the impact of each automatic hazard on the overall risk assessment were created using several methods. First, numbers were assigned as a result of qualitative understanding of physics, consulting of roughly one hundred survey results from the cyclist survey the Team performed, and research of dangerous cycling conditions. The survey results were particularly useful in these weights, as it was not immediately obvious exactly how to weigh certain obstacles. This is because even though the Team may think some hazards are worse than others, cyclists might not believe the same. Survey results informed the Team of

some conditions that are drastic enough to warrant a high hazard multiplier, such as a bottleneck in the road during a high-traffic race.

Next, quantitative physics analysis was used to refine these modifiers. The Team began by assigning qualitative weights to conditions such as slope and curvature using a linear categorization system. Basically, using physics and dynamics allowed the Team to have a better understanding of how hazardous some obstacles are, and better assign the modifier values to match this. Where there are several examples of how this can be applied, a particularly relevant example will be shown here. Physics indicates that risk levels do not increase linearly with percent grade or curvature radius. For example, when analyzing the slope factor, moving from a 0% to a 5% grade does not add as much to the hazardousness of a route as moving from 5% to 10% grade. Momentum is gained at a faster rate as the percent grade increases, indicating that the numerical weights should be applied in more of an exponential fashion. Analysis like this and further, more in depth dynamics analysis allowed the Team to get a good idea of what the multipliers should be for each hazard.

The third and most important part of the process is that the numerical weights were rigorously tested by riders who provided helpful feedback. Cyclists were given the route packet produced by the Team's algorithm and asked to ride the route. This allowed the Team to compare the computed score with the riders' experiences. If the rider believed that certain sections of the route should be rated as more or less dangerous by the algorithm, the Team looked at the code, determined what was causing the incorrect risk multiplier to be achieved, and adjusted the balancing of the multipliers. Overall, this is how the accuracy of the algorithm was perfected. The previous two methods were essential to setting up the algorithm initially, but rider feedback trumped any analysis that the Team did mathematically. With future testing, if enough riders are consulted until the algorithm is consistently matching their opinions without any adjustments, than the accuracy of the algorithm will be verified. The testing of the algorithm will be expanded upon in section 5.

3.4.2 Manual Hazard Multipliers

Unlike the automatic hazard multipliers, the manual hazard multipliers were harder to determine. This is because for many of the manual hazards, such as potholes, roundabouts, or bottlenecks, the degree of danger is very dependent on the conditions in which the hazard exist. For example, a small pothole at the center of the road will likely have very little to no effect on a cyclist. However, a large pothole on the side of the road, where most cyclists tend to ride, could have a rather large effect on the rider. In addition, there really isn't any physics or dynamics analysis which can inform the hazards that a pothole imposes. Therefore, the manual multipliers were based on a qualitative analysis that aimed to determine the severity of the impact that these manual hazards have on cyclists as they are riding. For example, a pothole, while seeming

relatively benign, can have a severe impact on cyclists if hit, because it could cause a cyclist to flip over their handlebars. Therefore, this hazard received a higher multiplier than some of the other hazards that are more predictable or avoidable. Overall, the method for determining the manual hazard multipliers is less robust. Therefore, while the Team has made our best effort to make the manual hazard multipliers as accurate as possible, the accuracy of the manual multipliers will have to be verified by testing with cyclists.

3.5 Code Walkthrough

The purpose of this section is to provide a brief overview of the code and the functions that it is executing. Before beginning the overview of the code, it is important to understand it the algorithm and its operation from a high level. A flowchart of the algorithm, depicting its operation, can be seen in Figure 14.

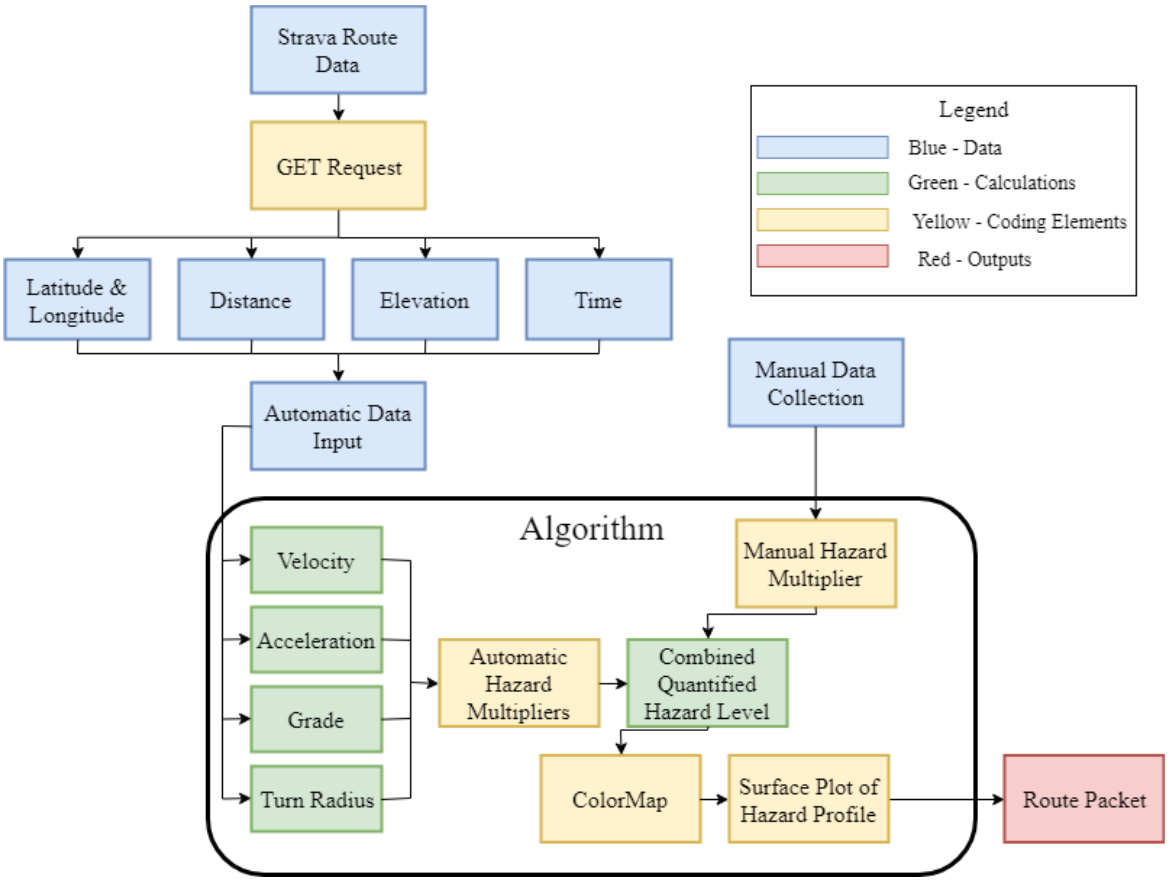

Figure 14: Product Operation Flow Chart

Much of the contents of this flow chart should already look familiar after reading the previous sections of this report. The operation starts with gathering data from Strava using the GET request. That data is then used to calculate velocity, acceleration, time, and turn radius. Those properties, along with any manual data that is collected, are then assigned a hazard multiplier

value, all of which gets multiplied together in order to get the overall hazard multiplier. A colorized map is then created, and along with the overall hazard value at each point. A surface plot is used to create the colorized hazard route profile which can then be included in the route packet.

Now that the algorithm is understood in general, a more in depth overview of the code can be presented. The code has been broken up into chunks of similar operations, and has been included in the appendix in figures A10 - A23.

Figure A10 is the very beginning of the code which begins with inputting the data that was collected from Strava. The four different data sets include: latitude and longitude, time, distance, and altitude. Each data set is a vector of over one thousand data points that are collected from Strava. The only processing that this data needs is to remove the commas that are in between each data point from the GET request. However, this can easily be done using the find and replace function. In addition Figure A10 shows the location of the manual hazards that are included in this particular data set. The next figure, which is Figure A11, shows the multiplier hazard values and initializes many of the vectors that will be used later in the code. The first portion of the figure (lines 10 - 23) show the values for both the automatic and manual hazard multipliers. There is a vector for each of the automatic hazards, as well as the many different manual hazards that the Team has implemented at this time. As seen from this figure, each hazard has a range of multipliers that can be assigned depending on the value of the property that the hazard corresponds to. These multipliers will be assigned later in the code. The second half of Figure A11 (lines 25 - 39) show several vectors that are initialized for later use in the code. Initializing a vector sets a placeholder for it in the code. It is not a required element, but it is good coding practice, and generally leads to faster code operation as well.

Figure A12 shows two different processes that are both related to latitude and longitude. The first half of the figure (lines 41 - 61) shows the creation of separate vectors for latitude and longitude. Originally, the GET request gives the latitude and longitude in a coupled format in one vector. This is difficult to work with, and impossible to use when plotting. Therefore, this code is included to break this vector into two different components. The second half of this figure (lines 62 - 81) shows the process used to determine the closest latitude and longitude point to a manual hazard. Since manual hazards are gathered separately from the GET request, the latitude and longitude points do not line up perfectly. Therefore, this section of code is used to determine the closest latitude and longitude point that the code can assign that manual hazard to so that the rest of the code can operate on a single common set of latitude and longitude.

Figures A13 - A16 show how the velocity, acceleration, grade, and turning radius are calculated by the algorithm. For more details on the calculations themselves see section 3.2 of the report.

As for these sections of the code, it is just applying the equations already discussed using a series of for loops to calculate these properties for each and every data point.

Figure A17 - A19 show the assigning of the hazard multipliers for the automatic algorithm. The first part of Figure A17 (lines 175 - 190) set up the range of velocity, acceleration, grade, and turn radius values that correspond to the automatic multipliers. The length of the vectors created in the first part of the figure, are the same length as the multiplier vectors given in Figure A11. Therefore, each value of these vectors corresponds with the multiplier in the same position of the multiplier vector. The second half of Figure A17, as well as Figures A18 & A19 then show how the code determines what multiplier to assign to each point. For each different rider and route property, the algorithm uses the vectors it created in Figure A17 to determine what multiplier it is closest to. Note that in doing so, the algorithm will always round up to provide the next highest multiplier if it is above the previous one. The second half of Figure A17 show this process for the grade, Figure A18 shows this process for the turn radius and velocity respectively, and the first half of Figure A19 shows the process for the acceleration. Finally the second half of Figure A19 (lines 268 - 283) shows the multipliers for the automatic algorithm being multiplied together and limited between the values of 1 and 5 to match up with the weight scale.

Figure A20 shows this same process but for the assigning of the manual multipliers. Overall, the process is a little more complex for the manual multipliers, but it follows the same basic philosophy: it compares the hazard against the multiplier vectors for the appropriate hazard to determine which multiplier is the closest for that hazard. Once all of the manual multipliers are assigned, the total hazard level can be found by multiplying the previous overall automatic hazard multiplier with the manual hazard multipliers. Once the total hazard value is known for each latitude and longitude point on the route, the algorithm is able to finish the process by plotting this information.

Figures A21 - A23, the final three figures, deal with plotting the information calculated by the rest of the code. Before the colorized route map can be plotted, the colormap for the plot must first be made, which shown in Figure A21. A colormap is the distribution of colors that MATLAB uses to make the colorized hazard profile. It is important when creating this colormap that all of the desired colors are included, and that they are equally spaced. It is important to equally space the colors so that the color transitions from green to yellow at a hazard value of 4, instead of a random number such as 4.12. Making the colors evenly spaced will result in even transitions in the color, which will result in a more intuitive color scale.

Figure A22 shows the actual plotting of the colorized hazard profile. The first half of the figure (lines 347 - 357) is used to initialize the plot. Because of how the colormap works, it will assign one end of the color scale to lowest total multiplier and the other end of the scale to the highest total multiplier. However this can create issues because now the same color scale might go

between 2.5 and 4.5, rather than 1 and 5. This will cause confusion when cyclists look over the route packet because a scale ranging from 2.5 to 4.5 does not make any sense. Therefore, the plot is initialized first by graphing random numbers between 1 and 5 for the entire route. This will ensure that the colormap properly goes between 1 and 5. After the plot is initialized, the data can be overwritten with the actual hazard information, which is shown in the second half of Figure A22 (lines 361 - 375). This is the same plot command, except it can be seen that the color variable, col, is now assigned to the hazard_level variable rather than a random number generator. The second surface plot will overwrite the first and produce the overall colorized hazard profile plot.

Finally, Figure A23 shows the last portion of the code which creates the option to produce different plots of the various different parameters used in calculating the hazards. This is how the plots of velocity, acceleration, grade, and latitude and longitude seen throughout this report were produced. With all of the necessary hazard information plotted, this is the end of the code. These plots are then used to create the route packet which will inform cyclists of the hazards they will face while riding the route. For more details on the colorized hazard profile produced by this code and the route packet, see section 4.

4 Product Outputs

4.1 Hazard Profile

After all the hazards and their multipliers are calculated and implemented into Equation 1, the hazard profile is constructed. This profile links every latitude and longitude point recorded along the route with a discrete quantized hazard level between 1 and 5. This array of data is best visualized through Figures 15 & 16 below. Both figures detail the Lookout Mountain downhill with colors corresponding to the hazard level at every point. The first shows the latitude and longitude coordinates of the ride, while the second demonstrates the altitude vs. the distance ridden. The color progression goes from blue, to green, to red corresponding to hazard levels of 1, 3, and 5 respectively.

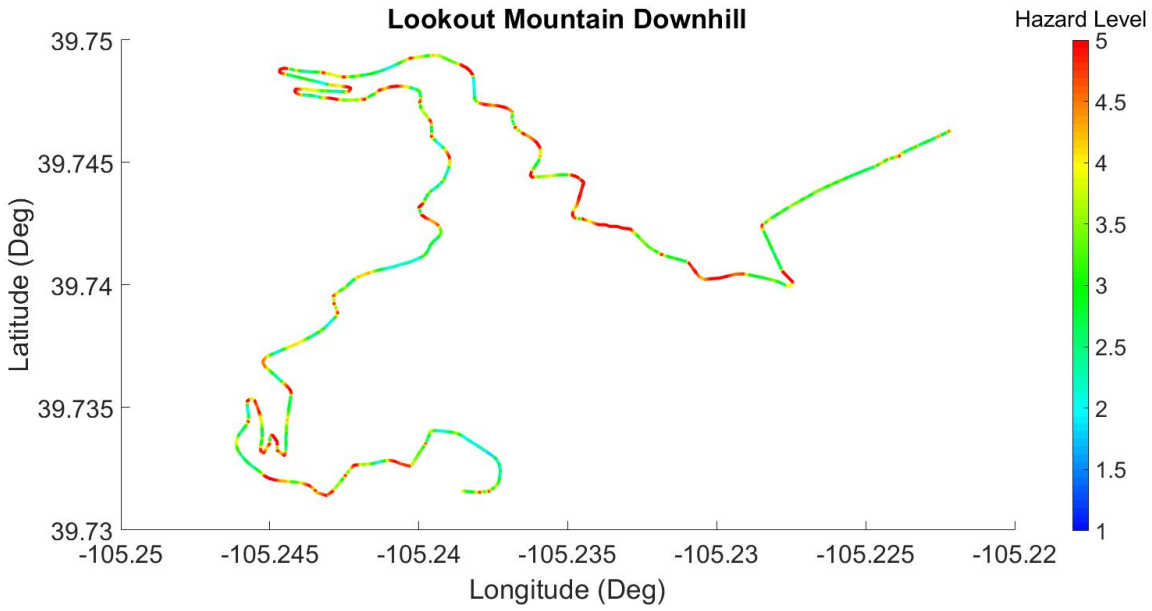

Figure 15: Latitude and Longitude Hazard Profile for Lookout Mountain

Figures 15 & 16 represent one of the most useful and important outputs of the algorithm as they provide an easy to consume medium for communicating hazardous locations. The color variation was specifically chosen so that the most hazardous sections are the easiest to identify while the low to medium hazard sections are not. Thus, cyclists need only briefly overview either figure to gain a better sense of which sections to be aware of while riding. This helps to further the Team's goal of producing a clear and concise route packet.

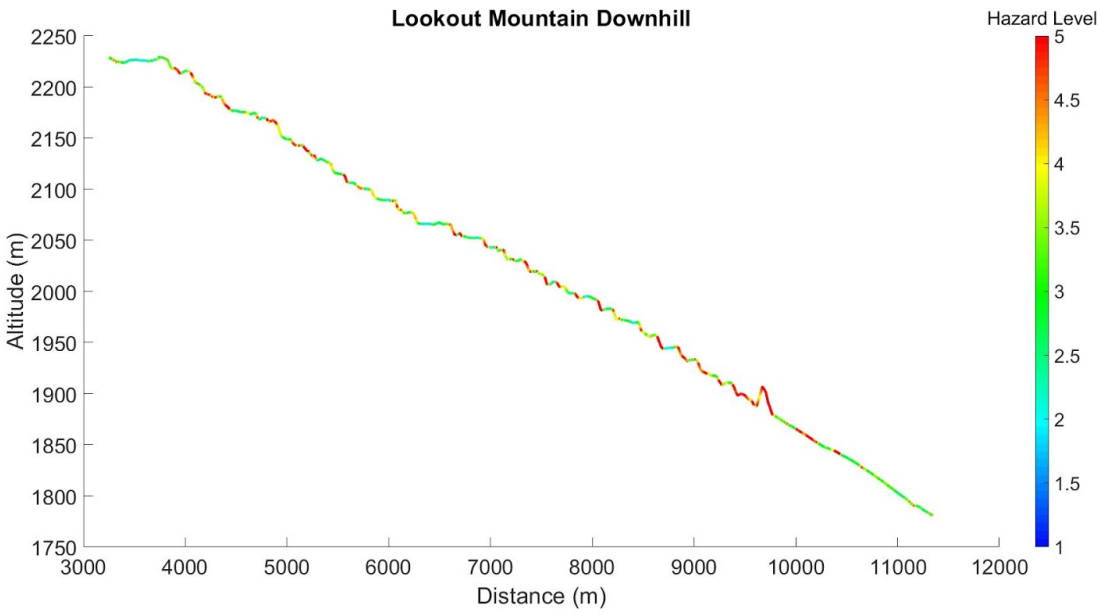

Figure 16: Altitude vs Distance Hazard Profile for Lookout Mountain

4.2 Hazard Profile Importance

The hazard profiles that the algorithm generates can be utilized in many different ways. One of the most applicable in its current form is for the generation of a route packet for official cycling events such as races and time trials. In many cases, the cyclists participating in an event have never ridden the route before competing on it. Currently, there is often very little information available to them beforehand that pertains to hazards and safety. As such, this algorithm can be extremely important in equipping cyclists with the necessary knowledge to be safe. While it is impossible to foresee and prevent every cycling accident and injury, this algorithm has the potential to reduce the risks inherently associated with the sport of cycling. Even if cyclists cannot remember where every red mark on the map occurs while they are racing, they should be able to remember a few of what the algorithm predicts the most hazardous sections will be. Even this is an improvement over current cycling races and events which present very little information to the cyclist beforehand.

However, it is important that the outputs strike the balance of containing enough information to be useful, but not so much that cyclists cannot retain the key points. The visual hazard profiles succeed in attaining this balance as the hazard information is succinct and easy to understand. Additionally, it can be deconstructed at any point to provide details to cyclists who desire more information about a given segment.

4.3 Instruction on how to use Outputs

For cyclists using the algorithm, it is recommended that they study the route until they reach a level where they are comfortable with riding it. This should start with overviewing the hazard profile figures and identifying the areas of highest hazard, particularly where the hazard level remains high for long stretches, or suddenly changes from low to high hazard. If the cyclists has a question about a specific section and why it is dangerous, then they can dig deeper into the hazard profile to find specifics on the hazard level at each point. The cyclists should then approach the areas of high hazard with caution. Ideally, the information each cyclists obtains ahead of riding the route will allow them enough warning to avoid any injury.

While cyclists make up the main body of intended users for the algorithm outputs, it can also be used by individuals who are affiliated with the sport of cycling, but may not necessarily be cyclists themselves. For example, event organizers can use the information contained in the hazard profile to better identify where safety equipment should be placed. The information can also be used to help identify stretches of road which can be focused on to improve cycling safety around the world.

5 Testing

5.1 Testing Methods

The development of the algorithm was initially based on sound engineering principles and literature. However, the most reliable way to evaluate its performance was through real-world testing. Specifically, the algorithm needed to be tested to be sure it was accurately weighing the different hazards appropriately with each other and thereby producing an authentic hazard profile. If real world testing were not performed, the resulting hazard profile could misidentify some sections as being too dangerous, and others as being not dangerous enough, thus further endangering any cyclist who uses it. As such, a two-pronged testing method was developed to maximize the useful feedback which was received.

First, the algorithm was used to analyze the stretch of Lariat Loop Rd in Jefferson County, CO from Buffalo Bill's Grave to the intersection of 6th Ave. and 19th st. This route is colloquially known as the Lookout Mountain Downhill. The first method of testing involved members of Team Cruise Control interacting directly with cyclists. Experienced cyclists would ride down the route, while the attending team member(s) followed behind them at a safe distance. The rider would then stop periodically and compare their experience riding to the hazard profile generated by the algorithm. A team member being present during this comparison period allowed them to ask in depth questions in order to ascertain the specific information on what differed between the hazard profile and the cyclists experiences. This method proved to be the most effective at providing information on which hazard multipliers needed adjustment.

The second method attempted to broaden the pool of testing candidates by asking cyclists to ride the route on their own and send in their feedback electronically. Initially, The Mines Cycling Team was contacted with a request for volunteers to ride the Lookout Mountain Downhill. Each volunteer was then sent a hazard profile with reference points labelled as seen in Figure 17, and photographs taken at each point to orient themselves. Additionally, a story was published on the cycling news outlet 303cycling.com, providing some information about the project and the necessary testing materials so that other cyclists in the local area could get involved if they so desired. Every rider was then asked to identify areas where the the hazard profile did not match with their experiences while riding and to describe those segments in relation to the reference points. Their feedback was then collected via e-mail by the Team’s communications lead.

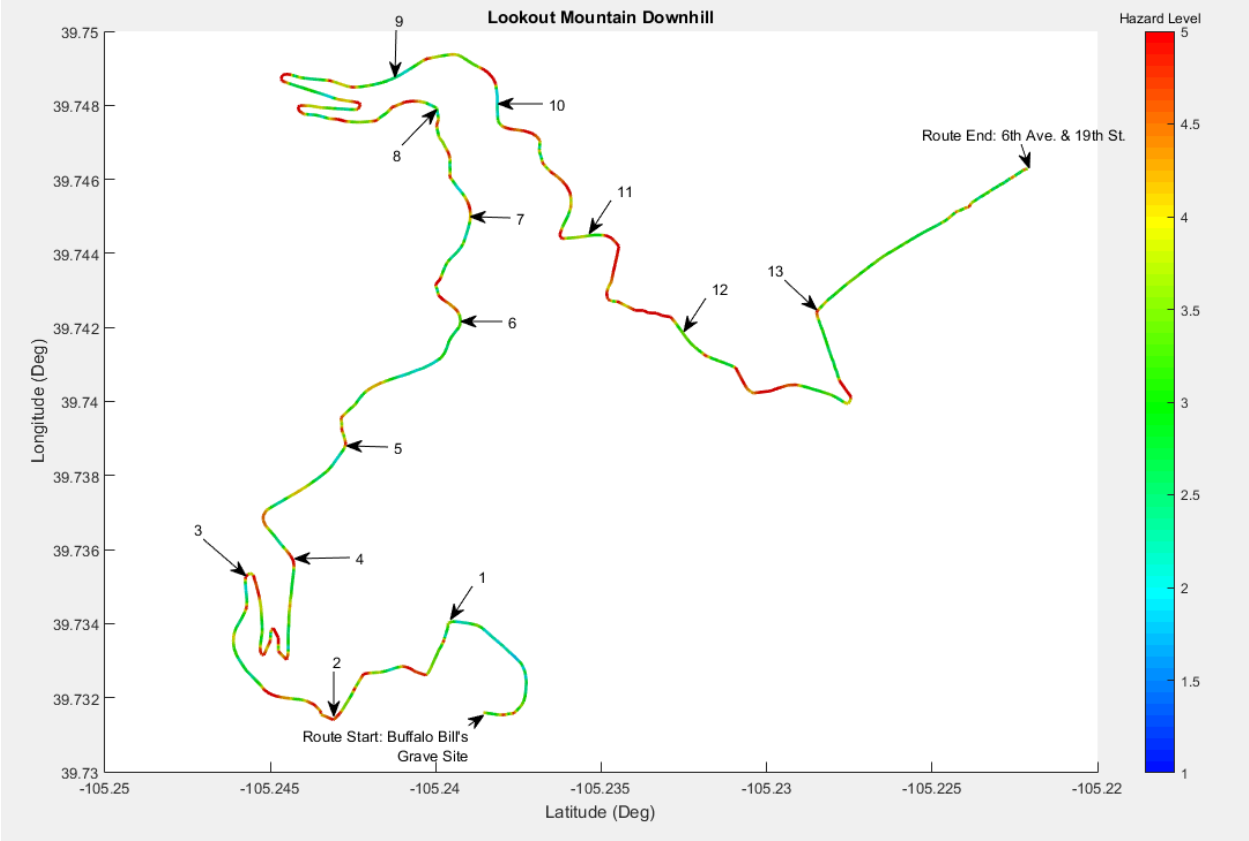

Figure 17: Lookout Mountain Test Figure

5.2 Testing Results

The results of the testing to date have been largely positive. Most cyclists have reported a strong correlation between the hazard profile and their own real-world experience. Feedback which has indicates any discrepancies in the hazard profile has largely related to two aspects. First, some of the wider and more gradual turns, such as at reference point 4 in Figure 17, have higher hazard levels than riders feel is necessary. Additionally, some straight sections where riders experience significant gains in velocity have lower hazard levels than the actual conditions depict. Specifically, the straight section at reference point 10 has been identified as one of these sections. as a result of the feedback to date, it appears the turn radius multipliers may be slightly too high and the velocity hazard multipliers too low. This feedback is extremely helpful to the Team as it allows for the algorithm to be corrected to make it as accurate, and therefore as safe as possible.

6 Project Management

Throughout the course of the project, the Team used many different project management tools to organize the tasks that needed to be completed and to keep the Team on schedule. Now that the project is complete, it is important to assess the up to date work breakdown structure, project schedule, and sprint layout structure in order to reflect accurately on the work that was accomplished and determine how well the Team used these tools to stay on track. The updated version of all three of these tools are shown in the appendix at the end of the report. Specifically, the work breakdown structure is shown in Figures A1 - A3, the project schedule is shown in Figures A4 & A5, and the sprint layout structure is shown in Figures A6 - A9.

6.1 Work Breakdown Structure

The work breakdown structure (WBS) is a project management tool that was first used at the very beginning of the project to get a sense of the entire scope of work that was needed to complete the project. The original work breakdown structure can be seen in the project charter and letter of intent. The original WBS was broken down into 6 sections: letter of intent, data collection & background research, risk analysis, implementation, possible real time feedback applications, and the project's final deliverables. Each of these sections then has subcategories and tasks that were further broken down for each subcategory. The boxes that are highlighted in a yellow-gold color denote which tasks are deliverables to the client. The WBS provided the Team's best estimate for all of the work that would need to be completed throughout the course of the project. While the initial WBS ended up being rather accurate, there are some elements that needed to be updated throughout the year.

The updated version of the work breakdown structure is shown in the appendix and is broken up into three subsequent figures to help with readability. The first two sections of the WBS are shown in Figure A1. The first section of the WBS, the letter of intent, did not need any updates. The second section, data collection and background research, required a few updates to accurately reflect the work that was done. The first update was to change gathering data from Strava with an API (application programing interface) to using the GET request. The other changes came from the other sources of data subcategory. This category did not have much to it originally because the Team was unsure if any data collection methods besides Strava would be used. This section has been updated to more accurately reflect different sources of data that the Team looked into.

The third and fourth sections of the WBS are shown in Figure A2. The third section of the WBS, risk analysis, required the most updating from the original version. However, these updates were needed to clarify, not change the work that was done. When the WBS was first made, the Team was unsure what exactly would go into the risk analysis. Therefore, what the original WBS included was quite general. It has now been updated to include a more accurate depiction of the

work that was done, including what properties were calculated to analyze risk (velocity, acceleration, grade, turning radius) and how these properties were assigned a multiplier to create the overall indexed safety rating. The fourth section of the updated WBS, cyclist outreach, is an entirely new section that was not included in the original WBS. This section replaces the section on possible real time feedback options from the original WBS as the Team was not able to get to this optional part of the project due to time constraints. The Team knew from the beginning of the project that getting opinions and feedback from cyclists would be important, but it was unknown at that point how this would be done, or to what degree it would be a part of the project. As the project went on, it became clear to the Team that cyclist outreach was an incredibly important part of the project. Therefore, in the updated WBS, a new section was created to reflect the work that was done in this area. The cyclist outreach section was broken down into the Lookout Mountain Memorial Time Trial and the algorithm testing as these are the two primary parts of the project where the Team got opinions and feedback from cyclists.

The fifth and sixth sections of the WBS are shown in Figure A3. Both of these sections also required several updates to clarify the work that was done over the course of the year. These sections were quite general in the original WBS as the Team was unsure of what work would be required since it was so far down the line from the letter of intent. Therefore, section five, implementation, has been updated to provide a better overview of the tasks that were required to create the algorithm code and route packet. Similarly, the sixth section, project finale/close out, was updated to better reflect the work that was done to prepare for the design showcase. Overall, while the work breakdown structure shown in Figures A1 - A3 is still a very high level overview of the project, it now accurately reflects the work that was done over the course of the past year to complete this project.

6.2 Project Schedule

The second project management tool that was used throughout the course of the project is the project schedule and gantt chart. As with the WBS, the original project schedule is included in the project charter and the letter of intent. The updated schedule is shown in the appendix in Figures A4 & A5, both of which contain the same information just in different formats. Unlike the WBS, the tasks included in the project schedule did not change throughout the project; however, the timeline of the schedule did change throughout the project from what it originally was. The biggest change to the schedule from what was originally planned is that the data collection and background research portion of the project took much longer than originally expected. Originally, this part of the project was planned to conclude sometime in mid to late March, which would have given the Team enough time to start coding the algorithm during the first semester of the project. However, the Team ran into difficulties trying to collect the data needed for this project, especially since the original plan of using the Strava API did not work out. In addition, the Team had to spend longer than expected investigating other sources of data

such as departments of transportation because it was hard to get reliable contacts to determine what information was available from those sources. Overall, the Team spent most of the first semester on the data collection and background research portion of the project. Therefore, the start of risk analysis and the coding of the algorithm was moved to August when second semester started, instead of being in April like originally planned.

This caused the schedule for second semester to be a lot more compressed, which required the Team to shift some of the start dates for several tasks back to the second half of the second semester. In addition, the Team found that it originally misjudged the amount of time it took to complete many of the deliverables required for the class, such as the several reports and reviews. This caused the initial project schedule estimates to be even more inaccurate because work on the project often had to be paused in order to finish deliverables required for the class. This was not properly taken into account at first, which meant that in addition to the previous reasons for delay, many of the tasks were accomplished after what the original schedule had planned due to extra time spent working on the deliverables for the class. Overall, the update schedule shown in Figures A4 & A5 are completely up to date and accurately reflect the start and end dates of the tasks needed to complete the project.

6.3 Sprint Architecture

The final project management tool used by the Team was implemented during the second semester of the project, which is the sprint and scrum architecture. Basically, this is a project management method where the remaining work on the project is broken down into smaller tasks, and then a few tasks are chosen to be worked on every sprint. These sprints only last two weeks, and the point is to focus the entire Team's effort and just a few smaller tasks, and finish those tasks completely during the sprint. In order to do this, the Team created a sprint layout spreadsheet that contained the tasks broken down by topic for each sprint. These sprint layouts are shown in the appendix in Figures A6 - A9 for all seven sprints that comprised all of the work done during the second semester of the project. In these figures, there are tasks highlighted in different colors, which have different meanings. If the task is highlighted in green, it means that it was accomplished during that sprint. If a task that was assigned to a certain sprint was not finished during that sprint, it was initially highlighted in red. However, these tasks were then carried over to the next sprint, and if they were finished in that sprint, then the color was changed from red to blue. This is to signify that while that task was not completed on time during that sprint, it was eventually completed. As can be seen from Figures A6 - A9, all tasks for the project are highlighted in green and blue, meaning that all tasks have been completed. There are only two final items which still need to be completed per the sprint layout, which involve presenting all project deliverables to the client after the conclusion of the design showcase.

Overall, the sprint project management tool was much better than the gantt chart at keeping the Team on task because it focused on just a few specific tasks at a time.

6.4 Budget

As established in the project charter, there is no established budget for this project. This is because neither the client or the Team foresaw any significant expenses related to this project. Since this project is based almost entirely on research and coding programs available for free through the school, there was no need for a budget. However, the client did establish that if any expenses came up, they would be handled one at a time, and these expenses would be covered by the client if deemed necessary. However, throughout the course of the project, no such expenses incurred. Therefore, no budget was needed for this project.

6.5 Unfinished Items

While it is important to give updates on the project management tools that show how the project was accomplished, it is also important to discuss the future of the project. As for suggestions for the future development of the project, this is the topic of section 7 below. Therefore, this section will cover items and tasks that were not fully completed during this project. While there are not any parts of the project that are completely unfinished, there is one area that the Team has identified that could use additional work, which is with the route packet. While the Team believes that all the elements for a proper route packet have been created, there was not enough time to think through how the route packet could be revolutionized to provide a better experience for cyclists. Due to the time constraints of the project, the Team followed a lot of the elements seen in other race packets, but included the colorized hazard profile. While this is a step in the right direction, it could ultimately use a little more work to create a product that is more in tune with what cyclists want, rather than just following the examples that already exist. Other than this, the Team does not feel like any of the work done for the project is unfinished. Though there are definitely elements that can be built and improved upon, there aren’t any other areas that are purely unfinished. Therefore, this item, along with those given in section 7, represent possible areas of work in the future for the client or possibly another project team.

7 Recommendations for Future Development

7.1 Algorithm Interface

While this project has been a successful proof of concept for using Strava data to predict hazards for cyclists, there are several avenues that could be followed to develop it further. The algorithm and its outputs are valuable in their simplicity as they are generic enough to allow for adaptation with numerable different systems. However, it is clear that one shortfall of the project is the lack of a friendly user-interface. However, before an appropriate user interface is built, a decision on the direction of development must be made. Below are some of the top recommendations for further development of the algorithm.

7.2 Data Collection Automation

The top recommendation is to develop a way of further automating the data collection process from Strava. Currently, the algorithm analyzes one set of data from a single rider at any given time. This means that for any new analysis to be done, the desired route must first be found on Strava, as well as the rider ID for the fastest rider on that route. Then, a GET request must be performed for the data to be saved into a text file. Then the data must be manually copied into the algorithm, with time being required to reformat the data so it can be utilized inside the algorithm. While this process only takes a few minutes to complete, it becomes a barrier for mass utilization of the code across multiple routes and multiple riders. Instead, it would be ideal to develop a method where any given route is taken as an input from a user, and the steps required to collect the data and insert it into the algorithm are automated. With this improvement, several possibilities open up. First, it would make analysis of multiple riders along a single route more feasible. This would allow for several hazard profiles to be generated, and then averaged together, to yield an averaged hazard profile which should be more inclusive of multiple hazards which may have been missed or overestimated when analyzing a single rider. Second, it would be seemingly easy to launch a web service which people in the cycling community around the world would be able access and use for any route which Strava has data for. This is beyond the capability of most engineering disciplines however, and will likely require individuals with expertise in computer science to complete.

7.3 Real-time Feedback

The second recommendation is to adapt the algorithm for use with a portable device that can give cyclists real-time feedback. It was understood from the beginning of the project, that this is one of the ideas the foundation would like to pursue with this technology. As such, consideration was given throughout the design process to allow for future adaptation with real-time feedback. The requirements for this adaptation are fairly simple. The portable device must have some small processing capabilities, a GPS antenna, some storage capacity for keeping hazard profiles on the

device, and an indicator of some sort that the cyclist can interpret while riding. When the Team conducted the cycling safety survey at the Chad Young Memorial Time Trial, one of the questions asked to cyclists was “What would be an effective method of warning cyclists while riding?”. The results from this question can be seen in Figure 18. It is recommended that if real time feedback is pursued, that the responses from the cyclists themselves be considered before determining what the method of feedback will be.

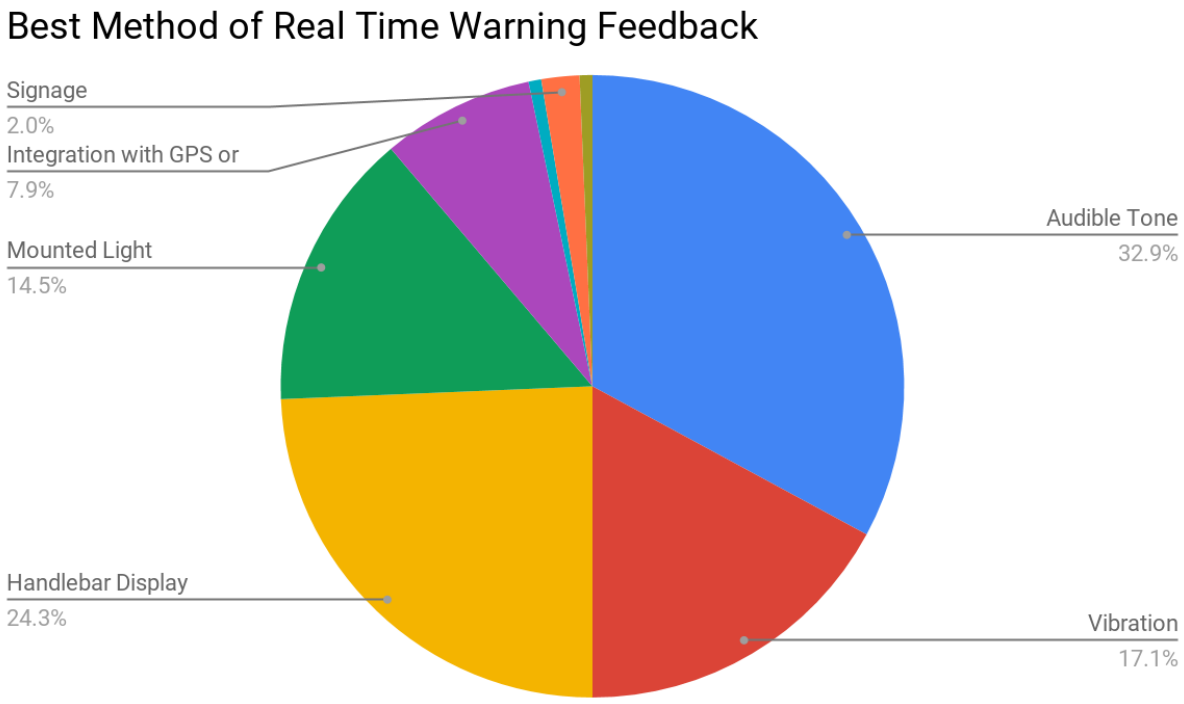

Figure 18: Possible Methods of Real-Time Feedback

The algorithm assigns a discrete hazard level to every GPS point recorded along the analyzed cyclists route. As such, at a bare minimum, any real-time feedback device needs to only find where the closest gps point is on the hazard profile to its current location, and convey the hazard level. There are several possible interpretations of this device, but two categories seem likely. First, a hazard profile for a route could be computed outside of the device and then transferred to the device. This has a drawback as it would only work while the user is riding along routes which have been transferred to the device. However, this cuts down on the level of computational hardware required within the device and as such would likely yield the smallest and most unobtrusive physical product for cyclists. The other category for a real-time feedback device is to develop a smartphone app which can pull data from Strava and run it through the algorithm in real-time. This eliminates the location drawbacks of the first category, and does not require the production of a specialized device as any rider with a smartphone could use it.

However, the disadvantage to using a smartphone again is the requirement of app developers and those with computer science expertise.

7.4 Additional Dataset Incorporation

The last recommendation is to continue expanding the algorithm by utilizing different datasets and different types of hazards. The principles used to analyze the available dataset from Strava are universal enough that they can also be applied to different hazard dataset with minimal effort. Every hazard must have a gps location, and a scale to determine the appropriate hazard multiplier. With those two pieces of information, integration into the current algorithm should be relatively seamless. Thus, if other data sets become available in the future, incorporating them into the algorithms analysis would undoubtedly prove beneficial. In particular, incorporation of traffic levels into the analysis was one of the most voiced concerns the Team received when interacting with cyclists.

7.5 Recommended Skill Sets

All the aforementioned recommendations require a good deal of computer science expertise. Fortunately, this expertise is relatively easy to acquire from a myriad of sources including app development firms, and computer science degree programs. The Colorado School of Mines offers summer project services to clients who wish to work with students. If it is the Foundations wish to continue their work with Colorado School of Mines students, they may find it beneficial to reach out to the Computer Science Department to inquire about their summer services.

8 Lessons Learned

There are many lessons that the Team learned over the course of the year working on this project. Perhaps the most critical lesson the Team learned is the importance of being an interdisciplinary engineer. Unfortunately, our Team did not have any diversity in terms of major, as the Team was composed of five mechanical engineers. However, most of this project revolve around coding and algorithm development, and had almost nothing to do with mechanical design. At first this was slightly discouraging because it seemed as if the Team was not really prepared for what we were being asked to do. However, this is what taught us the importance of reaching beyond the mechanical engineering curriculum, and learning to become a multidisciplinary engineer. While it was difficult at times to accomplish certain tasks due to the Team's lack of coding experience, it has made everyone on our Team a better engineer. In a work environment, while we may often deal with mechanical issues, it isn't always so clear cut. Sometimes we will be asked to work on a project that involves electrical, chemical, or coding elements. The best engineers will be able to take on these assignments and learn what they need to while working in order to adapt to the situation on hand. This is what we did on this project. When there were gaps in our knowledge in terms of coding or data analysis techniques, we sought out appropriate resources from which we learned the appropriate information necessary in order to finish the project. Overall, this process of stepping outside of our comfort zone to tackle a project outside our area of expertise has given us the skills to tackle projects in industry that we might otherwise have felt unprepared for.

Another very important lesson we learned over the course of the semester was the value of stakeholder outreach and engagement. No team member was an avid cyclist and as such felt that it was critically important for us to get out and talk to cyclists about what they thought are the biggest hazards facing cyclists, what they would like to see in a preventative solution, and if they would see themself using such a product. Luckily, Golden is full of cyclists who were willing to help, including cyclists from the Colorado School of Mines Cycling Team and the surrounding community. Stakeholder engagement is critically important because we wanted to make sure that we were developing a product that will meet cyclists needs, not a product that meets what we think cyclists need. In addition, we wanted to make sure that we developed the product in such a way that cyclists would want to use it. There are many examples where companies are not in touch with their stakeholders when they create a product, and many times the product is not successful. Therefore, in order to avoid this pitfall, we surveyed cyclists after the Lookout Mountain Memorial Time Trial, and we involved cyclists in the testing of the algorithm. This informed many aspects of the project including what manual hazards were included in the algorithm, the weighting of the hazard multipliers, and the accuracy of the algorithm. Without a doubt, involving cyclists in the development of the algorithm has made it more capable and accurate than it would have otherwise been.

One of the other lessons that we learned over the past year is how to deal with situations that present obstacles that are out of our control and how to get around those obstacles. The best example of this for our project came when we were looking into different methods of data collection for the algorithm. While we were initially steered toward Strava from the start, we still wanted to see what other options were possibly available to us. One such method that we looked into was gathering data from local Departments of Transportation. However, we soon ran into difficulties with this method because DOT was unresponsive to inquiries. Ultimately we decided that these issues with correspondence were outside of the Team's control, and we eventually agreed that we needed to focus our time and effort elsewhere, which is how Strava was selected. The original plan for the project was to use an API (Application Programing Interface) from Strava itself to gain access to this information. However, the Team was again not able to get any correspondence with Strava established despite our best efforts, which made using their API difficult. Ultimately, the Team had to find another way around this issue, which we did by reaching out to a knowledgeable programer, who gave us the idea of using the GET request. Overall, these problems and our subsequent solutions show how we were able to find ways around the roadblocks that came up over the course of the year.

Finally, in several aspects of the project our experience turned out differently than we originally thought they would be. One aspect of the project that definitely ended up being much harder than we originally thought it would be was the data collection. This was partially due to our lack of coding experience, but we originally thought it would be rather easy to gather data with the Strava API. However, when it became apparent that this would not work, the Team had a difficult time finding a replacement option. As previously mentioned, many of the options the Team thought of were either not feasible, or had other issues with them that were out of our control. Ultimately we caught a bit of a break by learning about the GET request, which allowed us to gather data from Strava. However, this did impact our overall project, because the data collection portion of our project lasted much longer than the original project schedule planned for. This meant that we weren't able to get to much of any coding during the first semester, which compressed the schedule for the second semester even more.

Another aspect of the project that ended up being harder than we originally planned was in determining the multipliers to use with the automatic and manual algorithms. At first the Team thought that it would be more difficult to calculate what was needed for the algorithm, but then coming up with the multipliers would be easier. However, it ended up being the opposite. The Team probably underestimated the difficulty of coming up with the multipliers because early on we created a table of what the 1-5 scale meant. It seemed it would be easy to come up with the whole list of multipliers because the descriptions were general statements like no danger, slight danger, or very dangerous. Of course the table was more detailed than this, but even so, it seems easier to apply when the scale is so general. However, when we begun to actually come up with and assign the multipliers for the different type of hazard, it became much more difficult. This is

because while it was easy to make the distinction between what is a 2 and a 3, it was much more difficult to distinguish between what was a 2.2 and a 2.3. In creating and assigning the multipliers, we had to determine not only what was a reasonable value for the hazard, but also justify why a 2.5 wasn't a 2.4 or a 2.6. Overall, this process ended up being more difficult and took longer than originally planned. This difficulty is part of the reason we begun to rely more on testing of the algorithm to get the multipliers perfect, rather than trying to do so solely out of analysis.

On the other hand, there are also parts of the project that ended up being easier than we first thought. One of these parts was the calculations necessary to calculated the hazards for the automatic algorithm. When first starting the project, we planned a lot of time for determining and implementing these equations because we thought there would be many complex and interconnected equations. However, as it turns out, the basic characteristics of a rider and the basica geometry of a route are both pretty easy to work with. In addition, the limited nature of the data that we gathered from Strava limited the data we had to use in calculations. Overall, we found that using velocity, acceleration, turn radius, and grade to predict hazards resulted in an accurate hazard profile. Therefore, we decided to stick with these calculations, which are all fairly simple, and keep the algorithm simple but accurate, rather than going crazy with physics and dynamics equations. In addition, this allows the algorithm to run faster, which will be helpful if this product is used for real time feedback in the future. Overall, it is not so much that we found the calculations themselves as easier than we were expecting, but rather, that what we had to calculate was easier than we first thought. This is also what helped us to regain some time in our schedule that we had lost due to the data collection taking longer than expected.

Finally, the other aspect of our project that we found easier than expected was the cyclist outreach. At first, the Team only knew a handful of cyclists. We were unsure of how to get in touch with a large number of cyclists of all skill levels. However, the Lookout Mountain Memorial Time Trial was a big help with this. This was an event that we were unaware of at the beginning of the project, and didn't find out about it until the Foundation mentioned it during a client meeting. This made outreach to cyclists from all different skill levels much easier and led to data that made a great impact of the project. In addition, the Team thought it would be difficult to find enough cyclists to test the algorithm for accuracy. Again, the Team already had a few cyclists in mind who were willing to test the algorithm, but we wanted to get as many cyclists involved as possible in order to validate the feedback we were getting. However, the Team was able to come up with a method for cyclists to test they algorithm on their own, and then report the results back at a later time, which allowed us to give this information to the Mines Cycling Team. This has allowed us to reach many more cyclists than we possibly could have if we were accompanying every cyclist as they rode the route. Overall, we have been very fortunate on this project to have great involvement from cyclists which has helped shape our algorithm.

9 Conclusion

The purpose of this final design review was to present the product that Team Cruise Control has created in its entirety from start to finish. This design report has conveyed how the product works, discussing everything from the data collection, to the data processing in the algorithm, to the algorithm outputs, and finishing with how this information is presented to cyclists. In addition, its included information on various project management details, possible future implementations for this product, and the lessons that the Team learned while working on this project. All of this information will be vital for getting any future groups up to speed on the current state of the project and how it works. In addition, the contents of this final design report conveys various options that Team Cruise Control pursued over the course of the year, as well as which of these worked and which did not work. This should also provide valuable information to the client and any future participants in this project so that they can avoid the issues that this Team already faced. Finally, before wrapping up this report, it is important to review the project requirements one last time to make sure that this project successfully accomplished all of its goals. These requirements are given in the the client needs table shown in Table 1. These requirements deal with collecting hazard data, calculating risk at a discrete location and along an entire route, conveying this information to race organizers and recreational cyclists alike in the form of a route packet, and finally, putting together this final design report. This final design report has shown that Team Cruise Control has successfully met all six of these project requirements with the creation of the algorithm, route packet, and other supporting documents. Therefore, Team Cruise Control has successfully completed this project for the Chad William Young Foundation, and hopes that this work will help increase the safety of cycling by preventing dangerous crashes before they ever happen.

Appendix

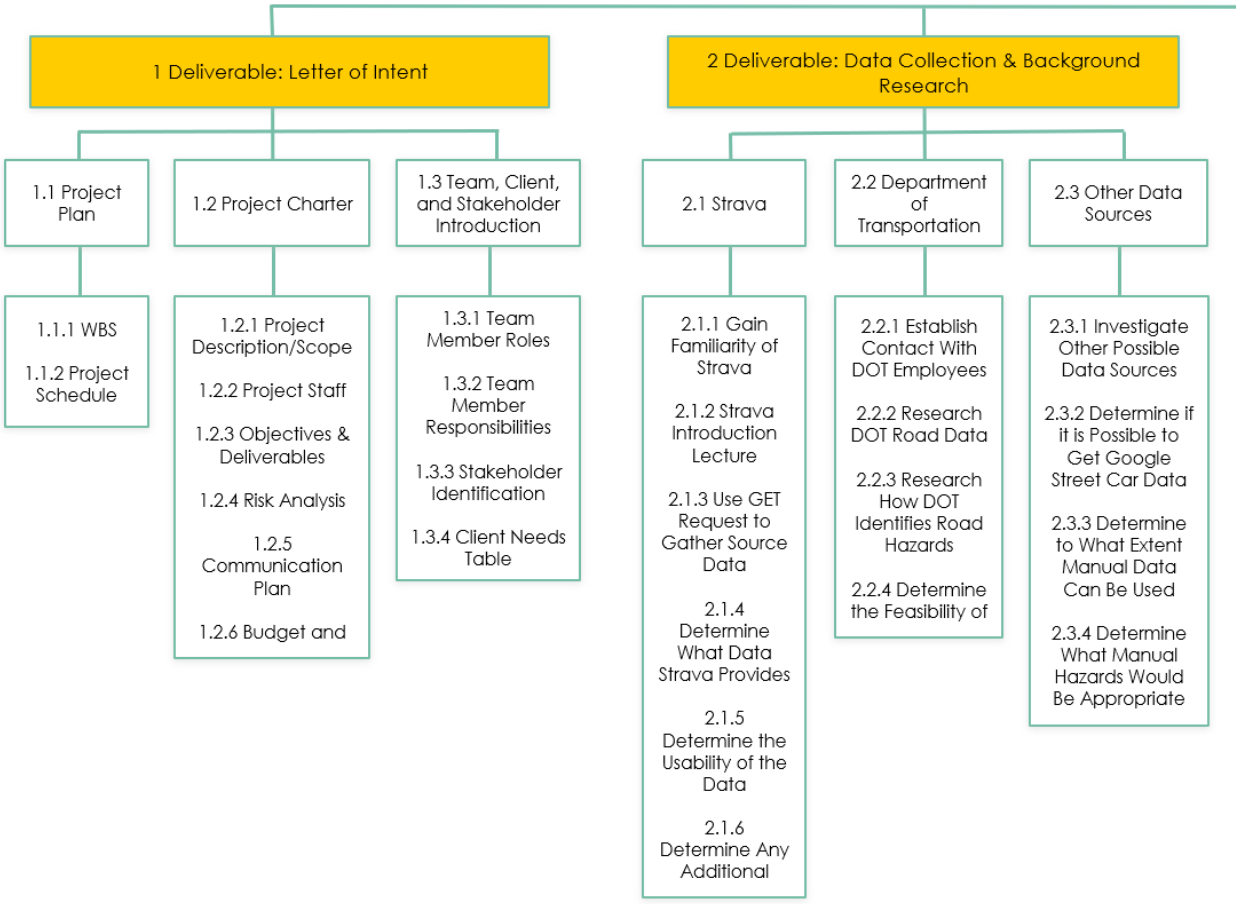

Figure A1: Updated Work Breakdown Structure Sections 1 & 2

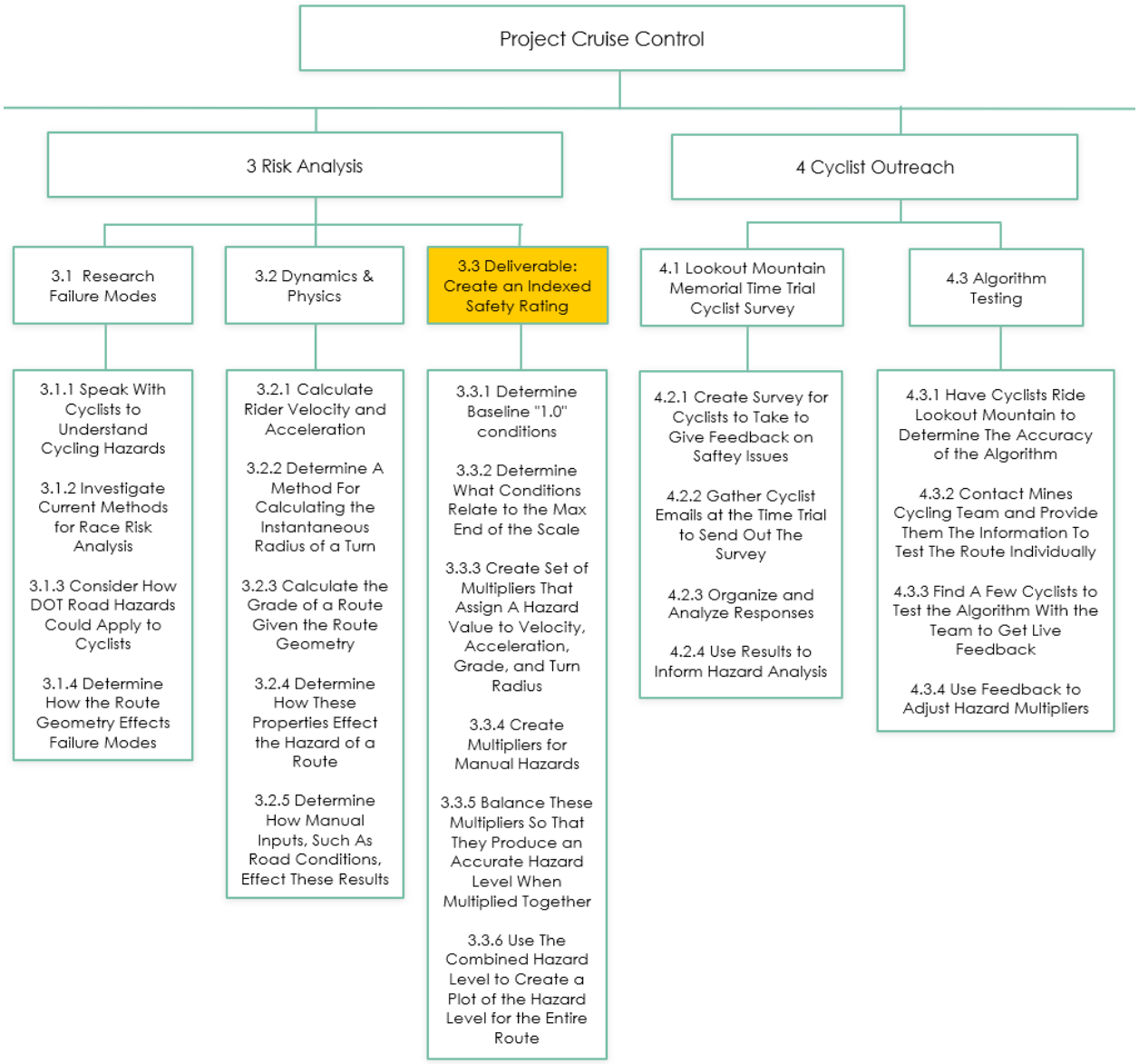

Figure A2: Updated Work Breakdown Structure Sections 3 & 4

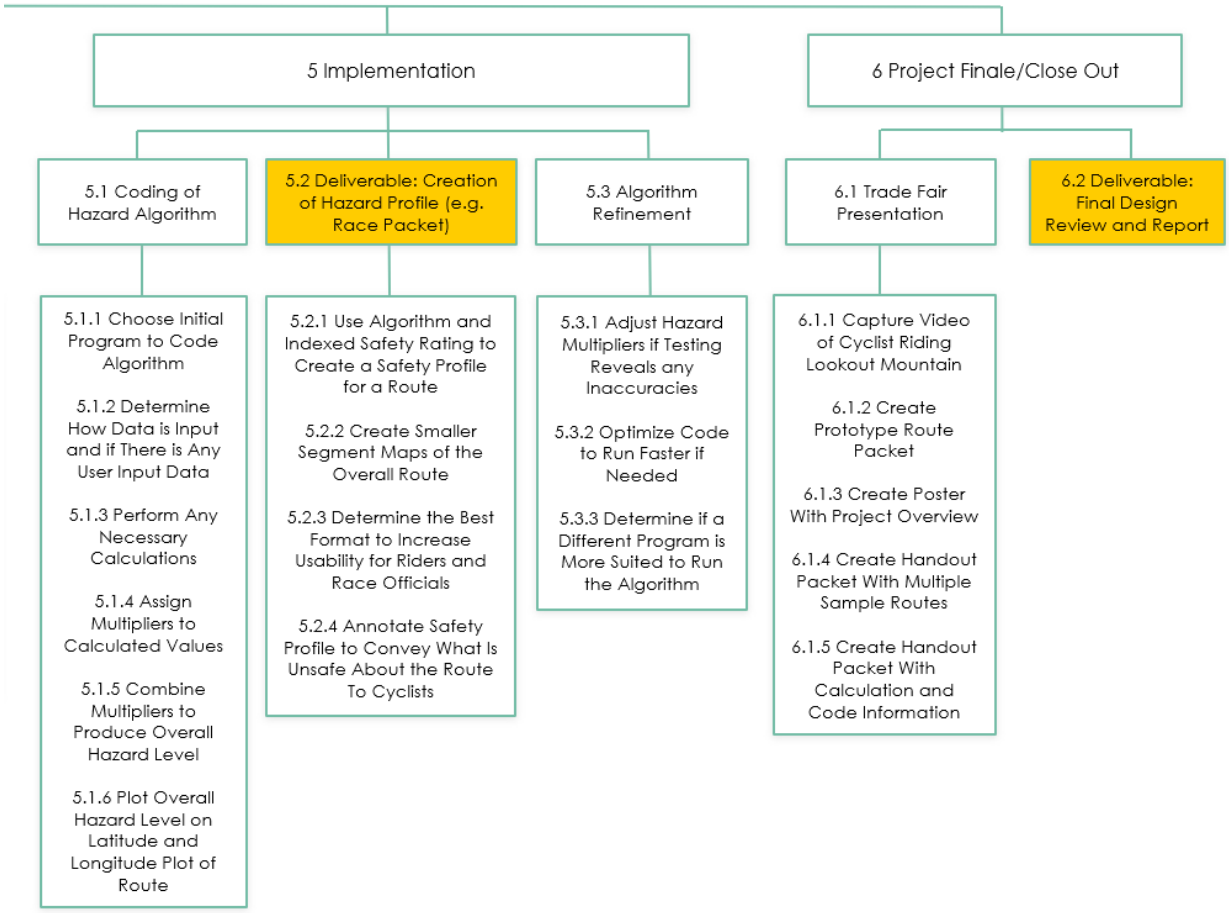

Figure A3: Updated Work Breakdown Structure Sections 5 & 6

Project: Cyclist Injury Prevention Challenge						
Team: Cruise Control (SD-13)		Date: February 18, 2018 (Rev 0)				
WBS Task	**Start Date**	**End Date**	**Duration (Days)**	**Days Complete**	**Days Remaining**	**Percent Complete**
Client & Project Work / Deliverables						
1 Deliverable: Letter of Intent	2/1/2018	2/22/2018	21	21.00	0.00	100%
1.1 Project Plan	2/1/2018	2/22/2018	21	21.00	0.00	100%
1.2 Project Charter	2/1/2018	2/22/2018	21	21.00	0.00	100%
1.3 Team, Client, and Stakeholder Introduction	2/1/2018	2/22/2018	21	21.00	0.00	100%
2 Deliverable: Data Collection & Background Research	2/15/2018	4/15/2018	59	59.00	0.00	100%
2.1 Strava	2/15/2018	4/15/2018	59	59.00	0.00	100%
2.2 Department of Transportation	2/15/2018	3/15/2018	28	28.00	0.00	100%
3 Risk Analysis	8/20/2018	11/1/2018	73	73.00	0.00	100%
3.1 Research Failure Modes	8/20/2018	9/15/2018	26	26.00	0.00	100%
3.2 Dynamics & Physics	9/15/2018	10/11/2018	26	26.00	0.00	100%
3.3 Deliverable: Create an Indexed Safety Rating	10/15/2018	11/1/2018	17	17.00	0.00	100%
4 Implementation	8/20/2018	12/1/2018	103	103.00	0.00	100%
4.1 Coding of Hazard Algorithm	8/20/2018	11/15/2018	87	87.00	0.00	100%
4.2 Deliverable: Creation of Hazard Profile	10/15/2018	11/15/2018	31	31.00	0.00	100%
4.3 Testing and Refinement	11/1/2018	12/1/2018	30	28.50	1.50	95%
5 Project Finale/Close Out	11/15/2018	12/4/2018	19	19.00	0.00	100%
5.1 Trade Fair Presentation	11/15/2018	12/4/2018	19	17.10	1.90	90%
5.2 Deliverable: Design Report	11/15/2018	11/29/2018	14	14.00	0.00	100%

Senior Design Class Deliverables / Assignments						
1 Letter of Intent	2/1/2018	2/22/2018	21	21.00	0.00	100%
2 Concept Portfolio	3/1/2018	3/15/2018	14	14.00	0.00	100%
3 Preliminary Design Review	3/25/2018	4/10/2018	16	16.00	0.00	100%
4 Concept Critique	4/10/2018	4/17/2018	7	7.00	0.00	100%
5 Next Steps Letter	4/25/2018	5/1/2018	6	6.00	0.00	100%
6 Project Packet	4/20/2018	5/3/2018	13	13.00	0.00	100%
7 Intermediate Design Review Presentation	9/10/2018	9/20/2018	10	10.00	0.00	100%
8 Final Design Review Presentation	11/1/2018	11/15/2018	14	14.00	0.00	100%
9 Trade Fair Presentation	11/15/2018	12/4/2018	19	17.10	1.90	90%
10 Design Report	11/15/2018	11/29/2018	14	14.00	0.00	100%

Figure A4: Updated Project Schedule Table

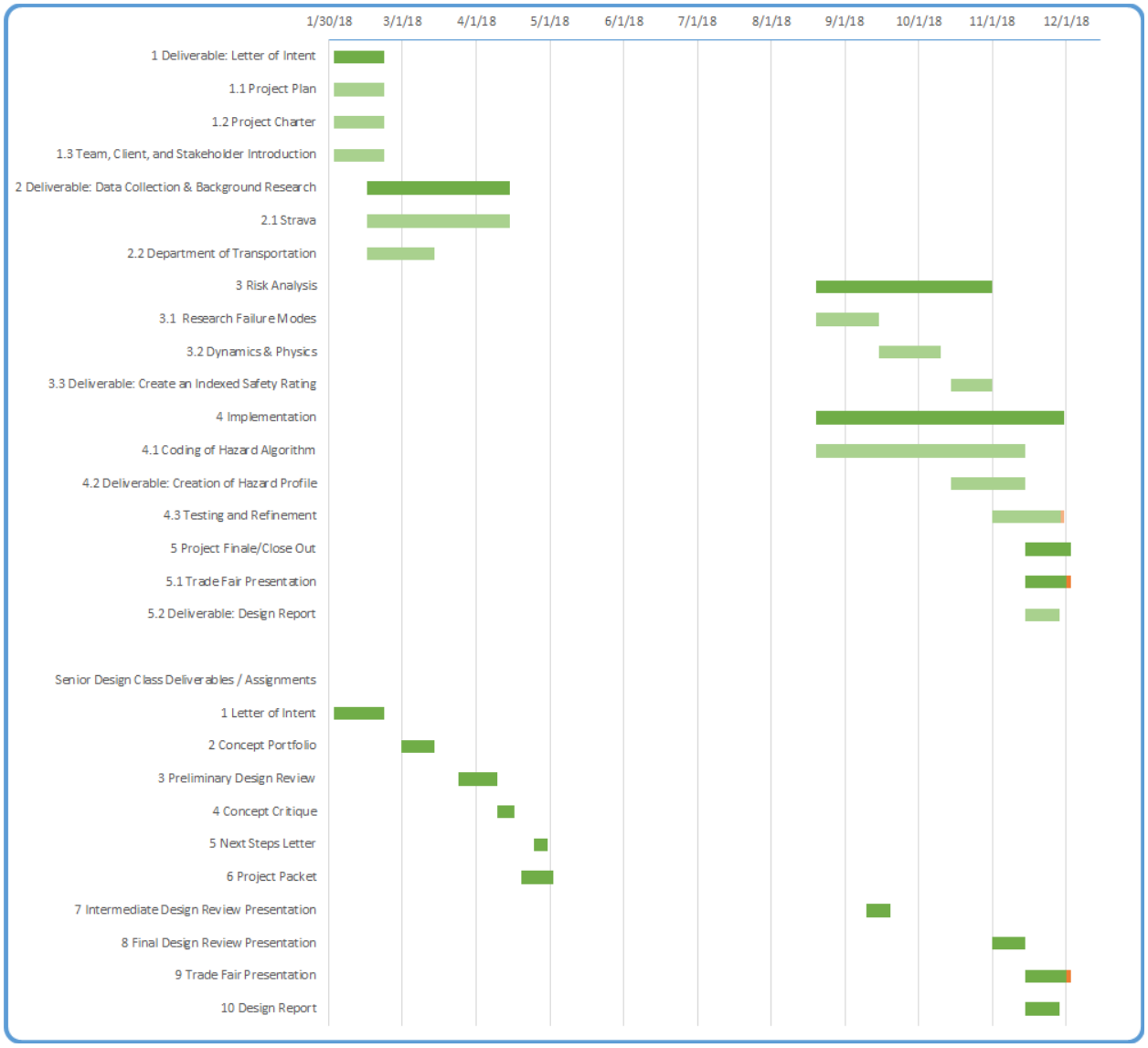

Figure A5: Updated Project Schedule Gantt Chart

Sprint 1		Sprint 2	
8/30/2018	9/13/2018	9/13/2018	9/27/2018
1) Risk Analysis		1) Intermediate Design Report	
a) Determine what hazards to code/display		a) Hold Intermediate Design Review	
b) Determine method for automatic hazard analysis		b) Write Intermediate Design Report	
c) Develop hazard analysis equations/algorithm		2) Risk Analysis	
2) Weight Scale		a) Develop hazard analysis equations/algorithm	
a) Decide on color/numeric scale		3) Weight Scale	
b) Begin developing risk mulitpliers and rational		a) Decide on color/numeric scale	
3) Algorithm Development		b) Begin developing risk mulitpliers and rational	
a) Experiment with combining multiple data sets		4) Algorithm Development	
b) Work on running average to normalize data		a) Colorizing data sets/graphs, linetypes	
c) Colorizing data sets/graphs, linetypes		b) Graphing individual hazards	
d) Graphing individual hazards			
4) Client Communication			
a) Kickoff Meeting - Intros			
b) Client Buyoff on Final Deliverables			

Figure A6: Sprint Layout 1 & 2 Tasks

Sprint 3		Sprint 4	
9/27/2018	10/11/2018	10/11/2018	10/25/2018
1) Risk Analysis		1) Risk Analysis	
a) Finalize hazard analysis equations for automatic analysis		a) Determine how to change lat and long data into x and y	
b) Begin determining how manual hazard effect route		b) Begin determining how manual hazard effect route	
2) Weight Scale		c) Continue looking into interpolation method	
a) Finalize hazard multipliers for automatic hazards		2) Weight Scale	
b) Begin determining the multipliers for manual hazards		a) Finalize hazard multipliers for automatic hazards	
3) Algorithm Development		b) Begin determining the multipliers for manual hazards	
a) Begin coding of the automatic hazard analysis protion		3) Algorithm Development	
b) Colorizing data sets/graphs, linetypes		a) Finalize coding of the automatic hazard analysis protion	
4) Algorithm Testing Program		b) Colorizing data sets/graphs, linetypes	
a) Set date for Lookout Mountain walkthrough		4) Algorithm Testing Program	
5) Class Deliverables		a) Create Rigid Plans For Test Day	
a) Write the Detailed Design Critique		5) Class Deliverables	
		a) Write the Project Synopsis	

Figure A7: Sprint Layout 3 & 4 Tasks

Sprint 5		Sprint 6	
10/25/2018	11/8/2018	11/8/2018	11/22/2018
1) Risk Analysis		1) Algorithm Development	
a) Determine if interpolation method is still necessary		a) Finish programing the manual algorithm	
2) Weight Scale		b) Make any necessary adjustments coming from testing	
a) Finalize hazard multipliers for manual hazards		c) Optimize Automatic Algorithm	
3) Algorithm Development		2) Algorithm Testing	
a) Refine the automatic algorithm		a) Send out test packet to cyclists	
b) Begin Programing of the manual algorithm		b) Gather feedback from cyclists	
c) Graphing individual hazards on the existing plot		c) Get raw fotage for video	
4) Algorithm Testing		3) Class Deliverables	
a) Begin testnig of the algorithm		a) Give the Final Design Review	
5) Class Deliverables		b) Create draft of showcase poster	
a) Finish final design review slides		c) Create draft of the final report	

Figure A8: Sprint Layout 5 & 6 Tasks

Sprint 7	
11/22/2018	12/6/2018
1) Algorithm Development	
a) Finish programing the manual algorithm	
b) Make any necessary adjustments coming from testing	
c) Produce plots for various other routes	
3) Class Deliverables	
a) Finalize showcase poster	
b) Finish the Final Design Report	
c) Anotate the video for the showcase	
d) Prepare digital design archive	
4) Project Closout	
a) Give all deliverable to the client	
b) Complete the checkout process for senior design	

Figure A9: Sprint Layout 7 Tasks

```
clear
clc

LatLong = [39.712943 -105.261579 39.712992 -105.261491 39.713035 -105.261397 39.713071 -105.261307 39.71
time = [0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 3
distance = [0.0 9.3 18.7 27.3 37.0 44.7 52.2 60.4 69.1 78.2 86.5 95.0 104.7 114.0 122.6 132.2 141.5 150.
altitude = [2271.5 2270.9 2270.3 2269.8 2269.9 2270.7 2271.3 2271.6 2271.9 2272.2 2272.5 2272.7 2273.0 2
manHaz = [39.7328203 -105.2373454 39.7340654 -105.2395680 39.7327430 -105.2421003 39.735322 -105.2457391

x = (1:length(time));
```

Figure A10: Code Overview - Data Input and Initialization

```
10 -    x = (1:length(time));
11
12 -    GradeMulti = [1.1 1.2 1.3 1.4 1.5 1.5 1.6 1.6 1.7 1.7 1.8 1.8 1.8 1.9 1.9 2.0];     %Length of 16
13 -    CurveMulti = [1.1 1.2 1.3 1.4 1.5 1.5 1.6 1.6 1.7 1.7 1.8 1.8 1.8 1.9 1.9 2.0];     %Length of 16
14 -    VelocityMulti = [1.1 1.2 1.3 1.4 1.5 1.6 1.7 1.8 1.9 2.0];                          %Length of 10
15 -    AccMulti = [1.1 1.3 1.4 1.6 1.7 1.8 1.8 1.9 1.9 2.0];                               %Length of 10
16 -    BNeck = [1.3 1.6 2.0 1.5];          %   25%, 50%, 75%+ traffic, 1.5 else
17 -    PHole = [1.2 1.4 1.6 2.0];          %   Small shallow, moderate size, huge, severe unavoidable dis
18 -    Blind = [1 1.2 1.5 1.8 2.0];        %   Fully visible, small visibility disruption, some visibilit
19 -    Camber = [0.8 1.2];                 %   Leaning into turn, leaning away from turn
20 -    Traffic = [1.0 1.2];                %   No traffic, on road with minimal traffic
21 -    numRiders = [0.8 1.0 1.2 1.4 1.6];  %   Alone, low, moderate, high, race amount
22 -    Weather = [1.2 1.4 1.6];            %   Damp, moderately wet, very wet
23 -    RoadMaterial = [1.2 1.5 1.6 1.7];   %   Some dirt/gravel, dirt/gravel on paved surface, lots of di
24
25 -    Lat = [];
26 -    Long = [];
27
28 -    velocity = [];
29 -    acceleration = [];
30 -    runAvgVel = [];
31 -    runAvgGrade = [];
32 -    grade = [];
33 -    X = [];
34 -    Y = [];
35 -    manLat = [];
36 -    manLong = [];
37 -    manMulti = ones(1,10);
38 -    j = 1;
39 -    k = 1;
```

Figure A11: Code Overview - Multiplier Assignment and Initialization

```
41          %Creates Latitude and Longitude arrays
42
43 -  for i = 1:length(LatLong)
44
45 -      if mod(i,2) == 0
46
47 -          Long(j) = LatLong(i);                %This works correctly
48
49 -          j = j + 1;
50
51 -      else
52
53 -          Lat(k) = LatLong(i);                 %This works correctly
54
55 -          k = k + 1;
56
57 -      end
58
59 -  end
60 -  j = 1;
61 -  k = 1;
62 -  for B = 1:length(manHaz)
63 -      if mod(B,2) == 0
64 -          manLong(j) = manHaz(B);                %This works correctly
65 -          j = j + 1;
66 -      else
67 -          manLat(k) = manHaz(B);                 %This works correctly
68 -          k = k + 1;
69 -      end
70 -  end
71 -  tol = 1E-4;
72 -  actLat = [];
73 -  actLong = [];
74 -  for Z = 1:length(Lat)
75 -      for W = 1:length(manLat)
76 -          if ((abs(manLat(W) - Lat(Z)) < tol) && (abs(manLong(W) - Long(Z)) < tol))
77 -             actLat(W) = manLat(W);
78 -             actLong(W) = manLong(W);
79 -          end
80 -      end
81 -  end
```

Figure A12: Code Overview - Creating Latitude and Longitude Arrays

```
84          %Creates velocity and acceleration arrays
85
86 -    for i = 2:(length(LatLong)/2)
87
88 -        velocity(i) = (distance(i) - distance(i-1))/(time(i) - time(i-1));
89
90
91 -    end
92
93 -     runAvgVel(1) = velocity(1);
94
95          %Creates average running velocity array runAvgVel and acc
96 -     acceleration(1) = 0;
97 -    for H = 2:length(velocity)-1
98
99 -        runAvgVel(H) = ((velocity(H-1) + velocity(H) + velocity(H+1))/3);
100 -       acceleration(H) = (runAvgVel(H)-runAvgVel(H-1))/(time(H) - time(H-1));
101 -   end
102 -    runAvgVel(length(velocity)) = velocity(end);
103 -    acceleration(length(velocity)) = 0;
104
```

Figure A13: Calculating Velocity and Acceleration Arrays

```
105         %Creates array for Grade
106 -   Q = 1;  %KEEP AS 1 FOR NOW
107
108 -    grade(1:Q) = 0;
109
110 -   for N = Q+1:length(distance)-Q
111
112 -       grade(N) = 100*(altitude(N+Q) - altitude(N-Q)) / (distance(N+Q) - distance(N-Q));
113
114 -   end
115
116 -    grade(length(velocity)) = grade(length(velocity)-Q);
117 -    grade(length(velocity)-Q+1) = grade(length(velocity)-Q);
118
119 -    runAvgGrade(1) = grade(1);
120
121 -   for H = 2:length(velocity)-1
122
123 -       runAvgGrade(H) = ((grade(H-1) + grade(H) + grade(H+1))/3);
124
125 -   end
126
127 -    runAvgGrade(end+1) = runAvgGrade(end-1);
```

Figure A14: Calculating Grade Array

```
129     % Turn Radius
130
131 -   R = 6371000;
132 -   X(1) = 0;
133 -   Y(1) = 0;
134
135 -   for i=1:length(Lat)
136 -       LatR(i) = Lat(i)*pi/180;
137 -       LongR(i) = Long(i)*pi/180;
138
139 -       if i>1
140 -           X(i) = (LongR(i)-LongR(1)) * cos((LatR(i)+LatR(i-1))/2)*R;
141 -           Y(i) = (LatR(i) - LatR(1)) * R;
142 -       end
143
144 -   end
145
146     % Calculate Turn Radius
147
148     % calculate Distances between points
149 -   for i = 1:length(Lat)-1
150 -       L(i) = ((X(i+1)-X(i))^2 + (Y(i+1)-Y(i))^2)^0.5;
151 -       if i>1
152 -           L3(i) = ((X(i+1)-X(i-1))^2 + (Y(i+1)-Y(i-1))^2)^0.5;
153 -       end
154 -       for j = 1:3
155 -           if j == 1
156 -               LV(i,j) = [X(i+1)-X(i)];
157 -           elseif j == 2
158 -               LV(i,j) = [Y(i+1)-Y(i)];
159 -           elseif j==3
160 -               LV(i,j) = 0;
161 -           end
162 -       end
163 -   end
```

Figure A15: Calculating Turning Radius Array Part 1 of 2

```
165     % Calculate area of vectors
166 -   for i = 2:length(Lat)-1
167 -       A(i) = 0.5 * norm(cross(-LV(i-1,:),LV(i,:)));
168 -   end
169
170     % Calculate Turn Radius
171 -   for i = 2:length(Lat)-1
172 -       r(i) = (L(i-1)*L(i)*L3(i))/(4*A(i));
173 -   end
```

Figure A16: Calculating Turning Radius Array Part 2 of 2

```
175 -   r(1) = r(2);
176 -   r(length(velocity)) = r(end-1);
177
178 -   GradeMultiX = -1:-1:-16;
179 -   CurveMultiX = 150:-10:10;
180 -   CurveMultiX(16) = 2;
181 -   VelocityMultiX = 0:5:45;
182 -   AccMultiX(1) = 0.01;
183 -   AccMultiX(2:10) = 1:0.5:5;
184
185 -   hazard_level = [];
186
187 -   GML = [];
188 -   RML = [];
189 -   VML = [];                    %VelMultiLoop
190 -   AML = [];
191
192 -   for A = 1:length(velocity)
193
194 -       for P = 1:length(GradeMultiX)
195
196 -           if runAvgGrade(A) > GradeMultiX(1)              %GRADE
197 -               GML(A) = 1;
198 -           end
199
200 -           if runAvgGrade(A) <= GradeMultiX(end)
201 -               GML(A) = GradeMulti(end);
202 -           end
203
204 -           if runAvgGrade(A) > GradeMultiX(P)
205
206 -               GML(A) = GradeMulti(P);
207 -               break
208 -           end
209
210 -       end
```

Figure A17: Assigning Automatic Multiplier Values Part 1 of 3

```
212 -     for P = 1:length(CurveMultiX)            %RADIUS
213
214 -         if r(A) >= CurveMultiX(1)
215 -             RML(A) = CurveMulti(1);
216 -         end
217
218 -         if r(A) <= CurveMultiX(end)
219 -             RML(A) = CurveMulti(end);
220 -         end
221
222 -         if r(A) > CurveMultiX(P)
223
224 -             RML(A) = CurveMulti(P);
225 -             break
226 -         end
227
228 -     end
229
230 -     for R = 1:length(VelocityMultiX)
231
232 -         if runAvgVel(A) <= VelocityMultiX(1)
233 -             VML(A) = VelocityMulti(1);
234 -         end
235
236 -         if runAvgVel(A) >= VelocityMultiX(end)
237 -             VML(A) = VelocityMulti(end);
238 -         end
239
240 -         if runAvgVel(A) < VelocityMultiX(R)
241
242 -             VML(A) = VelocityMulti(R);
243 -             break
244 -         end
245
246 -     end
247
```

Figure A18: Assigning Automatic Multiplier Values Part 2 of 3

```
248 -       for R = 1:length(AccMultiX)
249
250 -           if acceleration(A) <= AccMultiX(1)
251 -               AML(A) = AccMulti(1);
252 -           end
253
254 -           if acceleration(A) >= AccMultiX(end)
255 -               AML(A) = AccMulti(end);
256 -           end
257
258 -           if acceleration(A) < AccMultiX(R)
259
260 -               AML(A) = AccMulti(R);
261 -               break
262 -           end
263
264 -       end
265
266 -   end
267
268 -   hazard_level = GML.*VML.*AML.*RML;
269
270 - for i=1:length(hazard_level)
271
272 -       if hazard_level(i) > 5
273
274 -           hazard_level(i) = 5;
275
276 -       elseif hazard_level(i) < 1
277
278 -           hazard_level(i) = 1;
279
280 -       end
281 -   end
282
283 -   bigFactor = [];
```

Figure A19: Assigning Automatic Multiplier Values Part 3 of 3

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
MML = [2.648 4.011 2.648 15.667 11.754 3.972 5.014 3.310 6.016 3.310];
manHazLevel = ones([1,length(Lat)]);
for C = 1:length(Lat)
    for W = 1:length(MML)
        if ((Lat(C) == actLat(W)) && (Long(C) == actLong(W)))
            manHazLevel(C) = MML(W);
        end
    end
end
hazardLevel = GML.*VML.*AML.*RML.*manHazLevel;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
for B = 1:length(hazard_level)
    if hazard_level(B) > 4.75
        if AML(B) > RML(B)
            if AML(B) > GML(B)
                if AML(B) > VML(B)
                    bigFactor(B) = AML(B);
                end
            end
        elseif RML(B) > GML(B)
            if RML(B) > VML(B)
                bigFactor(B) = RML(B);
            end
        elseif GML(B) > VML(B)
            bigFactor(B) = GML(B);
        else
            bigFactor(B) = VML(B);
        end
    end
end

BFRange = 1:length(bigFactor);
FactRange = 1:length(AML);
```

Figure A20: Assigning Manual Multiplier Values

```
colo = zeros(64,3);
s = 1/16;

for i = 1:64
    if i <= 16
        colo(i,:) = [0 (0+i/16) 1];

    elseif i<=32
        colo(i,:) = [0 1 1-(i-16)/16];

    elseif i<= 48
        colo(i,:) = [(i-32)/16 1 0];

    elseif i<= 64
        colo(i,:) = [1 1-(i-48)/16 0];
    end
end
```

Figure A21: Creation of Colormap

```
% % Initialize Color Scale for Plot

x = Long(265:end);
y = Lat(265:end);
z = zeros(size(x));
col = randi(5,length(time),1);
surface([x;x],[y;y],[z;z],[col;col],...
        'facecol','no',...
        'edgecol','interp',...
        'linew',3);
colormap(colo)

% Overwrite Plot with Actual Data

x = Long(265:end);
y = Lat(265:end);
z = zeros(size(x));
col = hazard_level(265:end);
surface([x;x],[y;y],[z;z],[col;col],...
        'facecol','no',...
        'edgecol','interp',...
        'linew',3);
colormap(colo)
xlabel('Longitude (Deg)');
ylabel('Latitude (Deg)');
title('Lookout Mountain Downhill');
hcb = colorbar;
title(hcb,'Hazard Level');
set(gca,'FontSize',24)
```

Figure A22: Plotting of Hazard Profile

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%  PLOTS      %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

    %Average Velocity Plot
hold on
plot(x,runAvgVel);
plot(x,velocity);
legend('runAvgVel')%,'Velocity');
title('Running Average Velocity');

    %Portion of LatLong Plot
plot(Lat(1:120),Long(1:120));
xlabel('Latitude');
ylabel('Longitude');
title('Lookout Mountain Latitude and Longitude');

    %LatLong Plot
plot(Lat,Long);
xlabel('Latitude');
ylabel('Longitude');
title('Lookout Mountain Latitude and Longitude');

    %Distance vs Acceleration Plot
plot(distance,acceleration);
xlabel('Distance Traveled');
ylabel('Acceleration');
title('Lookout Mountain Distance Traveled and Acceleration');

    %Distance vs Altitude Plot
plot(distance,altitude);
xlabel('Distance Traveled');
ylabel('Altitude');
title('Lookout Mountain Distance Traveled and Altitude');

    %Grade Plot
Z = 1;
plot(distance(Z:end),runAvgGrade(Z:end));
xlabel('Distance');
ylabel('Running Average Grade %');
title('Grade Plot');

%%%%%%%%%%%%%%%%%%%%%%%%     END OF PLOTS       %%%%%%%%%%%%%%%%%%%%%%%%%%%
```

Figure A23: Plotting Various Route and Rider Characteristics

References

[1] "Deadliest States For Cyclists: Per Capita Fatality Rates," Governing, [Online]. Available: http://www.governing.com/gov-data/transportation-infrastructure/most-bicycle-cyclist-deaths-per-capita-by-state-data.html. [Accessed 29 November 2018].

[2] "The Chad William Young Foundation," The Chad William Young Foundation, 2017. [Online]. Available: https://cwyf.org/. [Accessed 21 Novemeber 2018].

[3] Weislo, Laura. “Riders Struggle With News of Chad Young’s Critical Injuries”. CyclingNews. 28 April, 2017. [Online.] Available: http://www.cyclingnews.com/news/riders-struggle-with-news-of-chad-youngs-critical-injuries/ [Accessed 14 March 2018]3

[4] *2017 USA Cycling Rulebook*. Colorado Springs, CO: USA Cycling, Inc., 2017.

[5] “UCI - Union Cycliste Internationale,” *Inside UCI - Rules and Regulations*, 05-Mar-2018. [Online]. Available: http://www.uci.ch/inside-uci/rules-and-regulations/. [Accessed: 15-Mar-2018].

[6] "31st Annual Tour of the Gila UCI Men April 19 Through April 23, 2017," 2017.

[7] Pedestrian and Bicycle Information Center, "Pedestrian and Bicyclist Crash Statistics," US Department of Transportation: Federal Highway Administration, 2015. [Online]. Available: http://www.pedbikeinfo.org/data/factsheet_crash.cfm. [Accessed 20 March 2018].

[8] "Gorgeous Washington - Ride," Strava, 2018. [Online]. Available: https://www.Strava.com/activities/379362833/analysis/1405/2617. [Accessed 20 March 2018].

[9] W. L. Briggs, L. Cochran, B. Gillett, and E. P. Schulz, *Calculus*. New York, NY: Pearson, 2011.

[10] R. C. Hibbeler, P. Schiavone, G. Emmert, and R. C. Hibbeler, *Engineering Mechanics: Dynamics*, Second Custom Edition for Colorado School of Mines. Upper Saddle River, NJ: Prentice Hall, 2016.

[11] P. Hadley, “Interpolation of data sets,” *Crystal structure*. [Online]. Available: http://lampx.tugraz.at/~hadley/num/ch3/3.php. [Accessed: 30-Sep-2018].