Terrain Roughness Classification for Off-Road Autonomous Vehicles

By

Gabriela Gresenz

Thesis

Submitted to the Faculty of the

Graduate School of Vanderbilt University

in partial fulfillment of the requirements

for the degree of

MASTER OF SCIENCE

in

Computer Science

May 14, 2021

Nashville, Tennessee

Approved:

Professor Christopher J. White

Professor Douglas C. Schmidt

ACKNOWLEDGMENTS

First and foremost, I would like to thank Dr. Jules White, my thesis advisor. Dr. White taught me how to thoroughly investigate a challenging research problem, guided me throughout the project, and has been an incredible mentor. I would also like to thank Dr. Douglas Schmidt for being on my thesis committee. I would like to thank the team of undergradate researchers who assisted with this project: Jiachen Xu, Shiliang Tian, and Acar Ary. Finally, I would like to thank my family for their love and support throughout the entirety of this project.

TABLE OF CONTENTS

Page

ACKNOWLEDGMENTS ii

LIST OF TABLES vi

LIST OF FIGURES viii

Chapter

1 Introduction 1

2 Research challenges 6

2.1 Challenge 1: Obtaining off-road terrain roughness data at scale 6

2.2 Challenge 2: Images may not be sufficient for learning tasks 6

2.3 Challenge 3: Labeling images with a single quantitative measure of roughness derived from the IMU 7

2.4 Challenge 4: Training deep learning models to effectively predict terrain roughness 7

3 Experimentation approach 10

3.1 Data collection 12

3.2 Data processing 13

3.2.1 Sensor data 13

3.2.2 Data Processing Pipeline 15

3.2.3 Video data 20

4 Research Question 1: What metric should be used to label images with a measure of terrain roughness? 21

4.1 Determining original groups 25

4.2 Labeling the images 28

4.3 Calculating image labels 29

5 Research Question 2: How do we select and filter images in our dataset? 32
5.1 Overview 32
5.2 Sensor validation 32
5.3 Visual validation 33
5.4 Results 36
6 Research Question 3: Which of the proposed labeling schemas can the roughness classifiers most effectively learn? 37
6.1 Overview 37
6.2 Training and testing split 38
6.3 Balanced classes 39
6.4 Models 41
6.5 Results 41
6.6 Evaluation on the test set 44
7 Research Question 4: How can we ensure the roughness classifiers are learning? . 45
7.1 Overview 45
7.2 Balanced classes 46
7.3 Models 46
7.4 Results 47
8 Research Question 5: Can an attention region around the upcoming drivable terrain assist the classifiers in predicting terrain roughness? 49
8.1 Overview 49
8.2 Method 49
8.3 Segmentation network 50
8.4 Preparing the dataset 51
8.5 Training and testing splits 53
8.6 Balanced classes 53
8.7 Models 54

8.8 Results 55
9 Future work 56
10 Concluding remarks & lessons learned 58
BIBLIOGRAPHY 61

LIST OF TABLES

Table Page

3.1 Example of raw CSV generated by the FitCSVTool 13
3.2 Example of calibrated raw CSV generated by the Three D Sensor Adjustment Plugin 14
3.3 Example raw CSV rows showing where the “distance” field of “record” data is in different columns 15
3.4 Example CSV rows generated by applying Step 2 of the Data Processing Pipeline to Table 3.3 16
3.5 Example rows of an accelerometer_calibrated.csv 16
3.6 Examples rows generated by applying Step 3 of the Data Processing Pipeline to Table 3.5 17

4.1 Discrete roughness metric groups 25

5.1 Images in each category for visual validation, predicted by the classifier and manually validated 35
5.2 Valid images for Labels 1 - 4 and Labels 5 - 8 after sensor and visual validation 36

6.1 Composition by class for original groups (Labels 1 and 5) 40
6.2 Composition by class for k = 2 groups (Labels 2 and 6) 40
6.3 Composition by class for k = 3 groups (Labels 3 and 7) 40
6.4 Composition by class for k = 4 groups (Labels 4 and 8) 40
6.5 Performance of Models 1 - 8 on the selection set 42
6.6 Comparing methods for sampling terrain based on selection set performance 43
6.7 Performance of Model 6 and Model 8 on the held out test set 44

7.1 Performance of Models 6, 8, 9, and 10 on their respective test sets 47

8.1 Roughness classifiers trained in the attention region experiments 50

8.2 Image scores for segmentation validation 53

8.3 Images in the dataset after removing images without a valid attention region . 53

8.4 Composition of training and validation sets prior to balancing classes for attention region networks 54

8.5 Composition of training and validation sets after balancing classes for attention region networks 54

8.6 Performance of Models 11 original, dark, and light and Models 12 original, dark, and light on the test set 55

LIST OF FIGURES

Figure Page

2.1 Sample batch of images labeled with Label 6 8

2.2 Sample batch of images labeled with Label 8 9

3.1 Roadmap 11

3.2 Step 1 of the Data Processing Pipeline 18

3.3 Step 2 of the Data Processing Pipeline 18

3.4 Step 3 of the Data Processing Pipeline 18

3.5 Step 4 of the Data Processing Pipeline 19

3.6 Final directory structure once the Data Processing Pipeline is complete . . . 19

4.1 Roughness metric with the center and ahead window plotted with z-axis acceleration 24

4.2 Visualization of center and ahead windows with different methods for discretizing the roughness metric 26

4.3 Visualization of continuous roughness metric with center window and corresponding z-axis acceleration readings 27

4.4 Visualization of original groups and center window for two 1-second ranges of z-axis acceleration readings 28

8.1 Example image, dark attention region, light attention region 49

8.2 Example images, images with correct labels, images with overlays (labels predicted by the segmentation network) 52

8.3 Example overlay images with scores -2 to 2 53

Chapter 1

Introduction

The ability for autonomous vehicles to drive safely in off-road settings is important for achieving increasing levels of autonomy in passenger-carrying autonomous vehicles. A major goal in the passenger-carrying autonomous vehicle industry has been to achieve "Level 4" autonomy, which describes vehicles that can operate autonomously in a specific set of predefined conditions [1]– such as along urban roads during clear weather conditions in a limited radius. However, a larger goal of autonomous driving is to achieve "Level 5" autonomy, which describes vehicles that can operate safely in all settings without human supervision [1]. This means that if the vehicle ends up along a route with unmarked or unpaved terrain, the vehicle should be able to safely traverse it with complete autonomy.

It is also important that Autonomous Ground Vehicles (AGV's) can safely handle the wide range of terrain they encounter in the various off-road settings in which they operate. An AGV is a type of autonomous vehicle which is designed to carry out specific tasks [2] in areas including planetary exploration, search and rescue, and mining [3, 4]. These types of situations are often without human supervision, so it is essential that these vehicles can handle with full autonomy the wide variety of terrain they encounter. AGV's must avoid harmful terrain, such as fine sand and rocks [3], and optimize their driving with respect to the upcoming drivable terrain (for example, slowing down for a rough patch).

Terrain processing is using one or more sensors to determine important information about terrain, such as terrain type, roughness, or drivability. These sensors can either be described as contact or non-contact sensors [5]. Non-contact sensors like visual and LiDAR imagery allow the vehicle to see what is coming up ahead. Visual sensors like monocular and omnidirectional cameras allow the vehicle to see the colors of the surrounding terrain [6], while LiDAR sensors allow the vehicle to effectively perceive the depth of objects in

its environment [7]. Contact sensors, such as the Inertial Measurement Unit (IMU), give a more robust look into the effect of the current terrain on the vehicle. The IMU is typically equipped with an accelerometer, gyroscope, and magnetometer [8]. The accelerometer contains information about the vehicle's acceleration along the x, y, and z planes. The z plane is of particular interest for problems of terrain processing because a vehicle's up-and-down motion as it traverses terrain can be an indicator of terrain roughness.

There has been a significant amount of work to classify terrain into discrete groups of qualitative terrain type. Kurup et al. [3] used a combination of visual and IMU data to classify patches of terrain already traversed by the vehicle into three classes: grass, gravel, and fine sand. Weiss et al. [9] also used a combination of visual and vibration data to classify patches of terrain into 14 classes, marking each as either "safe" or "unsafe". Instead of considering both visual and IMU data at once, as was the approach by Kurup et al. [3], Weiss et al. [9] initially classified upcoming terrain with visual imagery and later traversed the terrain in the image, obtaining IMU data used to verify this classification. Iwashita et al. [10] used visual and thermal imagery to classify pixels of the upcoming terrain into one of seven qualitative classes including sand, soil, and rocky terrain. Bai et al. [5] used IMU vibration data in the x, y, and z directions to classify patches of already traversed terrain into five classes (grass, sand, gravel, concrete, and mixed). Brooks and Iagnemma [11] also used strictly IMU data to perform two terrain classification experiments. In the first experiment, they classified terrain into four classes (sand, gravel, Mars-1, and unknown), and in the second experiment, they classified terrain into another set of four classes (sand, gravel, packed dirt, unknown). Weiss et al. [12] also used strictly IMU data to perform two classification experiments. The first experiment classified terrain into three classes (gravel, grass, boule court), and the second experiment expanded classification to seven classes (the three classes from the first experiment along with indoor floor, paving, asphalt, and no motion).

There has also been work in using semantic segmentation to understand terrain rough-

ness. Dahlkamp et al. [13] used visual and LiDAR imagery to create drivability maps of the upcoming terrain. The visual map classified each pixel of the upcoming terrain as drivable or unknown. The presence of unknown pixels in the visual map 40 meters ahead of the vehicle was used as an indicator to slow the vehicle down. When the vehicle reached the unknown terrain, it would safely traverse the terrain using information from the laser map (which classified pixels as drivable, unknown, or obstacle). This method was used in their robot Stanley [14], which won the DARPA Grand Challenge in 2005 [15]. Stavens and Thrun [16] used LiDAR imagery to classify the pixels composing the upcoming terrain as either "smooth" or "rugged." This binary classification was used to indicate areas of terrain where the vehicle should slow down or avoid altogether. Ram [17] experimented with custom roughness metrics to provide a more specific indication of upcoming terrain roughness using semantic segmentation. They used strictly image data as an input to predict these custom roughness metrics. Suryamurthy et al. [18] created a unified method for both terrain segmentation and roughness prediction. They performed terrain segmentation to classify each pixel of a camera image based on what kind of object or surface it belonged to (such as rock, grass, or ground), and used the bottom layers of the trained segmentation network to predict a continuous measure of roughness for each pixel in the image.

Past research has examined terrain processing through classification of qualitative terrain type and semantic segmentation for understanding roughness. However, to the best of our knowledge this research presents a novel way to examine terrain roughness prediction. We classify monocular images of off-road terrain by a single quantitative measure of roughness describing the vehicle's future kinetics as it traverses the upcoming visible terrain in the image. This research examines upcoming terrain roughness as an image classification problem, where a convolutional neural network is used to predict a custom roughness metric for labeling image frames taken by a single, monocular camera attached to the vehicle. This roughness metric is derived from the vehicle's IMU z-axis acceleration readings in order to provide a measure of terrain roughness in terms of how it will

directly affect the vehicle's motion. While past research in qualitative terrain classification [3, 5, 9, 10, 11, 12] aims to predict a visual label from the input data, our research poses a different kind of classification problem, instead predicting a measure of the vehicle's upcoming kinetic motion. While work in semantic segmentation for understanding roughness [13, 16, 17, 18] predicts roughness at the pixel level, this research predicts roughness at the scene level, learning from the image as a whole to predict upcoming terrain roughness as experienced by the vehicle itself.

Our classification problem thus examines a mapping from the outputs of the long-range camera sensor to the future outputs of the short-range IMU sensor. Nava et al. [19] propose a method to predict the future outputs of a short-range sensor from the current outputs of a long-range sensor which labels images (the output of their long-range camera sensor) with the outputs of the robot's short-range object detection sensors at multiple discrete intervals ahead of the timestamp of the image frame. The research in our study, however, uses methods of terrain sampling to label the outputs of the long-range sensor, in this case the images from the vehicle's camera, with a single comprehensive measure of the future outputs of the short-range sensor, in this case the IMU.

Another key differentiator of this work is that we aim to understand terrain roughness using nothing but images from a single, monocular camera as input. While many challenging problems in the autonomous vehicle space can be approached using expensive vehicles with more sophisticated sensor set-ups, including sensors such as LiDAR and omnidirectional cameras, our work examines terrain roughness prediction using a low-cost, single camera set-up for capturing monocular images for terrain learning.

This research examines how we can label monocular images with a measure of roughness derived from the IMU z-axis acceleration readings corresponding to the visible terrain in the images, experimenting with the derivation of a custom roughness metric and how it can be applied to images where the upcoming drivable terrain is clearly visible. We examine whether deep learning models can effectively predict the proposed labeling schemas,

and whether the use of an attention region around the upcoming drivable terrain can assist the roughness classification networks in predicting terrain roughness.

Chapter 2

Research challenges

There are a number of challenges involved in learning about off-road terrain roughness. This section discusses the challenges of obtaining data at scale, filtering data which may be insufficient for learning tasks, determining a measure of roughness with which to label images, and training deep learning models to learn these proposed measures of roughness.

2.1 Challenge 1: Obtaining off-road terrain roughness data at scale

Data collection at scale for roadway autonomous vehicles is a relatively feasible task. There exists a vast network of roads on which humans driving vehicles equipped with sensors such as cameras and LiDAR can travel to collect data. According to the Federal Highway Administration, there are almost 4.19 million miles of road in the US alone [20]. A driver could theoretically collect data for days without ever driving the same stretch of roadway twice.

Data collection at scale is difficult in an off-road setting for a number of reasons. First, off-road terrain may be undrivable by certain types of vehicles, greatly limiting the number of available vehicles that are equipped to collect off-road data. Second, there does not exist the same vast network of off-road drivable terrain as is available for roadway data collection, making data collection at scale difficult for off-road autonomous vehicles.

2.2 Challenge 2: Images may not be sufficient for learning tasks

Images where the upcoming drivable terrain is not clearly visible are insufficient for tasks of terrain learning. Collecting image data in off-road terrain settings can result in many such images. Traversing particularly rough terrain can lead to variance in the position and angle of the camera attached to the vehicle, resulting in images that are blurry or show

only trees and no visible terrain. Further, because trees surround most of the drivable terrain, images are susceptible to particularly poor lighting. For example, a cloudy day may mean that not enough light can pass through the trees, resulting in image lighting too dark to make out the drivable terrain. Additionally, intense rays of sunlight entering from openings in the trees may obstruct the image view entirely.

2.3 Challenge 3: Labeling images with a single quantitative measure of roughness derived from the IMU

Labeling images of upcoming drivable terrain with a single, comprehensive, quantitative roughness metric derived from the IMU z-axis acceleration readings is a challenging task on many levels. Creating a measure of roughness representing the entirety of the upcoming terrain is difficult because the exact length of terrain visible may be unknown. It may be challenging to determine exactly how far ahead the image sees, thus making it difficult to determine the z-axis acceleration readings corresponding to the upcoming drivable terrain in the image.

Validating that a given roughness labeling schema effectively labels the images in the dataset is difficult because while certain objects such as sticks and rocks may generally correspond to increased terrain roughness, the future z-axis acceleration experienced by the vehicle as a result of these objects may be unknown. For example, many sticks close together may cause increased terrain roughness compared to the same amount of sticks spread farther apart. Additionally, terrain that appears to be smooth may actually be very rough when traversed by the vehicle.

2.4 Challenge 4: Training deep learning models to effectively predict terrain roughness

As previously mentioned, the roughness label corresponding to a given image may not be intuitive to humans. This may also make it difficult for deep learning models to learn the proposed roughness labels. In traditional image classification tasks, such as given in

the ImageNet dataset [21], humans can easily validate images with labels such as “dog” or “cat”. However, determining a roughness label based on how the vehicle’s motion will be affected by what is shown in the image may be much more difficult. For example, a human may be able to tell that an image appears rough, but may have difficulty telling apart images assigned quantitative roughness labels of “2” and “3”. Figures 2.1 and 2.2 show images labeled with two of the proposed labeling schemas in this study (where the labels in Figure 2.1 range from 0 to 1 and the labels in Figure 2.2 range from 0 to 3, and a greater quantitative roughness label corresponds to increased terrain roughness). A human given these images without the labels would have great difficulty determining the quantitative labels corresponding to each image.

Figure 2.1: Sample batch of images labeled with Label 6

In addition to the upcoming terrain, images show a number of other intricate objects such as large trees and leaves. This may make it difficult for a roughness classifier to “focus” on the upcoming drivable terrain in an image and thus make it challenging to learn the roughness label.

Figure 2.2: Sample batch of images labeled with Label 8

Chapter 3

Experimentation approach

This research aims to address the challenges in this space. Through a number of detailed experiments, we answer the following key research questions:

1. Research Question 1: What metric should be used to label images with a measure of terrain roughness?
2. Research Question 2: How do we select and filter images in our dataset?
3. Research Question 3: Which of the proposed labeling schemas can the roughness classifiers most effectively learn?
4. Research Question 4: How can we ensure the roughness classifiers are learning?
5. Research Question 5: Can an attention region around the upcoming drivable terrain assist the classifiers in predicting terrain roughness?

Figure 3.1 details an overview of this paper. We begin by detailing our process for collecting and processing off-road terrain data. In Chapter 4, we derive eight different potential labeling schemas for labeling images of drivable terrain with a measure of upcoming terrain roughness. In Chapter 5, we filter out images insufficient for this learning task based on the sensor data needed to calculate labels for each image and certain visual criteria. In Chapter 6, we evaluate each of the eight potential labeling schemas and select the two which were most effectively learned, moving forward with these two labeling schemas for the experiments in Chapter 7 and Chapter 8. In Chapter 7, we evaluate whether the models are learning terrain roughness by evaluating these two labeling schemas on an alternative training and testing split. Chapter 7 contains our proposed models for the two selected

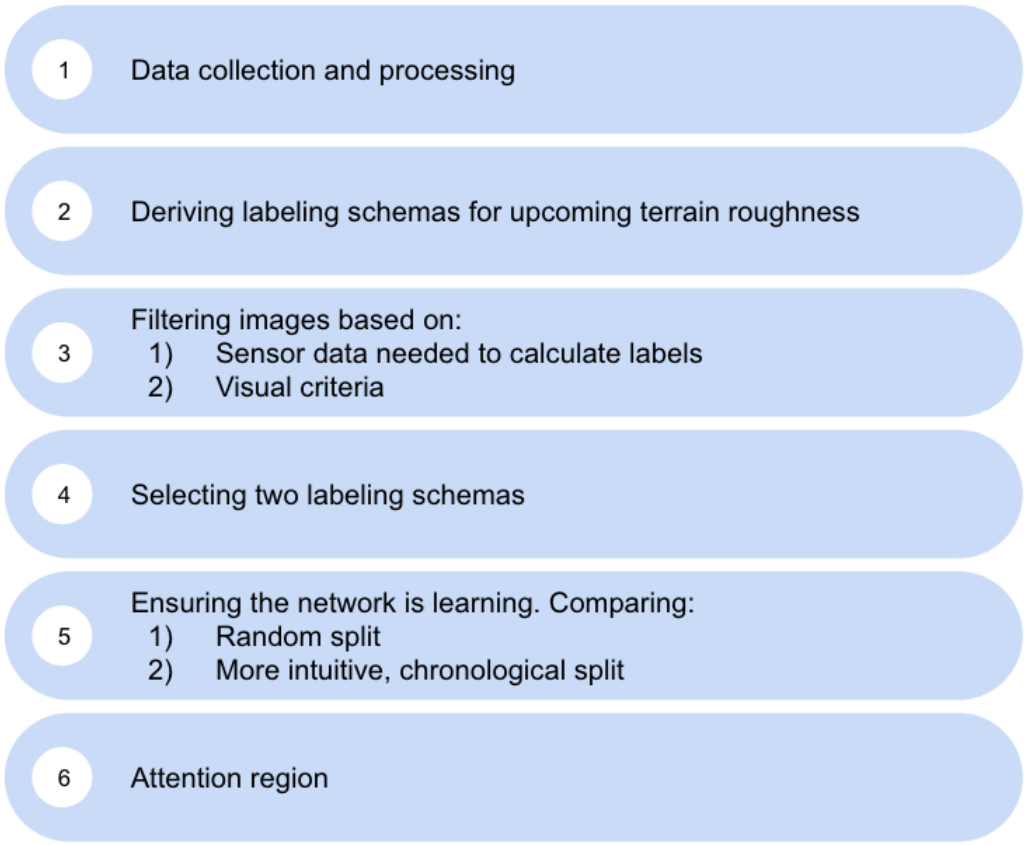

Figure 3.1: Roadmap

labeling schemas. Finally, in Chapter 8, we investigate whether an attention region around the drivable terrain can assist the networks in predicting terrain roughness.

The remainder of this section details our methods for collecting off-road terrain data at scale and processing it to allow for efficient use.

3.1 Data collection

A mountain bike was used to collect the data. The bike was equipped with the following sensors:

1. Garmin 830 dual GPS receivers.
2. Garmin Virb Ultra dual-high resolution IMU's.
3. Garmin Virb Ultra 4k 30fps camera time synchronized to both accelerometers.
4. Quarq Dzero XX1 Dub Power Sensor, a strain-gauge based power meter.
5. Garmin Bike Speed Sensor 2, a wheel rotation speed sensor.

Our experiment considered two forms of data: sensor data and video data. The video data was obtained from the camera attached to the handlebar of the mountain bike. The sensor data was obtained from the vehicle's other sensors, such as the GPS and IMU, and is in a file format called a "fit file" [22]. The sensor data is sorted into two main categories: Virb files, which are time synchronized with the camera, and Garmin files, which are not time synchronized with the camera. The Garmin files report all timestamps in UTC time, while the Virb files report all timestamps relative to the start of the data collection session. However, we can convert the timestamps of the video frames and sensor readings in the Virb files to UTC time in order to synchronize the Virb and Garmin files. The Virb files use GPS estimates of speed and distance traveled, while the Garmin files use the wheel rotation speed sensor in order to calculate speed and distance. This makes the speed and distance readings in the Garmin files more reliable.

Table 3.1: Example of raw CSV generated by the FitCSVTool

Message	**Field 1**	**Value 1**	**Units 1**	**Field 2**	**Value 2**	**Units 2**	**....**
accelerometer_data	timestamp	56	s	sample_time_offset	0|10|20|30|43|....	ms	....
gps_metadata	timestamp	62	s	enhanced_altitude	181.8	m	....

Data was collected from trails in Percy Warner Park in Nashville, TN during daytime in the late summer and beginning of fall in 2020. The data covers 43.9 miles of off-road terrain. Data was collected on five different dates, referred to as sessions: 2020-07-28, 2020-09-23, 2020-09-24, 2020-09-29, and 2020-10-02. Each session had one corresponding Virb file and one corresponding Garmin file. Videos began recording at the start of each session, but after about 25 minutes of recording the camera automatically ended one video and started another. Additionally, the user had the option to manually stop the current video and start another. Thus, multiple videos could correspond to one session. The start and end times of each video in a session are recorded in the Virb file.

3.2 Data processing

The first major component of this project was the processing of sensor and video data. This section presents a novel pipeline for processing fit files into state-based comma-separated value (CSV) files. We detail the Data Processing Pipeline for processing sensor data, then explain the algorithms and tools used for processing the video data. The data used throughout this project was processed using the steps outlined below.

3.2.1 Sensor data

Garmin provides a tool which converts fit files into CSVs called FitCSVTool [23]. This tool takes as input a fit file and outputs a single CSV containing the data read by each of the sensors, which will be referred to as a "raw CSV". Table 3.1 shows some example rows demonstrating the format of the CSV outputted by the tool. Some columns have been omitted for clarity, but the full raw CSVs are available in the project repository.

Table 3.2: Example of calibrated raw CSV generated by the Three D Sensor Adjustment Plugin

Message	Field 1	Value 1	Units 1	....	Field 3	Value 3	Units 3	....	Field 7	Value 7	Units 7
accelerometer_data	timestamp	55	s	....	accel_x	30943|....	counts	....	calibrated_accel_x	0.19238281|....	g
gyroscope_data	timestamp	55	s	....	gyro_x	32753|....	counts	....	calibrated_gyro_x	−0.0|....	deg/s

The sensors which provide three-dimensional readings (the accelerometer, gyroscope, magnetometer, and barometer) are uncalibrated, meaning that their units are not the conventionally understood units and the readings listed as the x-, y-, and z-axis readings do not correspond directly to these axes. Garmin provides a tool which calibrates these readings called the Three D Sensor Adjustment Plugin [24]. This tool takes as input a fit file and generates what will be referred to as a "calibrated raw CSV". It is in the same format as Table I, however, it contains readings only from the sensors which were calibrated by the plugin. The resulting readings in the calibrated raw CSVs are calibrated, meaning that they are converted to the conventionally understood units and the x-, y-, and z-axis readings correspond directly to these axes. For example, accelerometer readings in the raw CSVs are given in the units counts, while accelerometer readings in the calibrated raw CSVs are given in the units g (or 9.81 m/s^2). In addition to presenting the calibrated readings, the calibrated raw CSVs also retain the original, uncalibrated readings.

The Three D Sensor Adjustment Plugin calibrates all three-dimensional readings with a calibration factor provided in the fit file. These calibration factors can be seen in the raw CSVs with the message three_d_sensor_calibration. The Virb fit files in this project contained calibration factors for only the accelerometer and gyroscope, so readings from the other three-dimensional sensors were not included in the resulting calibrated raw CSV. Table 3.2 shows some example rows with the format of a calibrated raw CSV.

Both the raw CSVs and the calibrated raw CSVs contain data from a mix of sensors. Further, different readings from the same sensor may output the same field to a different column in the table depending on the data available at the time of the reading. For example, Table 3.3 shows that two readings from the "record" message contain distance readings in

Table 3.3: Example raw CSV rows showing where the "distance" field of "record" data is in different columns

Message	Field 1	Value 1	Units 1	Field 2	Value 2	Units 2	....	Field 4	Value 4	Units 4
record	timestamp	4	s	distance	0.0	m				
record	timestamp	47	s	position_lat	430494480	semicircles	....	distance	0.79	m

different columns.

In order to prepare the data into a format in which it could be used more efficiently, we developed an extensive Data Processing Pipeline. While this project only requires data containing readings from the messages "gps_metadata", "accelerometer_data", "record", "timestamp_correlation", and "camera_event", the Data Processing Pipeline prepares data from a number of other categories so that this tool can be used in the future by researchers who may need to effectively extract and format the relevant data from fit files into the more conventional format of a CSV.

3.2.2 Data Processing Pipeline

Step 1. The first step of the Data Processing Pipeline is the conversion of fit files to the raw CSVs and calibrated raw CSVs using the tools provided by Garmin [23, 24], as shown in Figure 3.2.

Step 2. The next step of the Data Processing Pipeline takes as input a folder containing two subdirectories: Original and Calibrated, containing the raw CSVs and calibrated raw CSVs for each session. A folder is created for each session, and the raw CSVs and calibrated raw CSVs are split into multiple files. gps.csv, magnetometer.csv, record.csv, accelerometer_calibrated.csv, and gyroscope_calibrated.csv contain readings only from the sensor specified in the title of the CSV. accelerometer_calibrated.csv and gyroscope_calibrated.csv contain both the original and calibrated readings for the specified sensors, as given in the calibrated raw CSVs. Additionally, each accelerometer reading given in units g is also converted to m/s^2 and included in accelerometer_calibrated.csv. Each of these 5 CSVs is formatted in a state-based representation where each unique times-

Table 3.4: Example CSV rows generated by applying Step 2 of the Data Processing Pipeline to Table 3.3

timestamp (s)	position_lat (semicircles)	position_long (semicircles)	distance (m)	enhanced_speed (m/s)	enhanced_altitude (m)
4			0.0		
47	430494480	-1036435710	0.79	0.0	191.2

Table 3.5: Example rows of an accelerometer_calibrated.csv

timestamp (s)	timestamp_ms (ms)	sample_time_offset (ms)	....	calibrated_accel_x (g)	....
55	865	0|10|20|30|40|....	....	0.19238281|0.18896484|0.19580078|0.21044922|0.23095703....	....

tamp corresponds to a single row containing all of the relevant readings at that timestamp. Rows are sorted in ascending order by timestamp. The field names and units are removed from the body of the CSVs and used as the column headers to allow for efficient data lookup. This step is shown in Figure 3.3. Consider the sample readings from the raw CSV with the record message, as shown in Table 3.3. In record.csv, these readings would be reformatted as shown in Table 3.4.

The sixth CSV generated by this step, other.csv, contains the unformatted data with messages other than the GPS, magnetometer, record, accelerometer, and gyroscope data. We decided to keep these readings in one CSV because they contain only a few readings per message, as opposed to the specified 5 sensors which can contain upwards of thousands of readings with that particular message.

Step 3. The third step of the Data Processing Pipeline further expands the state-based representation of magnetometer.csv, accelerometer_calibrated.csv, and gyroscope_calibrated.csv. Consider Table 3.5, which shows part of a sample row from an accelerometer_calibrated.csv.

The accelerometer readings occur around every 10 ms, where a single row contains 30 readings. The sample_time_offset (ms) column shows how far after the timestamp given by timestamp (s) and timestamp_ms (ms) the corresponding reading in calibrated_accel_x (g) occurred. This format, which condenses 30 readings into a single row, is used in magnetometer.csv, accelerometer_calibrated.csv, and gyroscope_calibrated.csv. This step takes as input magnetometer.csv, accelerome-

ter_calibrated.csv, and gyroscope_calibrated.csv, and creates magnetometer_split.csv, accelerometer_calibrated_split.csv, and gyroscope_calibrated_split.csv, which have unique rows for each reading. This step is shown in Figure 3.4. Table 3.6 shows a sample from an accelerometer_calibrated_split.csv generated from the sample in Table 3.5.

Table 3.6: Examples rows generated by applying Step 3 of the Data Processing Pipeline to Table 3.5

timestamp (s)	**timestamp_ms (ms)**	**....**	**calibrated_accel_x (g)**
55	865	....	0.19238281
55	875		0.18896484
55	885		0.19580078
55	895		0.21044922
55	905		0.23095703
55	915		0.24804688

Step 4. The timestamps listed in each of the Virb files are system timestamps, or the time elapsed since the start of the data collection session. The video data from a given session is time synchronized to these system timestamps, where other.csv contains the system timestamp at which the corresponding videos begin and end. However, there are many benefits to having the UTC timestamp readily available for each of these readings. For example, the corresponding Garmin files contain only UTC timestamps. Additionally, the system timestamps of different data collection sessions correspond to different UTC timestamps. The UTC timestamp allows for a standardized timestamp across all data. For each session, other.csv contains a timestamp_correlation message, which shows the UTC timestamp corresponding to a single system timestamp. This information can be used to obtain the UTC timestamps corresponding to each system timestamp in all of the generated CSVs. This step is shown in Figure 3.5.

Final output. Figure 3.6 shows the final directory structure once the Data Processing Pipeline is complete.

This pipeline was originally created to process the Virb files. However, it also works to process the Garmin files. The Garmin files, however, contain barometer readings which are not present in the Virb files. Thus, there are a large number of barometer readings present in the Garmin other.csv files. Future users may include or exclude different sensors from

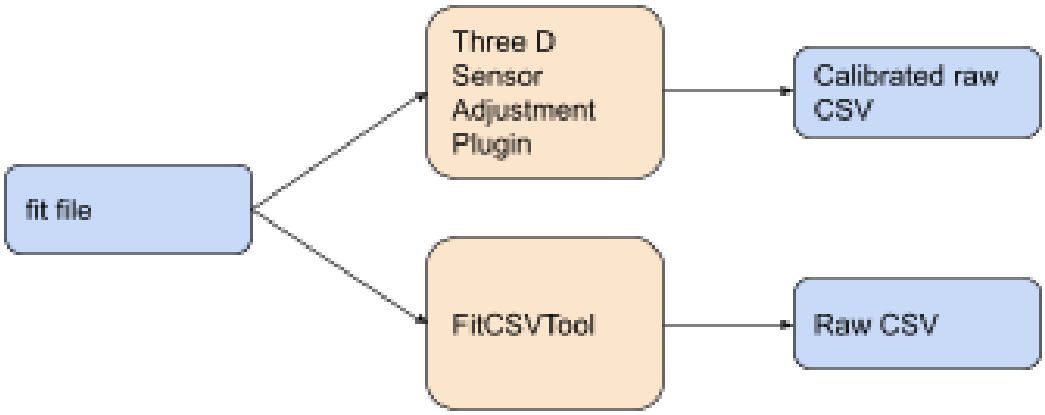

Figure 3.2: Step 1 of the Data Processing Pipeline

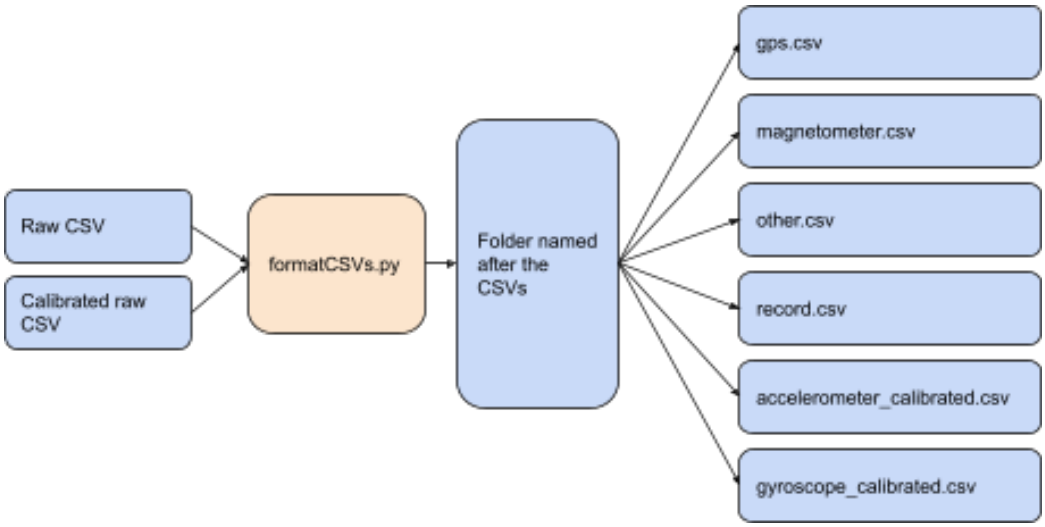

Figure 3.3: Step 2 of the Data Processing Pipeline

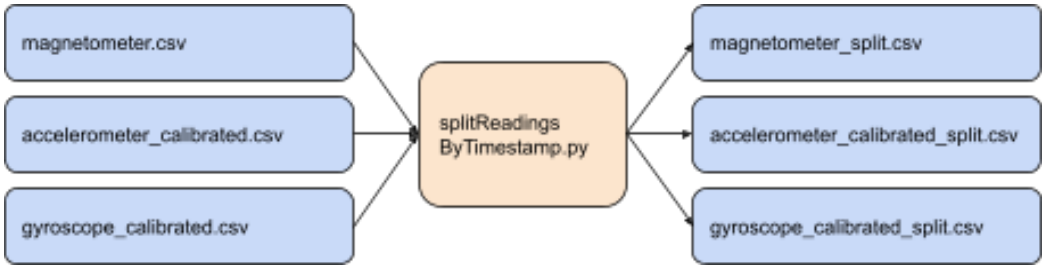

Figure 3.4: Step 3 of the Data Processing Pipeline

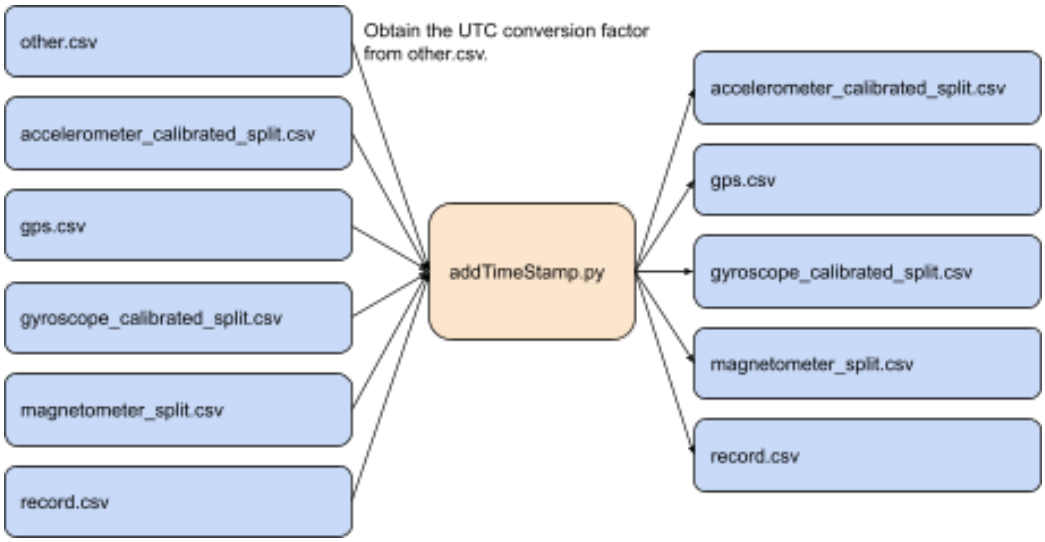

Figure 3.5: Step 4 of the Data Processing Pipeline

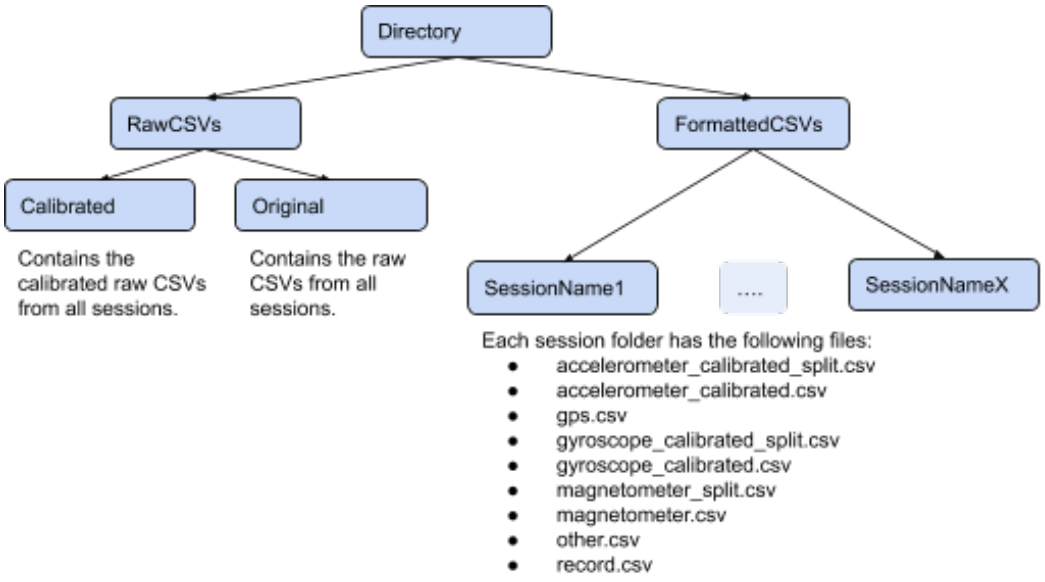

Figure 3.6: Final directory structure once the Data Processing Pipeline is complete

those processed by the Data Processing Pipeline. The code can be easily modified in the aforementioned scripts in order to account for different sensors.

3.2.3 Video data

Once the Data Processing Pipeline has been run for the sensor data, there are also tools for processing the video data. Since the video frames in this project were used as the inputs for our models, we wanted to extract as many video frames as possible while minimizing the overlap between frames. For this reason, we chose to extract video frames 1 second apart, beginning at the start of each video.

The frame rate of our camera was around 29.97 frames per second, so obtaining frames exactly 1 second apart was not possible. Instead, we find the frame most closely corresponding to each target timestamp using the following equation, where X is the index of the target frame, Y is the number of milliseconds ahead of the start of the video corresponding to the target frame, and Z is the number of milliseconds between frames. Z can be calculated by dividing 1 second (or 1000 milliseconds) by the frame rate, F.

$$Xframe = \frac{Yms}{Zms/frame}$$
$$Zms/frame = 1000/Ffps$$

For each target timestamp in the video, we find the frame most closely corresponding to that timestamp. The resulting image frame is named by its UTC timestamp in seconds and milliseconds (for example, "1000s100ms"). In total, we extracted 12,982 images.

The scripts for processing the video data are located in the project repository. First, videoStartTimes.py determines the UTC start time of each video and creates a CSV with this information. These start times must then be inputted to convertVideos.py, which performs the above algorithm on each of the videos and saves the image frames according to their UTC timestamps. A folder is created for each video with the images extracted from that video.

Chapter 4

Research Question 1: What metric should be used to label images with a measure of terrain roughness?

The first major question of this project was how to label images with a single, comprehensive measure of terrain roughness derived from the IMU z-axis acceleration readings corresponding to the visible terrain in each image. This section first explores potential methods for labeling the images, including the derivation of a custom roughness metric, and then details the equations used in calculating the resulting labels.

Z-axis acceleration, a measure of the vehicle's vertical acceleration, has been used in many studies involving terrain roughness [3, 12, 16, 17]. By labeling upcoming terrain with a measure of z-axis acceleration, we not only gauge terrain roughness but how the upcoming terrain will directly affect the vehicle's movement.

A few different methods have been used to extract measures of roughness from the IMU z-axis acceleration data. Ram [17] found that labeling sections of camera images with a roughness metric obtained from IMU data produced more desirable results than using pixel-wise roughness labels derived from point-to-plane roughness. Their IMU roughness metric grouped images into 4 discrete classes based on the single IMU reading at each pixel. Weiss et al. [12] derived custom roughness metrics from the raw IMU z-axis vibration data and used these metrics as inputs for learning. Kurup et al. [3] performed feature extraction from the raw IMU z-axis vibration data and used the extracted features as inputs for learning. Stavens et al. [16] used a roughness metric called the ruggedness coefficient which provides a measure of roughness that is standardized with respect to speed. As noted by Ram [17], one of the major downfalls of their roughness metric obtained from the IMU z-axis vibration data was that it did not account for the vehicle's velocity. Our initial thought was to account for speed in a similar way, however, we did not find this same linear

relationship between z-axis acceleration and speed in our data. This is likely because the vehicle operated at significantly lower speeds than the vehicles used by Stavens et al. [16] and Ram [17].

In order to gain a single measure of roughness indicative of the upcoming terrain as a whole, multiple z-axis acceleration readings should be used as opposed to the readings at a single timestamp, which represents one single position on the upcoming terrain. Thus, we needed to determine 1) which z-axis acceleration readings should be factored into the roughness metric at a given timestamp, and 2) what single measurement of roughness should be derived from this group of z-axis acceleration readings.

The roughness metric was derived from a 1 second window of z-axis acceleration readings. These readings were taken about 10 ms apart, so this window included roughly 100 z-axis acceleration readings. We had originally considered using a window of 100 ms, however, this only included 10 z-axis acceleration readings which not only accounted for a smaller stretch of terrain, but would be more susceptible to outliers in calculating a single measure of roughness. We then calculated the standard deviation of these roughly 100 z-axis acceleration readings. This measure takes into account each value present in the 1 second sample to provide a metric describing the entirety of the terrain included in the sample. The standard deviation also accounts for cases where the mean of the z-axis acceleration readings in the 1 second window is not 0. For example, if the vehicle is moving increasingly quickly down a hill at a constant high acceleration, the standard deviation of readings in this window will remain low although the z-axis acceleration is of high magnitude. Additionally, this metric is not significantly affected by outliers. For example, if the upcoming terrain is smooth except for a small rock, traveling over the rock will likely be reflected in only one out of the roughly 100 z-axis acceleration readings and thus will not outweigh the overall low z-axis acceleration values of the upcoming terrain. It is important to note that this 1 second range of z-axis acceleration readings could encompass anywhere from around 1 - 7 meters, depending on the speed of the vehicle (which was typ-

ically travelling between 1 - 7 m/s). However, in order to use the standard deviation as our roughness metric, we decided that it was best that each sample contain roughly the same amount of readings so that the standard deviation was calculated from a standard sample size. Had we instead sampled the terrain based on a set distance, some groups of z-axis acceleration would have contained significantly less data than others, making the standard deviation for that sample more susceptible to outliers than the standard deviation from a sample containing more readings.

The bike traveled along particularly rough terrain, causing the angle of the camera to vary between images and sometimes causing the camera to change positions. Thus, the amount of upcoming terrain and its distance from the vehicle was not constant across all images. For this reason, we decided to examine two separate terrain sampling approaches. Our initial approach was to label images with a 1 second sampling of z-axis acceleration readings centered around the timestamp corresponding to 5 meters ahead of the bike, referred to as Terrain Sampling Method 1 (TSM 1). After examining more images, we realized that the front tire of the bike was visible in a good number of images, indicating that the image showed the terrain directly ahead of the image's timestamp. For this reason, we also experimented with labeling images with a 1 second sampling of z-axis acceleration readings directly ahead of the image's timestamp, referred to as Terrain Sampling Method 2 (TSM 2). It is important to note that the first approach derives the roughness metric from a 1 second window centered around a timestamp, meaning that the z-axis acceleration readings were sampled from 500 ms before and after the timestamp, while the second approach derives the roughness metric from a 1 second window ahead of a timestamp, meaning that the z-axis acceleration readings were sampled from 1000 ms after the timestamp. For each timestamp in the accelerometer readings, we found both the standard deviation of a 1 second window of z-axis acceleration readings centered around the timestamp, referred to as the "center window", and the standard deviation of a 1 second window of z-axis acceleration readings ahead of the timestamp, referred to as the "ahead window". Figure 4.1 shows

the roughness metrics derived from the center and ahead windows plotted with z-axis acceleration. Then to label each image with TSM 1, we found the center window value at the timestamp most closely corresponding to 5 meters ahead of the image, and to label each image with TSM 2, we found the ahead window value at the timestamp most closely corresponding to the image's timestamp. The calculation of the center window left out the final z-axis acceleration value included in the window resulting in a 0.99 second sampling instead of a 1 second sampling.

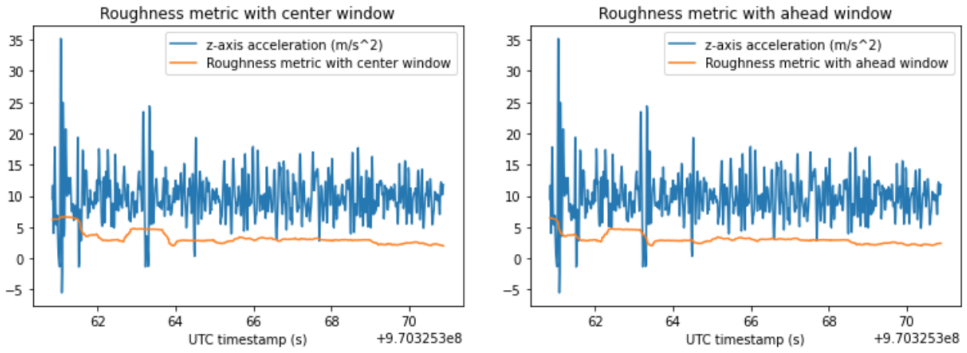

Figure 4.1: Roughness metric with the center and ahead window plotted with z-axis acceleration

The last step was to discretize the roughness metric. The standard deviation of a 1 second sampling of z-axis acceleration values is a continuous value, however our goal was to perform image classification so we needed to make this value discrete. We examined four methods for discretizing the roughness metric: using data visualization to determine 4 discrete groups, using k-means clustering with k = 2, using k-means clustering with k = 3, and using k-means clustering with k = 4. These will be referred to as original groups, k = 2 groups, k = 3 groups, and k = 4 groups, respectively.

The original groups were determined through data visualization on the roughness metric derived from a 1 second sampling centered around each timestamp, since our initial labeling approach was to label images using a 1 second sampling centered around the timestamp corresponding to 5 meters ahead of the image (TSM 1). These same groups were also used

Table 4.1: Discrete roughness metric groups

	Original groups	**k = 2 groups**	**k = 3 groups**	**k = 4 groups**
Center window	Group 0: 0 - 2.5 Group 1: 2.5 - 5.0 Group 2: 5.0 - 7.5 Group 3: 7.5 +	Group 0: 0 - 6.42 Group 1: 6.42 +	Group 0: 0 - 4.45 Group 1: 4.45 - 9.05 Group 2: 9.05 +	Group 0: 0 - 3.59 Group 1: 3.59 - 6.65 Group 2: 6.65 - 11.02 Group 3: 11.02 +
Ahead window	Group 0: 0 - 2.5 Group 1: 2.5 - 5.0 Group 2: 5.0 - 7.5 Group 3: 7.5 +	Group 0: 0 - 6.47 Group 1: 6.47 +	Group 0: 0 - 4.44 Group 1: 4.44 - 9.01 Group 2: 9.01 +	Group 0: 0 - 3.65 Group 1: 3.65 - 6.75 Group 2: 6.75 - 11.11 Group 3: 11.11 +

for labeling the image with a 1 second sampling directly ahead of the image's timestamp (TSM 2) in order to allow for comparison. The k-means groups were performed separately for the roughness metric with the center window and the roughness metric with the ahead window. The resulting groups are shown in Table 4.1. Figure 4.2 shows the center and ahead windows with the different methods for discretizing the roughness metric.

We also performed the Jenks Natural Breaks algorithm, a clustering algorithm for 1-dimensional data, on the ahead window data. This resulted in almost the same splits determined by k-means clustering for k = 2, k = 3, and k = 4, so we did not consider the Jenks Natural Breaks splits as another method for discretizing the roughness metric.

4.1 Determining original groups

The original groups were determined with data visualization on the center window data. We began by plotting scatterplots showing z-axis acceleration vs. time. The roughness metric was added to these scatterplots using a continuous color map ranging from light to dark, where a light value represented a low roughness metric and a dark value represented a high roughness metric. This allowed us to visualize how the roughness metric varied with different patterns of z-axis acceleration readings. This can be seen in Figure 4.3.

We examined the distribution of the roughness metric, finding that 25% of the data was below 2.45, 50% of the data was below 3.82, 75% of the data was below 6.05, and the mean of the data was 4.65. Using Figure 4.3 and the distribution information, it can be seen that a roughness metric of around 2.5 was quite low. The distribution information showed that

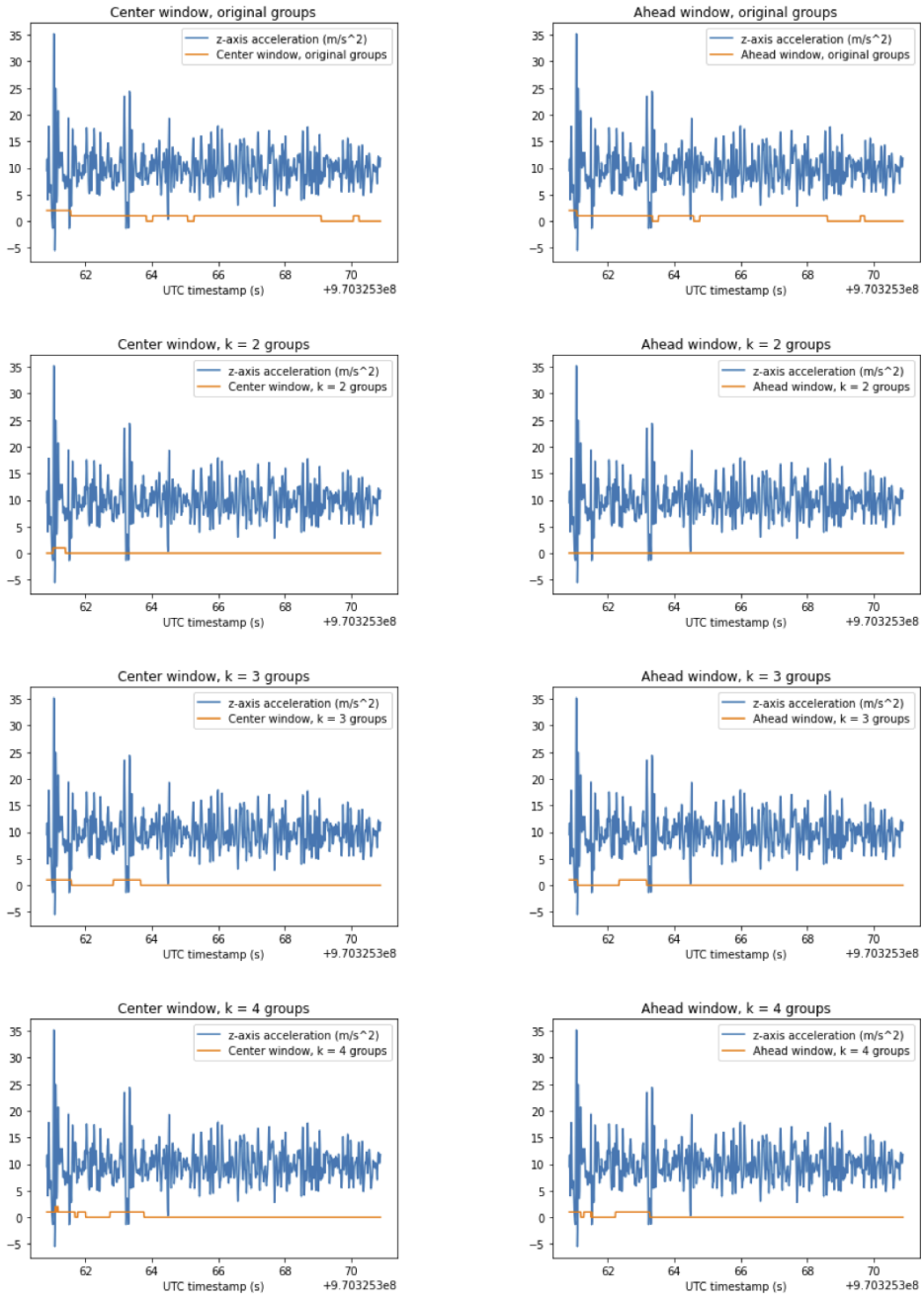

Figure 4.2: Visualization of center and ahead windows with different methods for discretizing the roughness metric

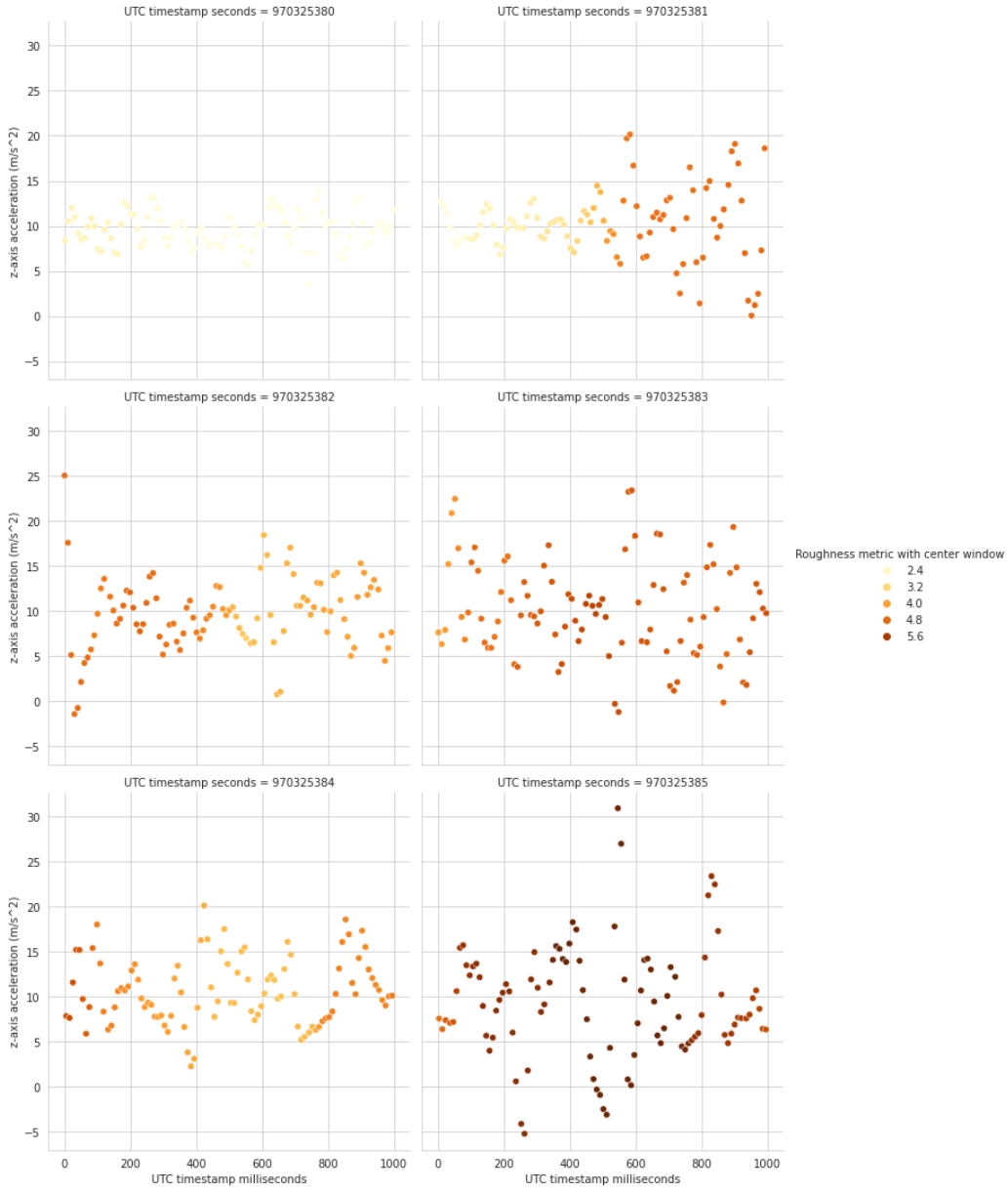

Figure 4.3: Visualization of continuous roughness metric with center window and corresponding z-axis acceleration readings

a roughness metric of 4.65 was average. To create groups of equal ranges, we discretized the roughness metric as described in Table VII using the following buckets: 0 - 2.5 (Group 0), 2.5 - 5.0 (Group 1), 5.0 - 7.5 (Group 2), and 7.5 + (Group 3). Finally, we visualized these discrete buckets using Figure 4.4, which is similar to Figure 4.3 except the scatterplot colors are no longer continuous and instead represent one of the four discrete roughness values.

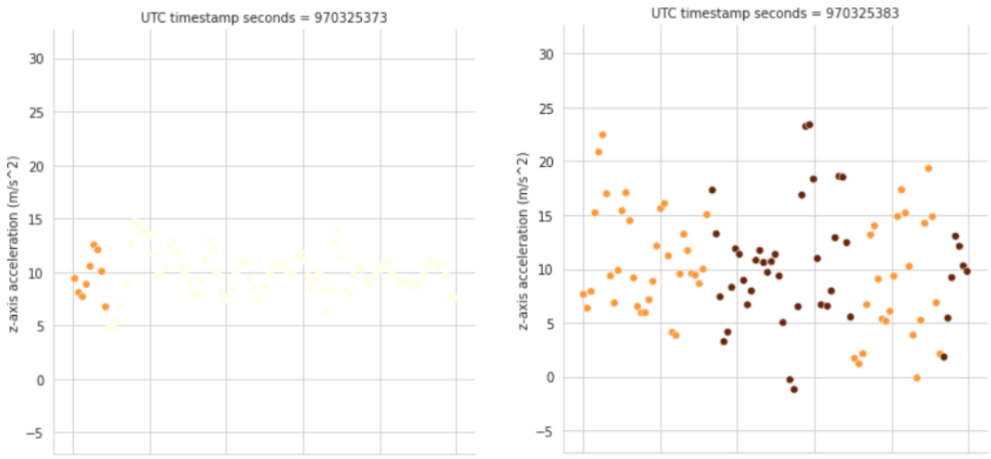

Figure 4.4: Visualization of original groups and center window for two 1-second ranges of z-axis acceleration readings

4.2 Labeling the images

Each image was assigned eight labels, one for each possible combination of the two methods of sampling the terrain (TSM 1 and TSM 2) and the four methods of discretizing the roughness metric (original groups, k = 2 groups, k = 3 groups, and k = 4 groups).

1. Label 1: TSM 1, original groups.
2. Label 2: TSM 1, k = 2 groups.
3. Label 3: TSM 1, k = 3 groups.
4. Label 4: TSM 1, k = 4 groups.

5. Label 5: TSM 2, original groups.

6. Label 6: TSM 2, k = 2 groups.

7. Label 7: TSM 2, k = 3 groups.

8. Label 8: TSM 2, k = 4 groups.

Labels 1 - 4 correspond to the first method of sampling the terrain (TSM 1), where each image is labeled with the roughness metric derived from a 1 second sampling of z-axis acceleration readings centered around the timestamp (center window) corresponding to 5 meters ahead of the vehicle. We used the vehicle's speed and the image's timestamp to calculate the timestamp corresponding to 5 meters ahead of the vehicle, which will be referred to as the target timestamp, then labeled the image with the original group, k = 2 group, k = 3 group, and k = 4 group values of the center window roughness metric most closely corresponding to the target timestamp.

Labels 5 - 8 correspond to the second method for sampling terrain (TSM 2), where each image is labeled with the roughness metric derived from a 1 second sampling of z-axis acceleration readings directly ahead of the timestamp (ahead window) corresponding to the vehicle's current position. To do this, we labeled the image with the original group, k = 2 group, k = 3 group, and k = 4 group values of the ahead window roughness metric most closely corresponding to the image's timestamp.

4.3 Calculating image labels

The following equations were used to determine the timestamp corresponding to 5 meters ahead of the image's timestamp, used for Labels 1 - 4. First, we calculate the distance d_A corresponding to the image's timestamp, t_A. Let d_1 and t_1 be the distance reading and timestamp closest to and before t_A, and let d_2 and t_2 be the distance reading and timestamp closest to and after t_A. Let s_{avg} be the average speed between t_1 and t_2, let $t_{elapsed}$ be the

time elapsed between t_1 and t_A, and let $d_{elapsed}$ be the distance elapsed between d1 and d_A. Distance readings are 1 second apart.

$$s_{avg} = d_2 - d_1, \text{ since distance readings are 1 second apart.}$$

$$t_{elapsed} = t_A - t_1$$

$$d_{elapsed} = s_{avg} * t_{elapsed}$$

$$d_A = d_1 + d_{elapsed}$$

Then, we calculate the target distance 5 meters ahead of the image, d_{target}, by adding 5 meters to d_A.

$$d_{target} = d_A + 5$$

Finally, we calculate the timestamp t_{target} corresponding to d_{target}. Let d_1 and t_1 be the distance reading and timestamp closest to and before d_{target}, and let d_2 and t_2 be the distance reading and timestamp closest to and after d_{target}. Let s_{avg} be the average speed between t_1 and $t2$. Let $t_{elapsed}$ be the time elapsed between d_1 and d_{target}.

$$s_{avg} = d_2 - d_1$$

$$t_{elapsed} = \frac{d_{target} - d_1}{s_{avg}}$$

$$t_{target} = t_1 + t_{elapsed}$$

Then, the center window values for the original groups, k = 2 groups, k = 3 groups, and k = 4 groups at the timestamp closest to t_{target} were assigned to Labels 1 - 4. The ahead window values for original groups, k = 2 groups, k = 3 groups, and k = 4 groups at the timestamp closest to the image's timestamp were assigned to Labels 5 - 8.

Initially, we planned to use the distance readings from the Virb files. However, upon performing some sample distance calculations, we realized that the distance and speed measurements in the Virb files were inconsistent with one another. This is likely due to the fact that the Virb files estimate distance and speed based on the GPS. The Garmin files, however, base their distance and speed measurements on wheel rotation. We performed some

sample distance calculations using the distance and speed measurements in the Garmin files to confirm that the distance and speed measurements in these files were consistent with one another. Thus, the distance measurements used in calculating Labels 1 - 4 were from the Garmin files. While the Garmin files are not time synchronized with the videos, we initially converted all measurements and images to UTC time, allowing us to use the Garmin files with the videos.

Chapter 5

Research Question 2: How do we select and filter images in our dataset?

5.1 Overview

The dataset consisted of 12,982 images. However, there were two stages of image selection to filter out images that should not be used during the learning process. The first stage of validation examined sensor data to confirm that 1) there was sufficient data available for performing the calculations necessary for labeling each image, and 2) if there was sufficient data available, this data met certain criteria. The second stage of validation was to visually confirm that the images used in the learning process met certain criteria, most importantly, that these images contained a visible path from which the model could learn terrain roughness.

5.2 Sensor validation

In this step, we confirmed that 1) there was sufficient data available for performing the calculations necessary for labeling each image, and 2) if there was sufficient data available, this data met certain criteria. This criteria was defined as follows:

1. All speeds should be nonzero in the entire time window used for calculating the roughness metric (the vehicle should not be stopped).

2. The distance readings and accelerometer readings should be continuous, defined as follows: In the window used for calculating the roughness metric, distance readings must be 1 second apart (crucial for performing the distance calculations), and accelerometer readings should be no more than 50 ms apart (most readings are 10 ms apart).

3. The readings from the GPS should be no more than 500 ms apart (most readings are 100 ms apart).

4. The UTC timestamp calculated from the system timestamp should be within 1 second of the UTC timestamp reported by the GPS.

It is important to note that the GPS provided UTC timestamp readings which sometimes did not match the calculated UTC timestamps. This was likely due to a lag in the sensors at certain parts of the forest. In sensor validation for Labels 1 - 4, we excluded image frames from areas where the two times did not match up in case other sensor readings were also affected during this time range. We excluded criteria 3 and 4 in calculating Labels 5 - 8 because significantly less sensor data was used in calculating these labels.

We first performed sensor validation for Labels 1 - 4, confirming that there was sufficient data available for performing the necessary calculations and that the above criteria were met. Of the 12,982 images in the dataset, only 8,982 were valid after the sensor validation step. We then performed sensor validation for Labels 5 - 8, resulting in 12,188 valid images.

5.3 Visual validation

The next step of validation was visual validation. In this step of validation, we confirmed that each image met the following criteria:

1. The image must contain a path.

2. The path must be clearly visible.

3. The image must not be obstructed by sunlight.

4. The image must not be blurry.

We began by training a simple image classifier to sort the 8,982 images valid under sensor validation for Labels 1 - 4 into one of four categories:

- Path: The image passed all validation checks.
- BeforePath: The image was taken before entering the off-road trail.
- Sunlight: The image was obstructed by sunlight.
- Trees: The image contained only trees and no path.

We trained this classifier with a dataset consisting of 460 images, distributed as follows:

- Path: 256 images.
- BeforePath: 80 images.
- Sunlight: 73 images.
- Trees: 51 images.

We used a random 20% of the training set as a validation set to determine the number of epochs to train and to prevent overfitting. This classifier achieved 97.83% accuracy on the validation set. We did not evaluate this classifier on a test set because it was not created to be a novel contribution, but rather a starting point for assisting us to sort the thousands of images in our dataset for visual validation. The reported accuracy on the validation set is included to provide an idea of how this tool assisted us in sorting the images in our dataset.

We used this classifier as a starting point to sort all 8,982 images into one of these four categories. We then manually validated the images in each of these four categories, creating a list of images that were misplaced. Upon performing this first round of validation, we noticed that there were a few additional categories that would need to be created:

- Blurry: The image was blurry.
- Cord: The cords attached to the camera and sensors obstructed the camera view.
- Dark: The image was too dark to clearly identify the path.

Table 5.1: Images in each category for visual validation, predicted by the classifier and manually validated

Category	**Predicted**	**Actual**
BeforePath	74	57
Blurry	0	662
Cord	0	62
Dark	0	277
MinimalPath	0	171
Path	8011	7070
Sunlight	154	110
Trees	743	573

- MinimalPath: The image had some path present, but it was hardly visible.

During the first round of validation, multiple individuals worked to determine which images were incorrectly classified. However, in order to account for potential subjectivity between individuals, we performed a second round of validation where one individual sorted through each of the images to confirm which category they belonged to. Table 5.1 shows the number of predicted images in each category and the number of images in each category after the second round of validation. It is important to note that the number of predicted images in Blurry, Cord, Dark, and Minimal is 0 because these categories were introduced during validation.

Evaluating the classifier's performance based on the predictions and actual labels is not an accurate indicator of the sorting classifier's performance because some images which may have been considered "Path" during training may have been later classified as "Dark" or "Blurry", for example, with the introduction of new categories. This table shows the resulting breakdown of how many images belonged to each category of the 8,982 images which were valid under sensor validation for Labels 1 - 4. The 7,070 images categorized as "Path" are such that they meet all of the criteria for visual validation.

Table 5.2: Valid images for Labels 1 - 4 and Labels 5 - 8 after sensor and visual validation

	Sensor validation	**Visual validation**
Labels 1 - 4	8,982	7,070
Labels 5 - 8 (filtered to contain only images also valid for Labels 1 - 4)	8,970	7,061

5.4 Results

We had originally considered only Labels 1 - 4 before deciding to also consider Labels 5 - 8, so we performed visual validation on all 8,982 images valid under sensor validation for Labels 1 - 4 and used the resulting 7,070 valid images for training. When we later decided to experiment with Labels 5 - 8 after noticing that the front tire of the bike was present in a good number of images, we wanted to use the same training and testing sets as were used for Labels 1 - 4 in order to compare the two methods for sampling the terrain. Thus, we filtered the 12,188 images valid under sensor validation for Labels 5 - 8 to only include images that were also valid for both sensor and visual validation for Labels 1 - 4. Table 5.2 summarizes the number of valid images after both visual and sensor validation for both methods of sampling terrain, where the valid images for Labels 5 - 8 were filtered to contain only images also valid for Labels 1 - 4. Ideally, we would have also filtered the valid images for Labels 1 - 4 to contain only images also valid for Labels 5 - 8 so that they had exactly the same training and testing sets, however we trained all models with Labels 1 - 4 before determining that we wanted to experiment with Labels 5 - 8. The resulting training set for Labels 5 - 8 contained only 9 fewer images than the training set for Labels 1 - 4, and the testing sets were exactly the same.

Chapter 6

Research Question 3: Which of the proposed labeling schemas can the roughness classifiers most effectively learn?

6.1 Overview

This first set of experiments aims to compare the eight different methods for labeling images with a measure of upcoming terrain roughness derived from the IMU z-axis acceleration readings. These eight different labels reflect two different methods for sampling the terrain and four different methods for discretizing the continuous roughness metric. The roughness metric selected was the standard deviation of a 1 second sampling of z-axis acceleration readings, taken at approximately 10 ms apart for a total of roughly 100 readings.

The following text provides a short summary of the eight different labels compared in this set of experiments, as explained in Chapter 4. The first method for sampling the terrain is selecting the 1 second sampling of z-axis acceleration readings centered around the timestamp corresponding to 5 meters ahead of the bike's position at the image's timestamp (TSM 1). The second method for sampling the terrain is selecting the 1 second sampling of z-axis acceleration readings directly after the timestamp of the image, or directly ahead of the vehicle (TSM 2). The standard deviation of the z-axis acceleration readings within these 1 second windows is a continuous variable, so we considered four methods of discretizing this value: original groups, k = 2 groups, k = 3 groups, and k = 4 groups. Together, these two methods for sampling terrain and four methods for discretizing the roughness metric resulted in a total of eight labels assigned to each image:

1. Label 1: TSM 1, original groups.
2. Label 2: TSM 1, k = 2 groups.
3. Label 3: TSM 1, k = 3 groups.

4. Label 4: TSM 1, k = 4 groups.

5. Label 5: TSM 2, original groups.

6. Label 6: TSM 2, k = 2 groups.

7. Label 7: TSM 2, k = 3 groups.

8. Label 8: TSM 2, k = 4 groups.

In this experiment, we train eight different roughness classifiers where each uses one of these eight labeling schemas.

1. Model 1: Label 1.

2. Model 2: Label 2.

3. Model 3: Label 3.

4. Model 4: Label 4.

5. Model 5: Label 5.

6. Model 6: Label 6.

7. Model 7: Label 7.

8. Model 8: Label 8.

6.2 Training and testing split

We split the data randomly as follows. 80% of the data was set aside for training and validation (referred to as the "training-validation set", split during training to create separate training and validation sets), 5% of the data was reserved for determining which labeling schemas to evaluate on the test set and use for the remainder of this project (this will be referred to as the "selection set"), and 15% of the data was reserved for the testing

set. The testing set was used for a final evaluation of the selected models, and had no influence on which labeling schemas we selected to move forward with.

In training each model, we used a random 80% of the training-validation set for training and a random 20% for validation. Because this split was performed after balancing classes for each labeling schema (see "Balanced classes" below), the specific images used for training and validation from the training-validation set differed between models. The validation set was used to determine the number of epochs to train the model and prevent overfitting.

This 80% / 5% / 15% split into training-validation, selection, and testing sets was performed on the 7,070 images which were valid under sensor and visual validation for Labels 1 - 4 and used as described in Models 1 - 4. In order to compare Models 1 - 4 with Models 5 - 8, we wanted to use the same datasets for Models 5 - 8 as were used in Models 1 - 4. Of the 7,070 images used in Models 1 - 4, 7,061 of these images were also valid for Labels 5 - 8, as described in Chapter 5. Of these 9 fewer images, all belonged to the training-validation set, so the selection and testing sets used for comparing and evaluating Models 1 - 8 were the same.

6.3 Balanced classes

The initial approach was to utilize all the data in the training-validation set. However, the classes were so skewed that our models were learning to predict almost solely the majority classes. The composition of the datasets by class for each of the eight different labeling schemas prior to balancing the classes are shown in Tables 6.1, 6.2, 6.3, and 6.4.

Thus, we created a more balanced training-validation set prior to training each model by undersampling the majority classes in the labeling schema used by the model. This allowed the model to more effectively learn the different classes. The resulting size of the training-validation set for each model differed depending on the composition of this set for the specific labeling schema being evaluated.

Table 6.1: Composition by class for original groups (Labels 1 and 5)

	Label 1	**Label 5**
Total images	7,070	7,061
Image class = 0	1,324 images (18.7%)	1,391 images (19.7%)
Image class = 1	2,905 images (41.1%)	2,970 images (42.1%)
Image class = 2	1,587 images (22.4%)	1,533 images (21.7%)
Image class = 3	1,254 images (17.7%)	1,167 images (16.5%)

Table 6.2: Composition by class for k = 2 groups (Labels 2 and 6)

	Label 2	**Label 6**
Total images	7,070	7,061
Image class = 0	5,274 images (74.6%)	5,432 images (76.9%)
Image class = 1	1,796 images (25.4%)	1,629 images (23.1%)

Table 6.3: Composition by class for k = 3 groups (Labels 3 and 7)

	Label 3	**Label 7**
Total images	7,070	7,061
Image class = 0	3,751 images (53.1%)	3,867 images (54.8%)
Image class = 1	2,534 images (35.8%)	2,474 images (35.0%)
Image class = 2	785 images (11.1%)	720 images (10.2%)

Table 6.4: Composition by class for k = 4 groups (Labels 4 and 8)

	Label 4	**Label 8**
Total images	7,070	7,061
Image class = 0	2,771 images (39.2%)	3,001 images (42.5%)
Image class = 1	2,640 images (37.3%)	2,593 images (36.7%)
Image class = 2	1,269 images (17.9%)	1,142 images (16.2%)
Image class = 3	390 images (5.5%)	325 images (4.6%)

Because the testing and selection sets remained skewed to reflect the real-world data, we evaluated the models on two metrics: overall accuracy and average accuracy by class, with a breakdown of each class's individual accuracy. Average accuracy by class more heavily accounts for the model's performance on the non-majority classes, while overall accuracy reflects the model's performance on the actual terrain.

6.4 Models

The image classifiers were trained in fastai [25] using transfer learning on ResNet50 [26], a convolutional residual neural network developed for image classification on the ImageNet dataset [21]. We trained the top layer of the network, then the entire network. The images in the training-validation, selection, and testing sets were resized to 270 pixels by 480 pixels from their original size of 2160 pixels by 3840 pixels, maintaining the original aspect ratio of the images. We used the fastai default transforms [27], which apply a series of common image transformations to random images in the training set. The default horizontal flip transform was excluded because an idea for future work would be to repeat these experiments, balancing classes by oversampling images in the non-majority classes with a horizontal flip. We initially tested Model 1 with and without the default transformations. The performance on the validation set was improved with transformations, leading us to use transformations in training each of the models.

6.5 Results

After training the models using the training set, we evaluated the model's ability to learn each of the eight labeling schemas with the selection set. Based on each model's performance on the selection set, we chose two labeling schemas to move forward with to evaluate on the test set and use as a basis for comparison in future experiments. The performance of each model on the selection set is listed in Table 6.5.

The first comparison was between the two methods for sampling terrain, either a 1

Table 6.5: Performance of Models 1 - 8 on the selection set

	Labeling schema	**Overall accuracy**	**Accuracy by class**
Model 1	Label 1	34.75%	Class 0: 36.11% Class 1: 30.13% Class 2: 29.69% Class 3: 50.00% Average: 36.48%
Model 2	Label 2	71.19%	Class 0: 71.06% Class 1: 71.60% Average: 71.33%
Model 3	Label 3	55.65%	Class 0: 67.63% Class 1: 43.69% Class 2: 27.27% Average: 46.20%
Model 4	Label 4	45.76%	Class 0: 65.75% Class 1: 31.25% Class 2: 40.32% Class 3: 5.56% Average: 35.72%
Model 5	Label 5	45.48%	Class 0: 68.60% Class 1: 33.33% Class 2: 37.04% Class 3: 51.92% Average: 47.72%
Model 6	Label 6	73.45%	Class 0: 72.20% Class 1: 77.92% Average: 75.06%
Model 7	Label 7	60.17%	Class 0: 68.00% Class 1: 54.62% Class 2: 34.29% Average: 52.30%
Model 8	Label 8	50.00%	Class 0: 61.25% Class 1: 32.28% Class 2: 68.00% Class 3: 23.53% Average: 46.27%

Table 6.6: Comparing methods for sampling terrain based on selection set performance

	TSM 1		**TSM 2**		**Difference (TSM 2 - TSM 1)**	
	Overall accuracy	**Avg class accuracy**	**Overall accuracy**	**Avg class accuracy**	**Overall accuracy**	**Avg class accuracy**
Original groups	34.75%	36.48%	45.48%	47.72%	10.73%	11.24%
k = 2 groups	71.19%	71.33%	73.45%	75.06%	2.26%	3.73%
k = 3 groups	55.65%	46.20%	60.17%	52.30%	4.52%	6.10%
k = 4 groups	45.76%	35.72%	50.00%	46.27%	4.24%	10.55%
				Average	5.44%	7.91%

second sampling of z-axis acceleration readings centered at the timestamp corresponding to 5 meters ahead of the image (TSM 1) or a 1 second sampling of z-axis acceleration readings directly ahead of the image's timestamp (TSM 2). We performed this comparison by examining the two methods side by side for each method of discretizing the roughness metric. This comparison is shown in Table 6.6.

TSM 2 performed better than TSM 1 in both overall accuracy (anywhere from 2.26% to 10.73% better, 5.44% better on average) and average accuracy by class (anywhere from 3.73% to 11.24% better, 7.91% better on average) for each method of discretizing the roughness metric, so we decided to move forward with TSM 2.

The next comparison was between methods for discretizing the roughness metric, examining only the second method for sampling the terrain. The model trained with k = 2 groups had both the highest accuracy and highest average accuracy by class, likely because the classifier had to learn only two classes as opposed to three or four.

The model trained with the k = 3 groups performed significantly worse than the model trained with the k = 2 groups, with a 13.28% decrease in overall accuracy and a 22.76% decrease in average accuracy by class, while it performed only slightly better than the model trained with the k = 4 groups, with 10.17% better overall accuracy and 6.03% better average accuracy by class. Thus, because the jump in performance from the model trained with the k = 2 groups to the model trained with the k = 3 groups was significantly larger than the jump in performance from the model trained with the k = 3 groups to the model

Table 6.7: Performance of Model 6 and Model 8 on the held out test set

	Labeling schema	**Overall accuracy**	**Accuracy by class**
Model 6	Label 6	69.91%	Class 0: 73.62% Class 1: 58.71% Average: 66.17%
Model 8	Label 8	51.32%	Class 0: 74.95% Class 1: 45.50% Class 2: 18.46% Class 3: 0.00% Average: 34.73%

trained with the k = 4 groups and the k = 4 groups provided more specific information about the upcoming terrain, we decided that the k = 4 groups were a more optimal choice than the k = 3 groups.

The overall accuracy of the model trained with the k = 4 groups was 4.52% better than the model trained with the original groups, while the average accuracy by class of the model trained with the original groups was 1.45% better than the model trained with the k = 4 groups. We decided that the increase in overall accuracy of the k = 4 groups compared to the much smaller increase in average accuracy by class of the original groups made the k = 4 groups a better method for discretizing the data.

Our final decision was to move forward with both Labels 6 and 8, the k = 2 and k = 4 groups for discretizing the data with TSM 2 (a 1 second sampling of z-axis acceleration directly ahead of the image's timestamp), as a baseline for comparison for the remaining experiments in this study. Figure 2.1 and Figure 2.2 show sample batches of images labeled with Labels 6 and 8.

6.6 Evaluation on the test set

Once we had chosen the two labeling schemas to use throughout the remaining experiments in this study, we evaluated the corresponding models on the held out test set. The results are summarized in Table 6.7.

Chapter 7

Research Question 4: How can we ensure the roughness classifiers are learning?

7.1 Overview

While we chose to extract images from the videos at 1 second intervals to decrease the amount of overlap between images, some images may have still contained parts of the terrain visible in chronologically consecutive images. For this reason, a random split between the training and testing sets, as was implemented in Chapter 6, could mean that parts of the terrain visible in the training set may be seen again in the testing set. While images with overlapping segments of terrain may have different labels, different lighting conditions, and different camera angles, among a variety of potential differences, we wanted to limit the amount of overlap as much as possible to ensure that the performance of the models was because the models had learned to predict terrain roughness as opposed to memorizing any potential overlap between chronologically consecutive images.

We decided to perform a second round of experiments using Labels 6 and 8 with a different method for splitting the data into training, validation, and testing sets. Instead of randomly splitting the data, we decided to split the data chronologically. This method will be referred to as a "chronological split". For each data collection session, the images were sorted chronologically. The first 70% of the images in each session were used for training, the next 15% of the images were used for validation, and the final 15% of the images were used for testing. While some sessions travelled the same paths as previous sessions, trail conditions varied much more significantly between sessions than within a single session. This split can be thought of as follows: "During each session, I taught the vehicle to predict terrain roughness for the first 85% of the ride, then the vehicle was able to predict terrain roughness on its own for the last 15% of the ride."

The random split assigned 80% of the data for training and validation, 5% of the data

for selecting labeling schemas, and 15% of the data for testing. Since we already selected labeling schemas, we no longer needed a selection set for this experiment. We wanted to maintain a testing set with 15% of the overall data in order to effectively compare the results from the models trained with the random split with the results from the models trained with the chronological split. Further, where we previously used a random 20% of the training-validation set for validation to determine the number of epochs to train and prevent overfitting, in the chronological split we wanted to make sure that the validation set did not have any overlap with the training set. For this reason, we created distinct training and validation sets.

7.2 Balanced classes

As discussed in Chapter 6, the classes in Labels 6 and 8 were very skewed. Similarly to the models trained with the random split, we trained the models in this experiment with balanced training and validation sets created by undersampling the majority classes. We had initially tried only balancing the training set, however, since the validation loss was used as an indicator for the number of epochs to train the model, a validation set with skewed classes minimized loss by predicting the majority class. Thus, we also balanced the validation set in order to ensure the model was learning to predict each of the classes.

7.3 Models

We used the same architecture as described for Models 1 - 8 with the same image resizing and transformations. Models 6 and 8 are the same models described in the previous chapter, and this chapter introduced two additional models.

Table 7.1: Performance of Models 6, 8, 9, and 10 on their respective test sets

Model	Labeling schema	Split	Overall accuracy	Accuracy by class
Model 6	Label 6	Random	69.91%	Class 0: 73.62% Class 1: 58.71% Average: 66.17%
Model 9	Label 6	Chronological	70.19%	Class 0: 72.76% Class 1: 62.11% Average: 67.44%
Model 8	Label 8	Random	51.32%	Class 0: 74.95% Class 1: 45.50% Class 2: 18.46% Class 3: 0.00% Average: 34.73%
Model 10	Label 8	Chronological	52.92%	Class 0: 72.41% Class 1: 45.95% Class 2: 34.22% Class 3: 7.14% Average: 39.93%

7.4 Results

We trained models using Labels 6 and 8, as selected in Chapter 6, with the training, validation, and testing sets split chronologically. These models will be referred to as follows: Model 9 was trained using Label 6 (TSM 2, k = 2 groups) and Model 10 was trained using Label 8 (TSM 2, k = 4 groups). The results are summarized and compared to Model 6 (Label 6, random split) and Model 8 (Label 8, random split) in Table 7.1.

Not only did Models 9 and 10 achieve comparable performance to Models 6 and 8, respectively, but they surpassed the corresponding models with the random splits in both overall accuracy and average accuracy by class. This increase in performance could be due to the fact that the models trained with the random split had 5% of the data allocated for selecting labeling schemas, while these images were used as part of the training and validation sets for Models 9 and 10. The goal of this experiment was to ensure that the models could learn to predict terrain roughness without memorizing parts of terrain potentially overlapping in chronologically consecutive images, which it succeeded in doing. However, in addition to achieving this goal, the models trained in this experiment with the chronological split using Labels 6 and 8 present our best models, on the basis of both

overall accuracy and average accuracy by class, for roughness classification with these two labels.

Chapter 8

Research Question 5: Can an attention region around the upcoming drivable terrain assist the classifiers in predicting terrain roughness?

8.1 Overview

The goal of this experiment was to determine whether adding an attention region around the drivable terrain would improve the performance of the roughness classifiers. In order to create an attention region around the drivable terrain, we trained a semantic segmentation network to identify the pixels in the image corresponding to drivable terrain and then colored all non-path pixels either black or white. We experimented with three variations of the images in our dataset, as shown in Figure 8.1: 1) the original images, 2) images where the attention region was specified by coloring all non-path pixels black (which will be referred to as the "dark attention region"), and 3) images where the attention region was specified by coloring all non-path pixels white (which will be referred to as the "light attention region"). We trained models with Labels 6 and 8 using a chronological split and each of these three image variations.

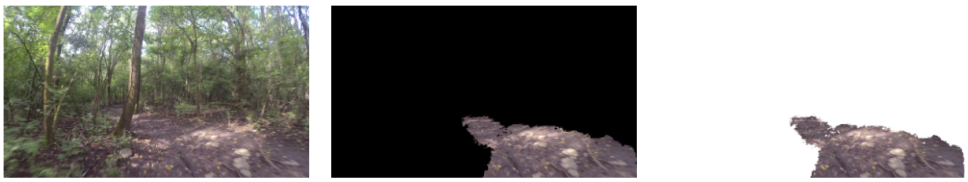

Figure 8.1: Example image, dark attention region, light attention region

8.2 Method

We began by training a semantic segmentation network to classify all pixels in our images as either "path" or "other". This segmentation network is described in detail in the

Table 8.1: Roughness classifiers trained in the attention region experiments

Model	**Label**	**Attention region**
Model 11 original	Label 6	None
Model 11 dark	Label 6	Dark
Model 11 light	Label 6	Light
Model 12 original	Label 8	None
Model 12 dark	Label 8	Dark
Model 12 light	Label 8	Light

"Segmentation network" section below. We used the predictions of the resulting network to save two versions of each image, one with a dark attention region and one with a light attention region. We trained 3 versions of the roughness classifier for each of Labels 6 and 8 using the original images, a light attention region, and a dark attention region. The models are evaluated on the test set of images with the same attention region used in training the model. These models are summarized in Table 8.1.

We used the same training, validation, and testing sets for training and evaluating models with the same labeling schemas to determine whether an attention region improved the performance of the roughness classifiers.

8.3 Segmentation network

This section describes the segmentation network trained to classify pixels as either "path" or "other". We started by selecting 187 random images of the 7,070 images validated to contain a path for the training set and 63 random images for the test set. We labeled each pixel as either "path" or "other". We used a random 20% of the training set for the validation set to determine the number of epochs to train and prevent overfitting. We evaluated our segmentation model on two metrics: pixel-wise accuracy and Intersection over Union (IoU). We used the UNet architecture provided by fastai [25]. All images were resized to half of their original size. We used the fastai default transformations [27], which apply some common image transformations to random images in the training set, ex-

cluding the horizontal flip in case our network did not achieve sufficient performance and we needed to augment all images in the training set with a horizontal flip. We evaluated the model on the test set and achieved an accuracy of 96.96% and an IoU of 0.9687 (where IoU ranges from 0 to 1, with 1 being the best possible value).

We then expanded our training set to include 376 images and repeated the above process, this time with the horizontal flip transformation applied randomly to some images in the training set. We achieved an accuracy of 97.78% and IoU of 0.9772.

Observing the test set, we noticed that the network struggled to make the correct predictions on images with dark lighting. Thus, we trained another version of this network where we added more images with dark lighting to the training set. Observing the same test set, this network produced much more desirable results on images with dark lighting. Additionally, this network was inclined to predict more path rather than less, which we felt was a desirable characteristic for our purposes. We would prefer that images with an attention region have some non-path regions visible as opposed to not enough visible path. This network achieved an accuracy of 97.15% and an IoU of 0.9704 on the test set. It is important to note that the results of the previous versions of this network on this same test set influenced the data we added to the training set, so unlike the previous two versions of the network, these metrics are not an indicator of how well the network will generalize. However, these metrics indicate that the segmentation network performed very well on the test data, and our visual confirmation that this version of the segmentation network was able to better detect the path in images with dark lighting meant that this version of the network was ideal for our purposes. We moved forward with this version of the network to apply the attention region to our images.

8.4 Preparing the dataset

We used the resulting network to save a version of each of the 7,061 images in our dataset where the pixels classified as “path” were tinted dark blue and the pixels classified

as “other” were tinted light blue, referred to as the “overlay”. Figure 8.2 shows example images with their correct labels and overlays. We then manually validated each of these 7,061 images with their overlays and assigned them a score from -2 to 2 based on the following criteria:

- -2: Many “path” pixels were classified as “other”.
- -1: A moderate amount of “path” pixels were classified as “other”.
- 0: Most pixels were correctly classified.
- 1: A moderate amount of “other” pixels were classified as “path”.
- 2: Many “other” pixels were classified as “path”.

This criteria can also be visualized in Figure 8.3. The number of images with each score are shown in Table 8.2.

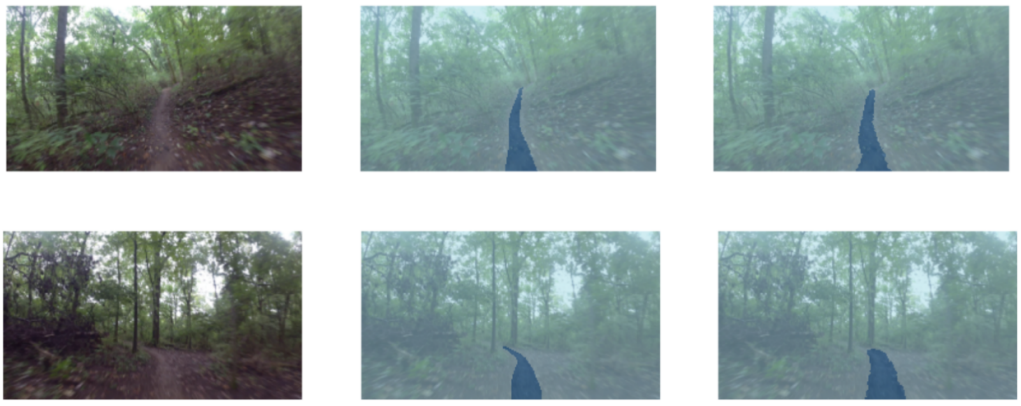

Figure 8.2: Example images, images with correct labels, images with overlays (labels predicted by the segmentation network)

We included all images with a score of 0 or 1 in our dataset. An image scoring -2 or -1 may not contain enough of the path to provide insight about terrain roughness, while an image scoring 2 may include too many non-path pixels, thus defeating the purpose of an attention region. Images scoring 0 were almost perfectly segmented, and images scoring 1 contained a little extra information, but not enough to detract from the attention region.

-2
Image 970326329s
801ms

-1
Image 969905951s
165ms

0
Image 970328645s
688ms

1
Image 970325796s
802ms

2
Image 970588434s
254ms

Figure 8.3: Example overlay images with scores -2 to 2

Table 8.2: Image scores for segmentation validation

Score	**Number of images**
-2	623 (8.81%)
-1	869 (12.29%)
0	3,642 (51.51%)
1	1,739 (24.60%)
2	197 (2.79%)

8.5 Training and testing splits

We used the chronological training, validation, and testing sets as described in Chapter 7. Table 8.3 describes the number of images in the dataset after removing images which did not contain a valid attention region.

8.6 Balanced classes

The compositions of the training and validation sets for Labels 6 and 8 are shown in Table 8.4. It can be observed that both sets are skewed towards classes of lower roughness.

Table 8.3: Images in the dataset after removing images without a valid attention region

	Original	**After attention region validation**
Total images	7,061	5,375
Images in training set	4,943 (70.00%)	3,589 (66.77%)
Images in validation set	1,058 (14.98%)	900 (16.74%)
Images in testing set	1,060 (15.01%)	886 (16.48%)

Table 8.4: Composition of training and validation sets prior to balancing classes for attention region networks

	Training set	**Validation set**
Label 6 **(Models 11 original, dark, and light)**	Class 0: 2,838 images (79.07%) Class 1: 751 images (20.93%) Total: 3589 images	Class 0: 676 images (75.11%) Class 1: 224 images (24.89%) Total: 900 images
Label 8 **(Models 12 original, dark, and light)**	Class 0: 1,648 images (45.92%) Class 1: 1,267 images (35.30%) Class 2: 529 images (14.74%) Class 3: 145 images (4.04%) Total: 3589 images	Class 0: 366 images (40.67%) Class 1: 333 images (37.00%) Class 2: 150 images (16.67%) Class 3: 51 images (5.67%) Total: 900 images

Table 8.5: Composition of training and validation sets after balancing classes for attention region networks

	Training set	**Validation set**
Label 6 **(Models 11 original, dark, and light)**	Class 0: 827 images (52.41%) Class 1: 751 images (47.59%) Total: 1578 images	Class 0: 246 images (52.34%) Class 1: 224 images (47.66%) Total: 470 images
Label 8 **(Models 12 original, dark, and light)**	Class 0: 1000 images (37.40%) Class 1: 1000 images (37.40%) Class 2: 529 images (19.78%) Class 3: 145 images (5.42%) Total: 2674 images	Class 0: 298 images (37.39%) Class 1: 298 images (37.39%) Class 2: 150 images (18.82%) Class 3: 51 images (6.40%) Total: 797 images

We balanced the training and validation sets by undersampling the majority classes. Table 8.5 shows the sizes of the balanced training and validation sets.

8.7 Models

We implemented our roughness classifiers using fastai [25]. We applied transfer learning with the ResNet50 [26] convolutional neural network as our architecture, training the top layer of the network then unfreezing the network to train every layer. The validation set was used to determine the number of epochs to train and prevent overfitting. The original images were 2160 by 3840 pixels and we resized them to 270 by 480 pixels, decreasing their dimensions by a factor of 8. This transformation was applied to all images in the training, validation, and testing sets. We used the default transformations in fastai [27], which apply some commonly used image transformations to images in the training set selected at random. However, we elected not to use the horizontal flip transform. A potential area for future work would be to replicate these experiments, balancing the classes by oversampling images in the non-majority classes with a horizontal flip.

Table 8.6: Performance of Models 11 original, dark, and light and Models 12 original, dark, and light on the test set

	Model 11 (Label 6)		**Model 12 (Label 8)**	
	Overall accuracy	Avg class accuracy	Overall accuracy	Avg class accuracy
Original images	68.28%	Class 0: 66.92% Class 1: 72.32% Average: 69.62%	55.19%	Class 0: 71.95% Class 1: 52.25% Class 2: 34.94% Class 3: 8.82% Average: 41.99%
Dark attention region	49.55%	Class 0: 39.73% Class 1: 78.57% Average: 59.15%	44.92%	Class 0: 67.42% Class 1: 43.54% Class 2: 9.04% Class 3: 0.00% Average: 30.00%
Light attention region	67.38%	Class 0: 72.36% Class 1: 52.68% Average: 62.52%	42.44%	Class 0: 62.89% Class 1: 15.02% Class 2: 62.05% Class 3: 2.94% Average: 35.72%

8.8 Results

The results are summarized in Table 8.6. In all cases, the models trained with the original images had a greater overall accuracy and average accuracy by class. Model 11 dark, however, was able to best identify Class 1, despite having the lowest overall accuracy between Model 11 original, dark, and light. Model 12 dark did not predict Class 3 for a single image in the test set. These results suggest that while the drivable terrain holds some indicators of upcoming terrain roughness, there are other contextual clues in the non-path regions of the image which assist the classifier in learning terrain roughness. It can be noted that Model 12 original has increased performance in both overall accuracy and average accuracy by class compared to Model 10 (Label 8, chronological split). However, the size of the testing set used for Model 12 contained only 886 out of the 1,060 in the testing set used for Model 10. Thus, Model 10 remains our proposed model for Label 8 due to its use of the full dataset after sensor and visual validation, as opposed to Model 12 which uses the subset of these images which passed the additional criteria of attention region validation.

Chapter 9

Future work

Because off-road self-driving is relatively new, there are many areas for expanding this research in future work. In the context of labeling images with a measure of roughness derived from the IMU z-axis acceleration readings, it would be beneficial to determine a method of terrain sampling that accounts for all of the visible terrain in an image. This may require advanced equipment at the time of data collection to store the amount of terrain visible at any given time. The 1 second sample used in this study corresponded to different lengths of terrain depending on the vehicle's speed. A roughness metric accounting for all of the visible terrain in an image could improve the performance of the roughness classifiers.

Another training and testing split that we considered investigating was splitting the data by session. Consider data collection from 20 sessions, each including the same amount of valid images. The images from the first 14 sessions would be used for training (70%), the images from the next 3 sessions would be used for validation (15%), and the images from the last 3 sessions would be used for testing (15%). This can be thought of intuitively as follows: "I taught the vehicle to predict terrain roughness for 17 sessions, then it was able to predict terrain roughness on its own for 3 sessions." We had originally wanted to test this with our data, however, the number of valid images in each session did not allow for this split.

The attention region models used all images which scored a 0 or 1 in attention region validation, where the images scoring 1 had some pixels marked as "path" that should have been marked "other". This meant that some images used for terrain learning did not have a precise attention region. These experiments could be repeated with a precise attention region manually drawn around the path for all of the images in the dataset. Alternatively,

instead of coloring all non-path pixels a different color, the image could be cropped to the smallest possible size including all of the path and maintaining the image's aspect ratio. This could be done with an object detection network. Eppel noted that both defining an attention region by replacing the color of all pixels not in the attention region and cropping the image to the smallest rectangular region around the attention region result in loss of contextual information contained in the pixels not in the attention region [28, 29]. Eppel found increased performance passing a binary attention region as an additional parameter to the network [29]. This is another method for specifying an attention region that could be explored for off-road terrain roughness prediction.

In each of our experiments, the composition of the dataset for each labeling schema was significantly skewed towards one or two majority classes. We balanced classes by undersampling the majority classes. However, sometimes the non-majority classes only contained a couple hundred images, which may not have been enough data to effectively learn these classes. Future work could oversample the non-majority classes by applying a horizontal flip to images in these classes. Our models intentionally did not use the default horizontal flip transform provided by fastai so that balancing classes by oversampling with a horizontal flip could be explored in future work.

Transfer learning was used in training each of our roughness classifiers. Future work could examine the use of custom architectures for off-road terrain roughness classification models.

Finally, the data was collected with a mountain bike in Percy Warner Park in Nashville, TN. Future work could explore collecting data from other trails in Nashville, TN, or in other geographical regions entirely. Additionally, data could be collected with other types of vehicles, such as a robot or a car.

The results of this study show that it is possible to use solely monocular image data to learn about off-road terrain roughness. Future work can expand on the experiments presented in this study in order to improve off-road terrain roughness prediction.

Chapter 10

Concluding remarks & lessons learned

This study examined terrain roughness classification for off-road autonomous vehicles. We collected data with a mountain bike in off-road trails where there was a clear distinction between drivable and undrivable terrain.

The first experiment in this study examined how to label images with the standard deviation of a 1-second sampling of z-axis acceleration readings. We considered two methods for sampling terrain (TSM 1 and TSM 2) and four methods for discretizing the continuous roughness metric (original groups, k = 2 groups, k = 3 groups, and k = 4 groups). This resulted in eight possible labeling schemas. We trained roughness classifiers to learn each of the eight potential labeling schemas, and determined that Labels 6 (TSM 2, k = 2 groups) and 8 (TSM 2, k = 4 groups) were most effectively learned.

These models were trained with a random training and testing split. While images were extracted at 1 second intervals to minimize overlap, we further minimized potential overlap by evaluating Labels 6 and 8 using a chronological training and testing split. The chronological split models achieved comparable results to the random split models for both Labels 6 and 8, suggesting that the models were, in fact, learning to predict terrain roughness. Our proposed roughness classification models for Labels 6 and 8 were Model 9 (Label 6, chronological split) and Model 10 (Label 8, chronological split). Model 9 achieved 70.19% overall accuracy and 67.44% average accuracy by class, and Model 10 achieved 52.92% overall accuracy and 39.93% average accuracy by class.

Finally, we examined whether the use of an attention region around the drivable terrain could improve terrain roughness prediction. We trained a semantic segmentation network to color all non-path pixels either black or white. We trained models for Labels 6 and 8 with each of the following variations: 1) no attention region, 2) dark attention region, and 3)

light attention region. In all cases, the models utilizing images without an attention region had increased performance compared to the models utilizing images with either variation of an attention region.

The following are the key lessons that we learned from conducting this research:

- **We can use a single, monocular image to learn about the future kinetics of the vehicle.** One of the major distinctions between roadway and off-road environments is terrain roughness. While both off-road and roadway driving concern semantics, such as whether there are objects present in front of the vehicle, off-road autonomous driving must also consider the roughness of the upcoming terrain. Particularly rough terrain could lead to negative consequences for the vehicle, such as a loss of traction or control. Semantics matter in an off-road setting, but unlike roadway settings, roughness matters, too. This study showed that we can use a single, monocular image to learn about the kinetics of the vehicle as it traverses the upcoming rough terrain shown in the image.

- **The attention region experiments showed that there is important contextual information in the non-path pixels of off-road images for understanding terrain roughness.** While models trained utilizing images both with and without an attention region demonstrated some ability to predict upcoming terrain roughness, indicating that there is information contained in the path pixels of the images for predicting terrain roughness, models utilizing images without an attention region consistently performed better in both evaluation metrics than models utilizing images with an attention region.

- While the overall accuracies and average accuracies by class of the roughness classifiers in this research show that some information is being learned about upcoming terrain roughness, **an open question is whether the performance of the models in our research is sufficient to assist real-world off-road autonomous driving or**

whether significant advancements will need to be made before these models can be implemented in a real-world setting.

BIBLIOGRAPHY

[1] NHTSA. Automated vehicles for safety, Jun 2020. Available: https://www.nhtsa.gov/technology-innovation/automated-vehicles.

[2] S. George Fernandez, K. Vijayakumar, R Palanisamy, K. Selvakumar, D. Karthikeyan, D. Selvabharathi, S. Vidyasagar, and V. Kalyanasundhram. Unmanned and autonomous ground vehicle. *International Journal of Electrical and Computer Engineering (IJECE)*, 9(5):4466, 2019.

[3] Akhil Kurup, Sam Kysar, and Jeremy P. Bos. SVM based sensor fusion for improved terrain classification. *Autonomous Systems: Sensors, Processing, and Security for Vehicles and Infrastructure 2020*, 2020.

[4] Mingliang Mei, Ji Chang, Yuling Li, Zerui Li, Xiaochuan Li, and Wenjun Lv. Comparative study of different methods in vibration-based terrain classification for wheeled robots with shock absorbers. *Sensors*, 19(5):1137, 2019.

[5] Chengchao Bai, Jifeng Guo, and Hongxing Zheng. Three-dimensional vibration-based terrain classification for mobile robots. *IEEE Access*, 7:63485–63492, May 2019.

[6] Ekim Yurtsever, Jacob Lambert, Alexander Carballo, and Kazuya Takeda. A survey of autonomous driving: Common practices and emerging technologies. *IEEE Access*, 8:58443–58469, 2020.

[7] Santiago Royo and Maria Ballesta-Garcia. An overview of lidar imaging systems for autonomous vehicles. *Applied Sciences*, 9(19):4093, 2019.

[8] Norhafizan Ahmad, Raja Ariffin Ghazilla, Nazirah M. Khairi, and Vijayabaskar Kasi.

Reviews on various inertial measurement unit (IMU) sensor applications. *International Journal of Signal Processing Systems*, page 256–262, 2013.

[9] Christian Weiss, Hashem Tamimi, and Andreas Zell. A combination of vision- and vibration-based terrain classification. *2008 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2008.

[10] Yumi Iwashita, Kazuto Nakashima, Adrian Stoica, and Ryo Kurazume. TU-Net and TDeepLab: Deep learning-based terrain classification robust to illumination changes, combining visible and thermal imagery. *2019 IEEE Conference on Multimedia Information Processing and Retrieval (MIPR)*, 2019.

[11] Christopher A. Brooks and Karl Iagnemma. Vibration-based terrain classification for planetary exploration rovers. *IEEE Transactions on Robotics*, 21(6):1185–1191, 2005.

[12] Christian Weiss, Holger Frohlich, and Andreas Zell. Vibration-based terrain classification using support vector machines. *2006 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2006.

[13] Hendrik Dahlkamp, Adrian Kaehler, David Stavens, Sebastian Thrun, and Gary Bradski. Self-supervised monocular road detection in desert terrain. *Robotics: Science and Systems II*, 2006.

[14] Sebastian Thrun, Mike Montemerlo, Hendrik Dahlkamp, David Stavens, Andrei Aron, James Diebel, Philip Fong, John Gale, Morgan Halpenny, Gabriel Hoffmann, and et al. Stanley: The robot that won the DARPA Grand Challenge. *Springer Tracts in Advanced Robotics*, page 1–43, 2007.

[15] DARPA RSS. The Grand Challenge. Available: https://www.darpa.mil/about-us/timeline/-grand-challenge-for-autonomous-vehicles.

[16] David Stavens and Sebastian Thrun. A self-supervised terrain roughness estimator for off-road autonomous driving. *arXiv:1206.6872*, 2006.

[17] Shastri Ram. *Semantic Segmentation for Terrain Roughness Estimation Using Data Autolabeled with a Custom Roughness Metric*. PhD thesis, Carnegie Mellon University, 2018.

[18] Vivekanandan Suryamurthy, Vignesh Sushrutha Raghavan, Arturo Laurenzi, Nikos G. Tsagarakis, and Dimitrios Kanoulas. Terrain segmentation and roughness estimation using rgb data: Path planning application on the centauro robot. *2019 IEEE-RAS 19th International Conference on Humanoid Robots (Humanoids)*, 2019.

[19] Mirko Nava, Jerome Guzzi, R. Omar Chavez-Garcia, Luca M. Gambardella, and Alessandro Giusti. Learning long-range perception using self-supervision from short-range sensors and odometry. *IEEE Robotics and Automation Letters*, 4(2):1279–1286, 2019.

[20] U.S. Department of Transportation/Federal Highway Administration. Highway statistics series, Sep 2020. Available: https://www.fhwa.dot.gov/policyinformation/statistics/2019/hm16.cfm.

[21] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. ImageNet: A Large-Scale Hierarchical Image Database. In *CVPR09*, 2009.

[22] Garmin Developers. Flexible and interoperable data transfer (fit) sdk. Available: https://developer.garmin.com/fit/overview/.

[23] Garmin Developers. FitCSVTool. Available: https://developer.garmin.com/fit/fitcsvtool/.

[24] Garmin Developers. Fit protocol. Available: https://developer.garmin.com/fit/protocol/.

[25] fastai. fastai v1 documentation. Available: https://fastai1.fast.ai/.

[26] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[27] fastai. vision.transform. Available: https://fastai1.fast.ai/vision.transform.html#Data-augmentation.

[28] Sagi Eppel. Setting an attention region for convolutional neural networks using region selective features, for recognition of materials within glass vessels. *arXiv:1708.08711*, 2017.

[29] Sagi Eppel. Classifying a specific image region using convolutional nets with an roi mask as input. *arXiv:1812.00291*, 2018.