AUTONOMOUS VEHICLE NAVIGATION AND OBSTACLE AVOIDANCE ALONG A PREDETERMINED VECTOR

by

Elijah Killian Pfeifer

A thesis submitted to the faculty of
The University of North Carolina at Charlotte
in partial fulfillment of the requirements
for the degree of Master of Science in
Applied Energy and Electromechanical Systems Engineering

Charlotte

2021

Approved by:

_____
Dr. Aidan Browne

_____
Dr. Michael Smith

_____
Dr. Wesley Williams

ABSTRACT

ELIJAH KILLIAN PFEIFER. Autonomous vehicle navigation and obstacle avoidance along a predetermined vector. (Under the direction of DR. AIDAN BROWNE)

Search and Rescue (SaR) operations rely on civilian volunteers to provide mandatory manpower when covering large areas of land. These SaR operations are typically conducted in remote environments where conditions can be hazardous and GPS signal intermittent. This project designed and tested an autonomous Unmanned Ground Vehicle (UGV) capable of sequentially navigating along a series of GPS waypoints in an effort to minimize harm to operators and bolster the ranks of SaR crews. An embedded controller was used to calculate vector trajectory for navigation and pathing corrections from data provided by an onboard GPS unit and compass. Scans from a 2D-LIDAR were processed and analyzed by an onboard computer before passing network-published shared variables to the embedded controller for obstacle detection and avoidance systems. A linear model dictated a predictive algorithm for navigating the GPS waypoints while GPS signal was unavailable. For testing, the UGV autonomously navigated within range of each waypoint before advancing to the next until all waypoints were met. For testing with intermittent GPS, the UGV autonomously navigated the same waypoints, but an additional system added randomly generated periods where the GPS was masked from the UGV's primary navigation system. This study results in a highly successful system of autonomous vehicle navigation and obstacle avoidance using vector trajectory but a limited system of navigating without GPS. An outline for future research in further improving the tested system is discussed.

DEDICATION

This thesis is dedicated to my parents, Lance and Thea. Their unwavering love and support have guided me throughout my pursuit of higher education.

ACKNOWLEDGEMENTS

I would like to acknowledge the guidance and professional support Dr. Aidan Browne has offered, not only throughout this thesis but also throughout my entire graduate career. The completion of this thesis would not have been possible without Dr. Browne's volunteered time and resources.

I would also like to acknowledge the valued input and guidance Dr. Wesley Williams and Dr. Michael Smith offered during this process; their expertise and critiques weighed heavily in the completion of this thesis.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

AI	Analog Input
AMR	Anisotropic Magnetoresistive
AO	Analog Output
AV	Autonomous Vehicle
CIM	Components Limited Industrial Motor
DIO	Digital Input and Output
GPS	Global Positioning System
HWF	Heading Weight Function
I^2C	Inter-Integrated Circuit
LIDAR	Light Detection and Ranging
MXP	myRIO Expansion Port
NI	National Instruments
NI-PSP	NI Publish-Subscribe Protocol
OCU	Operator's Control Unit
PoI	Point of Interest
SaR	Search and Rescue
SCL	Serial Clock
SDA	Serial Data
SRF	Sonic Range Finder

UART.RX	UART Receive Input
UART.TX	UART Transmit Output
UAV	Unmanned Aerial Vehicle
UGV	Unmanned Ground Vehicle
USB	Universal Serial Bus
UTC	Ultra-Tightly Coupled
VI	Virtual Instrument

CHAPTER 1: INTRODUCTION

1.1 Background

Search and Rescue (SaR) operations are practiced globally to locate, stabilize, and extract distressed individuals. These SaR operations typically consist of trained volunteers in teams who hike from a staging base to a strategic location known as a point of interest (PoI). During this hike, the SaR teams search PoIs and surrounding areas for any clues that might help determine the missing person's location. The teams may need to hike more than 20 miles before either finding the missing person or needing to set up camp until the morning to either resume searching or returning to the staging base. During their operations, complications are prevalent and dangerously unforeseeable as it is nearly impossible to know the physical state or exact location of the missing personnel.[1][2]

Finding the missing personnel is only the first step in a SaR operation, and the situation the individual may be in is wildly unpredictable; therefore, it is nearly impossible for the SaR operators to be prepared for all possible scenarios. To illustrate, if SaR teams find an individual with severe injuries who cannot be moved for fear of head trauma, then a helicopter evacuation may become necessary. In the likely case that a helicopter is not actively on standby, it can take several hours to prepare a helicopter and reach the PoI before the patient is extracted. During this waiting period, the SaR team may need medication for stabilizing, heavy splints to secure broken limbs, basic necessities such as food and water, or even warm clothes in case of rapidly dropping temperatures from approaching nightfall. In these conditions, the SaR team might not have the manpower nor the daylight to send another team from the staging base to the PoI to deliver the necessary items. Thus in SaR operations,

total manpower is one of the most crucial resources; the more manpower involved, the faster and safer the operation can be completed, which can be further improved by incorporating robotic platforms to assist operators in fundamentally reducing the required manpower and increasing the overall safety of the SaR operators.[1][2]

The first-ever documented use of a robotic platform in a SaR operation was the use of remote-operated Unmanned Ground Vehicles (UGV) during the 9/11 event, where the small teleoperated robots were able to navigate locations inaccessible to humans in search of survivors under the debris. Since then, remote-operated UGVs have assisted in dozens of SaR operations and disaster relief by offering several means of assistance, such as delivering weighted payloads or using onboard sensors to gather crucial information in hazardous zones [3].

1.2 Problem Statement

Through the years, advancements in technology have opened research opportunities in furthering the autonomy of UGVs and opening prospects of providing robotic platforms that are more accessible to SaR operations as autonomous navigation becomes more reliable. This research aims to produce a working proof-of-concept for autonomous vehicle navigation and obstacle avoidance along outdoor vectors predetermined by GPS waypoints with accommodation for potential intermittent GPS signal loss. Autonomous UGVs have several areas where complications arise during navigation of outdoor and unstructured environments, especially when there is an intermittent GPS signal. SaR operations would benefit greatly from cost-effective and reliable systems that would reduce required manpower; to this end, this research focuses on removing the need to manually pilot a UGV along a predetermined path by enabling the UGV to analyze its environment and determine navigation pathways autonomously.

1.3 Conceptual Design

This research's primary focus is on the autonomous navigation of a UGV in an outdoor unstructured environment while traveling along a series of GPS waypoints using vector trajectory. For this research, a GPS waypoint communication system acts as the breadcrumb trail for an autonomous UGV to follow. In practice, this trail would be strategically placed by SaR operators as they ventured forward until a PoI is reached where a robotic platform such as the UGV could assist. In such a case, the operator would radio the staging base with a request, resulting in the dispatch of an adequately equipped UGV that would autonomously navigate that particular operator's pre-placed GPS waypoint trail. Each waypoint would act as a checkpoint for the navigation system allowing the UGV to follow approximately the SaR operator's footsteps.

In this research, it is assumed that the GPS waypoints have been activated, and therefore the system has received all of the GPS coordinates. With a series of GPS waypoints loaded, the proposed system would calculate the vector trajectory between the UGV's current GPS location and the next waypoint in sequential order for a primary drive function. Once initiated, the drive function undergoes constant advance manipulation based on variables provided by the onboard sensor suite. Such manipulations include pathing decisions from LIDAR-based obstacle detection and GPS signal combined with onboard compass orientation to provide pathing corrections from drifting or obstacle avoidance. Ideally, an autonomous UGV could utilize high-end 3-dimensional LIDAR systems or computer stereo vision to assist in navigation, but these are expensive and computationally taxing. Since this research is cost-limited to using a 2D-LIDAR for obstacle detection, there are expected limitations to the capabilities of obstacle identification.

CHAPTER 2: LITERATURE REVIEW

Previous research, detailed below, has concluded that autonomous vehicles (AV) can reduce required manpower for dangerous operations such as SaR. Often during these dangerous situations, a robotic platform operating with full autonomy faces several challenges such as GPS signal intermittency and avoiding obstacles. While the recorded applications of demonstrated uses of AVs in SaR operations lack significant documentation, there has been research into practical solutions to the aforementioned challenges. Relevant documentation to the theories that were applied to this research paper is summarized below.

2.1 Unmanned Vehicles in Search and Rescue

The study performed for this paper was focused on unstructured environments like wooded areas; however, several environments that SaR operations are conducted in can significantly affect how efficiently different types of AVs can perform. As noted by Zoltan Beck, the most widely used type of robotics in search and rescue practice are Unmanned Aerial Vehicles (UAV). While UAVs are versatile in open-air environments, they have significant limitations when obstacles become clustered.[4] SaR operations are often unpredictable and can need to be performed in uncharted territory where outdoor terrain and foliage may limit UAV's accessibility. Beck claims that another AV, commonly known as a UGV, is highly applicable and flexible in the situations mentioned above. Unlike their UAV counterparts that have a payload limit, robust UGV platforms can be outfitted with an indefinite range of complex sensors or assembled for heavy payload operations such as injured personnel transportation.

The wide varieties of SaR operations result in the necessity of specialized payloads

for a UGV to be applicable within each unique situation's scope. To demonstrate this, Karma et al. summarized the relevant sensors that could be used on a UGV explicitly engaged in forest fire SaR operations and presented them in Table 2.1.

Table 2.1: Sensors for Payloads on UGVs Within Forest Fire SaR Operations [5]

Sensor	Applications
Visible camera	Video (motion, still); forest fire front expansion; identification of new fire spots; damage of infrastructures; weather assessment; location of people entrapped in fire or smoke; evacuation progress of an area
Infrared camera	Video (motion, still); detecting people and animals at night; detecting lost people in foliage, thick fog, or smoke; location of fire spots in smoldering fires
Radar	Detecting people in vegetation under all weather conditions; detecting animals in motion under trees and foliage
Chemical sensors	Detection of toxic chemical compounds; smoke detection; air quality monitoring
Mobile phones detector	Detection of mobile phones signal, carried by trapped people beyond fire smoke
LIDAR	Detection of clouds of emissions and obstacles; see through the fire smoke
Location sensors	GPS for providing coordinates and mapping
Environmental sensors	Devices that provide meteorological data such as precipitation, temperature, relative humidity, wind speed and direction

Furthermore, Karma et al. observed the advantages and limitations of unmanned vehicles in forest fire SaR operations in a field trial. The operational scenario tested

three UAVs and two UGVs within the four phases typically observed in a forest fire: pre-ignition, flaming, smoldering, and glowing. Each of these phases introduces its unique challenges with varying operating temperatures and a plethora of combustive or toxic gasses. The two UGVs were deployed for SaR operatives to locate an entrapped crew and provide safe guidance to an extraction zone. During this trial, the unmanned vehicles were equipped with chemical sensors allowing air quality monitoring while traversing areas that would otherwise put personnel at risk. The results of these tests proved the feasibility of incorporating unmanned vehicles in SaR operations

2.2 GPS Denied Operations and Vector Tracking

With manually-operated UGVs being dependent on a reliable wired or wireless connection with low latency and fully autonomous UGVs relying on a strong GPS signal, Endo et al. [6] suggest a compromise with a semi-autonomous control system. By utilizing environmental landmarks defined by the robot's view and displayed in the Operator's Control Unit (OCU), the robot is able to navigate to a designated landmark with minimal reliance on GPS or manual signal. In this particular study, the UGV control system is designed to assist in military operations that could reduce a human operator's exposure to high-risk situations in the field, such as explosive ordnance defusal. The landmark-based navigation system breaks down into two principal subsystems; the UGV and the OCU. The UGV is equipped with an onboard computer to process the IMU data and the images captured by the three cameras. This processed information is then sent to the OCU, allowing the user to select the desired landmark predefined by the UGV. While this methodology is not fully autonomous, it could prove to be a viable option during periods of GPS downtime.

Li et al.[7] proposed an ultratightly coupled (UTC) GPS/INS integrated navigation system specifically designed for an environment with severe GPS signal jamming to achieve full autonomy in navigation. In this theoretical system, a vector tracking

algorithm was integrated with Kalman filters as opposed to the traditional scalar tracking as seen in most UTC systems. The simulation results demonstrated a significant performance improvement in GPS denied situations; however, the complexity of the algorithm and the architecture behind it is difficult to implement into a real-time system with limited computational power.

2.3 Obstacle Detection and Avoidance

As opposed to the cost-expensive and processing-intensive 3D LIDARs commonly seen in autonomous road vehicles, Higuti et al. [8] developed a 2D LIDAR-based navigation algorithm allowing for a low cost and easily processed UGV navigation. The proposed problem was autonomous navigation of a small UGV through semi-structured agricultural rows of crops under cluttered canopies with no GPS signal. Higuti used the algorithm with a Raspberry Pi and a low-cost 2D LIDAR to achieve a 90 percent success rate in over 300 fully autonomous runs. The algorithm required such low computational complexity that it could complete its process in under 6.9 ms, falling well below the 2D LIDAR's update time of 25 ms. The limiting factor in this study was the need for a semi-structured environment to be successful. A primary factor of autonomously avoiding objects is deciding the best course of action for the UGV if an object is encountered. El Houddein Chouaib Harik [9] devised a novel approach to this problem by implementing a new Heading Weight Function (HWF). The HWF is designed to apply a decision-altering weight to beams of a LIDAR in the event that an obstacle is obstructing the robot's path. The determined heading (i.e., which side of the obstacle) produced by the function ensured a path temporarily void of other obstacles while maintaining navigation towards the desired location. Harik demonstrated in both simulations and experiments the effectiveness of the HWF in determining the optimal heading and was able to incorporate a condition that allowed the robot to derive a solution when presented with symmetrical beam weights on either side of the object.

CHAPTER 3: DESIGN OVERVIEW

3.1 System Overview

An unmanned ground vehicle (UGV) was designed to autonomously navigate and avoid obstacles along a path of GPS waypoints. The autonomous navigation of the UGV was also designed with a secondary goal of maintaining navigation, while the GPS location of the UGV was masked from the waypoint navigation system. A program was developed to allow the UGV to make autonomous decisions based on a suite of onboard sensors, including a GPS, compass, sonic range finder (SRF), and a LIDAR. The programs were developed using the LabVIEW development environment and Microsoft Visual Studio.

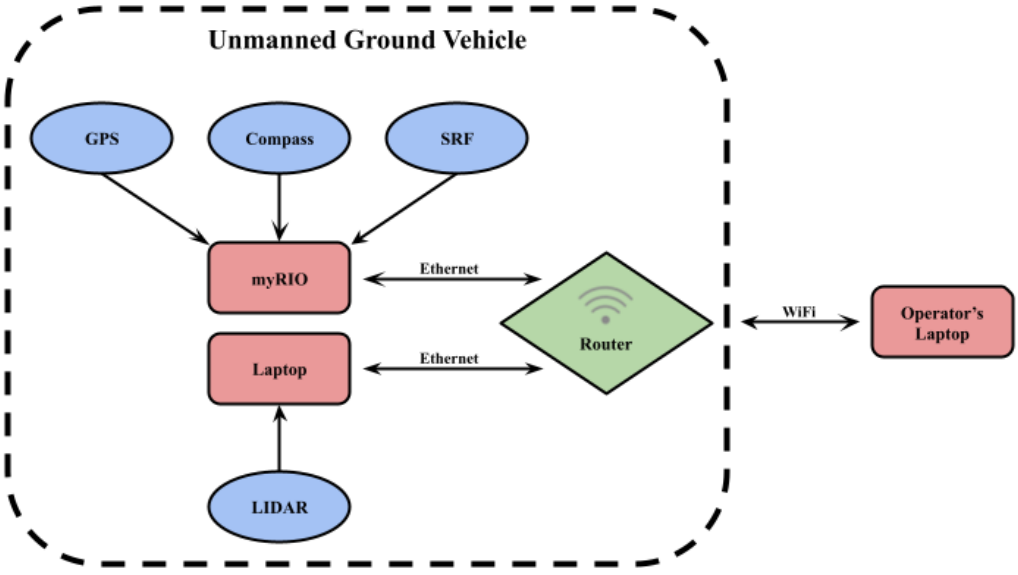

Figure 3.1: Communication between critical components

Figure 3.1 above displays how the UGV was equipped with a myRIO embedded controller and an onboard laptop connected via Ethernet to a TP-Link router allowing

for communication over a wireless network with the separate operator's laptop. The system allowed the operator's laptop to connect to the myRIO through the LabVIEW software enabling the UGV to have an autonomous and a manual operating mode. The manual operating mode allowed for the operator to input directional commands utilizing a USB joystick connected to the operator's laptop. The autonomous mode took user-defined parameters from the operator's laptop, such as GPS waypoint coordinates, and derived directional controls via the sensor data processing loops within the myRIO's program.

This system uses Microsoft Remote Desktop, which allows the operator to connect to and control the onboard laptop remotely. The onboard laptop's purpose was to provide power to the LIDAR and run a C++ script that set all local LIDAR parameters such as RPM and interpreted the data output. The onboard laptop also processed the raw data from the LIDAR through a LabVIEW VI and acted as a filter before passing critical information to the myRIO via network-published shared NI Publish-Subscribe Protocol (NI-PSP) variables.

3.2 System Configuration

Figure 3.2 below shows a top-down view of the UGV's hinged hood measuring 0.8 meters on each side and the mounting locations of the LV-MaxSonar-EZ1 MB1010 SRF, HP laptop, TP-Link router, and a GlobalSat BU-353-S4 GPS module.

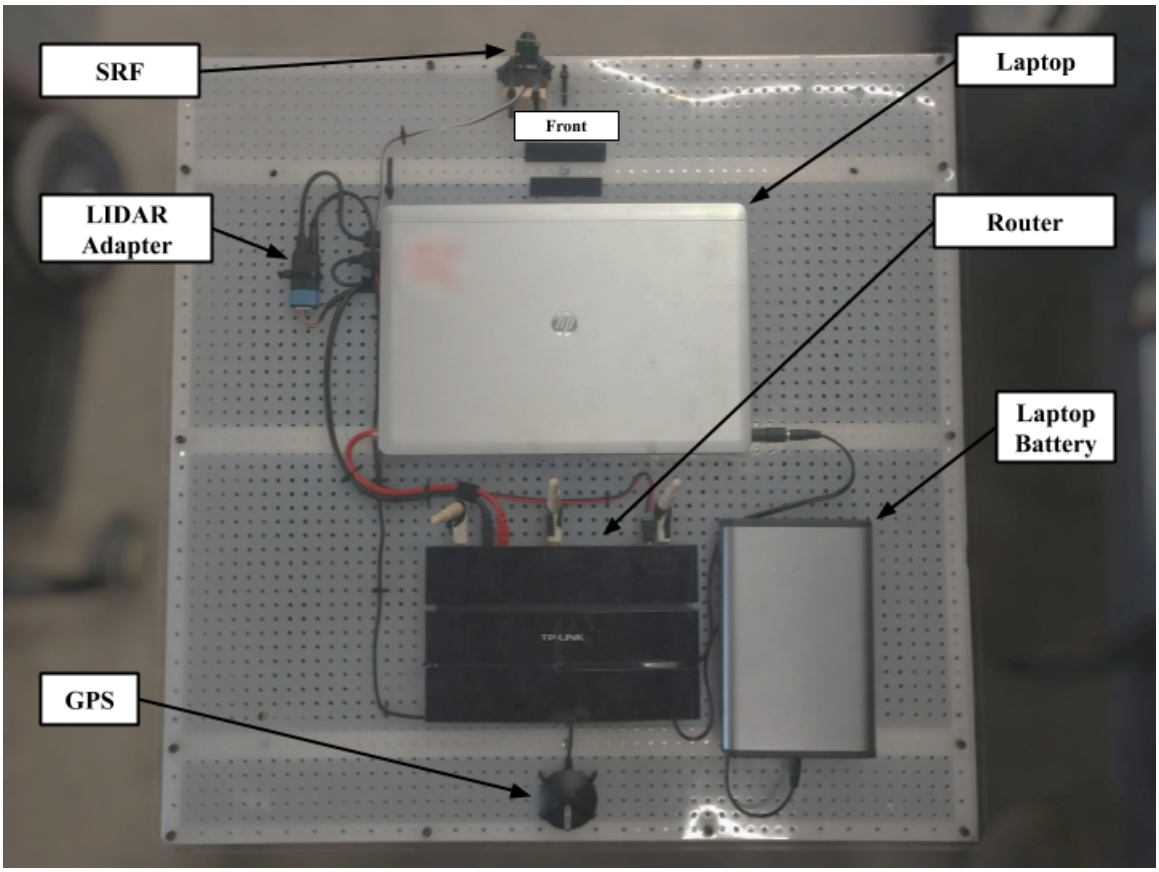

Figure 3.2: Top-Down view of UGV with labeled components

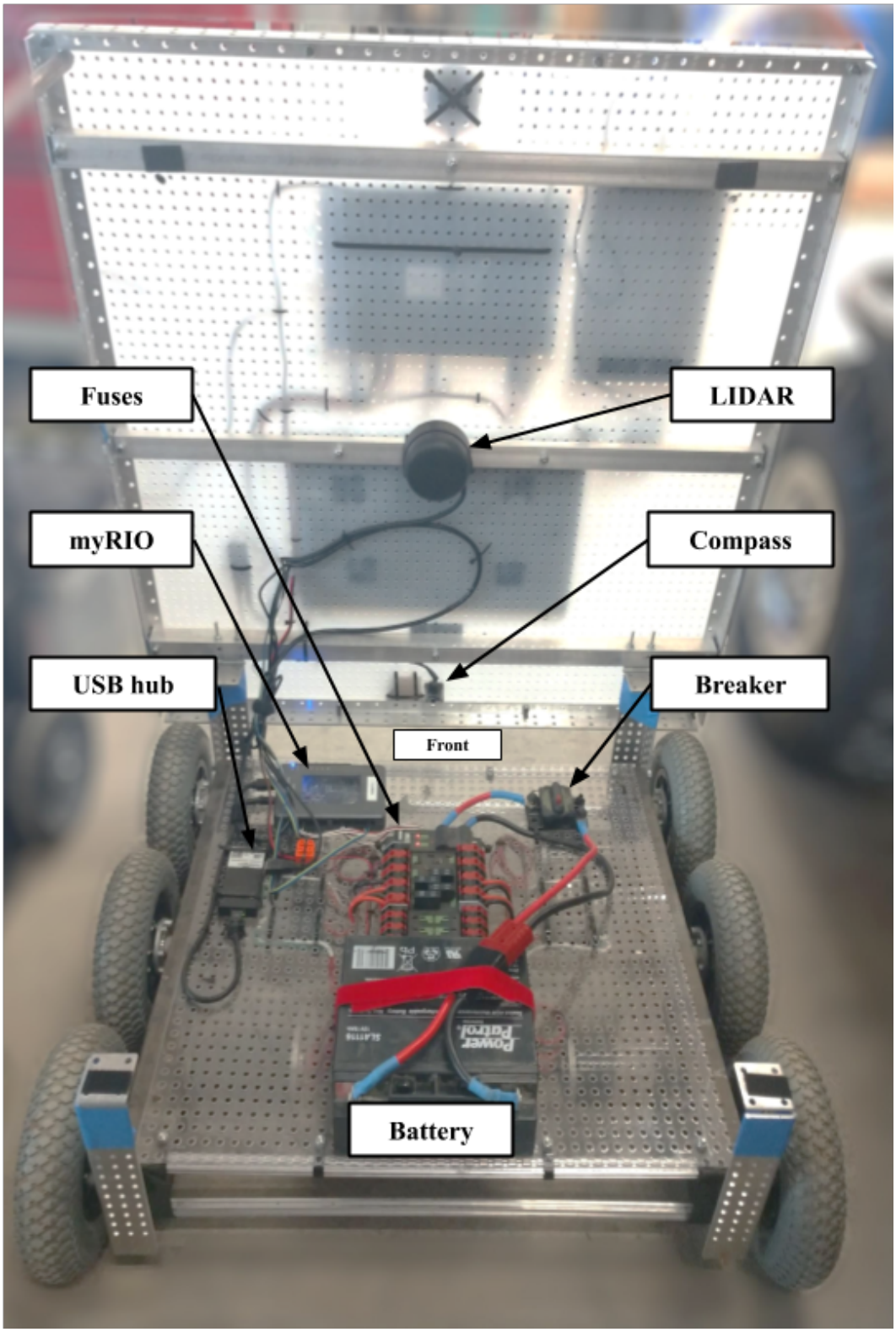

Figure 3.3: Mounting location of chassis equipment with the hood up

Figure 3.3 shows the chassis with the hinge-mounted hood in the raised position. The hood's under-side provided a mounting surface for the SLAMTEC RPLIDAR A3M1 and the Diligent PMOD CMPS1 compass, while the main chassis housed the motors gearboxes and all other electronics as labeled in the figure. The chassis and hood were both comprised of angle brackets for the framework and perforated polycarbonate sheets for mounting platforms. The chassis allowed for two sets of three 8-inch pneumatic wheels connected internally by a gearbox. Each set used two Chiaphua Components Limited Industrial Motor (CIM) motors and two VEX Robotics Victor SP motor drivers. Additionally, each set's middle wheel was slightly lower than the others to provide a more effortless skid-steer operation.

The motors and electronics on the UGV, except for the onboard laptop and LIDAR, were powered by a 12V, 18Ah SLA battery. The battery leads were fed to a distributing fuse box via a resettable breaker. The myRIO and router were spliced through a 5V regulator, and the motors were directly powered via a fuse box with a PWM signal set by the myRIO and interpreted by the motor drivers. The chassis, motor configuration, and power distribution were all developed by Steven Padgett under the guidance of Dr. Aidan Browne.[10] The total cost estimate of the UGV and all sensors was approximately $4,000, as seen in the Bill of Materials (BOM) located in Appendix A. From a cost-effectiveness perspective, it should be noted that a typical SaR helicopter costs $6,000 per hour of usage.[11]

CHAPTER 4: METHODOLOGY

4.1 Instrument Configuration

Due to hardware and software compatibility, the myRIO contained the final program where all inputs and controls eventually coincided. To allow for a responsive system in a time-sensitive test environment, measures needed to be taken in reducing the computational load on the myRIO. The following sections explain the configuration and purpose of each component within the autonomous system.

4.1.1 myRIO

A National Instruments (NI) myRIO-1900 embedded device was chosen as the system's primary control. A StarTech ST3300GU3B USB Hub with Ethernet Adapter was connected to the myRIO's USB Host Port to provide an Ethernet connection to the TP-LINK Wifi Router and allow for two additional USB ports that would be utilized by the GPS unit and an 8GB flash drive for data recording.

The myRIO-1900 offers two identical Expansion Port (MXP) connectors, "A" and "B". The configuration of these connectors can be seen in Figure 4.1, taken from the NI myRIO-1900 User Guide and Specifications document. This figure serves as a reference for wiring schematics found later in this chapter. [12]

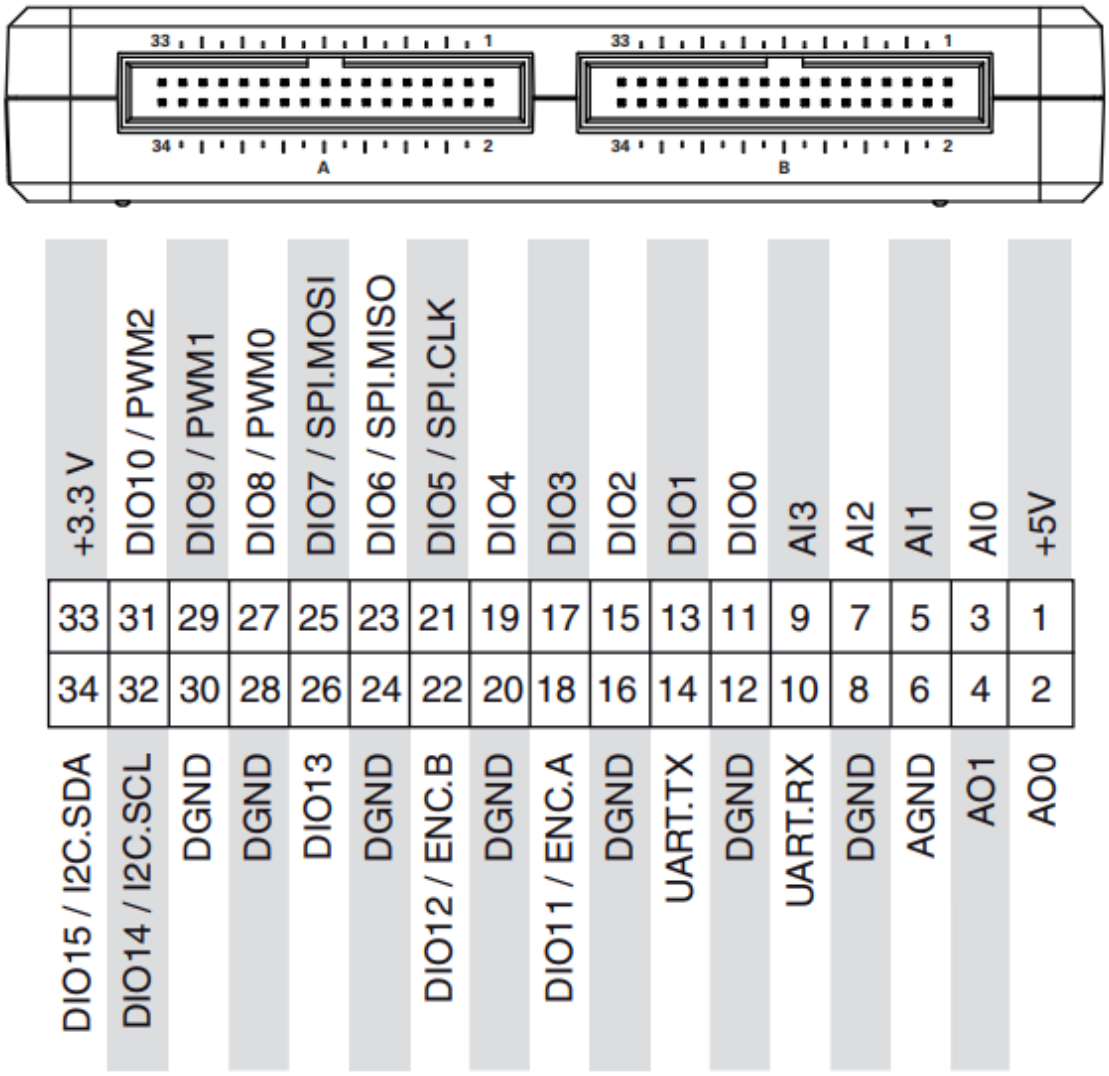

Figure 4.1: Primary/Secondary Signals on MXP Connectors A and B

The MXP connectors in Figure 4.1 each offer analog inputs (AI), analog outputs (AO), and digital inputs and outputs (DIO). The UART-receive input (UART.RX) and the UART-transmit output (UART.TX) on the connectors are electrically identical to the DIO lines. The connectors each offer a +5V and a +3.3V power output and ground references for both the digital and analog signals.

4.1.2 Compass

The Digilent PMOD CMPS1 compass utilizes a Honeywell HMC5883L 3-axis digital compass reading to the Digilent host board with an I^2C interface. The compass offers a 2 milli-gauss field resolution in ±8 gauss fields and uses Anisotropic Magnetoresistive (AMR) technology to allow for minimal interference between the three individual coordinate-dependant sensors.[13] The compass configuration with the myRIO's MXP connector A is shown in Figure 4.2.

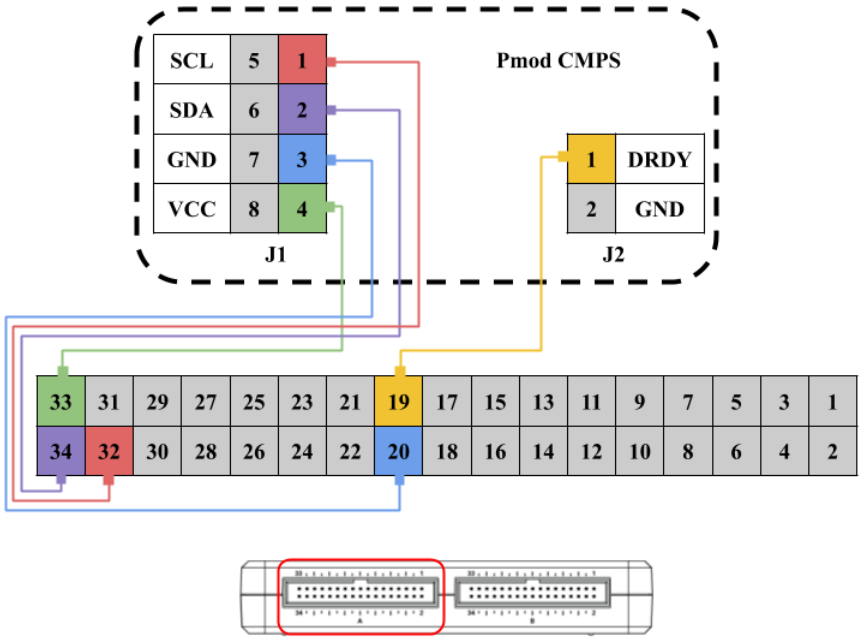

Figure 4.2: PMOD CMPS configuration with MXP connector A

In reference to Figures 4.1 and 4.2, the compass was powered by a +3.3V power output (pin 33) and referenced with the digital ground (pin 20). The compass uses I2C via a serial clock (SCL) line and a serial data (SDA) line to communicate with the myRIO MXP connector A's pins 34 and 32 utilizing the pin's secondary signals I2C.SDA and I2C.SCL, respectively.[13]

This research used the compass configuration and calibration routine designed by Steven Padgett [10]. The compass can be calibrated by the operator toggling a Boolean switch to the True state on the operator's laptop and manually driving the

UGV in one full circular rotation aiming for a minimum radius of 2 meters for optimal performance. Once the calibration routine is complete and the operator has toggled the Boolean switch to the False state, a loop within the UGV's system calculates the average and range of each axis to center the values on zero and normalize them to ± 1. The designed system then uses the adjusted values and inverse tangent function to calculate the orientation within the cardinal directions assuming $0°$ as north, $90°$ as east, $-90°$ as west, and $\pm 180°$ as south. To remove outliers from the system, five values were recorded and averaged, then validation checked by comparing each average to the previous two for equality, either triggering a flag for invalid data or continuing to record the data as the orientation of the UGV. [10]

4.1.3 Sonic Range Finder

The LV-MaxSonar-EZ1 MB1010 SRF offers a free-running operation at a 20-Hz rate and can detect objects from 6 to 254 inches away. The purpose of the SRF on the UGV is to detect drop-offs in front of the robot and prevent it from driving off a ledge greater than 4 inches.

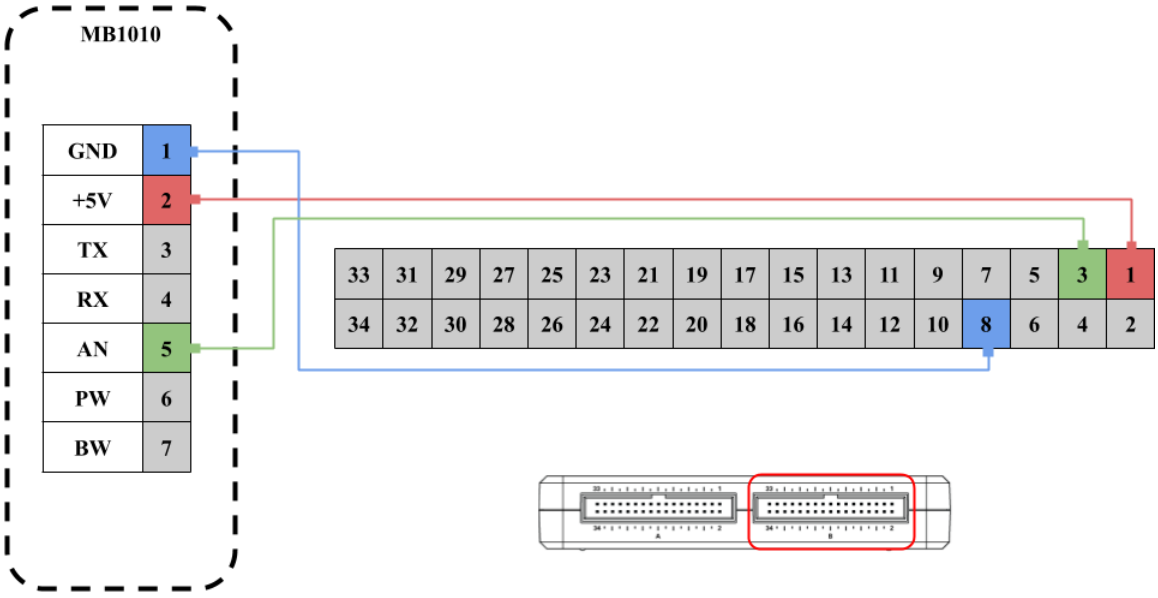

Figure 4.3: MB1010 SRF configuration with MXP connector B

As shown in Figure 4.3, the SRF was configured to use the myRIO MXP connector B, which provided a +5V power supply (pin1), analog ground reference (pin 8), and an AIO (pin 3). Since the SRF was receiving a supply of 5V, it was able to yield approximately 9.8mV/in. The conversion of the analog input Volts to inches was determined by dividing the analog input Voltage by 0.0098V. After converting, the system averaged the two most recent inputs and then compared the previous two averages for distances greater than 24 inches. The SRF was mounted on the UGV towards the front and angled downward at 35° below the horizon. With this method, the UGV was able to detect drop-offs 18 to 20 inches from the front of the robot while traveling. If a drop-off was encountered, the emergency stop function was triggered immediately, reducing all throttle to zero.

4.1.4 GPS

The GlobalSat BU-353-S4 GPS module was connected to the myRIO via a USB hub allowing for serial communication using standard NMEA sentences of data. The GPS operated with a 1 Hz update rate and a -163 dBm tracking sensitivity providing $<$2.5 m accuracy. The myRIO used a loop designed by Steven Padgett to concatenate bytes read from the GPS until the end of a sentence was read. The loop matched each sentence with the GPS fix information (GPGGA) or the recommended minimum data for GPS (GPRMC) to select an appropriate case statement. For the GPRMC data set, the following was used:

- GPS status (valid or not)
- Speed (in knots)
- Course (in degrees from north)

The GPGGA case statement read the data found below:

- UTC (time)

- Longitude (degrees)
- Latitude (degrees)
- Directions (north or south and east or west)
- Number of satellites found
- Altitude (in meters)

Both case statements took the strings and converted them into double-precision numbers to allow for calculations with the exceptions of GPS status, UTC, and directions. One such calculation took the speed and multiplied it by 0.5144 to convert knots into meters per second. [10]

4.1.5 LIDAR

The SLAMTEC RPLIDAR A3M1 offers an outdoor object detection range up to 20 m, a scanning frequency of 10Hz(600RPM), a sampling rate of 10kHz, and an angular resolution of 0.54°. The A3M1 model only offers a communication speed of 256000 baud through a TTL UART interface, and since the myRIO-1900 is only equipped with USB 2.0 ports capable of 115200 baud, a laptop with a USB port capable of 256000 baud located on the UGV became a necessity. The laptop provided the 5V power supply and the USB 3.0 port and was able to communicate through the RPLIDAR USB Adapter.

4.2 Autonomous Navigation

The autonomous navigation system required a sequence of setup operations, controlled by the user via the operator's laptop, to be performed before the autonomy could begin. The operations included the initialization of the LIDAR, the compass calibration routine, the "Home" or starting location of the UGV to be recorded, and the destination waypoint array to be sent to the myRIO. Once all beginning operations were complete, the user switched the drive control from *Manual* to *Travel*

To thereby initiating the autonomous navigation system. The following subsections cover the methodology used in obstacle detection, waypoint navigation, motor controls, obstacle avoidance algorithms, and intermittent GPS operations.

4.2.1 Obstacle Detection

By modifying the C++ program from the SLAMTEC Software Development Kit (SDK), the LIDAR was configured to output a line of information to the Windows Console in the form of the 8-character string shown below:

$$AAAQDDDD \tag{4.1}$$

Where AAA is a decimal floating-point of the angle $0° - 360°$ rounded up to the nearest integer, Q is the quality check-bit (that defaulted to 0 and only read as a 1 if there was corruption within the data), and $DDDD$ is a decimal floating-point of the distance measured at the angle AAA in units of millimeters, rounded up to the nearest integer. Once the Console program was running, LabVIEW read each line and converted it into an ASCII decimal byte array. The array had 48 subtracted from each element to transform the ASCII decimal code into the printable characters 0-9. Each character was then assigned a variable based on the character's respective index value within the byte array for later processing, as shown in Table 4.1 below:

Table 4.1: Assigned Variables to String Characters

String Character	Assigned Variable
A	$a3$
A	$a2$
A	$a1$
Q	NA
D	$d4$
D	$d3$
D	$d2$
D	$d1$

The elements with assigned variables were then fed into the following equations:

$$\theta = (a3 \times 100) + (a2 \times 10) + a1 \tag{4.2}$$

$$r = ((d4 \times 1000) + (d3 \times 100) + (d2 \times 10) + d1) * 0.001 \tag{4.3}$$

Where θ is the angle (in degrees) and r is the magnitude of the vector at θ in meters. Q's character value was either a 1 or 0; any strings including a 1 were flagged as corrupted data and were discarded. θ and r were inserted into an array until $\theta = 360°$, at which point all buffered data was discarded, and the array was enqueued to be processed in another loop.

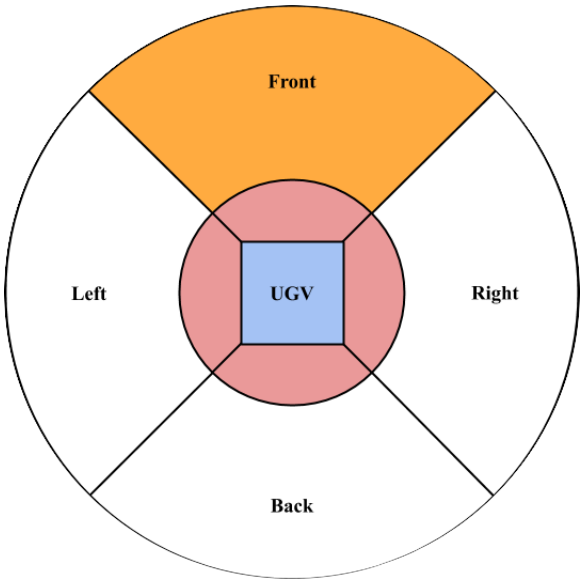

Figure 4.4: LIDAR data sections

The dequeue loop sorted the $r\angle\theta°$ vectors within the array into four quadrants as seen in Figure 4.4 and discarded any vector with $r < 0.45m$ to accommodate obstructions caused by the UGV internal structure denoted as the red circle in the figure. Each quadrant array was then scanned for the vector with the lowest r-value for that quadrant's closest object. The value of the closest r for each quadrant was then sent every 50 ms from the onboard laptop to the myRIO via NI-PSP variables over Ethernet, resulting in the myRIO system variables F_{dist}, R_{dist}, L_{dist}, B_{dist} for the front, right, left, and back quadrants respectively. The front quadrant array was then sent to a separate loop for further processing within the obstacle avoidance algorithm.

The front quadrant array was processed in a loop to determine if there was an apparently approaching obstacle and, if so, the best course of action to avoid said obstacle would be. This process began with each $r\angle\theta°$ vector being checked through the following statements:

$$|sin(\theta) * r| < 0.5 \tag{4.4}$$

$$|cos(\theta) * r| < 2.5 \tag{4.5}$$

Equation 4.4 checked the vector for being located within ±0.5 meters of the front-facing central axis, while Equation 4.5 checked the vector for being located within 2.5 meters in front of the UGV. If either statement was false, then the vector was discarded as the point did not lie within the UGV's current path or was too far away to be considered an obstacle. In the case that both statements were true, then the system loop signaled that there was an obstacle in the path and fed the vector into the equation shown below:

$$tan^{-1}\left(\frac{sin(\theta) * r \pm 0.4}{cos(\theta) * r - 0.4}\right) \tag{4.6}$$

In Equation 4.6, the vectors were broken down into X and Y components then adjusted to compensate for the depth and width of the UGV before using the inverse tangent function to determine the corrected angle. The adjustment of the X component of the vector added or subtracted 0.4 depending on which side of the front-facing central axis the initial vector fell on. The loop then compared the new adjusted angle to the previously recorded angles and assigned it to either θ_{min} or θ_{max} with θ_{max} being a positive integer and θ_{min} negative, corresponding to the left $(0° - 45°)$ or right $(0° -^{-} 45°)$ side of the front-facing central axis respectively. The values of θ_{min} and θ_{max} were then sent to the myRIO via NI-PSP variables to assist in the obstacle avoidance algorithm.

4.2.2 Motor Controls

Two algorithms managed the motor controls of the UGV. The first algorithm adjusted the throttle, depending on the direction of travel, by reading the NI-PSP shared variables from the LIDAR. While there were no obstacles present, the maximum throttle was set to 30% of the CIM motors' maximum duty cycle. The standard 30% duty cycle was determined through trial as the maximum operating speed of the UGV while still allowing for optimal object recognition from the LIDAR system.

If the LIDAR system detected an object, then the maximum throttle (T_{max}) was calculated using the equation below:

$$T_{max} = \left(\frac{X_{dist} - 0.4}{1.6}\right) * 30 \tag{4.7}$$

Where X_{dist} was either the F_{dist} or B_{dist} from the LIDAR obstacle detection system depending on the direction of the current throttle, forward or backward.

The second motor control algorithm took the forward and rotational throttle values from either the manual control or the autonomous navigation system and transformed them into duty cycles for the CIM motors. This algorithm smoothed the throttle change by only allowing a 2% change for the forward throttle and a 1% change for the rotational throttle every 20 ms. If an object detected on either side of the UGV by L_{dist} or R_{dist} was within 0.45m of the UGV, then the rotational throttle was immediately set to and held at zero. The same was true for the forward throttle and F_{dist} or B_{dist}, depending on the input throttle's direction. Additionally, an emergency stop option was available that would bypass the deceleration algorithms and reduce all throttles to zero if the user toggled it or if the UGV was unable to ping the operator's laptop at least once per second.

With the maximum throttle, forward throttle, and rotational throttle percentages input through the system, the motors' duty cycles were then set accordingly. The motor drivers operated at a 50 Hz frequency with a pulse width of 1 ms. Due to the motors' orientation, a positive rotational throttle resulted in a clockwise turn of the UGV. By adding the rotational throttle to the forward throttle of the right motor set and subtracting the rotational throttle from the forward throttle of the left motor set, the rotational and forward throttles of both sets were adjusted into two independent throttles, each relative to its respective side. The adjusted throttle value of each side was then divided by 4000 and coerced between 0.05 and 0.1 to determine the final

duty cycle of each motor set. This method allowed the UGV to be able to make precise turns while maintaining forward movement.

4.2.3 Waypoint Navigation

Every 10-millisecond iteration of the autonomous navigation system loop used equations derived from the haversine formula shown below to calculate the shortest distance over the Earth's surface between the UGV's current GPS position and the GPS coordinates of the next waypoint.[14]

$$a = sin^2\left(\frac{\triangle\phi}{2}\right) + cos(\phi_1) * cos(\phi_2) * sin^2\left(\frac{\triangle\lambda}{2}\right) \tag{4.8}$$

$$c = 2 * atan^2\left(\frac{\sqrt{a}}{\sqrt{1-a}}\right) \tag{4.9}$$

$$d = R * c \tag{4.10}$$

Where ϕ is latitude, λ is longitude, and R is the Earth's mean radius of 6,351 km. An approximation based on the haversine formula within the testing location was made to reduce the computationally taxing calculations resulting in a simplified linear relation between latitude, longitude, and distance. The approximation determined that at the testing location, the distance between each degree of latitude was 111.195 km, and the distance between each degree of longitude was 90.745 km.[10] With this approximation, the autonomous navigation system was able to calculate the latitudinal error (ϕ_e) and the longitudinal error (λ_e), in meters, using the equations shown below:

$$\phi_e = 111195(\phi_{wp} - \phi) \tag{4.11}$$

$$\lambda_e = 90745(\lambda_{wp} - \lambda) \tag{4.12}$$

Where ϕ_{wp} and λ_{wp} are the latitude and longitude of the next waypoint and ϕ and λ are the UGV's current latitude and longitude. The total distance error (D_e) was

then calculated using the Pythagorean theorem as shown below:

$$D_e = \sqrt{\phi_e^2 + \lambda_e^2} \tag{4.13}$$

The total distance error was then multiplied by 5 and coerced between the maximum and minimum forward throttle determined by the motor control system resulting in the forward throttle. If the total distance error was less than 2 meters, then a flag was thrown, and the system considered successful arrival at the target waypoint. The angle correction (θ_{cr}) for the UGV was also calculated every 10 ms using the equation below

$$\theta_{cr} = tan^{-1}\left(\frac{\lambda_e}{\phi_e}\right) - \theta_{cmps} \tag{4.14}$$

Where θ_{cmps} was the compass-derived orientation. θ_{cr} was then converted into rotational throttle (T_{rot}) by first dividing by three then coercing the value through the following formula based on the previously determined forward throttle (T_{fwd})

$$T_{rot} = \pm(3\sqrt{T_{fwd}}) \tag{4.15}$$

Where $\pm$ represents the upper and lower limits to the coercion, with the lower limit being the inverse of the upper. The previous calculations were only active while there was no detected obstacle in front of the UGV

4.2.4 Obstacle Avoidance

After the forward throttle's preliminary calculation, any adjustment to the throttle came from either the motor control system setting the maximum throttle percentage based on the F_{dist} or B_{dist} as seen in equation 4.7, or the obstacle avoidance algorithm. While the forward throttle was not equal to zero and an obstacle was detected in front of the UGV, the obstacle avoidance algorithm shown below would adjust the rotational throttle based on the θ_{min} or θ_{max} from the LIDAR obstacle detection

system.

$$T_{rot} = 2(T_{fwd} \pm 20) * tan^{-1}(\Theta) \tag{4.16}$$

Where the greater between the absolute values of θ_{min} and θ_{max} determined Θ and whether 20 was added or subtracted from T_{fwd} with θ_{max} corresponding to addition. This method allowed the UGV to make quick adjustments in direction without overshooting the corrected distance since the UGV slowed down as it approached an object.

4.2.5 Intermittent GPS

The intermittent GPS masking loop ran as a subsidiary function within the waypoint navigation system loop. While the GPS was active, the UGV passed and updated the last recorded speed, latitude, latitude, and total distance error through each loop iteration. By passing these values through each cycle, the UGV would have the last known values to proceed with its prediction algorithm if the GPS signal was lost.

The intermittent GPS loop's prediction algorithm was triggered by a Boolean switch with the case statement of GPS signal being true or false. Once the loop began, the absolute value of the compass orientation (θ_{cmps}) was converted to radians and then passed through a case statement consisting of eight sections based on the original compass orientation, as shown in Figure 4.5 below

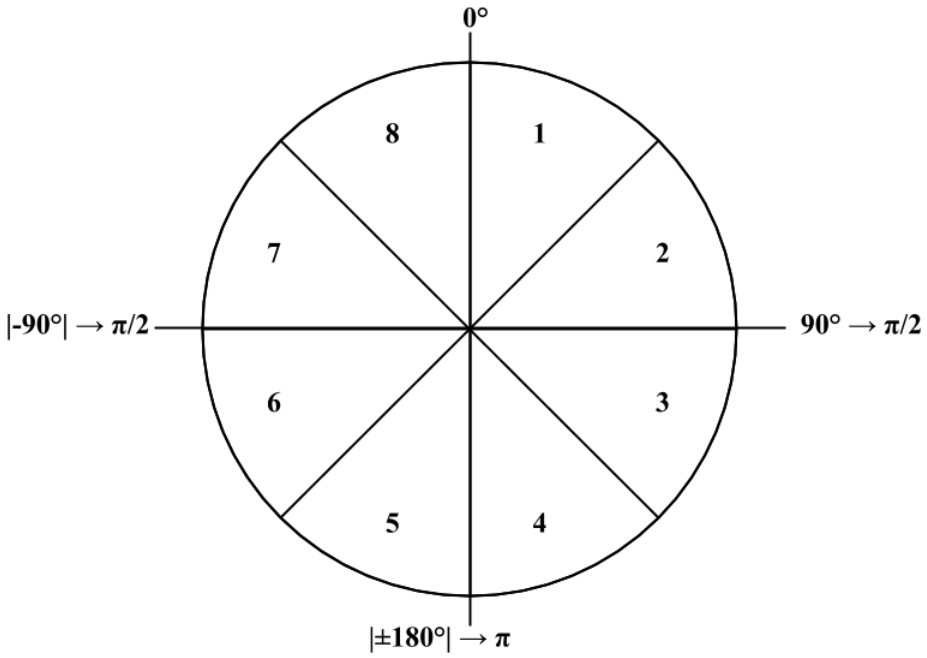

Figure 4.5: Compass Orientation Sections

The equations applied to each section (1-8) in Figure 4.5 are outlined in Table 4.2

Table 4.2: Applied Equations for Each Orientation Section

Orientation Section	Applied Equations
1	$\phi_{prd} = \phi_{lk} + \left(\frac{cos(\theta_{cmps}) * D_{prd}}{111195}\right)$ $\lambda_{prd} = \lambda_{lk} + \left(\frac{sin(\theta_{cmps}) * D_{prd}}{90745}\right)$
2	$\phi_{prd} = \phi_{lk} + \left(\frac{cos(\theta_{cmps}) * D_{prd}}{111195}\right)$ $\lambda_{prd} = \lambda_{lk} + \left(\frac{sin(\theta_{cmps}) * D_{prd}}{90745}\right)$
3	$\phi_{prd} = \phi_{lk} - \left(\frac{sin(\theta_{cmps} - \frac{\pi}{2}) * D_{prd}}{111195}\right)$ $\lambda_{prd} = \lambda_{lk} + \left(\frac{cos(\theta_{cmps} - \frac{\pi}{2}) * D_{prd}}{90745}\right)$
4	$\phi_{prd} = \phi_{lk} - \left(\frac{cos(\pi - \theta_{cmps}) * D_{prd}}{111195}\right)$ $\lambda_{prd} = \lambda_{lk} + \left(\frac{sin(\pi - \theta_{cmps}) * D_{prd}}{90745}\right)$
5	$\phi_{prd} = \phi_{lk} - \left(\frac{cos(\pi - \theta_{cmps}) * D_{prd}}{111195}\right)$ $\lambda_{prd} = \lambda_{lk} - \left(\frac{sin(\pi - \theta_{cmps}) * D_{prd}}{90745}\right)$
6	$\phi_{prd} = \phi_{lk} - \left(\frac{sin(\theta_{cmps} - \frac{\pi}{2}) * D_{prd}}{111195}\right)$ $\lambda_{prd} = \lambda_{lk} - \left(\frac{cos(\theta_{cmps} - \frac{\pi}{2}) * D_{prd}}{90745}\right)$
7	$\phi_{prd} = \phi_{lk} + \left(\frac{cos(\theta_{cmps}) * D_{prd}}{111195}\right)$ $\lambda_{prd} = \lambda_{lk} - \left(\frac{sin(\theta_{cmps}) * D_{prd}}{90745}\right)$
8	$\phi_{prd} = \phi_{lk} + \left(\frac{cos(\theta_{cmps}) * D_{prd}}{111195}\right)$ $\lambda_{prd} = \lambda_{lk} - \left(\frac{sin(\theta_{cmps}) * D_{prd}}{90745}\right)$

Where ϕ_{prd} and λ_{prd} were the predicted latitude and longitude that proceeded to act as substitutes for ϕ and λ in Equations 4.11 and 4.12. The variables ϕ_{lk} and λ_{lk} were the last known latitude and latitude which were recorded on each loop iteration with GPS signal on such that on the first loop iteration without GPS signal, the values were recycled from the previous iteration and then updated for the next iteration by the calculations for ϕ_{prd} and λ_{prd}. The distance predicted (D_{prd}) was calculated by multiplying the last recorded speed by 0.01 to estimate the distance traveled in meters within the last 10 milliseconds.

After the calculations in Table 4.2, the predicted latitude and latitude were processed through the same logic and formulas as their real counterparts would if GPS signal was enabled, as discussed in Chapter 4.2.3. During GPS downtime, the waypoint navigation system mirrored that of the navigation system with a signal, the exception being how the current latitude and longitude were obtained and that the total distance error (D_e) calculated in Equation 4.13 was recorded as a "predicted error." This predicted error still determined the forward throttle (T_{fwd}) used throughout the motor control system; however, it was also used to predict the UGV's speed for calculating the predicted distance traveled (D_{prd}) used in Table 4.2.

Predicting the speed of the UGV during GPS intermittency required a model of the relationship between the UGV's speed and the distance to the next waypoint (total distance error). A linear model was chosen to represent the relationship due to the throttle being linearly proportional to the distance to the next waypoint. The model was derived by first recording the speed and distance over eight test runs. These runs were then filtered to remove any recordings of a distance of over 6 meters since the highest allowed maximum throttle was 30 percent. Additionally, any recording of distance below 3.25 meters was also discarded as the time spent during periods where the rover was checking to see if it was within the 2-meter acceptable radius from the waypoint significantly outweighed that of the actual travel time between waypoints. With the applied filters, the eight test runs were plotted, and a linear model was developed with a slope of 0.125 and an adjusted R-Squared value of 0.8261, as seen in Figure 4.6 below.

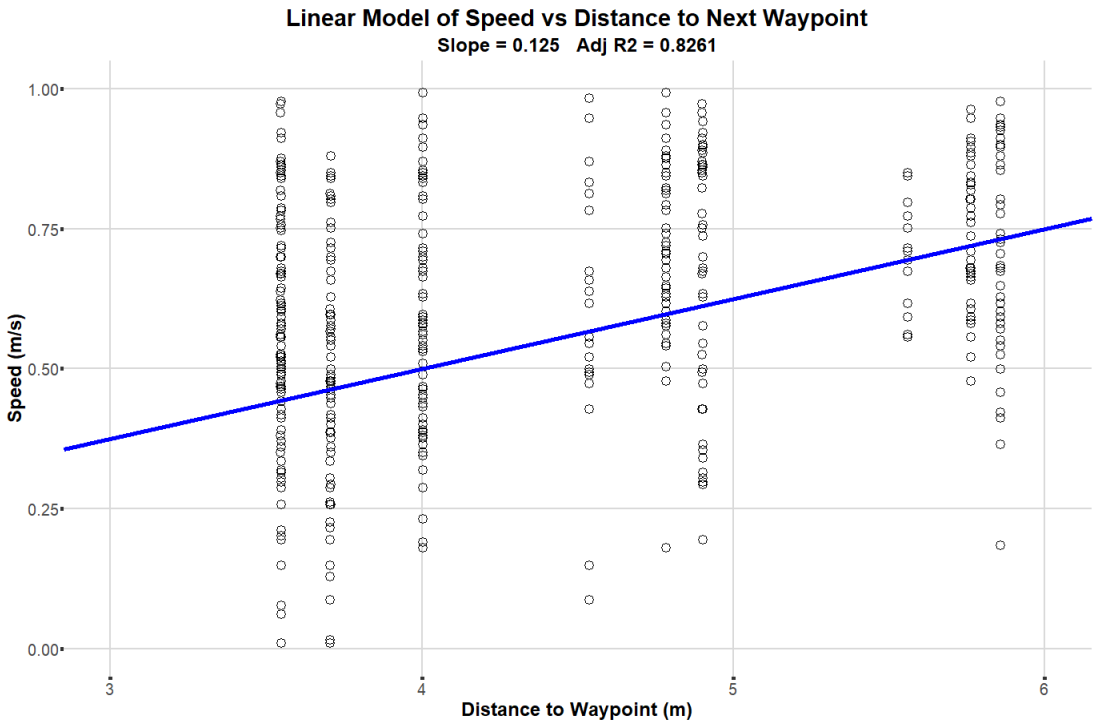

Figure 4.6: Linear Model of Speed vs. Distance to Next Waypoint

The linear model was then applied to the intermittent GPS navigation system to calculate the predicted speed using the equation below.

$$V_{prd} = \left(\frac{T_{fwd}}{5}\right) * 0.125 \tag{4.17}$$

Where V_{prd} was the predicted speed, and T_{fwd} was the forward throttle coerced through the limits set by the maximum throttle percentage.

4.3 Testing Parameters

Two methods of testing were implemented: one with and one without GPS intermittency; both were fully autonomous runs designed to analyze the waypoint navigation system and the obstacle avoidance algorithm.

4.3.1 Testing Location

In Figure 4.7 below, the locations of each waypoint are shown along with the ideal path between each point. The ideal path is the shortest distance from waypoint to waypoint with no regard for any obstacle or terrain that may lie between.

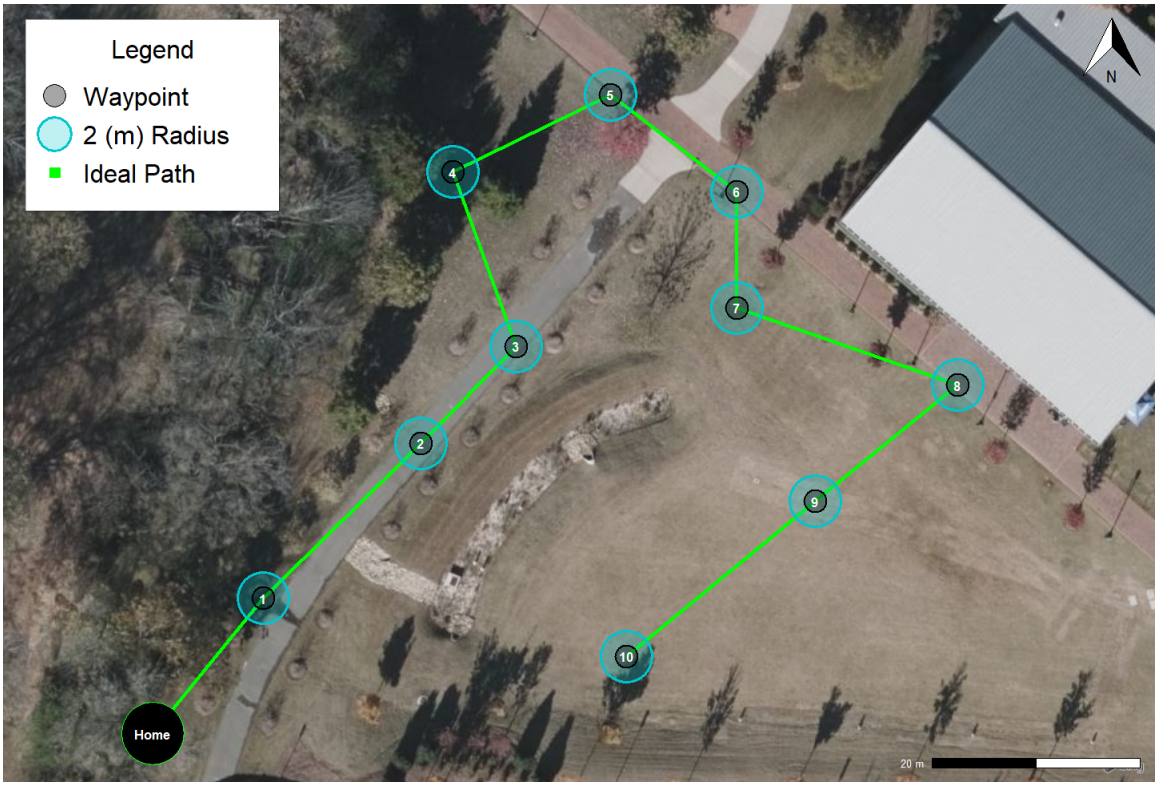

Figure 4.7: Testing Area with Waypoint Locations and Ideal Path

The area chosen provided ample distance between each waypoint with a moderate amount of potential obstacles, including trees, bushes, rocks, trash bins, and lamp posts. A general "U" shape was chosen for the test's overall path as the range of the operator's laptop WIFI connection to the UGV was limited and therefore required the operator to be located within a central location of the entire test. The distances between each waypoint and the cumulative distance of the UGV traveling along the ideal path are displayed in Table 4.3 below.

Table 4.3: GPS Waypoints Used in Testing

WP	Latitude	Longitude	Dist. To Next (m)	Total Dist. (m)
Home	35.30811667	-80.73781667	17	0
1	35.30823333	-80.73773333	21	17
2	35.30836667	-80.73756667	13	38
3	35.30845000	-80.73746667	18	51
4	35.30860000	-80.73753333	17	69
5	35.30866667	-80.73736667	15	86
6	35.30858333	-80.73723333	11	101
7	35.30848333	-80.73723333	23	112
8	35.30841667	-80.73700000	18	134
9	35.30831667	-80.73715000	24	152
10	35.30818333	-80.73735000	0	175

This testing layout resulted in an average distance between waypoints of 18 meters (58 ft) and a total distance traveled of 175 meters (575 ft) along the line of best fit shown in Figure 4.7. The nature of the chosen waypoint locations allowed for the UGV to encounter obstacle interference between several destinations and for the UGV to experience different terrains and elevation grades along its run. As the total distance in Table 4.3 is along the ideal path with no obstacle interference, it is expected that the actual distance traveled by the UGV during testing runs will exceed 175 meters.

4.3.2 Uninterrupted-GPS Autonomous Navigation

For the primary test of the UGV's autonomous capabilities, the longitude and latitude of the waypoints shown in Table 4.3 were sent to the myRIO's main system loop as an array. The main system would then feed a single row at a time to the

waypoint navigation system and would only advance to the next row or waypoint when the UGV's GPS had flagged that it was located within a 2-meter radius of the waypoint. The purpose of this test was to demonstrate the waypoint navigation and obstacle avoidance systems with expected results of the UGV to stay under an orthogonal distance of 5 meters from the ideal path between waypoints.

4.3.3 Intermittent-GPS Autonomous Navigation

The secondary test of the UGV's autonomous capabilities included the GPS intermittency system integrated within the waypoint navigation and obstacle avoidance systems. Since periods of GPS downtime are often unpredictable, a random number generator (RNG) was used to add a factor of indeterminacy to the testing. The RNG produced an integer ranging from 1 to 100 in 15-second intervals and compared it to a user-defined parameter determined at the beginning of the test run. The RNG value was directed to the argument of being greater than or equal to the user-defined parameter. The Boolean toggle resulting from this argument was assigned such that a "true" resulted in GPS being allowed for the following 15-second interval, while a "false" would mask the GPS from the UGV's navigation system and toggle the position prediction algorithm outlined in Chapter 4.2.5. This system allowed the operator to input the desired percentage of GPS downtime, and the RNG routine would attempt to match that.

A 15-second interval was chosen under the principle that if the UGV was traveling at the maximum allowed speed of 1 (m/s), then in the chosen interval, it would theoretically travel 15 meters, which, as seen in Table 4.7, would cover the majority of the average distance between waypoints. The theory behind this is that if the UGV lost the GPS signal after reaching a waypoint, it should be able to reach the next waypoint without a signal. The user-defined variables for testing were 25, 50, 65, and 75, referring to the desired percent of GPS masked time during the run with one to three tests performed for each of the variables.

The intervals were designed such that each was an independent variable. This allowed for an additional factor of indeterminacy in both the overall length of intervals with repeated Boolean values and the total amount of GPS-down intervals within a single run. Figure 4.8 below provides two plots with different probability functions for each of the user-defined variables dictating the intervals.

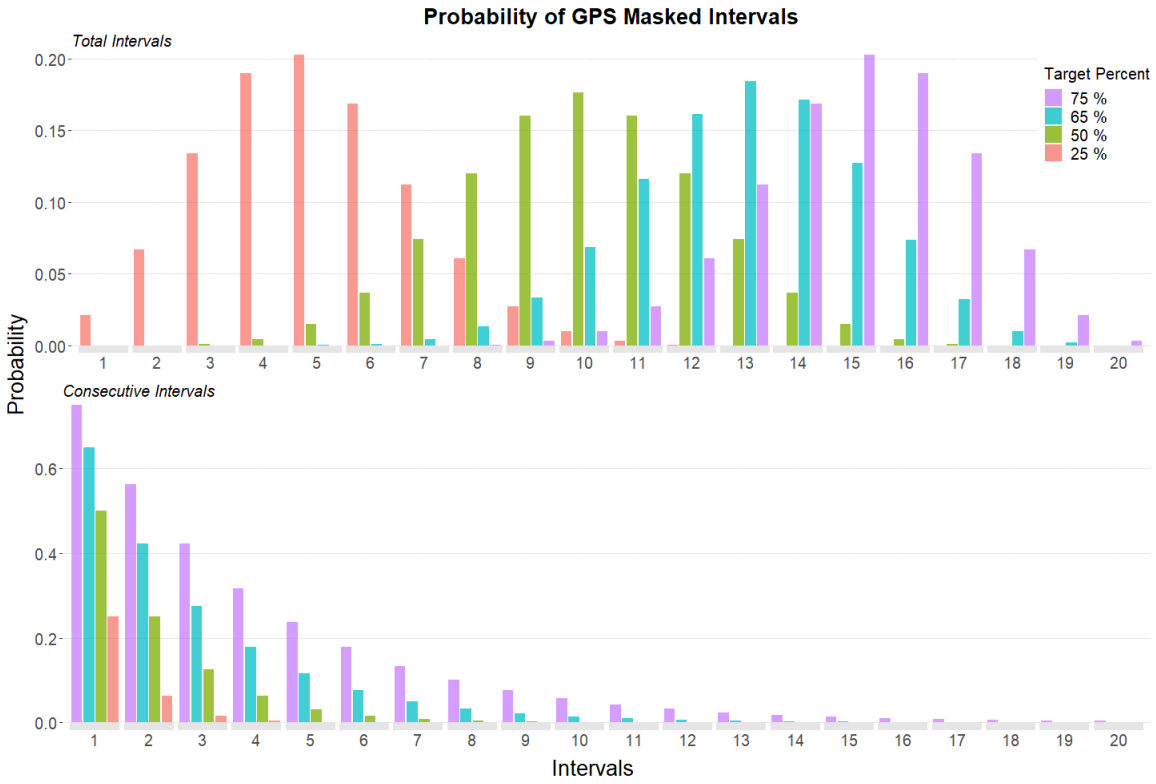

Figure 4.8: Masked GPS Interval Probability Distribution

The top plot for "Total Intervals" displays the probability of X total amount of 15-second intervals during a run. This was generated using a binomial distribution with a range of 20 independent observations. An observation pool of $n = 20$ was determined due to the average testing time being approximately 300 seconds, consisting of twenty 15-second intervals. The bottom plot for "Consecutive Intervals" provides the probability of the RNG repeating a "false" Boolean statement X amount of times consecutively. This was generated by taking the target percentile and raising it by the number of intervals.

4.4 Data Processing

To collect numerical data during testing runs, an algorithm was written on the myRIO such that it recorded local variables and compiled a 1D array every 50 milliseconds while the autonomous mode was activated. This array was shared with the operator's laptop via NI-PSP shared variables from which the array was concatenated onto a comma-separated values (CSV) file. The resulting delimited text file was saved and updated each iteration onto an 8 gigabyte USB drive connected to the operator's laptop.

Recorded data was later processed through the open-source integrated development environment (IDE), RStudio [15]. The software provided an IDE for the statistical programming language, R. Within RStudio, the primary libraries utilized were Tidyverse [16] for graphical statistical analysis and Rosm [17] for access to Bing Maps API and Open Street Map.

CHAPTER 5: RESULTS AND DISCUSSION

5.1 Data Validation

To properly visualize the test runs that took place, a mapping agent is required to plot the recorded latitude and longitude points of the UGV over the course of each test. The accuracy of this mapping agent is critical to the visual interpretation of the testing results as it displays whether or not the UGV was able to successfully reach all 10 waypoints sequentially. As previously mentioned, the Rosm package for RStudio was used for Bing Maps satellite-aerial imagery and Open Street Map for GPS coordinate plotting. According to Microsoft documentation on the resolution and scale of Bing Maps [18], using the Mercator projection of Earth for each tile allows the equation below to calculate the map's resolution in meters per pixel.

$$R = 156543.04 \left[\frac{meters}{pixel}\right] * \left(\frac{cos(\phi)}{2^z}\right) \tag{5.1}$$

Where R was the map's resolution in meters per pixel, ϕ was the median latitude in degrees, which in the case of the testing area was 35.3084°, and z was the chosen zoom level of 20. Using the developed test maps with an image resolution of 1440 x 971 pixels, the above equation determined the resolution to be 0.122 meters per pixel. Theoretical distances were calculated by measuring the pixels between several key landmarks visible in the generated test maps and the UGV's GPS coordinates. These were compared to the actual physically-measured distances to determine the accuracy of the plotted GPS coordinates at approximately ±1.5 meters, which is less than the UGV's onboard GPS module claim of within 2.5 meters; therefore, any inaccuracy in coordinate plotting was likely a result of the UGV's GPS module.

5.2 Uninterrupted-GPS Tests

Three tests were performed with the same waypoint coordinates as shown in Table 4.3 and the same ideal path as seen in Figure 4.7. The tests resulted in a 100% success rate of the waypoint navigation system guiding the UGV to within 2 meters of the desired waypoints in a sequential manner, as shown in Figure 5.1 below. The plotted point's opacity and size are to give visual aid to the number of observations at that point.

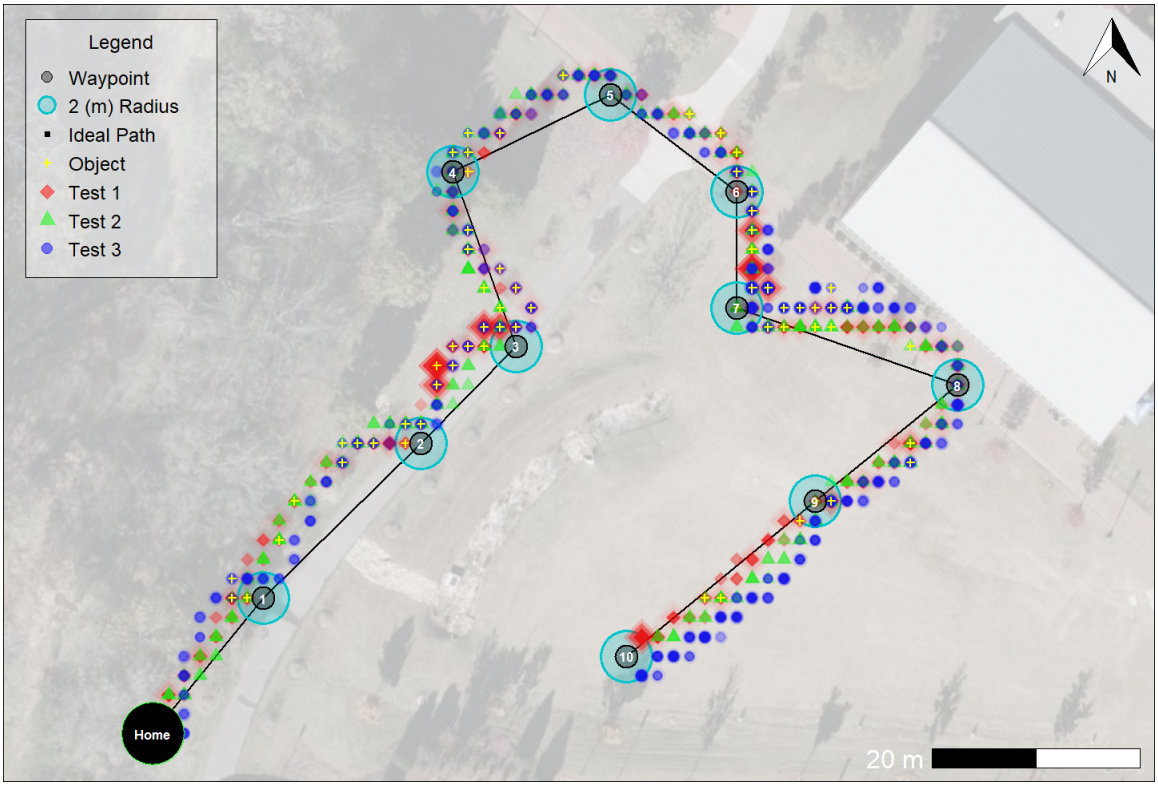

Figure 5.1: Combined Plot of Three Tests with Uninterrupted-GPS

Of the combined 56 obstacles encountered during all three tests, only two obstacles required manual interference resulting in an overall 96% success rate of the obstacle avoidance system with uninterrupted GPS. These two obstacles were encountered during Test 1 while the UGV was traveling between waypoints 2 and 3, denoted in Figure 5.1 by the larger, more vibrant red diamonds. The two obstacles were thin, twig-like shrubbery that the UGV's 2D-LIDAR could not identify as avoidable objects

due to the porous nature of the two-dimensional planer profile of the plants. This is one of the primary disadvantages of a two-dimensional LIDAR since everything is scanned on a single plane, the field of view for the LIDAR is often unable to recognize objects as a whole if there are significant gaps throughout the scanned plane of the object. For this scenario, one solution would be to incorporate a significantly more expensive three-dimensional LIDAR so that the multitude of small twig-like branches could be summarized together as one object and therefore avoidable. Another option would be to integrate the stereo vision-based method of obstacle avoidance commonly seen in modern autonomous cars.

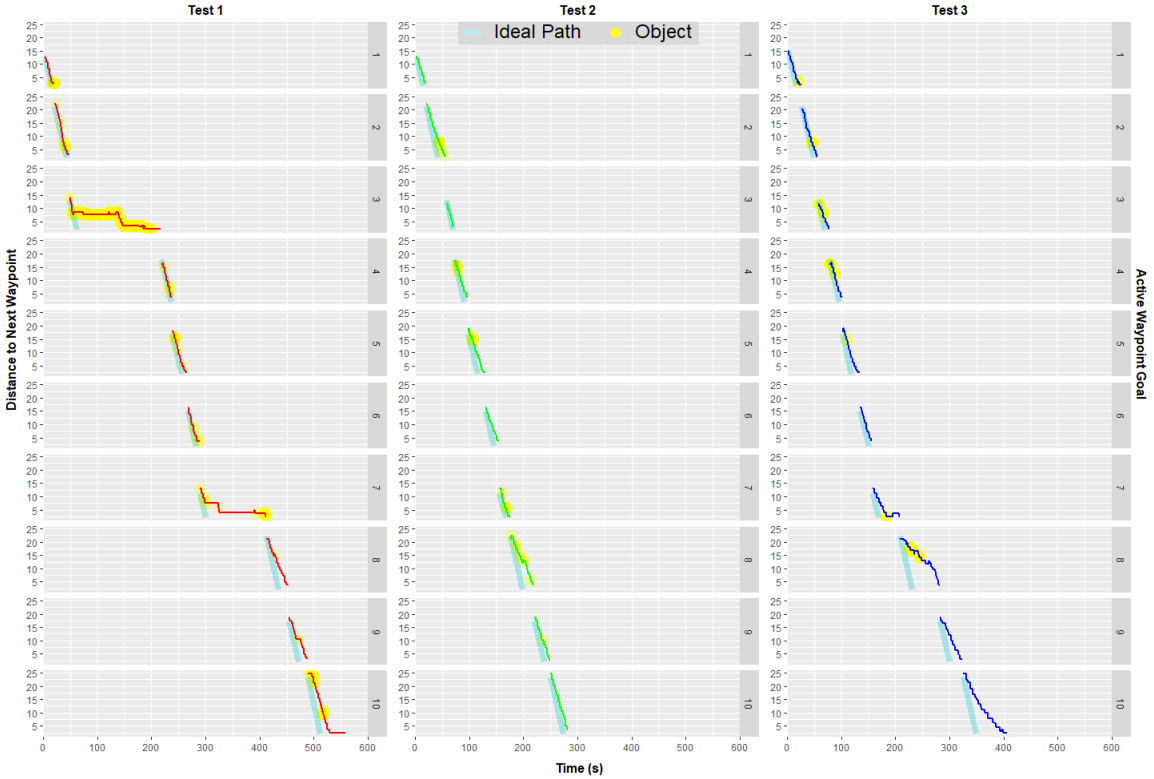

Figure 5.2: Actual Time vs. Ideal Case for Each Waypoint

Figure 5.2 above displays the time to completion for each of the three tests broken down by active waypoint goal and compared to the ideal case adjusted such that the comparison does not include compounding errors from previous segments. In the Figure, the two shrubs that were encountered in Test 1 can be seen as the UGV

headed towards waypoint 3, where there are two periods during which the UGV spent an extended amount of time not closing the distance between itself and the active waypoint goal. The only other time this observation is significant is during the approach to waypoint 7 within the same test run. This scenario was caused not by object interference; it was due to instead the UGV attempting to slow down so that it could check to see if it was within the 2-meter radius of the waypoint and doing so while on a steep incline resulting in there not being enough dedicated throttle to move forward. The encountered situation could be avoided by integrating a clinometer system that could measure the change in the tilt of the UGV's horizontal axis and provide accommodations to the forward throttle to compensate for the increase in slope. Another viable solution to the problem would be encoders on the wheels' axes; for this implementation, the UGV would need to be able to recognize the difference between the desired and the measured encoder count for the set speed and apply adjustments to the throttle accordingly.

For numerical analysis, each test's results of Figure 5.2 are summarized in Table 5.1 below, with the overall unit of measurement being seconds.

Table 5.1: Summary of Difference in Actual Time and Ideal Case

	Test 1	Test 2	Test 3
Max	156.15	20.87	58.91
Min	1.65	2.38	5.16
Mean	38.07	10.50	22.74
Std Dev	53.08	5.38	19.90

The combined average time difference between the ideal case and the actual for all three tests was 24 seconds with a standard deviation of 34 seconds. In Table 5.1, it is observable that when there were not substantial time delays such as the three in Test 1, the UGV managed to arrive at each waypoint less than 30 seconds behind the

ideal case on average. It should also be noted that the ideal case is considered always traveling at 1 meter per second, the maximum speed of the UGV. In reality, the UGV was required to slow down as it approached a waypoint to increase the accuracy of the onboard GPS unit; therefore, delays in the arrival time are expected.

The following three figures display the closest distance of the UGV from the ideal path per active waypoint goal for each of the three tests. These plots were generated using the function $dist2Line$ from the "geosphere" package in Rstudio, which uses the Haversine equation to calculate the shortest distance between geographical points and spacial lines [19]. The combined average distance for all three tests was 1.9 meters with a standard deviation of 0.75 meters. It is important to note that the UGV did not have the goal of maintaining the ideal case and therefore did not attempt to return to that path whenever it deviated. The purpose of the following plots is to show that with real-world interference and obstacle avoidance, the UGV was able to repeatably make autonomous decisions towards successfully navigating between each waypoint without deviating more than two meters off of the ideal course.

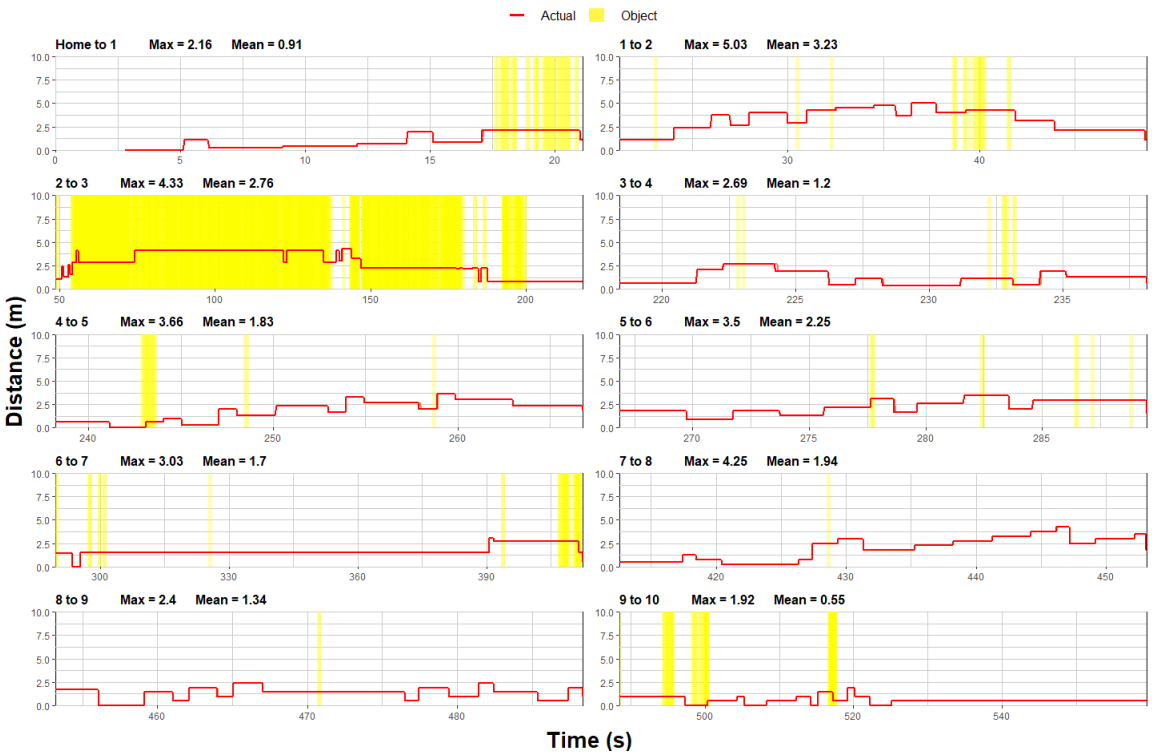

Figure 5.3: Test 1 Closest Distance to Ideal Case

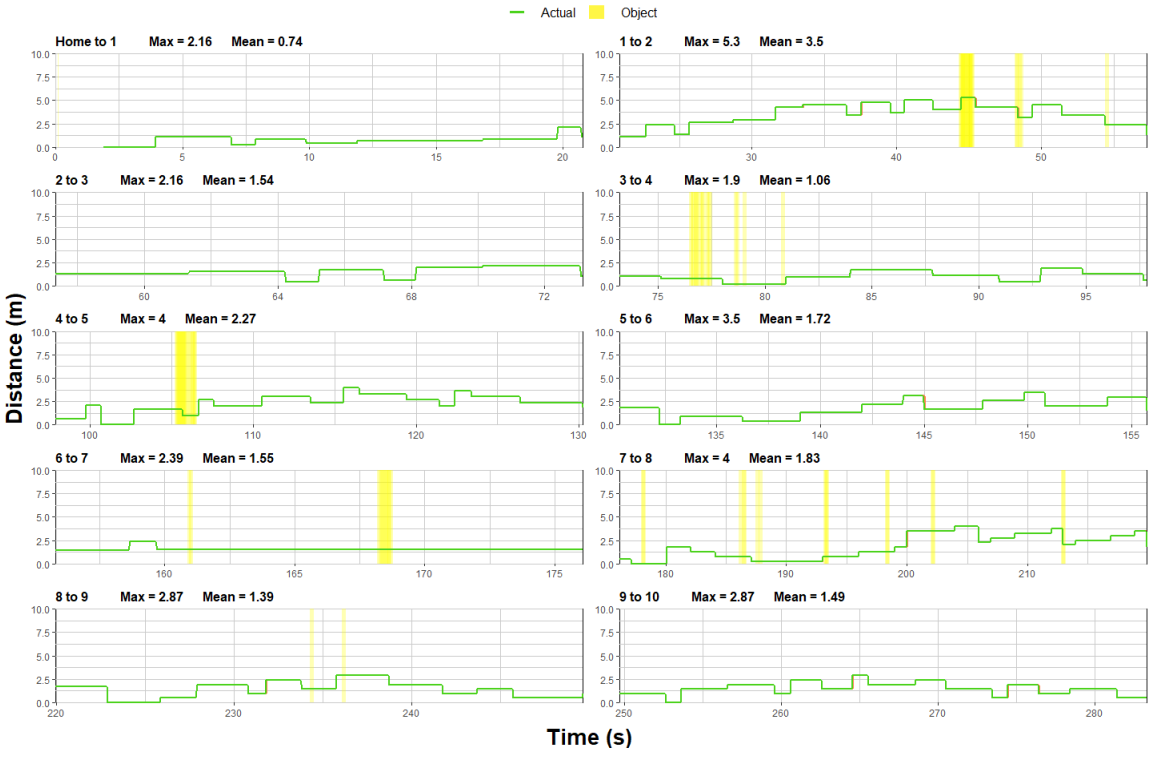

Figure 5.4: Test 2 Closest Distance to Ideal Case

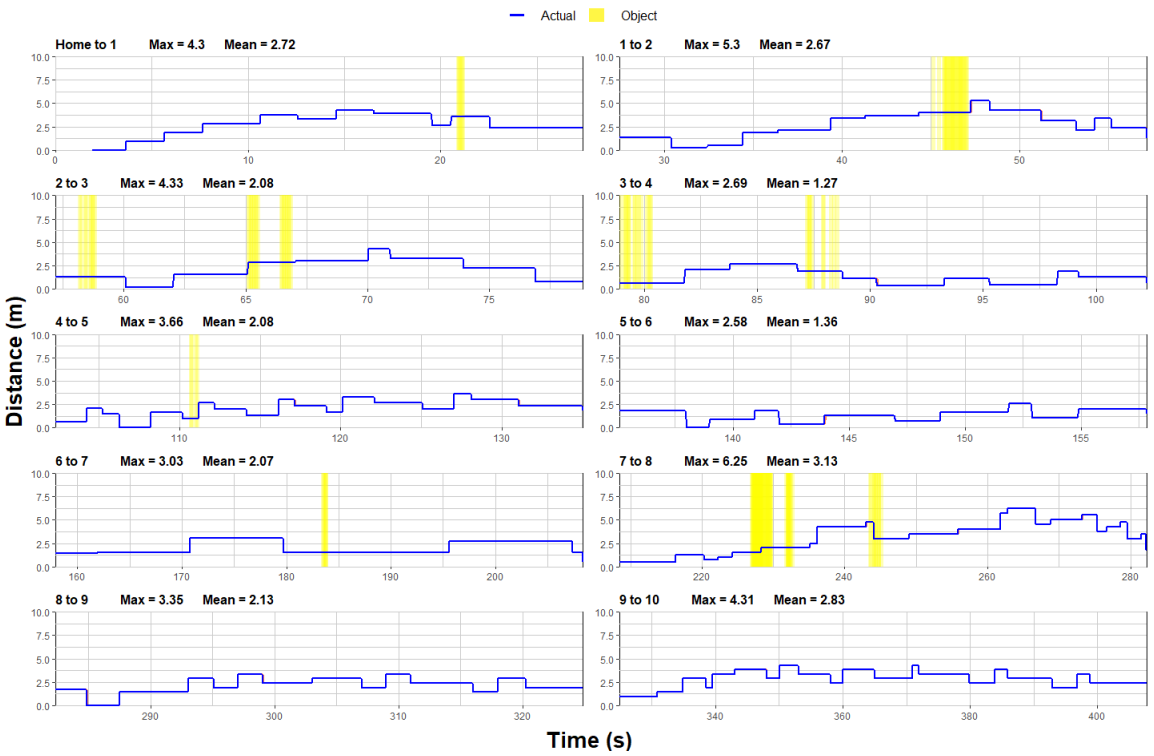

Figure 5.5: Test 3 Closest Distance to Ideal Case

In Figures 5.3, 5.4, and 5.5, it is observable that when an obstacle was encountered, except for the two shrubs seen in Test 1 between waypoints 2 and 3, the UGV would navigate around the obstacle without deviating more than a few meters off course and once clear it would maintain its heading towards the waypoint goal eventually reaching the 2-meter radius.

5.3 Intermittent-GPS Tests

There were a total of eight tests performed with GPS masked intervals. The target percent, number of tests performed, and actual percent of the tests are outlined in Table 5.2 below

Table 5.2: Summary of Intermittent GPS RNG Results

Target	Actual
25%	13.8%
50%	35.6%
	38.2%
	59.6%
65%	45.1%
	72.1%
75%	69.3%
	80.7%

As seen in Table 5.2 above, the RNG component implemented to randomly toggle the GPS masking did add variance and unpredictability to the testing. More tests were performed with a higher target percentage so that any issues encountered with the prediction system would theoretically be recurring.

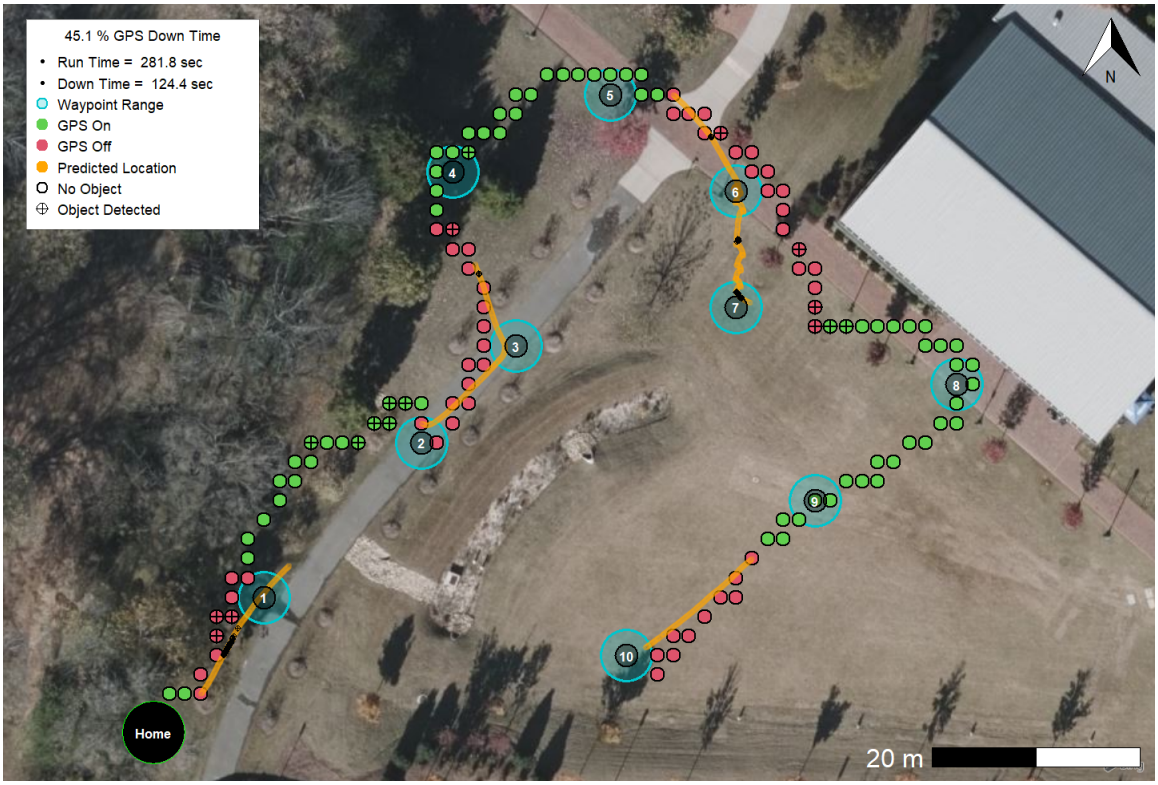

Figure 5.6: 45.1% Masked GPS Map

Figure 5.6 above displays the actual GPS position of the UGV and the predicted position of it during times of GPS masking for the test with a targeted percentage of 65% but an actual of 45.1% GPS downtime. Figure 5.7 below displays the actual GPS distance of the UGV and the predicted distance of it from the active waypoint goal for the same test. All maps and plots for the other seven tests can be found in Appendix A and are referenced throughout this discussion.

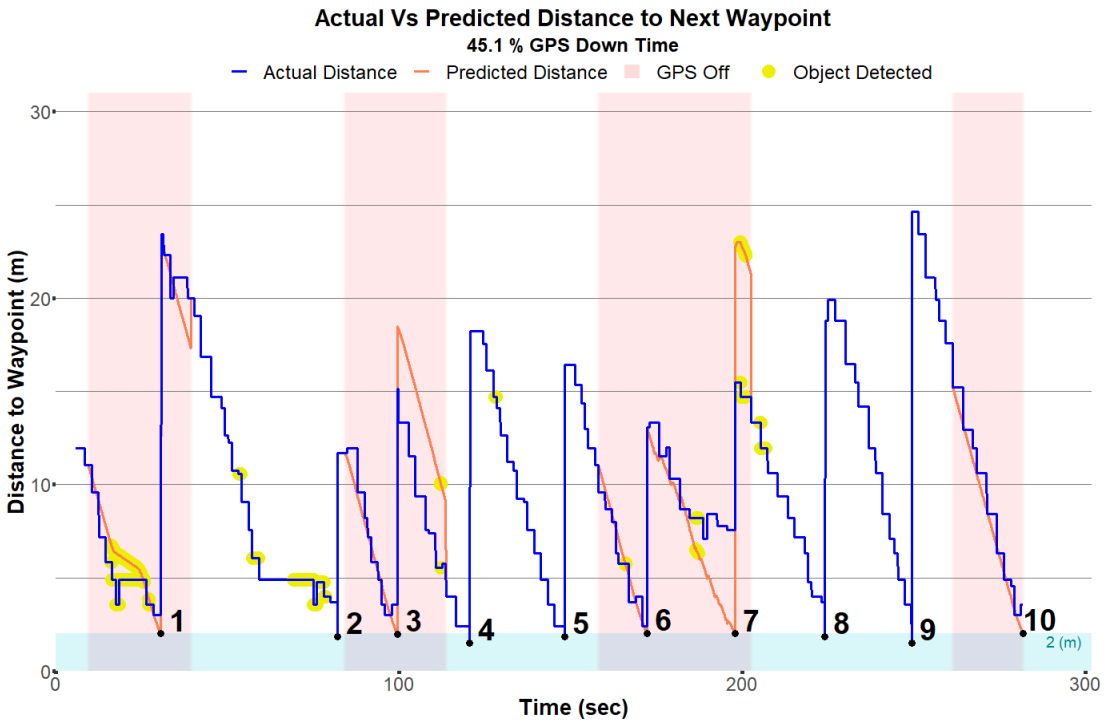

Figure 5.7: 45.1% Masked GPS Distance From Waypoint

During the eight intermittent GPS tests, the UGV's autonomous waypoint navigation system flagged that it had reached the 2-meter radius of its active waypoint goal a combined total of 50 times while GPS was masked. Of these 50 flags, the UGV was only within 2 meters of the active waypoint goal 18% of the time but within 5 meters of the goal for 66% of the flags. An example of these flags can be observed in Figure 5.7 when the orange predicted distance reaches the cyan 2-meter radius indicator while GPS is off during the approach to waypoints 1, 3, 6, 7, and 10. Most of the flags that were thrown with GPS masked and that were not within 5 meters of the goal waypoint were farther away due to compounding errors from previously encountered complications.

One such discovered issue occurred when obstacles were avoided along the path and not compensated for in the prediction algorithm. An example of this issue can be seen in Figure C.7; as the UGV approached waypoint 1 and it avoided an obstacle

by quickly moving west, but the prediction algorithm did not detect the abrupt temporary change in direction. This was likely due to the smoothing algorithm for reading the compass's inherently erratic behavior, as discussed in Section 4.1.2. This issue could be solved with a number of approaches such as using a compass with a higher degree of accuracy, integrating an algorithm to compare an encoder count of the wheels such that a significant difference would imply a change in direction, or by measuring the rotational change in acceleration via a gyro.

Another discovered issue was caused by the UGV not actually reaching the previous waypoint, and therefore the next segment began with an unaccounted-for offset. This issue can be seen in Figure 5.6 and 5.7, where the UGV approached waypoint 6 with GPS masked, flagged that it had reached its goal, and advanced to waypoint 7 under the assumption that it began the path to waypoint 7 from the GPS coordinates of 6, while in reality the UGV was approximately 7 meters northeast of waypoint 6.

There were 43 total obstacles encountered while GPS was actively masked. Twelve of these obstacles required manual interference either by manually unmasking the GPS signal or by physically moving the UGV; this resulted in a 72% success rate of the obstacle avoidance algorithm while GPS was masked. There are several reasons that the UGV had a notably more difficult time avoiding obstacles while the GPS was masked. One cause of this is that the UGV would be in a situation where the distance between the predicted and actual location of the UGV had grown too vast such as in Figure C.7 between waypoint 3 and 4. In this example, the GPS had to be manually unmasked so that the UGV would stop attempting to drive towards the thick underbrush to the west of waypoint 4. The UGV kept attempting to navigate further into the thicket due to the prediction algorithm being under the impression that it was much farther east near waypoint 3 than it was. This error was caused due to a combination of the two previously discussed issues in the prediction algorithm. One of which is unaccounted for obstacle avoidance and the other the compounding

error in the UGV not entirely reaching the previous waypoint with GPS masked, as can be seen in Figure C.14 between waypoints 1, 2, and 3. It should also be noted that the shown flag for reaching waypoint 3 in Figure C.14 appears to show the actual position at 2 meters from the waypoint, but the flag is actually connected to the predicted distance line, which reaches the 2-meter distance while the actual distance is 20 meters from the waypoint.

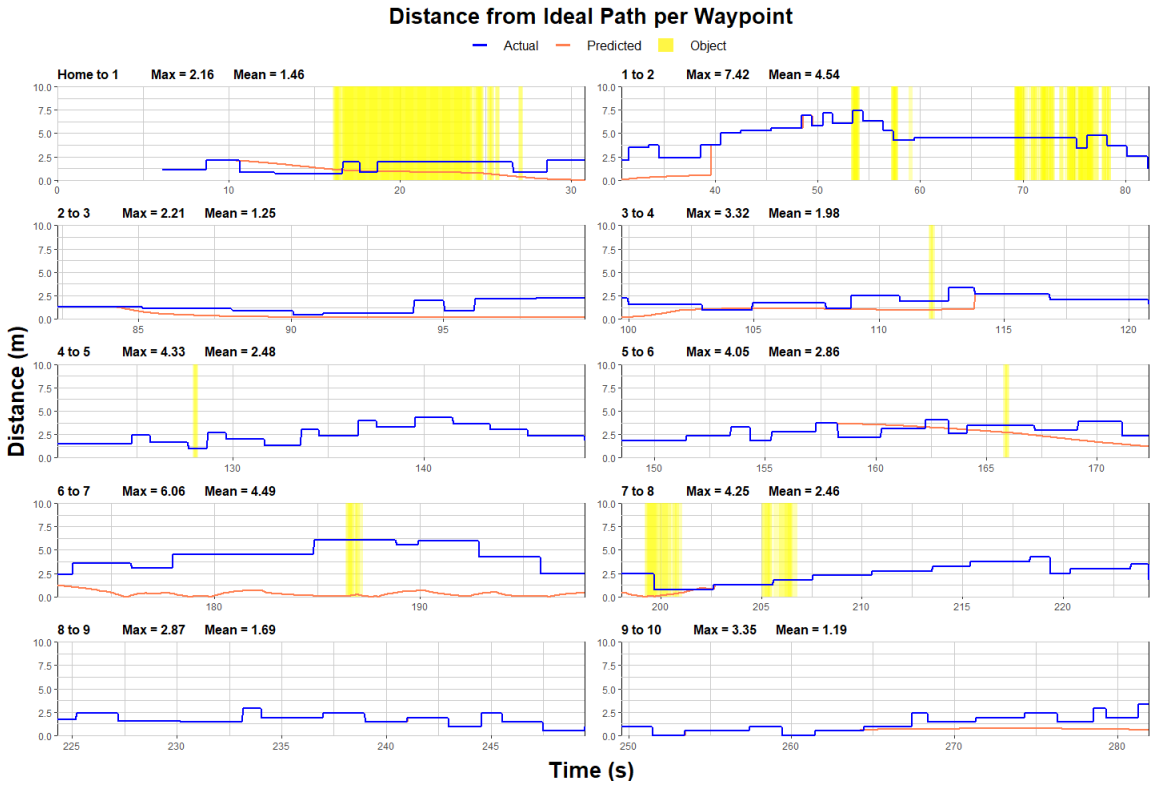

Figure 5.8: 45.1% Masked GPS Closest Distance to Ideal Case

Figure 5.8 above displays the closest distance of the actual and the predicted position of the UGV from the ideal path per active waypoint goal for the same test as the previous representative map and plot. A similarly generated plot can be found for each of the remaining tests in Appendix A.

It is important to note that in these plots, the function *dist2Line* does not account for the time when it takes the GPS coordinate and finds the closest distance of that point to the ideal case, which is modeled as a spatial line constructed by the GPS

coordinates of the waypoints. This unaccounted time is apparent when comparing the actual distance of the UGV from waypoint 7 when the flag is triggered, in Figure 5.7, and the distance from the ideal path at the end of the segment "6 to 7", in Figure 5.8. The latter shows 2.5 meters from the ideal path while the UGV was actually 7 meters away from the waypoint; this is due to the line drawn between waypoints 7 and 8 being closer to the UGV than the actual waypoint goal.

Exceptional cases such as the one just described can be seen in similar scenarios throughout these plots, but valuable observations can be made when using the $dist2Line$ plot in tandem with the previous plots and maps. For example, when comparing Figures 5.6, 5.7, and 5.8, it is apparent that the prediction algorithm manages to accurately calculate the corrections that would be necessary to maneuver the UGV towards the waypoint goal if the UGV were actually at the predicted location. Unfortunately, due to the encountered issues discussed previously in this chapter, the UGV was not always in close enough proximity to where it predicted it was located.

CHAPTER 6: CONCLUSIONS

6.1 Summary

As advancements in technologies offer increasingly versatile robotic platforms, the prospect of unmanned vehicle assistance in search and rescue operations continues to grow. This research produced a working proof-of-concept for autonomous vehicle navigation and obstacle avoidance along vectors predetermined by GPS waypoints with a system and procedure for potential intermittent GPS signal loss.

An obstacle detection and avoidance system was developed using a 2D-LIDAR that produced results with consistently high levels of success in an unstructured environment. A GPS waypoint navigation system comprised of a GPS unit and a compass successfully calculated trajectory vectors to guide an autonomous UGV sequentially along 10 predetermined GPS waypoints. The waypoint navigation and obstacle avoidance systems were successfully integrated to produce a fully autonomous UGV capable of navigating around encountered obstacles and making adjustments to continue along a path of GPS waypoints. Analysis of autonomous runs provided a linear model between speed and distance, which allowed an algorithm to be developed to maintain the waypoint navigation system in cases of temporary GPS dropout. The product of this study is a plausible representation of an autonomous GPS waypoint navigation system in an unstructured environment for an unmanned ground vehicle.

6.2 Significant Findings

Within large-scale global navigation, the great-circle effect of traveling along a straight line on the surface of a sphere introduces complications to traditional Euclidean vector analysis. However, it was observable that navigating between GPS

waypoints using vector analysis was successful within this study's scope. Even with simplification to the Haversine formula to convert the great-circle distances between GPS coordinates into Cartesian coordinate distances, the developed system excelled in accurately navigating the 10 waypoints sequentially. It is possible that the methodology used in this study might not be suitable for navigation between several degrees of longitude or latitude.

The system's accuracy on this scale is observable as the prediction algorithm during GPS downtime displays where the UGV would be with no real-world interference. This study also showed how predicting a location within a GPS denied navigation system is possible for short amounts of time ($<$30 seconds) but impractical on longer downtimes without measured data to compensate for inevitable compounding errors.

As for obstacle detection with a 2D-LIDAR, this study was able to cope with the two-dimensional view's limitations with high levels of success within a proof-of-concept. Considering a real-world application of this instrument, use of a single 2D-LIDAR is not sufficient, and any future work would be ill-advised to attempt progression in this methodology of using a single 2D-LIDAR without considering the suggestions outlined below.

6.3 Future Work

Regarding autonomy in robotic platforms, improvements and upgrades are perpetually available due to continuous technology growth. It is imperative to understand the limitations that were experienced in this study and what measures could be taken to improve the overall design.

One area of improvement that could be seen in future work would be the obstacle detection system. A two-dimensional LIDAR has significant limitations due to the nature of each 360-degree scan being performed upon a single plane. A common method of improving 2D-LIDARs is increasing the sampling rate such that more observations can be made per revolution, but even this is limited to a single plane of

view. This study had several encountered issues where the obstacle avoidance system was unable to detect an object in front of the UGV due to the single plane profile of the object. A three-dimensional LIDAR could theoretically manage the issues observed in this study. However, in an unstructured environment, laser-based vision is at a disadvantage compared to computer stereo vision, which is becoming more prevalent in self-driving cars.

Another area of improvement would be the navigation system while GPS is unavailable. This study's system was limited to a linear model predicting speed and distance traveled along a vector aimed by a compass. In any predictive situation, there should be at least one method for comparison to allow a higher degree of accuracy in predictions. An example would be the INS system integrated with Kalman filters discussed in Chapter 2. If an accelerometer were utilized to compare the predicted to the measured travel direction, then situations such as the one covered in Section 5.3 could potentially be avoided. Another recommendation would be to install encoders to the wheel's main axes, allowing for an accounting of common real-world issues such as slippage.

REFERENCES

[1] C. C. Government, *SEARCH & RESCUE*, 2021. Last retrieved 2021-02-19.

[2] NASAR, *About NASAR*, 2021. Last retrieved 2021-02-19.

[3] CRASAR, "The legacy of 9/11 for disaster robotics." Available at http://crasar.org/the-legacy-of-911-for-disaster-robotics/.

[4] Z. Beck, *Collaborative Search and Rescue by Autonomous Robots*. PhD thesis, University of South Hampton, December 2016. Last retrieved 2020-12-18.

[5] S. Karma, E. Zorba, G. Pallis, G. Statheropoulos, K. M. I. Balta, J. Vamvakari, A. Pappa, M. Chalaris, G. Xanthopoulos, and M. Statheropoulos, "Use of unmanned vehicles in search and rescue operations in forest fires: Advantages and limitations observed in a field trial," *International Journal of Disaster Risk Reduction*, vol. 13, pp. 307–312, July 2015. Last retrieved 2020-12-18.

[6] Y. Endo, J. C. Balloch, A. Grushin, M. W. Lee, and D. Handelman, "Landmark-based robust navigation for tactical ugv control in gps-denied communication-degraded environments," *Unmanned Systems Technology XVIII*, vol. 9837, May 2016. Last retrieved 2020-12-19.

[7] K. Li, J. Zhao, X. Wang, and L. Wang, "Federated ultra-tightly coupled gps/ins integrated navigation system based on vector tracking for severe jamming environment," *IET Radar, Sonar and Navigation*, vol. 10, pp. 1030–1037, July 2016. Last retrieved 2020-12-19.

[8] V. Higuti, A. Velasquez, D. Magalhaes, M. Becker, and G. Chowdhary, "Under canopy light detection and ranging based autonomous navigation," *Journal of field robotics*, vol. 36, pp. 547–567, May 2019. Last retrieved 2020-12-19.

[9] Harik and Korsaeth, "The heading weight function: A novel lidar-based local planner for nonholonomic mobile robots," *Sensors (Basel, Switzerland)*, vol. 19, p. 3606, August 2019. Last retrieved 2020-12-19.

[10] S. Padgett, "Autonomous unmanned ground-aerial surveillance system for areas with underdeveloped security infrastructure," Master's thesis, University of North Carolina at Charlotte, 2018. Last retrieved 2020-11-19.

[11] S. Mu-chun and J. Yeh, *Hikers urged to be better prepared amid rising mountain accidents*, 2021. Last retrieved 2021-02-19.

[12] N. Instruments, *User Guide and Specifications. NI myRIO-1900*. Available at https://www.ni.com/pdf/manuals/376047c.pdf.

[13] Digilent, "Pmod cmps reference manual." Available at https://reference.digilentinc.com/reference/pmod/pmodcmps/reference-manual.

[14] C. Veness, "Calculate distance, bearing and more between latitude/longitude points." Available at https://www.movable-type.co.uk/scripts/latlong.html.

[15] R Core Team, *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2020.

[16] H. Wickham, M. Averick, J. Bryan, W. Chang, L. D. McGowan, R. FranÃ§ois, G. Grolemund, A. Hayes, L. Henry, J. Hester, M. Kuhn, T. L. Pedersen, E. Miller, S. M. Bache, K. MÃŒller, J. Ooms, D. Robinson, D. P. Seidel, V. Spinu, K. Takahashi, D. Vaughan, C. Wilke, K. Woo, and H. Yutani, "Welcome to the tidyverse," *Journal of Open Source Software*, vol. 4, no. 43, p. 1686, 2019.

[17] D. Dunnington, *rosm: Plot Raster Map Tiles from Open Street Map and Other Sources*, 2019. R package version 0.2.5.

[18] J. Schwartz, "Bing maps tile system, understanding scale and resolution." Available at https://docs.microsoft.com/en-us/bingmaps/articles/bing-maps-tile-system.

[19] R. J. Hijmans, *geosphere: Spherical Trigonometry*, 2019. R package version 1.5-10.

APPENDIX A: Bill of Materials

Table A.1: Bill of Materials

Component	Quantity	Price (USD)
AM14U4 Robot Drive Base	1	689.00
BCxS2R3 Electronics Bundle	1	999.00
Dell Inspiron 5000	1	999.00
GlobalSat Bu-353-S4 GPS	1	27.99
LV-MaxSonar-EZ1 SRF	1	24.95
NI myRIO 1900	1	450.00
Pmod CMPS	1	7.99
RPLIDAR A3M1	1	699.95
TP-Link AC 1900 Router	1	69.99
	Total	3967.87

APPENDIX B: Uninterrupted-GPS Test Maps and Plots

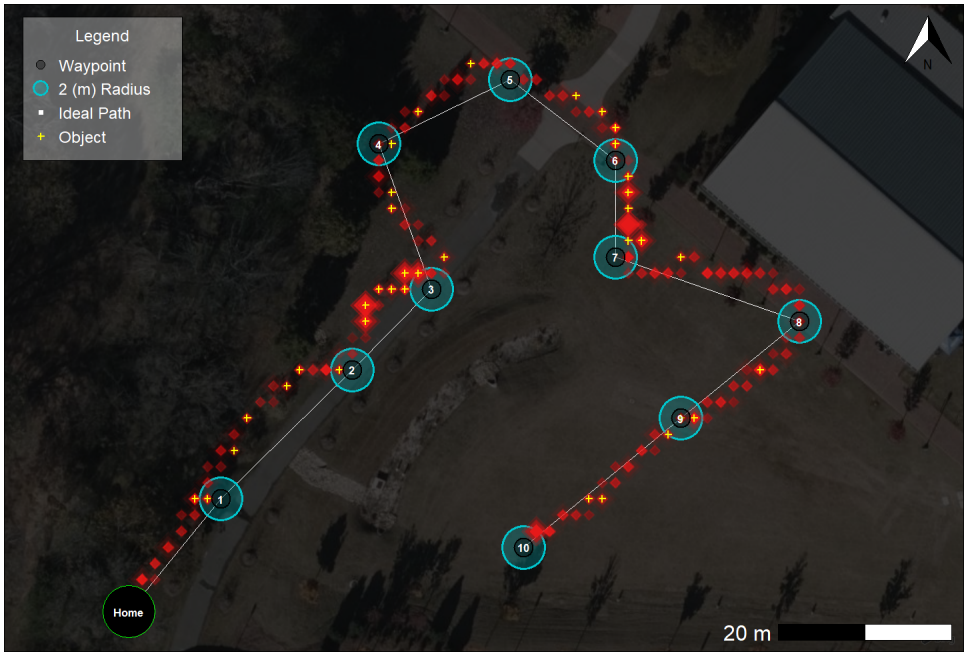

Figure B.1: Uninterrupted-GPS: Test 1

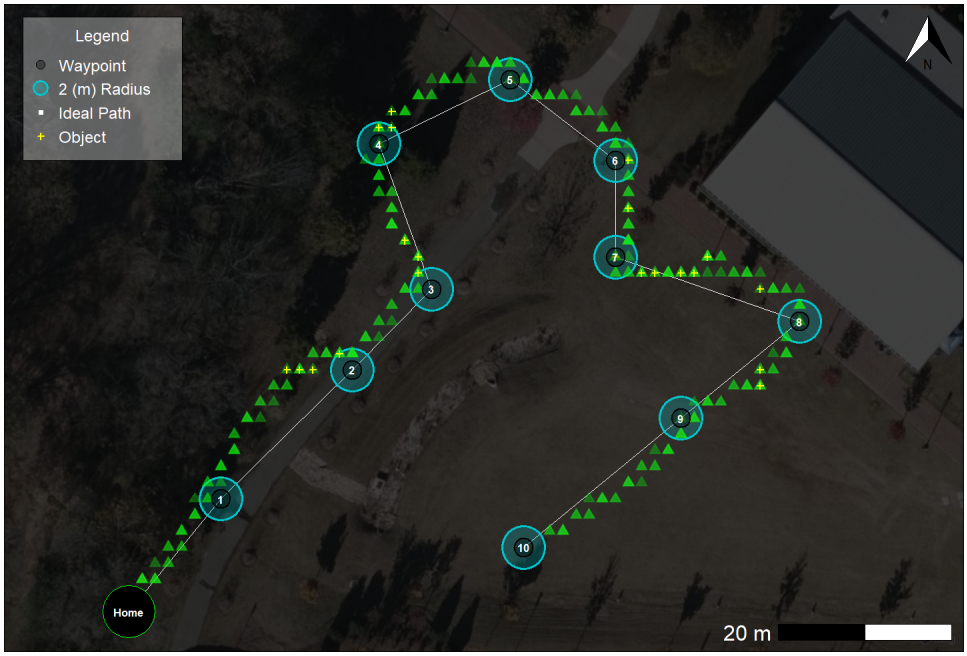

Figure B.2: Uninterrupted-GPS: Test 2

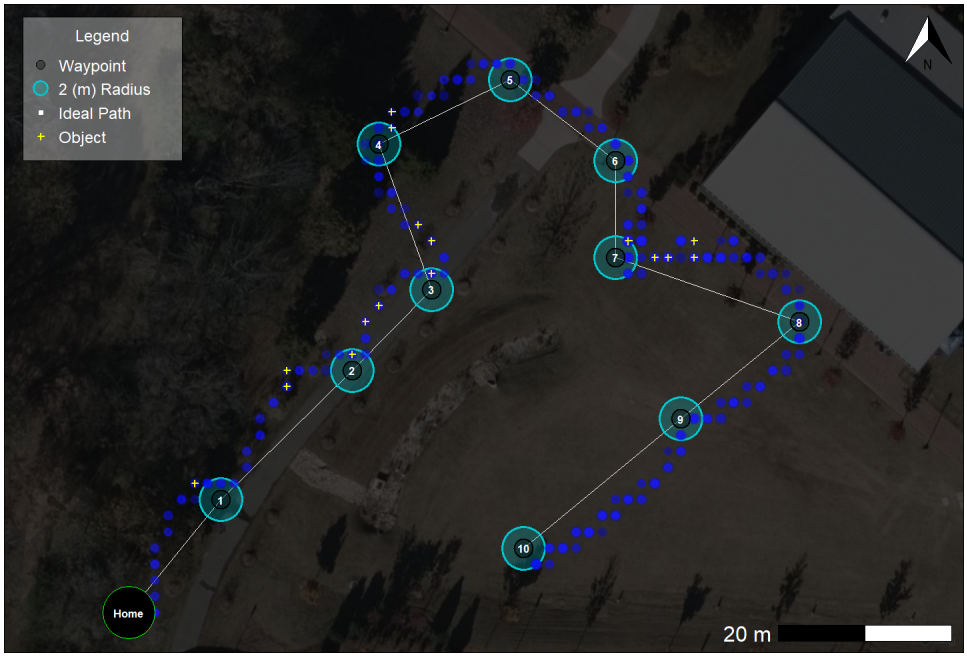

Figure B.3: Uninterrupted-GPS: Test 3

APPENDIX C: Intermittent-GPS Test Maps and Plots

C.1 Test Maps

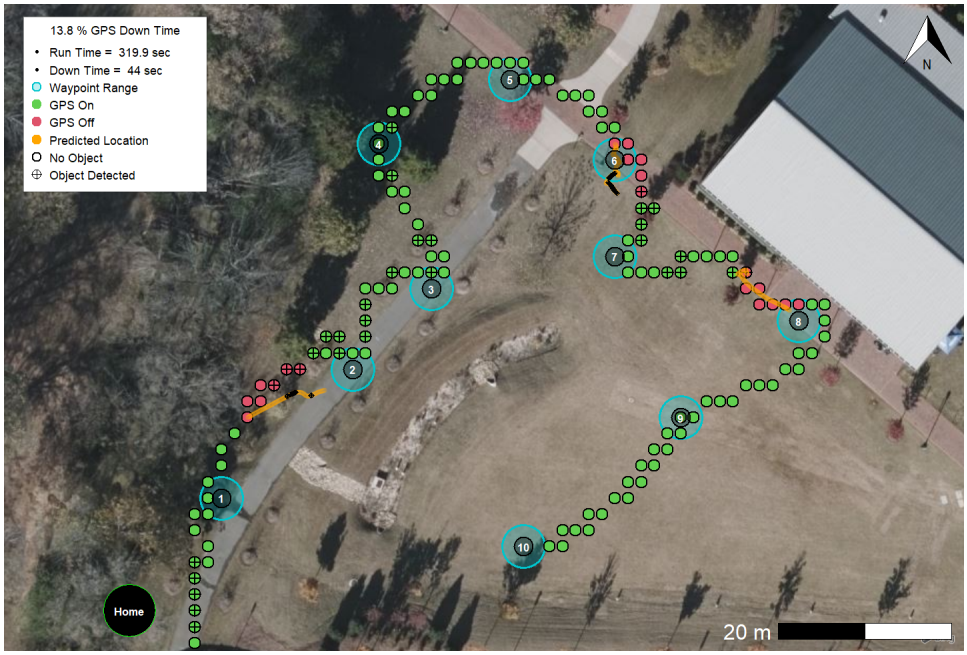

Figure C.1: 13.8% Masked GPS Map

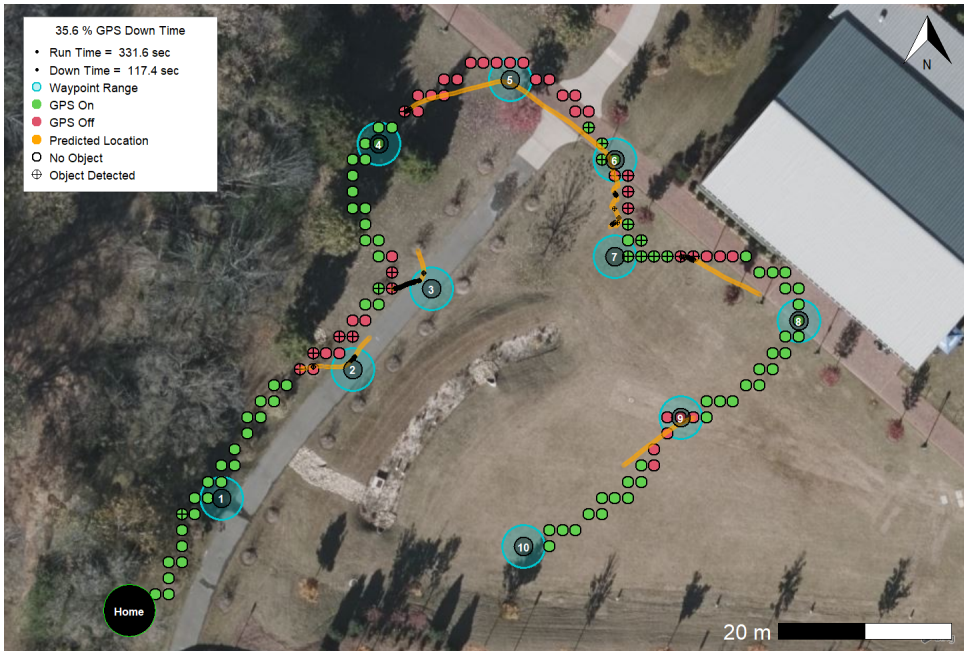

Figure C.2: 35.6% Masked GPS Map

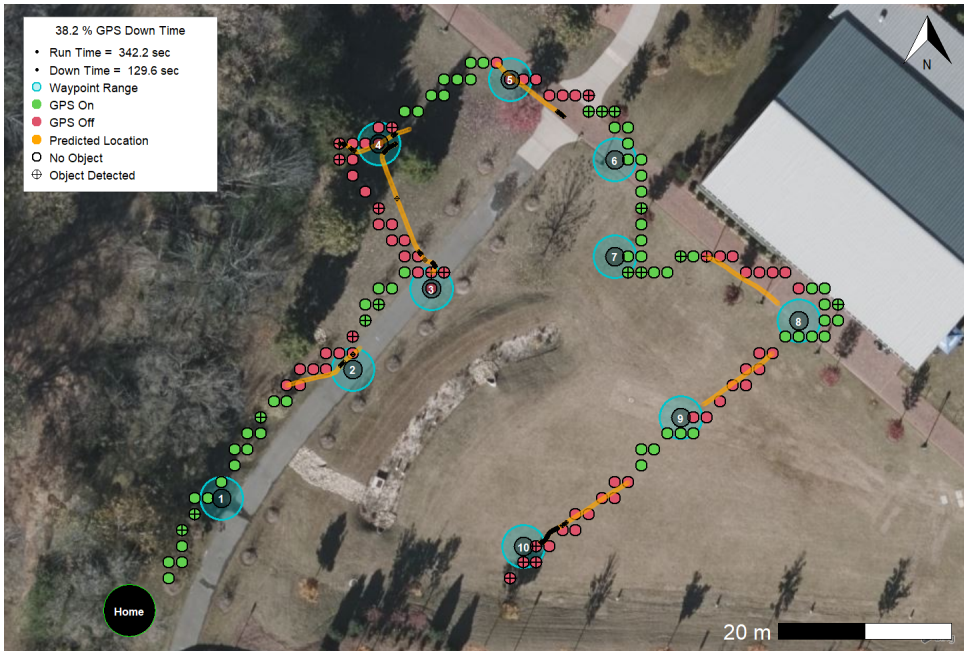

Figure C.3: 38.2% Masked GPS Map

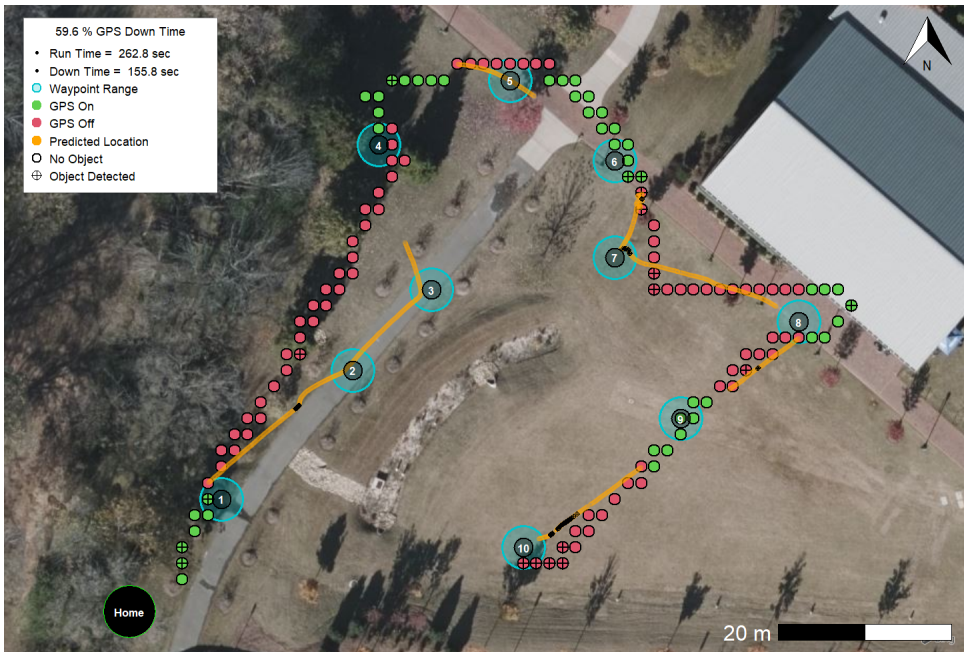

Figure C.4: 59.6% Masked GPS Map

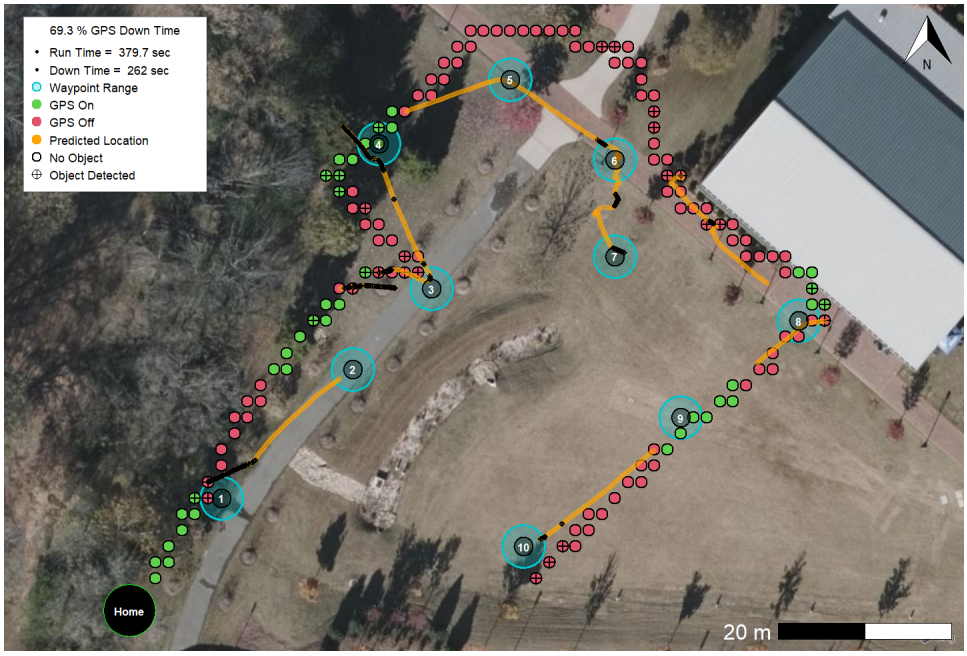

Figure C.5: 69.3% Masked GPS Map

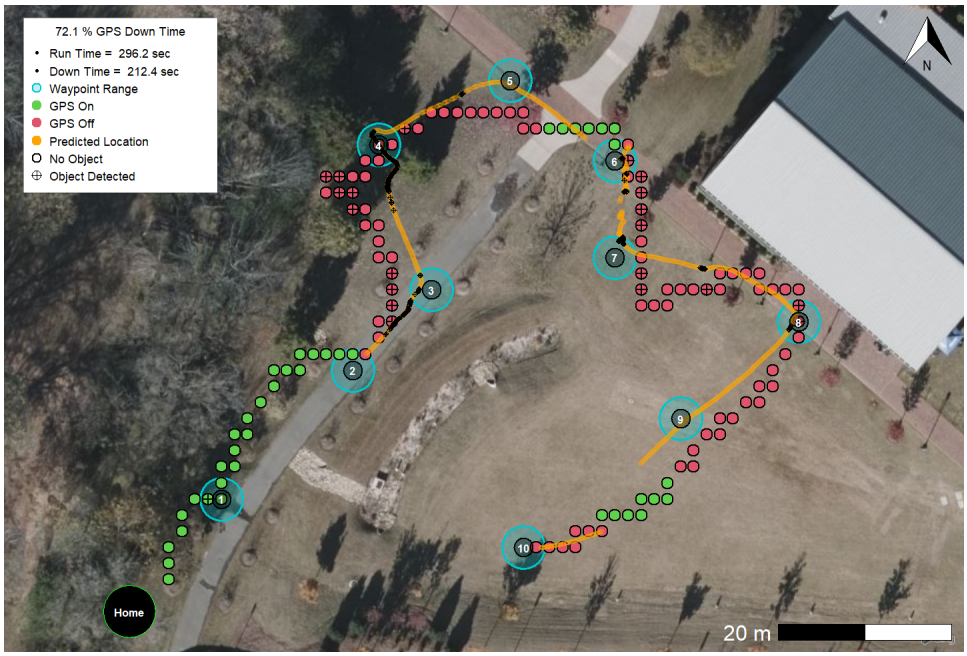

Figure C.6: 72.1% Masked GPS Map

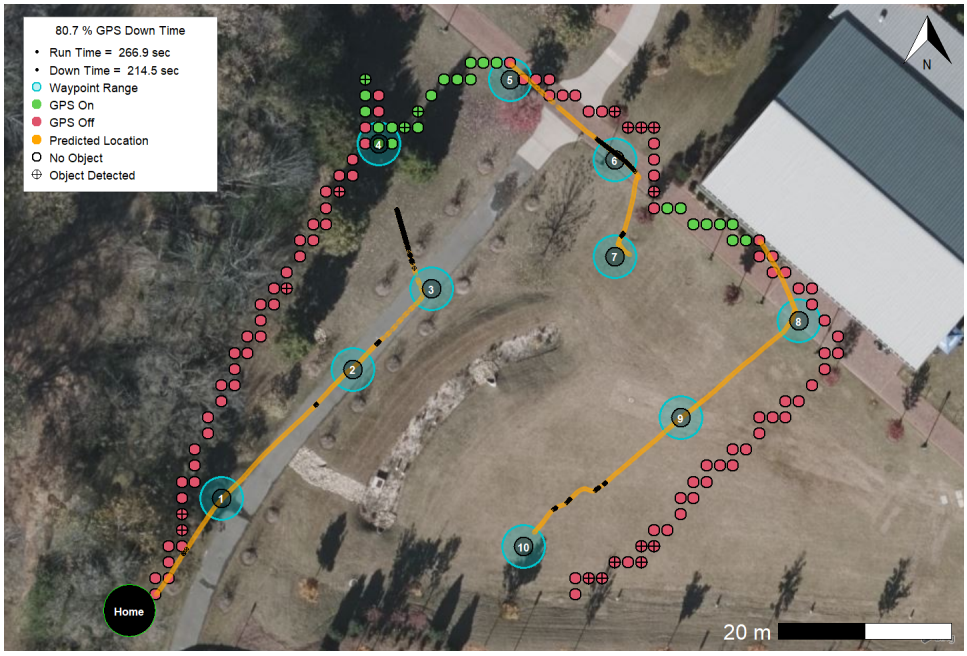

Figure C.7: 80.7% Masked GPS Map

C.2 Distance From Waypoint Plots

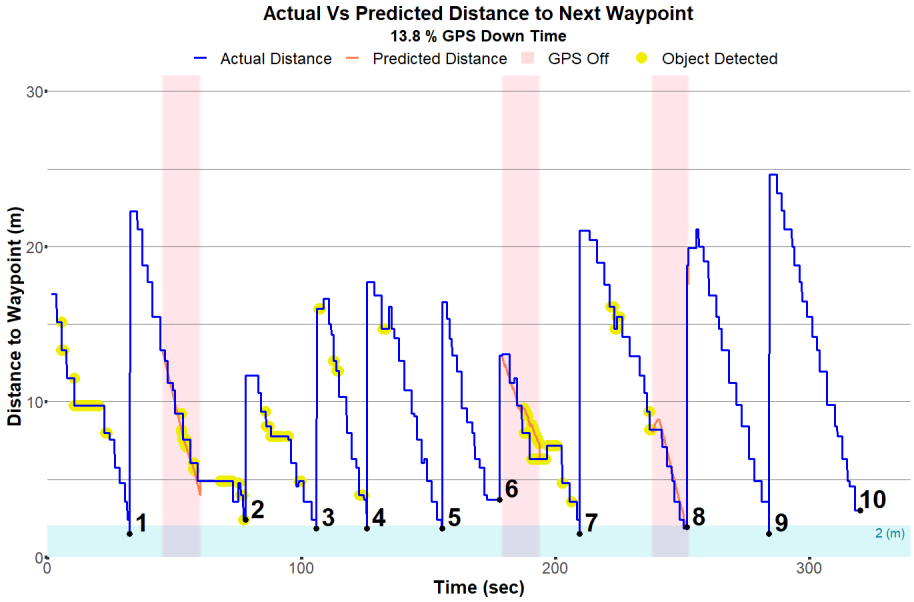

Figure C.8: 13.8% Masked GPS Distance From Waypoint

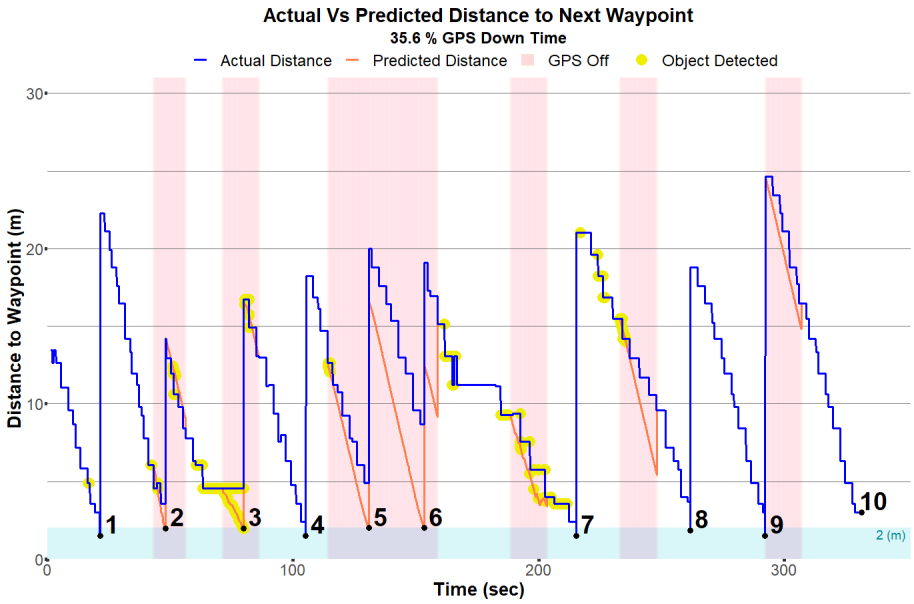

Figure C.9: 35.6% Masked GPS Distance From Waypoint

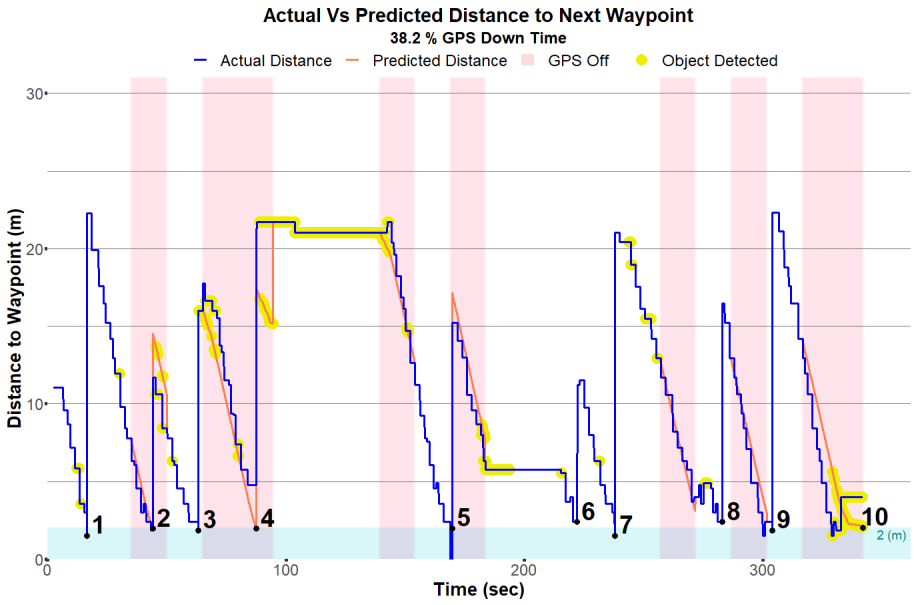

Figure C.10: 38.2% Masked GPS Distance From Waypoint

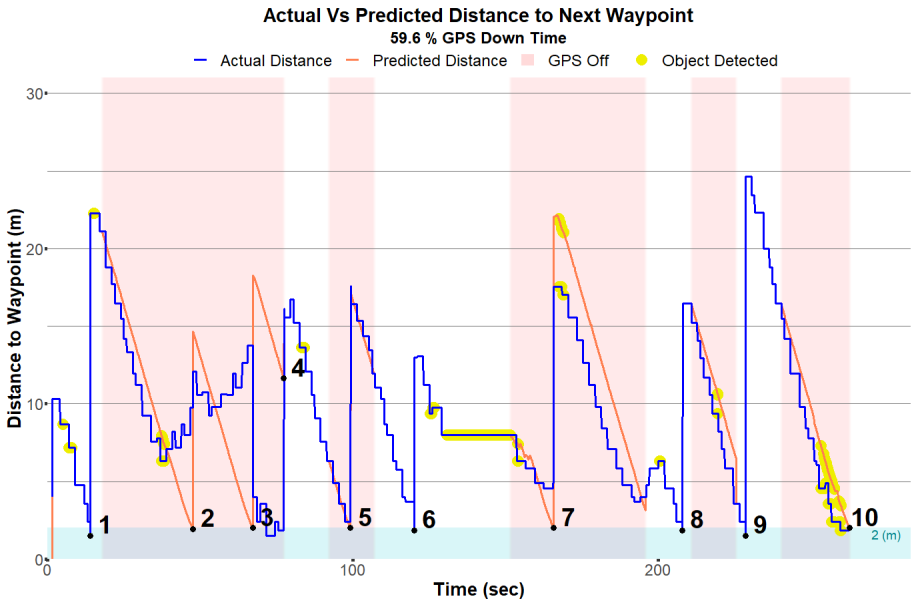

Figure C.11: 59.6% Masked GPS Distance From Waypoint

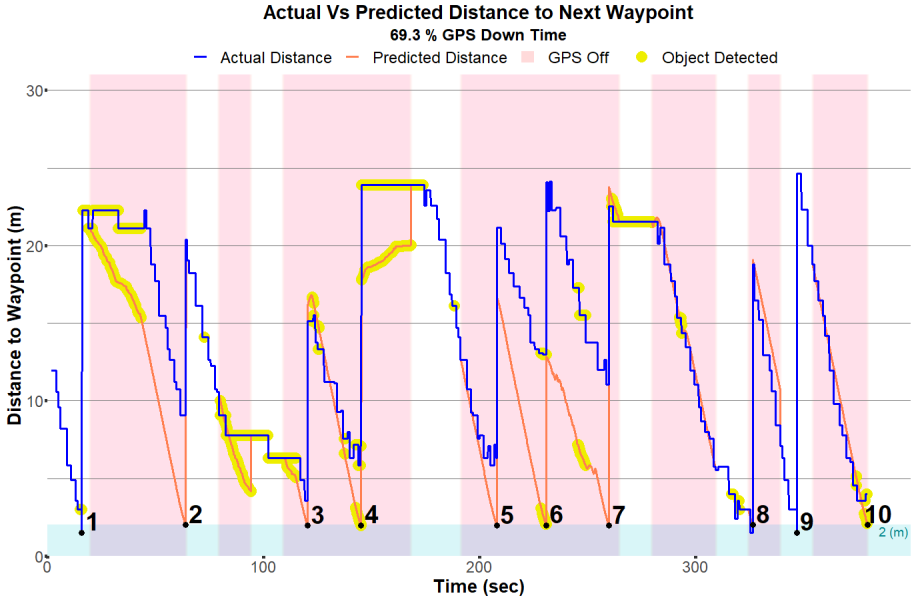

Figure C.12: 69.3% Masked GPS Distance From Waypoint

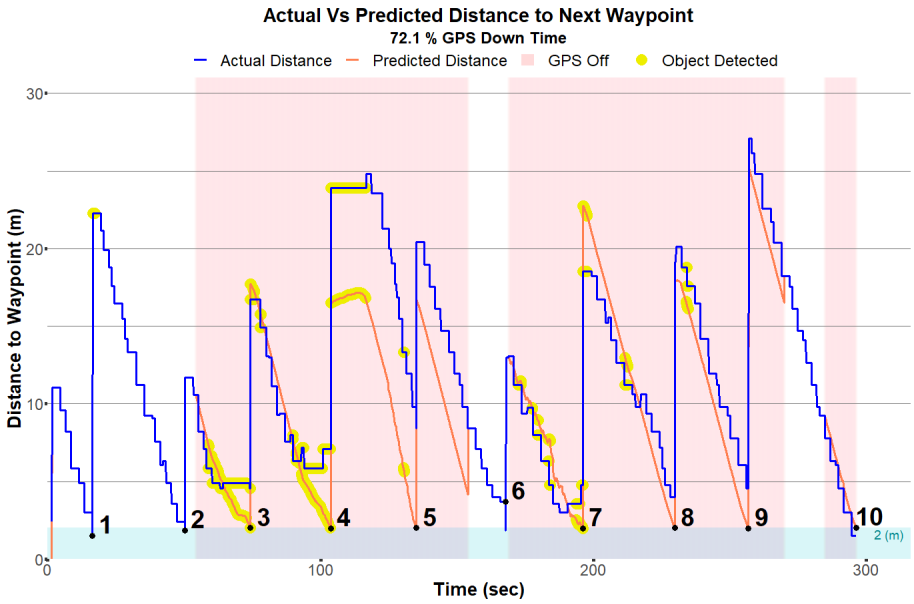

Figure C.13: 72.1% Masked GPS Distance From Waypoint

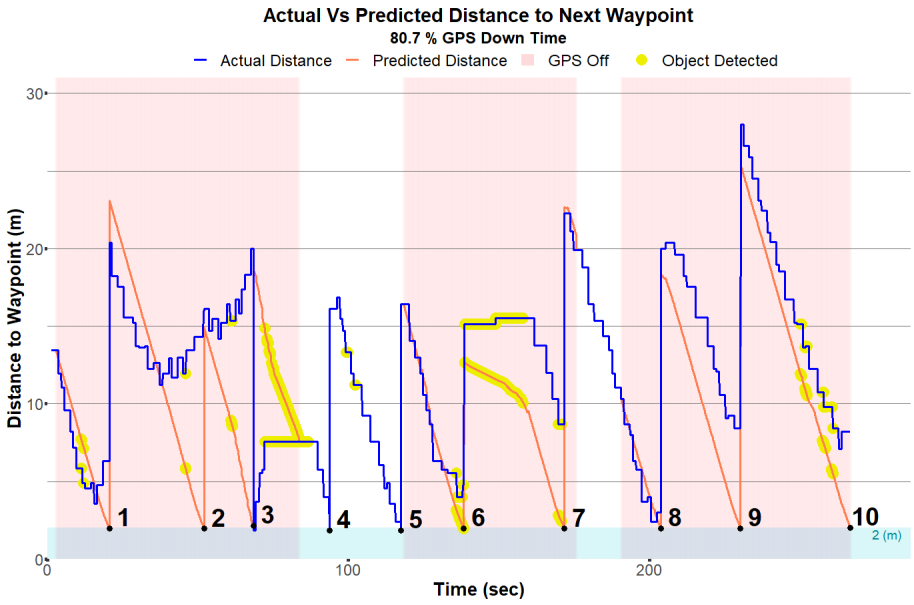

Figure C.14: 80.7% Masked GPS Distance From Waypoint

C.3 Closest Distance to Ideal Plots

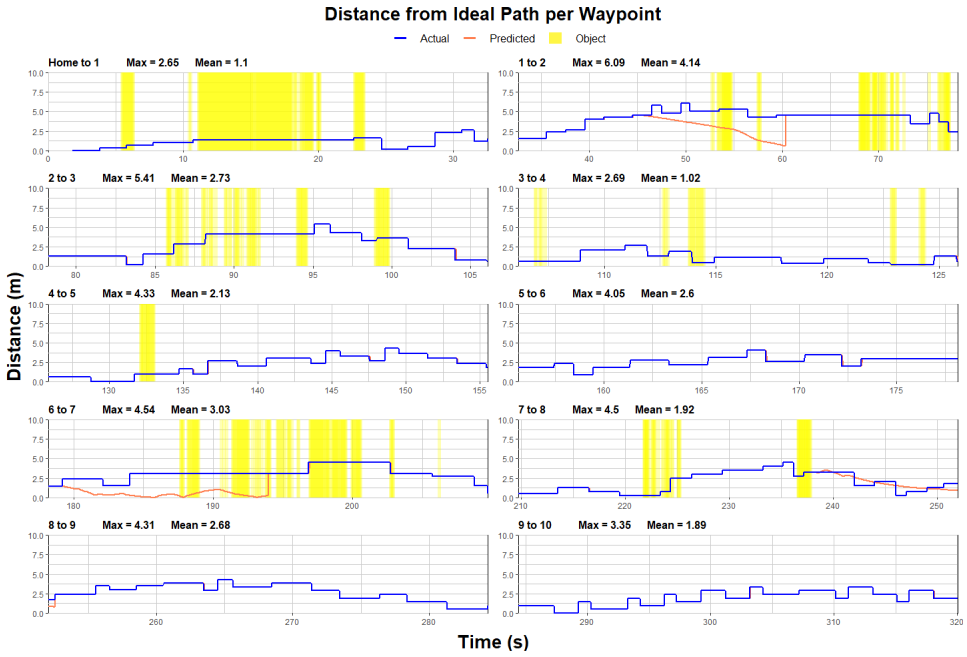

Figure C.15: 13.8% Masked GPS Closest Distance to Ideal Case

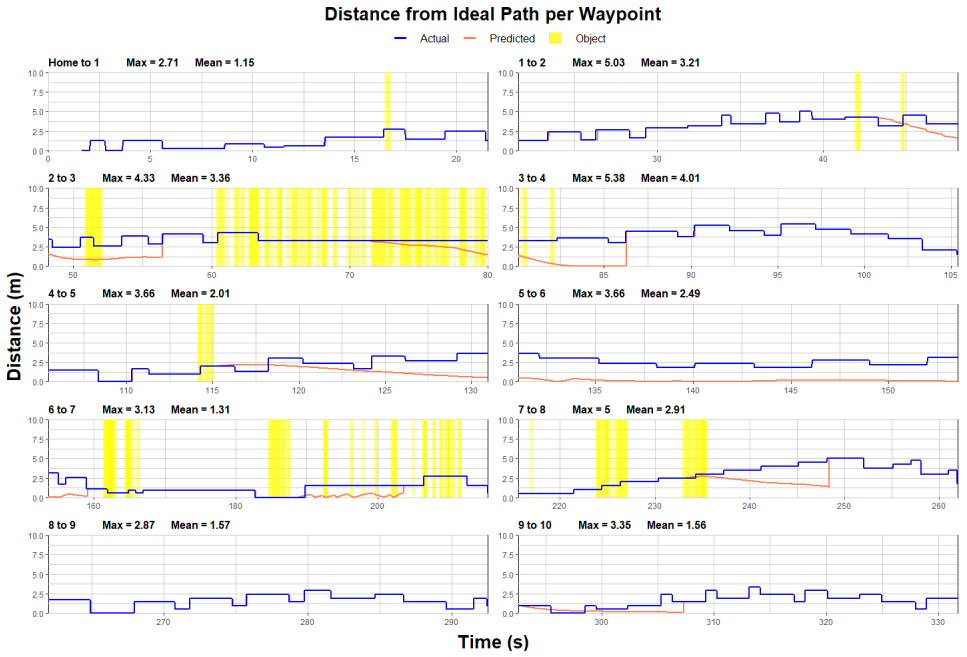

Figure C.16: 35.6% Masked GPS Closest Distance to Ideal Case

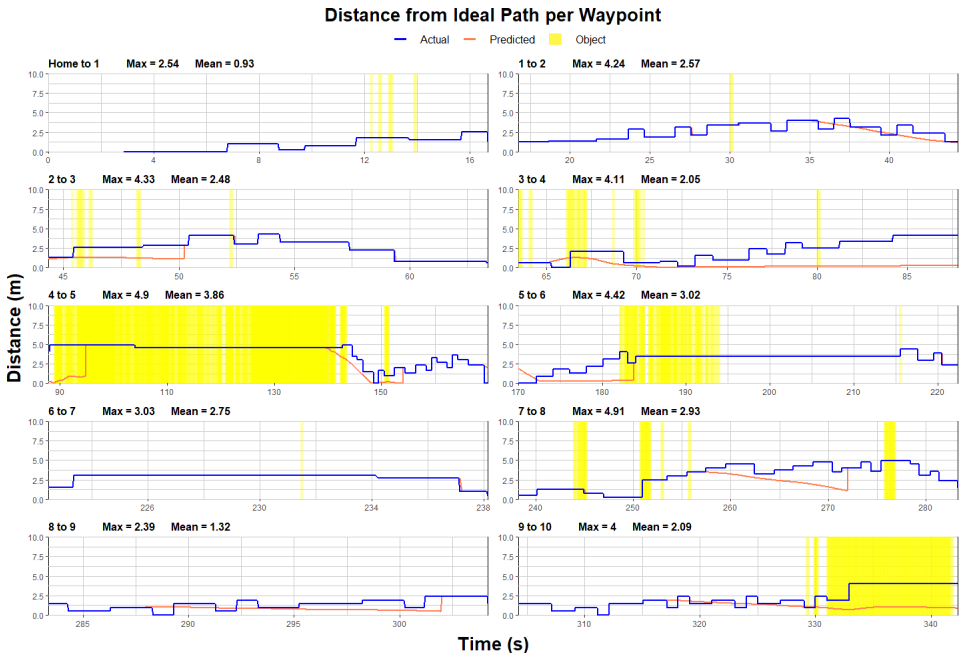

Figure C.17: 38.2% Masked GPS Closest Distance to Ideal Case

Figure C.18: 59.6% Masked GPS Closest Distance to Ideal Case

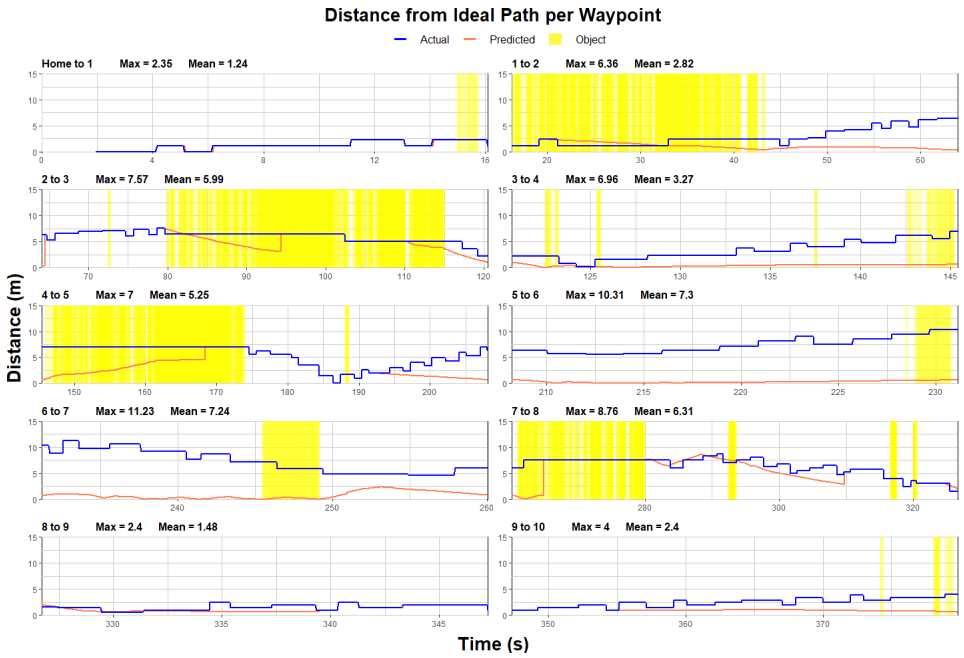

Figure C.19: 69.3% Masked GPS Closest Distance to Ideal Case

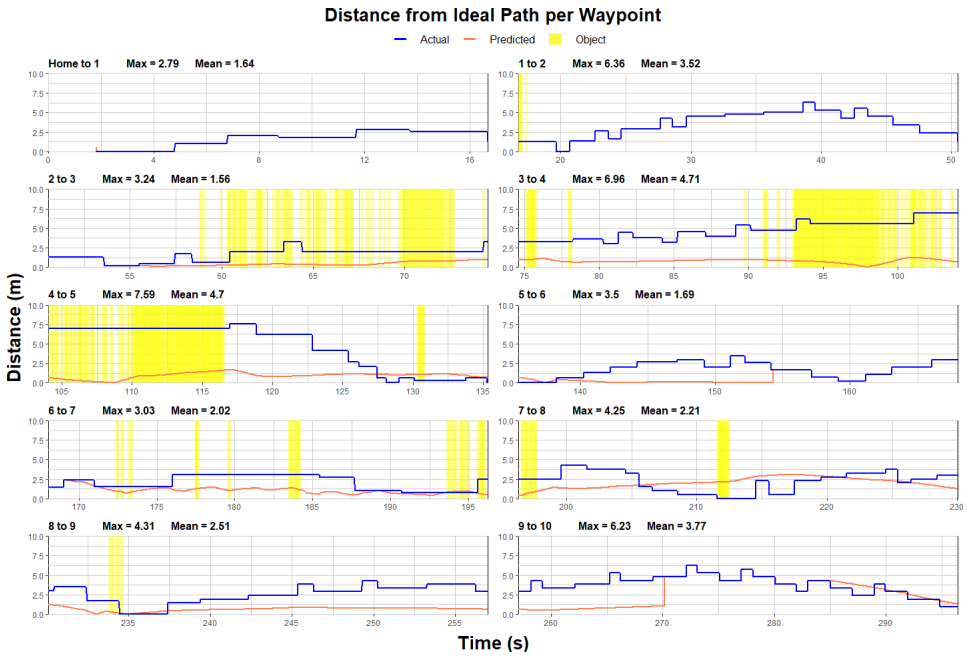

Figure C.20: 72.1% Masked GPS Closest Distance to Ideal Case

Figure C.21: 80.7% Masked GPS Closest Distance to Ideal Case