A NOVEL FRAMEWORK FOR INTEGRATING LEGACY VEHICLES INTO AN INTELLIGENT TRANSPORTATION SYSTEM

by

Benjamin Brian Rhoades

A **dissertation** submitted to the faculty of
The University of North Carolina at Charlotte
in partial fulfillment of the requirements
for the degree of Doctor of Philosophy in
Electrical Engineering

Charlotte

2018

Approved by:

____

Dr. James M. Conrad

____

Dr. Asis Nasipuri

____

Dr. Courtney Smith-Orr

____

Dr. Maciej Noras

ABSTRACT

BENJAMIN BRIAN RHOADES. A novel framework for integrating legacy vehicles into an intelligent transportation system. (Under the direction of DR. JAMES M. CONRAD)

The last few years have seen rapid advancements in deploying semi and fully-autonomous vehicles into the mass market. Several car manufactures and technology companies have spent years developing these vehicles with aspirations to mass produce their own autonomous automobiles. In the same time period, however, the average age of a passenger vehicle on the road has increased to 11.4 years, as of 2015. This has led to an approximate 50:1 ratio of standard passenger vehicles as compared to their fully and semi autonomous counterparts, and a widening gap between newer and older (legacy) vehicles occupying the same highways. This research seeks to address this issue by outlining a framework that would provide legacy vehicles and newer autonomous enabled automobiles a method to interact with each other and their surrounding infrastructure, creating a true Intelligent Transportation System (ITS).

Intelligent transportation systems use two types of communications, Vehicle-to-Vehicle (V2V) and Vehicle-to-Infrastructure (V2I). Both of these communication types already have several Institute of Electrical and Electronics Engineers (IEEE) protocols and International Standards Organization (ISO) standards / hardware specifications that dictate how they function, however, these methods often involve adding costly sensors and other peripherals to the vehicle / roadway that must adhere to these strict standards to enable them to communicate effectively. Most proposed ITS systems do not incorporate older (legacy) vehicles into their communication schemes or algorithms. The research outlined in this dissertation highlights a novel framework that would enable full integration of legacy vehicles into an all encompassing ITS. This framework would ensure all automobiles, not just modern ones, are able to transmit and receive critical vehicle telemetry and other vital data points to one another to improve roadway safety. This framework would aid in expediting the shift from standard manual vehicle control to a fully autonomous and connected infrastructure future.

ACKNOWLEDGEMENTS

I would like to first thank my advisor, Dr. James M. Conrad for his careful guidance throughout this project and my college career as well as Dr. Asis Nasipuri, Dr. Courtney Smith-Orr, and Dr. Maciej Noras for serving on my committee.

I want to thank my understudy graduate students Vedhas Deshpande, Chinmay Kishor Admane, Vinit Katariya, Pallavi Mahendra Avle, and Disha Srivastava who, through many hours of tireless work, made large portions of this research possible.

I want to also thank my parents for their unfettered love, financial sacrifice, and unmatched integrity that gave me the inspiration and courage to pursue even the highest of aspirations in my life.

But chiefly I want to thank God for giving me the physical and mental strength to endure to the end and finish the race set before me.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

ACK	Acknowledgment
API	Application Program Interface
App	Application
BBW	Brake-by-Wire
BCU	Brake Control Unit
BMW	Bavarian Motor Works
BT	Bluetooth
CAN	Controller Area Network
CAN-H	CAN High
CAN-L	CAN Low
CBU	Controller Base Unit
CPU	Central Processing Unit
CRC	Cyclic Redundancy Check
DBW	Drive-by-Wire
DC	Direct Current
DDM	Decimal Degree Minutes
DLC	Data Link Connector
DOT	Department of Transportation
DSRC	Dedicated Short Range Communications

ECM	Electronic Control Unit
ECU	Electronic Control Unit
EPA	Environmental Protection Agency
EPIC	Energy Production and Infrastructure Center
FFD	Full-Fuction Device
FHWA	Federal Highway Administration
GHz	Gigahertz
GND	Ground
GPS	Global Positioning System
GUI	Graphical User Interface
GVWR	Gross Vehicle Weight Rating
HUD	Heads Up Display
IEEE	Institute of Electrical and Electronics Engineers
ISO	International Organization for Standardization
ITS	Intelligent Transportation System
JSON	JavaScript Object Notation
Kbps	kilobytes per second
KWP	Keyword Protocol
LCD	Liquid Crystal Display
LVTT	Legacy Vehicle Telemetry Transmitter

Mbps	Megabit per Second
MCU	Microcontroller Unit
MHz	Megahertz
NGTOC	National Geospatial Technical Operations Center
NHTSA	National Highway Traffic Administration
NMEA	National Marine Electronics Association
NRZ	Non-Return to Zero
OBU	On-Board Unit
OTS	Off-the-Shelf
P2P	Point-to-Point
PWM	Pulse Width Modulation
PWR	Power
RF	Radio Frequency
RFD	Reduced-Function Device
RGB	Red Green Blue
RPi	Raspberry Pi
RPM	Revolutions per Minute
RSU	Road Side Unit
RSU	Road-Side Unit
Rx	Receive / Reception

S1	Series 1
S2	Series 2
SAE	Society of Automotive Engineers
SBW	Steering-by-Wire
TBW	Throttle-by-Wire
Tx	Transmit / Transmission
UART	Universal Asynchronous Receiver/Transmitter
URL	Uniform Resource Locator
US	United States
USB	Universal Serial Bus
USDOE	United States Department of Energy
USDOT	United States Department of Transportation
USGS	United States Geological Survey
V	Volts
V2I	Vehicle to Infrastructure
V2V	Vehicle to Vehicle
VIN	Vehicle Identification Number
VPW	Variable Pulse Width
Wi-Fi	Wireless Fidelity
WLAN	Wireless Local Area Network

WSN	Wireless Sensor Network

CHAPTER 1: INTRODUCTION

An Intelligent Transportation System (ITS) that would enable all vehicles on the road to communicate with each other and their surrounding infrastructure is not a new concept [1]. In the past few years there has been extensive research and development (R&D) in the field of autonomous and semi-autonomous vehicles, however, most of this R&D has been focused on improving existing fully-autonomous vehicles. Significantly less research has been focused on integrating current non-autonomous, automobiles into a cohesive ITS ecosystem where older (legacy) and newer autonomous vehicles can communicate with one another and their surrounding infrastructure. Fully-autonomous vehicles have the prospect to drastically change the driving experience and landscape of our modern highways; however, there is an increasing number of legacy vehicles that occupy the same roadway space as their high tech counterparts. This has created a gap in the automotive market where older vehicles are on the road longer and newer semi and fully-autonomous vehicles are beginning to become more widespread. This causes a problem when the ultimate goal of an ITS is to have all vehicles communicating with each other and their surrounding infrastructure concurrently. The current solutions to enable these older vehicles to interact with each other and their surroundings is to equip them with a barrage of expensive sensors that often results in the vehicles being defaced and forever altered. A novel framework is needed to enable communication between legacy and newer autonomous vehicles. Such a framework would ensure a smooth transition as society moves away from the traditional driver-centric model of vehicles to the newly emerging fully-autonomous model.

1.1 Motivation

According to the United States Department of Transportation (USDOT), the average age of a vehicle on the road today has increased from 8.4 years in 1995 to 11.5 years in 2015 [2]. This is due, in part, to the increased number of after market parts to repair these vehicles and the increased reliability of automobiles produced in the last 10 years. Car and technology companies, such as Tesla, Alphabet (parent company of Google), and UBER have invested heavily in the technologies that enable semi and even fully autonomous vehicles to operate on commercial highways. The market share of these vehicles, however, are small as compared to the mass produced passenger and commercial vehicles. According to first quarter 2016 production numbers from Tesla and GM they produced 14,820 and 720,000 vehicles in the US, respectively [3, 4]. This implies that for every semi and fully autonomous enabled vehicle on the road there are at least 50 standard passenger vehicles. Combine that with average age of an automobile on the road today it is clear that older, non autonomous vehicles, will be on the highways for the foreseeable future. This creates an issue when trying to develop an infrastructure that demands real-time communications across the vehicle capable spectrum. While newer vehicles may come pre-equipped with these technologies, older legacy vehicles will need to communicate with these newer, technology laden, vehicles and the surrounding infrastructure.

If legacy vehicles were able to communicate their respective telemetry and other time critical data to a centralized system then all vehicles would have basic data about one another. It is important to note that protocols and standards already exist to facilitate this communication, however these technologies, in their present form, are costly in both ends of the implementation spectrum. This has led to a slowdown in the distribution and adoption of these technologies. Even with these current standards a novel framework is needed to help bridge the gap until these standards and systems become widespread and accessible to all vehicles on the roadway.

1.2 Objective of This Work

A framework is needed to ensure legacy vehicles can communicate with newer technology enable automobiles. This framework would have to be robust enough to integrate into all present and existing vehicles on the road today. The framework would also have to be low cost and easily deployed to ensure that full integration would occur on an aggressive time scale. The system would also need to utilize existing open source technology solutions to enable rapid development and deployment. Hardware that will be interfaced with the legacy vehicle needs to be discrete and not alter the overall aesthetics of the vehicle. The software solutions, provided by this framework, would have to interface with existing infrastructure.

The implementation of this framework, along with the hardware and software solutions produced, are discussed in this dissertation. A novel framework has been developed that utilized existing open source radio frequency (RF) radios and microcontroller platforms to enable legacy vehicles to communicate with a centralized ITS system.

1.3 Contribution

The main contribution provided by this framework is proving a method for legacy vehicles to interact and contribute their telemetry and other time critical information to a centralized ITS. The hardware and software solutions provided by this framework ensure that all vehicles on the road today will have a method to communicate with one another and with their surrounding infrastructure. This framework will aid in the transitional period between semi and fully autonomous vehicles. This framework will also provide a method for owners of legacy vehicles to receive time critical information about roadway conditions and even real time vehicle diagnostics, all without physically altering the vehicles' appearance or performance. Currently there is no framework or working group that would allow legacy vehicles a way to communicate, not only with another vehicle, but a broader ITS that could provide crucial driving data and diagnostics to the driver.

1.4 Organization

This dissertation is organized into seven chapters. Chapter Two provides a review of previous work that provides background information on the dissertation topic and the current state of research in this topic area. Chapter Three provides a brief overview of all relevant protocols and standards associated with the framework development and validation. Chapters Four and Five are structured to mirror one another. Chapter Four outlines the necessary components that are required to develop an ITS framework. Chapter Five provides its complement by expanding upon the implementation of each aforementioned component. Chapter Five will also discuss the physical implementation developed by the author to test and validate each aspect of the ITS framework. Chapter Six provides a detailed description of the results from this research. Chapter Seven provides a conclusion of work completed along with ideals for future work with this research.

CHAPTER 2: REVIEW OF PREVIOUS WORK

The idea of having a cohesive and collaborative union between interconnected vehicles and their surrounding infrastructure is not new. Numerous standards and protocols have been developed in the past few decades to pave the way to that reality, however, the deployment of these technologies have been slow. Many interim solutions have been designed and developed to act as a bridge between the current driver centered model to a future driverless centric one. This paper provides an overview of the alternate techniques that are currently being implemented to drive us toward an interconnected infrastructure and vehicle future. Relative research from the author will also be discussed.

2.1 Introduction

The concept of an intelligent and interconnected infrastructure that communicated with non-autonomous and autonomous enabled vehicles is not new [5]. Since the first wide installment of embedded processors and sensors into vehicles began in the late 1980's there has been development to create an intelligent transportation system (ITS). Many standards and even a special frequency spectrum have been allocated to make this become a reality. The concept of an ITS is simple; all the vehicles on the road are in constant communication with one another, Vehicle-to-Vehicle (V2V), and their surrounding infrastructure, Vehicle-to-Infrastructure (V2I). The deployment of the technologies needed to accomplish this has been a slow and expensive process. It requires the installation of multiple pieces of large equipment both on the roadway (Roadside Units (RSU)) and inside the vehicle (On-Board Units (OBU)). Fig. 2.1a and Fig. 2.1b show a OBU and RSU produced by Cohda Wireless. These units must operate in tandem with one another. This implies that a complete saturation of these units must exist to enable a fully functional ITS to exist. To ease the transition of a standardized ITS, the research community has offered alternative methods and techniques that are able to be quickly deployed and at a substantially lower cost while still holding true to the concept of an ITS. This paper aims to look into these alternative techniques that are currently under development. The advantages and disadvantages of each alternate method will be discussed. Each method will also be compared using the metrics of cost, ease of deployment, security, and scalability will be used to determine the feasibility of each method.

(a) Cohda Brand On-Board Unit (OBU)

(b) Cohda Brand Road-Side Unit (RSU)

Figure 2.1: Cohda Brand On-Board Unit (OBU) and Road Side Unit (RSU) [6] [7]

2.2 Background

There already exist many protocols and standards that ensure the deployment of an ITS that is consistent across all states and vehicles. Many of these standards were created before the widespread telecommunication backbone network, that enabled modern smart devices to function, was widely deployed. These standards also were initially created for simplistic systems such as loop detection circuits that tie into traffic signals and other low level traffic tasks. These protocols and standards also predate the advent of semi-autonomous vehicles, such as the Tesla Model S. This has resulted in a delay of deployment of a nationwide ITS. The following sections breakdown the current protocols and standards established for an ITS and relay some relevant background information on each enabling technology.

2.2.1 IEEE 802.11p, VANET, WAVE, WSMP, and DSRC

In 1999 the US Federal Communication Commission (FCC) announced that it had allocated 75 Megahertz (MHz) of Dedicated Short Range Communications (DSRC) spectrum located in the 5.9 Gigahertz (GHz) band, which would exclusively be used for V2V and V2I communications [8]. The image in Fig. 2.2 shows a breakdown of the allocated spectrum.

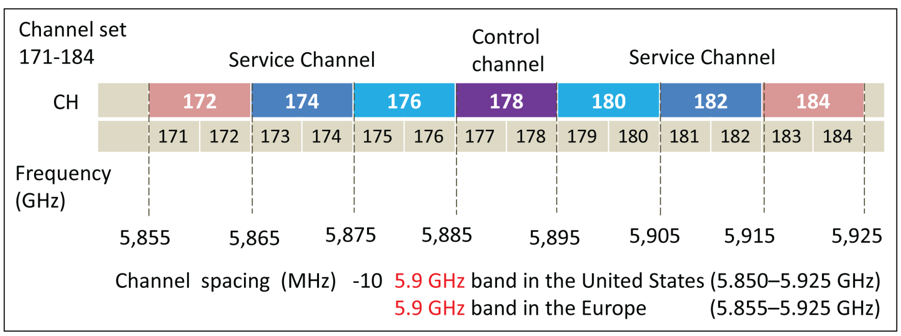


Figure 2.2: Spectrum allocation in IEEE 802.11p [9]

WAVE or Wireless Access in Vehicular Environments is a mode of operation used by 802.11P devices to operate in the DSRC frequency band. WAVE is also synonymous with the Institute of Electrical and Electronics Engineers (IEEE) P1609.X Standard. It operates in the upper five layers of the Open Systems Interconnection (OSI) stack. The lower two layers, Data Link and Physical, is where the IEEE 802.11P Standard resides. Fig. 2.3 ,obtained from the National Highway Traffic Safety Administration (NHTSA), illustrates how these standards are comprised on the OSI stack.

Fig. 2.4 illustrates a more in-depth breakdown of the architecture. Fig. 2.4 also shows the two tracks for safety and non-safety applications. This ensures that when time crucial information, such as a vehicle abruptly braking ahead, can propagate

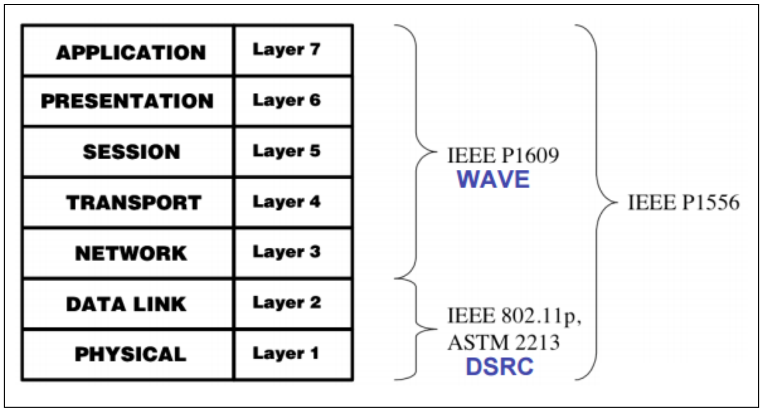


Figure 2.3: OSI Protocol Stack for IEEE 802.11P and WAVE [10]

through the network with low latency. This safety centric low latency message is in the form of a WAVE Short Message Protocol (WSMP). It is important to note that WAVE also has a robust safety structure in the form of the IEEE 1609.2 standard. This ensures that point-to-point (P2P) communication is secure and not easily susceptible to outside malicious activity [11].

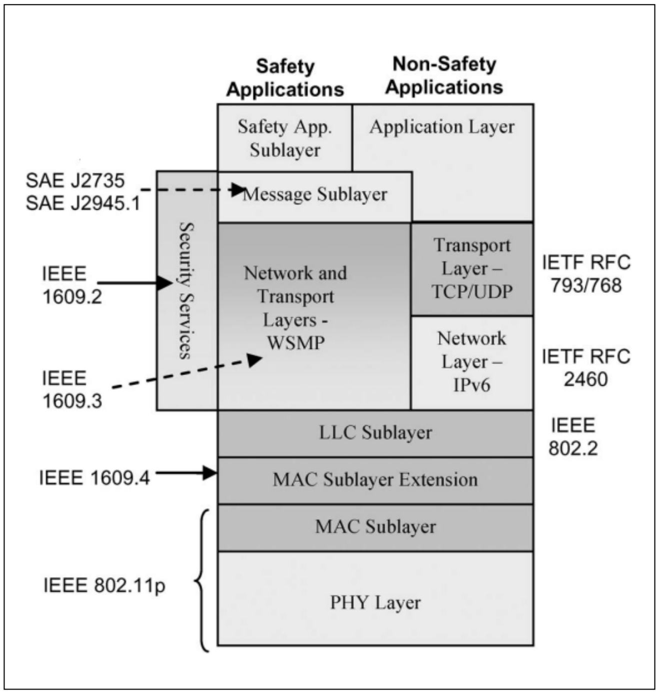


Figure 2.4: Layered Architecture for DSRC Communication [9]

Vehicular Ad hoc Networks (VANET) are designed to act as a large scale Mobile

Ad-hoc network (MANET) that enable moving vehicles to act as wireless nodes. A VANET has to handle the high mobility of vehicular nodes and the varying speeds of each vehicle in relation to one another. This is typically not an issue since most vehicular nodes move in an organized pattern, i.e. well defined streets. The backbone of a VANET typically consists of two main components, a On-Board Unit (OBU) and a Road-Side Unit (RSU). The OBUs and RSUs typically use the aforementioned protocols to communicate with one another with low latency and high security, however, due to the need to physically install devices in each vehicle and periodically on the surrounding physical infrastructure has led to a delay in this scheme being widely adopted [11].

2.3 Alternate ITS Deployment Techniques

The widespread distribution of the enabling technologies to distribute WAVE and other ITS applications has been slow due to a number of factors that include cost, ease of deployment, and security. Since there are many other external factors that have delayed the widespread adoption and distribution of ITS there exist many alternate methods and techniques to deploy an ITS. The following looks into three alternative techniques that are active research areas that aim to expedite the deployment of ITS type systems. These three techniques overlap in part in their implementation and distribution, however, they each represent their own advantages and disadvantages for widespread deployment. These alternate methods are not intended to replace the current protocols and standards, nor do they claim to have the perfect solution, however, they do aim to bridge the divide that currently exists between an ideal fully functioning ITS and the current state of ITS technologies.

2.3.1 Wi-Fi Based ITS

Wireless Fidelity (Wi-Fi) is commonplace in almost every corner of our existence. It is becoming rare to find any location, minus some rural areas, that do not have access to a broadband internet connection. Fig. 2.5, provided by WiGLE.net provides an illustration of the scale and distribution of Wi-Fi networks. Fig. 2.6 shows the steady increase in the number of access connections since January of 2010. The two data points highlighted in the figure show that, as of 1/5/2010 and 1/10/2017, the site had a reported 18.9M (million) and 304.39M, respectively, APs reported by their site.

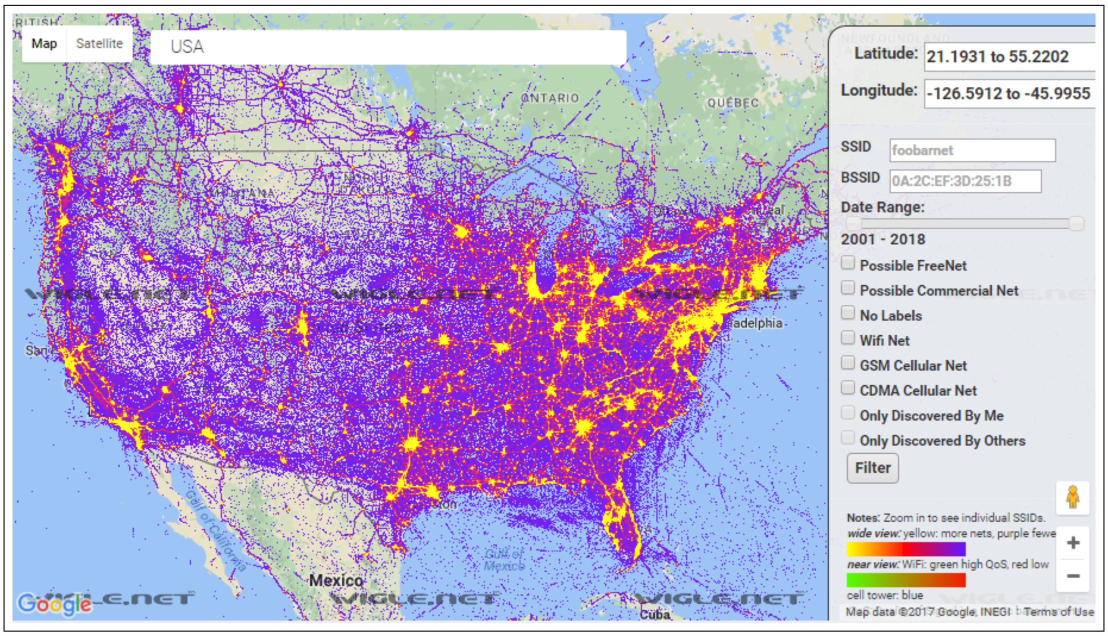


Figure 2.5: Approximate Wi-Fi coverage map of the USA (as reported by WiGLE.net) [12]

With the vast infrastructure that already exists to support the Wi-Fi network, researchers have begun to consider the possibility of using it as a prime mover for quickly distributing and deploying a nationwide ITS. The advantages and disadvantages of this alternate method of ITS deployment is investigated and reported in the following two sections.

2.3.1.1 Wi-Fi Based ITS Advantages

Deploying a Wi-Fi based ITS has two key advantages over using the current standards. The first advantage of using Wi-Fi over the traditional standards is the cost.

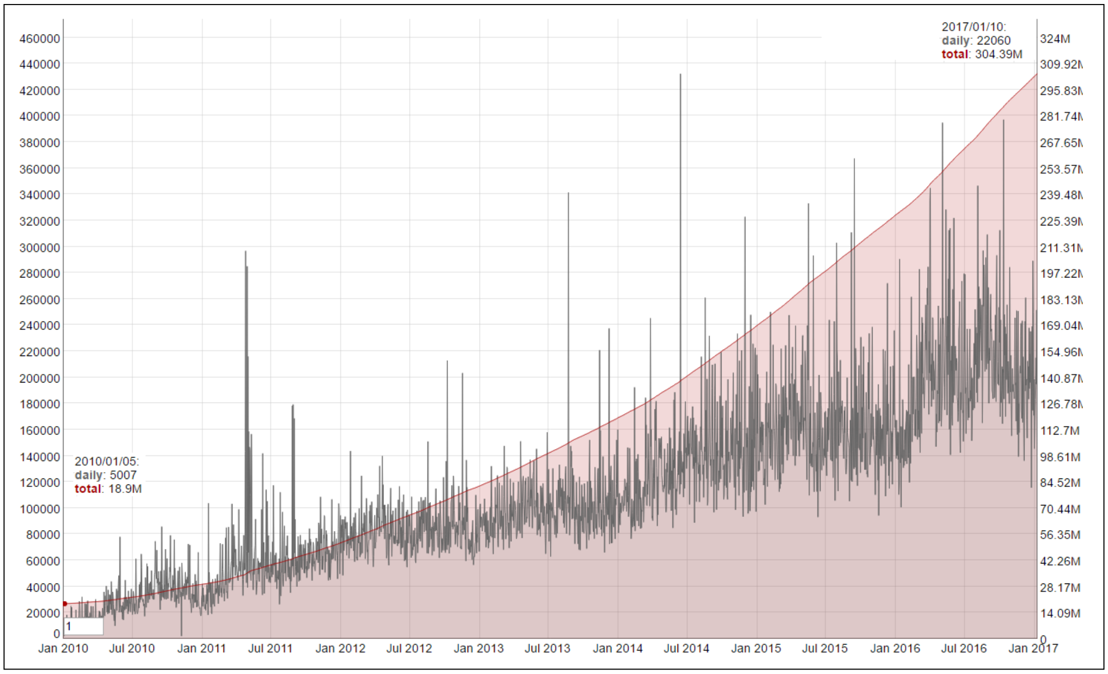


Figure 2.6: Number of Access Points (APs) in millions from 1/5/2010 to 1/10/2017 (as reported by WiGLE.net) [12]

Wi-Fi operates in the 2.4 GHz ISM (industrial, scientific, and medical radio band) which is free of regulation and cost to operate in. According to FCC.gov, the only portion of DSRC that can operate in the ISM band is the 5.850 - 5.875 MHz band. There is however, regulation and licensing when it comes to operating and distribution of RSUs and OBUs. A quote from the FCC website states "We license DSRC Roadside Units (RSUs), communication units that are fixed along the roadside, under subpart M (Intelligent Transportation Radio Service) of Part 90 of the Commission's Rules. We require licensees to register RSUs by site and segment(s). We license On-Board Units (OBUs), in-vehicle communications units, by rule under new subpart L of Part 95 of our Rules. Governmental entities will be authorized a geographic-area license based on that entity's legal jurisdictional area of operations. Non-governmental entities, will be licensed based on each applicant's area-of-operation i.e., by county, state, multi-state, or nationwide" [13]. While there are portions of the DSRC spectrum that can be freely accessed there are still many regulatory steps that must take place to deploy any testable solution(s). The authors in [14] breakdown the economics of distributing an ITS based in UHF (Ultra High Frequency), DSRC, Wi-Fi, and LTE access networks.

Cost factors	Symb.	Unit	UHF	Wi-Fi	DSRC	LTE
Number of BS	N_{BS}	#	9	3000	1500	70
BS Price	c_{BS}	mil. Rub.	1.2	0.05	0.55	2.57
Cost of frequency resource	c_{freq}	mil. Rub.	0.15	0.1	0.1	0.14
Cost of project analysis	$c_{pre-proj}$	mil. Rub.	0.167	0.01	0.01	0.07
Cost of network design, construction and launch	c_{deploy}	mil. Rub.	0.3	0.1	0.1	0.5
Cost of user equipment	c_{UE}	mil. Rub.	0.028	0.003	0.05	0.01
Number of UE	N_{UE}	#	5000	5000	5000	5000

Figure 2.7: Capital Expenditures for Public ITS Radio Area Network (RAN) Based on Different Wireless Access Technologies [14]

The cost of Wi-Fi base station is substantially lower than all the other wireless options due to the wide availability of OTS (off-the-shelf) Wi-Fi radios and other components. While this paper concluded that UHF was the most economical choice for their region this paper does illustrate that Wi-Fi, when compared to other wireless access technologies, does provide an economical route for widespread deployment and use.

The second advantage of a Wi-FI based ITS versus using the established standards is the ease of deployment. To test any solution developed for a Wi-Fi based ITS one only needs to purchase a standard Wi-FI router that is capable of operating in the ISM band. The equipment cost that would ensure the deployment of this solution would be minimal. While the current 802.11p standard is based on the WiFI protocol stack it still inhibits researchers from testing and deploying their solutions due to regulations and costs associated with testing the required equipment. The RSU and OBU in Fig. 2.1a and Fig. 2.1b would prove difficult to deploy given the amount of RSUs needed to test any tracking or avoidance algorithms. In contrast, Wi-Fi access points provide a more cost-effective solution to test new algorithms. The authors in [15] provide a comprehensive overview of what enabling technologies would be required to widely distribute an Internet-of-Vehicles (IoV) based ITS. The paper highlights the challenges that accompany distributing ITS at scale and using a Wi-Fi based VANET architecture. Fig. 2.8 illustrates three options for a Wi-Fi based ITS. While this paper proposes utilizing WAVE as its primary communications source, modification to use the simpler 802.11 b/g/n standard / protocol stack could be used to offset some of the challenges mentioned in the paper.

2.3.1.2 Wi-Fi Based ITS Disadvantages

While Wi-Fi offers many advantages over other wireless access technologies it also exhibits some major drawbacks when it comes to widespread and high density deployment as an ITS alternative. There are two major disadvantages that potentially inhibit Wi-Fi from being established as a robust ITS wireless alternative. The first disadvantage, mentioned by the authors in [16], is the throughput of Wi-Fi in high vehicle density scenarios. This research concluded that around 5 clients was the maximum number that could connect to a single AP before the throughput was compromised. This is a major drawback since the density of vehicles in an urban area would far exceed the 5 client limit. Even with multiple APs spread throughout the area it would be difficult to ensure high bandwidth for each vehicle in the network.

The second major drawback for using Wi-Fi as the primary ITS system is channel congestion. It is well known that the 2.4GHz band is very crowded with all Wi-Fi, Bluetooth devices, phones, ZigBee radios, etc. that occupy the same frequency space. This channel congestion can lead to high latency propagation of time critical data.

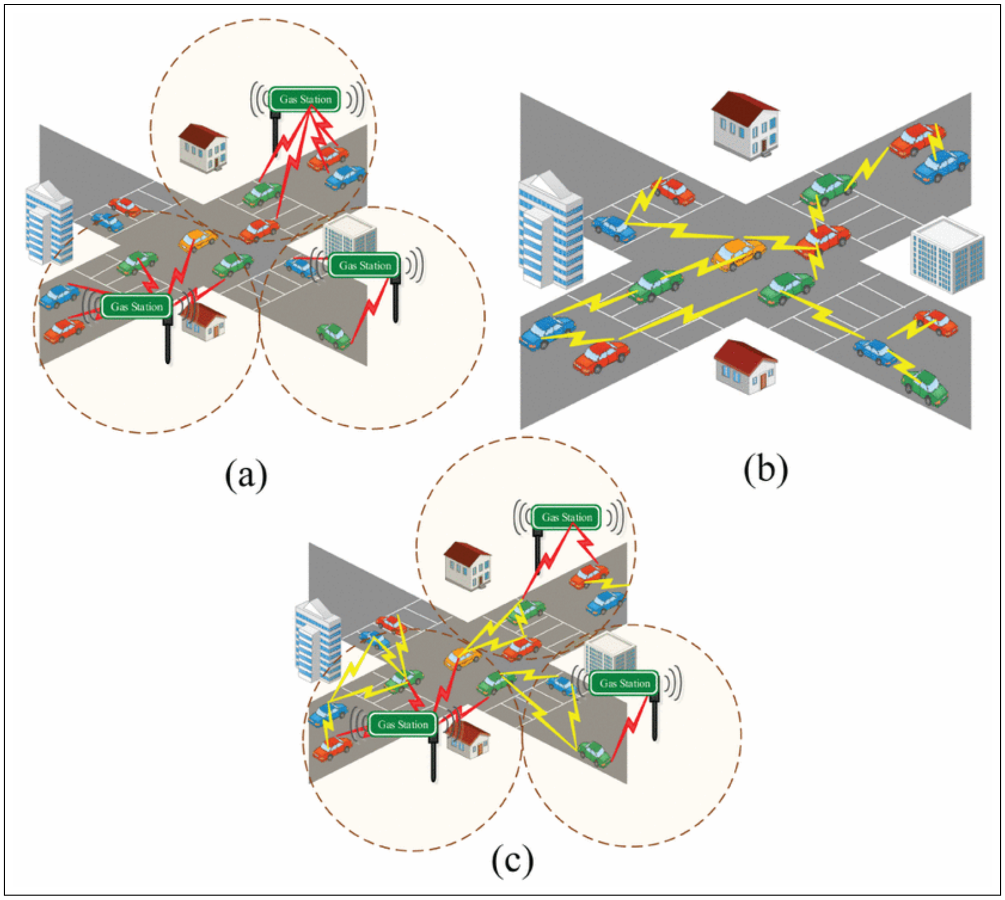


Figure 2.8: The Communication Architecture of VANETs: (a) WAVE Based Wi-Fi; (b) Adhoc; (c) Hybrid [15]

This increases the probability of accidents and erroneous information when solely relying on an Wi-Fi based ITS. The current DSRC channels that operate in the less congested 5.9GHz band aims to mitigate the problem of traffic congestion.

2.3.1.3 Wi-Fi Based ITS Case Studies

The authors in [14] discuss the economics of using a Wi-Fi based ITS versus the 802.11P and DSRC standard. The chart in Fig. 2.9 shows a comparison between standards that are currently being used for ITS development. This comparison shows that Wi-Fi offers some advantages over DSRC. These include a higher throughput, similar delay times, and effective range.

The authors in [17] compared Wi-Fi and WiMAX as viable solutions for V2V and

V2I communication. Their results indicate that, given the two metrics of throughput and latency, Wi-Fi provided satisfactory performance given a range of less than 100 meters. This also assumed the traffic on the 2.4GHz channel is minimal in the surrounding area.

A Wi-Fi based ITS can also be used in non-standard vehicle applications. The authors in [18] experimented with using Wi-Fi LoS (Line-of-Sight) and non LoS for increasing safety and production with mining vehicles. They cited the economics of using OTS Wi-Fi based radios as their reason for using this standard as opposed to WAVE and other ITS specific protocols and radios. The research concluded that Wi-Fi was a viable solution for V2V communications with the mining vehicles. The researchers conceded that these vehicles operated at a much lower speed and density than traditional highway vehicles. This work does provide a viable application for Wi-Fi based ITS even if the solution is specialized to scenarios such as these.

The authors in [16] explore the feasibility of using the 802.11a (WLAN) standard as a viable ITS solution. The results of this research indicate that, given a low number of 5 clients on a single Access-Point (AP) that the 802.11a standard had an average throughput of 10 Mbps even at speeds in excess of 90kmph (kilometers per hour).

The authors in [19] and [20] researched the idea of using Wi-Fi direct to facilitate the relaying of information from one vehicle to another. Wi-Fi direct enables devices to communicate with one another without the need of a centralized AP. The authors proposed that, before the widespread adoption of 802.11p OBUs, smartphones could be used as a means to facilitate messages between vehicles. In a later section of this paper the advantages of using a cellular network will be addressed, however, this research focused on Wi-Fi direct as the primary mover of data between moving vehicles. They concluded that the effective commutation distance between vehicles was limited to approximately 50 meters that the relatively low 30ms latency was low enough to facilitate most ITS messages effectively.

Characteristics	UHF [10]	Wi-Fi [11]	DSRC [3]	LTE [12]
Frequency bands, MHz	385-388 442-450	2400-2483 5625-6050	5855-5925	3400-3600
Bandwidth	12,5/25 kHz	20 MHz 40 MHz	10 MHz 20 MHz	5 MHz 7 MHz 10 MHz
Multiple Access	TDMA	CSMA/CA	CSMA/CA	OFDMA
Throughput [a]	12 kbps	Up to 100 mbps	Up to 18 mbps	Up to 40 mbps
Set-up time, ms[a]	Up to 2000	1000	100	100
Delay, ms [b]	300	100	100	100
Range, km [a]	20	0.3	0.6	Up to 8
User mobility support [c]	poor	poor	Up to 80 kmph	Up to 160 kmph
Scalability [c]	poor	poor	poor	good
Pros (service perspective)	Compressed voice; Emergency services; Short data dissimilation;	All the planned services;	All the planned services; V2V and V2I	All the planned services;
Cons (service perspective)	No broadband services; Only V2I;	Short range; Only V2I;	Short range;	Only V2I;

[a] Measured
[b] Theoretical
[c] Empirical

Figure 2.9: Comparison of UHF. Wi-Fi, DSRC, and LTE for ITS Based Systems [14]

Based on the research in this area it seems that a Wi-Fi based ITS would be best suited for low density V2I communication in rural areas where the cost of expensive 802.11p RSUs could be offset by more economical Wi-Fi radios.

2.3.2 Smartphone / Cellular Based ITS

Smartphones have almost become an ubiquitous part of society. From the statistics illustrated in Fig. 2.10 it is evident that smartphones will soon overtake standard feature phones. A direct result of this market saturation of smartphones has been the processing and sensing power that individuals inside of vehicles have at their disposal. Smartphones typically come pre-installed with a wide array of sensors, such as accelerometers, cameras, GPS, Inertial Measurement Units (IMU), etc. The array of sensors that these phones posses provide a platform for researchers to rapidly disseminate information from one vehicle to another without the need for the expensive 802.11p based OBUs and RSUs. Smartphones can collect and transmit time critical sensory information about the speed and position of the vehicle in which it resides. The authors in [21] and [22] use the internal IMU to relay the vehicle's position and relative base frame so other more complex driving algorithms could be applied. While these authors aim was focused on safety based application, such as distracted driving and insurance telematics, this research could be applied to many navigation algorithms to expedite the process of deploying fully autonomous vehicles. Additionally, the signal strength and distribution of high bandwidth networks, such as the fourth generation (4G) Long Term Evolution (LTE) network, has allowed the infrastructure to be in place to use smartphones as the main means of relaying and communicating V2I and V2V information within a network of vehicles. There are two basic applications that are being researched that utilize the smartphone as the main means to communicate from V2V and V2I; they are safety applications and vehicle tracking / positioning. Both of these areas of active research will be discussed below along with the pros and cons of deploying a smartphone / cellular based ITS.

2.3.2.1 Smartphone / Cellular Based ITS Advantages

There are two main advantages of using a cellular based smartphone as the primary device to relay time crucial information to both the surrounding infrastructure and vehicles on the roadway. The first advantage is the shear number of devices that are available that enable ease of deployment of this method. One of the primary inhibitors to using the current ITS standards, such as WAVE and DSRC, is the task of equipping all vehicles on the roadway with devices that need to be correctly connected

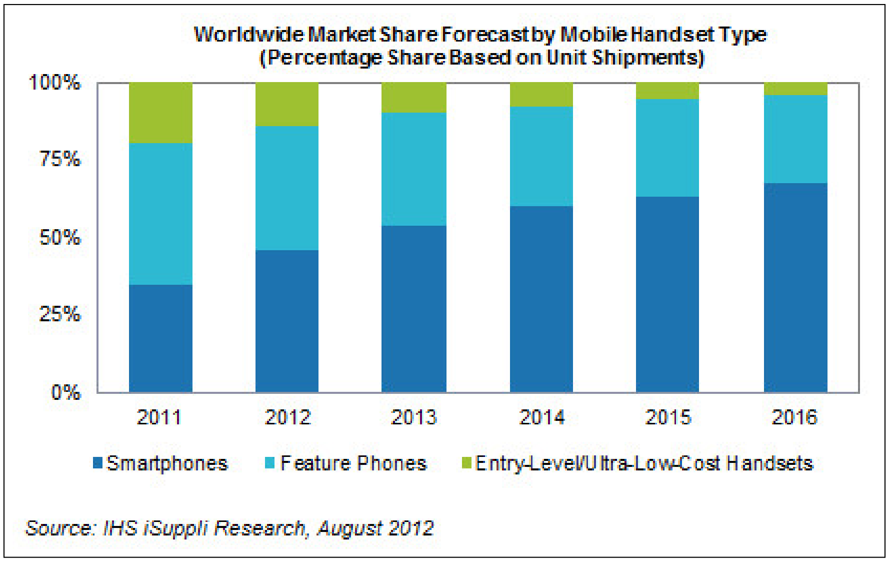


Figure 2.10: Worldwide MarketShare Forecast by Mobile Headset Type (Percentage Share Based on Unit Shipments) [23]

and calibrated to communicate with the surrounding RSUs that, in their own right, are difficult to set up. This is not an issue with a smartphone / cellular based ITS. Based on the forecast provided by Fig. 2.10 it is clear that soon over half of the vehicles on the road will have at least one occupant that owns a smartphone. These devices can quickly be configured to communicate any number of vehicle parameters with a simple installation of an application (App) or On-Board Diagnostic (OBD) Bluetooth reader. Fig. 2.11a and Fig. 2.11b show an example application that can transmit telemetry information from the vehicle and an example Bluetooth vehicle interface that connects with the vehicle's OBD port respectably.

The second main advantage of utilizing a smartphone / cellular network as a base for an ITS is the cost. Since most users already have a smartphone and the surrounding Wi-Fi and cellular network infrastructure is already in place, the potential to have a plug and play solution is high for this alternate ITS alternative technology. One of the main hindrances to deploying OBUs and RSUs to realize the 802.11p standard is the cost of each unit. Consumers would have to purchase these devices that do essentially many of the same tasks that their smartphone could theoretically

(a) SensoDuino Application (App)

(b) ELM 327 Bluetooth OBDII Vehicle Diagnostic Tool

Figure 2.11: SensoDuino App and ELM327 Bluetooth Adapter [24]

perform. With the onset of a fifth generation (5G) cellular network service on the horizon many researchers have initiated the task to test the validity of using it as the primary method of V2I and V2V communications [25–29].

2.3.2.2 Smartphone / Cellular based ITS Disadvantages

The prospects of using the already established cellular network infrastructure as the primary method of V2I and V2V is high and not without its shortcomings. There are three main disadvantages to using a smartphone / cellular network as the sole provider of V2V and V2I. The first disadvantage is the relatively high latency levels when connected to the slower third generation (3G) network as opposed to the higher bandwidth and lower latency 4G LTE network. Vehicles are traveling quickly and thus any application that transmits time critical information must be low latency in order for the message to remain relevant. While research has shown that 4G LTE networks can relay high bandwidths worth of information, however, it falls short when the data is time critical. [30] Along these same lines, wide spread data is not currently available as to what would transpire if all vehicles on the roadway transmitted high data rates concurrently.

The second disadvantage of using a smartphone for a primary V2V or V2I device is the infrastructure load. Much of the research that has been done that has investigated the feasibility of using cellular networks as the foundation for a V2V and V2I system has involved validating their tests with a low number of active nodes in the network at a given time. If cellular networks would be the primary ITS solution further tests would need to be executed to determine the data rates at more realistic vehicle density levels.

Security concerns is the last disadvantage of using a cellular based smartphone ITS solution. While the prospect of using the existing cellular infrastructure to relay life saving information is a noble one, however, as with any enabling technology malicious activity is to be expected and addressed. There are many researchers trying to tackle this problem since even if smartphones are not used as a primary mover of ITS data they will undoubtedly be used to augment and aid the ultimate and final form that a national wide ITS takes on [31–33].

2.3.2.3 Smartphone / Cellular Based ITS Case Studies

There are two primary paradigms that are actively investigated by researchers. The first is using the smartphone / cellular network for vehicle monitoring and safety reporting that would potentially augment some of the computational load of a distributed ITS and using cellular based smartphones as the primary method for distribution of a widespread ITS. Each of these methods face similar issues to see them to fruition.

The authors in [34–38] have researched the feasibility of using a cellular / smartphone based ITS for vehicle and driver monitoring scenarios. While these methods would help augment an ITS they would not be solely responsible for relaying all time critical information to the V2I or V2V systems. While vehicle and driver monitoring are crucial components of using a smartphone to aid in a widespread ITS there are a few interesting applications that have been researched and developed. They include

Table 2.1: Empirical Comparison between Three Alternative ITS Solutions and the Established Standard

	Alternative ITS Technology			
Metric	**Wi-Fi**	**Smartphone / Cellular**	**ZigBee**	**Planned DSRC / 802.11P**
Key Advantage	Operates in the ISM band	Existing Infrastructure	Decentralized	Robust Standards
Key Disadvantage	Channel Congestion	Network Load	Low Range	OBU and RSU Adoption
Deployment Cost	$$	$	$$$	$$
Scalability	Poor	Good	Poor	Medium
Security	Medium	Medium	Low	Medium
Consumer Cost	$	$	$$	$$$

a driver to infrastructure interaction system for a motorcycle [39] and some unique solutions for protecting pedestrians [40, 41].

Based on the research in this area it seems that a smartphone / cellular based ITS would be best applied as an aid for another overreaching ITS system. While a smartphone / cellular based ITS would be easily deployed and could be secured there is a lack of data that suggest the network would be able to handle the additional influx of vehicles communicating alongside the normal consumer voice and data load.

2.3.3 ZigBee Based ITS

ZigBee radios are based on the 802.15.4 IEEE standard that operate in the 2.4 GHz ISM band. The concept of these devices was to create a low power radio that could form a Personal Area Network (PAN) that could communicate with or without a centralized system. In 2007 the ZigBee alliance introduced ZigBee Pro that incorporate more advanced routing algorithms. These radios tend to be low cost, however, they require a steep learning curve to configure properly. The active research in this area tend to focus on supplementing a ZigBee network to accomplish low level tasks. The decentralized structure provides opportunities to deploy these systems in remote or rural areas. The advantages and disadvantages of this alternate method of ITS deployment is investigates and reported in the following two sections.

2.3.3.1 ZigBee Based ITS Advantages

An ITS based on a ZigBee network would have one key advantage over other alternative techniques, decentralization. The ZigBee protocol allows for the rapid deployment of low power and relatively inexpensive radio modules to be deployed in remote and urban locations. Many of the other alternate techniques discussed in this paper require a vast network of towers to distribute their systems, however, under the right conditions a ZigBee network can be deployed with very little infrastructure and able to run efficiently due to its inherent low power design. Research has also been conducted that discuss using ZigBee networks to reduce the number of wires in a vehicle by wirelessly transmitting some of the information from the vehicles' sensors to the electronic control unit (ECU) [42, 43].

2.3.3.2 ZigBee Based ITS Disadvantages

The main disadvantage of using a ZigBee based ITS would be the limitation of active users. Since ZigBee is based on a PAN it does not perform well when subjected to high user loads. This would create a large bottleneck, especially in an urban environment. In addition, the routing protocols that ZigBee and ZigBee Pro utilize tend to favor large overhead and data conservation as opposed to the high speed and time critical data that is needed for an effective ITS.

2.3.3.3 ZigBee Based ITS Case Studies

The research community has focused on specific applications regarding ZigBees that target more in the realm of supporting an ITS rather than using them as the principal method of distribution. The main application that ZigBees are utilized for, based on the literary search, is for vehicle detection, classification, and tracking. The authors in [44–50] highlight this use. As with many alternative ITS technologies safety is always an aspiring goal, the authors in [51, 52] focused their research on applying a ZigBee network to vehicle and driver safety applications. Researchers have also validated that ZigBee can operate efficiently even in the presence of other conflicting devices such as Bluetooth that is present in the vehicle [53]. The authors in [54] embrace the concept of utilizing Zigbees in a rural setting as they tracked the location of a large piece of agricultural equipment.

2.4 Analysis of Approaches & Assessment of Feasibility

The three alternative methods for deploying an ITS discussed in this paper highlights a need for a cohesive and quick deployment of enabling technologies to ensure that a widespread ITS is possible in the foreseeable future. The author, based on the research presented in this paper, has summarized the key advantages and disadvantage for all three alternative techniques in Table 2.1. The table also gives an empirical comparison of the three technologies based on the metrics of scalability, security, deployment costs, and consumer costs.

The author concluded that, based on the evident data, a hybrid solution could be developed that exploits the advantages for each alternative technique. It is vital to understand that these techniques are not meant to replace the 802.11p and DSRC standards but they are intended to augment and thus expedite the deployment process of a nationwide ITS.

2.4.1 Authors Relative Research

The author is currently investigating a hybrid approach that could utilize the strengths of many of these aforementioned methods. The author is proposing a distributing a ZigBee based Wireless Sensor Network (WSN) that is deployed along the roadway to report back roadway conditions such as slope and curvature to the vehicle. The vehicle also has a ZigBee radio for inter-vehicle and V2I communications. A smartphone App is used to connect to the vehicle's ECU to classify and relay time critical diagnostic and telemetry information to the WSN. This hybrid approach could expedite the widespread distribution of a nationwide ITS.

2.5 Conclusion

From the research that has been discussed in the literature review, the research community is actively seeking alternate methods to deploy a nationwide ITS. The benefits of having a far reaching and interconnected ITS are significant and a connected future is on the horizon that will likely decrease fatalities due to vehicle collisions and aid in decongesting our roadways as drivers and vehicles have more information at their disposal. While all of these alternative wireless access technologies will not singularly solve all the issues related to the deployment of a nationwide or even global ITS, however, research in this area has shown that the best possible solution could potentially be exploiting the strengths that each of these alternative wireless technologies can contribute in order to enable the vision of a global or even local ITS to become a reality.

CHAPTER 3: PROTOCOLS AND STANDARDS

3.1 Drive-by-Wire (DBW) Systems

Drive-by-Wire, sometimes referred to as x-by-wire, is a network of “by-wire ”systems that enable various vehicle functions, such as steering, throttle, and brake, to be controlled via electrical signals rather than with physical connections. Many car manufactures have spent years developing these technologies, first introduced in the aviation sector [55], this technology enables digital control of the various drive, power, and steering systems of an automobile. There are many advantages to these by-wire systems, however, the most intriguing benefit is the ability to control all aspects of the vehicles' movement with digital signals. The older analog systems that currently control the basic features and functions of a vehicles' movement do not integrate well into a centralized digitally vehicle control system. The following sections examine the three main by-wire systems currently in development or already implemented by major car manufactures. It is important to note that legacy vehicles will not have these technologies or will have a reduced set, however, newer vehicles, with this technology, will drive the future of a fully-autonomous vehicle landscape.

3.1.1 Steering-by-Wire (SBW) System

Steering-by-Wire (SBW) is a "by-Wire" driving system that has been around on most high end production vehicle for the past few years. The aviation equivalent of "fly-by-wire" has been around since the early Apollo spacecraft era and was fully introduced into commercial air craft with its installment in the Concorde aircraft in 1969 [55].

Traditional steering, in most vehicles, is provided by a rack-and-pinion system.

With a rack-and-pinion steering setup the movement of the steering wheel, by the driver, turns a shaft that is connected to a rack via gears and a fluid assist system. The turn of the drivers steering wheel translate to wheel movement directly. There are, however, some lag and discrepancies between the drivers intended movements and the actual movements of the vehicles wheels. The image in Figure 3.1 illustrates a generic rack-and-pinion system. Vehicles equipped with a rack-and-pinion steering system are typically difficult to transform into semi-autonomous vehicles since the relative position of the wheels and steering cannot be directly measured in real-time. Vehicles that incorporate features, such as hands free parallel parking, often are equipped with steering columns that contain high precision encoders to keep track of the exact position of the steering wheel and thus the relative location of the wheels. SBW systems may overtake traditional steering systems in the near future due to their faster response time to drivers actions and their ability to integrate easier in a semi-autonomous driving environment.

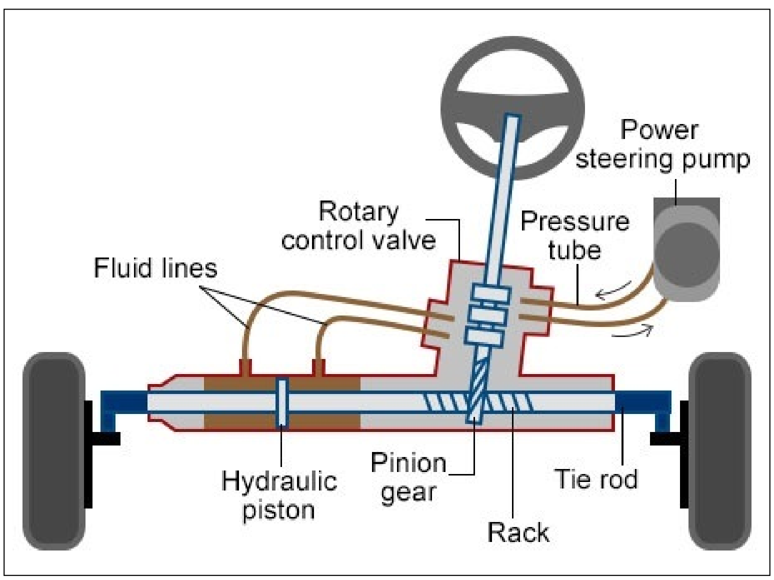


Figure 3.1: Components of a Rack-and-Pinion Steering System [56]

Nissan was the first major car manufacture to introduced and implement a true SBW technology for a production vehicles back in 2013 [57]. The image in Figure 3.2

shows the SBW system developed by Nissan. It consists of four main components that are show in the image and listed below [57]

1. **Steering Force Sensor:** Playing two roles, this unit sends commands to the control modules and acts as the driver's feedback source by varying resistance to the wheel.

2. **Clutch:** Most of the time it's open. Faults in the electronics force it closed, creating a solid mechanical connection between the steering wheel and the rack.

3. **Control Modules:** This trio controls the electric-assist motors and the steering-force sensor. They also act as redundancies; you know, for safety.

4. **Steering-Assist Motors:** Two of these smaller motors are cheaper than one large one. Plus, this arrangement frees some space for a low-slung longitudinal engine.

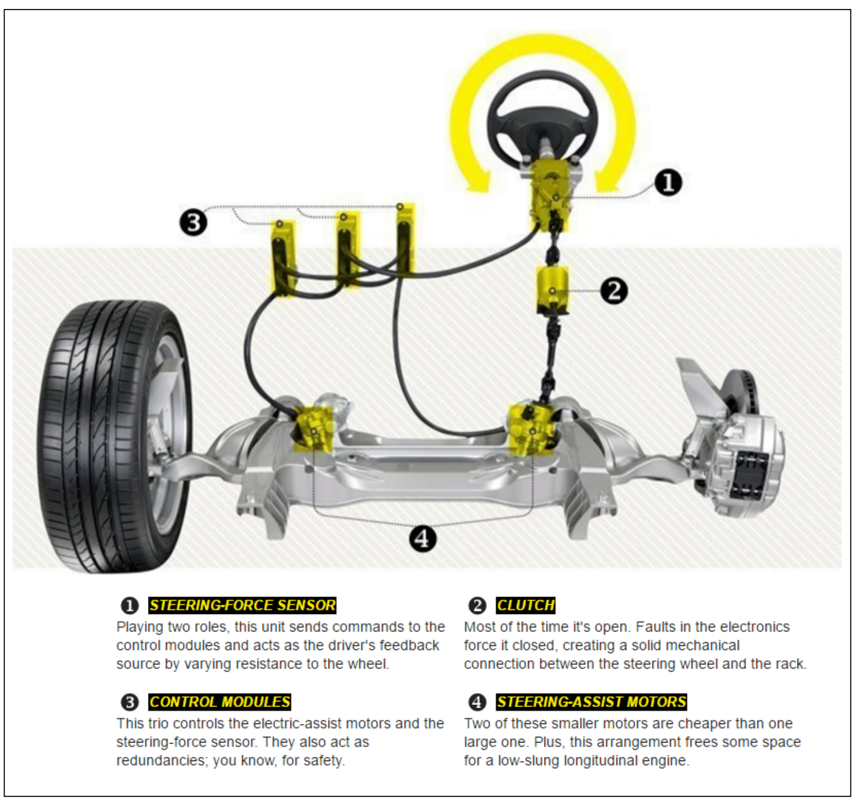


Figure 3.2: Components of a Steering-by-Wire System [57]

SBW systems, such as the one implemented by Nissan, has many benefits to the driver and for vehicle autonomy. Some of those benefits include: isolation from road impacts, elimination of steering flex (tendency of steering wheel to shake when traversing uneven terrain) and lash (moving steering wheel through a portion of a revolution with no resulting wheel change or movement), and infinite steering ratios. Having a SBW system implemented in a vehicle also enables the vehicle to switch to a semi autonomous state (such as automatic parallel parking) easily since the on board computer of the vehicle knows the exact position of the steering column and wheels at any given moment in time. It is important to note that the current iterations of a SBW system incorporate a ”fail-safe” mode where a clutch is engaged that enables a one-to-one steering movement and control as seen in typical rack-and-pinion driving systems.

3.1.2 Brake-by-Wire (BBW) System

Brake-by-Wire (BBW) systems are the newest of the ”by-Wire” technologies to be offered to the in the consumer vehicle market. The research in [58–63] show many early developments that led to the current state of BBW systems. Many companies such as BMW, Mercedes Benz, and Toyota have been incorporating this technology in all of their hybrid and hybrid-electric vehicles for many years with the most notable of these being the Toyota Prius [64]. The image in Figure 3.3 shows the basic components of a BBW system. The Brake Controller Unit (BCU) interprets the signals from the pedal and translates that into braking action via the hydraulic pressure modulator thus completely isolating the braking action from any direct mechanic coupling to the driver.

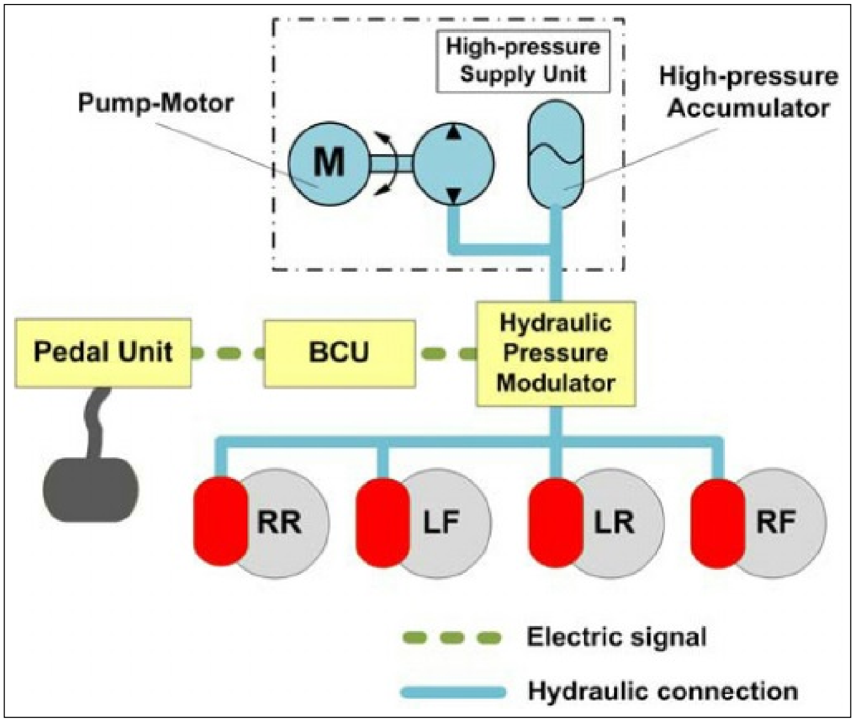


Figure 3.3: Components of a Brake-by-Wire System [65]

Some of the advantages of having a vehicle equipped with a BBW system include: faster braking response time, lower production cost (fewer moving mechanical parts), and no vibration felt at the driver's pedal to name a few, however, the most appealing feature of this technology is the ability for the vehicles on-board computer to use external sensory information (either via a camera or range sensing system) to automatically stop the vehicle without any driver action. This safety feature is slated to save many lives and will continue to be distributed to vehicles of every class and price level. [66] This adaptive braking technology is crucial for vehicle to posses to take advantage of safety signals sent from a centralized ITS that will inevitably lead to fewer fatalities on the motorways.

3.1.3 Throttle-by-Wire (TBW) System

A Throttle-by-Wire (TBW) system was first introduced in an automobile by Bavarian Motor Works (BMW) for their 7 series sedans back in 1988 [67]. This system was the first "by-wire" system that was widespread upon its release. This type of by-wire system provided many advantages both to the vehicle manufacturer and the consumer. The lack of a physical cable meant lower production costs for the manufacturer and the consumer experienced elevated throttle response times and it paved the way for more advanced adaptive cruise control system that are implemented in many higher end vehicles today. The image in Figure 3.4 illustrates the three main components of a TBW system. A summary of each of these sections are provided below: [68]

1. **Throttle Actuator Control Module:** The Throttle Actuator Control Module uses the vehicles electronic control module (ECM) to calculate and control the position of the electronic throttle body unit. This in-turn provides less or more air to the vehicle's manifold.

2. **Electronic Throttle Body:** The electronic throttle body consists of a direct current (DC) motor attached to the side of the standard throttle body. In a typical non by wire system there would be a physical cable attached to the throttle body directly from the pedal. The throttle body is responsible for controlling the amount of air intake to the manifold of the engine that translates into more or less power to the wheels.

3. **Throttle Position / Accelerator Pedal Position Sensor:** The throttle position / accelerator pedal position sensor is used to translate driver pedal movement into an electrical signal. Typically there are two or more redundant systems that either increase or decease in voltage depending on the position of the pedal.

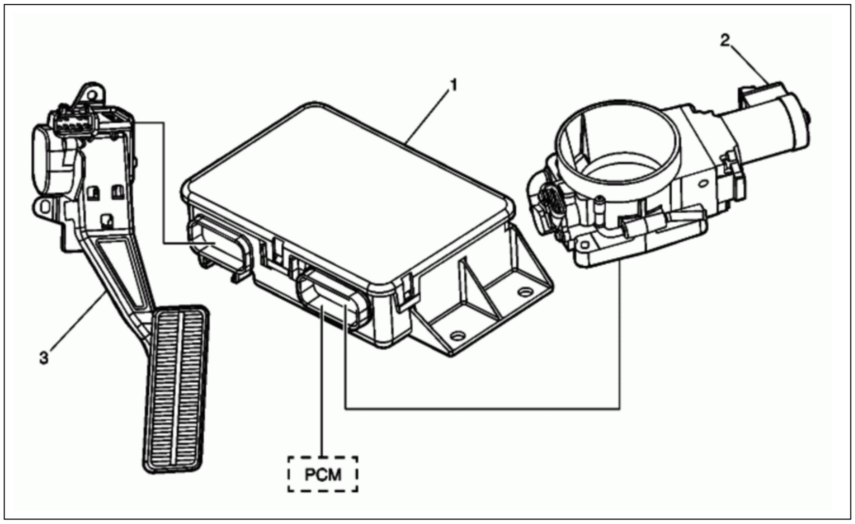


Figure 3.4: Components of a Throttle-by-Wire System [68]

The image in Figure 3.5 shows a functional block diagram of a TBW system. It is important to observe that this by-wire system does not have a fail safe backup. If there is a problem in translating relative throttle position to or from the vehicle's ECU then the vehicle will speed cannot be controlled.

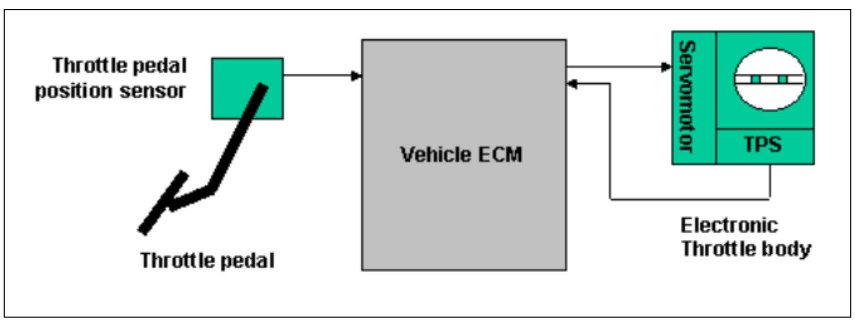


Figure 3.5: Throttle-by-Wire System Block Diagram [67]

There are many advantages to a TBW system which include: better fuel-to-air ratios (resulting in better fuel mileage), a background "service" that can deal with drastic torque changes (i.e. rapid acceleration / deceleration) , and it contains less moving parts (resulting in lower manufacturing costs to the manufacturer). A TBW can also be useful in transitioning a vehicle to a semi-autonomous state. Most modern vehicles capitalize on this automation in the form of cruise control. Since the throttle body has a DC motor than can be controlled via a digital signal the cruise control system can react quickly and adapt to surrounding roadway changes (hills, valleys, etc.)

3.2 NHTSA Vehicle Automation Levels

The National Highway Traffic Safety Administration (NHTSA), in 2013, issued guidelines on classifying vehicle automation [69]. The NHTSA divides vehicles into five levels of automation (Levels 0-4). A breakdown of what constitutes each level follows: [69]

Non-Automation (Level 0): The driver is in complete and sole control of the primary vehicle controls - brake, steering, throttle, and motive power - at all times.

Function-specific Automation (Level 1): Automation at this level involves one or more specific control functions. Examples include electronic stability control or pre-charged brakes, where the vehicle automatically assists with braking to enable the driver to regain control of the vehicle or stop faster than possible by acting alone.

Combined Function Automation (Level 2): This level involves automation of at least two primary control functions designed to work in unison to relieve the driver of control of those functions. An example of combined function system would include a vehicle that featured adaptive cruise control in combination with lane variance monitoring.

Limited Self-Driving Automation (Level 3): Vehicles at this level of automation enable the driver to cede full control of all safety-critical functions under certain traffic or environmental conditions. This automation level relies heavily on vehicle monitoring systems to ensure partial control can be sustained or if conditions require transitioning back to full driver control. The driver is expected to be available for occasional control, but with sufficiently comfortable transition time. The Google car is an example of limited self-driving automation.

Full Self-Driving Automation (Level 4): The vehicle is designed to perform all safety-critical driving functions and monitor roadway conditions for an entire trip. Such a design anticipates that the driver will provide destination or navigation input, but is not expected to be available for control at any time during the trip. This includes both occupied and unoccupied vehicles.

These automation levels set forth by the NHTSA are useful to evaluate what automation class a vehicle resides in. To enable a fully integrated ITS a framework that encompasses all automation levels will need to be developed to ensure all legacy, present, and future vehicles can collaborate and communicate with each other and their surrounding infrastructure.

3.3 IEEE 802.11p, VANET, WAVE, WSMP, and DSRC

These standards and protocols are discussed in the background section of the previous works section of this dissertation. Please refer to that section for an in depth overview of these vehicle communication specific technologies and standards. [70]

3.4 Roadway Grade Classification

The grade of a highway is a measure of the rise (or fall) of its incline (or decline). There are whole fields of study dedicated to calculating the safest possible grades for a given stretch of roadway. The Federal Highway Administration (FHWA) sends out best practices for local governments and municipalities to follow for new road construction, however, each state has its own guidelines for maximum and minimum roadway slope (grade) percentages [71]. The image in Figure 3.6 shows an illustration of road grades and slope percentages. While the chart goes up to 90° it is important to note that the steepest recorded roadway, Baldwin Street in New Zealand, only has a slope of 35% or 19°. A completely flat road would have a slope (grade) of zero.

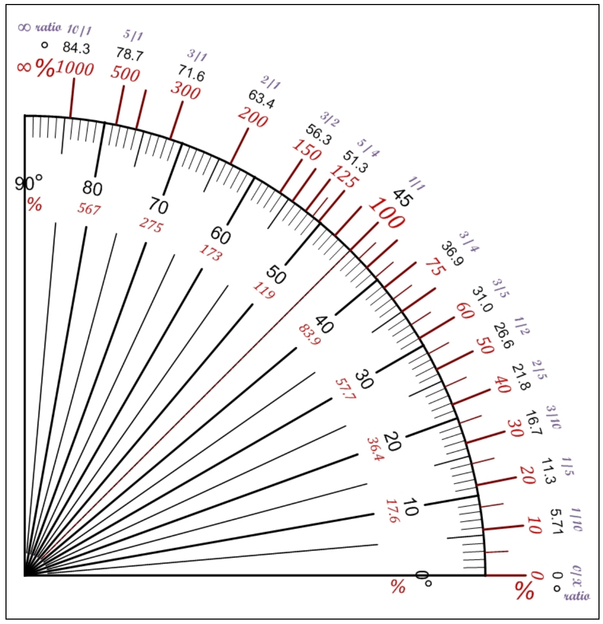


Figure 3.6: Grades (Percentages), Angles in Degrees and Ratio [72]

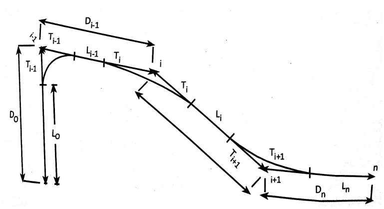
There are many guidelines that govern how fast a slope can descend or ascend, however, this research has not found a definitive classification technique/chart that could be used to encompass all roadway slopes (grades). This is vital to integrating older and newer vehicles into an ITS as every vehicle's dynamics will be slightly different for each grade or slope level. A characterization technique that could classify both the roadway's slope and curvature would enable a standardizing method to be developed that could dictate how a vehicle should approach, via throttle and braking control, to successfully traverse a given terrain.

There are many mapping services, such as Apple and Google, however, even these broad reaching services do not provide the driver with real time slope (grade) information for the intended path to traverse.

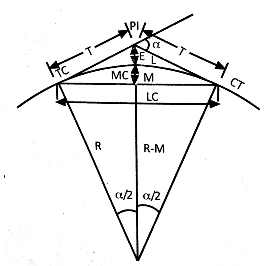
3.5 Roadway Curve Classification

There are many methods to calculate the curvature of a given roadway. Some of those methods include: Tape-Measure, Triangulation, Formula, Map, Pace, and Angle-of-Slope methods [73]. There are whole sectors of Civil Engineering dedicated to the design considerations of roadway curvature and slope. The image in Figures 3.7 - 3.8 show some of the levels of detail and scrutiny that goes into designing the curves and slopes of a highway. These methods are often aimed at increasing driver safety and nighttime visibility for the driver.

While these methods are typically standardized across both federal and local levels, this research did not find a comprehensive set of parameters that could breakdown curves into a simple classification system based solely on the curvature/curve angle of the whole or portion of the roadway.



(a) Example of Tangent Curve Segmentation



(b) Elements of a Circular Curve

Figure 3.7: Example of Tangent and Circular Curves [74]

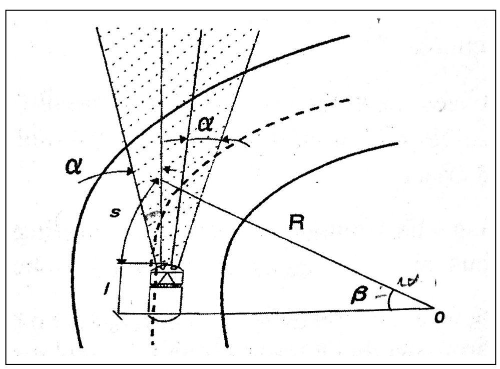


Figure 3.8: Horizontal Curve (Nighttime Consideration) [74]

Lombard Street, located in San Francisco, California. Is one of the crookedest roads in the U.S. While Lombard Street is not the crookedest street in the U.S. it is perhaps the most famous and was used as a case study in this research for both its extreme curves and quickly descending slope.

3.6 Vehicle Gradeability

Vehicles have what is called gradeability. It is defined as the highest grade a vehicle can ascend maintaining a particular speed [75]. Vehicles are typically designed to meet a minimum gradability standard. The size, weight, wheelbase, and other factors contribute to the gradeability a vehicle has. Most companies have their own standard that they want their vehicles to achieve. As an example, General Motors has the GMW16297 standard that defines the global vehicle top gear gradeability test. This parameter is important when designing an intelligent transportation system as all vehicles on the roadway will have differing degrees of gradeability that will need to be taken into account when designing an all encompassing ”smart highway” infrastructure.

3.7 Vehicle Classification

The images in Figures 3.9, 3.10, and 3.11 show the motor vehicle classifications according to the United States Department of Energy (USDOE), National Highway Traffic Safety Administration (NHTSA) handbook on the requirements of motor vehicles and motor vehicle equipment, and the Federal Highway Administration (FHWA), respectively.

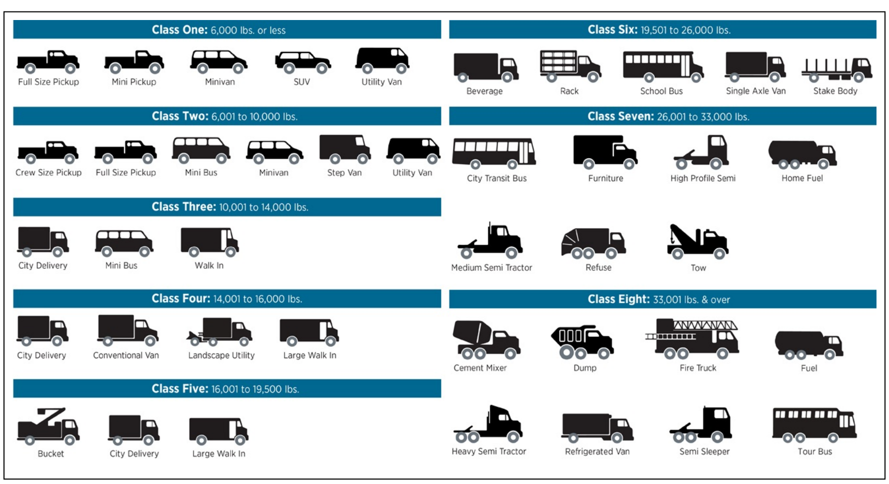


Figure 3.9: USDOE Vehicle Classification Chart (by GVWR) [76]

Classification	Definition
Passenger car	A motor vehicle with motive power, except a low-speed vehicle, multipurpose passenger vehicle, motorcycle, or trailer, designed for carrying 10 persons or less.
Multipurpose passenger vehicle	A motor vehicle with motive power, except a low-speed vehicle or trailer, designed to carry 10 persons or less which is constructed either on a truck chassis or with special features for occasional off-road operation.
Truck	A motor vehicle with motive power, except a trailer, designed primarily for the transportation of property or special purpose equipment.
Bus	A motor vehicle with motive power, except a trailer, designed for carrying more than 10 persons.
Motorcycle	A motor vehicle with motive power having a seat or saddle for the use of the rider and designed to travel on not more than three wheels in contact with the ground.
Motor driven cycle	A motorcycle with a motor that produces 5 brake horsepower or less.
Trailer	A motor vehicle with or without motive power, designed for carrying persons or property and for being drawn by another motor vehicle.
Low-speed vehicle	A motor vehicle, that is 4-wheeled, whose speed attainable in 1 mile (1.6 km) is more than 20 miles per hour (32 kilometers per hour) and not more than 25 miles per hour (40 kilometers per hour) on a paved level surface, and whose GVWR is less than 3,000 pounds (1,361 kilograms).
Pole Trailer	A motor vehicle without motive power designed to be drawn by another motor vehicle and attached to the towing vehicle by means of a reach or pole, or by being boomed or otherwise secured to the towing vehicle, for transporting long or irregularly shaped loads such as poles, pipes, or structural members capable generally of sustaining themselves as beams between the supporting connections.

Figure 3.10: NHTSA Vehicle Classification Types [77]

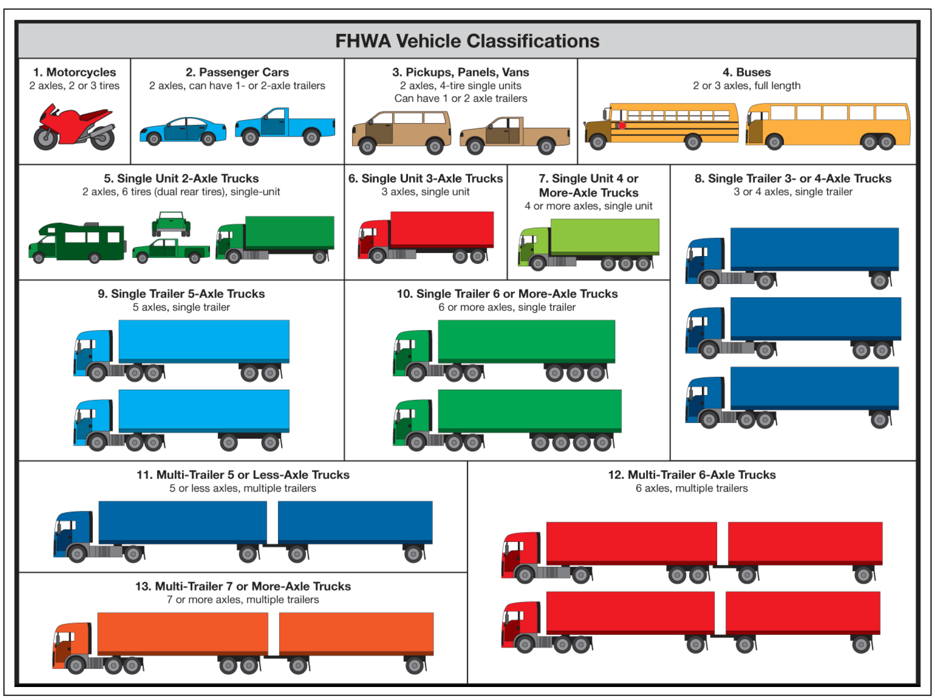


Figure 3.11: FHWA Vehicle Classification Chart [78]

There are multiple metrics that go into these classification techniques, some of which include, but are not limited to: the number of axles the vehicle contains, the Gross Vehicle Weight Rating (GVWR), the number of total occupants the vehicle can hold, and the total number of tires. Most of these classifications and classification techniques are implemented for government and insurance agencies to develop rules, regulations, emission standards, etc; however, this research did not find a de facto method to sub divide a class, such as a passenger car, into sub-classes. A sub-classification technique that could discern one passenger vehicle from another, based on a prescribed set of metrics, would not only increase the resolution of vehicle classes, it would also enable future autonomy research the ability to target certain sub-classes of vehicles with specific enabling technologies for that particular sub-class of vehicle.

3.8 Vehicle Identification Number (VIN)

According to the National Highway Traffic Administration (NHTSA) handbook on the requirements of motor vehicles and motor vehicle equipment:

> "Among other things, NHTSA's regulations at 49 CFR Part 565 require a motor vehicle manufacturer to assign to each motor vehicle manufactured for sale in the United States (US) a 17-character VIN that uniquely identifies the vehicle. The VIN must be correctly formatted and include a check digit in Position 9 that is mathematically correct under a formula that is included in the regulation. VINS are required to have 17 characters that do not include the letters I, O, or Q. Beginning with the 1980 model year, the VINs of any two vehicles manufactured within a 60-year period must not be identical. All spaces provided for in the VIN must be occupied by a character specified in Part 565 and the type face used for each VIN must be in capitals and use san serif characters. This means that the characters will not have fine lines or "serifs" finishing off the main strokes of the letters. The VIN of each vehicle must appear clearly and indelibly upon either a part of the vehicle, other than the glazing, that is not designed to be removed except for repair or upon a separate plate or label that is permanently affixed to such a part."

The guidelines laid out in this handbook also dictate how the contents of the 17 digit VIN are comprised. The image in Figure 3.12 shows the general VIN format that is laid out by the handbook.

Some important parameters to note is the new requirement of the vehicles make to be included in the second section of the VIN.

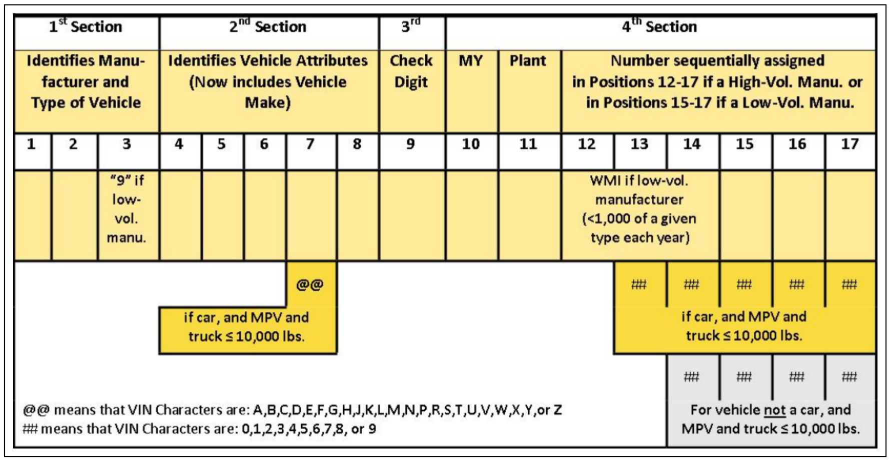


Figure 3.12: Breakdown of NHTSA Required Vehicle VIN Number [77]

New and current vehicle manufactures must adhere to this general VIN format, however, car manufacturer are given some liberty when it comes to certain sections of the VIN. The image in Figure 3.13 shows an example of attributes the manufacture can encode in digits 4-8 of the 17 digit VIN number. In this example the manufacturer can encode the engine size and the vehicles gross weight into these digits.

This research will use the 17 digit VIN as a method to potentially identity every vehicle in production based on the attributes listed in its unique 17 digit VIN. One of the advantages of using the VIN, aside from immediate identification, is that is give the infrastructure a method to quickly parse vehicles simply based on their unique VIN.

Passenger Car Vehicle Attributes That Must Be Identified							
VIN 4-8 Code	Make	Line or Model	Series	Body Type	Engine Type	GVWR Class @	Restraint System @@
RP1A3	USA Car Co.	Super	LV	3 Dr. Coupe	2.4L 4 cyl. 180hp gas	A	A
RP1B3	USA Car Co.	Super	DV	4 Dr. Sedan	2.4L 4 cyl. 180hp Gas	B	B

Manufacturer's Notes to Lookup Table:
@GVWR Class is from 49 CFR 565.15 "Table II - Gross Vehicle Weight Rating Classes"
A= Not greater than 1360 kg. (3,000 lbs.)
B=Greater than 1360 kg. to 1814 kg. (3,001–4,000 lbs.)
@@Restraint System
A= Front: Seat Belt, Air Bag, Side Air Bag, and Side Curtain Air Bag (Driver and Passenger) and Rear: Seat Belt and Side Curtain Air Bag.

Figure 3.13: Example Manufacturer Attributes for VIN Digits 4-8 [77]

3.9 Controller Area Network (CAN)

The Controller Area Network (CAN, also known as CAN Bus) is a vehicle bus standard designed to allow electronic control units (ECUs) and devices to communicate with each other in applications without a host computer. It was developed by Robert Bosch GmbH. As an alternative to conventional multi-wire looms, CAN Bus allows various electronic components, such as ECUs, microcontrollers, sensors, actuators etc., to communicate on a single or dual-wire network data bus up to 1 Mbps [79]. The image in Figure 3.14 shows the typical configuration of a CAN Bus enabled vehicle. This image illustrates how all of the vehicle's various ECUs are connected to the CAN Bus and also connect to the vehicles On-Board Diagnostics port.

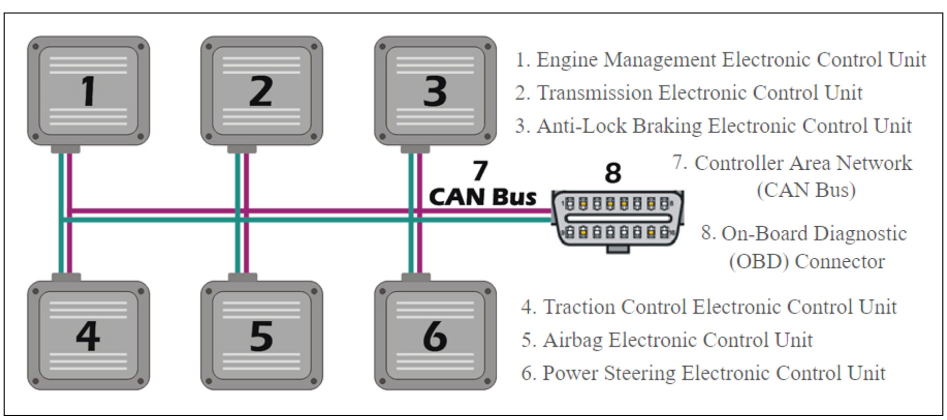


Figure 3.14: Example CAN Bus System Diagram [79]

3.9.1 Wiring Format

CAN Bus generally has four wires, CAN-High (CAN-H),CAN-Low (CAN-L), Ground (GND), and Power (PWR). CAN-H and CAN-L are the data lines that carry the differential signals and connect to all the devices in the network. The signals on the two CAN lines have the same sequence of data, but their amplitudes are opposite. For example, if a pulse on the CAN-H line goes from 2.5V (Volts) to 3.75V then the corresponding pulse on the CAN-L line goes from 2.5V to 1.25V (opposite from CAN-H). By sending the data in equal and opposite ways like this allows for greater noise immunity and therefore less chance of the data being corrupted [79]. The image in Figure 3.15 illustrates the aforementioned example and illustrates the recessive and dominant states of the signal. The status of the bit with the value of zero (2.5V differential voltage) is called the Dominant State, while the status of the bit with the value of 1 (0V differential voltage) is called the recessive state.

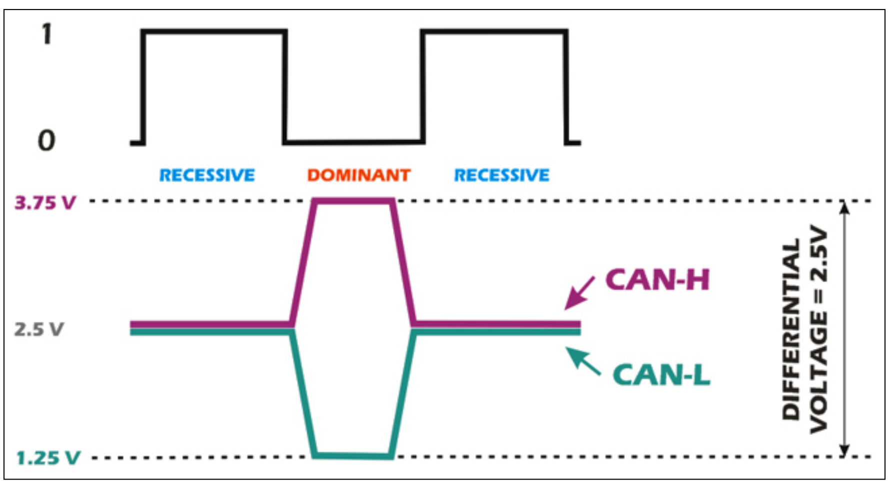


Figure 3.15: CAN Bus Differential Signal Illustration [79]

Another note to make about the CAN Bus is that it is a multi-master serial bus. Each communicating controller is called a node. Each node requires three elements: a microcontroller, a CAN Controller and a CAN Transceiver [80]. The bus also requires a bus terminal that is typically a 120Ω (Ohm) resistor. The image in Figure

3.16 shows the block diagram of all the necessary parts of a CAN bus including the two CAN Nodes.

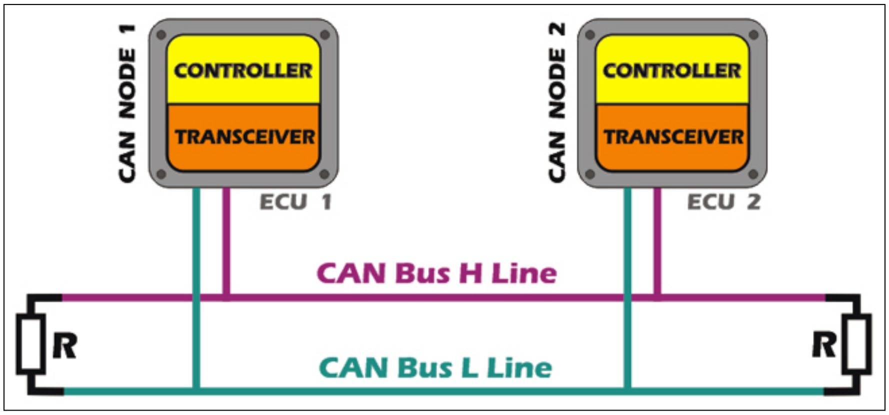


Figure 3.16: CAN Bus Node Components [79]

The following list outlines the main functions and features of each component [79].

1. **CAN Controller** receives the transfer data from the microcomputer integrated in the control unit/device (also known as CAN Node). The CAN controller processes this data and relays it to the CAN transceiver. Also, the CAN controller receives data from the CAN transceiver, processes it, and relays it to the microcomputer integrated in the control unit/device (CAN Node).

2. **CAN Transceiver** is a transmitter and receiver all-in one. It converts the data which the CAN controller supplies into electrical signals and sends this data over the data bus lines. Also, it receives data and converts this data for the CAN controller.

3. **CAN Data Bus Terminal** is a resistor (R) wih a typical value of 120Ω. It prevents data sent from being reflected at the ends and returning as an echo.

3.9.2 Frame Format

There are four frame types that comprise a CAN bus message. They are the following: [80]

1. **Data Frame** The frame that contains the data.

2. **Remote Frame** The frame that requests data with its identifier.

3. **Error Frame** The frame, transmitted by a node, to detect errors.

4. **Overload Frame** The frame used to insert delay(s) between the data and remote frames.

The image in Figure 3.17 shows a highlighted version of the CAN Bus frame format.

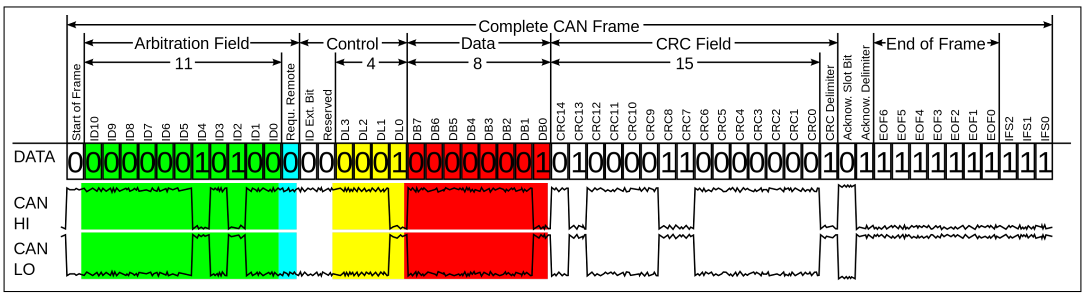


Figure 3.17: CAN Bus Base Frame Format (Without Bit Stuffing) [80]

The corresponding information in Tables 3.1 (Standard Frame Format) and 3.2 (Extended Frame Format) breakdowns the purpose of each section of the frame. The two frame types, standard (version 2.0A) and extended (version 2.0B) only differ in the size of the identifiers (ID) bits in the frame. The Standard CAN uses 11 bit identifiers in the arbitration field while Extended CAN supports a length of 29 bits for the identifier, made up of the 11 bit identifier (base identifier) and an 18 bit extension.

Table 3.1: CAN Base Frame Format [80]

Field Name	Lenght (bits)	Purpose
Start-of-frame	1	Denotes the start of frame transmission
Identifier (green)	11	A (unique) identifier which also represents the message priority
Remote transmission request (RTR) (blue)	1	Must be dominant (0) for data frames and recessive (1) for remote request frames
Identifier extension bit (IDE)	1	Must be dominant (0) for base frame format with 11-bit identifiers
Reserved bit (r0)	1	Reserved bit. Must be dominant (0), but accepted as either dominant or recessive
Data length code (DLC) (yellow)	4	Number of bytes of data (0-8 bytes)
Data field (red)	0-64 (0-8 bytes)	Data to be transmitted (length in bytes dictated by DLC field)
CRC	15	Cyclic redundancy check
CRC delimiter	1	Must be recessive (1)
ACK slot	1	Transmitter sends recessive (1) and any receiver can assert a dominant (0)
ACK delimiter	1	Must be recessive (1)
End-of-frame (EOF)	7	Must be recessive (1)

Table 3.2: CAN Extended Frame Format [80]

Field Name	Length (bits)	Purpose
Start-of-frame	1	Denotes the start of frame transmission
Identifier A (green)	11	First part of the (unique) identifier which also represents the message priority
Substitute remote request (SRR)	1	Must be recessive (1)
Identifier extension bit (IDE)	1	Must be recessive (1) for extended frame format with 29-bit identifiers
Identifier B (green)	18	Second part of the (unique) identifier which also represents the message priority
Remote transmission request (RTR) (blue)	1	Must be dominant (0) for data frames and recessive (1) for remote request frames
Reserved bits (r1, r0)	1	Reserved bits which must be set dominant (0), but accepted as either dominant or recessive
Data length code (DLC) (yellow)	4	Number of bytes of data (0-8 bytes)
Data field (red)	0-64 (0-8 bytes)	Data to be transmitted (length in bytes dictated by DLC field)
CRC	15	Cyclic redundancy check
CRC delimiter	1	Must be recessive (1)
ACK slot	1	Transmitter sends recessive (1) and any receiver can assert a dominant (0)
ACK delimiter	1	Must be recessive (1)
End-of-frame (EOF)	7	Must be recessive (1)

3.9.3 Data Flow Process

The data transfer process in a CAN network general consist of five stages: Supply, Sending, Receiving, Checking, and Accepting data. The image in Figure 3.18 shows an illustration of the data flow process.

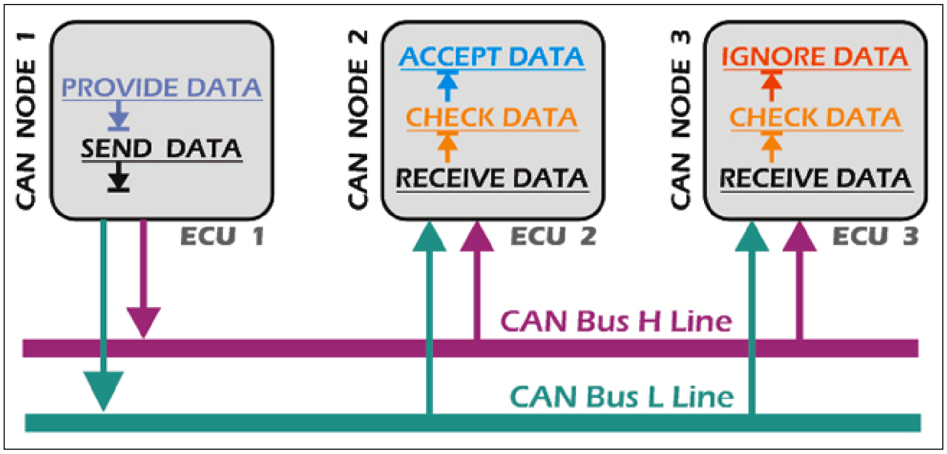


Figure 3.18: CAN Bus General Data Flow Process [79]

The following list details the general actions performed at each stage of the process [80].

Supplying Data: The CAN Node provides data to the CAN controller for transfer.

Sending Data: The CAN transceiver receives data from the CAN controller, converts it into electrical signals and sends them back into the network.

Receiving Data: All other CAN Nodes networked with the CAN data bus become receivers.

Checking Data: The CAN Node checks whether they require the data they have received for their functions or not.

Accepting Data: If the received data is important, it is accepted and processed. If not, the received data is ignored.

3.9.4 Bit Stuffing

The differential voltage scheme that CAN uses to reduce noise means that the signal never return to a zero voltage potential level. This is call a Non-Return to Zero (NRZ) signal. The practice of "Bit Stuffing" is implemented in CAN to ensure the proper synchronization of messages. Bit Stuffing is implemented in CAN by inserting a bit of opposite polarity after five consecutive bits of the same polarity. If the receiver indicates that six consecutive bits of the same polarity has been received then it knows a transmission error has occurred and will set the appropriate error flag(s). The image in Figure 3.19 shows the CAN frame before and after implementing the stuffbits that are indicated in purple.

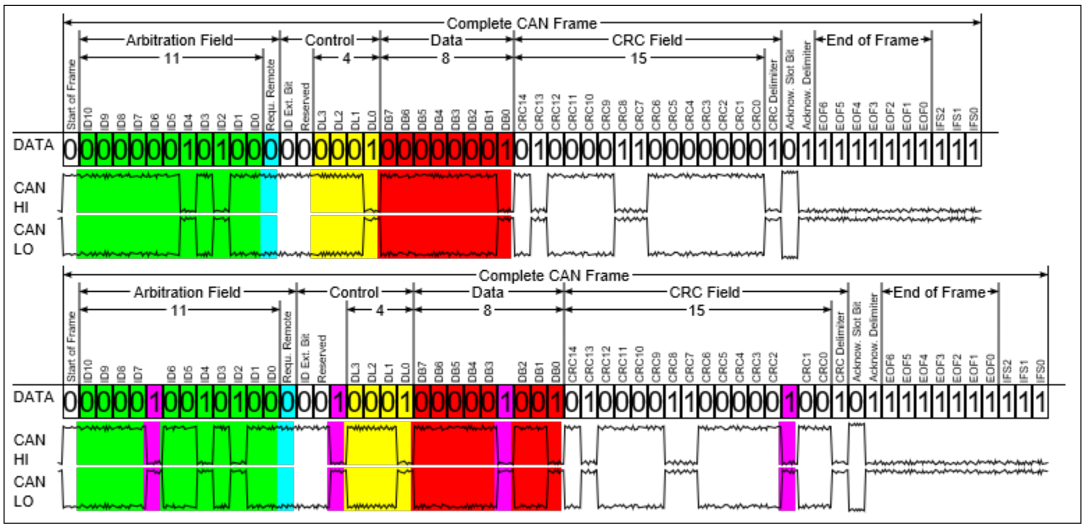


Figure 3.19: CAN Bus Base Frame Format (With Bit Stuffing) [80]

The standardization that came with the widespread adoption of the CAN bus protocol is critical to an ITS since it allows basic and advanced vehicle diagnostics and real-time monitoring to potentially be relayed back to a centralized ITS database.

3.10 On-Board Diagnostic OBDII

According to the May 1997 Environmental Face Sheet from the Environmental Protection Agency (EPA), The purpose of the On-Board Diagnostics or OBD system is to "assure proper emission control systems operation for the vehicle's lifetime by monitoring emission-related components and systems for deterioration and malfunction" [81].

The first version of the OBD system (OBDI) was introduced in the late 1980's to the early 1990's. It was used to encourage car manufacturers to produce vehicles with better emissions standards. This first version of the OBD system was very crude and only gave minimal information to the user and mechanic that was diagnosing the vehicles In late 1996 the second version of OBD was introduced (OBDII). It was federally required that all vehicles come equipped with an OBDII connector (SAE standard J1962). This connector, typically located under or around the driver compartment area is also called the Data Link Connector (DLC) The image in Figure 3.20 shows an image of the OBDII / DLC. This standardized port allows the lay person to read real-time data about the vehicle. It is important to note that vehicles manufactured before 1996 will not posses this port and thus will not be able to report this information directly to the end user.

Figure 3.20: OBDII Port (On All Vehicles Produced after 1996) [82]

Vehicle parameters such as engine RPM, throttle position, coolant temperature, and vehicle speed can be polled from the DLC of the OBDII system. These parameters can be accessed using a Parameter Identification (PID) that specifies what data from the system you would like to poll. OBDII has 10 modes of operation that can be used to obtain vehicle parameters. [83]. Several of these PIDs were used during this research.

Vehicle manufactures have a considerable amount of leeway when it comes to complying with the OBDII standard. The items in Table 3.3 illustrate the variability across the car manufacturing spectrum when it comes to the purpose of each pin on the DLC connector. In addition to the purpose of each pin, many vehicle manufactures use different protocols under the generic OBDII system. A total of five protocols have been established in the OBDII system. Below is a list and quick description of each of these protocols [84].

1. **SAE J1850 PWM:** This is a Pulse Width Modulation (PWM) based protocol, which runs at 41.6 kilobytes per second (kbps), that is generally used in Ford vehicles

2. **SAE J1850 VPW:** A Variable Pulse Width (VPW) based protocol, which runs at 10.4 kbps, that is typically used in GM vehicles

3. **ISO 9141-2:** An asynchronous serial communication used in Chrysler, European, or Asian vehicles that runs at 10.4 kbps.

4. **ISO 14230 KWP2000** A Keyword Protocol (KWP) 2000 and is used in Chrysler, European, or Asian vehicles that runs at 10.4 kbps.

5. **ISO 15765 CAN:** This is CAN bus protocol and is mandatory in all vehicles sold after 2008. This protocol can run up to 1 Mbps.

The image in Figure 3.21 and companion data in Table 3.3 show the pinout and manufacture dependent pins that comprise the OBDII connector. While there are many variances in the manufacturer specific pins it is important to note that many modern OBDII readers can decipher these discrepancies and still report real-time vehicle data.

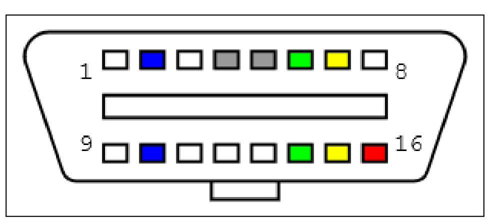


Figure 3.21: OBDII Female Port Pinout [85]

Table 3.3: SAE J1962 ODB II Connector Pinout [85]

1	**Manufacturer discretion:** • GM: J2411 GMLAN/SWC/Single-Wire CAN • VW/Audi/BMW: Switched +12V to tell a scan tool whether the ignition is on • Ford, FIAT: Infotainment CAN-High	9	**Manufacturer discretion:** • BMW: TD (Tachometer Display) signal aka engine RPM signal • GM: 8192 bit/s ALDL where fitted • Ford: Infotainment CAN-Low
2	Bus Positive Line of SAE J1850 PWM and VPW	10	Bus Negative Line of SAE J1850 PWM only (not SAE J1850 VPW)
3	**Manufacturer discretion:** • Ford: DCL(+) Argentina, Brazil (pre OBD-II) 1997-2000, USA, Europe, etc. • Ford: Medium Speed CAN-High • Chrysler: CCD Bus(+)	11	**Manufacturer discretion:** • Ford: DCL(-) Argentina, Brazil (pre OBD-II) 1997-2000, USA, Europe, etc. • Ford: Medium Speed CAN-Low • Chrysler: CCD Bus(-)
4	Chassis ground	12	**Manufacturer discretion:** • GM: Diagnostic codes to DIC (1994-2004 Corvette)
5	Signal ground	13	**Manufacturer discretion:** • Ford: FEPS - Programming PCM voltage
6	CAN-High (ISO 15765-4 and SAE J2284)	14	CAN-Low (ISO 15765-4 and SAE J2284)
7	K-Line of ISO 9141-2 and ISO 14230-4	15	L-Line of ISO 9141-2 and ISO 14230-4
8	**Manufacturer discretion:** • BMW: Second K-Line for non OBD-II (Body/Chassis/Infotainment) systems • FIAT: Infotainment CAN-Low	16	**Battery voltage:** • Type "A" 12V/4A • Type "B" 24V/2A

The parameters polled from the vehicle for this research included the engine RPM, vehicle speed, and throttle position. These parameters provided a method to determine the vehicles relative speed and future position. This data also provided a redundant feedback loop to compare other telemetry information obtained about the vehicle.

3.11 NMEA 0183

The National Marine Electronics Association (NMEA) 0183 standard defines the electrical signal requirements, data transmission protocol and time, and specific sentence formats for a 4800-baud serial data bus. [86]. Although the NMEA 0183 standard was widespread before GPS it is now the standard output protocol of the GPS devices and other marine equipment.

The output from a GPS, that uses the NMEA 0183 protocol, are called sentences. These sentences are a sequence of ASCII symbols beginning with a dollar sign ($) and ending with a carriage return/line feed sequence. Each sentence can be no longer than 80 characters and a null terminator. The generic frame format of a NMEA message is shown in the image in Figure 3.22.

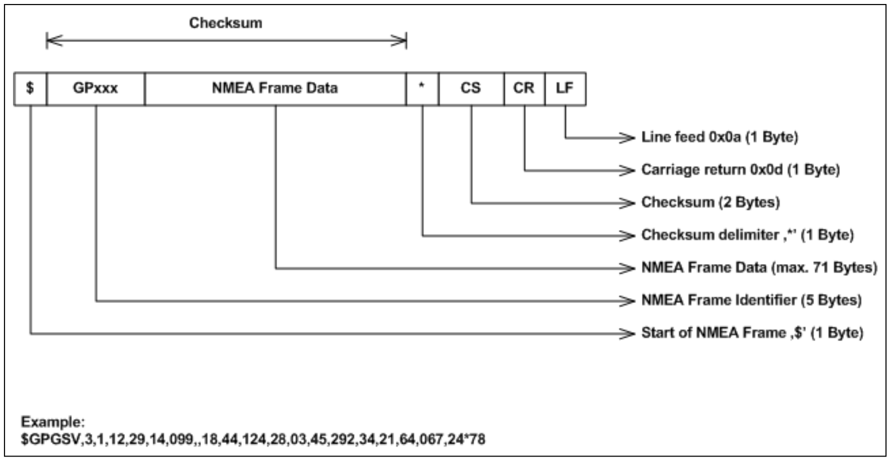


Figure 3.22: NMEA 0183 Protocol Frame Format [87]

There are many sentence types that can be parsed when receiving raw GPS data. Some of the more common sentence types such as GPGGA(Global Positioning System Fix Data), GPGLL (Geographic Position, Latitude / Longitude and time), GPGSA (GPS DOP and Active Satellites), GPGSV (GPS Satellites in View), and GPRMC (Recommended Minimum Specific GPS/Transit data) can be parsed from any GPS reciever receiving raw GPS information. The main sentences that were used in this research were the $GPGGA and the $GPRMC. The images in Figures 3.23 and 3.24 show a breakdown of how each sentence is structured and an example message being parsed into its respective category.

GPS information, provided by NMEA 0183 sentences, is another critical element to an intelligent transportation system. The relative position of each vehicle on the roadway enables the infrastructure to better calculate future vehicle maneuvers for all surrounding vehicles.

3.12 ZigBee / XBee Personal Area Network PAN

In 2003, the ZigBee Alliance introduced the ZigBee standard protocol. ZigBee builds upon the IEEE 802.15.4 standard. XBee's radio modules use the ZigBee protocol stack and are compliment with the 802.15.4 IEEE standard. They are currently produced by Digi International. The image in Figure 3.25 show an example of a

$GPGGA

Global Positioning System Fix Data

eg1. $GPGGA,170834,4124.8963,N,08151.6838,W,1,05,1.5,280.2,M,-34.0,M,,,*59

Name	Example Data	Description
Sentence Identifier	$GPGGA	Global Positioning System Fix Data
Time	170834	17:08:34 UTC
Latitude	4124.8963, N	41d 24.8963' N or 41d 24' 54" N
Longitude	08151.6838, W	81d 51.6838' W or 81d 51' 41" W
Fix Quality: - 0 = Invalid - 1 = GPS fix - 2 = DGPS fix	1	Data is from a GPS fix
Number of Satellites	05	5 Satellites are in view
Horizontal Dilution of Precision (HDOP)	1.5	Relative accuracy of horizontal position
Altitude	280.2, M	280.2 meters above mean sea level
Height of geoid above WGS84 ellipsoid	-34.0, M	-34.0 meters
Time since last DGPS update	blank	No last update
DGPS reference station id	blank	No station id
Checksum	*75	Used by program to check for transmission errors

Figure 3.23: NMEA Frame Format $GPGGA Example Message [88]

```
$GPRMC,123519,A,4807.038,N,01131.000,E,022.4,084.4,230394,003.1,W*6A

Where:
     RMC          Recommended Minimum sentence C
     123519       Fix taken at 12:35:19 UTC
     A            Status A=active or V=Void.
     4807.038,N   Latitude 48 deg 07.038' N
     01131.000,E  Longitude 11 deg 31.000' E
     022.4        Speed over the ground in knots
     084.4        Track angle in degrees True
     230394       Date - 23rd of March 1994
     003.1,W      Magnetic Variation
     *6A          The checksum data, always begins with *
```

Figure 3.24: NMEA Frame Format $GPRMC Example Message [88]

series 1 XBee radio module. Each radio module has a unuqie address that is used to distinguish each unit on the same Personal Area Network (PAN).

The following list describes the three types of nodes that comprise a ZigBee based PAN. They include a central coordinator, routers, and end devices. A node can operate as either a full-function device (FFD) or reduced-function device (RFD). An FFD can perform all the tasks that are defined by the ZigBee standard, and it operates in the full set of the IEEE 802.15.4 MAC layer. An RFD performs only a limited number of tasks [89].

1. **Coordinator:** A coordinator is an FFD and responsible for overall network management. Each network has exactly one coordinator. The coordinator performs the following functions: Selects the channel to be used by the network

    - Starts the network
    - Assigns how addresses are allocated to nodes or routers
    - Permits other devices to join or leave the network
    - Holds a list of neighbors and routers
    - Transfers application packets

2. **Router** A router is an FFD. A router is used in tree and mesh topologies to expand network coverage. The function of a router is to find the best route to the destination over which to transfer a message. A router performs all functions similar to a coordinator except the establishing of a network.

3. **End Device** An end device can be an RFD. An RFD operates within a limited set of the IEEE 802.15.4 MAC layer, enabling it to consume less power. The end device (child) can be connected to a router or coordinator (parent). It also operates at low dutycycle power, meaning it consumes power only while transmitting information. Therefore, ZigBee architecture is designed so that an end device transmission time is short. The end device performs the following functions:

    - Joins or leaves a network
    - Transfers application packets

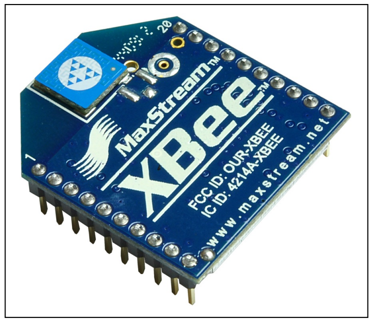


Figure 3.25: XBee S1 Radio Module [90]

The image in Figure 3.26 shows an example of a complete PAN with the one required central coordinator. Take note that there can be multiple routers and end devices for a single centralized coordinator.

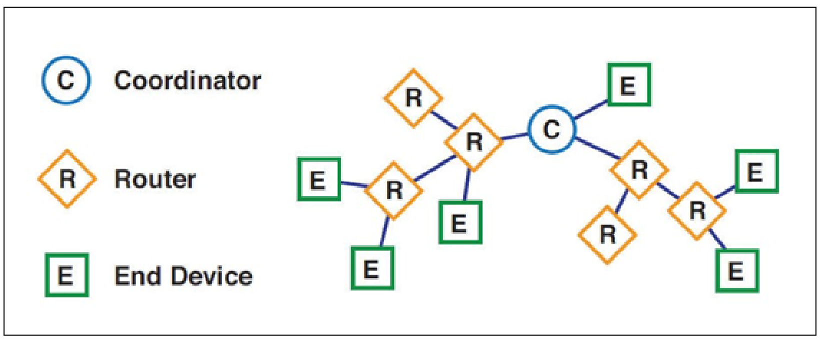


Figure 3.26: ZigBee Personal Area Network [91]

There are two types of XBee modules that are frequently used, series 1 (S1) and series 2 (S2). S1 modules can be configured as a coordinator or as an end device only, while the S2 modules can be configured as a coordinator, router, or end device.

There are three modes of operation for an XBee radio module. Each mode and a breif explination of each mode follows:

1. **Application Transparent (AT) Mode:** In this mode, all the serial data that appears on the radio module input pin is transmitted over the RF medium. The user does not worry about framing the data in a packet structure format. This mode acts similar to Universal Asynchronous Receiver/Transmitter (UART) communication for all XBees in same network.

2. **Application Programming Interface (API) Mode:** In this mode, the user must form a proper packet format and encapsulate the desired data in it. Then the complete frame can be transmitted by the XBee radio module to others on the same network.

3. **API Escape Operation Mode:** This mode is similar API mode, however, it escapes certain data to avoid conflict with special characters. For example a start delimiter of (0x7E). This translates to a higher level of reliability of the RF transmitted data packet.

CHAPTER 4: COMPONENTS OF AN INTELLIGENT TRANSPORTATION SYSTEM FRAMEWORK

There are numerous components that comprise an Intelligent Transportation System (ITS). These systems can be divided into two logical partitions; vehicle and infrastructure. Both partitions are, in themselves, complete areas of heightened research, however, to develop a comprehensive ITS one must consider the interaction and challenges that are present on both partitions to successfully deploy a nationwide ITS framework. The block diagram in Figure 4.1 showcases, at a high level, the minimum framework needed for an ITS. The following sections break down these sub-components in detail and describe the unique challenges that each piece presents.

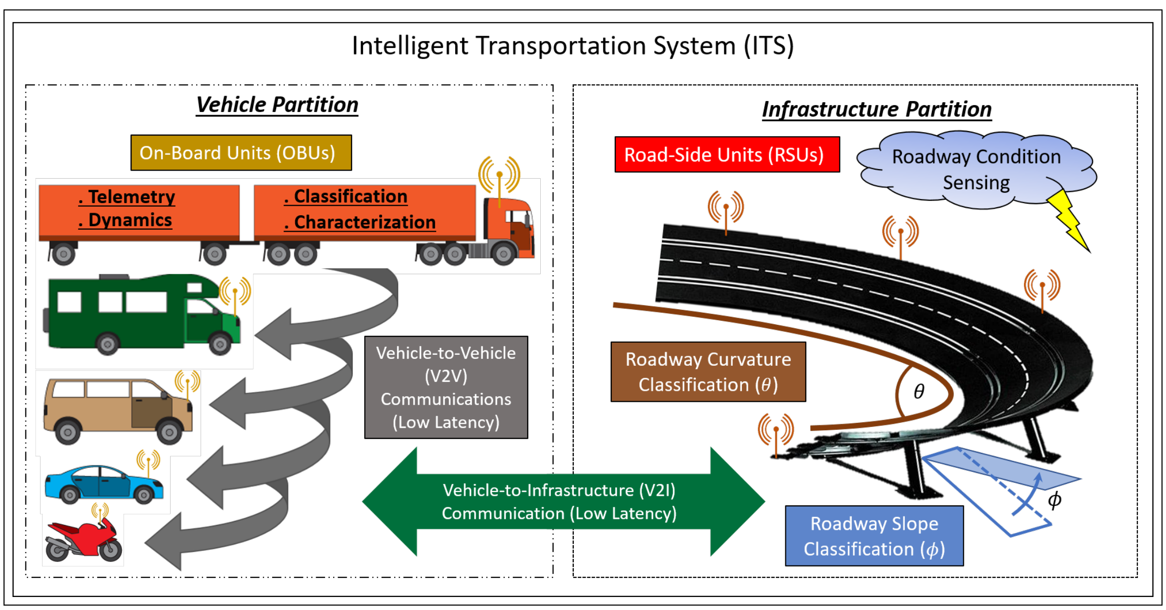


Figure 4.1: Components of an Intelligent Transportation System (ITS) [69, 92]

4.1 Vehicle Partition

Vehicle data collection and dispersion is the first crucial partition of an ITS. Unlike the surrounding infrastructure, which is static, vehicles on the highway are in a constant state of motion. Information about each vehicle, such as relative speed, direction, location, and a host of other parameters must be communicated to all other vehicles on the roadway, and the infrastructure, on a sub-millisecond scale. The following sections highlight the purpose of each vehicle partition section and relays how each section is critical to the overall ITS framework.

4.1.1 Vehicle Classification & Characterization

The first challenge that the vehicle partition presents is developing a method to broadly classify and characterize all vehicles on the highway. The rules and regulations provided by both the NHTSA and the other organizations as to what defines a class of a vehicle and the particular specifications that divide and subdivide these classes are not consistent. A system needs to be developed that can quickly identify and classify each vehicle in real-time. A centralized database of all the major manufacturer's vehicles would need to be assembled to enable the ITS framework to differentiate vehicle dynamics of each automobile on the highway. This type of database and characterization method would allow control algorithms to be implemented, since the dynamics of each vehicle posses a challenge in automating their movements around the various roadways.

4.1.2 Vehicle Telemetry & Dynamics

The concept of gathering vehicle data and transmitting it is not a new one. The patent information shown in [93] illustrates how a small and simple device can be used to transmit vehicle telemetry data. The connection used for this data is the On-Board Diagnostic (OBD) port located in all vehicles produced after 1996. Some of the protocols in Chapter 3 highlight the plethora of vehicle diagnostic data that is available in real-time to both the driver and vehicle technician. Some of the more noteworthy include the vehicles speed, throttle position, fuel level, and various sensor statuses. To date, most of this information that the vehicle's ECUs collect and display are isolated solely to that vehicle. In order for a broad ITS to take hold, each vehicle's diagnostic and real-time sensor data needs to be communicated to both the surrounding vehicles, as well as, a centralized data collection system. This system would be able to take the broad vehicle data and send real-time driving decisions to each vehicle on the roadway.

The challenge of classifying and characterizing each vehicle on the roadway posses enough of a challenge, however, this information must be collected to develop accurate dynamic models of each vehicle. In order to incorporate legacy vehicles into an intelligent transportation system a broad dynamic model for each vehicle class and type will need to be developed. Once such a model exists, digital control signals can be sent to the various vehicles on a given stretch of roadway to enable them to autonomously and safely traverse that stretch of highway. While this research does not set out to develop such models they are a necessary part of a complete ITS.

4.1.3 Vehicle-to-Vehicle (V2V) Communication

To this point, it may seem that a centralized system that contains all the crucial vehicle information and can dynamically send real-time and receive data to and from each automobile would be sufficient for an ITS; however, vehicle-to-vehicle (V2V) communication holds the same weight when dealing with an ITS framework. A centralized database does provide pertinent information to each vehicle, however, due to communication delay it does little to address the nearly real-time response required to orchestrate a multiple vehicle type highway. In this realistic scenario each vehicle has its own dynamics and levels of sensory data. To enable all vehicles, including legacy ones, to occupy the same roadway many levels of V2V communication needs to be available. For example, each vehicle would need to know the approximate speed and braking distance of the vehicle in front and behind it. As mentioned in Chapter 2 and 3 there exist many protocols and standards that are aimed to address this issue, however, the widespread distribution of these systems have been slow due to a wide gamut of reasons. This portion of an ITS framework is crucial for moving forward as real-time data needs to be shared between all vehicles which contain varying levels of data resolution. The author's research does not deal directly with this type of communication, however, it is still part of an overall ITS framework that needs to be fully implemented to realize the goals of a complete ITS framework.

4.1.4 On-Board Units (OBUs)

The wide array of sensor data and vehicle telemetry is often tucked away into the ECUs of each vehicle. This data is often used for diagnostic purposes by technicians, however, since most vehicle data is kept local, it does not inform surrounding vehicles with any information about its own characteristics or sensory data. There needs to be a paradigm shift in how vehicle data is stored and distributed. To date, there is not an easy method to relay such information. Many companies are tying to develop radios and devices that adhere to the 802.11p and other vehicle specific standards; however, this often defaces the automobile or requires placing bulky equipment inside each vehicle. It is clear that this method is not practical as manufacturers are hesitant to place such devices in their vehicles and consumers seem to have little interest in retrofitting large pieces of equipment inside their limited cargo spaces. A widget of some kind needs to be developed that can easily be placed in the vehicle, with little obstruction, and be able to adapt to the various degrees of complexity that each vehicle contains. This widget would also need to have the ability to communicate with the surrounding vehicles, via V2V, and transmit data to the surrounding infrastructure.

4.2 Infrastructure Partition

In addition to inter-vehicle and V2V communications, an ITS framework must also be able to relay information about the surrounding infrastructure and roadway conditions. Many services, such as GPS, exists to allow the average driver to successfully drive from one point to another. These services, however, do little to alert the driver, in real-time, of the ever changing roadway landscape in front or behind them. In order for an ITS to have a complete picture, the roadway and other surrounding infrastructure needs to be characterized and reported back to a centralize database system. The following sections highlight some of the infrastructure parameters needed to be collected in order to develop a comprehensive view of the roadways that vehicles must travel.

4.2.1 Roadway Characterization

One of the most fundamental parts of characterizing the infrastructure is to develop a method that can quickly and accurately provide the end driver with complete picture of the approaching roadway. Many services, such as Google Maps ™ and the Waze application, provide GPS navigation and real-time traffic updates. While these services do provide a wide array of roadway conditions, no one method has been developed to provide a comprehensive snapshot of roadway data. The infrastructure by definition is static; however, each roadway has its own age and environmental effects that make developing a comprehensive roadway characterization difficult. In order to address this issue a method needs to be developed that can quickly characterize the many factors of a given stretch of highway. The following two sections highlight the main components required to build a complete view of roadway conditions.

4.2.1.1 Roadway Curvature & Slope Classification

The slope of the road and the curve or banking of the highway are two parameters that need to be characterized in order to provide the ITS framework with enough data to allow it to send the relative movement commands to each vehicle. Based on this research there seems to be no existing method that comprehensively characterizes both the curve and slope of a given stretch of highway. The need exists for a crude method developed that can relay this relative static roadway information to the ITS framework. A non-real time example of such data collection can be found in the fast-paced sport of rally car racing. There method consists of two phases. Phase one is data collection. During this phase the co-driver and driver will either walk or slowly drive on the given track. The co-driver will then make detailed personal notes on the roadway conditions. These notes tell the slope and curve angle (often denoted by on scale of severity from 1 to 6). The second phase involved the driver and co-driver traversing the track at a high rate of speed where the co-driver relays the layout of the approaching turns and winds in the roadway. A similar technique could be developed for the more tame conditions found on most highways and rural roads. This form of having foreknowledge of the approaching roadway is a vital key in realizing a comprehensive ITS framework.

4.2.1.2 Roadway Condition Sensing

Roadway condition sensing combined with characterization and classification is the only way to provide a complete picture of the past, present, and future roadway conditions that a vehicle may face as it traverses each stretch of roadway. Each vehicle has its own dynamics that are drastically altered based on the conditions of the roadway. As an example, a large vehicle would need to increase its stopping distance if the roadway was wet versus dry. A comprehensive ITS would need to have this additional data in order to adapt the control messages sent to the various types of vehicles approaching that particular section of highway. Even with the most advanced GPS navigations systems, there still does not exist a universal system that informs the driver of the approaching roadway's current condition. This information could be easily obtained via a wireless sensor network (WSN) that could collect data such as road surface temperature, rainfall accumulation, etc. This form of data collection is necessary to dynamically change the roadway picture sent to the vehicle. This would also enable dangerous highway conditions to be reported to surrounding vehicles so an alternate route could be determined.

4.2.2 Vehicle-to-Infrastructure (V2I) Communication

The process of gathering data of the roadway conditions and characteristics is an important process, however, this information needs to be transmitted to each passing vehicles on that given stretch of highway. This is where Vehicle-to-Infrastructure (V2I) communication becomes vital as a key aspect of the overall ITS framework. The main objective behind V2I communication is for the vehicle to be able to transmit real-time traffic and roadway condition data back to a centralized system that can, in turn, propagate this information to other vehicles in the surrounding area. This type of communication enables the infrastructure to take some of the processing load off the vehicle. This concept is in stark contrast to the current paradigm as most semi and fully-autonomous vehicle manufactures tend to load the vehicle with the burden of calculating all the real-time risks and obstacles that are in its immediate path of intended travel. With V2I communication this takes most of the computational burden off the vehicle as it would only be responsible for direct obstacle avoidance vs avoidance and navigation. V2I communications also allows the vehicle to have a complete picture of the surrounding vehicles and the approaching roadway conditions. This transfer of computing power will allow the vehicle to reduce weight and cost as the number of sensors and avoidance systems could dramatically be reduced. This would be possible if a wide scale ITS framework, that fully incorporated V2I communication, would be implemented across all urban and rural highways.

4.2.3 Road-Side Units (RSUs)

In order for V2I communications to occur, data needs to be gathered about the roadway conditions and characteristics and then transmitted back to a centralized system. Road-side units (RSUs) are used to accomplish this data gathering task. The basic internal components of an RSU include: a GPS unit, some type of micro-controller unit (MCU) or central processing unit (CPU), and a radio unit to allow for transmission (Tx) and reception (Rx) of the data to the centralized ITS. It is important to note that companies are currently developing RSUs that comply with the communications standards setup and discussed in Chapter 3, however, the wide spread adoption of these devices has been slow at best. There exists a need for an RSU that can be quickly deployed at a much lower cost that current units. If such a device can be developed then the mapping and condition gathering data for all roadways could commence. The RSUs would also need to be easily deployed and consist of static and movable units. Services such as Google Maps ™ do provide much of the navigation data needed for basic vehicle navigation, however, only with a wide spread deployment of RSU sensors could a comprehensive roadway characterization be accomplished.

4.2.4 ITS Framework Summary & Author's Contributing Work

The image in Figure 4.2 shows the flow of data needed for a complete ITS. This research has focused on the items denoted in the filled blocks. The unfilled blocks represent necessary components of an ITS framework, however, this research did not focus on pursuing these active areas of research. The author's contribution to this system will aid in the adoption and distribution of a nationwide intelligent transportation system that has the capacity to include all vehicles on the roadway, including legacy ones. A brief summary of each filled block that represents the author's originals work follows:

- On-Board Units (OBUs): Represents a small device(s) that is used to collect vehicle data and transmit that information to the RSU
  - Vehicle Telemetry: Represents vehicle specific data, such as RPM, throttle position, etc, that is collected via the OBUs
  - Vehicle Characterization /Classification: Represents a novel technique, developed by the author, that divides all vehicles on the roadway into a six class system based on vehicle parameters such as length, height, etc.
- Road Side Units (RSUs): Represents a device(s) that is used to collect roadway condition data and transmit that information to the OBU
  - Roadway Curvature Calculation: Represents a novel technique, developed by the author, that measures and calculates the curvature for a given stretch of roadway based on RSU GPS data. The results were divided into a six level classification system.
  - Roadway Slope Classification: Represents a novel technique, developed by the author, that measures and calculates the slope for a given stretch of roadway based on RSU GPS data. The results were divided into a five level classification system.
- Vehicle-to-Infrastructure (V2I) Communication: Represents the transfer of data between the RSUs and the OBU. A novel XBee packet scheme was developed by the author to accomplish this task.

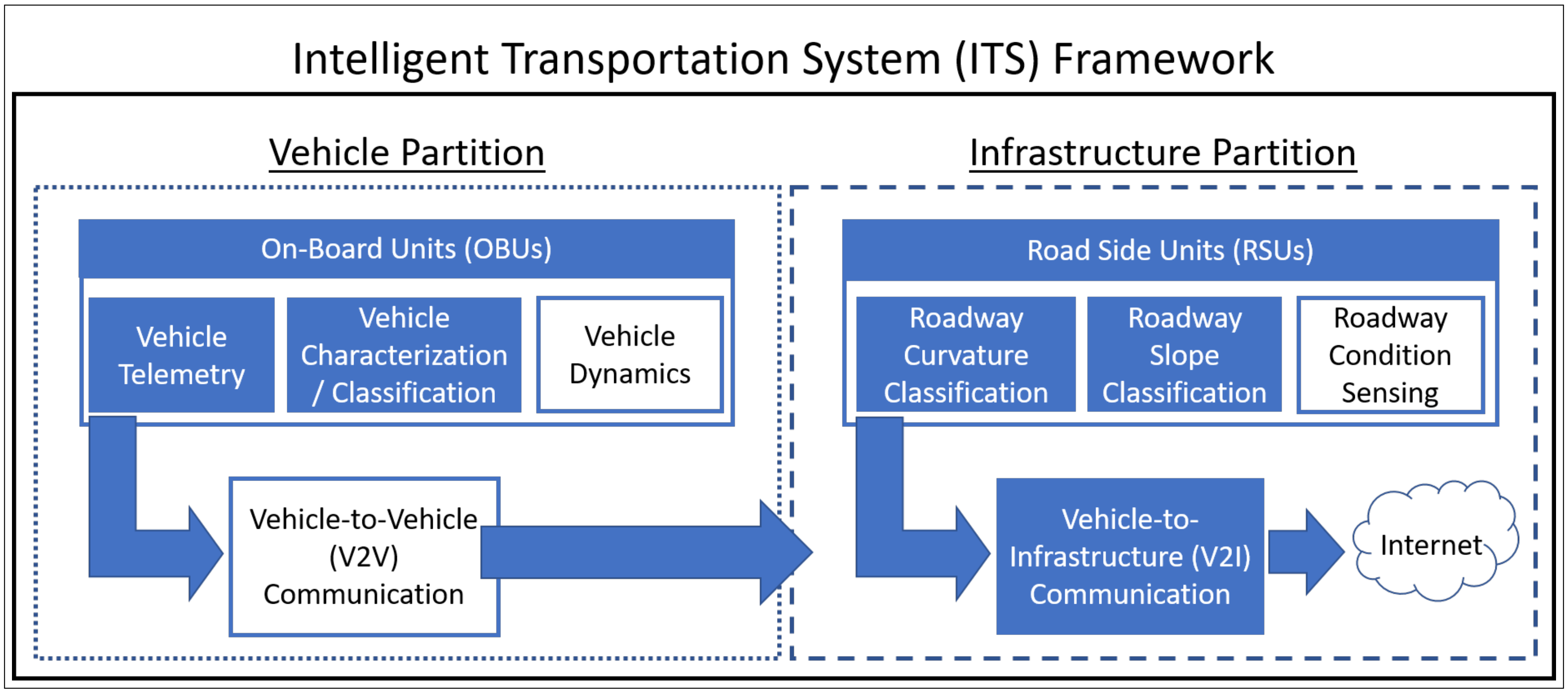


Figure 4.2: ITS Framework Flowchart

CHAPTER 5: FRAMEWORK IMPLEMENTATION & VALIDATION

The previous chapter has broken down each section and subsection that comprise an ITS. This chapter will mirror the last and describe how each of the aforementioned sections were implemented in the author's proposed framework. It is important to note that many of the hardware and software decisions that were made throughout the course of this implementation were based solely on cost efficiency and the author's level of confidence with the various software platforms and languages.

5.1 Vehicle Partition

There are many methods to gather data from most modern vehicles. Cars and trucks produced after the year 1996 are required to have an OBDII port that allows even novice car owners the ability to read real-time data from their vehicle. Most of the current ITS solutions take advantage of this port to transmit and receive data from the vehicle, however, no current system exists that encompasses all vehicles on the road today. These systems typically are not suited for vehicles produced before 1996. This research will refer to these older cars and trucks as legacy vehicles. The data collection process for the vehicle partition consists of two major parts: vehicle classification / characterization, and vehicle telemetry / dynamics data collection. The following sections show how this data was collected and processed via novel methods that were developed by the author.

5.1.1 Vehicle Classification & Characterization

The diversity of vehicles on the roadway highlights a major hurdle that intelligent transportation systems have to deal with and address. Since the goal of an ITS is to be applicable to all vehicles on the road, there needs to be a method in place to characterize and classify every possible type of vehicle. The author has developed a rudimentary method to accomplish this task. As mentioned in Section 3.7, there are many standards that are used to accomplish the tasks of classification / characterizing vehicles that typically use parameters such as weight, number of axles, etc; however, there is not universal method or means by way to classify or characterize all vehicles on the highway.

The author arbitrarily chose to break down all vehicles into six classes. The course division of classes was due, in part, to the broad parameters used to place vehicles in the six class system. Both hard and soft parameters were used to place each vehicle into their receptive class. After observing many types of vehicles and their various measurements, it was determined that the vehicle length was the single most determining factor that could be used to break down a vehicle into a certain class. Based on this observation, the length of the vehicle was used as a hard parameter in the sorting process into the six class system. Two soft parameters that were used to provide better resolution in the sorting process was the curb weight and the height of the vehicle. Based on the hard and soft parameters it was determined that all vehicles on the roadway could be divided into this six class system. The first five classes would contain all the regular vehicles on the roadway, while class six was specifically reserved for emergency vehicles. Class six would also be divided into three sub-classes to provide a better method of distinguishing what type of emergency vehicle was being used. The consideration and thus separate classification of emergency vehicles was necessary since an ITS needs to account for emergency vehicles "overriding" the typical flow of traffic for emergency situations. The data in Table 5.1 shows the breakdown of how each vehicle class in the six class system was structured. This classification and characterization technique establishes the necessary foundation upon which telemetry and dynamics data can be meshed together to provide a complete picture of every vehicle on a given stretch of roadway.

Table 5.1: Vehicle Characterization and Classification

Class	Length (in)	Weight (lbs)	Height (in)	Example
1	x < 100	x < 1000	x < 50	**Motorcycle:** BMW 1200R
2	100 < x < 185	1000 < x < 3500	50 < x < 60	**Coupe:** Nissan GT-R
3	185 < x < 200	2500 < x < 4500	55 < x < 65	**Sedan:** Honda Accord
4	200 < x < 215	4500 < x < 7000	60 < x < 75	**Truck:** RAM 2500
5	x > 215	x > 7000	x > 75	**Bus:** Volvo Tour Bus
6A	x < 100	x < 1000	x < 50	**Emergency Vehicle:** Police Motor Bike
6B	100 < x < 215	1000 < x < 7000	50 < x < 75	**Emergency Vehicle:** Ambulance
6C	x > 215	x >7000	x > 75	**Emergency Vehicle:** Fire Truck

5.1.2 Vehicle Telemetry & Dynamics

The process of gathering and transmitting vehicle data is one of the crucial components of an ITS. It is important to again note that multiple protocols and standards that well define how this process should take place already exists and are discussed at length in Chapter 3; however, due to the limited distribution of these standards the author has developed an alternative method that could bridge the gap between the current state of these standards and the rapidly changing vehicle landscape present on modern highways.

The image in Figure 5.1 shows a high level block diagram of all the various components that were developed by the author to accomplish the task of collecting and transmitting vehicle data. It is important to note in Figure 5.1 the two different systems that are illustrated. There is a system for vehicles produced after the 1996 production year (brown dotted line) and a system that utilizes a smartphone based reporting system for legacy vehicles (black dashed line).

The components that comprise both systems will be discussed, in detail, along with the algorithms and processes developed by the author to regulate data collection and transmission.

As stated earlier, there are a wide spectrum of vehicles that make up most modern

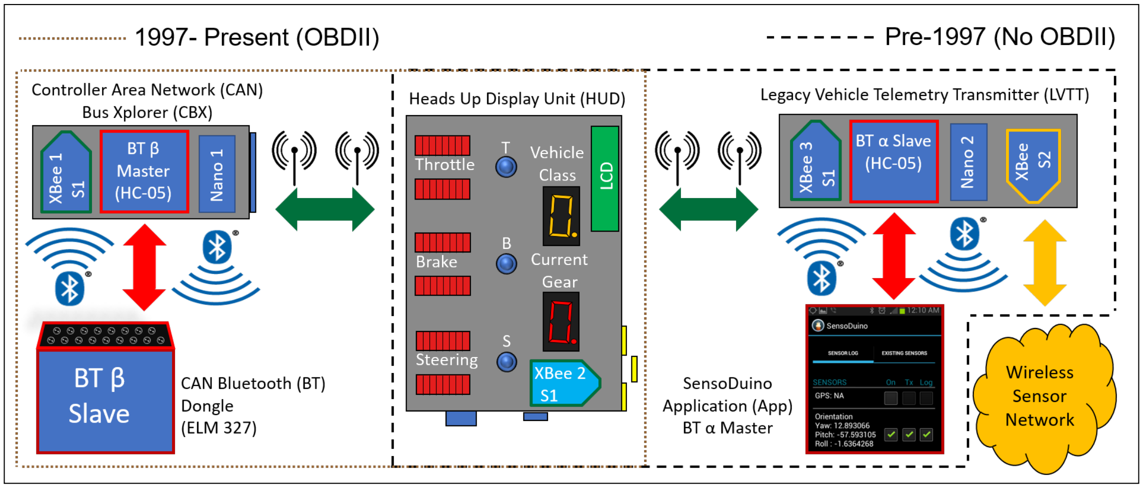


Figure 5.1: High Level Block Diagram

highways. Automobiles are becoming more efficient and reliable and thus the length of ownership for a given vehicle is much longer than in decades past. This presents a major hurdle for an ITS as such a system needs to receive basic data points from all vehicles on the roadway regardless of age or degrees of technology that it natively contains. The author has developed a novel method that could allow all vehicles, both new and legacy, to transmit basic vehicle data and information.

The author's method of vehicle data collection consists of three main devices. The three devices used are based upon the vehicle's model year. Automobiles produced before the production year of 1996, and those produced after. The following breaks down which devices are used for these two different scenarios.

Post 1996 Vehicles (Modern) Modern vehicles contain the OBDII connector that allows the user and automotive technicians access to the CAN bus to interpret and read those signals. Those signals include data about multiple vehicle parameters such as the throttle position, current speed, vehicle VIN number, etc. The access to these parameters are made possible by inexpensive devices that can interpret the OBDII protocol and display the various vehicle metrics to the user.

- OBDII Bluetooth Dongle (ELM327)
- CAN Bus Xplorer (CBX)
- Heads Up Display Unit (HUD)

Pre 1996 Vehicles (Legacy) Legacy vehicles consist of any vehicle produced prior to 1996. While many of these vehicles include electronics that report vehicle data, this data is not easily accessible to the user or they simply do not exist in the case of very old vehicles.

- Smartphone Application (SensoDuino)
- Legacy Vehicle Telemetry Transmitter (LVTT)
- Heads Up Display Unit (HUD)

Due to the wide array of vehicles on the road, this two part system was developed to encompass all of them and capture as much information about the vehicle as possible. Note that the Head Up Display Unit (HUD) was present in both systems. This centralized unit was used to visualize the incoming vehicle data along with the data from the surrounding infrastructure data, discussed later in this chapter. It is important to note that gathering vehicle data from modern vehicles is not a new system; however, the author's method of integrating legacy and modern vehicles into a complete system provides a unique solution to this data collection problem. The following two sections describe the setup and data collection process that was developed by the author to tackle the data collection process for the vehicle partition of an ITS.

5.1.2.1 Modern Vehicle Data Collection Process

Vehicles produced after the year 1996 are mandated to have an OBDII connector located inside the cabin of the vehicle. This diagnostic port gives the driver and technician access to vehicle data that had previously been inaccessible to the average user. The ODBII port connection can access a wide array of vehicle data. Most of the data provided by this port are used for diagnostic and emission standard purposes, however, it also provides relative real-time information about the various vehicle ECUs and other sensory data. It is important to note that although the OBDII standard was implemented in 1996 each car manufacturer was given the flexibility to define their own sub protocol and message handling schemes. Many OBDII scanners need to be able to interpret this wide array of different standards in order to decipher the ECU data being transmitted through the OBDII port. The data that was of interest for this research was the following:

- Vehicle Identification Number (VIN) *When available
- Throttle Position Data *When available
- Vehicle Speed *When available
- Vehicle RPM *When available

Referring back to the image in Figure 5.1, there are multiple levels of communications represented by the high level block diagram. For quick validation purposes, and based on the current knowledge of the author, Bluetooth and XBee radios were used for the wireless transmission methods. It has been previously stated that protocols and standards are in place that accomplish this same task, however, due to the price and lack of knowledge of these devices by the author those two radio mediums and standards were used in the framework validation process. The author also simulated the use of a vehicle via a gaming steering wheel that provided feedback voltages into the system to simulate the throttle, brake, and steering angles of a full size vehicle. The following paragraphs explain the pairing and data dispersion processes between the three devices (Bluetooth Dongle (ELM327), CBX, and HUD). The novel data packet structure, developed by the author, will also be discussed.

The first step in the data gathering process for modern vehicles involves connecting a Bluetooth dongle (ELM 327) to the OBDII connector of the vehicle. Again this is only applicable to vehicles produced after the year 1996. The image in Figure 5.2 shows a stock image of the OBDII Bluetooth dongle that was used in this research. As denoted in the high level block diagram, the ELM 327 Bluetooth dongle was the Bluetooth beta (β) slave device. The Bluetooth β master device (for both the CBX and LVTT) was the HC-05 Bluetooth radio module. The pairing process is detailed in Algorithm 1 and an example code listing is shown in Figure 5.3.

Figure 5.2: ELM327 Bluetooth Dongle for OBDII Port [94]

Algorithm 1 Bluetooth Pair Procedure for HC-05 and ELM327

Steps

1: Enter AT mode by sending a logic `HIGH` signal to the Key pin on the HC-05 Module

2: HC-05 configured as Master Device (The ELM327 OBDII Module is a slave device, therefore, the HC-05 must be configured as the Master device)

3: Obtain MAC address from HC-05 via the AT command `AT+INQ`

4: Pair HC-05 and ELM327 via the AT commands `AT+PAIR` and `AT+BIND`

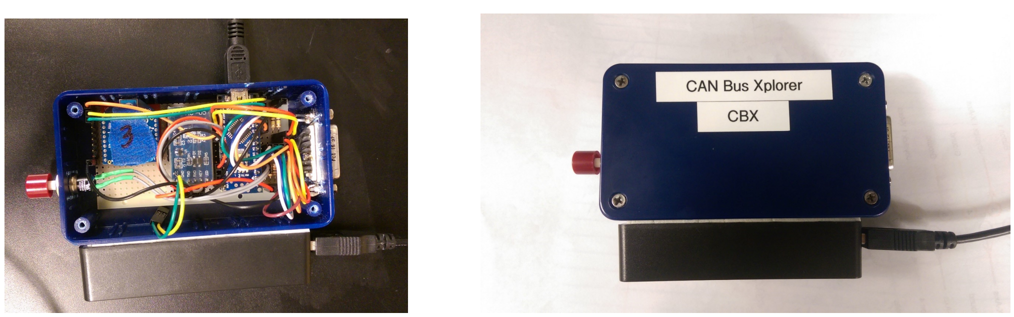
5: Exit AT mode by sending a logic `LOW` signal to the Key pin on the HC-05 Module

The CBX was designed and built to be the liaison between the OBDII port messages and the Heads Up Display Unit (HUD). The image in Figure 5.4 shows the internal wiring and the completed device. The high level block diagram in Figure 5.1 also shows that a Series 1 XBee was used in the CBX and the HUD. It is also important to note that there are both Series 1 (S1) and Series 2 (S2) XBee radio modules were

```
enterATMode();                              //enter HC-05 AT mode
delay(500);
sendATCommand("RESET");                     //send to HC-05 RESET
delay(1000);
sendATCommand("RMAAD");                     //remove previous pairs
sendATCommand("ORGL");                      //send ORGL, reset to original properties
sendATCommand("ROLE=1");                    //send ROLE=1, set role to master
sendATCommand("CMODE=0");                   //send CMODE=0, set connection mode to specific address
sendATCommand("BIND=000D,18,3A6789");       //send BIND=??, bind HC-05 to OBD Bluetooth address
sendATCommand("INIT");                      //send INIT, cant connect without this cmd
delay(1000);
sendATCommand("PAIR=000D,18,3A6789,20"); //send PAIR, pair with OBD address
delay(1000);
sendATCommand("LINK=000D,18,3A6789");       //send LINK, link with OBD address
delay(1000);
enterComMode();                             //enter HC-05 comunication mode
delay(500);
```

Figure 5.3: Sample Arduino Code to Initialize the HC-05 BT Module

used to relay data to the HUD and the corresponding infrastructure via a wireless sensor network (WSN). For a more detailed description for the distinction between S1 and S2 devices please refer to Section 3.12.



(a) CAN Bus Xplorer (CBX) Internal Wiring

(b) CAN Bus Xplorer (CBX) Completed Assembly

Figure 5.4: CAN Bus Xplorer (CBX) Internal Wiring and Completed Assembly

The process for transmitting data from the ELM 327 to the CBX is detailed in Algorithm 2. Note that the level of details about the vehicle will vary.

Algorithm 2 CBX XBee Execution Flow (OBD Mode)

Steps

1: Set up Bluetooth connection with ELM-327 module

2: Initialize the ELM-327 OBD-II Module (NOTE - The ELM327 Module only works with vehicles equipped with an OBDII port, 1996 and newer)

3: Send request command to obtain Vehicle Identification Number (VIN)

4: Construct an XBee data packet, place VIN inside payload portion then transmit data packet

5: Send additional request commands to obtain other vehicle parameters, such as Engine Revolutions per Minute (RPM), Throttle Position, and Vehicle Speed (NOTE - amount of parameters available depends on year, make, and model of vehicle)

6: Construct an XBee data packet, put collected data inside payload portion then transmit data packet

7: Check for any receive mode XBee packets

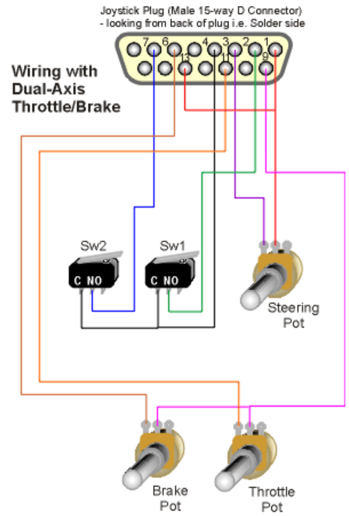
8: If any receive packet consist of mode change command, then change current mode to Joystick mode and send acknowledge to Heads Up Display (HUD) Module

9: If no Tx or Rx errors occur, then loop back to step 5

The CBX also contains a 15 pin game port that was used to connect the gaming joystick steering wheel. The steering wheel and pedals were used as a stand-in for a full size vehicle. The steering wheel and pedals were constructed out of simple switches and potentiometers that varied the resistance proportional to the movements of the steering wheel and pedals. This simulated the actual measurements of the steering angle and throttle / brake positions that could be reported by the OBDII port of an actual vehicle. The image in Figure 5.5 shows the joystick steering wheel and companion pinout that was used for this research. This steering wheel was also used to set the current gear and vehicle class that was visualized on the heads up display unit (HUD).

The HUD was developed as a way to visualize all the data for both the modern and legacy vehicle systems. The HUD contains a single S1 XBee radio that can accept messages from both the modern and legacy devices developed for this research. The images in Figure 5.6 shows the completed HUD and internal wiring. The HUD also

(a) Saitek R100 Joystick Steering Wheel with Annotations

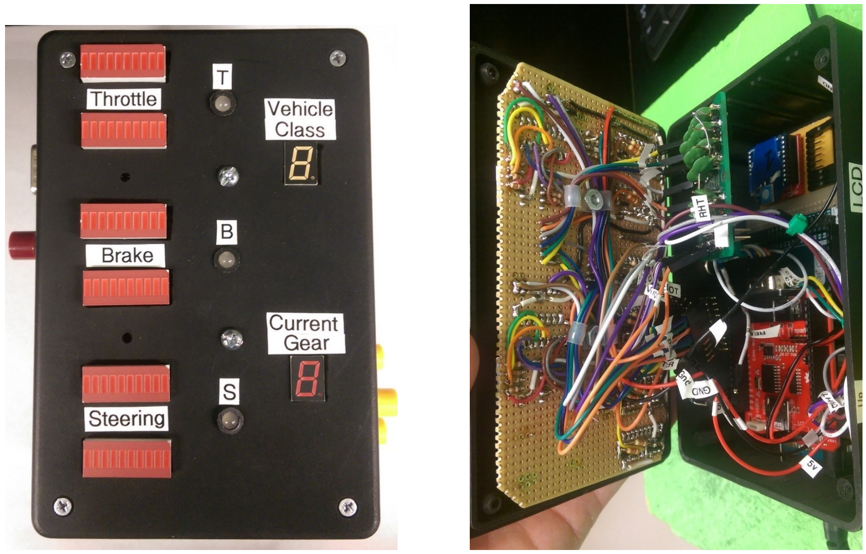


(b) 15 Pin GamePort Pinout for Steering Wheel with Pedals

Figure 5.5: Joystick Steering Wheel and Companion Pinout [95]

received data from the gaming joystick steering wheel which was used to simulate an actual vehicle. The HUD allowed the driver to quickly visualize all the key parameters provided by the OBDII port readings. Those parameters included the throttle, brake, and steering positions of the vehicle. This visualization was accomplished in two ways, one was from the red LED bar lights located in the front of the HUD, and the other was the LCD screen located on the side of the device. The internal workings of the HUD will be discussed in detail in Section 5.1.2.2 in regards to the legacy vehicle data transmission and collection process.

The XBee Series 1 radios in both the CBX and LVTT (discussed in detail in Section 5.1.2.2) wirelessly transmit their data packets to the HUD. The HUD is the end location and visualization hub for all the data being passed inside the vehicle. To ensure all data packets were parsed and displayed correctly to the HUD a novel packet structure scheme was developed. This packet structuring system enabled the HUD to quickly parse and display all the time crucial data being received from both devices. This ensured a smooth and accurate user experience while visualizing the



(a) Heads-Up Display (HUD) Completed (Top View)

(b) Heads-Up Display (HUD) Internal Wiring

Figure 5.6: Heads-Up Display (HUD) Internal and Complete Assembly

incoming vehicle and infrastructure data. There was a total of seven custom packets that were developed. The following list, and ensuing figures, detail the information about each of the packets.

1. **VIN Parameter Packet (CBX):** This packet encapsulates the Vehicle Identification Number (VIN) as data. It is transmitted by the CBX Module to the HUD Module. The general frame format is as follows:

   #C1:VIN=01234567891234567#

   Please refer to Figure 5.7 for a detailed breakdown of each portion of the packet.

2. **OBDII Parameter Packet (CBX):** This packet encapsulates the Engine RPM, Throttle Position, and Speed as data. It is transmitted by the CBX Module to the HUD Module. The general frame format is as follows:

   #C3O:A,RPM=01*A,Throttle=0*A,Speed=0#

VIN XBee Data Packet Format:

#C1:VIN=01234567891234567#

Below is a detailed breakdown of each segment of the packet:

Packet:	#	C	1	:	VIN	=	012345678901234567	#
Element:	0	1	2	3	4-6	7	8-24	25
Position:	1	2	3	4	5-7	8	9-25	26

Where:

#	:	Start and end packet delimiter
C	:	in position 2 signals the message source is from CBX Module
1	:	in position 3 indicates the message contains 1 vehicle (data) element
:	:	in position 4 is a header and data separator
VIN	:	in positions 5-7 is the parameter name
=	:	in position 8 is a data separator
0123..7	:	in positions 9-25 is 17 digit VIN number of the vehicle

Figure 5.7: VIN XBee Packet Frame Breakdown

Please refer to Figure 5.8 for a detailed breakdown of each portion of the packet.

3. **Joystick Parameter Packet (CBX):** This packet encapsulates the Brake, Throttle, and Steering Wheel Potentiometer positions (resistance)as data. It is transmitted by the CBX Module to the HUD Module. The general frame format is as follows:

   #C3J:A,Brake=0*A,Throttle=0*A,Speed=0#

   Please refer to Figure 5.9 for a detailed breakdown of each portion of the packet.

4. **Expected Value Parameter Packet (LVTT):** This packet encapsulates the expected (calculated) values for the vehicle's Brake, Throttle, and Steering Wheel positions as data. It is transmitted by the LVTT Module to the HUD Module. The expected data would theoretically be calculated based on

OBDII XBee Data Packet Format:

#C3O:A,RPM=01*A,Throttle=0*A,Speed=0#

Below is a detailed breakdown of each segment of the packet:

Packet:	#	C	3	O	:	A	,	RPM	=	01	*	A	,	Throttle	=	0	*	A	,	Speed	=	0	#
Element	0	1	2	3	4	5	6	7-9	10	11-12	13	14	15	16-23	24	25	26	27	28	29-33	34	35	36
Position	1	2	3	4	5	6	7	8-10	11	12-13	14	15	16	17-24	25	26	27	28	29	30-34	35	36	37

Where:

#	:	Start and end packet delimiter
C	:	in position 2 signals the message source is from CBX Module
3	:	in position 3 indicates the message contains 3 vehicle (data) elements
O	:	in position 4 indicates the vehicle parameters are requested from ELM-327 OBD-II interface
:	:	in position 5 is a header and data separator
A	:	in positions 6, 15, 28 indicates that the parameter is an actual value (vs expected)
,	:	in positions 7, 16, and 29 are data separators
RPM	:	in positions 8-10 is a parameter name
=	:	in positions 11, 25, and 35 are data separators
01	:	in positions 12-13 hold the 2 BYTEs allocated for RPM data
*	:	in positions 14 and 27 are data separators
Throttle	:	in positions 17-24 is a parameter name
0	:	in positions 26 and 36 hold the 1 BYTE allocated for both the Throttle and Speed Positions data
Speed	:	in positions 30-34 is a parameter name

Figure 5.8: OBDII XBee Packet Frame Breakdown

that individual vehicle's dynamics. This calculation was not in the scope of this research but could be a future expansion of this framework. The general frame format is as follows:

#L3:A,Brake=0*A,Throttle=0*A,Speed=0#

Please refer to Figure 5.10 for a detailed breakdown of each portion of the packet.

Joystick (Steering Wheel) XBee Data Packet Format:

#C3J:A,Brake=0*A,Throttle=0*A,Speed=0#

Below is a detailed breakdown of each segment of the packet:

Packet:	#	C	3	J	:	A	,	Brake	=	0	*	A	,	Throttle	=	0	*	A	,	Speed	=	0	#
Element	0	1	2	3	4	5	6	7-11	12	13	14	15	16	17-24	25	26	27	28	29	30-34	35	36	37
Position	1	2	3	4	5	6	7	8-12	13	14	15	16	17	18-25	26	27	28	29	30	31-35	36	37	38

Where:

#	: Start and end packet delimiter
C	: in position 2 signals the message source is from CBX Module
3	: in position 3 indicates the message contains 3 vehicle (data) elements
J	: in position 4 indicates the vehicle parameters are requested from the Joystick (Steering Wheel) interface
:	: in position 5 is a header and data separator
A	: in positions 6, 16 and 29 indicate that the parameter is an actual value (vs expected)
,	: in positions 7, 16, and 29 are data separators
Brake	: in positions 8-12 is a parameter name
=	: in positions 13, 26, and 36 are data separators
0	: in positions 14, 27, and 37 hold the 1 BYTE allocated for the Brake, Throttle, and Steering position data
*	: in positions 14 and 27 are data separators
Throttle	: in positions 18-25 is a parameter name
Speed	: in positions 31-35 is a parameter name

Figure 5.9: Joystick (Steering Wheel) XBee Packet Frame Breakdown

5. **Source Switch Packet (HUD):** This packet signals the CBX Module to switch to either OBDII (OB) or Joystick mode (JY). It is sent from the HUD to the CBX. The general frame format is as follows:

   #H1:SSJY# (Joystick Mode) or #H1:SSOB# (OBDII Mode)

   Please refer to Figure 5.11 for a detailed breakdown of each portion of the packet.

6. **Acknowledgment Packet (HUD):** This packet acknowledges the Source

Expected Value (LVTT) XBee Data Packet Format:

#L3:A,Brake=0*A,Throttle=0*A,Speed=0#

Below is a detailed breakdown of each segment of the packet:

Packet:	#	L	3	:	E	,	Brake	=	0	*	E	,	Throttle	=	0	*	E	,	Speed	=	0	#
Element	0	1	2	3	4	5	6-10	11	12	13	14	15	16-23	24	25	26	27	28	29-33	34	35	36
Position	1	2	3	4	5	6	7-11	12	13	14	15	16	17-24	25	26	27	28	29	30-34	35	36	37

Where:

#	:	Start and end packet delimiter
L	:	in position 2 signals the message source is from LVTT Module
3	:	in position 3 indicates the message contains 3 vehicle (data) elements
:	:	in position 4 is a header and data separator
E	:	in positions 5, 15, and 28 indicate that the parameter is an expected value (vs actual)
,	:	in positions 6, 16, and 29 are data separators
Brake	:	in positions 7-11 is a parameter name
=	:	in positions 12, 25, and 35 are data separators
0	:	in positions 13, 26, and 36 hold the 1 BYTE allocated for the Brake, Throttle, and Steering position data (expected / calculated)
*	:	in positions 14 and 27 are data separators
Throttle	:	in positions 17-24 is a parameter name
Speed	:	in positions 30-34 is a parameter name

Figure 5.10: Expected Value (LVTT) XBee Packet Frame Breakdown

Switch Packet sent from the HUD. It is transmitted by the CBX to the HUD Module. The general frame format is as follows:

#ACK:SSJY# (Joystick Mode) or #ACK:SSOB# (OBDII Mode)

Please refer to Figure 5.12 for a detailed breakdown of each portion of the packet.

7. **Acknowledgment Packet (HUD):** This packet acknowledges the Source Switch Packet sent from the HUD. It is transmitted by the CBX to the HUD Module. The general frame format is as follows:

Source Switch XBee Data Packet Format:

#H1:SSJY# (Joystick Mode) or #H1:SSOB# (OBDII Mode)

Below is a detailed breakdown of each segment of the packet:

Packet:	#	H	1	:	S	S	J	Y	#
Element:	0	1	2	3	4	5	6	7	8
Position:	1	2	3	4	5	6	7	8	9

Where:

- # : Start and end packet delimiter
- H : in position 2 signals the message source is from HUD Module
- 1 : in position 3 indicates the message contains 1 vehicle (data) element
- : : in position 4 is a header and data separator
- S : in positions 5 and 6 indicate this is a mode change command
- J Y : in positions 7 and 8 indicate that this is a acknowledgment for changing to Joystick (JY) mode [vs. OBDII (OB) mode]

Figure 5.11: Source Switch (HUD) XBee Packet Frame Breakdown

Source Switch XBee Data Packet Format:

#ACK:SSJY# (Joystick Mode) or #ACK:SSOB# (OBDII Mode)

Below is a detailed breakdown of each segment of the packet:

Packet:	#	ACK	:	S	S	J	Y	#
Element:	0	1-3	4	5	6	7	8	9
Position:	1	2-4	5	6	7	8	9	10

Where:

- # : Start and end packet delimiter
- ACK : in positions 2-4 indicate this is an acknowledgment packet
- : : in position 5 is a header and data separator
- S : in positions 6 and 7 indicate this is a mode change command (acknowledgment)
- J Y : in positions 8 and 9 indicate that this is a acknowledgment for changing to Joystick (JY) mode [vs. OBDII (OB) mode]

Figure 5.12: Source Switch (Acknowledgment) XBee Packet Frame Breakdown

#C1:Bx# (Button x Pressed), Where x = 1, 2, 3, or 4

Please refer to Figure 5.13 for a detailed breakdown of each portion of the packet.

Acknowledgment XBee Data Packet Format:

#C1:Bx# (Button x Pressed), Where $x = 1, 2, 3,$ or 4

Below is a detailed breakdown of each segment of the packet:

Packet:	#	C	1	:	B	x	#
Element:	0	1	2	3	4	5	6
Position:	1	2	3	4	5	6	7

Where:
- # : Start and end packet delimiter
- C : in position 2 signals the message source is from CBX Module
- 1 : in position 3 indicates the message contains 1 vehicle (data) element
- : : in position 4 is a header and data separator
- B : in position 5 indicates a button press
- x : in position 6 indicates which button is pressed where $x = 1, 2, 3,$ or 4

Figure 5.13: Acknowledgment XBee Packet Frame Breakdown

These simple packets were developed to decrease overhead and transmission times. The current standards, such as IEEE 802.11P, have similar performance characteristics, however, the devices that are used to Tx and Rx data have a substantially larger footprint versus the devices developed by the author. This system also takes into account the varying levels of information that is available in modern vehicles with the newest ones containing the most data versus those closer to the 1996 year cutoff.

5.1.2.2 Legacy Vehicle Data Collection Process

Section 5.1.2.1 highlighted the hardware and software solutions that were unique to vehicles that came equipped with an OBDII port. Other devices exist that accomplish the same task that the author was testing, however, the author's research is focused on integrating legacy vehicles (produced prior to 1996) into the existing systems that utilize the data provided by the OBDII port. The following explains the unique hardware and software solutions, developed by the author, that address the issue of gathering similar data points from vehicles that do not possess an OBDII port.

The first issue that must be addressed when dealing with a legacy vehicle is how to collect basic vehicle data when there is no direct connection to this information i.e. no ODBII port. Another issue occurs if the vehicle was produced before the year 1980. Before this production year, vehicles did not even contain ECUs or any other digital control device as mechanical systems were vastly used to operate all the necessary functions of the vehicle. This posses a great obstacle for an ITS that needs digital based data from even these much older, and often times non digital, automobiles. The author's research seeks to solve this issue by utilizing a smartphone based data collection and reporting system.

As stated in Section 2.2, the current market saturation of smartphones indicates that most drivers on the roadways will have a smartphone, or at the very least a passenger in the vehicle will possess one. The right portion of the high level block diagram in Figure 5.1 shows the devices that were developed by the author that integrate legacy vehicles into the overall data collection system. The following describes each of these devices in detail and highlights the data setup and collection process.

The smartphone is the key component of integrating legacy vehicles into a complete ITS framework. The image in Figure 5.14 shows a screenshot of an application (app) called SensoDuino. This app polls the smartphone's various sensors such as the accelerometer, GPS, gyroscope, etc. For the purpose of this research, the SensoDuino app was used as a stand in for the OBDII data provided by modern vehicles. While the app would not provide the same level of detail that an OBDII port would, however, the basic data of GPS location, and relative speed could be determined from the application. The author has also considered developing a custom app that could take user inputs for details such as the VIN, and number of occupants in the vehicle. The available data provided by the app is restricted to the amount of sensors that the user's smartphone possesses, however, based on the author's research no other system currently exists that performs this basic level of vehicle data reporting. This application also allows the user to transmit the real-time data from the sensors, via Bluetooth, to other devices. This property was used as the app transmitted the smartphone data to the HC-05 Bluetooth module located inside the Legacy Vehicle Telemetry Transmitter (LVTT).

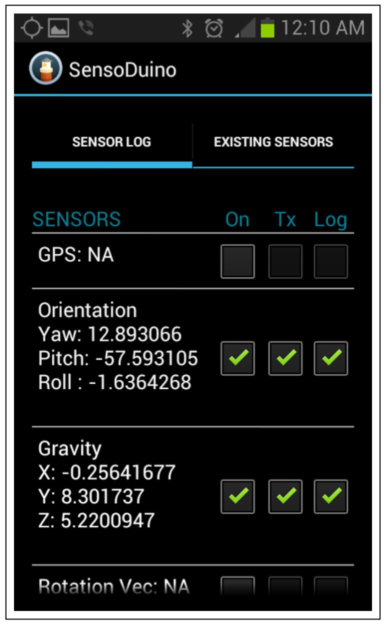
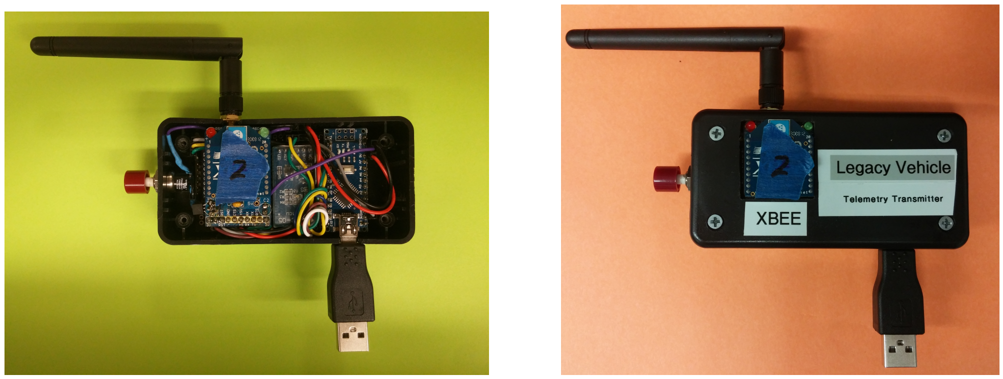


Figure 5.14: SensoDuino Android Application (APP) Screenshot

The image in Figure 5.15 shows the internal wiring and completed LVTT. The LVTT is a unique hardware solution developed by the author to solve the issue of integrating legacy vehicles into an ITS framework. The LVTT has two functions; it first serves as a reception device for all the smartphone app data, and it also received data from the wireless sensor network (WSN) located on the roadway. As denoted in the high level block diagram, the SensoDuno app was the Bluetooth alpha (α) master device. The HC-05, located inside the LVTT, acts as the slave device for the data transmission. The LVTT also contains an XBee radio that sends the received smartphone app data to the HUD to be visualized for the user.

It is also important to note that the current generation of devices that the author has developed only allows the LVTT to communicate with the WSN. In future generations of this design, the author aims to add the necessary S2 XBee to the CBX in order to fully separate the two systems.



(a) LVTT with Internal Wiring Shown (b) LVTT Completed (Top View)

Figure 5.15: LVTT Internal Wiring and Completed Assembly

The HUD is part of the the legacy vehicle data collection system. The HUD, much like with the modern collection system, is a centralized unit that allows the user to visualize the various data points that are being taken about the vehicle via the SensoDuino smartphone app. The HUD contains many visualization methods to aid the user in interpreting the data in real-time. The image in Figure 5.6 shows the completed HUD. On the top of the HUD, there are six light emitting diodes (LEDs) bar graphs with the words 'Throttle', 'Brake', and 'Steering' underneath the first, third, and fifth LED bar graph respectively. These bar graphs are used to visualize the current and expected values for the throttle, brake, and steering respective. The three tri-color LEDs labeled 'T', 'B', and 'S' will show red if the expected and current values differ, and will shine green if the expected and current values line up within a five percent margin. The images in Figures 5.16 and 5.17 show an example of this behavior.

The LED bar graphs do fall short in showing the user the exact expected and current values. To solve this issue the author added a scrolling LCD screen on the

Figure 5.16: HUD Steering Expected Value (Class 2 Vehicle)

Figure 5.17: HUD Steering Expected Value (Class 3 Vehicle)

side of the HUD that shows, among other things, the exact expected and current values of the throttle, brake, and steering. The images in Figures 5.18 and 5.19 show the side view of the HUD and the scrolling LCD menu respectively. A summary about the expected values that are displayed on the HUD is discussed in Section 5.2.3.

The scroll menu provides the user with many details other than the expected values for the throttle, brake, and steering. The HUD can receive data from either the CBX or the LVTT. The LCD menu informs the user of the class of the vehicle, current gear, RPM (CBX only), VIN, and current speed. While these parameters do not cover the wide gamut of vehicle parameters, the author used these key data points to provide a proof of concept for integrating legacy vehicles into the vehicle partition of an ITS.

Figure 5.18: Heads-Up Display (HUD) Completed (Side View)

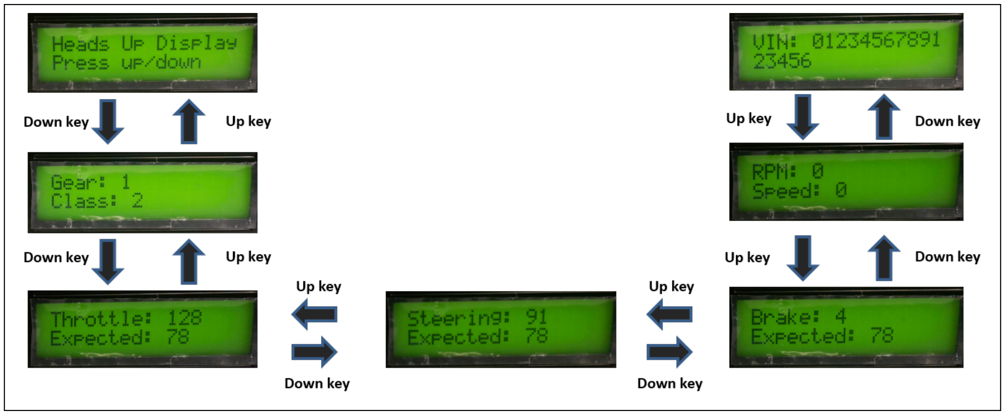


Figure 5.19: Heads-Up Display (HUD) LCD Scroll Menu

5.1.3 Vehicle-to-Vehicle (V2V) Communication

The concept of vehicles being able to communicate with one another is an important part of any ITS framework. The author did not specifically design or build any device that could accomplish this type of communication, however, the LVTT could be modified to accomplish such a task, or a custom application could be developed that could transfer vehicle data from one automobile to another in its direct vicinity. Vehicle-to-Vehicle (V2V) communication will allow vehicles to identify and propagate treacherous road information and alert other vehicles of dangerous activity in their surrounding area. The ability for legacy vehicles to also communicate with one another, as well as more modern vehicles, is an important aspect of an ITS and will likely be pursued by the author in future research.

5.1.4 On-Board Units (OBUs)

As mentioned in Chapter 2, the vast majority of devices that are designed to be compliant with the current V2V and V2I standards are bulky and force the user to place large antennas or other devices on the outside of their vehicle. This solution has not been widespread for this very reason. The author has shown two devices that can be used as OBUs, the CBX and the LVTT. Both of these devices are substantially smaller than the current OBUs offered for sale and do not require the user to modify their vehicle in any way. The author's current generation of devices are not intended to replace the current OBU models on the market, however, they do provide an alternative that could aid in expediting the adaption of OBUs in all vehicles on the roadway.

5.2 Infrastructure Partition

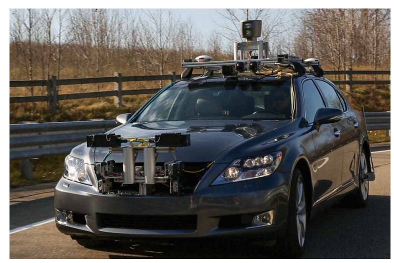
Many research methods and solutions that aim to develop autonomous vehicles tend to have one common pitfall, they all focus on making the vehicle as sensor ridden as possible to account for the ever-changing infrastructure. The image in Figure 5.20 shows some early prototype self-driving cars developed by Uber and Toyota respectively. The last few years have shown great improvements in the design and sensor integrating of vehicles, as showcased in the Tesla line of autonomous automobiles, however, the same culture of loading down the vehicle with as many sensors as possible is still prevalent in this emerging technology space.

The author has proposed a technique that would off-load some of the processing power from the vehicle to the infrastructure. This technique of characterizing and sensing the roadway conditions would allow the infrastructure to calculate the necessary trajectory for the vehicle and theoretically send digital control signals to the vehicle for parameters such as throttle position, braking, and steering angle.

This method of utilizing Road-Side Units (RSUs) to communicate with the vehicle is not new, however, characterizing and classifying the roadway so computational trajectories can be calculated and sent to the vehicle is new and was explored in this research by the author.

The following sections explain the novel hardware and software solutions developed by the author to characterize and classify key features of the roadway and surrounding infrastructure. The importance of roadways condition sensing will also be discussed with possible integration methods that would enable this feature to be added to the currently deployed system.

(a) UBER Self Driving Car Prototype



(b) Toyota Self Driving Car Prototype

Figure 5.20: Toyota and UBER Self Driving Car Prototypes [96, 97]

5.2.1 Roadway Characterization

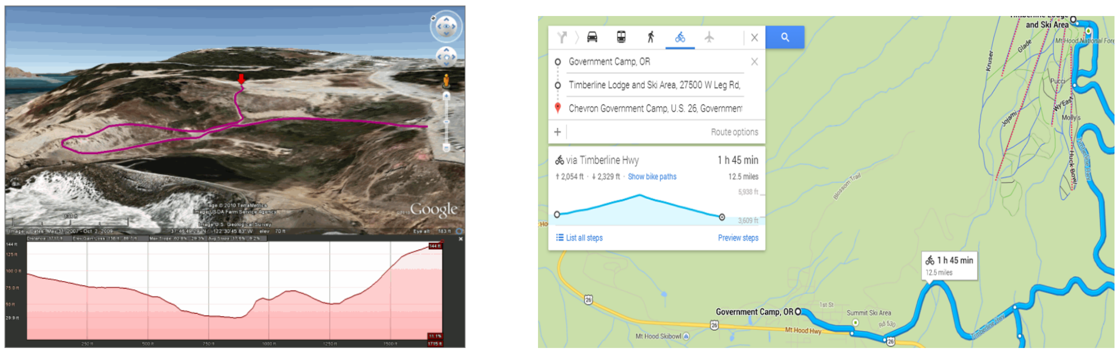
The diversity of roadway types span just as wide a spectrum as the vehicles on those roadways. This creates a unique challenge for an ITS that, so far, has not been addressed by the research community. As stated earlier the current paradigm for semi and fully autonomous vehicles is to load down the vehicle with a wide array of sensors that are able to detect and adapt the vehicles movements, in real-time, to accomplish both obstacle avoidance and navigation simultaneously. The author is proposing new novel methods that eliminate the need for vehicles to house all the navigation and obstacle avoidance computational power, but rather, off-load some of this processing and sensing tasks to the infrastructure. These novel methods and techniques for roadway classification and characterization are a needed complement to the earlier vehicle partition research methods to realize the goals of an ITS.

The author has developed a method to classify and characterize a given stretch of roadway based on simple parameters and with data provided by a custom wireless sensor network (WSN). The following sections explain, in detail, how these new methods of roadway classification and sensing operate and some of the underlying hardware and software solutions that accomplish these tasks.

5.2.1.1 Roadway Curvature & Slope Classification

The current fleet of semi and fully-autonomous vehicles use many different methods for navigation, however, they all use some form of mapping system. This could be, for example, Google™/ Apple™ Maps. These corporations have spent countless time and effort to develop these maps and these car manufacturers capitalize on this work by using these maps as a baseline for their navigation techniques. While these maps do provide a high degree of accuracy that is crucial for vehicle navigation, they often only have very basic data about the roadway available to the vehicles control system.

These companies typically provide P2P directions that can tell the user the most time efficient route to their destination, however, these services and maps currently do not include any topographical information as it pertains to roadways. The image in Figure 5.21 shows the elevation profile feature that Google Maps™ offers as an overlay layer on their software or desktop apps. This information is intended for hikers and others who need detailed info about the slope and severity of trails. While this technology exists, there appears to be no plans to expand this feature to include the slope and curvature data for roadways. The author has developed a novel technique that aims to mimic the terrain data produced for hiking and biking purposes and translate that into a workable system that provides topographical data, such as roadway curvature and slope to the driver. This system would also alleviate some of the computational burdens of navigation from the vehicle. The following explains, in detail, the novel hardware and software solutions developed to accomplish the goal of roadway curvature and slope characterization and classification.



(a) Google Maps™ Elevation Profile Feature

(b) Google Maps™ Elevation Profile Feature (Biking)

Figure 5.21: Google Maps™ Terrain Visualization Maps [98, 99]

The process of data collecting and processing of the roadway conditions and characteristics was accomplished via a custom wireless sensor network (WSN) developed by the author. The WSN consisted of two parts, a Controller Base Unit (CBU) and a Road Side Unit (RSU). It should be noted that the current IEEE 802.11P standard does incorporate RSUs into their protocol, however, a widespread deployment of devices that are compliant with this standard is currently not sufficient enough for this type of data collection. The image in Figure 5.22 shows a high level block diagram of the WSN developed by the author. The RSUs are comprised of a microcontroller, GPS unit, and a XBee S2 device. The XBees, located inside the RSUs, acted as either routers or end devices that reported to the central coordinator that is located inside the CBU. The CBU is comprised of two parts, a complete RSU, denoted as RSU0, and the Raspberry Pi (RPi) mini computer. The RPi is wirelessly linked to an off site computer running a multi-threaded python script that is used to collect the various data points from the WSN and visualize the data for the user.

The image in Figure 5.23 shows the completed CBU and one of the four RSUs that were developed and constructed for this research. All of the devices were built to be able to withstand the harsh conditions of being placed in an outdoor environment. Both the CBU and the RSUs used the EM406A GPS unit. This GPS unit was used as the basis of all data collection from the devices. The image in Figure 5.24 shows the GPS unit, as well as, the accompanying pinout of the GPS unit. The GPS that was chosen for the WSN nodes (EM406A) is compact, however, it does have a moderate margin of error when reporting coordinates. To ensure that the reported GPS data was correct a stand along, high precision, GPS was used as a validation tool to ensure the reported GPS data was accurate enough to provide reliable results moving forward. The image in Figure 5.25 shows the stand along GPS unit along side one of the RSUs. The data what was collected was averaged together and are presented in Table 5.2. The margin of error was determined to sufficient enough to proceed forward with this GPS unit.

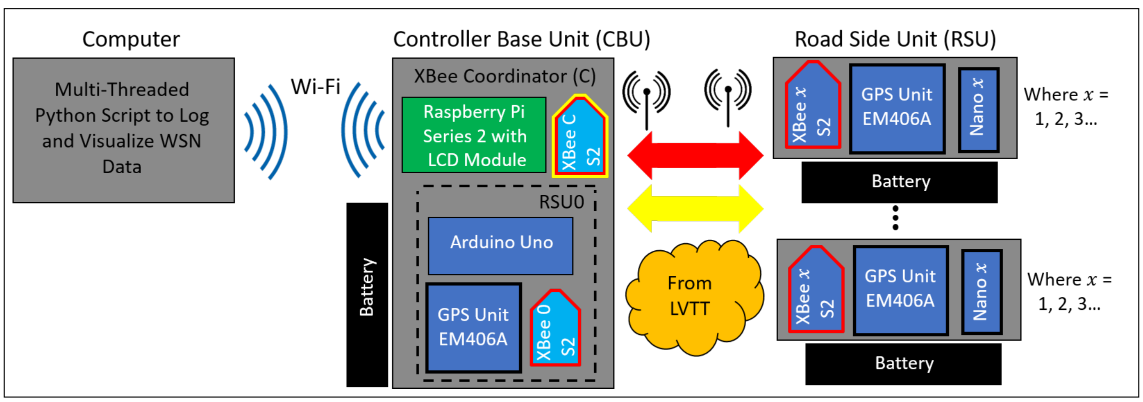
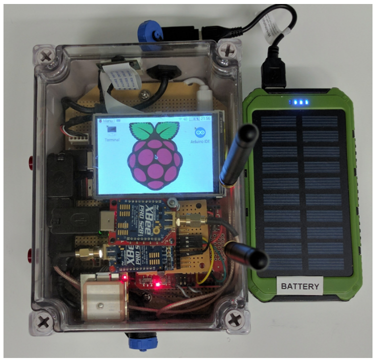
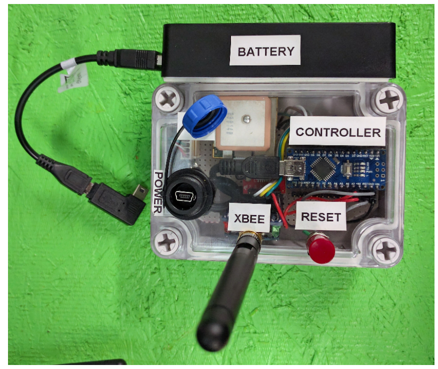


Figure 5.22: WSN High Level Block Diagram

The author's method of roadway classification consists of two parameters, curve and slope. Both of these parameters are determined by the data collected from the WSN and transmitted back to the centralized PC. The following will explain, in detail, the process of determining the curvature and slope of the roadway.

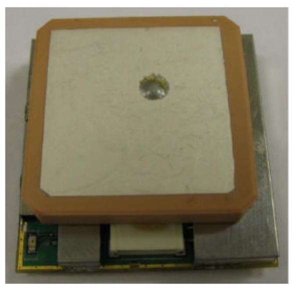


(a) Controller Base Unit (CBU) Completed (Top View)

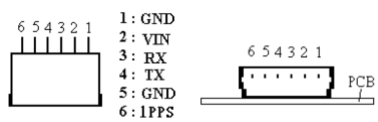


(b) Road Side Unit (RSU) Completed (Top View)

Figure 5.23: CBU and RSU Completed (Top View)



(a) EM406A GPS Module



(b) EM406A GPS Module Pinout

Figure 5.24: EM406A GPS Module and Pinout [100]

Figure 5.25: RSU Accuracy Check with Garmin GPS

Table 5.2: EM406A Data vs. Garmin GPS Data (DDM)

RSU #	**Latitude from EM406A**	**Latitude from Garmin**	**Longitude from EM406A**	**Longitude from Garmin**
RSU0	$35° \ 18.33'$	$35° \ 18.54'$	-80° $44.29'$	-80° $44.59'$
RSU1	$35° \ 18.35'$	$35° \ 18.58'$	-80° $44.30'$	-80° $44.58'$
RSU2	$35° \ 18.37'$	$35° \ 18.61'$	-80° $44.31'$	-80° $44.51'$
RSU3	$35° \ 18.39'$	$35° \ 18.64'$	-80° $44.32'$	-80° $44.50'$

To calculate the slope, the altitude and distance between each RSU is required. This distance, measured in meters, is calculated by applying the Haversine Function. Equation (5.1) shows the general form and parameters needed for the Haversine Function.

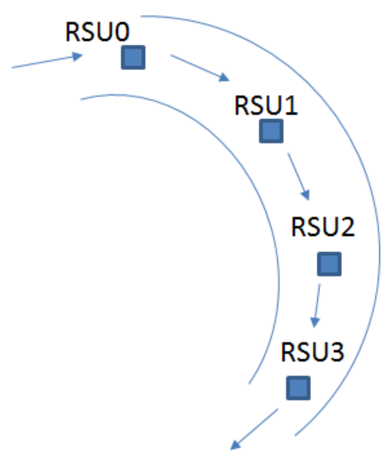
$$
\begin{aligned}
x &= \text{long } \beta - \text{long } \alpha \\
y &= \text{lat } \beta - \text{lat } \alpha \\
a &= \sin\left(\frac{y}{2}\right)^2 - \cos(\text{lat } \alpha) \times \cos(\text{lat } \beta) \times \sin\left(\frac{x}{2}\right)^2 \\
c &= 2 \times \tan^{-1}\left(\sqrt{a}, \sqrt{1-a}\right)\big) \\
D &= c \times R
\end{aligned}
\tag{5.1}
$$

as proved by [101] & [102]

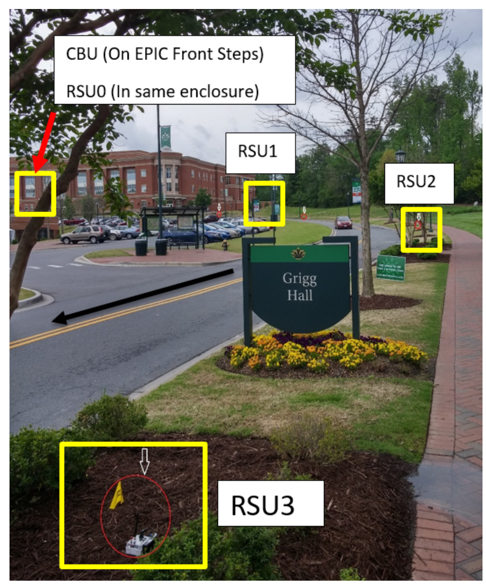
Where:

lat α	:	Latitude coordinate of first RSU
long α	:	Longitude coordinate of first RSU
lat β	:	Latitude coordinate of second RSU
long β	:	Longitude coordinate of second RSU
a	:	Square of half the chord length between the points
c	:	Angular distance (radians)
R	:	Radius of Earth $\cong$ 6370.0 kilometers(km)
D	:	Final, calculated distance between two RSUs (in Meters)

The Haversine Function is a mathematical formula that uses latitude and longitudinal data to determine the distance between to points. In order for the Haversine Function to accurately determine the distance between two of the RSUs nodes the GPS coordinates needed to be distinguishable from one another. To optimize the results from the Haversine Function the RSU nodes needed to be placed a sufficient distance away from one another so their GPS locations could be unique and not overlap one another. The image in Figure 5.26 shows the ideal placement of each RSU for a high location resolution and it also shows the actual sensor arrangement that was deployed by the author. Note that the RSUs are placed at the beginning, apex, and end of the given curve.



(a) Ideal RSU Placement for Curve Calculation(s)



(b) RSU Deployment on campus

Figure 5.26: Ideal and Actual RSU Deployment

To validate the processes of using the Haversine Function, a known, control, roadway was tested. To ensure that the extremes that the WSN could face it was determined that the extremely curved Lombard Street would be most extreme curvature condition. The image in Figure 5.27 shows the coordinates that were placed into the Haversine Function via Google Maps™. The two coordinates were placed into the formula and the results are shown in the image in Figure 5.28. The curve angle for this particular section of Lombard Street was approximately 30.3 degrees while the distance between the two points was calculated to be approximately 17.7 meters. Based upon this data the author used a six class system to classify the curvature for a given stretch of roadway. Table 5.3 shows the breakdown of each class based on the calculated curve angle.

Table 5.3: Curve Classification (from Haversine "c" Parameter)

Class	**Degree of Road Curvature (from Haversine "c" Parameter)**
1	$157^\circ < c \leq 180^\circ$
2	$133^\circ < c \leq 157^\circ$
3	$109^\circ < c \leq 133^\circ$
4	$85^\circ < c \leq 109^\circ$
5	$62^\circ < c \leq 85^\circ$
6	$38^\circ < c \leq 62^\circ$

The arbitrary assignment of a six class system was based on the scheme that is used in rally car racing. They classify the courses curves based on a six class system as well. The purpose of classifying curves in this manner is two fold. First, the drivers of legacy and modern vehicles benefit from this data collection the curvature class informs them of any potentially dangerous curves on their intended path of travel far before they each them. Secondly, once a portion of roadway has been given a curvature class, control algorithms could calculate the optional travel path for a vehicle based on their specific dynamics and this roadway curvature classification. Curve classification is one of two crucial steps needed to fully classify a given stretch of roadway.

The second crucial step of classifying a roadway is to characterize and classify the varying slopes that a given roadway possesses. As stated earlier all of the major mapping companies provide this topographical information for hikers and bikers about a given terrain, however, this information is currently not available or accessible for vehicle drivers about the slope of a given stretch of highway. The following explains the calculation and implementation of the novel slope characterization and classification technique developed by the author.

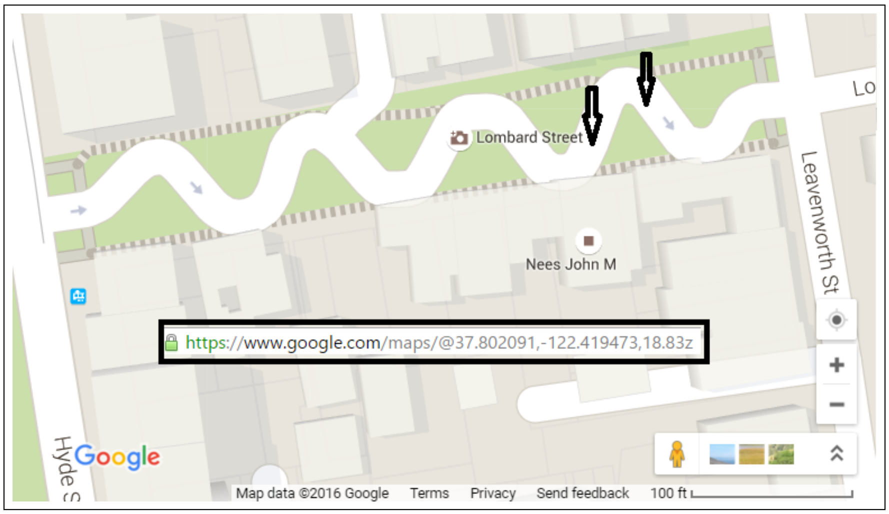


Figure 5.27: Haversine Function Validation using Lombard Street

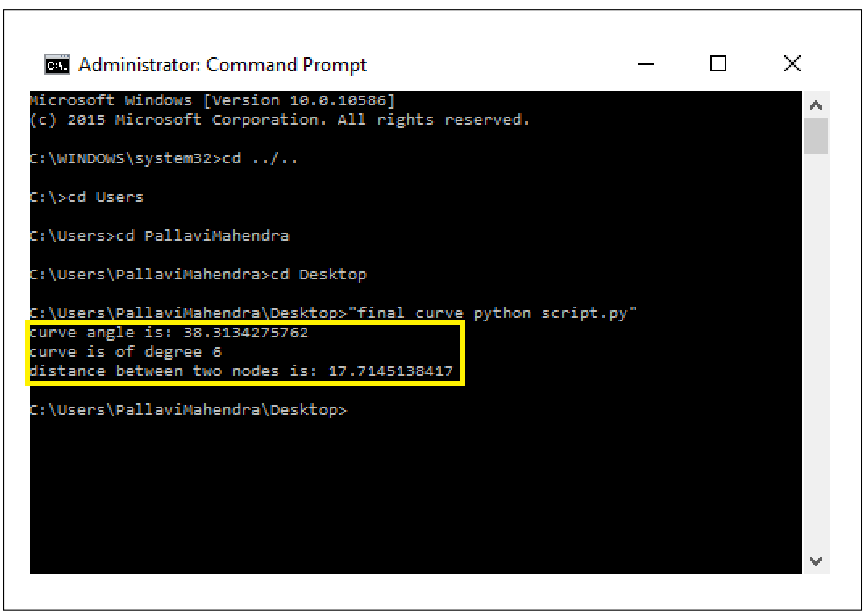


Figure 5.28: Curve Calculation for Lombard Street

To determine the slope of a given portion of highway the GPS altitude data was extracted from each node in relation to one another, i.e.the altitude difference between RSU0 and RSU1, RSU1 and RSU2, etc. The ideal placement of the RSUs to capture this information would be to locate them at any local extrema relative to the roadway. The sensor density would determine the resolution and granularity that this model provided. The image in Figure 5.29 shows a visual representation of how the slope between two RSU nodes is determined using the change in horizontal distance versus the change in altitude (both based on GPS data from the RSUs).

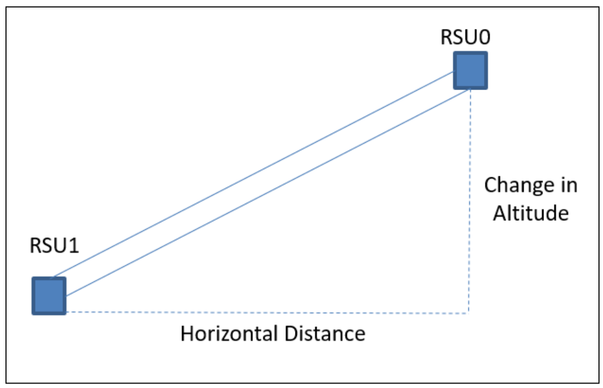


Figure 5.29: RSU Slope Calculation

To find out the slope percentage between two RSUs, the basic rise over run mathematical formula was used. Equation (5.2) shows the general form and parameters needed for the Rise over Run Formula.

$$\begin{aligned} \text{Run} &= \sqrt{\text{D}^2 - |(\text{alt } \beta - \text{alt } \alpha)|^2} \\ \text{Slope \%} &= \frac{|(\text{alt } \beta - \text{alt } \alpha)|}{\text{Run}} * 100 \end{aligned} \tag{5.2}$$

Where:

alt α	:	Altitude measurement of first RSU
alt β	:	Altitude measurement of second RSU
D	:	Distance between two RSUs
Run	:	Vertical change of distance between the two RSUs
Slope %	:	Slope percentage between the two RSUs

After the slope grade percentage was calculated, a five class system was used to assign a slope class to that particular distance between RSUs. The data in Table 5.4 shows the breakdown of each slope percentage range used for the five classification system. Baldwin Street in New Zealand is classified as the steepest roadway in the world and it contains a grade percentage of approximately 35%. It is important to note that these slope percentages are typically determined over a long stretch of roadway. For tests that involve relatively short distances (i.e. less than 100 feet between points) between data points a scaled version of this table will need to be considered. d

Table 5.4: Slope Classification

Class	Degree of Road Slope
0	0% – 20%
1	20% – 40%
2	40% – 60%
3	60% – 80%
4	80% – 100%

The slope and curve classifications were combined together to provide the ITS framework with a complete topographical picture of the mapped roadway. This data could be applied to advanced control algorithms that would take the individual vehicle dynamics into account and thus provide a higher level of accuracy in determining the optimal path of travel for the vehicle. This could also lead to the creation / adoption of hybrid vehicles that could be non or semi-autonomous after retrofitting them with actuators or other mechanisms to take control of the vehicle from the driver and perform the driving maneuvers based on these dynamic calculations.

5.2.1.2 Roadway Condition Sensing

The current iteration of the custom WSN developed by the author does not offer any condition sensing of the roadway. In future iterations of the WSN the RSUs and the CBU could contain environmental sensors that could report roadway data such as the temperature, moisture content, wear level, etc. These senors would allow the centralize control algorithm more data inputs that would, in turn, allow the algorithm to correct for harsh or abnormal driving conditions before it sends the digital control signals to the vehicle.

5.2.2 Vehicle-to-Infrastructure (V2I) Communication

After the given roadway has been characterized and classified, by the methods discussed in Section 5.2.1, this data / control algorithms are ready to be transmitted back to the vehicle. The setup algorithms for the CBU will be discussed in this section, while information regarding the setup algorithms for the RSUs will be discussed in detail in Section 5.2.3.

The CBU, in the authors WSN, is the central communications coordinator between the RSUs, Off-site computer, and the XBee network located inside the vehicle via the LVTT. All of this data is collected by the CBU via a multi-threaded python script that polls for each of these data points. The flow chart in Figure 5.30 shows the data flow of the python script, as well as, the commands used to execute each portion of the script. A detailed execution flow of the CBU (coordinator) script is outlined in Algorithm 3.

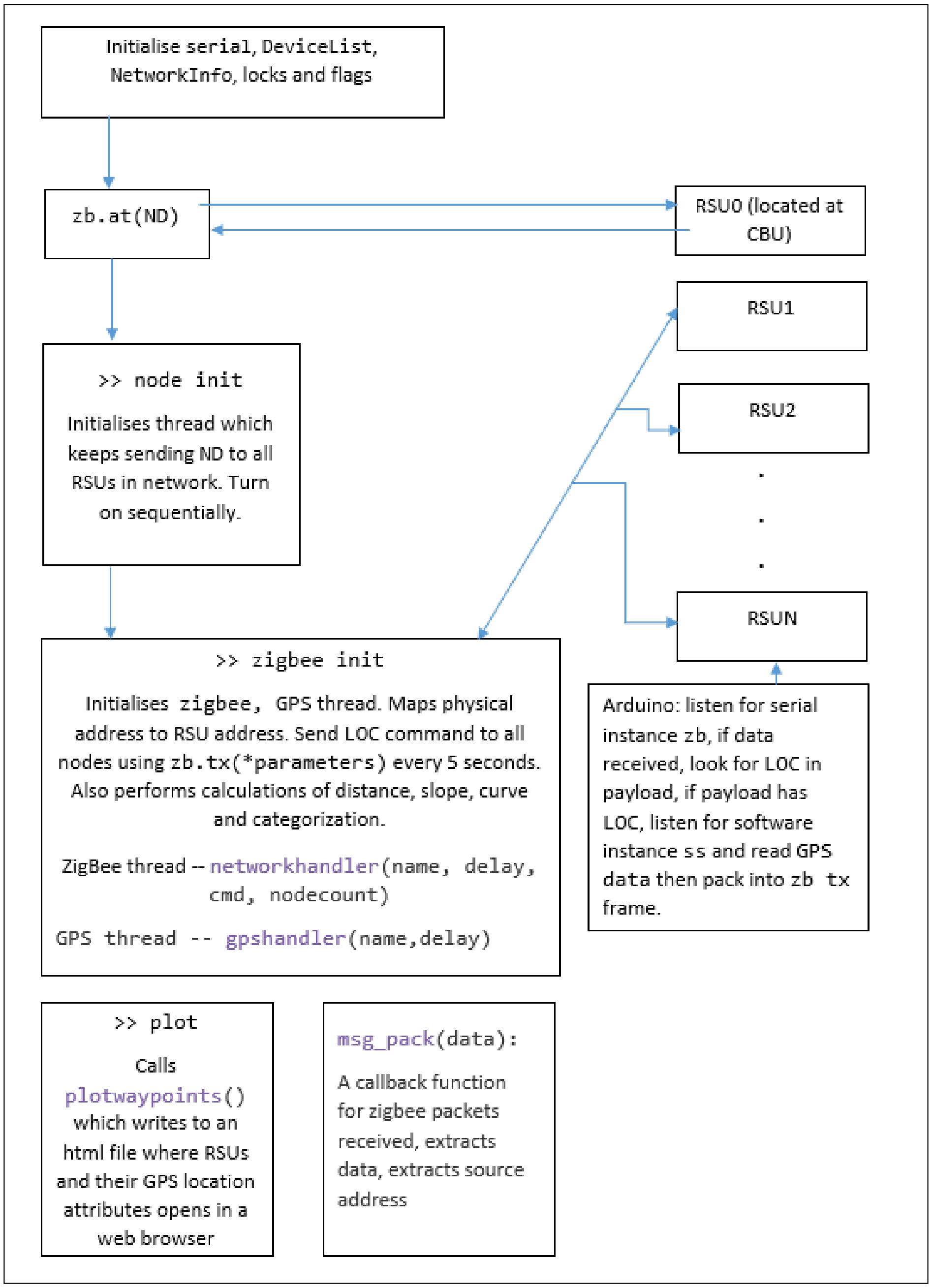


Figure 5.30: WSN Python Thread Visualized

The multi-threaded Python script was developed and deployed on the Raspberry Pi (RPi) board located inside the CBU. The RSUs are switched on one after the other so that the CBU knows their sequence of deployment. Initially, the CBU acquires the 64-bit physical addresses of all the RSU nodes which are in network sequentially. It periodically keeps checking for every new nodes added or deleted from the network.

Algorithm 3 CBU (Coordinator) Python Script (General Execution Flow)

Steps

1: Starts script with AT command `ND` into the network, at system startup, only the RSU located in the CBU is powered on and available to the incorporated into the network.

2: The reply is in the form of a python dictionary, the value corresponding to `parameter` is a list which has the `source_addr_long` key. The value corresponding to this key is the address represented in hex dec byte array.

3: This reply is stored into a list called `DeviceList`.

4: Once the terminal shows the message `'device added to the list...'` , the `node init` command is typed into the terminal. This initiates a node discovery thread every twenty seconds. This thread keep track of all the nodes introduced into the network.

5: Sequentially turn on RSUs, one at a time, in the direction of mapping. After the message `'device added to the list...'` and the appended list is displayed on the terminal, turn on the next RSU and so on until all RSUs have been added to the list.

6: Wait for one to two minutes for the GPS sensors, located in the RSUs, to lock onto at least four satellites (A flashing GPS sensor indicates a successful lock)

7: The terminal will now display a >> indicating it is ready for the next command. Type the command `zigbee init` into the terminal. This initializes the zigbee thread. This thread pings all the RSUs in the network with the `LOC` command in payload. The routers respond with their GPS coordinates. This command also initializes the gps thread which sequentially calculates the slope, curve, angle, and distances of the RSUs using the `SlopeDistance` and `CurveCalculator` modules, written locally in python.

8: The received data is in the form of a python dictionary. To extract data from it, make sure for the key `'id'` the value is `rx`. If it is `rx`, then the data is in the value corresponding to the `rf_data` key.

9: The `zigbee` thread also maps the addresses in `DeviceList` to a python dictionary called `NetworkInfo`. This dictionary binds together the human readable RSU ID, with the `NetworkInfo` dictionary. An example of the `NetworkInfo` dictionary, that contains the RSUs latitude, longitude and altitude is as follows:

```
\x00\x12..@n:[RSUx,[\x00,\x12..\x40\x6E],35.12345,-80.34562,140].
```

10: This is now a continuous running program; you can plot the GPS points using the `plot` command. The `plot` command writes to an html script that dynamically appends the coordinate information from all current RSUs then opens up a browser to display the coordinates. An example output can be seen in the screenshot in Figure 5.32.

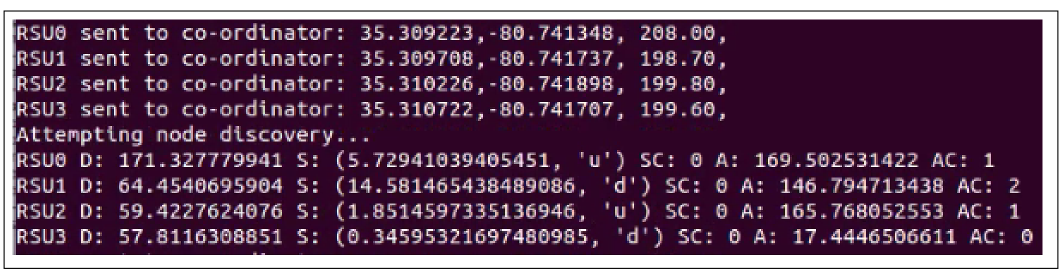
11: To kill all threads and exit the program type the `exit` command into the terminal

Once all sensors are deployed and the GPS in each RSU is locked onto at least four satellites, the CBU continuously receives the real-time location data from all the RSUs. Altitude data of two consecutive RSUs are used to calculate the slope between them using the rise over run formula. The Haversine Function is calculated using the longitude and latitude data, provided by the GPS, to determine the curve classification between two consecutive nodes.

All of the collected data is them parsed and displayed in two forms; terminal output and a dynamically updating HTML Google Maps ™ page. The general form of the output from the python script, visualized in the terminal, is as follows:

```
RSUX D: (Distance) S: (Slope, up 'u' / down 'd') SC: (Slope Class) A:
↪  (Angle) AC: (Angle Class a.k.a Curve Class)
```

The image in Figure 5.31 shows a snipped of terminal output that resulted from the on-campus test with nodes placed as shown in Figure 5.26b. The data was also visualized in an dynamically updating HTML page that placed all of the location data,, and in combination with some Google Maps ™ API, onto a Google Maps Google Maps ™ HTML page. The image in Figure 5.32 presents the same data presented in Figure 5.31. The dynamic HTML page was used as a secondary means of visualizing the data to ensure the GPS data was accurate and the RSUs location data was within acceptable margins of error. Section 5.2.3 will give some details as to how the RSUs were configured and deployed.



```
RSU0 sent to co-ordinator: 35.309223,-80.741348, 208.00,
RSU1 sent to co-ordinator: 35.309708,-80.741737, 198.70,
RSU2 sent to co-ordinator: 35.310226,-80.741898, 199.80,
RSU3 sent to co-ordinator: 35.310722,-80.741707, 199.60,
Attempting node discovery...
RSU0 D: 171.327779941 S: (5.72941039405451, 'u') SC: 0 A: 169.502531422 AC: 1
RSU1 D: 64.4540695904 S: (14.581465438489086, 'd') SC: 0 A: 146.794713438 AC: 2
RSU2 D: 59.4227624076 S: (1.8514597335136946, 'u') SC: 0 A: 165.768052553 AC: 1
RSU3 D: 57.8116308851 S: (0.34595321697480985, 'd') SC: 0 A: 17.4446506611 AC: 0
```

Figure 5.31: Slope and Curve Classification Terminal Output from RPi

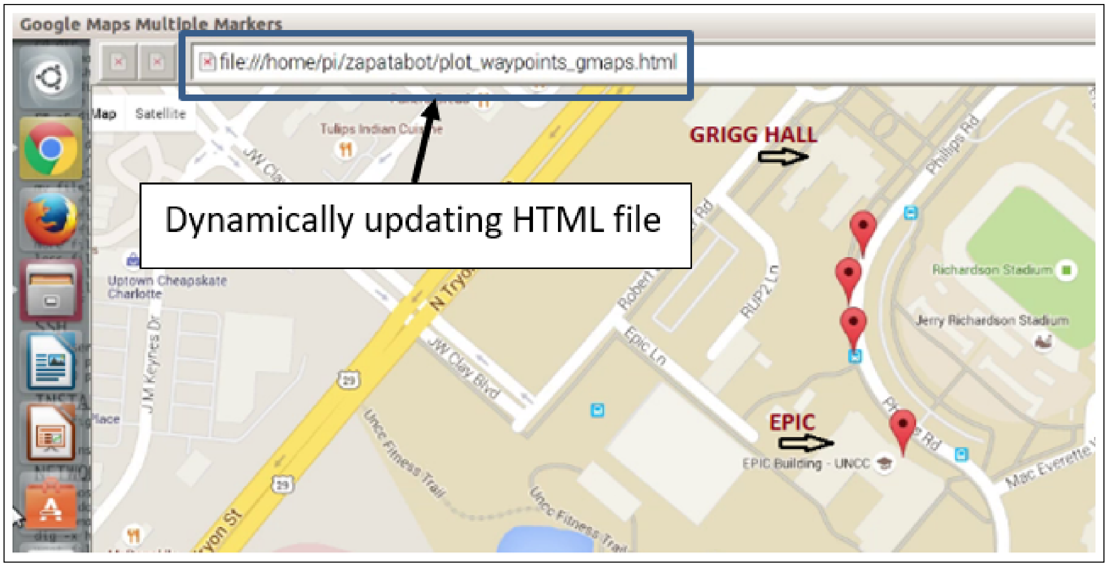


Figure 5.32: Dynamic Google Maps ™ HTML Page for RSU Deployment

5.2.3 Road-Side Units (RSUs)

The deployment of RSUs, in regards to an ITS framework, should be easier to deploy than their OBU counterparts. RSUs are suppose to be designed to be easily integrable into the surrounding infrastructure of a highway. The adoption of this ITS framework portion has also been slow to occur. While there are many manufactures that produce RSUs, they typically are at a very high cost and they must be paired with their brand of OBUs in order to function properly. The author's approach, while not IEEE 802.11P compliant, does provide a more open approach to tackling to this issue. As stated earlier each of the RSUs developed by the author contained an MCU, GPS, and S2 XBee. The modular design of the RSUs allows for future expansion. The deployment process for RSUs has been outlined in Section 5.2.2. The execution flow of the RSU (router / end device) script is outlined in Algorithm 4.

Algorithm 4 RSU (Router / End Device) Python Script (General Execution Flow)

Steps

1: The Arduino Nano, located in each RSU, boots up with some initializations, such as creating `SoftwareSerial` instances: `zb` and `ss` for ZigBee and GPS respectively.

2: The two software serial instances `zb` and `ss`, share the same receive buffer, therefore the recieve buffer is sequentially polled to a particular `SoftwareSerial` instance to pull data from it.

3: The Arduino Nano continuously polls for the zigbee packet, once a zigbee packet is received, its response is compared to check if it corresponds to the desired receive packet. If the packet is correct then the payload is read into a `char` array called `data`.

4: The received data is compared using the command `LOC`.

5: If the `LOC` is the same then the Arduino Nano listens to the `SoftwareSerial` instance `ss` sequentially via the call `printFloat(float, bool, int, int)` which concatenates the `latitude`, `longitude` and `altitude` into a `char` array called `reply`.

6: The contents from `reply` are copied into an `uint8_t` array called `payload` via the `charToBuf()` call.

7: Finally the `payload` is packed into xbee packet using the xbee library on the Arduino Nano and Transmitted over the network.

5.3 Method Comparison

One of the major portions of the framework mentioned thus far relies on sensor data to calculate the topology data (i.e. curvature and slope) of the roadway. This method of using RSUs was implemented in an attempt to mirror the IEEE 802.11P standard, however, topology data can be gathered via other means. These alternative methods, aside from physical roadway sensors, need to be explored as one of the hindrances of the widespread distribution of the IEEE 802.11P standards is the necessity of placing RSUs along every possible highway route. Typically, these RSUs are used only for establishing a communication link with the OBUs located in each vehicle (ideally). Since these RSUs do not gather any other data it could be argued that an alternative, sensorless approach, could be used to aid in the deployment of an ITS. Two such approaches, and how they compare to using physical sensors follow.

The first repository of data is the United States Geological Survey (USGS) data. The data present in this repository is part of the National American Datum of 1983 (NAD83). NAD83 is the horizontal control datum for the United States and most of the northern hemisphere that is based on a geocentric origin and the Geodetic Reference System 1980 [103]. The method of topology classification mentioned thus far utilizes the GPS coordinates (longitude and Latitude values) along with the altitude data to perform the novel classification technique. The Elevation Point Query Service is a webpage that can be used to access the altitude data of the USGS database. Figure 5.33 shows a snapshot of this webpage. A user can enter the x (longitude), y (latitude) point from the USGS 3DEP 1/3 arc-second layer hosted at the National Geospatial Technical Operations Center NGTOC. Based on the USGS data the webpage will produce an elevation, in meters or feet, for that point in an Extensible Markup Language (XML) format. Figure 5.34 shows an example XML output given one GPS longitude and latitude data point. This service could be automated to reach out to this webpage and query for the elevation data based on the real-time GPS data of a vehicle on a given stretch of highway.

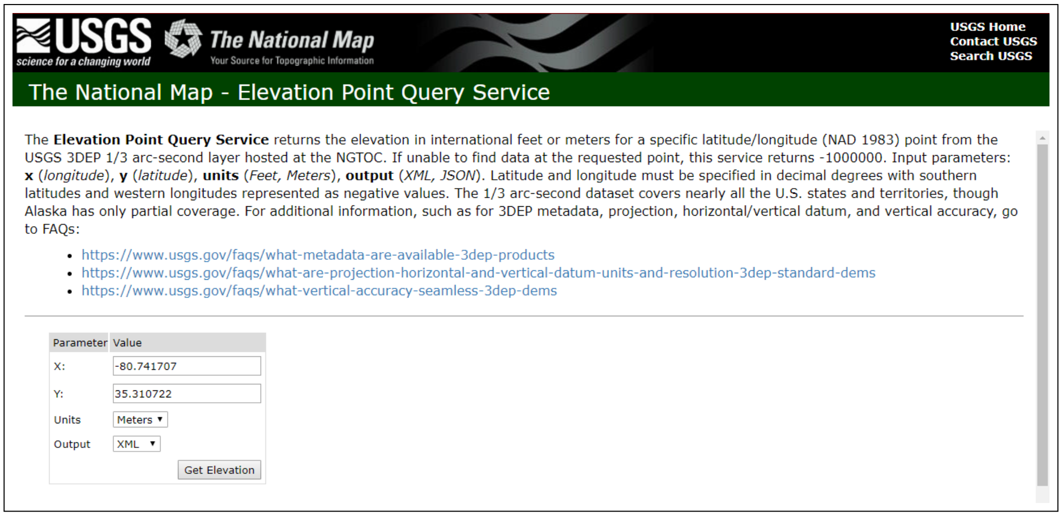


Figure 5.33: USGS Elevation Query Service Webpage [103]

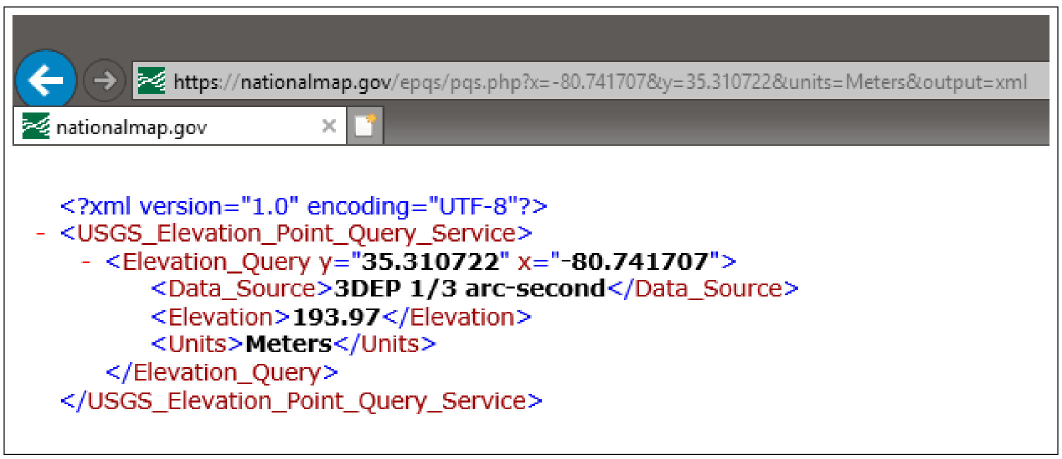


Figure 5.34: USGS Elevation Query Service XML Output [103]

The second repository of elevation data can be accessed via a Google Maps ™ API webpage. Figure 5.35 shows the help page where the data and commands used to query the Google Maps ™ elevation database. The access method is similar to that of the USGA database. A simple JavaScript Object Notation (JSON) file format is used as an output method for any query of the master elevation database. One key difference from the UDGS elevation method is the lack of a graphical webpage where a user can enter the specific longitude and latitude point. To query the elevation data via the Google Maps ™ API the longitude and latitude points must be manually entered into a single Uniform Resource Locator (URL) inside a browser environment. Figure 5.36 shows an example query response given a specific longitude and latitude location.

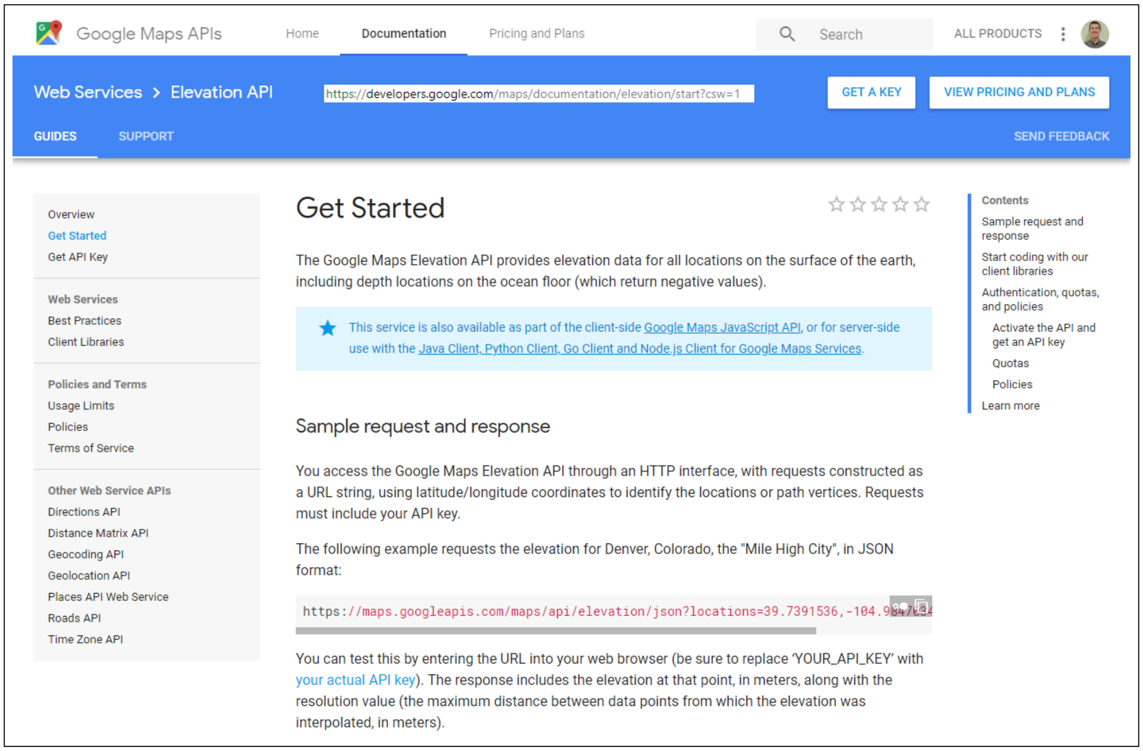


Figure 5.35: Google ™ Maps Elevation API Webpage [104]


```
{
   "results" : [
      {
         "elevation" : 193.9542999267578,
         "location" : {
            "lat" : 35.310722,
            "lng" : -80.74170700000001
         },
         "resolution" : 4.771975994110107
      }
   ],
   "status" : "OK"
}
```


https://maps.googleapis.com/maps/api/elevation/json?locations=35.310722,-80.741707&key=AIzaSyAnHCUpPvQIcngtgGHhthJWfL2nT1S4QI0

Figure 5.36: Google ™ Maps Elevation API JSON Output File [104]

Given that the altitude data for any given GPS coordinate could be attained through one of these two methods, a field test was performed to test the concept of a sensorless framework where the topology for a given stretch of roadway could be characterized without the need for any sensors to be physically deployed. The aerial Google Maps ™ photo of the sensor / sensorless placement scenario is shown in Figure 5.37. To compare the two methods the high accuracy Garmin GPS was used to represent a sensorless approach where any device that could collect accurate GPS coordinates could be used in lieu of the RSUs. Since a limited numbers of physical RSUs were constructed multiple tests were conducted and the results combined to achieve the same eleven sensor layout shown in Figure 5.37

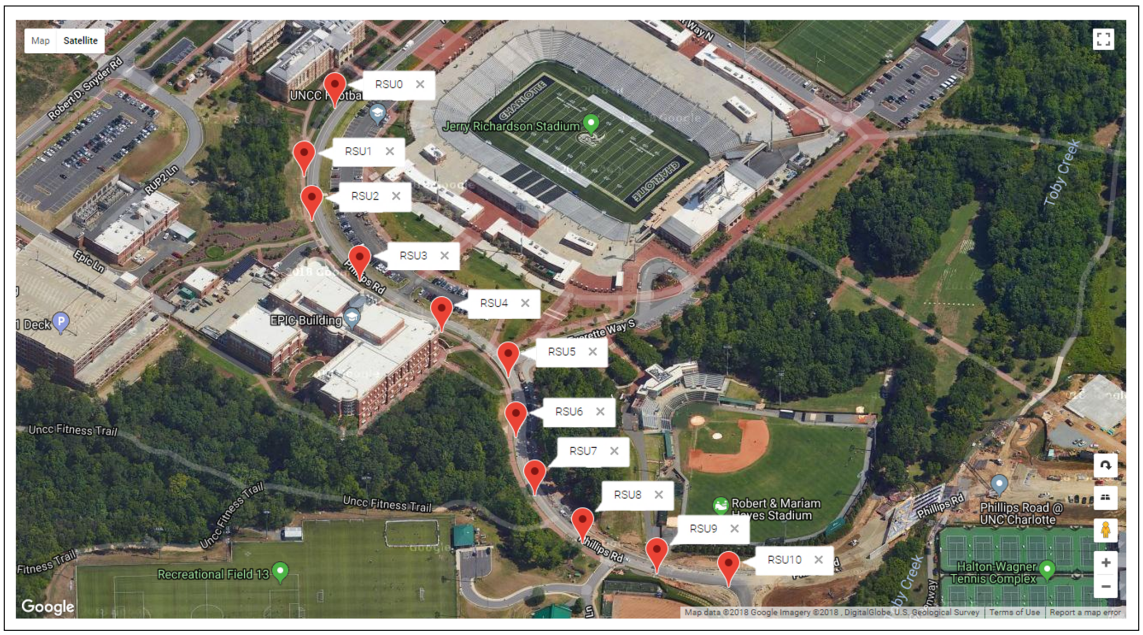


Figure 5.37: 2D Google Map ™ RSU Placement for Method Comparison

The collected GPS data for these two scenarios can be seen in Tables 5.5 and 5.6. These data point were then passed through the main python script to characterize the topology of the roadway scenario. The results of this processing, along with the USGA XML and Google Maps ™ Elevation requests were tabulated and shown in Tables 5.7 and 5.8 respectively. It is important to note that the deviation from the USGA and Google Maps ™ data were within a few percent of one another. This was partly due to Google ™ using USGA data in some of their own mapping data.

Table 5.5: RSU GPS Data (Using Garmin GPS Data)

RSU #	**Latitude**	**Longitude**	**Altitude (m)**	**USGA Elevation (m)**	**Google Elevation (m)**
RSU0	35.310722	-80.741707	199.60	193.97	193.9543
RSU1	35.310226	-80.741898	199.80	196.01	196.1198
RSU2	35.309900	-80.741850	202.08	197.39	197.4743
RSU3	35.309467	-80.741550	203.00	198.35	198.4967
RSU4	35.309100	-80.741033	193.55	196.90	196.9732
RSU5	35.308767	-80.740617	192.94	193.48	193.5948
RSU6	35.308333	-80.740567	190.50	191.00	190.9830
RSU7	35.307917	-80.740450	188.37	188.24	188.2165
RSU9	35.307567	-80.740150	184.71	185.93	185.9257
RSU9	35.307350	-80.739683	182.88	184.91	184.8891
RSU10	35.307250	-80.739233	181.97	183.79	183.5773

Table 5.6: RSU GPS Data (Using RSU GPS Data)

RSU #	**Latitude**	**Longitude**	**Altitude (m)**	**USGA Elevation (m)**	**Google Elevation (m)**
RSU0	35.310733	-80.741717	193.97	193.97	193.9632
RSU1	35.310225	-80.741883	196.01	196.01	196.0118
RSU2	35.309967	-80.741867	197.48	197.48	197.2057
RSU3	35.309433	-80.741517	198.58	198.58	198.4601
RSU4	35.309150	-80.741017	196.27	196.27	196.5657
RSU5	35.308767	-80.740617	193.48	193.48	193.5948
RSU6	35.308283	-80.740517	190.77	190.77	190.5891
RSU7	35.307933	-80.740512	187.76	187.76	187.8350
RSU9	35.307583	-80.740133	185.93	185.93	185.9660
RSU9	35.307350	-80.739683	184.91	184.91	184.8891
RSU10	35.307217	-80.739233	183.66	183.66	184.0412

Table 5.7: RSU Calculation Results (Using Garmin GPS Data)

Raw Data Output from Multi-Threaded Python Script	**Slope Class**	**Angle Class**
RSU0 D: 446.593238098 S: (3.95074352055, 'u') SC: 0 A: 149.823550482 AC: 2	0 'u'	2
RSU1 D: 57.8116308851 S: (0.34595321698, 'u') SC: 0 A: 17.4447610602 AC: 0	0 'u'	0
RSU2 D: 36.5102692161 S: (6.25703079913, 'u') SC: 0 A: 173.148596324 AC: 1	0 'u'	1
RSU3 D: 55.3100491024 S: (1.66358068951, 'u') SC: 0 A: 150.516719664 AC: 2	0 'u'	2
RSU4 D: 62.1781664376 S: (15.3768917389, 'd') SC: 0 A: 131.019377495 AC: 3	0 'd'	3
RSU5 D: 52.8769917656 S: (1.15369761844, 'd') SC: 0 A: 134.448154606 AC: 2	0 'd'	2
RSU6 D: 48.4714040969 S: (5.04028597163, 'd') SC: 0 A: 174.629102259 AC: 1	0 'd'	1
RSU7 D: 47.4598061057 S: (4.49253498317, 'd') SC: 0 A: 167.074356150 AC: 1	0 'd'	1
RSU8 D: 47.4941348820 S: (7.72919907842, 'd') SC: 0 A: 145.027910304 AC: 2	0 'd'	2
RSU9 D: 48.7646571519 S: (3.75536311366, 'd') SC: 0 A: 119.657167224 AC: 3	0 'd'	3
RSU10 D: 42.320879365 S: (2.15073613100, 'd') SC: 0 A: 105.232723407 AC: 4	0 'd'	4

Table 5.8: RSU Calculation Results (Using RSU GPS Data)

Raw Data Output from Multi-Threaded Python Script	**Slope Class**	**Angle Class**
RSU0 D: 451.282002205 S: (2.28519896838, 'u') SC: 0 A: 150.036206044 AC: 2	0 'u'	2
RSU1 D: 58.4608066668 S: (3.49164385240, 'd') SC: 0 A: 14.9308993930 AC: 0	0 'u'	0
RSU2 D: 28.7250037314 S: (5.12407031020, 'u') SC: 0 A: 177.102906912 AC: 1	0 'u'	1
RSU3 D: 67.3378065533 S: (1.63377284362, 'u') SC: 0 A: 151.859465181 AC: 2	0 'd'	2
RSU4 D: 55.2148481694 S: (4.18732339116, 'd') SC: 0 A: 124.744651782 AC: 3	0 'd'	3
RSU5 D: 55.9563864931 S: (4.99223536101, 'd') SC: 0 A: 139.559969927 AC: 2	0 'd'	2
RSU6 D: 54.5779546020 S: (4.97150718682, 'd') SC: 0 A: 170.429587072 AC: 1	0 'd'	1
RSU7 D: 38.9208688767 S: (7.75687169847, 'd') SC: 0 A: 179.332078045 AC: 1	0 'd'	1
RSU8 D: 51.9362604932 S: (3.52573914813, 'd') SC: 0 A: 138.533640706 AC: 2	0 'd'	2
RSU9 D: 48.3596212580 S: (2.10966697762, 'd') SC: 0 A: 122.394309193 AC: 3	0 'd'	3
RSU10 D: 43.429560246 S: (2.87941686640, 'd') SC: 0 A: 109.908837255 AC: 3	0 'd'	3

When comparing the sensor versus the sensorless output data, the results suggest that both methods provided relatively the same topology characterization result. It can be concluded that if the RSUs are only being used to relay vehicle data then using the already available altitude and GPS data, provided by the USGA or similar services, would be a more feasible approach in the scope this framework model. It is important to note that the RSUs could be equipped with environmental sensors or at least adopt other functions that simply relay vehicle data back to a centralized repository. Also, research shown in [105–107] indicates that image processing could be used on Google Maps™ to characterize the curvature for a given stretch of roadway.

CHAPTER 6: RESULTS

To validate the author's framework, an off-road test vehicle (rover) equipped with a compass, GPS unit, myRIO controller, and USB webcam was constructed to act as a vehicle stand in. The off-road vehicle was remotely controlled via a game controller with dual analog joysticks. This scaled setup provided the author a platform to validate the aforementioned WSN data and characterization technique. Figure 6.1 shows the High Level Block diagram of the test setup. The myRIO controller has a built in wireless access point that the PC used to wirelessly connect and control the vehicle. The test vehicle / rover is shown in Figure 6.2.

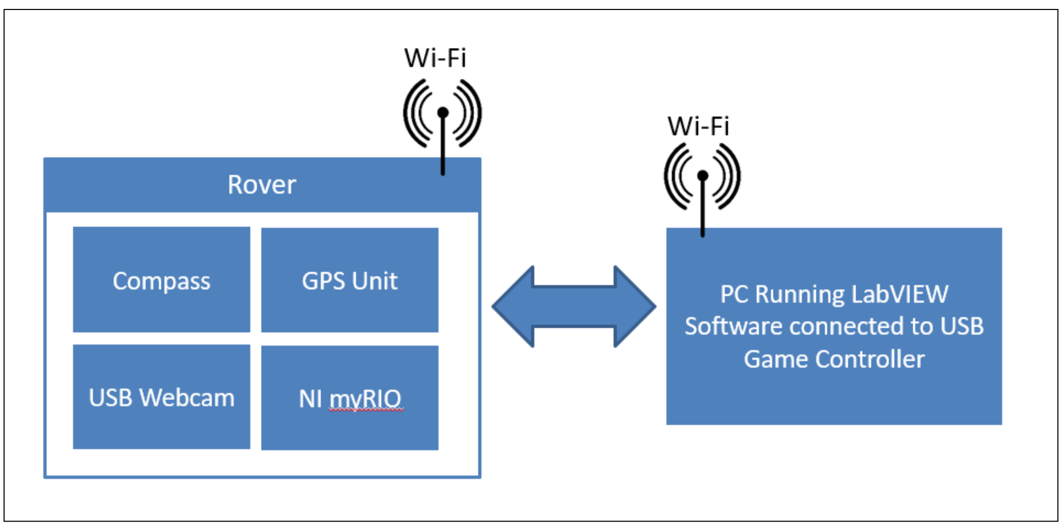


Figure 6.1: Validation High Level Block Diagram

It was determined that given the small size of the rover, in comparison to a full-size vehicle, a smaller test area should be used to test the various components of the WSN data (i.e. curvature and slope classification and characterization). To this end the walking path behind the EPIC building was chosen as a roadway replacement for the

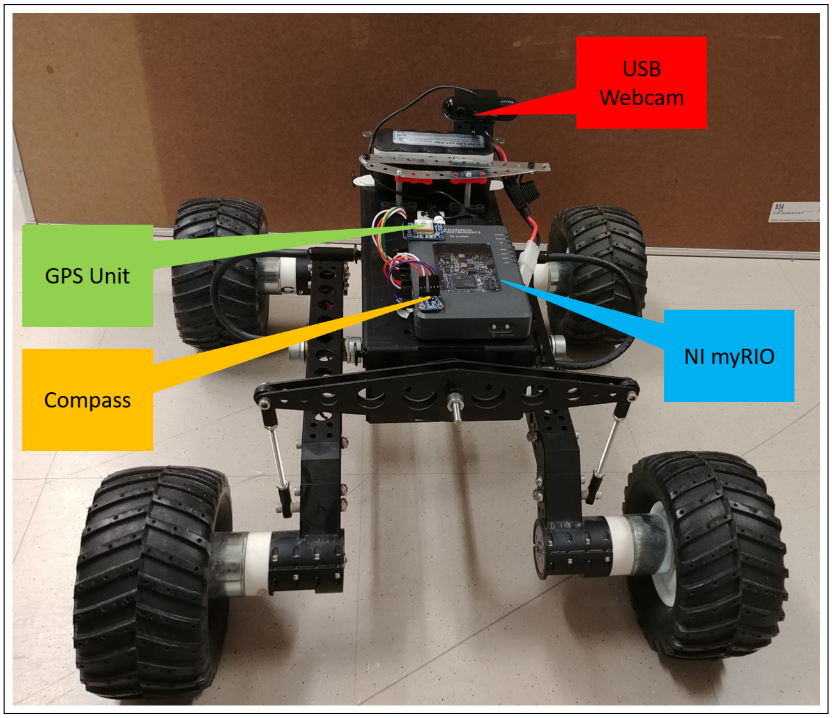


Figure 6.2: Annotated Rover Test Vehicle

rover's test track. The Google Maps ™ aerial image in Figure 6.3 shows the testing area with the four latitude and longitude endpoints annotated. To validate the data calculated from the WSN, two different driving scenarios were tested. Each of these scenarios were intended to validated various aspects of the WSN data to ensure the calculations and actual driving lined up correctly. The two driving scenarios chosen for testing were straight line segment (with a downhill slope section incorporated between two RSUs) and a curved segment (with an uphill slope section incorporated between two RSUs). The goal of these tests were for the driver to mainly rely on the classified curvature and slope data (provided by the WSN data) to traverse from one RSU to another.

To visualize the wide array of data being received and sent from the rover to the PC the author developed a Graphical User Interface (GUI) using the National Instruments LabVIEW program. Figure 6.4 shows an annotated screenshot of the developed LabVIEW interface. A detailed breakdown of all the segmented boxes (labeled A through H) is provided to clarify what each section of the GUI represented as displayed.

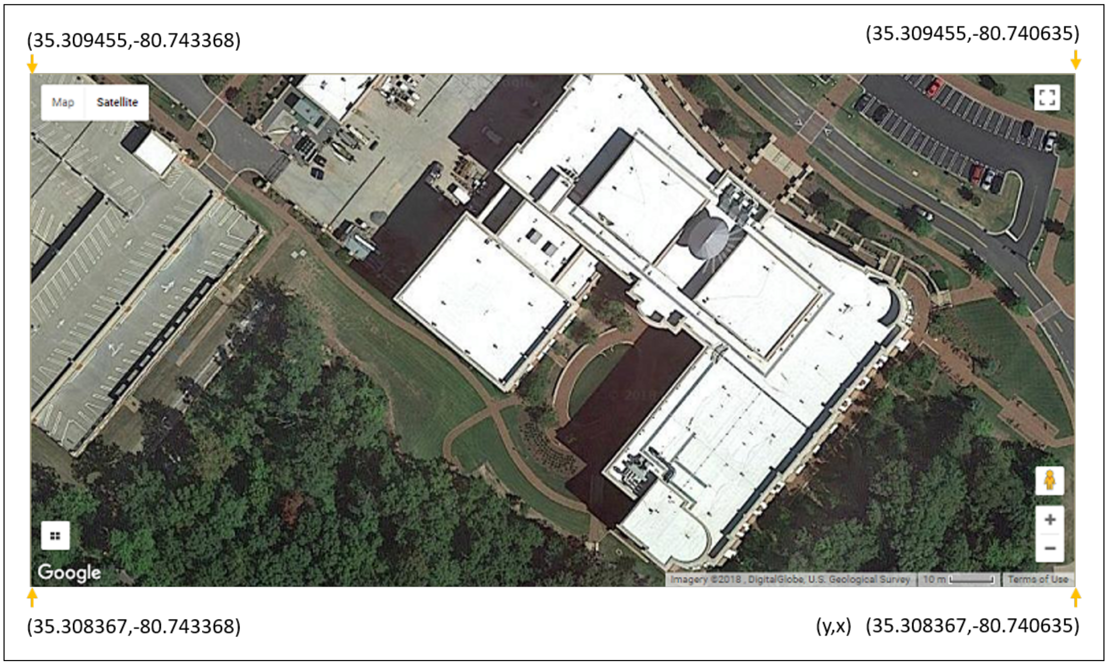


Figure 6.3: 2D Google Map ™ of Testing Area (Endpoints Annotated)

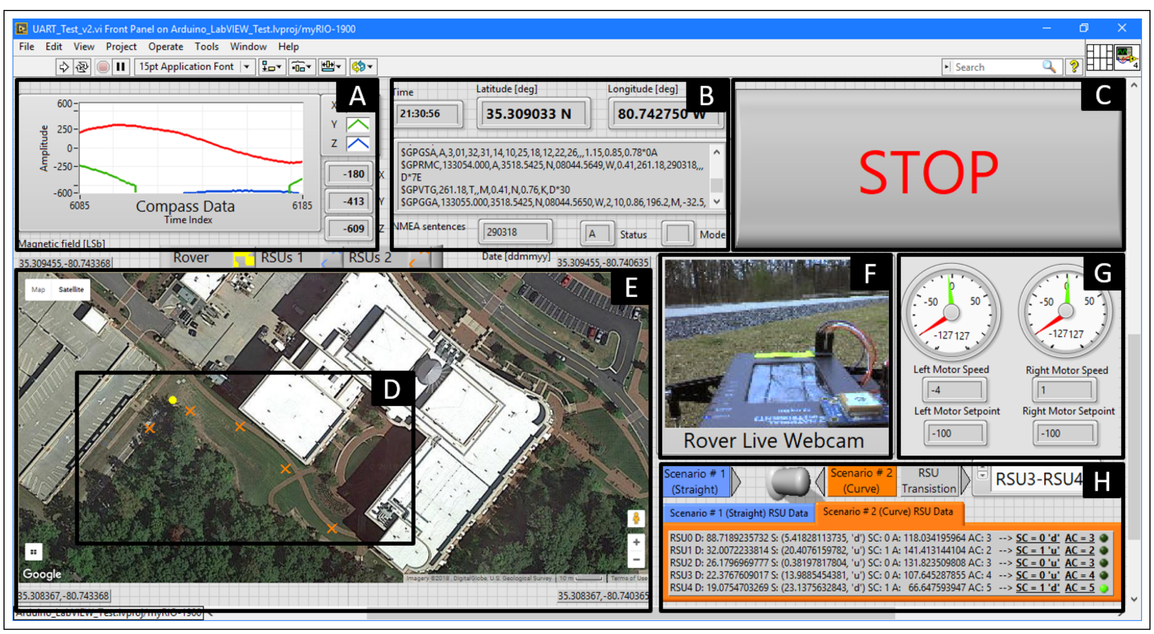


Figure 6.4: GUI Interface Breakdown

Box A. contains the plot to visualize the received data from the on-board compass (located on the top of the rover)

Box B. contains the on-board rover GPS data (Longitude, Latitude, and Altitude)

Box C. contains the stop button used to terminate the program

Box D. contains the RSU locations denoted by a blue (scenario # 1) or orange (scenario # 2) x's. This box also contains the real-time rover GPS position (denoted by a round yellow dot)

Box E. contains the Google Map ™ overlay for testing area behind the EPIC building

Box F. contains a live view of the rover as it traverses the intended path of travel (via a USB webcam)

Box G. contains the current right and left motor speeds (based on the game controller's pair of analog joysticks) are denoted by the green needle and the set points (analogous to the expected value displayed on the Heads-Up Display unit) for both the left and right motors (based on the WSN calculations) are denoted by the red needle

Box H. contains the WSN data (i.e. curvature and slope classification) and the RSU transition indicator

The set points (shown in Box G of Figure 6.4), illustrates the calcuated set points for the rover to aim for each section of it's intended path of travel. These set points are based on the WSN calculations and roadway topology classification (i.e. curvature and slope classes). An example of the topology data, provided by the WSN, is shown in Figure 6.5. Note that the highlighted transition is for scenario # 1 (straight) RSU1. This line provides data for the slope and angle class in relation from RSU0 to RSU1. For this example the slope class is downhill with a slope of 1 and the angle class was determined to be a level 4 out of 6 (based on the scale in Table 5.3). These set points, to reiterate, are used as a basis to navigate the vehicle on the intended path of travel.

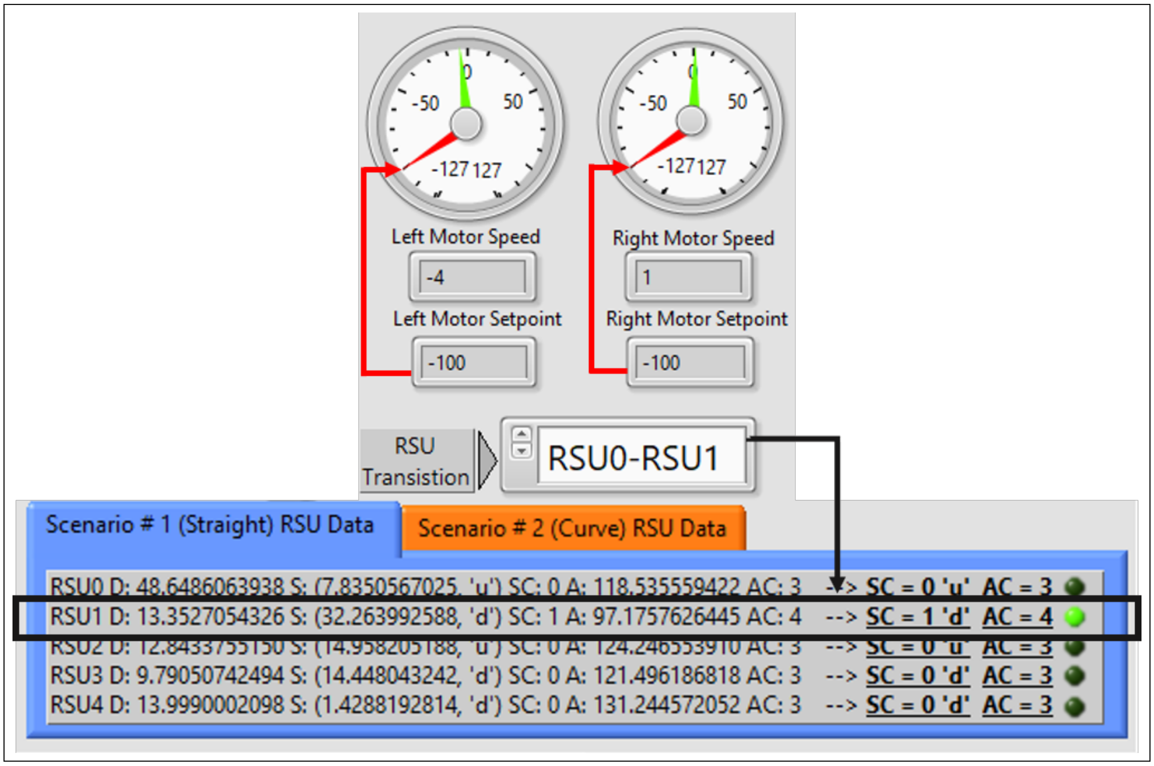


Figure 6.5: Scenario # 1 RSU0 to RSU1 Transition Set Point Example

The following provides detailed information about both of the aforementioned driving scenarios:

Scenario # 1 involved placing the CBU, and RSUs zero through four, along a relative, evenly spaces, straight path, except RSU0 (located inside the CBU) was intentionally placed at a higher elevation that RSU1. This scenario was intended to test how the set points (i.e. expected values) would handle a down slope event. This scenario also tested what values the set points would need to maintain a relatively straight course of travel. Figure 6.6 shows the annotated locations of RSU0 through RSU4 for Scenario # 1. Note the sensors were left for approximately 30 minutes in their respective locations to ensure the GPS data that was acquired would be as accurate as possible. This time frame was repeated for scenario # 2. The GPS longitude, latitude, and altitude data for all five sensors (RSU0 - RSU4) is shown in Table 6.1. The GPS points were used to calculate and classify the terrain topology (i.e. curvature and slope classification). The results from those calculations are shown in Table 6.2.

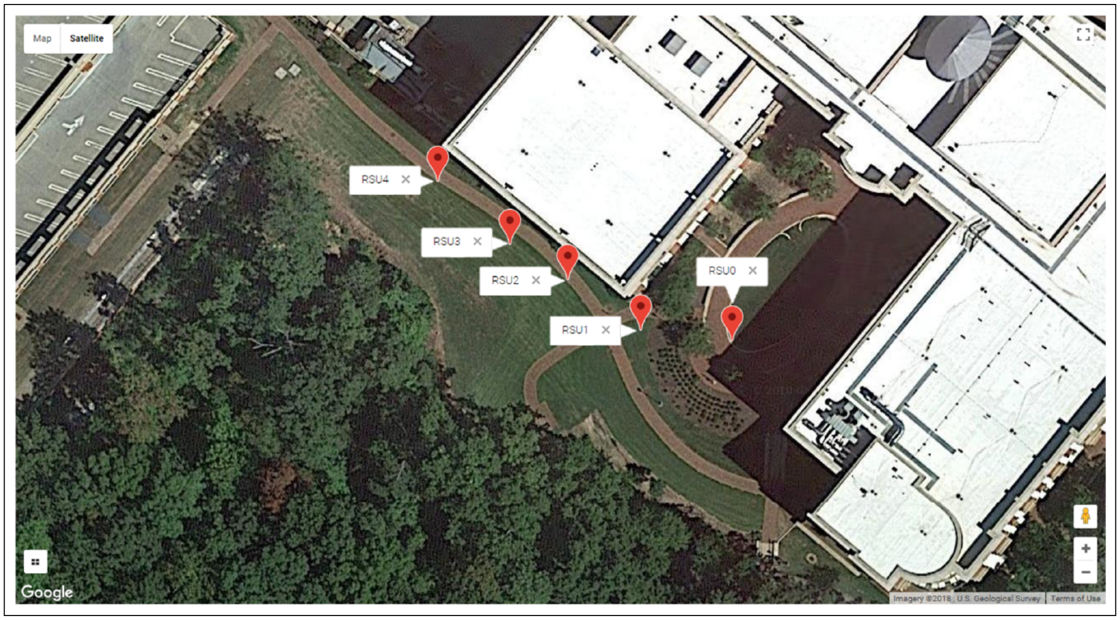


Figure 6.6: 2D Google Map ™ of Scenario # 1 Testing Area (RSUs Annotated)

Table 6.1: RSU GPS Data (Scenario # 1 Straight)

RSU #	**Latitude**	**Longitude**	**Altitude (m)**
RSU0	35.308720	-80.741955	205.10
RSU1	35.308735	-80.742101	201.00
RSU2	35.308800	-80.742218	202.90
RSU3	35.308846	-80.742310	201.50
RSU4	35.308929	-80.742426	201.30

Table 6.2: RSU Calculation Results (Scenario # 1 Straight)

Raw Data Output from Multi-Threaded Python Script from CBU RPi	**Slope Class**	**Angle Class**
RSU0 D: 48.6486063938 S: (7.8350567025, 'u') SC: 0 A: 118.535559422 AC: 3	0 'u'	3
RSU1 D: 13.3527054326 S: (32.263992588, 'd') SC: 1 A: 97.1757626445 AC: 4	1 'd'	4
RSU2 D: 12.8433755150 S: (14.958205188, 'u') SC: 0 A: 124.246553910 AC: 3	0 'u'	3
RSU3 D: 9.79050742494 S: (14.448043242, 'd') SC: 0 A: 121.496186818 AC: 3	0 'd'	3
RSU4 D: 13.9990002098 S: (1.4288192814, 'd') SC: 0 A: 131.244572052 AC: 3	0 'd'	3

Figure 6.7 shows a snapshot of the initial state of the GUI that represents a transition from RSU0 to RSU1 of scenario # 1. Recall that this transition equates to a downhill slope condition based on the down slope class of 1 calculated by the WSN. Based on the WSN data the setpoints for both the left and right motor were set to -100 / -127 noting that having both motors in the negative translates to forward movement of the rover.

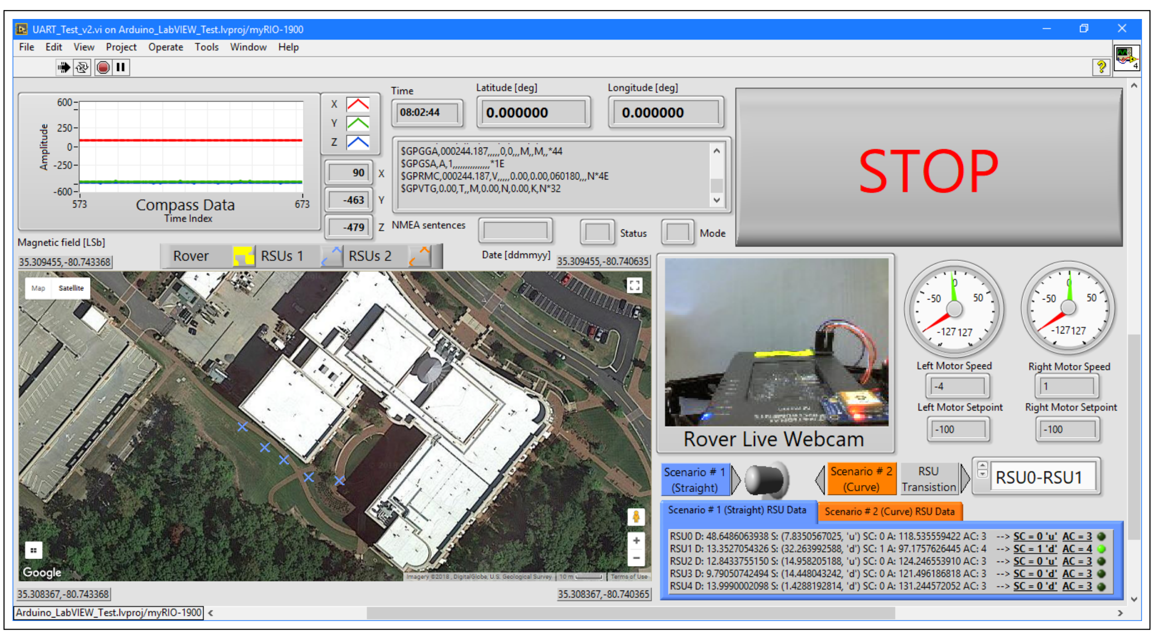


Figure 6.7: Scenario # 1 Curve Test GUI RSU0 to RSU1 Transition

The other transitions, in this scenario, mimicked straight driving conditions and thus a constant slope of and zero and a consistent curve / angle classification of three is reflected in the calculated RSU data.

Scenario # 2 involved placing the CBU, and RSUs zero through four, along a curved path with RSU2 - RSU4 placed in such a manner than RSU3 represented the apex of the curve and RSUs two and four represented the trailing points. Note that RSU0 (located inside the CBU) was intentionally placed at a lower elevation that RSU1. This scenario was intended to test how the set points (i.e. expected values) would handle an up slope event. The image in Figure 6.8 shows the annotated locations of RSU0 through RSU4 for Scenario # 2. The GPS longitude, latitude, and altitude data for all five sensors (RSU0 - RSU4) is shown in Table 6.3. The results from the CBU calculations are shown in Table 6.4.

Figure 6.8: 2D Google Map ™ of Scenario # 2 Testing Area (RSUs Annotated)

Table 6.3: RSU GPS Data (Scenario # 2 Curve)

RSU #	Latitude	Longitude	Altitude (m)
RSU0	35.308547	-80.741991	194.80
RSU1	35.308772	-80.742211	201.20
RSU2	35.308929	-80.742426	201.30
RSU3	35.308990	-80.742661	204.40
RSU4	35.308922	-80.742854	200.10

Table 6.4: RSU Calculation Results (Scenario # 2 Curve)

Raw Data Output from Multi-Threaded Python Script from CBU RPi	Slope Class	Angle Class
RSU0 D: 88.7189235732 S: (5.41828113735, 'd') SC: 0 A: 118.034195964 AC: 3	0 'd'	3
RSU1 D: 32.0072233814 S: (20.4076159782, 'u') SC: 1 A: 141.413144104 AC: 2	1 'u'	2
RSU2 D: 26.1796969777 S: (0.38197817804, 'u') SC: 0 A: 131.823509808 AC: 3	0 'u'	3
RSU3 D: 22.3767609017 S: (13.9885454381, 'u') SC: 0 A: 107.645287855 AC: 4	0 'u'	4
RSU4 D: 19.0754703269 S: (23.1375632843, 'd') SC: 1 A: 66.647593947 AC: 5	1 'd'	5

Figure 6.9 shows a snapshot of the initial state of the GUI that represents a transition from RSU0 to RSU1 for scenario # 2. Recall that this transition equates to an uphill slope condition based on the up slope class of 1 calculated by the WSN. Based on the WSN data the setpoints for both the left and right motor were set to -127 / -127 noting that having both motors in the negative translates to forward movement of the rover.

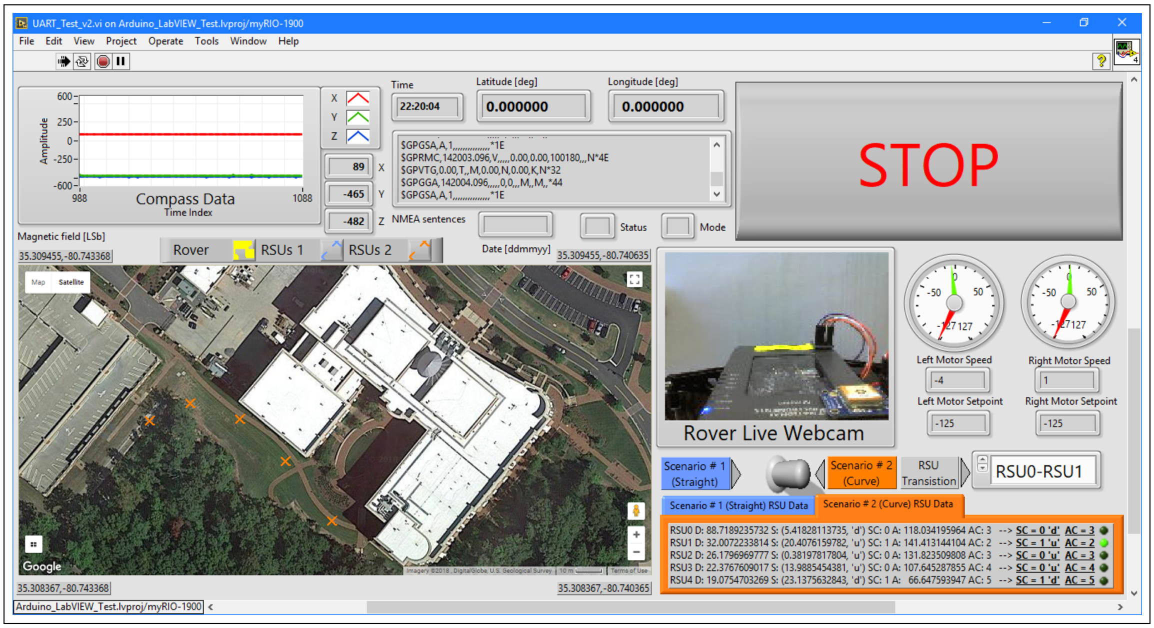


Figure 6.9: Scenario # 2 Curve Test GUI (RSU0 to RSU1 Transition)

It is important to note that the angle classification (via the Haversine "c"parameter) works best for slightly longer distances than those that were used in the smaller scale test setup for the rover. To re-iterate this fact, the data from the preliminary test with this system was performed on a larger scale more fitting for a full size vehicle versus the rover. Figure 6.10 shows the annotated locations of RSU0 through RSU0. The GPS longitude, latitude, and altitude data for all four sensors (RSU0 - RSU3) is shown in Table 6.5. The results from the CBU calculations are shown in Table 6.4. The angle classification of this preliminary scenario shows curve classes more in line with the actual curvature of the roadway. Ideally these sensors would be placed in this manner as the goal of this WSN is to map out long stretches of highway given enough sensor density. The data provided by this method of roadway topology characterization provides a baseline from which vehicle dynamics can be taken into account and more accurate driving algorithms can be developed to aid in the deployment of an intelligent transportation system.

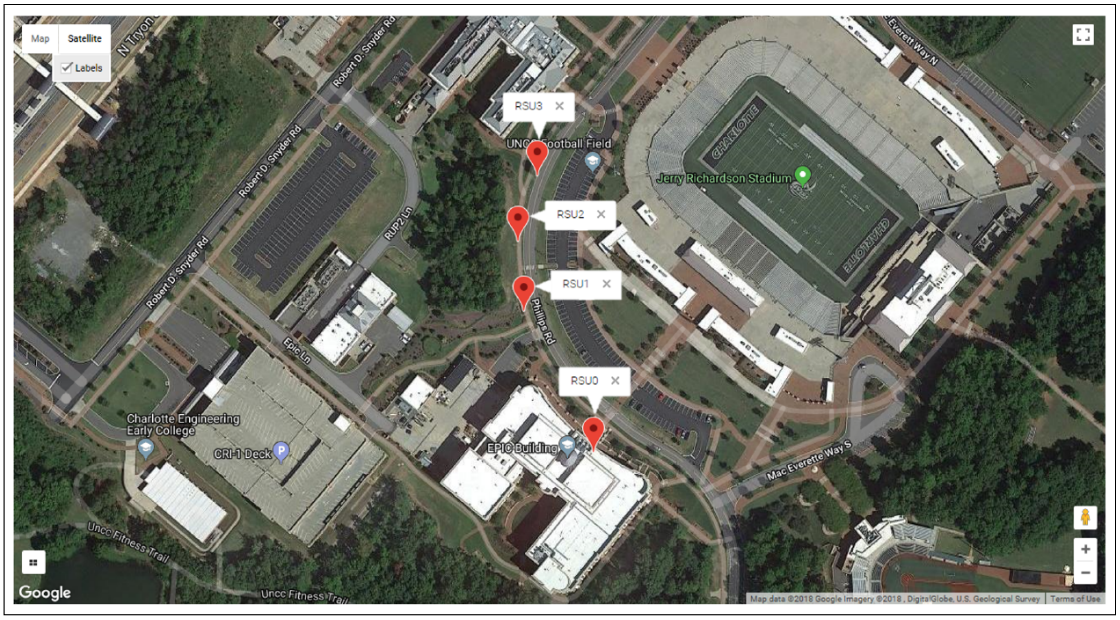


Figure 6.10: 2D Google Map ™ of Preliminary Testing Scenario (RSUs Annotated)

Table 6.5: RSU GPS Data (Preliminary Scenario)

RSU #	Latitude	Longitude	Altitude (m)
RSU0	35.309223	-80.741348	208.00
RSU1	35.309708	-80.741737	198.70
RSU2	35.310226	-80.741989	199.80
RSU3	35.310722	-80.741707	199.60

Table 6.6: RSU Calculation Results (Preliminary Scenario)

Raw Data Output from Multi-Threaded Python Script from CBU RPi	**Slope Class**	**Angle Class**
RSU0 D: 168.932342002 S: (9.51401601542888, ’u’) SC: 0 A: 169.132672555 AC: 1	0 ’u’	1
RSU1 D: 62.6918163173 S: (25.516300085410553, ’d’) SC: 1 A: 145.0286886 AC: 2	1 ’d’	2
RSU2 D: 60.1136307686 S: (3.662189018798813, ’u’) SC: 0 A: 166.647181611 AC: 1	0 ’u’	1
RSU3 D: 58.8516683889 S: (4.592641197364964, ’d’) SC: 0 A: 17.7749582003 AC: 0	0 ’d’	0

CHAPTER 7: CONCLUSION & FUTURE WORK

This research has shown multiple novel hardware and software solutions that provide a framework for integrating legacy vehicles into an intelligent transportation system. The lack of a centralized way to classify vehicles lead the author to develop a method of vehicle classification that would enable further research to take these differing vehicle dynamics into account when working with various new driving algorithms/ schemes. The framework also addresses the issue of how to gather data from legacy vehicles that do not contain the standardized OBDII port via an innovated smartphone and Bluetooth adapter solution. This enables even the oldest vehicles on the roadway to communicate basic telemetry data to a centralized system. The novel framework developed also addressed the lack of roadway characterization. 802.11P, the working group and standard for to be used for V2V and V2I communication, however, the deployment of these systems have been slow and thus a novel XBee based WSN was developed by the author to enable faster deployment and roadway topology classification. Currently, all the major mapping providers do not provide any or limited roadway topology data to the end user. The framework address this issue by developing a multi-threaded python based script that takes the data collected from the wireless sensor network GPS data and performs various calculations on that data to achieve a slope and curvature classification for a given stretch of highway. All of this data provides a complete vehicle and roadway picture that included all vehicles on the road versus only the smart enabled ones that currently only make up a minuscule part of the overall vehicle market. The following list highlights the goals of this research and how each goal was met and what unique hardware or software solution were developed to address these goals.

Vehicle Characterization / Classification: Many classification methods exist, however, there is no collective agreement on how to classify all roadway vehicles. The author chose length as a hard parameter while vehicle weight and height was used a soft

- Heads-Up Display (HUD)Unit to visualize all vehicle data and setpoints
- CAN Bus Explorer (CBX) used to translate CAN bus messages from the vehicle (1996 or newer) to the HUD

Vehicle Telemetry Data: To have a cohesive ITS all vehicles (including legacy ones) need to be able to transmit at least basic telemetry data to a centralized system. This enables all vehicles on the roadway to know the relative position and speed of all the surrounding vehicles on the roadway.

- Legacy Vehicle Telemetry Transmitter (LVTT) pairs with smartphone app to send basic telemetry data to HUD. Also can communicate with WSN

Roadway Curvature and Slope Characterization / Classification: Topology data is not provided to the end user by the major mapping companies. The author developed a XBee based WSN that could be rapidly deployed (as opposed to the slow deployment of 802.11P compliant devices). The WSN used GPS data gathered from the various RSUs and calculated / characterized that given stretch of roadway via a slope and curve classification method.

- Controller Base Unit (CBU) contained RSU0 and a RPi 2 running a multi-threaded python script to visualize all RSU data via Google Maps ™ API
- Road Side Unit (RSU) housed a GPS, long range XBee, microcontroller used to gather and transmit GPS data to the CBU

Vehicle-to-Infrastructure Communication: The RPi 2 is able to connect to the internet via Wi-Fi, thus allowing the topology and vehicle classification data to be housed in a centralized repository.

7.1 Applications

As mentioned in section 5.3, a sensorless approach (i.e. no RSUs) could be a feasible approach to implement an ITS framework such as the one discussed in this dissertation, however, there are many applications in which having static RSUs positioned on the roadway could be beneficial and be more applicable than a senor-less approach. Some arguments and examples where this would be the case follow.

Miltitary Application / Temporary Roadways: Although Section 5.3 concludes that alternatives exists to gathering roadway topology what is not addressed, when pursuing a sensorless framework, is accounting for temporary roadways or areas that have not been properly mapped by any major mapping project. This would include temporary roads setup due to extreme weather events or even military type applications. In these situations a CBU and RSU based framework would be able to take the methods discussed in this dissertation and determine the temporary roadway's topology. This data could then be relayed to the military or off-road type vehicles so they could traverse the new uncharted roadway.

Weather Data Collection / Vehicle Dynamics: One of the best arguments in favor for having physical RSUs distributed along the highway would be to enable those RSUs to act as weather gathering and reporting devices. The changing weather conditions of a roadway would dramatically change the calculus for the vehicle setpoints addressed in this research. For example a track of roadway that is experiencing wet conditions due to rainfall would need to reduce the safe driving speed and distance observed by all approaching vehicles. The RSUs could also supplement readily available weather data as a means of a more granular data set as compared to a general forecast of rainy conditions. The packet structure in Figure 7.1 shows a modified expected value packet structure that could be used to account for hazardous weather conditions. For example, a value of 1 could indicate a sunny day, while a value of 9 could indicate an extreme snow event.

Expected Value (LVTT) XBee Data Packet Format (Weather Scenario):

#L3:A,Brake=0*A,Throttle=0*A,Speed=0**W**#

Below is a detailed breakdown of each segment of the packet:

Packet:	#	L	3	**:**	E	**,**	Brake	=	0	*	E	**,**	Throttle	=	0	*	E	**,**	Speed	=	0	**W**	#
Element	0	1	2	3	4	5	6-10	11	12	13	14	15	16-23	24	25	26	27	28	29-33	34	35	**36**	37
Position	1	2	3	4	5	6	7-11	12	13	14	15	16	17-24	25	26	27	28	29	30-34	35	36	**37**	38

Where:

#	: Start and end packet delimiter
L	: in position 2 signals the message source is from LVTT Module
3	: in position 3 indicates the message contains 3 vehicle (data) elements
:	: in position 4 is a header and data separator
E	: in positions 5, 15, and 28 indicate that the parameter is an expected value (vs actual)
,	: in positions 6, 16, and 29 are data separators
Brake	: in positions 7-11 is a parameter name
=	: in positions 12, 25, and 35 are data separators
0	: in positions 13, 26, and 36 hold the 1 BYTE allocated for the Brake, Throttle, and Steering position data (expected / calculated)
*	: in positions 14 and 27 are data separators
Throttle	: in positions 17-24 is a parameter name
Speed	: in positions 30-34 is a parameter name
W	: in position 37 is a reserved weather characterization bit

Figure 7.1: Expected Value (LVTT) XBee Packet Frame Breakdown

Vehicle / Roadway Condition Relaying: Another major argument in favor of deploying RSUs along a given stretch of highway is to relay traffic and or roadway safety conditions. For example, if a vehicle is broken down in the middle of a blind curve the RSUs could determine the speed of the disabled vehicle was zero. This data could be related to approaching vehicles to indicate that they need to slow down or avoid this stretch of highway all together. This type of roadway to vehicle collaboration would increase safety and avoid delays on a particular route.

7.2 Future Work

With the increasing number of older vehicles on the roadway a system similar to the proposed framework implementation will need to be implemented in order for all vehicles on the roadway to communicate with one another. This framework implementation could one of the many applications aided by the up and coming fifth generation (5G) cellular network as a means of mass distribution of the WSN. The preliminary throughput values for the 5G networks appear to be able to handle the low latency and high bandwidth that V2I and V2V data streams will need moving forward.

The frameworks could also be ported to the 802.11P standard once the widespread distribution of these systems are in place. The need for a method to classify all vehicles and roadway topology will still exist even when these technologies are available. Further research on integrating these methods into the 802.11P system could be addressed. The author has published multiple publications as a result from this research [108–110]. The author plans to continue to pursue research in the area of legacy vehicle integration and ITS framework development and distribution.

REFERENCES

[1] A. Goldsmith and G. Cleven, "Highway Electronic Systems Today and Tomorrow," *IEEE Trans. Veh. Technol.*, vol. 19, pp. 161–167, Feb 1970.

[2] U. S. Department of Transportation, "Average age of automobiles and trucks in operation in the united states — bureau of transportation statistics." http://www.rita.dot.gov/bts/sites/rita.dot.gov.bts/files/publications/national_transportation_statistics/html/table_01_26.html_mfd, April 2016.

[3] F. Lambert, "Tesla missed guidance with only 14,820 vehicles delivered in q1 2016, blames model x supplier parts shortages — electrek." https://electrek.co/2016/04/04/tesla-deliveries-q1-2016/, April 2016. (Accessed on 04/01/2016).

[4] "Gm sales numbers, figures, results — gm authority." http://gmauthority.com/blog/gm/general-motors-sales-numbers/, April 2016. (Accessed on 03/02/2016).

[5] USDOT, "History of Intelligent Transportation Systems." http://www.its.dot.gov/history/, May 2016. (Accessed on 11/02/2016).

[6] "Cohda wireless — products — hardware — mk5." http://cohdawireless.com/Products/Hardware.aspx. (Accessed on 10/11/2016).

[7] "Cohda wireless — products — hardware." http://cohdawireless.com/Products/Hardware.aspx. (Accessed on 10/11/2016).

[8] F. C. Commission, "Fcc allocates spectrum 5.9 ghz range for intelligent transportation systems uses." https://transition.fcc.gov/Bureaus/Engineering_Technology/News_Releases/1999/nret9006.html, October 1999. (Accessed on 10/06/2016).

[9] P. Sahoo, M.-J. Chiang, and S.-L. Wu, "SVANET: A Smart Vehicular Ad Hoc Network for Efficient Data Transmission with Wireless Sensors," *Sensors*, vol. 14, pp. 22230–22260, Nov 2014.

[10] D. M. Weigle, "Standards: WAVE / DSRC/ 802.11P." http://www.cvt-project.ir/Admin/Files/eventAttachments/109.pdf, 2008. (Accessed on 11/03/2016).

[11] S. A. M. Ahmed, S. H. S. Ariffin, and N. Fisal, "Overview of Wireless Access in Vehicular Environment (WAVE) Protocols and Standards," *Indian Journal of Science and Technology*, vol. 6, no. 7, 2013.

[12] WiGLE, "WiGLE: Wireless Network Mapping." https://wigle.net/, Jan 2017. (Accessed on 01/10/2017).

[13] F. C. Commission, "FCC: Wireless Services: Dedicated Short Range Communications (DSRC) Service : DSRC Home." `http://wireless.fcc.gov/services/index.htm?job=service_home&id=dedicated_src`, Mar 2007. (Accessed on 01/09/2017).

[14] V. Grigoryev, I. Khvorov, Y. Raspaev, and E. Grigoreva, "Intelligent transportation systems: techno-economic comparison of dedicated UHF, DSRC, Wi-Fi and LTE access networks: Case study of St. Petersburg, Russia," in *2015 Conf. Telecommun. Media Internet Techno-Economics*, pp. 1–8, IEEE, Nov 2015.

[15] O. Kaiwartya, A. H. Abdullah, Y. Cao, A. Altameem, M. Prasad, C.-T. Lin, and X. Liu, "Internet of Vehicles: Motivation, Layered Architecture, Network Model, Challenges, and Future Aspects," *IEEE Access*, vol. 4, pp. 5356–5373, 2016.

[16] J. Jansons and A. Barancevs, "Using wireless networking for vehicular environment: IEEE 802.11a standard performance," in *2012 Second Int. Conf. Digit. Inf. Process. Commun.*, pp. 5–9, IEEE, Jul 2012.

[17] C.-M. Chou, C.-Y. Li, W.-M. Chien, and K.-c. Lan, "A Feasibility Study on Vehicle-to-Infrastructure Communication: WiFi vs. WiMAX," in *2009 Tenth Int. Conf. Mob. Data Manag. Syst. Serv. Middlew.*, pp. 397–398, IEEE, 2009.

[18] H. Viittala, S. Soderi, J. Saloranta, M. Hamalainen, and J. Iinatti, "An Experimental Evaluation of WiFi-Based Vehicle-to-Vehicle (V2V) Communication in a Tunnel," in *2013 IEEE 77th Veh. Technol. Conf. (VTC Spring)*, pp. 1–5, IEEE, Jun 2013.

[19] K.-C. Su, H.-M. Wu, W.-L. Chang, and Y.-H. Chou, "Vehicle-to-Vehicle Communication System through Wi-Fi Network Using Android Smartphone," in *2012 Int. Conf. Connect. Veh. Expo*, pp. 191–196, IEEE, Dec 2012.

[20] S. Jeong, Y. Baek, and S. H. Son, "A Hybrid V2X System for Safety-Critical Applications in VANET," in *2016 IEEE 4th Int. Conf. Cyber-Physical Syst. Networks, Appl.*, pp. 13–18, IEEE, Oct 2016.

[21] H. Johan, Wahlström ; Isaac, Skog ; Peter, "IMU alignment for smartphone-based automotive navigation," in *Inf. Fusion (Fusion), 2015 18th Int. Conf.*, IEEE, 2015.

[22] J. Wahlstrom, I. Skog, P. Handel, and A. Nehorai, "IMU-based Smartphone-to-Vehicle Positioning," *IEEE Trans. Intell. Veh.*, pp. 1–1, 2016.

[23] W. Lam, "Smartphones See Accelerated Rise to Dominance - IHS Technology." `https://technology.ihs.com/409728/smartphones-see-accelerated-rise-to-dominance`, Aug 2012. (Accessed on 01/10/2017).

[24] AutoSky, "ELM327 Bluetooth OBD2 CAN-BUS Scanner Tool Work with Android." `http://www.obd2tool.com/goods-4503-ELM327+Bluetooth+OBD2+CAN-BUS+Scanner+Tool++Work+with+Android.html`, Jan 2017. (Accessed on 01/10/2017).

[25] X. Ge, H. Cheng, G. Mao, Y. Yang, and S. Tu, "Vehicular Communications for 5G Cooperative Small-Cell Networks," *IEEE Trans. Veh. Technol.*, vol. 65, pp. 7882–7894, Oct 2016.

[26] X. Cui, A. Gulliver, J. Li, and H. Zhang, "Vehicle Positioning Using 5G Millimeter-Wave Systems," *IEEE Access*, pp. 1–1, 2016.

[27] J. A. del Peral-Rosado, J. A. Lopez-Salcedo, Sunwoo Kim, and G. Seco-Granados, "Feasibility study of 5G-based localization for assisted driving," in *2016 Int. Conf. Localization GNSS*, pp. 1–6, IEEE, Jun 2016.

[28] Z. Kljaic, P. Skorput, and N. Amin, "The Challenge of Cellular Cooperative ITS services based on 5G Communications Technology," in *2016 39th Int. Conv. Inf. Commun. Technol. Electron. Microelectron.*, pp. 587–594, IEEE, May 2016.

[29] A. Nshimiyimana, D. Agrawal, and W. Arif, "Comprehensive survey of V2V communication for 4G mobile and wireless technology," in *2016 Int. Conf. Wirel. Commun. Signal Process. Netw.*, pp. 1722–1726, IEEE, Mar 2016.

[30] G. Araniti, C. Campolo, M. Condoluci, A. Iera, A. Molinaro, J. Ha, C.-J. Liang, K. JeongGil, C. Park, M. Ryu, K. Cho, K. Hafeez, L. Zhao, Z. Liao, B. Ma, J. Misic, G. Badawy, S. Rashwand, V. Misic, G. Araniti, C. Campolo, M. Condoluci, A. Iera, A. Molinaro, H. Kim, D. Kang, J. Lee, T. Chung, K. Trichias, A. Vinel, P. Gil, C. Gil, J. Gil, M. Ambrosin, A. Bujari, M. Conti, F. D. Gaspari, C. Palazzi, S. Tornell, C. Calafate, J.-C. Cano, P. Manzoni, M. Fogue, F. Martinez, Q. Zhao, Y. Zhu, C. Chen, H. Zhu, L. Bo, S. Busanelli, F. Rebecchi, M. Picone, N. Iotti, G. Ferrari, K. Katsaros, M. Dianati, L. Le, G. Karagiannis, O. Altintas, E. Ekici, G. Heijenk, B. Jarupan, K. Lin, T. Weil, D. Krajzewicz, J. Erdmann, M. Behrisch, and L. Bieker, "LTE for vehicular networking: a survey," *IEEE Commun. Mag.*, vol. 51, pp. 148–157, May 2013.

[31] A. Dardanelli, F. Maggi, M. Tanelli, S. Zanero, S. M. Savaresi, R. Kochanek, and T. Holz, "A Security Layer for Smartphone-to-Vehicle Communication Over Bluetooth," *IEEE Embed. Syst. Lett.*, vol. 5, pp. 34–37, Sep 2013.

[32] Abitha M.W and Shyni S. M, "AES algorithm for Smartphone to vehicle communication," in *2016 Int. Conf. Comput. Power, Energy Inf. Commuincation*, pp. 796–799, IEEE, Apr 2016.

[33] V. Manolopoulos, P. Papadimitratos, S. Tao, and A. Rusu, "Securing smartphone based ITS," in *2011 11th Int. Conf. ITS Telecommun.*, pp. 201–206, IEEE, aug 2011.

[34] V. P. Barcelos, T. C. Amarante, C. D. Drury, and L. H. A. Correia, "Vehicle monitoring system using IEEE 802.11p device and Android application," in *2014 IEEE Symp. Comput. Commun.*, pp. 1–7, IEEE, Jun 2014.

[35] R. Murugesh, U. Ramanadhan, N. Vasudevan, A. Devassy, D. Krishnaswamy, and A. Ramachandran, "Smartphone based driver assistance system for coordinated lane change," in *2015 Int. Conf. Connect. Veh. Expo*, pp. 385–386, IEEE, Oct 2015.

[36] A. Smirnov, A. Kashevnik, I. Lashkov, O. Baraniuc, and V. Parfenov, "Smartphone-based identification of dangerous driving situations: Algorithms and implementation," in *2016 18th Conf. Open Innov. Assoc. Semin. Inf. Secur. Prot. Inf. Technol.*, pp. 306–313, IEEE, Apr 2016.

[37] M. H. Alsibai and H. M. Siang, "A smart driver monitoring system using android application and embedded system," in *2015 IEEE Int. Conf. Control Syst. Comput. Eng.*, pp. 242–247, IEEE, Nov 2015.

[38] S. M. Tornell, C. T. Calafate, J.-C. Cano, P. Manzoni, M. Fogue, and F. J. Martinez, "Evaluating the Feasibility of Using Smartphones for ITS Safety Applications," in *2013 IEEE 77th Veh. Technol. Conf. (VTC Spring)*, pp. 1–5, IEEE, Jun 2013.

[39] V. Manzoni, A. Corti, C. Spelta, and S. M. Savaresi, "A driver-to-infrastructure interaction system for motorcycles based on smartphone," in *13th Int. IEEE Conf. Intell. Transp. Syst.*, pp. 1442–1447, IEEE, Sep 2010.

[40] Chika Sugimoto, Yasuhisa Nakamura, and Takuya Hashimoto, "Prototype of pedestrian-to-vehicle communication system for the prevention of pedestrian accidents using both 3G wireless and WLAN communication," in *2008 3rd Int. Symp. Wirel. Pervasive Comput.*, pp. 764–767, IEEE, May 2008.

[41] M. Bagheri, M. Siekkinen, and J. K. Nurminen, "Cellular-based vehicle to pedestrian (V2P) adaptive communication for collision avoidance," in *2014 Int. Conf. Connect. Veh. Expo*, pp. 450–456, IEEE, Nov 2014.

[42] A. Durga Ganesh Reddy and B. Ramkumar, "Simulation studies on ZigBee network for in-vehicle wireless communications," in *2014 Int. Conf. Comput. Commun. Informatics*, pp. 1–6, IEEE, Jan 2014.

[43] P. Eamsomboon, P. Keeratiwintakorn, and C. Mitrpant, "The performance of Wi-Fi and Zigbee networks for inter-vehicle communication in Bangkok metropolitan area," in *2008 8th Int. Conf. ITS Telecommun.*, pp. 408–411, IEEE, Oct 2008.

[44] N. Karyemsetty, B. Samatha, and K. H. Rao, "Design and deployment of Vehicle Tracking System in VANETs using Xbee Pro: Prototype model," in *2015 Int. Conf. Commun. Networks*, pp. 97–100, IEEE, Nov 2015.

[45] S. S. Dorle, D. M. Deshpande, A. G. Keskar, and M. Chakole, "Vehicle Classification and Communication Using Zigbee Protocol," in *2010 3rd Int. Conf. Emerg. Trends Eng. Technol.*, pp. 106–109, IEEE, Nov 2010.

[46] P. Anuradha and R. Sendhilkumar, "Design and implementation of Zigbee-RFID based vehicle tracking," in *Int. Conf. Sustain. Energy Intell. Syst. (SEISCON 2011)*, pp. 689–694, IET, 2011.

[47] L. Sha and F. Tao, "Design of vehicle detection system based on ZigBee and AMR," in *2013 2nd Int. Symp. Instrum. Meas. Sens. Netw. Autom.*, pp. 223–226, IEEE, Dec 2013.

[48] Li Lirong, Zhang Yunmei, Lv Shanshan, Chen Ye, Liu Tianqi, and Xu Xinyuan, "Design of intelligent infrared vehicle detect system based on ZigBee," in *2015 Chinese Autom. Congr.*, pp. 1845–1849, IEEE, Nov 2015.

[49] H. Fan, "The Vehicle Positioning Algorithm of the ZigBee Technology," in *2016 Int. Symp. Comput. Consum. Control*, pp. 911–914, IEEE, Jul 2016.

[50] S. D. Dissanayake, P. P. C. R. Karunasekara, D. D. Lakmanaarachchi, A. J. D. Rathnayaka, and A. T. L. K. Samarasinghe, "Zigbee Wireless Vehicular Identification and Authentication System," in *2008 4th Int. Conf. Inf. Autom. Sustain.*, pp. 257–260, IEEE, Dec 2008.

[51] X. Yang, C. Lu, and W. Pan, "A ZigBee-Based Highway Vehicles Prevent Collision Warning System Research," in *2011 Fourth Int. Symp. Comput. Intell. Des.*, pp. 270–273, IEEE, Oct 2011.

[52] W.-C. Hsiao, M.-F. Horng, Y.-J. Tsai, T.-Y. Chen, and B.-Y. Liao, "A Driving Behavior Detection Based on a Zigbee Network for Moving Vehicles," in *2012 Conf. Technol. Appl. Artif. Intell.*, pp. 91–96, IEEE, Nov 2012.

[53] R. de Francisco, L. Huang, and G. Dolmans, "Coexistence of ZigBee wireless sensor networks and Bluetooth inside a vehicle," in *2009 IEEE 20th Int. Symp. Pers. Indoor Mob. Radio Commun.*, pp. 2700–2704, IEEE, Sep 2009.

[54] G. Angel and A. Brindha, "Real-time monitoring of GPS-tracking multifunctional vehicle path control and data acquisition based on ZigBee multi-hop mesh network," in *2011 Int. Conf. Recent Adv. Electr. Electron. Control Eng.*, pp. 398–400, IEEE, Dec 2011.

[55] Wikipedia, "Fly-by-wire - wikipedia, the free encyclopedia." `https://en.wikipedia.org/wiki/Fly-by-wire#History`, September 2013. (Accessed on 10/10/2016).

[56] BUZZLE, "An easy-to-understand guide on how does a power steering work." `http://www.buzzle.com/articles/how-does-power-steering-work.html`. (Accessed on 10/10/2016).

[57] K. C. Colwell, “Electric feel: Nissan digitizes steering, but the wheel remains - feature - car and driver.” http://www.caranddriver.com/features/electric-feel-nissan-digitizes-steering-but-the-wheel-remains-feature, January 2013. (Accessed on 10/10/2016).

[58] R. Schwarz, R. Isermann, J. Bohm, J. Nell, and P. Rieth, “Modeling and Control of an Electromechanical Disk Brake,” SAE International, February 1998.

[59] W.-D. Jonner, H. Winner, L. Dreilich, and E. Schunck, “Electrohydraulic Brake System - The First Approach to Brake-By-Wire Technology,” SAE International, February 1996.

[60] R. Schwarz, R. Isermann, J. Bohm, J. Nell, and P. Rieth, “Clamping Force Estimation for a Brake-by-Wire Actuator,” SAE International, March 1999.

[61] J. Scobie, M. Maiolani, and M. Jordan, “A Cost Efficient Fault Tolerant Brake By Wire Architecture,” SAE International, March 2000.

[62] N. A. Kelling and W. Heck, “The BRAKE Project - Centralized Versus Distributed Redundancy for Brake-by-Wire Systems,” SAE International, March 2002.

[63] N. A. Kelling and P. Leteinturier, “X-by-Wire: Opportunities, Challenges and Trends,” SAE International, March 2003.

[64] Wikipedia, “Brake-by-wire - wikipedia, the free encyclopedia.” https://en.wikipedia.org/wiki/Brake-by-wire, August 2016. (Accessed on 10/10/2016).

[65] C. Lv, J. Zhang, Y. Li, and Y. Yuan, “Regenerative Braking Control Algorithm for an Electrified Vehicle Equipped with a By-Wire Brake System,” vol. 1, April 2014.

[66] BrakeByWire, “Brake by wire advantages.” http://www.brakebywire.com/brake-by-wire-advantages.html. (Accessed on 10/10/2016).

[67] PicoAuto, “Electronic throttle control (drive by wire).” https://www.picoauto.com/library/training/electronic-throttle-control-drive-by-wire-or-fly-by-wire. (Accessed on 10/09/2016).

[68] AJxtcman, “Pennock’s fiero forum - drive by wire (by ajxtcman).” http://www.fiero.nl/cgi-bin/fiero/showThread.cgi?forum=Archive-000001&thread=20090219-2-080795&style=printable. (Accessed on 10/10/2016).

[69] NHTSA, “U.s. department of transportation releases policy on automated vehicle development — national highway traffic safety administration (nhtsa).” http://www.nhtsa.gov/About+NHTSA/Press+Releases/U.S.+Department+of+Transportation+Releases+Policy+on+Automated+Vehicle+Development, May 2013. (Accessed on 10/07/2016).

[70] B. Rhoades and J. Conrad, "A survey of alternate methods and implementations of an intelligent transportation system," in *Conf. Proc. - IEEE SOUTHEASTCON*, 2017.

[71] U. DOT, "Utah dot summit." http://www.planning.utah.gov/Index_files/PDFs/summit10.4.10c.pdf. (Accessed on 10/11/2016).

[72] Wikipedia, "Grade (slope) - wikipedia, the free encyclopedia." https://en.wikipedia.org/wiki/Grade_(slope), October 2016. (Accessed on 10/11/2016).

[73] U. Marines, "Engineer reconnaissance." http://www.marines.mil/Portals/59/Publications/MCWP%203-17.4.pdf?ver=2016-05-04-131618-220, March 2016. (Accessed on 10/11/2016).

[74] Kullabs, "Geometric design of highway (elements of horizontal alignment) — kullabs.com." https://www.kullabs.com/class-engineering/civil-engineering/transportation-engineering/geometric-design-of-highway/geometric-design-of-highway-elements-of-horizontal-alignment. (Accessed on 10/11/2016).

[75] A. Yamsani, "Gradeability for Automobiles," *IOSR Journal of Mechanical and Civil Engineering (IOSR-JMCE*, vol. 11, no. 2, pp. 35–41, 2014.

[76] U. D. of Energy, "Types of vehicles by weight class us department of energy." http://www.afdc.energy.gov/data/widgets/10381. (Accessed on 10/07/2016).

[77] NHTSA, "Information for new manufacturers registered importers." http://www.nhtsa.gov/cars/rules/maninfo/, March 2014. (Accessed on 10/07/2016).

[78] FHWA, "Traffic recorder instruction manual: Classifying vehicles." http://onlinemanuals.txdot.gov/txdotmanuals/tri/classifying_vehicles.htm, February 2012. (Accessed on 10/07/2016).

[79] K. Mucevski, "Automotive can bus system explained — kiril mucevski." https://www.linkedin.com/pulse/automotive-can-bus-system-explained-kiril-mucevski, December 2015. (Accessed on 10/07/2016).

[80] Wikipedia, "Can bus - wikipedia, the free encyclopedia." https://en.wikipedia.org/wiki/CAN_bus, September 2016. (Accessed on 09/30/2016).

[81] E. P. Agency, "obd-faq.pdf." https://www3.epa.gov/otaq/consumer/obd-faq.pdf, May 1997. (Accessed on 10/11/2016).

[82] Sparkfun, "Getting started with obd-ii - learn.sparkfun.com." https://learn.sparkfun.com/tutorials/getting-started-with-obd-ii. (Accessed on 09/30/2016).

[83] Wikipedia, "Obd-ii pids - wikipedia, the free encyclopedia." https://en.wikipedia.org/wiki/OBD-II_PIDs, August 2016. (Accessed on 10/11/2016).

[84] Sparkfun, "Getting started with obd-ii - learn.sparkfun.com." https://learn.sparkfun.com/tutorials/getting-started-with-obd-ii. (Accessed on 10/11/2016).

[85] Wikipedia, "On-board diagnostics - wikipedia, the free encyclopedia." https://en.wikipedia.org/wiki/On-board_diagnostics, September 2016. (Accessed on 09/30/2016).

[86] N. M. E. Association, "Nmea." https://www.nmea.org/content/nmea_standards/nmea_0183_v_410.asp, 2016. (Accessed on 10/11/2016).

[87] ralf hochhausen, "rhotronic blog — nmea protocol." http://www.ralf-hochhausen.de/?tag=nmea-protokoll, March 2012. (Accessed on 09/29/2016).

[88] G. Baddeley, "Gps - nmea sentence information." http://aprs.gids.nl/nmea/. (Accessed on 10/12/2016).

[89] P. Babington, *ZigBee Wireless Sensor and Control Network*. http://www.mypearsonstore.com/bookstore/zigbee-wireless-sensor-and-control-network-0132650576: The name of the publisher, 1 ed., 10 2009.

[90] gravitech, "Xbee 802.15.4 (series 1) 1mw point-to-multipoint rf module with chip antenna." http://www.gravitech.us/xb80s163porf.html. (Accessed on 09/30/2016).

[91] Libelium, "Meshlium not receiving zigbee frames." https://www.libelium.com/forum/viewtopic.php?f=4&t=13341&sid=e40e4aa824353dc42ec84bd302353138, December 2013. (Accessed on 09/30/2016).

[92] ChicagoLandToysandHobby, "6 high banked curve 2/30 deg - chicagoland toys and hobbies." https://chicagolandtoysandhobbies.com/collections/1-32nd-track/products/6-high-banked-curve-2-30-deg. (Accessed on 05/04/2017).

[93] T. Harrington, D. Bowman, W. Johnson, R. Benner, R. Capener, and H. Han, "Vehicle tag used for transmitting vehicle telemetry data," Dec. 9 2004. US Patent App. 10/855,871.

[94] Kersen, "Xr7 elm327 bluetooth obdii scan tool for honda / toyota / ford rover/ audi / opel." http://www.kersen-shop.com/xr7-elm327-bluetooth-obdii-scan-tool-for-honda-toyota-ford-rover-audi-opel-p-5722.html, 2015. (Accessed on 09/30/2016).

[95] tigerheli, "Byo wheel & pedals - wiring." http://tigerheli.mameworld.info/encoder/pedals/gunpowder/wiring.htm. (Accessed on 09/30/2016).

[96] C. D. Jesus, "Uber launching its self-driving cars this month." http://futurism.com/uber-launching-its-self-driving-cars-this-month/, August 2016. (Accessed on 10/06/2016).

[97] B. Howard, "Toyota and audi showcase self-driving cars at ces no sign of google via extremetech." http://www.extremetech.com/extreme/145035-toyota-audi-showcase-self-driving-cars-at-ces-no-sign-of-google, January 2013. (Accessed on 09/30/2016).

[98] Google, "Elevation profile - earth help." https://support.google.com/earth/answer/181393?hl=en, 2016. (Accessed on 10/06/2016).

[99] F. Lardinois, "Google maps adds elevation profiles to bike routes to help you avoid those steep hills — techcrunch." https://techcrunch.com/2014/05/16/google-maps-adds-elevation-profiles-to-bike-routes-to-help-you-avoid-those-steep-hills/, May 2014. (Accessed on 11/10/2017).

[100] GlobalSat, "Globalsat em406a gps module manual.pdf." https://www.sparkfun.com/datasheets/GPS/EM-406A_User_Manual.PDF. (Accessed on 09/29/2016).

[101] A. Upadhyay, "Haversine formula - calculate geographic distance on earth." http://www.igismap.com/haversine-formula-calculate-geographic-distance-earth/, 2016. (Accessed on 10/03/2016).

[102] R. Sinnott, "Virtues of the haversine," *skytel*, vol. 68, p. 158, Dec. 1984.

[103] USGS, "Elevation point query service." https://nationalmap.gov/epqs/. (Accessed on 04/10/2018).

[104] Google, "Get started google maps elevation api." https://developers.google.com/maps/documentation/elevation/start?csw=1. (Accessed on 04/10/2018).

[105] O. Esrafilian and D. Gesbert, "3D City Map Reconstruction from UAV-Based Radio Measurements," in *GLOBECOM 2017 - 2017 IEEE Glob. Commun. Conf.*, pp. 1–6, IEEE, dec 2017.

[106] T. Vardoulaki, M. Vakalopoulou, and K. Karantzalos, "Per city-block, density estimation at build-up areas from aerial RGB imagery with deep learning," in *2017 Jt. Urban Remote Sens. Event*, pp. 1–4, IEEE, mar 2017.

[107] A. Agrawal, M. Matsumura, A. Nakazawa, and H. Takemura, "Large-scale 3D scene modeling by registration of laser range data with Google Maps images," in *2009 16th IEEE Int. Conf. Image Process.*, pp. 589–592, IEEE, nov 2009.

[108] B. Rhoades, V. Katariya, and J. Conrad, “A Novel RF (XBee) and IR LoS (Line-of-Sight) Collaborative Vehicle-to-Vehicle Navigation Technique,” in *Conf. Proc. - IEEE SOUTHEASTCON*, 2018.

[109] B. Rhoades and J. Conrad, “A Novel Terrain Topology Classification and Navigation for an Autonomous CAN Based All-Terrain Vehicle,” in *Conf. Proc. - IEEE SOUTHEASTCON*, 2018.

[110] B. Rhoades, D. Srivastava, and J. Conrad, “Design and Development of a ROS Enabled CAN Based All-Terrain Vehicle,” in *Conf. Proc. - IEEE SOUTHEASTCON*, 2018.