

Task 8 Deliverable - Final Report
Contract BDV24-977-30

Using Smartphone as On-board unit (OBU) Emulator Implementation Study

Mohamed A. Abdel-Aty, Ph.D., P.E.

Qing Cai, Ph.D.

Shaurya Agarwal, Ph.D.

Zubayer Islam

Pei Li

Shile Zhang

Dhrubo Hasan

Jiheng Huang

University of Central Florida

Department of Civil, Environmental & Construction Engineering

Orlando, FL 32816-2450



UNIVERSITY OF
CENTRAL FLORIDA

February 2020

Disclaimer

The opinions, findings, and conclusions expressed in this publication are those of the authors and not necessarily those of the State of Florida Department of Transportation.

UNITS CONVERSION

APPROXIMATE CONVERSIONS TO SI UNITS

SYMBOL	WHEN YOU KNOW	MULTIPLY BY	TO FIND	SYMBOL
LENGTH				
in	inches	25.4	millimeters	mm
ft	feet	0.305	meters	m
yd	yards	0.914	meters	m
mi	miles	1.61	kilometers	km
SYMBOL	WHEN YOU KNOW	MULTIPLY BY	TO FIND	SYMBOL
AREA				
in²	squareinches	645.2	square millimeters	mm ²
ft²	squarefeet	0.093	square meters	m ²
yd²	square yard	0.836	square meters	m ²
ac	acres	0.405	hectares	ha
mi²	square miles	2.59	square kilometers	km ²
SYMBOL	WHEN YOU KNOW	MULTIPLY BY	TO FIND	SYMBOL
VOLUME				
fl oz	fluid ounces	29.57	milliliters	mL
gal	gallons	3.785	liters	L
ft³	cubic feet	0.028	cubic meters	m ³
yd³	cubic yards	0.765	cubic meters	m ³
NOTE: volumes greater than 1000 L shall be shown in m ³				
SYMBOL	WHEN YOU KNOW	MULTIPLY BY	TO FIND	SYMBOL
MASS				
oz	ounces	28.35	grams	g
lb	pounds	0.454	kilograms	kg
T	short tons (2000 lb)	0.907	megagrams (or "metric ton")	Mg (or "t")
SYMBOL	WHEN YOU KNOW	MULTIPLY BY	TO FIND	SYMBOL
TEMPERATURE (exact degrees)				
°F	Fahrenheit	5 (F-32)/9 or (F-32)/1.8	Celsius	°C
SYMBOL	WHEN YOU KNOW	MULTIPLY BY	TO FIND	SYMBOL
ILLUMINATION				
fc	foot-candles	10.76	lux	lx
fl	foot-Lamberts	3.426	candela/m ²	cd/m ²
SYMBOL	WHEN YOU KNOW	MULTIPLY BY	TO FIND	SYMBOL
FORCE and PRESSURE or STRESS				
lbf	poundforce	4.45	newtons	N
lbf/in²	poundforce per square inch	6.89	kilopascals	kPa

SYMBOL	WHEN YOU KNOW	MULTIPLY BY	TO FIND	SYMBOL
LENGTH				
mm	millimeters	0.039	inches	in
m	meters	3.28	feet	ft
m	meters	1.09	yards	yd
km	kilometers	0.621	miles	mi
SYMBOL	WHEN YOU KNOW	MULTIPLY BY	TO FIND	SYMBOL
AREA				
mm ²	square millimeters	0.0016	square inches	in ²
m ²	square meters	10.764	square feet	ft ²
m ²	square meters	1.195	square yards	yd ²
ha	hectares	2.47	acres	ac
km ²	square kilometers	0.386	square miles	mi ²
SYMBOL	WHEN YOU KNOW	MULTIPLY BY	TO FIND	SYMBOL
VOLUME				
mL	milliliters	0.034	fluid ounces	fl oz
L	liters	0.264	gallons	gal
m ³	cubic meters	35.314	cubic feet	ft ³
m ³	cubic meters	1.307	cubic yards	yd ³
SYMBOL	WHEN YOU KNOW	MULTIPLY BY	TO FIND	SYMBOL
MASS				
g	grams	0.035	ounces	oz
kg	kilograms	2.202	pounds	lb
Mg (or "t")	megagrams (or "metric ton")	1.103	short tons (2000 lb)	T
SYMBOL	WHEN YOU KNOW	MULTIPLY BY	TO FIND	SYMBOL
TEMPERATURE (exact degrees)				
°C	Celsius	1.8C+32	Fahrenheit	°F
SYMBOL	WHEN YOU KNOW	MULTIPLY BY	TO FIND	SYMBOL
ILLUMINATION				
lx	lux	0.0929	foot-candles	fc
cd/m ²	candela/m ²	0.2919	foot-Lamberts	fl
SYMBOL	WHEN YOU KNOW	MULTIPLY BY	TO FIND	SYMBOL
FORCE and PRESSURE or STRESS				
N	newtons	0.225	poundforce	lbf
kPa	kilopascals	0.145	poundforce per square inch	lbf/in ²

1. Report No.	2. Government Accession No.	3. Recipient's Catalog No.	
4. Title and Subtitle Using Smartphone as On-board unit (OBU) Emulator Implementation Study		5. Report Date February, 2020	6. Performing Organization Code
7. Author(s) Mohamed A. Abdel-Aty, Ph.D., PE; Qing Cai, Ph.D.; Shaurya Agarwal, Ph.D.; Zubayer Islam; Pei Li; Shile Zhang; Dhruvo Hasan; Jiheng Huang			
9. Performing Organization Name and Address Department of Civil, Environmental & Construction Engineering University of Central Florida 12800 Pegasus Drive, Suite 211 Orlando, FL 32816-2450		10. Work Unit No. (TRAIS)	11. Contract or Grant No. BDV24-977-30
12. Sponsoring Agency Name and Address Florida Department of Transportation 605 Suwannee Street, Tallahassee, FL 32399		13. Type of Report and Period Covered Final Deliverable, 10/2018-02/2020	14. Sponsoring Agency Code
15. Supplementary Note			
16. Abstract <p>This project aimed at using smartphones to emulate the On-Board Unit (OBU) for Infrastructure-to-Vehicle (I2V) and Pedestrian-to-Vehicle (P2V) applications. In order to achieve the objectives, the following tasks were performed. The research team first reviewed existing studies and practices to fully understand smartphone sensors, roadside sensors, data communication technologies, and OBU technologies and applications. Smartphone apps were developed to collect smartphone sensor data, and the cloud server was set up to allow the apps to upload and receive data. The feasibility of using smartphones as OBU emulators was validated by examining the data communication latency and smartphone battery consumption. The system was programmed to collect the data of smartphone sensors, such as Global Positioning System (GPS), Accelerometer, Gyroscope, and Magnetometer, into the cloud server. Different methods and algorithms were developed and programmed to obtain users' traffic parameters, including position, surrounding environment, speed, acceleration, transportation mode, and turning movement. The accuracy of the users' traffic parameters was evaluated by conducting various experiments, which suggested that the data could well reflect users' statuses. In addition to smartphone data, the camera was used as a type of external sensor to provide additional information. The computer vision technology was used to detect pedestrians even if they did not have the developed apps. The camera was further applied to detect queue length at intersections in real time. The accuracy of computer vision technologies was tested and validated. A communication system was set up to upload the camera data to the cloud server with low latency. Based on the communication system between smartphone apps and the cloud server, the research team used smartphones to emulate the OBUs for I2V applications. Two applications were included: (1) curve warning; (2) queue warning. The warning logic was proposed under different scenarios and programmed into the system. Besides, we further used smartphones as OBU emulators for P2V warning. The P2V warning considered two conditions: (1) both drivers and pedestrians have the developed smartphone apps; (2) only drivers have the developed smartphone apps and pedestrians were detected by cameras. Different warning logics were proposed at intersections and segments. Different drivers' turning movements, including driving through, turning left, and turning right, were incorporated into the system. Extensive experiments were conducted to validate the developed OBU emulator system. The results suggested that we successfully used smartphones as OBU emulators for I2V and P2V applications. It is intended that the developed system could expand the benefits of OBUs by leveraging smartphones, which have a high market penetration.</p>			
17. Key Word Smartphone, On-board unit (OBU) emulator, I2V and P2V applications, Cloud server, Computer vision		18. Distribution Statement	
19. Security Classif. (of this report)	20. Security Classif. (of this page) Unclassified	21. No. of Pages 236	22. Price

EXECUTIVE SUMMARY

The On-Board Unit (OBU) is integrated into a vehicle for interaction with drivers by displaying warnings, issuing alerts, offering automotive services, and managing the communication with a vehicle's surroundings. With these functions, OBU could help alleviate the misjudgment of the driver, avoid traffic crashes, and lay out the efficient driving route. However, only a fraction of drivers on roads can benefit from OBU implementation because OBU is expensive and currently only available in some luxury car models. The research team intends to use smartphones to emulate the OBUs for Infrastructure-to-Vehicle (I2V) and Pedestrian-to-Vehicle (P2V) applications. To achieve the goal, we developed a system based on the cloud server and smartphone and conducted extensive experiments to validate the developed system.

First, the research team conducted a comprehensive literature review of the existing studies and practices that are relevant to the smartphone sensors, data communication, roadside sensors, OBU application, etc. Based on the literature review, the important features of smartphone and OBU applications were identified for the app development of OBU emulators.

Subsequently, the preliminary apps were developed to validate the feasibility of using the smartphone as the OBU emulator. The developed smartphone apps could collect data from smartphone sensors, including GPS, accelerometer, gyroscope, magnetometer, and barometer. The cloud server was set up to realize the communication between the developed apps and the server. The feasibility was validated based on the two measures: (1) whether the latency of data communication could meet the requirement of V2X application; (2) whether battery capacity of smartphones could support to use the developed apps.

The developed apps and server were further improved to enhance data communication and enable the data storage at the server. Multiple methods were proposed to obtain different traffic parameters of smartphone users based on the extensive understanding of the smartphone data. Numerous experiments were conducted to evaluate the obtained traffic parameters. The results suggested that the traffic parameters based on the smartphone data could well reflect users' traffic statuses. Besides, the research team developed a prototype of a real-time pedestrian detection system and vehicular queue detection system with an external complementary sensor. Two emerging sensors, including LiDAR and camera, were evaluated for detection at fixed locations, and the camera was finally selected. The prototype of a real-time detection system was set up. Extensive experiments were conducted to evaluate the proposed system. The results suggested that the detection system could detect pedestrians and queue statuses with a high accuracy. The cloud server was utilized to realize the communication between external video sensor and drivers with the developed apps. Meanwhile, detection results could be sent to the server in real time with a latency less than 200 ms. Hence, the suggested pedestrian detection system was able to proactively detect pedestrians in real time even the pedestrians do not have the developed apps. Meanwhile, the system could successfully identify the queuing status at intersections.

With the input data from smartphone apps and roadside sensors, the research team successfully used smartphone to emulate OBU applications for I2V and P2V warnings. First, the I2V applications, including curve warning and queue warning, were developed. Second, the P2V applications were developed at intersections under two conditions: (1) both the driver and the pedestrian have smartphone apps; (2) the driver has the smartphone app and the pedestrian could be proactively detected by the camera in real time. For each condition, three vehicle movements

(i.e., driving through, turning left, and turning right) were included. In addition, the P2V application were developed at the segment, which could be applied when a pedestrian is difficult for a driver to observe. All I2V and P2V applications were validated by conducting a variety of experiments. The experiment results suggested that the developed system could identify the potential conflicts with high accuracy and send the warning in real time. Through the completed tasks, it could be concluded that the research team successfully developed smartphone apps to emulate the OBU for the I2V and P2V applications.

In summary, this project developed smartphone apps as well as the cloud server and set up the communication between them to emulate OBU applications. The data of smartphone sensors were extensively explored to understand users' statuses such as location, transportation mode, speed, acceleration/deceleration, and turning movement. The camera was used as a complementary sensor to proactively detect pedestrians and queue length at intersections. Based on all efforts, we developed smartphone apps as the OBU emulators for I2V and P2V applications. Various experiments were conducted, and the results validated the developed OBU emulator applications.

TABLE OF CONTENTS

EXECUTIVE SUMMARY	vi
TABLE OF CONTENTS.....	ix
LIST OF FIGURES	xiv
LIST OF TABLES.....	xix
CHAPTER 1.INTRODUCTION	1
CHAPTER 2. REVIEW OF STUDIES ABOUT SMARTPHONE-BASED COMMUNICATION AND OBU APPLICATIONS.....	6
2.1. Smartphone Systems	7
2.1.1. Smartphone Sensors.....	7
2.1.2. Smartphone Energy Consumptions.....	9
2.1.3. Sensors Data Streaming	10
2.1.4. Human Machine Interaction (HMI).....	11
2.1.5. Summary.....	13
2.2. Smartphone-Based Data Analysis.....	14
2.2.1. Smartphone-Based Driver Behavior Detection	14
2.2.2. Methodologies.....	15
2.2.3. Detection Algorithms.....	16
2.2.4. Accuracy of Smartphones.....	18
2.2.5. Battery-Saving Strategies.....	19
2.2.6. Smartphone-Based Transportation Mode Classification	20
2.2.7. Smartphone Application Development for Transportation.....	26
2.2.8. APP Development.....	28

2.2.9.	Summary	30
2.3.	Detection Sensors and Communication Technology	30
2.3.1.	LiDAR Vendors and Products	31
2.3.2.	Detection and Tracking Methods.....	39
2.3.3.	Communication Technology.....	52
2.3.4.	Mobile Cloud Computing	58
2.3.5.	Summary	60
2.4.	OBU Applications	61
2.4.1.	OBU Function Required of Connected Vehicle System Operation.....	64
2.4.2.	OBU Hardware Components of V2V System	66
2.4.3.	Summary	82
2.5.	Conclusions	82
CHAPTER 3. VALIDATION OF THE FEASIBILITY OF USING SMARTPHONE AS OBU		
.....	84
3.1.	Development of Apps for Data Collection.....	85
3.1.1.	Develop Application to Collect Data.....	85
3.1.2.	Data Stream.....	86
3.1.3.	Summary	89
3.2.	Feasibility Test.....	89
3.2.1.	Demonstration of Data Communication	89
3.2.2.	Latency of Data Transmission	91
3.2.3.	Battery Consumption	95
3.2.4.	Summary	99

3.3. Conclusions	100
CHAPTER 4. VALIDATION OF THE DATA ACCURACY AND REFINEMENT OF THE DATA STREAM	101
4.1. Development of Cloud Server and Apps.....	102
4.1.1. Architectures of Cloud Server and Apps	102
4.1.2. Overview of Smartphone Sensor Data.....	106
4.1.3. Summary	108
4.2. Smartphone Data Analysis and Validation	108
4.2.1. Smartphone Position Data Accuracy Validation	108
4.2.2. Validation of Smartphone Speed	114
4.2.3. Localize Users.....	117
4.2.4. Identification of User Travel Modes.....	121
4.2.5. Acceleration Estimation.....	124
4.2.6. Vehicle Movement Detection	130
4.2.7. Summary	138
4.3. CONCLUSIONS.....	139
CHAPTER 5. TEST OF COMPLEMENTARY SENSORS	140
5.1. Developing a Real-Time Pedestrian Detection System	141
5.1.1. The Structure of the P2V Warning System with Complementary Sensors	141
5.1.2. The Real-Time Pedestrian Detection System	143
5.1.3. Evaluation Experiments and Results	151
5.1.4. Summary	156
5.2. Video-Based Queue Detection System	157

5.2.1.	Queue Detection Algorithm.....	157
5.2.2.	Evaluation Experiment and Results.....	159
5.2.3.	Summary.....	160
5.3.	Conclusions.....	160
CHAPTER 6. DEVELOPMENT OF PRELIMINARY SMARTPHONE-BASED APPLICATION.....		12V 162
6.1.	Preparation of Database.....	163
6.2.	Development of Warning Logic.....	166
6.2.1.	Description of Data Transfer between Smartphone and Server.....	166
6.2.2.	Warning Estimation in Server.....	167
6.2.3.	Warning Display on the Smartphone.....	168
6.2.4.	Results and Discussions.....	170
6.3.	Conclusions.....	173
CHAPTER 7. DEVELOPMENT OF PROACTIVE SMARTPHONE-BASED APPLICATION.....		P2V 174
7.1.	Smartphone-Based P2V Warning Logic.....	175
7.1.1.	Smartphone-Based P2V Warning Logic Based on Basemap.....	175
7.1.2.	Smartphone-Based P2V Warning Logic Based on Proximity.....	178
7.2.	Camera-Based P2V Warning Logic.....	179
7.3.	Experiments and Results.....	180
7.3.1.	Warning Display on the Smartphone.....	180
7.3.2.	Experiment Design.....	181
7.3.3.	Experiment Results.....	183

7.4. Conclusions	185
CHAPTER 8. SUMMARY AND CONCLUSIONS	186
8.1. Summary	186
8.2. Conclusions	188
PROJECT SCHEDULE.....	194
REFERENCES	195

LIST OF FIGURES

Figure 1. Performance of smartphone-embedded sensors (Wahlström et al., 2017).....	10
Figure 2. iPhone App for data collection (Su, 2017).....	11
Figure 3. Illustration of accelerometer and gyroscope sensors.....	14
Figure 4. Values of $G\mu S$ in the moving status in motorized and non-motorized modes.....	21
Figure 5. Threshold based on distribution function (Eftekhari & Ghatee, 2016).....	22
Figure 6. Flowchart of the implementation of the prototype (Biljecki et al., 2013).....	23
Figure 7. A typical added feature and comparison with an original one in transportation mode (Fang et al., 2016).....	24
Figure 8. Basic model for the PDR algorithm (Tian et al., 2014).....	25
Figure 9. Detection of unusual traffic status on a road segment (Yoon et al., 2007).....	27
Figure 10. Quanergy LiDAR S3-8.....	32
Figure 11. Quanergy LiDAR M8.....	33
Figure 12. Quanergy LiDAR S3-1.....	34
Figure 13. Qortex interface.....	34
Figure 14. QPU unit.....	35
Figure 15. Transformation of image coordinates to world coordinates.....	40
Figure 16. Block diagram of the vehicle tracking system.....	41
Figure 17. (a) Corner features identified by the tracker, (b) sample features from the tracker, and (c) sample feature groups from the grouper.....	42
Figure 18. OBU unit mounted with multiple antenna and communication connection (Xu et al., 2017).....	57
Figure 19. Proposed cloud-based vehicular network architecture (Yu et al., 2013).....	59

Figure 20. In-Vehicle components of a V2V system (Harding et al., 2014)	67
Figure 21. Illustration of the TRP system with all hardware components (Burt et al., 2014c)	68
Figure 22. WSU example.....	70
Figure 23. Construction of miniWSU1	71
Figure 24. DENSO WSU1.5 with automotive-grade and style connectors.....	72
Figure 25. DSRC antenna 1 (left), DSRC antenna 2 (right)	73
Figure 26. OBU DGPS receiver.....	74
Figure 27. Samsung Galaxy tab (SCH-I957).....	75
Figure 28. Driver-Vehicle interface (DVI).....	76
Figure 29. Illustration of the UMTRI GEN5 DAS (Burt et al., 2014c).....	77
Figure 30. DAS mounted in a cargo compartment in the sleeper cabs (left) and under the passenger seat in the day cab (right).....	79
Figure 31. DAS IMU in sample location between frame rails behind cab on unsprung mass (LeBlanc et al., 2014a).....	80
Figure 32. Relationship between requirements and TRP architecture components (Burt et al., 2014a)	81
Figure 33. OBU system architecture (Stephens et al., 2014).....	82
Figure 34. A framework of smartphone data communication	84
Figure 35. Screen-shot of developed apps	86
Figure 36. Screenshot of data file generated by the smartphone app	88
Figure 37. Visualization of data communication (through Android studio) for Android.....	90
Figure 38. Visualization of data communication for iOS	90
Figure 39. Screen-shot of battery consumption status	96

Figure 40. Battery consumption for Android for different sampling rate.....	98
Figure 41. Battery consumption for different sensors	99
Figure 42. Server architecture.....	102
Figure 43. Mobile application architecture.....	104
Figure 44. Accelerometer of a smartphone.....	106
Figure 45. Accelerometer and gyroscope of smartphone	107
Figure 46. RTK GNSS.....	109
Figure 47. Validating GNSS data using Google map.....	110
Figure 48. GNSS data points vs smartphone data points.....	111
Figure 49. Locations of smartphones and GNSS.....	112
Figure 50. GNSS and smartphone points for the vehicle.....	113
Figure 51. Location for pedestrian speed validation.....	114
Figure 52. Validation of vehicle speed using adaptive cruise control	116
Figure 53. Flowchart for radius neighbor classifier.....	119
Figure 54. Base map points captured by GNSS.....	120
Figure 55. Example data of accelerometer, gyroscope and speed sensors from the smartphone	122
Figure 56. Experiment routes for driving and walking.....	123
Figure 57. Coordinate systems of smartphone and vehicle (Eboli et al., 2016)	124
Figure 58. Euler angles illustration.....	125
Figure 59. Coordinate reorientation.....	126
Figure 60. World's coordinate system.....	126
Figure 61. Bearing illustration (Bhoraskar et al. 2012)	127
Figure 62. Vehicle's trajectory	128

Figure 63. Smartphone’s positions	128
Figure 64. Reorientation results of Y axis	129
Figure 65. Reorientation results of X axis	130
Figure 66. Gyroscope of smartphone and motion of car	131
Figure 67. value of Z axis for left turn and right turn	132
Figure 68. Value of X axis of accelerometer before and after filtering	133
Figure 69. Feature correlation.....	134
Figure 70. Driving trajectory	135
Figure 71. Classification process	136
Figure 72. Classification result	138
Figure 73. A framework of collision warning system	141
Figure 74. Illustration of P2V warning at an intersection.....	142
Figure 75. Quanergy M8 LiDAR.....	144
Figure 76. Setup for the M8 LiDAR.....	145
Figure 77. Screenshot of Qortex interface with LiDAR detection results	146
Figure 78. Screenshot of video from the camera	147
Figure 79. YOLO detection (Redmon et al., 2016)	149
Figure 80. Setup for the camera.....	150
Figure 81. Detection zones and conflict points (spatial).....	152
Figure 82. Experiment locations on map	153
Figure 83. Screenshots of detection results.....	154
Figure 84. Virtual loop detector (spatial).....	158
Figure 85. Screenshot of vehicular queue detection algorithm.....	159

Figure 86. Structure of the database	164
Figure 87. An example of a curve data in the database	166
Figure 88. Client-Server communication.....	167
Figure 89. Screenshot of curve warning display in the smartphone app along with auditory warning message.....	169
Figure 90. Screenshot of queue warning displayed in the smartphone app.....	170
Figure 91. Illustration of curve warning received in car.....	171
Figure 92. Illustration of queue warning.....	172
Figure 95. Illustration of base map and potential conflicts.....	177
Figure 96. Illustration of the experiment on a segment	178
Figure 97. Screenshots of warning display on the driver’s and pedestrian’s smartphones, respectively	181
Figure 98. Experiment locations	182
Figure 99. Experiment results.....	184

LIST OF TABLES

Table 1. List of smartphone sensors with functions (Su, 2017).....	8
Table 2. Characteristics of typical forms of smartphone-related driver distractions (Wahlström et al., 2017)	12
Table 3. Detailed information of driver behavior detection methods.....	18
Table 4. A 30-minute WalkSafe car detection experiments in real world (Wang et al., 2012)....	29
Table 5. Velodyne LiDARs	37
Table 6. LeddarTech LiDARs.....	38
Table 7. Motion segmentation studies	44
Table 8. Object tracking studies.....	46
Table 9. Comparison between different sensor modalities for pedestrian detection (Viola, Jones, & Snow, 2005).....	50
Table 10. Pedestrian detection using LiDAR sensors.....	51
Table 11. Pedestrian detection using multiple sensors	51
Table 12. Comparison of wireless protocols (Dhondge et al., 2014)	53
Table 13. Active safety latency requirements (units: seconds) (Xu et al., 2017)	58
Table 14. Comparative Study of VCC and CC (Whaiduzzaman et al., 2014)	60
Table 15. Aftermarket safety device types (Harding et al., 2014).....	63
Table 16. Attributes of WSU 1.5	72
Table 17. List of sensors used in apps for data collection	87
Table 18. Uploading, downloading and round-trip duration for the walking mode.....	94
Table 19. Uploading, downloading, and round-trip duration for the driving mode	94
Table 20. Specifications of Samsung Note 9 and iPhone XS.....	95

Table 21. Validation results for iPhone	111
Table 22. Results for vehicle localization data validation	113
Table 23. Smartphone walking speed validation results.....	115
Table 24. Vehicle speed validation results	116
Table 25. Results from the localization algorithm for vehicles	121
Table 26. Classification results	137
Table 27. Summary of traffic data computation with smartphone data.....	139
Table 28. List of devices.....	151
Table 29. Diagram for metrics calculation	155
Table 30. Experiment result of detection algorithm	156
Table 31. Experiment result of queue detection algorithm.....	160
Table 32. Distance as a function of speed.....	168
Table 33. Results of I2V warning system.....	171
Table 34. Potential conflict pairs	177
Table 35. Experiment results	183
Table 36 Summary of traffic data computation with smartphone data.....	190
Table 37 Summary of developed applications.....	192

CHAPTER 1. INTRODUCTION

In-vehicle communication has experienced great expansion in the current Intelligent Transportation Systems (ITS). The on-board unit (OBU), which can capture information from multiple sensors and manage large amounts of data at high computation speed, has become a main solution for in-vehicle communication. An OBU is integrated into a vehicle for interaction with drivers by displaying warnings, issuing alerts, offering automotive services, and managing the communication with a vehicle's surroundings. With these functions, an OBU could help alleviate the misjudgment of the driver, prevent traffic crashes, and lay out the efficient driving route. Many ongoing efforts have been conducted to use OBU for the Infrastructure-to-Vehicle (I2V) and Pedestrian-to-Vehicle (P2V) applications. However, only a fraction of the drivers on the road could benefit from OBU implementation because an OBU is expensive and currently only available in some luxury car models. With the development of smartphone technologies, smartphones have many built-in sensors which could reflect user status on the road required by the OBU communication. Hence, it is possible to use smartphones as OBU emulators. Given the fact that the percentage of smartphone ownership of U.S. adults is around 80%, there could be great benefits to use smartphones as OBU emulators for I2V and P2V applications.

To emulate on-board units (OBUs), it is necessary to set up the smartphone data communication. Apps are needed that can collect data of smartphone sensors, and a cloud server must be set up to allow data communication with smartphone apps. It is necessary to test the feasibility of using smartphones as OBU emulators before the start of development. The feasibility should be validated through the following aspects:

- Whether it is feasible move data from the user to cloud computation and back to the user via a smartphone
- Whether the latency of information flow from the smartphone to the cloud computation and vice versa could meet the requirement of I2V and P2V warnings in a variety of circumstances
- Which data should be used to stream information from the apps
- Whether the battery capacity of smartphones could support the use of the developed apps

The smartphones have many built-in sensors including the exteroceptive (acquiring external information such Global Positioning System (GPS), Wi-Fi-based positioning, and magnetometers) and proprioceptive (acquiring internal information such accelerometers and gyroscopes). The GPS and other positioning technologies could provide vehicle position and speed information in real time. Meanwhile, accelerometer and gyroscope could provide three-dimensional measurements of specific force (non-gravitational acceleration) and angular velocity. The sensors could be used to determine other user information such as transportation modes, acceleration or deceleration rates, and left- or right-turn movements. It is worth mentioning that significant improvement of smartphones' GPS accuracy is expected in the near future, while the data uploading/downloading speeds would also increase significantly when 5G becomes available. However, all data sources have limitation in accuracy. Smartphones are highly complex microcomputers with more computing power than many early mainframes, but they still are limited based on the many factors to the accuracy of information that is provided. While validation of individual components has been completed, the ability to combine them to ascertain important transportation information has not been fully studied. Hence, studies should be

conducted to obtain transportation data based on smartphone sensors and validate the accuracy of the data. Building off the findings of the feasibility and the data accuracy validation results, the data stream from the road user to the cloud computation need to be optimized by mode. In addition to using built-in smartphone sensors, other roadside sensors such as camera could be used to provide more information. The sensors could be installed at critical locations such as dangerous segments or high-risk intersections. The sensor data are complementary data, which could help increase the accuracy and detection rate of smartphone data. With the complementary sensor data, road users could be detected proactively, even if they do not have the developed apps.

OBU's have been applied for I2V and P2V warning under various traffic conditions. In this phase, it is not possible to consider all possible conditions. Hence, possible scenarios should be predefined. For the I2V warning, two possible test scenarios could be included, i.e., curve warning and queue warning. When approaching a sharp curve, a driver may lose control of the car and result in a rollover crash. Through the I2V communication, the OBU could warn the driver of the curve and the driver can adjust his/her lane control behavior. On the other hand, the driver could take an early deceleration to avoid a rear-end crash if a queue warning is delivered through the OBU. The potential conflicts between pedestrians and drivers could be at intersections and segments. Vehicles of different movement at intersection could hit pedestrians and jaywalking pedestrians could get hit by vehicles at segments. The OBU could reduce the risk by warning drivers the potential conflicts with pedestrians. These scenarios should be considered in the development of OBU-emulator apps. With the data based on smartphone and roadside sensor, the potential conflicts could be identified based on measures such as distance and surrogate safety measures. Appropriate thresholds to activate the warning message should be carefully determined.

It requires to design a framework to get different road users connected, enabling to emulate OBU for I2V and P2V warning under different scenarios.

In summary, our vision is to use smartphones as OBUs emulator for I2V and P2V applications. Based on the above discussion, the main objectives of this project are summarized as follows:

1. Review studies and practice about smartphone sensor, data communication, roadside sensors, and OBU applications
2. Test the feasibility of using smartphones as OBU emulators
3. Explore smartphone sensor data
4. Test other roadside sensors to proactively detect pedestrians and queue lengths
5. Develop the smartphone-based OBU emulator for I2V warning
6. Develop the smartphone-based OBU emulator for P2V warning

Chapters by each task in this research project are as follows:

- Chapter 1: Introduction
- Chapter 2: Review of studies about smartphone-based communication and OBU applications
- Chapter 3: Validation of using smartphone as OBU
- Chapter 4: Validation of the data accuracy and refine data stream

- Chapter 5: Development of preliminary smartphone-based I2V application
- Chapter 6: Development of complementary sensors for P2V application
- Chapter 7: Development of proactive smartphone-based P2V application
- Chapter 8: Summary and conclusions

CHAPTER 2. REVIEW OF STUDIES ABOUT SMARTPHONE-BASED COMMUNICATION AND OBU APPLICATIONS

Smartphones are becoming more and more popular in recent years, and the percentage of smartphone ownership of U.S. adults have increased to 77% by 2018. Meanwhile, smartphones are equipped with a variety of sensors which can provide valuable information for the deployment of many ITS related applications. With the high penetration rate of smartphones, there could be great benefits to use smartphones as On-Board Unit (OBU) emulators for real-time communication in vehicles (V2V) and between vehicles and pedestrians (V2P). Since smartphones could only provide raw sensor data, it is necessary to develop an application to compile and translate sensor data to reflect users' statuses. Meanwhile, users could receive information through the application about their surrounding traffic conditions.

To ensure the success of the developed application, a comprehensive literature review about the smartphone sensor data, data processing and OBUs, is needed. According to the tasks in the project scope, the literature review needs to cover the following concepts:

- Systems aspects related to sensors, energy consumption, cloud computing, vehicle ad hoc-network, and human-machine interfaces
- Studies and practices about smartphone-based data analysis including transportation mode classification, drivers' movement identification, and road condition monitoring
- Existing smartphone applications related to connected vehicles
- Sensors for detecting road users and communication technologies for the connected vehicle
- OBU applications for the connected vehicle communications

2.1. Smartphone Systems

2.1.1. Smartphone Sensors

Smartphone sensors are one of the most available resources to collect data for various research projects such as traffic safety application. Smartphone sensors can be divided into four categories: motion sensors, environmental sensors, position sensors, and connection sensors (Wahlström et al., 2017).

- i. **Motion Sensors:** The sensors measuring the acceleration forces and rotational forces along with the three axes of the phone's coordination. Sensors include accelerometers, gravity sensors, gyroscopes and rotational vector sensors.
- ii. **Environmental Sensors:** The sensors measuring different attributes of environment, e.g., ambient temperature, air pressure, illumination, humidity etc. Sensors include barometer, thermometers, and photometers.
- iii. **Position Sensors:** These sensors measure the physical position of the device. This category includes orientation sensors, Global Positioning System (GPS) Sensors, and magnetometers.
- iv. **Connection Sensors:** These sensors enable the smartphone to connect and communicate with other devices via various protocols. This category includes Bluetooth, Wireless sensors, and Standard cellular connection modules.

A brief summary of smartphone sensors and their sensing data is shown in Table 1. The most frequently used sensors with their sensing data, dimensions of measurements, and units are listed. There are some sensors that measure only one-dimension data (e.g., barometer, proximity sensor, light sensor, humidity sensors, and temperature sensors) and others measure three-

dimensional data (e.g., accelerometer, gyroscope, magnetometer, gravity sensor and rotation sensor).

Table 1. List of smartphone sensors with functions (Su, 2017)

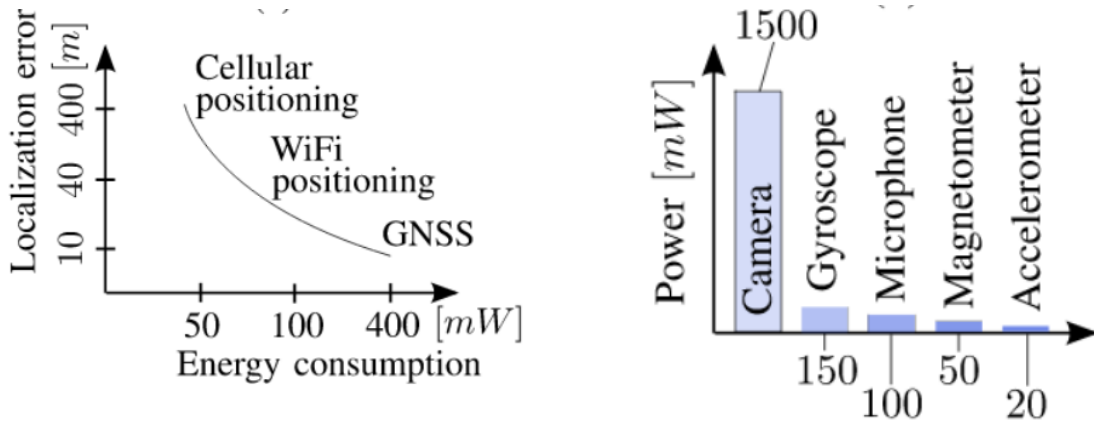
Sensor Name	Data Collected	Dimensions	Unit
Accelerometer	Acceleration	x, y, z	g-force
Gravity Sensor	Gravity	x, y, z	m/s^2
Gyroscope	Rotation Rate	$x, y, z,$ x_calibration, y_calibration, z_calibration	rad/s
Magnetometer	Magnetic Field	$x, y, z,$ x_calibration, y_calibration, z_calibration	μT
Barometer	Ambient Air Pressure	1	hPa
Rotation Sensor	Rotation Degree (y axis pointing to magnetic north as the default)	Azimuth: Rotation around z Axis Pitch: Rotation around x Axis Roll: Rotation around y Axis	Degree
Proximity Sensor	Relative distance from an object to the view screen of a device	1	cm
Light Sensor	the ambient light level	1	lx
Humidity Sensor	the relative ambient humidity	1	Percentage
Temperature Sensor	Ambient temperature	1	Celcius
GPS Sensor	Geographical description of current location and estimated speed	Latitude, Longitude, Speed	Degree, m/s

Wahlström et al (2017) described the smartphone sensors in two categories: exteroceptive (e.g., GPS, Global Navigation Satellite System (GNSS), Wi-Fi positioning, Bluetooth positioning system) and proprioceptive sensors (e.g., accelerometer, and gyroscope). In iPhone devices, there is a built-in black-box feature called Location Services to provide apps with navigation updates. The updates are obtained by fusing Wi-Fi, cellular, Bluetooth, and GNSS data. Wi-Fi and cellular positioning are mainly used to aid in the initialization of the GNSS receiver. In contrast to smartphones from the iPhone series, Android devices enable apps to read GNSS messages in the National Marine Electronics Association (NMEA) 1803 standard. The standard includes not only GNSS measurements of position, planar speed, and planar course, but also additional information such as detailed satellite data.

2.1.2. Smartphone Energy Consumptions

One of the limitations of smartphone-based apps is the battery energy storage capacity of the smartphone. Hence, energy consumption by the technology and hardware in smartphone is a key factor in the design of new services and applications. Both sensing and processing data require a significant level of energy considering the limited energy storage of the smartphone. The selection of sensors and methods to optimize sensing using suitable sampling frequency to leverage energy consumption and system performance is one of the primary concerns in smartphone-based applications. Data processing energy can be reduced using effective and optimized algorithms. In data sensing, Lane et al. (2013) proposed a method called piggyback crowd sensing (PCS). In PCS, sensor data is collected by exploiting smartphone app opportunities when the smartphone users place call or use applications. During the time, phone does not need to wake up from idle state.

An important application of smartphone app is Location-Based Service (LBS). For localization, Cellular positioning, Wi-Fi positioning, and GNSS sensors data are usually used. In Figure 1(a), the localization accuracy versus energy consumption for these sensors are illustrated. Cellular positioning requires minimum energy but maximum localization error, while GNSS requires maximum energy but lowest localization error. Hence, exploiting the application requirement smartphone can dynamically choose the trade-off between accuracy and energy consumption. Energy consumption for the frequently used smartphone embedded sensors is illustrated in Figure 1 (b). It is indicated that energy consumption of the camera is very high compared with other sensors. Hence, it is necessary to avoid or minimize camera data while developing smartphone apps.



(a) Tradeoff between localization error and energy consumptions at 0.1 Hz for cellular positioning, Wi-Fi positioning, and GNSS

(b) Typical figures of energy consumption for smartphone-embedded sensors

Figure 1. Performance of smartphone-embedded sensors (Wahlström et al., 2017)

2.1.3. Sensors Data Streaming

Real-time data collection is one of the major functions of transportation agencies to inform travelers and provide reliable transportation services. Traffic data providers aggregate real-time traffic data from multiple sources, including smartphone, freight fleets, GPS data, etc. To collect sensor data, we need to develop (or use open source) smartphone apps based on iOS or Android system. Su (2017) developed a smartphone app to collect sensor data both online and offline for travel mode classification. Figure 2 shows the screen shot of their app. During the data collection, user can set the sampling rate. Since the sampling rate is directly related to the energy consumption, they set the optimal sampling frequency as default sampling rate. Depending on the sampling rate, the signal (Data) quality can vary but latency of the application remains unchanged. So, for our project, we can use the default frequency of 16 Hz as presented in (Su, 2017). LTE/4G cellular data or Wi-Fi/Bluetooth can be used to stream the data into the server.

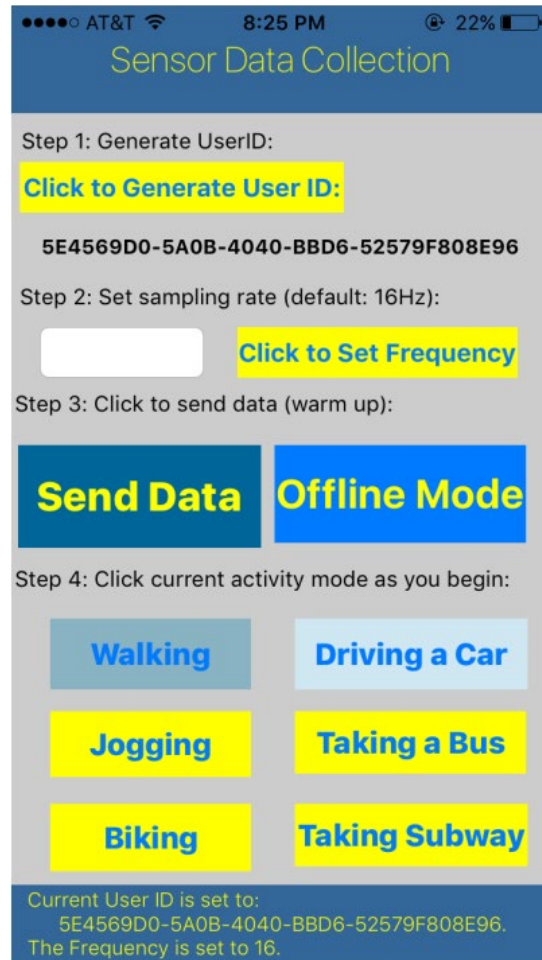


Figure 2. iPhone App for data collection (Su, 2017)

2.1.4. Human Machine Interaction (HMI)

The term HMI encompasses all functionalities that allow a human to interact with a machine. In case of HMI, we are mainly interested in the human-smartphone interface. The design of a human-smartphone interface should include two objectives (Yamabe & Kiyohara, 2014). First, the interface should be efficient so that the task of control and monitoring have minimum computational effort and human interaction. Second, the interface should be configured to minimize driver distraction. Hence, all driver-smartphone communication should be dynamically integrated with the driving operations and discourage excessive engagement in secondary tasks. Driver distraction due to smartphone usage has become one of the leading factors in fatal and

serious injury crashes (Wahlström et al., 2017). Four facets of distracted driving can be identified: (i) cognitive distraction which is taking your mind off the road, (ii) visual distraction which is taking your eyes off the road, (iii) auditory distraction which is focusing your auditory attention on sounds that are unrelated to the traffic environment, and (iv) manual distraction which is taking your hands off the wheel (Yamabe & Kiyohara, 2014). Table 4 describes how the activities of texting, dialing a phone number, having a conversation on the phone, and delivering voice commands relate to different forms of distractions. The activities of texting and dialing are the more distractive activities than phone call and voice control. Hence, human-smartphone interface should be developed in a way to avoid texting and dialing especially while in driving.

Table 2. Characteristics of typical forms of smartphone-related driver distractions (Wahlström et al., 2017)

Activity	Distraction			
	Cognitive	Visual	Auditory	Manual
Texting	High	High	Low	High
Dialing	Medium	High	Low	High
Phone call	High	Low	High	Low
Voice control	High	Low	Medium	Low

Most human-smartphone interfaces are based on visual communication utilizing touch-based operations and virtual keyboards. The smartphone display may be used for augmented reality, i.e., to overlay computer generated graphics onto images of reality. This has been utilized in apps such as iOnRoad (Lendino, 2012), which highlights the vehicle's current lane in a real-time video shown in the smartphone display. Another app is Hudway ("HUDWAY Cast -Safe Driving"), which creates a head-up display (HUD) based on the existing vehicle technology, by reflecting smartphone images in the windshield. The latter technique is particularly valuable in low visibility conditions since it enables the driver to see the curvature of the upcoming road

overlaid in sharp lines on the windshield. By using the Google and Apple standards Android Auto and CarPlay, respectively, any smartphone operation can be directly integrated into the HUD, thereby further reducing visual and manual distractions. However, according to predictions, only 9% of the automobiles on the road in the year 2020 will have a built-in HUD, and consequently, there will continue to be a high demand for inexpensive and flexible aftermarket solutions (Wahlström et al., 2017). While developing smartphone apps for our project, we can make an option to incorporate HUD (if exists) to reduce visual and mental distractions.

2.1.5. Summary

A smartphone usually has many built-in sensors, which could provide valuable data for the connected-vehicle applications. In this project, to validate the feasibility of using smartphone application to emulate the OBU, the following aspects should be considered:

- The capability of moving data from the smartphone apps to cloud computation and back to the smartphone
- The latency of information flow from the smartphone to the cloud computation and vice versa in a variety of circumstances
- Impact of the smartphone application on the battery life
- The distraction caused by the application for the drivers

2.2. Smartphone-Based Data Analysis

2.2.1. Smartphone-Based Driver Behavior Detection

Among smartphone sensors, several of them can provide valuable information for the detection of driver behavior. For example, as shown in Figure 3, accelerometer gives the amount of acceleration applied to the phone in the (x, y, z) planes, If the smartphone is placed on the same plane with the vehicle, the acceleration of y axis can be used to identify acceleration and braking. On the other hand, the acceleration of x axis can be applied to detect left turn and right turn. Gyroscope measures the rate at which a device rotates around a spatial axis, which is also shown in Figure 3. The gyroscope data provide the information regarding lane departure and turning events (Eren, et al., 2012). Besides, magnetometer measures the strength of magnetic field which can provide the direction of smartphone with respect to the magnetic north (Saiprasert et al., 2013). In the end, GPS provides the location of the phone in terms of latitude and longitude.

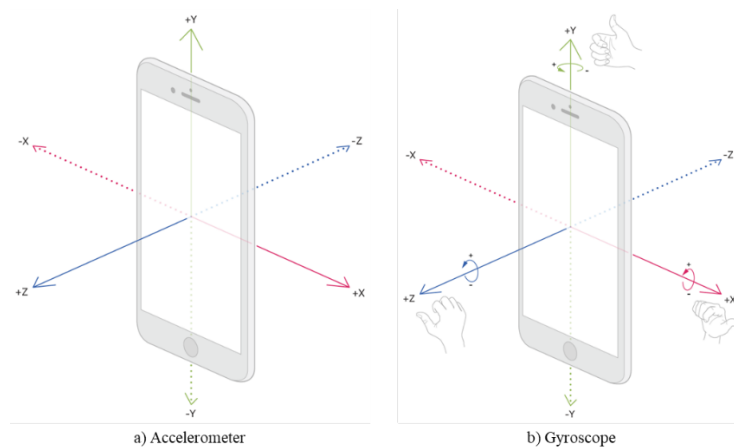


Figure 3. Illustration of accelerometer and gyroscope sensors

2.2.2. Methodologies

Several key factors need to be considered before the detection of driver behavior, which include the choice of sensors, the choice of axes, and the methods of data pre-processing.

(1) Sensor Selection

Two strategies are available for sensor selection, i.e., single sensor and sensor-fusion. Single sensor can provide valuable information for detecting driver behavior. For example, the gyroscope can detect aggressive turning and aggressive braking. The accelerometer can detect aggressive acceleration and aggressive lane changing behaviors (Ferreira et al., 2017).

Sensor-fusion combines different sensor data to detect driver behavior. For example, gyroscope signal is the indication of vehicle turning movement, while we can get a more accurate reading of device altitude(orientation) by using the accelerometer and magnetometer in conjunction with the gyroscope. Johnson and Trivedi (2011) found sensor-fusion strategy could increase the accuracy of detecting U-turn compared to use only either accelerometer or gyroscope data. Besides, GPS and accelerometer data fusion can be used to improve the accuracy of speed estimation (Chowdhury et al., 2014).

(2) Axis Selection

Generally, there are three axes, x, y and z for accelerometer, magnetometer and gyroscope. GPS has two axes, longitude and latitude. However, the x axis values of magnetometer are always 0 or close after translated to Earth's coordinate system. Hence, usually there is no magnetometer_x data set. Besides, more accurate detection results could be obtained by using all sensor axes to detect driver behavior than using a single axis. The only exception is the z axis of the gyroscope which can best detect aggressive left turns solely (Ferreira et al., 2017).

(3) Data Pre-processing Methods

First, the orientation of a smartphone needs to be corrected with the respect of the orientation of the vehicle. Vlahogianni and Barmponakis (2017) developed a dynamically updated reorientation algorithm based on Euler rotation theorem, which can be applied to accelerometer, gyroscope, and GPS sensor data. Similarly, Johnson and Trivedi (2011) also used gyroscope, accelerometer and magnetometer to find the Euler rotation (yaw, pitch, and roll) from a reference attitude.

Second, the raw sensor data suffer from multiple noise effects and are influenced by gravity component (Vlahogianni & Barmponakis, 2017). Allan deviation can be used for estimating and compensating the noise of accelerometer data (Chowdhury et al., 2014). Besides, moving average filter (SMAF), dynamic exponential smoothing filter (DESF), Gaussian filter (GF) and band pass filter (BDF) can also be used as filters to smooth the sensor data (Eren et al., 2012; Singh et al., 2017; Yu et al., 2017).

2.2.3. Detection Algorithms

Generally, driver behavior detection is a classification problem. Several methods can be applied, including rule-based methods and machine learning methods.

(1) Rule-based Methods

For rule-based methods, driver behavior could be classified based on certain values. For example, acceleration, braking and left/right turn, are considered as violations of thresholds imposed on vehicle acceleration of different axes (Paefgen et al., 2012). Moreover, left/right turn can also be detected according to certain constrains on heading, which can be obtained from GPS sensor (Saiprasert et al., 2013). However, Saiprasert et al. (2013) found rule-based method was not accurate enough compared to dynamic time warping (DTW). The main reason may be that rule-

based methods are not flexible as the thresholds are manually set. Thus, Vlahogianni and Barmounakis (2017) provided a better rule-based method, which could find the optimum threshold values according to rough set theory.

(2) Machine Learning Methods

Machine learning methods are also crucial for detecting driver behavior (Lu et al., 2018), which include Random Forest (RF), Support Vector Machines (SVM), Neural Network (NN), K Nearest Neighbor (KNN), Decision Tree (DT), and Bayesian Network (BN).

The DTW algorithm has been widely used in many studies. The main function of this algorithm is to compute the similarity of each event with template event. Eren et al. (2012) applied DTW and Bayesian classification to estimate the probability of safe or unsafe drive, which had higher accuracy than RF and hidden Markov models. Johnson and Trivedi (2011) proposed a DTW-based driver profile algorithm by using smartphone sensors including GPS and camera. The study evaluated the performance of different sensor fusion sets to detect lateral and longitudinal movements. After evaluating over 200 driving events, the authors showed that the sensor fusion set could provide the best classification performance using DTW. The performance of different machine learning algorithms has been compared in several studies. Ferreira et al. (2017) chose Artificial Neural Network (ANN), SVM, RF and BN to compare their performance on the classification of different driver behavior. RF was proved as the best algorithm which had the highest area under the ROC curve (AUC) value, followed by ANN. Similarly, Lu et al. (2018) compared the performance of RF, Naïve Bayes, DT, KNN, and SVM for detecting stopping, going straight, turning left and turning right. Results shown the RF had highest accuracy of 98.95%. Jiadi Yu et al. (2017) applied SVM and NN to detect abnormal drive behavior, including weaving, swerving, fast U-turn, and etc. The accuracy of SVM and NN were 95.36% and 96.88%,

respectively. In the end, all the detailed information of the above methods is summarized in Table 3.

Table 3. Detailed information of driver behavior detection methods

Reference	Sensors	Methods	Driver Behavior	Results
(Paefgen et al., 2012)	Accelerometer, Gyroscope, GPS	Rule-based	Acceleration, Braking, Left/Right Turn	Smartphone detects more events than IMU
(Vlahogianni and Barmponakis, 2017)	Accelerometer, Gyroscope, GPS	Rule-based	Harsh braking, Harsh acceleration, Right hash cornering, Left hash cornering	Precision reached up to 96.7%
(Saiprasert et al., 2013)	Accelerometer, Magnetometer, GPS	Rule-based, DTW	Right/left turn, Right/left lane change, Braking, Acceleration	DTW has higher accuracy than rule-based method
(Eren et al., 2012)	Accelerometer, Gyroscope, Magnetometer	Bayesian Classification, DTW	Safe driving, Unsafe driving	Correctly Classified Instances by 93.3%
(Johnson and Trivedi, 2011)	Accelerometer, Gyroscope, Magnetometer, GPS, Video	DTW	Left/Right turn, U-turn, Acceleration, Braking, Swerve, Device removal, Excessive speed	Precision reached up to 91%
(Ferreira et al., 2017)	Accelerometer, Magnetometer, Gyroscope, Linear Acceleration	ANN, SVM, RF, BN	Aggressive braking, Aggressive acceleration, Aggressive left/right turn, Aggressive left/right lane, changing, Non-aggressive, events	RF has the highest accuracy
(Yu et al., 2017)	Accelerometer, Orientation	SVM, NN	Weaving, Swerving, Side slipping, Fast U-turn, Turning with a wide radius, Sudden braking	The accuracy of SVM reached up to 95.36% while neural network has accuracy of 96.88%
(Lu et al., 2018)	Accelerometer, Magnetometer, Gyroscope	RF, Naïve Bayes, DT, KNN, SVM	Stopping, Going straight, Left/Right turn	RF has highest accuracy

2.2.4. Accuracy of Smartphones

Only few studies have been conducted to compare the accuracy and reliability of smartphone-based collected data to other sources of information such as fixed GPS devices and Inertial Measurement Units (IMUs). Paefgen et al. (2012) conducted a study of driving events detection to compare the performance of smartphones and On-Board Diagnostics (OBD). Results

confirmed the correlation between the events detected by smartphones and ODB devices. Smartphones were found to overestimate critical events when compared to OBDs, whereas roadway conditions and smartphone's positions could significantly affect the performance of smartphones. Saiprasert et al. (2013) showed that speed data from smartphone were as accurate as the values from car's speedometer with a speed offset of approximately 4 km/h. Data fusion algorithms were developed in the previous studies (Chowdhury et al., 2014; Ghose et al., 2016) based on GPS and inertial collected data to infer the speed of a vehicle. It was concluded that the estimated speed is comparable to the OBD based tachometer readings.

Smartphones have several limits compared to OBDs. For example, pose uncertainties and other noise-inducing factors make smartphone be potentially less reliable as sensor platforms. Moreover, the mobile measurements tend to overestimate critical driving events, possibly due to deviation from the calibrated initial device pose, and road type is a significant factor that is not considered in most current state-of-the-art implementations (Paefgen et al., 2012). Besides, Vlahogianni and Barmounakis (2017) found smartphone sensors would get triggered by mobile phone usage while driving, which could affect the event detection process and the critical events' threshold values.

2.2.5. Battery-Saving Strategies

Battery consumption is another concern for the implementation of the driver behavior detection system, especially for the GPS sensor, since it has a high battery consumption. Generally, accelerometer consumes much less power than the GPS receiver, and battery depletion rate increases with the frequency of sensor readings (Prelicean et al., 2014). Moreover, other studies (Ben Abdesslem et al., 2009; Prelicean et al., 2014) showed that it would consume more power

than the normal situation if the GPS sensor was running and there were not enough visible satellites (fewer than 3 visible satellites).

Several approaches can be applied as battery saving strategies. First, the detection system only requires GPS data when a vehicle is moving (Oshin et al., 2012). Second, from a sensor-fusion point of view, the combination of accelerometer and GPS can also reduce battery consumption, which requires the absolute position correction using GPS signals at frequent intervals (Lin et al., 2014). Third, Bareth and Kupper (2011) proposed a hierarchical approach which utilizes either the cellular network, Wi-Fi network, or GPS according to specific situations. For example, it is not recommended to use Wi-Fi in rural areas because of its low accuracy. Fourth, Zhuang et al. (2010) applied a Sensing Piggybacking strategy to the vehicle location problem, which can share the GPS sensor data between different applications. This method decreases the number of GPS invocations and saves battery power accordingly. The last approach can reduce battery consumption by transferring the computationally intensive calculations from the smartphone to the cloud, depending on the mobile network capacity and available bandwidth (Akherfi et al., 2018; Liu et al., 2016a).

2.2.6. Smartphone-Based Transportation Mode Classification

Cellular positioning was used for mobile-based transportation mode classification in previous studies while a GNSS receiver and accelerometers were used more recently (Oshin et al., 2012). Motorized transportation modes are more likely to change direction than non-motorized modes. Distribution of mean absolute accelerations can be expected to be very different for trains, cars, and buses. Cars are more prone to be involved in quick driving maneuvers. Public transport tends to make a larger number of stops per driving kilometer. Also, semantic information such as home

and work location or information on behavior models can be used for transportation mode identification. Several smartphone sensors such as gyroscope, accelerometer, and GPS have been used to classify the transportation modes.

Eftekhari and Ghatte (2016) used gyroscope to identify motorized and non-motorized modes. They also used accelerometer to detect stationary states in either non-motorized mode or motorized mode. Gyroscope sensor could provide three values $g_{x_t}, g_{y_t}, g_{z_t}$ corresponding to the angular velocity of the smartphone in x, y and z axes in the time sample t, respectively. The new feature $G_{\mu s}$ was defined in the following equation:

$$G_{\mu s} = \frac{1}{6} \sum_{t=3s}^{3s+5} \sqrt{g_{x_t}^2 + g_{y_t}^2 + g_{z_t}^2}$$

As shown in Figure 4, $G_{\mu s}$ was the value for the moving status in the motorized and non-motorized modes. The moving status of the non-motorized mode was determined by applying appropriate threshold θ for $G_{\mu s}$.

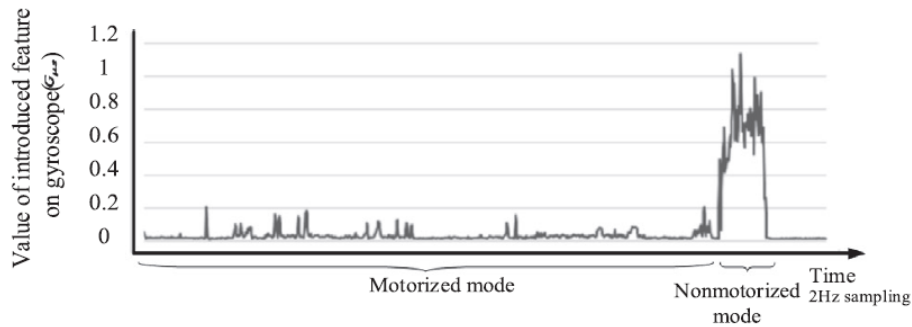


Figure 4. Values of $G_{\mu s}$ in the moving status in motorized and non-motorized modes (Eftekhari & Ghatte, 2016)

As Figure 5 **Figure 5. Threshold based on distribution function (Eftekhari & Ghatte, 2016)** shows threshold θ was determined by mean and the standard deviation values from the best-fit distribution function in each dataset.

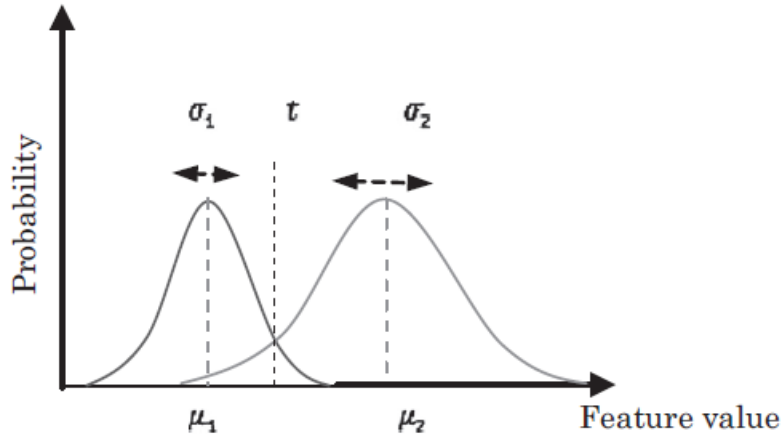


Figure 5. Threshold based on distribution function (Eftekhari & Ghatee, 2016)

Xiao et al. (2012) designed a transportation mode detection algorithm derived from GPS, GSM, and accelerometer data. Positions within a moving time window not smaller than a threshold could be clustered into a stop, and transportation mode between stops could be identified. Semanjski and Gautama (2016) used GPS position, duration of trip, distance covered, user id and timestamps by means of SVM to perform the classification. Biljecki et al. (2013) described a novel algorithm by means of GPS data each as a single segmentation. Trajectories were extracted from GPS data and broken into single-journey segments. The MF value was defined as membership function for every transportation mode. As shown in Figure 6, a classification approach was used to generate a hierarchy.

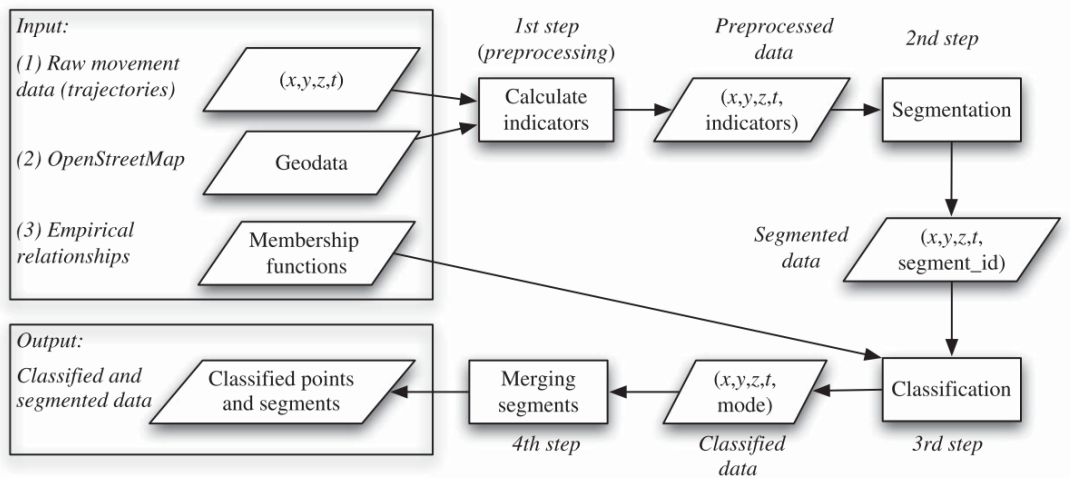


Figure 6. Flowchart of the implementation of the prototype (Biljecki et al., 2013)

Fang et al. (2016) used a database collected by HTC Company since 2012, involving 224 volunteers of 10 transportation modes. The paper classified motorized modes (motorcycle, car, bus, metro, train, and HSR) as single mode, i.e., on a vehicle. The rest were divided into still, walking, running, biking. As Figure 7 shows, various machine learning approach were used. Results showed that the proposed features could enhance accuracy, in which the SVM algorithm provided the best performance in classification accuracy whereas it consumed the largest prediction time.

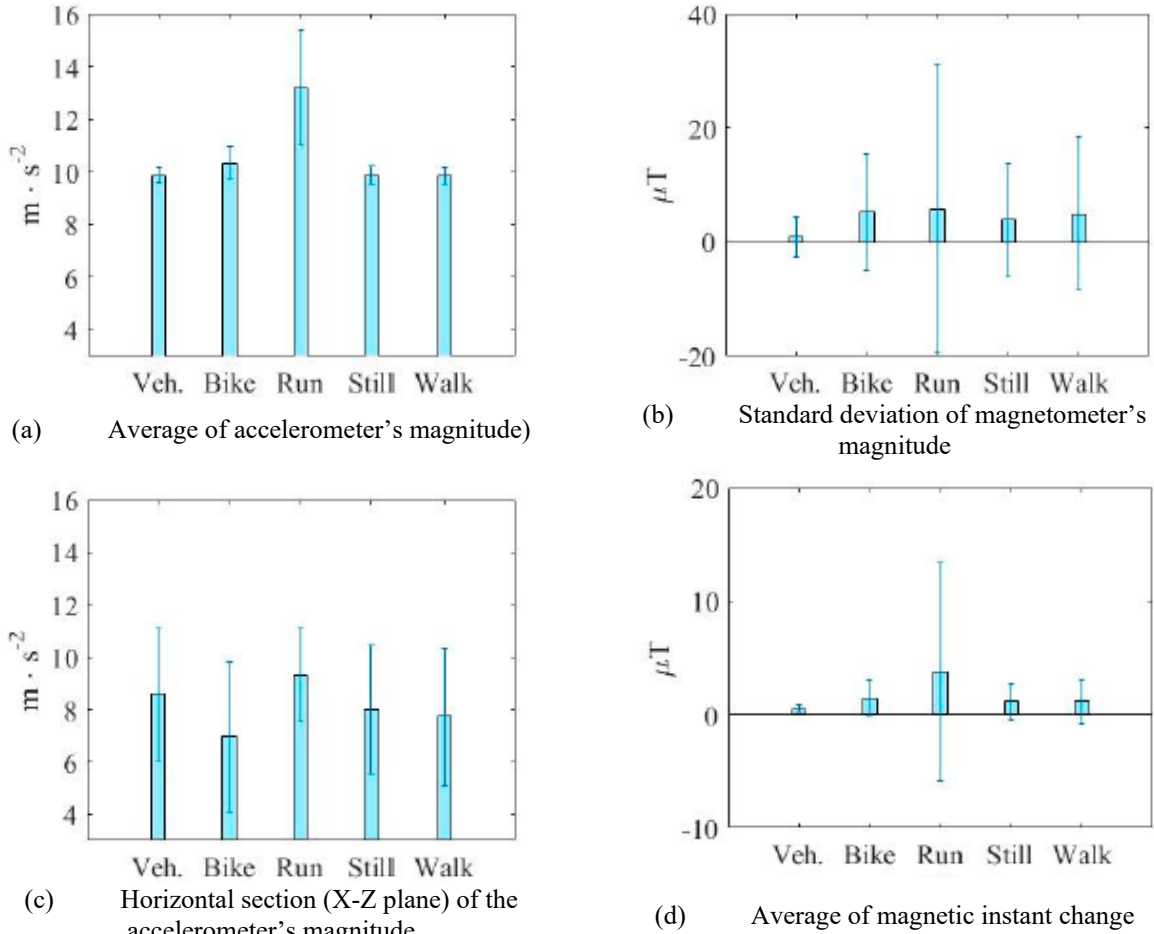


Figure 7. A typical added feature and comparison with an original one in transportation mode (Fang et al., 2016)

Tian et al. (2014) proposed a pedestrian dead reckoning (PDR) based navigation algorithm to update pedestrian's location information by using the magnetic, angular rate and gravity (MARG) sensors, all of which are equipped in existing commercial smartphone. The proposed navigation algorithm consists of step detection, stride length estimation, and heading estimation.

Kang and Han (2014) showed the step detection and stride length estimation consists of detecting the peak point of pedestrian movement using the accelerometer on pedestrian's smart devices. Consequently, the algorithm calculated the step number. The basic model for the PDR algorithm is shown in Figure 8.

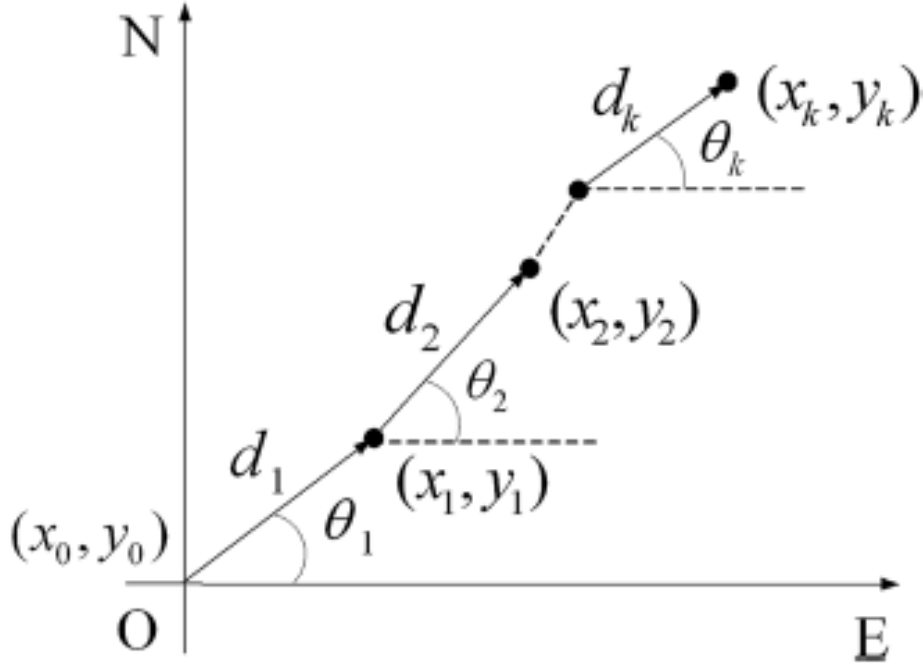


Figure 8. Basic model for the PDR algorithm (Tian et al., 2014)

Starting from the user's initial position (x_0, y_0) , the next position (x_1, y_1) can be calculated by utilizing the heading angle θ_1 and the displacement d_1 . And the pedestrian's k th position are calculated as shown below.

$$\begin{cases} x_k = x_0 + \sum_{i=1}^k d_i \cos \theta_i \\ y_k = y_0 + \sum_{i=1}^k d_i \sin \theta_i \end{cases}$$

where θ_i and d_i ($i = 1, \dots, k$) stand for the heading angle and the stride length of step i .

A quaternion-based extended Kalman filter (EKF) is introduced to determine the user's heading direction for each step (Chen et al., 2015). First, the algorithm uses the three-axis gyroscope to measure the angular velocities of the smartphone. In these experiments, the attitude angles collected from the gyroscope may start from incorrect initial conditions (Valérie & Combettes, 2014). And when either the accelerometer is not stationary, or the magnetometer is

exposed to interferences, the accumulated errors by the gyroscope measurement noise and the absolute attitude angles from magnetometer/accelerometer could provide an incorrect estimation on the heading angle. Therefore, the integration of gyroscope, accelerometer and the magnetometer for the calculation of attitude angles can effectively improve the heading precision (Deng et al., 2015). That's where the extended Kalman filter is applied to merge all the sensors' information to obtain an accurate estimation on attitude angles. The EKF model is shown as:

$$\begin{cases} X_{k+1} = FX_k + W_k \\ Z_{k+1} = HX_{k+1} + V_{k+1} \end{cases}$$

Tian et al. (2014) showed that the proposed EKF can provide more accurate and stable estimation of pedestrian heading compared to conventional attitude fusion methods, such as gradient descent algorithm (GDA), especially in the environment where the magnetic disturbance exists.

2.2.7. Smartphone Application Development for Transportation

2.2.7.1. Congested Traffic Condition Detection

According to Bhoraskar et al. (2012), frequently braking indicated congested traffic conditions. Also, magnetometer could be used to find horizontal orientation for axes reorientation instead of waiting for braking events.

Yu et al. (2016) proposed a system called SenSpeed to estimate the vehicle speed using accelerometer and gyroscope. Speed could be calculated from integral of the acceleration data over time. However, due to engine vibrations and white noise, the accumulative errors caused large deviations from the true speed of the vehicle. Thus, this system took turns, stops and uneven

road segments as reference points to eliminate acceleration errors. The experiment result demonstrated that SenSpeed was accurate and robust in real driving environments.

Yoon et al. (2007) proposed a method using GPS location data to identify traffic condition. After getting GPS data of spatio-temporal traffic status, traffic patterns on each road traffic states in a road-specific manner could be identified. Two experiments showed this system achieved over 90% accuracy, as Figure 9 showed, when there was a construction work in the experimental area, temporal and spatial speed got some significant changes. Figure 9(a) presents the cumulative time-location plot, Figure 9(b) shows the spatio-temporal traffic status plot using above algorithm, and Figure 9(c) the relative traffic status and the percentile from past data.

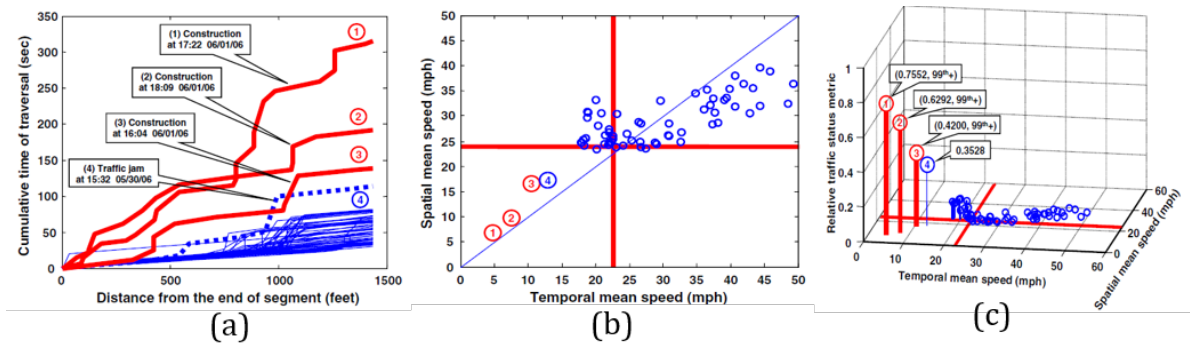


Figure 9. Detection of unusual traffic status on a road segment (Yoon et al., 2007)

2.2.7.2. Accident Detection

In Nericell system developed by Microsoft, accelerometer, microphone, GPS and GSM radio were used to detect bumps, potholes, braking and honking, in which case there might exist congested road ahead. Then the information was aggregated to annotate traffic map. This system helps to find a relatively stress-free route for users (Mohan et al., 2008).

In another App called WreckWatch, smartphones automatically detected traffic accidents using accelerometers and acoustic data, then notified central server after an accident immediately (White et al., 2011).

Base on road acoustics, noise signals such as tire noise, engine noise, engine idling noise, occasional honks and air turbulence noise of multiple vehicles were used to discriminate between the different traffic density states (Tyagi et al., 2012).

2.2.8. APP Development

An APP called WalkSafe is a pedestrian safety app for mobile phone users who walk and talk while crossing roads. WalkSafe can be activated during phone calls. It uses back camera and accelerometer sensors in smartphones and classification pipeline to alert users of unsafe cars approaching them.

The training process was offline. It used Gentle AdaBoost algorithm with CART decision tree and simple tree stump models. Online detection process contained camera sampling, preprocessing, and pre-trained model to do vehicle detection and alert sending. Accelerometer was used to identify orientation of the mobile phones, in order to make the test image aligned to the direction of gravity (Wang et al., 2012).

To evaluate the performance of this APP, cropped car front views, rear-view images and Google Street View images were selected to split as training set and testing set. The mobile phone was held by a pedestrian standing at the side of the street. The test results are shown in Table 4. The image preprocessing was done locally instead of in the cloud due to the limited latency of transmission speed. So, the algorithm took up about 15% of the mobile phone (Nexus One).

Table 4. A 30-minute WalkSafe car detection experiments in real world (Wang et al., 2012).

Coming car direction	Front view	Rear view
Number of cars detected	84	57
Number of cars missed	25	18
True positive rate	77%	76%
Number of false positive	3	
Maximum detection distance	50 meters	

There is a project led by Collaborative Sciences Center for Road Safety (CSCRS). This research project aims at developing an Android mobile app to alert pedestrian when they are near areas of high traffic density, including the presence of autonomous vehicles and when it is unsafe to cross the street. Also explore how the arrival of vehicle-to-vehicle services can improve upon the base case. The goal will be to measure the effectiveness of the app as an alerting and warning system for distracted users.

Samsung has a walking app that alerts distracted pedestrians when they're about to bump into a wall or cross at a red light. Though it is unknown which part of smartphone sensor this app makes use of, it can send alerts to users that they are using phone while walking and detect vehicle horns to alert users about nearby cars.

Liu et al. (2016b) proposes a framework only utilizes the smartphone (similarly, using the accelerometer, gyroscope, GPS) to sense the surrounding events and provides alerts to drivers.

Local device location, speed and travel direction information can be broadcast (Chandra et al., 2007) to calculate its direction vector. Using these direction vectors and local information, a logical map can be generated as shown below. Various pedestrians' and vehicles' vectors can be compared to determine the potential conflict at a point on the logical map at the same time. If a possibility of collision in their future travel paths is determined, using the speed and location

information, the estimated time to collision (ETC) will be calculated. Time to register an alert for a vulnerable road user is around 1 to 2 seconds (Renaudin et al., 2013). Using calculated ETC and comparing it to a pre-set threshold table can determine whether an emergency alert should be pushed to drivers and pedestrians.

2.2.9. Summary

This section has reviewed the existing studies about driver behavior detection and transportation mode classification based on smartphone sensors. Several key issues can be concluded. First, accelerometer, gyroscope are the two fundamental sensors for the detection of abnormal driver behavior, such as aggressive right/left turn, aggressive acceleration/breaking, U-turn, etc. In addition, it is better to use all three axes of the two sensors. Second, sensor-fusion is a better method than using single sensor. Third, raw sensor data need some pre-processing approaches such as reorientation and filtering. Rough set theory can make the rule-based method to become more accurate and flexible, and random forest is more accurate than other machine learning methods. Also, to classify transportation modes, different sensor data should be used including accelerometer, magnetometer, gyroscope, and GPS. Finally, the smartphone data could also be used for traffic condition and crash detection, which could be used for developing warning messages.

2.3. Detection Sensors and Communication Technology

In practice, the major sensors used to detect different transportation modes are LiDAR, cameras, etc. In the following sections, the LiDAR vendors are summarized. The detection and tracking methods by using LiDAR or camera are reviewed. In addition, the different communication technologies for the connected vehicle are also reviewed.

2.3.1. LiDAR Vendors and Products

LiDAR is an exceptional device as it can literally provide a 3D view of the world as the infrared pulses bounce off surrounding objects and return to the sensor with precise distance information. Because the pulses are so close, the speed at which objects are moving relative to the vehicle is easily calculated. Currently, the major LiDAR vendors are as follows:

- Quanergy
- Velodyne
- LeddarTech
- Continental
- Luminar Technologies
- Innoviz Technologies
- Trilumina
- Phantom Intelligence
- Oryx Vision

More discussion on Quanergy, Velodyne and LeddarTech is included here because of their rich product variety, strong market presence, and applications. Luminar Technologies has industry in Silicon Valley and in Orlando, but their products are not very suitable for the specific application of vehicle and pedestrian detection from a single point.

2.3.1.1. Quanergy LiDARs

Quanergy, headquartered in Silicon Valley, is the World's Leading Provider of LiDAR-based Smart Sensing Solutions. Quanergy provides a sensing system including three components: sensors, software platform, and processing unit. Three types of LiDAR sensors are provided including S3-8, M8, and S3-1.

The appearance of S3-8 is shown in Figure 10. S3-8 contains no moving parts on either the macro or micro scale. The sensor contains three main components including emitter, receiver, and processor. Here, optical phased array technology enables electronic beam steering. Electronic beam formation and steering are achieved by interference among light beams emitted from a large number of phase-controlled optical antenna elements. Through the interaction of three major components, the S3-8 generates half a million data points per second. The directional beam, formed by interference, hits a target and is received and processed to measure the Time of Flight that gives the precise distance. It can provide real-time 3D object detection, classification, and tracking ("S3-8 LiDAR").



Figure 10. Quanergy LiDAR S3-8

As shown in Figure 11 , M8 is a 360-degree long range LiDAR sensor enabling ubiquitous uses of smart sensing in dynamic situations-made and tested for 3D mapping, security, and harsh industrial environments. The M8 sensor's small design works well both in day and night with no IR Infra-Red signature needed. Also, it can work in any weather condition. Multiple laser beams and Time-of-Flight (TOF) depth perception develop 3D point clouds for spatial sensing. The M8 has a detection range that exceeds 200 meters on high reflectivity surfaces. At 400 feet, the sensor

can map almost any surface, including low reflectivity ones. The high performance makes it an ideal scanner for UAV mapping applications ("M8 LiDAR").



Figure 11. Quanergy LiDAR M8

The S3-1 is a small LiDAR sensor (Figure 12) with a range over 100 meters. This solid-state sensor is ideal for applications requiring object detection at long range including drones, intelligent robotics, security, smart spaces, and industrial automation. For drones, payload and battery runtime benefit greatly from these compact sensors (S3-1 LiDAR).



Figure 12. Quanergy LiDAR S3-1

Qortex is Quanergy's core perception software platform which serves as the main technology framework for LiDAR sensor perception. This software is compatible with Quanergy's LiDAR sensors. The platform is shown in Figure 13.



Figure 13. Qortex interface

The three primary functional building blocks of the platform are object detection, tracking, and classification. When integrated with Quanergy's LiDAR sensors, Qortex provides the necessary information in real time, which could give vehicles an advanced perception of their environment. Qortex also offers a solution package to address smart city applications such as counting vehicles across multiple lanes, tracking vehicles across intersections, detecting pedestrians, and DSRC communication of obstacles and road hazards("Qortex,").

The last component of the system that needs to be considered is the central processing unit, known as the Quanergy Processing Unit (QPU). The QPU comes into two varieties, the QPU-L7 and the QPU-Mini (Figure 14). The units could vary based on processing power, which limits the maximum number of sensors each device can handle.



(a) QPU L7



(b) QPU Mini

Figure 14. QPU unit

The QPU is a portable computer with software pre-installed for sensor operation, data collection, and running other Quanergy applications. The QPU can manage up to four of Quanergy's M8 LiDAR sensors in applications related to transportation, industrial automation, 3D mapping, and security. The QPU is designed to be used in a complete system involving other essential Quanergy parts including:

- 19 VDC QPU power supply with cables connecting to wall power and to 12 VDC car power

outlet

- M8 sensor, with IP69K-rated ingress protection from dust and water
- 24 V AC/DC sensor power adapter with cable connecting to wall power
- Base plate: pre-attached to the sensor with four screws
- 3-foot sensor cable: wired into sensor, with other end IP67-rated dust/waterproof connector
- Sensor interface cable: one end an IP67-rated connector, the other end subdivided into:
 - Ethernet sub-cable with RJ45 connector (attaches to network port of QPU)
 - Power sub-cable with ferrule connector (attaches to 24 V power supply)
 - A sub-cable with M12 3-pin connector (attaches to PPS, NMEA, and ground)






To complete the system, users also need to provide:

- Preferred mouse + keyboard + monitor
- Power source (100-240 VAC or 12 VDC)
- Mounting surface (e.g., vehicle grill-height platform or camera tripod).

2.3.1.2. Velodyne LiDARs

Founded in 1983, Silicon Valley-based Velodyne is regarded as a LiDAR pioneer after developing some of the first LiDAR sensors for industrial systems. Five sensor types are provided by Velodyne including VLS-128, ULTRA PUCK, HDL-32E, HDL-64E, and VLP-16. The summary of the different sensors is presented in Table 5. Velodyne LiDARs are suitable for use in autonomous vehicles, industry equipment, 3D mapping and surveillance.

Table 5. Velodyne LiDARs

Types	Product Appearance	Description
VLS-128		<p>This device is especially built for autonomous vehicle applications. It provides real-time 3D data up to 0.1-degree vertical and horizontal resolution with up to 300-meter range and 360° surround view.</p>
ULTRA PUCK		<p>This LiDAR has 32 channels, dual returns, up to 200m range, up to 1.2 million points per second. It could provide real-time 3D data that includes distance and calibrated reflectivity measurements along with rotational angles.</p>
HDL-32E		<p>The HDL-32E LiDAR sensor is small, lightweight, and ruggedly built. It features up to 32 lasers across a 40° vertical field of view. The HDL-32E measures only 5.7" high x 3.4" in diameter, weighs less than two kilograms and was designed miscellaneous real-world autonomous navigation, 3D mobile mapping and other LiDAR applications.</p>
HDL-64E		<p>The HDL-64E LiDAR sensor is designed for obstacle detection and navigation of autonomous ground vehicles and marine vessels. Its durability, 360° field of view and very high data rate makes this sensor ideal for the most demanding perception applications as well as 3D mobile data collection and mapping applications. The HDL-64E's innovative laser array enables navigation and mapping systems to observe more of their environment.</p>
VLP-16		<p>VLP-16 is a light weight LiDAR sensor. It has 16 channels, 100m range, 360-degree horizontal field of view and 15-degree vertical field of view.</p>

2.3.1.3. Leddartech LiDARs

LeddarTech is a Canada-based LiDAR manufacturing company. Six sensor types are provided by LeddarTech including LeddarVu, Leddar M16, LeddarOne, T16, IS16 and, d-tec. The summary of the different sensors is presented in Table 6. Among these sensors, T16 and d-tec can be utilized in traffic detection at intersections. Others have different application areas.

Table 6. LeddarTech LiDARs


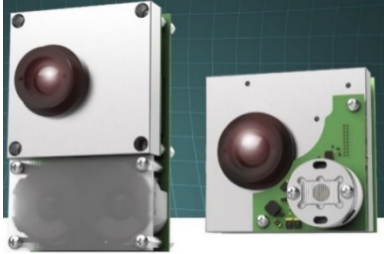
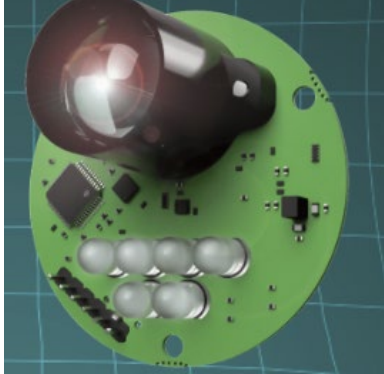



Types	Appearance	Description
LeddarVu		<p>LeddarVu is a versatile solid-state LiDAR sensor module that delivers very good detection and ranging performance in a small, robust package. LeddarVu modules provide the ability to detect and track multiple objects simultaneously over eight distinct segments with superior lateral discrimination capabilities. It has a refresh rate upto 100Hz and detection range up to 185m.</p>
Leddar M16		<p>This one is a solid-state LiDAR solution that combine wide-beam flash illumination with 16 independent detection segments to simultaneously deliver rapid, continuous and precise detection and ranging for multiple objects along with good lateral discrimination. Its range is up to 146m.</p>
LeddarOne		<p>LeddarOne Sensor Module is dedicated to single point detection and precise distance measurements. Its range is up to 40m.</p>

Table 6, continued

Types	Appearance	Description
T-16	 <p>The image shows the LeDdar T-16 sensor, which is a white, dome-shaped device with a clear lens and a black base. The text 'LEDDAR T-16' is visible on the side.</p>	<p>Specifically designed for traffic management systems. T-16 gives 200Hz measurement rate for high speed vehicle detection. Its detection range is up to 75m.</p>
IS16	 <p>The image shows the LeDdar IS16 sensor, which is a bright orange, rectangular device with a circular lens and a black base. The text 'LeDdar' is visible on the top.</p>	<p>With detection range up to 50m, IS16 is specifically designed solid-state LiDAR sensor for industrial market.</p>
d-tec	 <p>The image shows the LeDdar d-tec sensor, which is a white, dome-shaped device with a clear lens and a black base. The text 'LeDdar' is visible on the side.</p>	<p>Compiling data at a rate of thousands of times per second, LeDdar d-tec provides reliable detection of all kinds of traffic in different weather and lighting conditions.</p>

2.3.2. Detection and Tracking Methods

2.3.2.1. Video-Based Detection and Tracking

A video-based tracking system should be able to meet the following criteria:

- Automatically segment each vehicle from the background and from other vehicles so that all vehicles are detected.
- Correctly detect all types of vehicles including motorcycles, passenger cars, buses, etc.

- Functions under a wide range of traffic conditions such as light traffic, congestion, varying speeds in different lanes.
- Functions under a wide variety of lighting conditions including sunny, overcast, twilight, night, rainy, etc.
- It should be a real-time system.

Coifman et al. (1998) proposed a real-time Digital Signal Processing chip-based computer vision system which demonstrated good performance under different challenging conditions. The algorithm has been used for detecting vehicles in daylight, twilight, and nighttime conditions.

The system computes a projective transform or homography between image coordinates (x, y) to world coordinates (X, Y) . Features are tracked in world coordinates to exploit known physical constraints of vehicle motion (Figure 15).

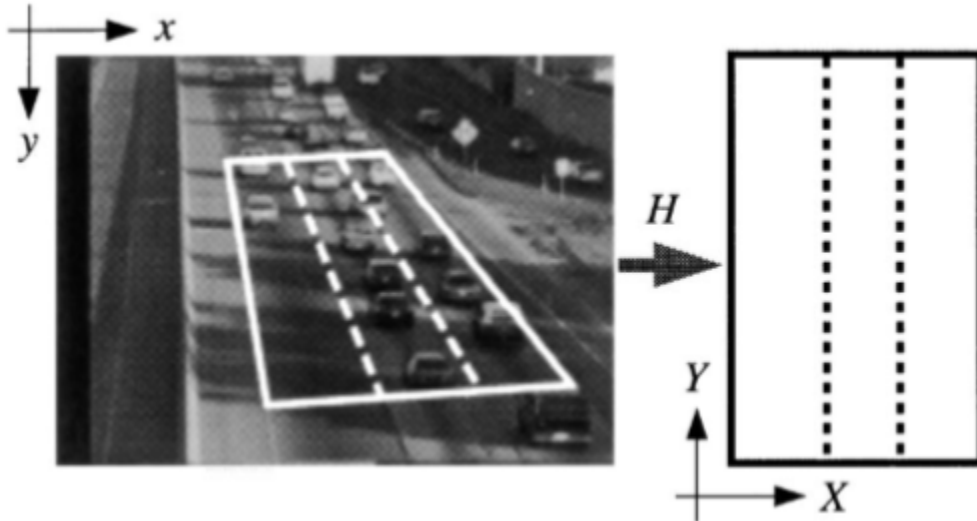


Figure 15. Transformation of image coordinates to world coordinates

Figure 16 demonstrates the block diagram of the whole system. Features from the same vehicle will follow similar tracks and features from different vehicles will have distinctively different tracks. Common motion over entire feature tracks is used to group features from

individual vehicles and reduce the probability that long shadows will link vehicles together. The tracking module uses Kalman filtering to predict a given corners location and velocity in the next frame using world coordinates.

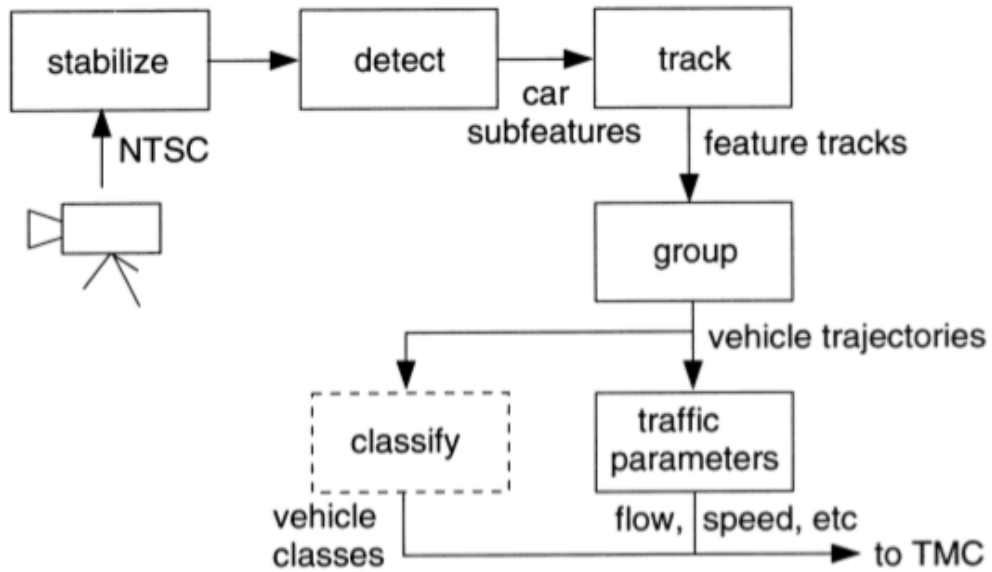


Figure 16. Block diagram of the vehicle tracking system

Figure 17 (a) shows some example corners detected by the system. Figure 17 (b) demonstrates the temporal progression of several corner features in the image plane. Once corner features reach the exit region, they are grouped into vehicle hypotheses by the grouping module, shown in Figure 17 (c).

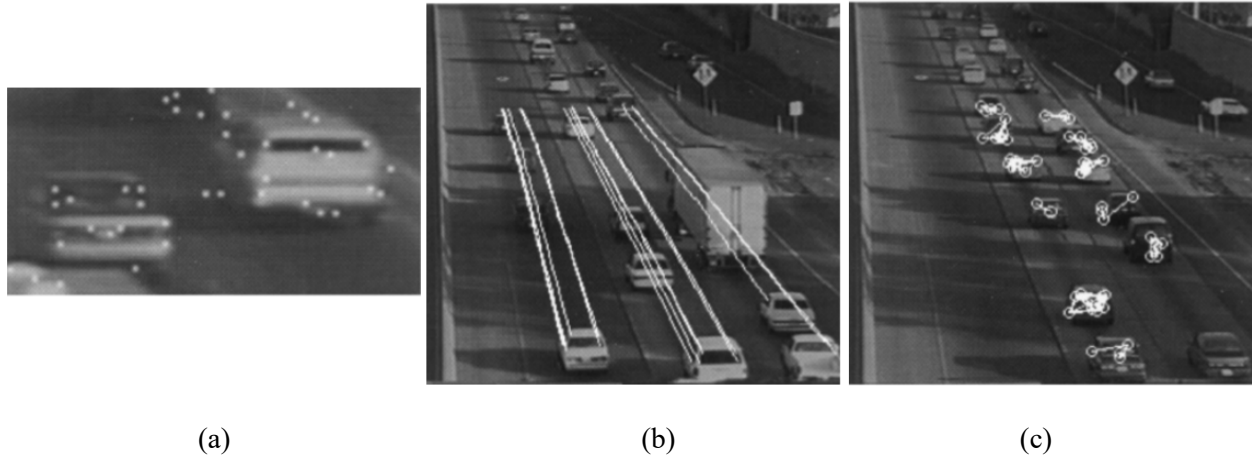


Figure 17. (a) Corner features identified by the tracker, (b) sample features from the tracker, and (c) sample feature groups from the grouper.

After the tracker has extracted vehicle trajectories, it is possible to send this data over a low bandwidth communication link for scene reconstruction and automated surveillance at a remote location. The system tracks the vehicles well when they are mobile and also it does not lose them when they are stuck on the jam and partially occluded. The performance of the vehicle tracker is 7.5 Hz in uncongested traffic, dropping to 2 Hz in congested traffic.

The system showed promising results during day to night transition, night to day transitions, congestions, occlusion, camera vibration due to wind, and long shadows linking vehicles together. Camera motion during high wind is accounted for by tracking a small number of fiducial points.

1) Motion detection

In order to correctly detect and track vehicles and pedestrians, it is important to discuss motion detection first. Motion detection aims at segmenting regions corresponding to moving objects

from the rest of the system. The process of motion detection usually involves environment modelling, motion segmentation, and object classification.

i. Environment Modelling

The active construction and updating of environmental models are most important to object tracking. Environmental models can be classified into 2-D models in the image plane and 3-D models in real-world coordinates. Due to their simplicity, 2-D models have more applications. The key problem for fixed cameras is to automatically recover and update background images from a dynamic sequence. Illumination variance, shadows, and shrinking branches could bring many difficulties to the update of background images. Different approaches have been undertaken for solving this problem, including adaptive Gaussian estimation (Kohle et al., 1997), temporal average of image sequence (Friedman & Russell, 1997; Koller et al., 1994), and parameter estimation based on pixel processes (Sun et al., 2000; Grimson et al., 1998). Ridder et al. (1995) modeled each pixel value with a Kalman filter to compensate for illumination variance. Stauffer and Grimson (1999) presented a theoretical framework for recovering and updating background images based on a process in which a mixed Gaussian model was used for each pixel value and online estimation was used to update background images in order to adapt to illumination variance and disturbance in background.

ii. Motion Segmentation

The goal of motion segmentation is to detect regions corresponding to moving objects such as vehicles and humans. Several conventional approaches for motion segmentation are summarized in

Table 7. The table indicates that the background subtraction is highly dependent on a good background model to reduce the effect of dynamic scene changes. Meanwhile, the temporal differencing uses pixel-wise subtraction. The optical flow method is computationally expensive for real-time video applications.

Table 7. Motion segmentation studies

Method	Description	Studies
Background subtraction	This method is popular in static background cases. It detects moving region in an image by taking the difference between the current image and the reference background in a pixel by pixel fashion. It is simple, but extremely sensitive to changes in dynamic scenes derived from lighting, extraneous events, and etc.	(Stauffer & Grimson, 1999)
Temporal differencing	Temporal differencing makes use of the pixel-wise differences between two or three consecutive frames in an image sequence to extract moving regions. Temporal differencing is very adaptive to dynamic environments, but generally does a poor job of extracting all the relevant pixels, e.g., there may be holes left inside moving entities	(Lipton et al., 1998)
Optical flow	Optical-flow-based motion segmentation uses characteristics of flow vectors of moving objects over time to detect moving regions in an image sequence. Optical-flow-based methods can be used to detect independently moving objects even in the presence of camera motion. However, most flow computation methods are computationally complex and very sensitive to noise and cannot be applied to video streams in real time without specialized hardware.	(Meyer, et al., 1997; Meyer et al., 1998)

Besides these methods, Cutler et al. (2000) implemented a mixed Gaussian classification model for each pixel to extend expectation maximization (EM) algorithm. This model classifies the pixel values into three separate predetermined distributions corresponding to background, foreground, and shadow. It also updates the mixed component automatically for each class

according to the likelihood of membership. Hence, slowly moving objects are handled perfectly, while shadows are eliminated much more effectively.

2) Object Classification

Image sequences captured by a camera at an intersection probably include vehicles, humans, bicycles, etc. To further analyze their behaviors, it is mandatory to correctly classify moving objects. The main methods for classifying moving objects are shape- (Lipton et al., 1998) and motion-based (Cutler et al., 2000) methods. For the shape-based methods, different shape information of motion regions such as points, boxes, silhouettes and blobs can be used for classifying moving objects. The dispersion and area of image blobs are used as classification metrics to classify all moving-object blobs into humans, vehicles and clutter. Temporal consistency constraints are considered to make classification results more precise. On the other hand, a similarity-based technique is described to detect and analyze periodic motion for the motion-based method. By tracking an interesting moving object, its self-similarity is computed as it evolves over time. For periodic motion, its self-similarity measure is also periodic. Therefore time-frequency analysis is applied to detect and characterize the periodic motion, and tracking and classification of moving objects are implemented using periodicity.

3) Object Tracking

After motion detection, it is necessary to track the trajectory of the vehicle, pedestrian, and bicycle. There are four major categories for object tracking: region-based, active –contour based, feature –based, and model-based tracking. Each method has its own advantage and disadvantages (Table 8). Occlusion handling, computational complexity, initialization etc. are important parameters for choosing a tracking method.

Table 8. Object tracking studies

Method	Study	Description	Advantage	Disadvantage
Region based tracking	(Kalman et al., 1990; Kilger, 1992)	Region-based tracking algorithms track objects according to variations of the image regions corresponding to the moving objects. For these algorithms, the background image is maintained dynamically, and motion regions are usually detected by subtracting the background from the current image.	Works well in scenes containing only few objects	Cannot reliably handle occlusion between objects and 3-D pose of objects cannot be acquired
Active contour-based tracking	(Baumberg and Hogg, 1997; Galata et al., 2001; Mohan et al., 2001; Wu & Huang, 2001)	Active contour-based tracking algorithms track objects by representing their outlines as bounding contours and updating these contours dynamically in successive frames. These algorithms aim at directly extracting shapes of subjects and provide more effective descriptions of objects than region-based algorithms.	Reduces computational complexity, simple description of objects	Tracking precision is limited at contour level, highly sensitive to initialization of tracking
Feature based tracking	(Chachich et al., 1997; Schiele, 2006; Coifman et al., 1998; Malik et al., 1997)	Feature-based tracking algorithms perform recognition and tracking of objects by extracting elements, clustering them into higher level features and then matching the features between images. The features used in these algorithms include centroids, perimeters, areas, some orders of quadrature and colors etc. The features also include line segments, curve segments, corner vertices, and etc.	Can adapt successfully and rapidly to allow real time processing and tracking of multiple objects. Feature based tracking algorithms can handle partial occlusions by using information on object motion, local features and dependence graphs.	Recognition rate of objects based on 2-D feature is low and these algorithms are generally unable to recover 3-D pose of objects. Stability of dealing effectively with occlusion is generally poor.
Model based tracking	(Tan et al., 1994); (Lou et al., 2002) (Koller et al., 1993)	Model-based tracking algorithms track objects by matching projected object models, produced with prior knowledge, to image data. Vehicle pose estimation based on single characteristic line segment matching, 3-D wire-frame vehicle model with edge-based image featuring etc. are conducted in these studies.	Robust-algorithms, obtains good result under occlusion. Algorithms can acquire the 3-D pose of the objects.	Different models are to be constructed, high computational cost

Recently, more advanced tracking methods are STR-STRuck (Hare et al., 2015), TLD-Tracking Learning and Detection (Kalal et al., 2010), and FBT-Foreground Background Tracker (Nguyen & Smeulders, 2006). Most of the works on object detection were based on SIFT-Scale Invariant Feature Transform (Lowe, 2004) and HOG-Histogram of Oriented Generations (Dalal & Triggs, 2005) in the last decade. Some recent studies on object detection are YOLO (Redmon et al., 2016), Fast YOLO (Redmon et al., 2016), R-CNN (Girshick et al., 2014), and Faster R-CNN (Ren et al., 2015). Regions with Convolutional Neural Network features or R-CNN is a simple and scalable object detection algorithm. But cost of R-CNN is an issue, which has been drastically reduced at proposals (Girshick, 2015; Ren et al., 2015). YOLO can process at 45 frames per second and a smaller version of it, named fast YOLO, can achieve 155 frames per second and maintain excellent accuracy at the same time.

4) Vision-Based Pedestrian Detection and Tracking

Pedestrian detection methods have employed a variety of techniques and features. Some have focused on increasing the speed of detection, whereas others have focused on accuracy.

Papageorgiou and Poggio (2000) proposed sliding window detectors by applying support vector machines (SVM) to an over complete dictionary of multiscale Haar wavelets. Viola and Jones (2004) introduced integral images for fast feature computation and a cascade structure for efficient detection by utilizing AdaBoost for automatic feature selection.

Inspired by SIFT (Lowe, 2004), Dalal and Triggs (2005) popularized histogram of oriented gradient (HOG) features for detection by showing substantial gains over intensity-based features. Zhu et al. (2006) sped up HOG features by using integral histograms.

Gavrila and Philomin (1999) employed the Hausdorff distance transform and a template hierarchy to rapidly match image edges to a set of shape templates. Wu & Nevatia (2005) utilized a large pool of short line and curve segments, called “edgelet” features, to represent shape locally. Boosting was used to learn head, torso, leg, and full body detectors. Then, Wu and Nevatia (2007) extended this approach to handle multiple viewpoints. Similarly, “shapelets” (Sabzmeydani & Mori, 2007) are shape descriptors discriminatively learned from gradients in local patches. Liu et al. (2009) proposed “granularity-tunable” features that allow for representations with levels of uncertainty ranging from edgeless to HOG type features.

Viola et al. (2005) proposed computing Haar-like features on difference images, resulting in large performance gains. For non-static imaging setups, however, camera motion must be factored out. Dalal et al. (2005) modeled motion statistics based on an optical flow field’s internal differences, compensating for uniform image motion locally.

There has been a lot of work on the tracking of pedestrians to infer trajectory-level information. Some researchers formulated tracking as frame-by-frame association of detections based on geometry and dynamics without considering particular pedestrian appearance models (Alonso et al., 2007; Gavrila & Munder, 2007). Other researchers utilized pedestrian appearance models coupled with geometry and dynamics (Baumberg, 1998; Heap and Hogg, 1998). Some studies furthermore integrated detection and tracking in a Bayesian framework, combining appearance models with an observation density, dynamics, and probabilistic inference of the posterior state density. The integration of multiple cues involved combining separate models for each cue into a joint observation density. The inference of the posterior state density is usually formulated as a recursive filtering process (Arulampalam et al., 2001). Extensions that are

especially relevant for pedestrian tracking involve hybrid discrete / continuous state-spaces and efficient sampling strategies (Heap & Hogg, 1998).

Recently novel range of methods have been emerged based on Deep Neural Networks, showing impressive accuracy gains in pedestrian detection. However, Deep Neural Network (DNN) models are known to be very slow (Girshick et al., 2014; Giustiet al., 2013), especially when used as sliding-window classifiers. Ouyang and Wang (2013) and Luo et al. (2014) both apply deep networks in combination with fast feature cascades. However, getting real-time solutions for pedestrian detection is difficult. WordChannel features (Daniel and Nedeveschi, 2014) provide a real-time solution on the GPU (16FPS), but at a notable loss in average miss rate (42%). The seminal VeryFast method (Benenson et al., 2012) runs at 100 FPS but with even further loss in miss rate. Angelova et al. (2015) presented a real-time approach to object detection that exploits the efficiency of cascade classifiers with the accuracy of deep neural networks.

2.3.2.2. LiDAR-Based Detection and Tracking

Zhao et al. (2006) developed a joint tracking and classification algorithm for moving objects for intersection monitoring. Mendes et al. (2004) used their multi-target detection and tracking system for collision avoidance for their Cybercar.

Several fixed 270° LiDARs were used by Zhao et al. (2005) to cover a large area and track people in crowds. The system was able to handle 100 trajectories simultaneously. Fod et al. (2002) also used multiple LiDARs. They devoted a significant amount of work to evaluating the accuracy of their tracking system by comparing it to ground truth obtained by cameras. Fod et al. (2002) developed a feature detection system for real-time identification of lines, circles, and people's legs. Xavier et al. (2005) used an adaptation of the Hough transform in the feature extraction procedure to interpret scanned segments as primitive features. Montemerlo et al. (2002) presented

a conditional particle filter for simultaneously estimating the pose of a mobile robot and the positions of nearby people in a previously mapped environment. Arras et al. (2007) applied AdaBoost to train a strong classifier from simple features of groups of neighboring beams corresponding to legs in range data. Table 9 compares different sensors for pedestrian detection. LASER scanner and RADAR has the capability to work with challenging illumination with relatively simpler algorithm. LASER scanners have a comparatively higher field of view than RADAR.

Table 9. Comparison between different sensor modalities for pedestrian detection (Viola, Jones, & Snow, 2005)

Sensor type	Field of view	Angular resolution	Detection range	Range resolution	Illumination	Hardware cost	Algorithmic complexity
Rectilinear camera	Med.	Med./High	Low/Med.	Med.	Passive reflective, needs ambient light	Low	High
Omni camera	Large	Low/Med.	Low	Low	Passive reflective, needs ambient light	Med.	High
Near IR	Med.	Med./High	Med.	Med.	Active, works in dark	Low	High
Thermal IR	Med.	Low/Med.	Low/Med	Low	Emissive, works in dark	High	Med.
PMD sensor	Med.	Low	Med.	Low/Med.	Modulated light source	Med.	Med.
RADAR	Small	Low	High	High	Active, works in dark, rain, fog.	Med.	Low
LASER scanner	Large	Med.	Med.	High	Active, works in dark	High	Low

Table 10 lists three studies which used LiDAR sensors for pedestrian detection. LiDAR scanners output radial distance at discrete azimuth angles in the scanning plane. These data are clustered into object-based range discontinuities and grouping measurements near each other in the 3D space (Fuerstenberg and Willhoeft, 2001; Fuerstenberg et al., 2002).

Table 10. Pedestrian detection using LiDAR sensors

Publication	Sensors	Attention focusing stage	Classification/verification stage
(Bellotti et al., 2004)	NIR cameras, synchronized LiDAR illuminators	Pixel level correlation between image windows and pedestrian models to discard non-pedestrian areas.	Haar wavelets features and separate SVMs for frontal and lateral pedestrians
(Fuerstenberg and Willhoeft, 2001)	LiDAR scanner	Raw data is clustered into objects based on range discontinuities and tracked over time	The objects are classified using models of the object outlines and dynamic behavior.
(Fuerstenberg et al., 2002)	Multilayer LiDAR scanner	Detects objects by grouping the measurements that are near each other in 3D space, tracks them with Kalman filter.	The objects are classified using models of the object outlines and dynamic behavior, the system also warns the driver or activates ABS in case of imminent collision.

Table 11 summarizes studies using multiple sensors to detect pedestrians. Steinfeld et al. (2004) described a side collision warning system for transit buses with the use of multiple sensors in order to cover the surroundings of the vehicle. Szarvas et al. (2006) used a sequential combination of LiDAR-based object detector and an image classifier using CNN.

Table 11. Pedestrian detection using multiple sensors

Publication	Sensors	Attention focusing stage	Classification/verification stage
(Steinfeld et al., 2004)	Cameras, LiDAR scanners, RADAR	Time of flight, video-based detection, LiDAR-based triangulation. Combines outputs from number of sensors to create a map	Generates multiple levels of warnings from front and side components, such as for passing and cutting in. LASER line generator used for curb detector.
(Szarvas et al., 2006)	LiDAR, visible light	LiDAR used to create range map that is used to identify image locations and scale to search pedestrians	Convolutional neural network used for image-based feature extraction and classification

2.3.3. Communication Technology

In the recent years, different types of communication technologies (e.g., Wi-Fi, LTE, and DSRC) are suggested for connected vehicle communication. Considering the high level of the mobility of the nodes, multipath, and environmental dynamics caused by vehicles and pedestrians, IEEE proposed a modified version of the Wireless Local Area Network (WLAN) protocol, IEEE802.11p which is known as Dedicated Short-Range Communication (DSRC), for V2V and V2I communication. A dedicated bandwidth of 75 MHz in the 5.850 to 5.925 GHz band has been allocated by the US Federal Communications Commission (FCC) (Lee & Lim, 2013). Xu et al. (2017) compared the performance of DSRC and LTE and it was showed that DSRC outperformed LTE in case of all safety latency, but LTE has higher coverage. David and Flach (2010) mentioned that the most common technologies used for V2P communications are Cellular (UMTS/LTE), Wi-Fi Direct, and DSRC technologies.

The communication delay of cellular networks is in the order of seconds. Wi-Fi Direct enables ad hoc communication of Wi-Fi devices without the necessity of an access point (AP). However, its frequent network reformation might result in an unacceptable delay.

2.3.3.1. Wi-Fi Communication

Anaya et al. (2014) considered that the communication connection between pedestrians and vehicles via Wi-Fi is unstable, sometime even broken, especially in suburban areas. If the signal is blocked by the obstacles, the communication distance would be significantly shorter. Dhondge et al. (2014) lists the comparison of various wireless protocols, as shown in Table 12.

Table 12. Comparison of wireless protocols (Dhondge et al., 2014)

Protocol	Data Rate	Range	Device Mobility Constraint
DSRC	3 - 27 Mbps	< 1 Km	> 60 Mph
Wi-Fi (With Association)	6 - 54 Mbps	< 100 m	< 5 Mph
Cellular	< 2 Mbps	< 10 Km	> 60 Mph

Zhu et al., (2013) mentioned that Wi-Fi has the process of authentication and association before data transmission and is more suitable for 1-to-N communication. This association and authentication process can cause the latency.

Wi-Fi Direct is a Wi-Fi standard enabling devices to establish a direct Wi-Fi connection with each other without requiring a wireless router (access point). Dhondge et al. (2014) used the Wi-Fi Direct feature of Android powered devices to establish an ad-hoc network between the smart devices in the vehicles. It will eliminate the association latency of the Wi-Fi feature using Wi-Fi Beacon stuffing. The smart devices operate on Wi-Fi direct mode will transmit beacon signals every 100 ms and are passively scanned in Wi-Fi Hotspot/Direct discovery mode. The experiment conducted showed it works well up to vehicle traveling speed of 70 Mph.

Wi-fi Beacon Stuffing can carry the sensing and location information via beacon frames (Radu & Marina, 2013). These beacon frames are used to declare the existing Access Point (AP) and can be transferred by the AP without association and authentication process and embeds the intended messages within the Service Set Identifier (SSID) field, which consists of 32 characters.

The communication latency to notify a pedestrian regarding danger is around 1 second for the Wi-Fi Direct association time (Liebner et al., 2013). However, Wi-Fi Direct feature has great limitation on the coverage distance between smart devices of pedestrians and drivers. According to Hussein et al.(2016), in nearly 90% of all accidents the vehicle speed is up to 70km/h (20m/s), and Wi-Fi Direct communications can only cover a speed of less than 25 km/h.

Valérie and Combettes (2014) found that ad-hoc communication with WLAN chipsets is not applicable in high travel speed scenarios. And a problem of using beacon to transfer sensing information is that the beacon can only be transferred from the AP to the client, which causes the one-way transmission. In practical scenario, it is not enough that only a part of the devices transfers the information, as every device needs to broadcast its mobility information to the others.

2.3.3.2. Dedicated Short-Range Communications (DSRC)

Dedicated short-range communications (DSRC) are a set of wireless communication protocols and standards designed for automotive use. The range and mobility that DSRC offers cannot be matched by the regular Wi-Fi. However, it will require vehicle manufacturers to provide the DSRC, also known as 802.11p enabled modules in the vehicles. It is also not ubiquitously available yet on smart devices such as a pedestrians' cellphones.

Tahmasbi-Sarvestani et al. (2017) proposes a Vehicle-to-Pedestrian (V2P) framework to provide situational awareness and hazard detection based on the most common and injury-prone crash scenarios. In this research, it used a Qualcomm DSRC-enabled smartphone and a Hyundai-Kia DSRC-equipped vehicle.

DSRC is particularly designed for vehicle safety purposes to have low latency and high interoperability. The USDOT plans to mandate the use of DSRC-enabled units in all vehicles. The main barrier on deploying the DSRC technology is equipping the smartphones of vulnerable road users (VRU) with DSRC. Recently Qualcomm addressed this concern by announcing their capability to override and upgrade existing Wi-Fi firmware to operate in DSRC band without any additional hardware cost.

Wu et al. (2014) mentioned that National Highway Traffic Safety Administration (NHTSA) announced in February 2014 to move towards mandating DSRC for all new light vehicles in the near future. DSRC is a reliable, non-line of sight (NLOS), relatively long range and low latency communication between vehicles and pedestrians.

Artail et al.(2017) proposed a system enabling the Road Side Unit (RSU) and used the cellular communication and vehicular communication to obtain vectors of positions of cars and nearby pedestrian to predict probable collision events. The experiment results showed that in order to achieve 80% packets delivery rate, the transmission distance needs to be smaller than 100 meters without any obstacles. This may be hard to obtain in real-life traffic scenarios.

Liu et al. (2016c) compared the performance of communication technology DSRC and Long-Term Evaluation (LTE). It's proved that although the communication system based on LTE is better than DSRC in terms of packets lost rate, a larger communication delay exists on the LTE system.

The key enabling components start with the implementation of a DSRC stack within the Wi-Fi chipset on the smartphone and leveraging the phone's GPS and inertial system.

The transmission range of DSRC is about 300 meters. Its band is in the range of 5.85GHz to 5.925 GHz. Since it is adjacent to the legacy 5GHz Wi-Fi band, no hardware modification is required at the RF front end. However, modifications were needed in the firmware and driver for the current generation of Qualcomm Wi-Fi chipset to tune to the DSRC band.

The firmware and the drivers mainly deal with I/O operations on the data arriving from communication links and sensors. The firmware modifications involved the inclusion of the DSRC band operation as well as enabling the reception of broadcast packets by setting the

interface to operate in promiscuous mode. Enabling the functionality of broadcast packets is a key component to the DSRC solution since there is no Wi-Fi association in the system architecture.

Jing et al.(2017) systematically evaluated and accessed the reliability of car-to-pedestrian communication safety systems based on the vehicular ad-hoc network environment. It validated that the DSRC communication (packet arrive rates) performed well, in the experiment scenario that the distance between pedestrian and vehicles is 50 meters or more. The pedestrian preferred warning position ranged from 6 meters to 9 meters before the conflict point.

The DSRC-based communication requires on-board unit (OBU) for V2V and road side unit (RSU) for V2I communication. One of the main drawbacks of DSRC-based communication is that OBU is not financially feasible to incorporate in all existing vehicles. Since OBU is financially infeasible, we are trying to emulate the functions of OBU into smartphone apps. So, in a hybrid scenario, some vehicles (CVs) are OBU embedded and some vehicles are connected through the smartphone. There are two cases for connected vehicles - (i) vehicles with smartphone apps to vehicles with DSRC, and (ii) vehicles with smartphone apps to RSU communication, where one end uses DSRC communication and another end uses cellular communication. For the first case, vehicle with DSRC can add an LTE transceiver to communicate with vehicles with smartphone apps (Xu et al., 2017). In Figure 18, CVs communication using both DSRC and LTE is illustrated. For vehicles with smartphone apps to RSU communication, RSU can only communicate with DSRC communication band. In that case, conventional vehicles need an access point (AP) to build communication link in between DSRC and cellular band. Software-defined radio (SDR) can be used as AP. Another solution could be the use of a low-cost in-vehicle IEEE802.11p (DSRC) device. Smartphone can send data to the device using Bluetooth or Wi-Fi. The IEEE802.11p device can be used to communicate with RSU or other on-board units.

According to Qualcomm Mobile Developers ("China Mobile Develops Roadside Units for LTE-V2X PC5 Direct Communication Featuring Qualcomm C-V2X Chipset Solution,"), they will manufacture cellular band RSU for cellular-V2X. If they can successfully manufacture cellular-RSU with maintaining required latency, smartphone app will be able to communicate with RSU in cellular band.

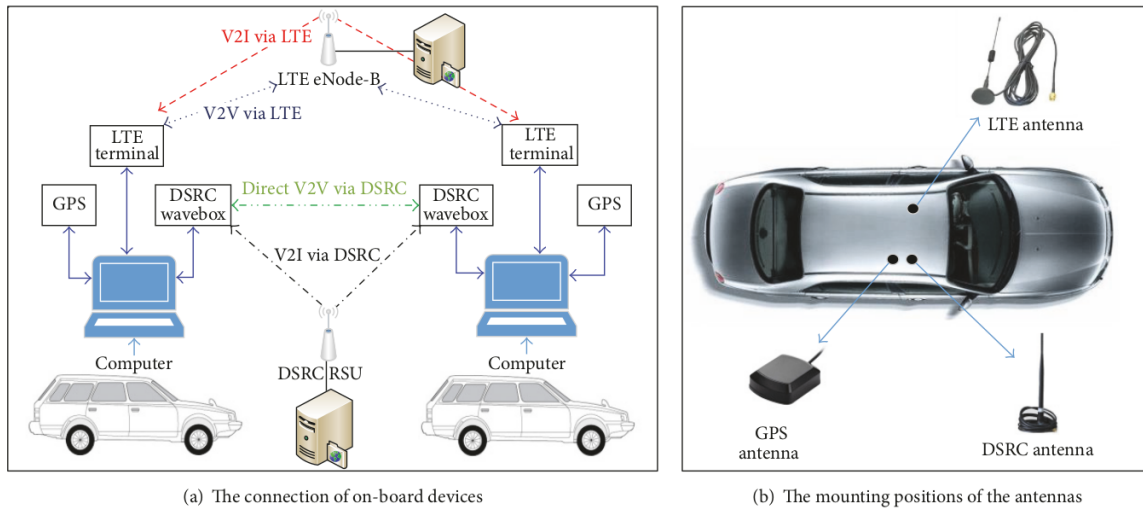


Figure 18. OBU unit mounted with multiple antenna and communication connection (Xu et al., 2017)

2.3.3.3. Vehicular Ad-hoc Network

Vehicular Ad-hoc Network (VANET) is the most common connected vehicles (CVs) data delivery communication method. VANET is basically ad-hoc communication mesh network of highly dynamic vehicular nodes. Traffic data (Packets) from different sources being forwarded in the mesh nodes (i.e., vehicles) need to know the structure of the mesh network, the direction to forward the data flow/traffic based on the vehicle location and route to the destination nodes/vehicles (Dey et al., 2016). Multi-hop routing protocols for VANET provide different options that could be used for CV technology applications based on their specific latency

requirements. In Table 13, different active safety latency requirements are shown (Xu et al., 2017). The same latency should be applied for the smartphone applications.

Table 13. Active safety latency requirements (units: seconds) (Xu et al., 2017)

Function	Latency (second)
Traffic signal violation warning	0.1
Curve speed warning	1.0
Emergency electronic brake lights	0.1
Pre-crash sensing	0.02
Cooperative forward collision warning	0.1
Left turn assistant	0.1
Lane change warning	0.1
Stop sign movement assistance	0.1

2.3.4. Mobile Cloud Computing

Mobile Cloud Computing (MCC) forum defines MCC (Sanaei et al., 2014) as “Mobile Cloud Computing at its simplest refers to an infrastructure where both the data storage and the data processing happen outside of the mobile device. Mobile cloud applications move the computing power and data storage away from mobile phones and into the cloud, bringing applications and mobile computing to not just smartphone users but a much broader range of mobile subscribers”. MCC addresses the inherent mobile computing problems including resource scarcity, frequent disconnections, and mobility. MCC is basically a remote server of offloading computation from mobile devices. Computational resources can either be provided by a remote cloud to serve all mobile devices or be provided by a local cloud by utilizing resources from other nearby mobile devices (Fernando et al., 2013). MCC applications have also been discussed in the context of VANETs, giving rise to the term vehicular cloud computing (VCC). Vehicular clouds differ from internet clouds in case of the unpredictability of the vehicles’ behavior. Vehicles can

unexpectedly leave the VANET, and hence the computation and storage scheme must be made resilient to such events. VCC is often performed utilizing the resources of surrounding vehicles without requiring an active internet connection (vehicular cloudlets). The use of vehicular cloudlets will generally both reduce the transmission cost and increase service availability (Wahlström et al., 2017). Yu et al. (2013) proposed a three-layer vehicular clouds architecture. These are the vehicular cloudlet, the roadside cloudlet which is composed of dedicated local servers and roadside units, and the central cloud (the basic internet cloud). As illustrated in Figure 19, usage of the added resources provided by the outer layers of the vehicular cloud tends to come at the expense of an increased communications delay.

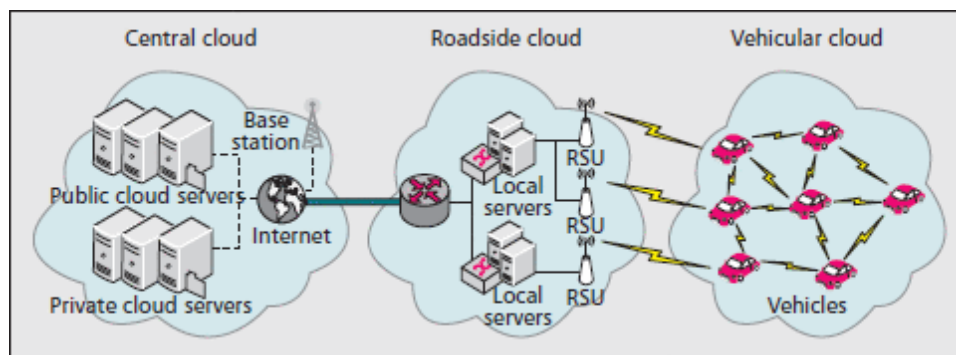


Figure 19. Proposed cloud-based vehicular network architecture (Yu et al., 2013)

Although Vehicular Cloud (VC) in a static mode has the same behavior as conventional cloud computing (CC), cars remain on the road busy with various dynamically changing situations or events every day. The occurrence of vehicles in close vicinity to an incident is very often an unplanned process which can be called mobility. Again, the resources of those vehicles that form a VC to alleviate the event must occur naturally, which can be called autonomy. Conventional clouds do not have these features which are considered a significant major characteristic of VCs (Whaiduzzaman et al., 2014). Whaiduzzaman et al. (2014) presented an extensive comparative study of CC and VCC, as shown in Table 14. Although CC can be accessed

anytime and anywhere, VCC can only be accessed where vehicle network is existing. Running or idle vehicle can make a cloud autonomously while CC is not autonomous. VCC provides pooling storage and serves this to user as storage service provider same as CC.

Table 14. Comparative Study of VCC and CC (Whaiduzzaman et al., 2014)

Characteristics	Description	CC	VCC
On demand elastic application	Get the required service when necessary and applications can run and uses resources dynamically.	Yes	Possible
Virtualization	Several requests can be served by one machine but pretend to be separate machines.	Yes	Yes
Pay as you go	The business model of paying and using the services.	Yes	Possible
Any time any where	Services are available and online every moment from anywhere.	Yes	No
Network as a Service	Providing communication and network-related services.	Yes	Yes
Storage as a Service	Provide pooling storage and serve this to the user as a storage service provider.	Yes	Yes
Corporation as a Service	By carpooling, information and entertainment services are provided.	Possible	Yes
Commercials and Infotainment	Commercials, information and entertainment for the drivers.	Yes	Yes
Planned and unplanned disaster management	Disaster management using roads and vehicles.	Possible	Yes
Large traffic event management	Managing large-scale traffic management.	Possible	Yes
Moving Network Pool	Moving vehicles are dynamically pooled and make a huge network.	No	Yes
Autonomous Cloud formation	Running or idle vehicle can make a cloud autonomously.	No	Yes
Automatic Cloud Federation	On the move, several clouds can be formed as a large single cloud.	No	Yes
Trust and authentication management	Provide trust management and authentication to build confidence.	Yes	Yes
Mobility of clouds	Clouds or cloud provider serves while moving.	No	Yes

2.3.5. Summary

Among the LiDAR vendors, only Quanergy is interested in passive vehicle and pedestrian detection from a single point at an intersection. Other vendors either focus on other sectors of the industry or lack tangible experience with this application. Quanergy had previous deployments in THEA CV Pilot and the City of Las Vegas project. Quanergy M8 or solid state LiDARs are possible options. Qortex software and Quanergy Processing Unit QPU are also required. The QPU can upload the data to the server. The QPU runs the Q-Guard/Qortex software and translates the raw input detection feed and point-cloud data and outputs an object list that can be utilized in various functions.

Computationally efficient algorithms like cascaded deep neural networks, or YOLO, or fast R-CNN methods have the potential to be useful in vehicle and pedestrian detection and

tracking. An integrated vision and LiDAR-based approach would also be a good option for smartphone-application-based vehicle-to-pedestrian communication project.

Wi-Fi (LTE) and DSRC are the two main communication technologies for the connected-vehicle information example. The smartphone application should mainly rely on Wi-Fi (LTE) communication. However, DSRC communication technology is USDOT's protocol dedicated for future V2V and V2P information exchange. Hence, it could be beneficial if the smartphone application could communicate with the DSRC.

Finally, efficient computation methods should be adopted for smartphone application development to save battery and communication resources.

2.4. OBU Applications

The vehicle-to-vehicle (V2V) devices could be categorized into three types: the original equipment manufacturer (OEM) devices, the aftermarket devices, and the infrastructure-based devices. The first two types of devices are commonly referred to as on-board units (OBUs). On-board units intuitively mean those V2V devices built into the vehicle.

An OEM device is an electronic device fully built or integrated into a vehicle at the time of vehicle manufacture. An integrated V2V system comprising OEM devices could potentially provide haptic warnings to alert the driver (such as tightening the seat belt or vibrating the driver's seat) in addition to audio and visual warnings provided by the aftermarket devices (Harding et al., 2014).

An aftermarket device is a device added to a motor vehicle after its original assembly, which aims to improve the vehicle's comfort, convenience, performance, or safety. The devices can be added to a vehicle at a vehicle dealership, as well as by authorized dealers or installers of

automotive equipment. An aftermarket-device-based V2V system could provide advisories and warnings to the driver of a vehicle similar to those provided by an OEM-installed V2V device.

In the Connected Vehicle Safety Pilot Program conducted by NHTSA (Harding et al., 2014), three types of aftermarket devices were installed into vehicles: vehicle awareness devices (VAD), aftermarket safety devices (ASD), and retrofit safety devices (RSD). The VAD is the simplest one among the three and it only transmits basic safety messages (BSMs) to nearby vehicles.

A VAD does not have any safety applications or driver-vehicle interfaces (DVI), and it cannot provide any advisories or warnings to a driver. The ASD, is similar to the VAD, but also has the ability to both receive and transmit a BSM to nearby vehicles. Also, it contains safety applications that can provide advisories or warnings to the driver. The RSD is more fully integrated than the ASD: it connects to the vehicle and receives information from the vehicle's data bus to support operation of various applications on the device. The vehicle's data bus is a communication system that could transfer data between different components of the vehicle. The advantage to RSDs, as compared to the other types of aftermarket devices, is that they can potentially perform different or enhanced safety applications or execute more sophisticated applications because they can access a richer set of data (i.e., data from the data bus). Table 15 shows the definition and functionality of three types of aftermarket devices tested in the Connected Vehicle Safety Pilot Program conducted by NHTSA (Harding et al., 2014).

Table 15. Aftermarket safety device types (Harding et al., 2014)

Device Type	Definition	Method of Installation	Functionality
Vehicle Awareness Device	Device is able to be connected to the vehicle for power source. Device provides Basic Safety Message for surrounding vehicles.	Device would need to be installed by a certified installer on vehicles not equipped with V2V technology to ensure correct antenna placement and security. In the future, VADs might be mobile devices or stand-alone key fobs.	<ul style="list-style-type: none"> • Transmits BSM
Aftermarket Safety Devices (i.e., Self-contained)	Device is connected to the vehicle for power source, Device transmits BSM and receives BSMs to support safety applications for the driver of the vehicle in which it is installed.	This device only receives power from the vehicle; however, a certified installer would need to ensure correct antenna placement and security.	<ul style="list-style-type: none"> • V2V Safety applications • Receives and Transmits BSM • Driver-Vehicle Interface
Retrofit Safety Devices	Device is connected to the vehicle's data bus that provides BSM and safety applications for the driver of the vehicle in which it is installed.	This device needs to be connected to the vehicle's data bus, therefore would require an installer that can access this for the particular make of vehicle. Also, a certified installer would need to ensure correct antenna placement and security.	<ul style="list-style-type: none"> • V2V Safety applications • Receives and Transmits BSM • Driver Vehicle Interface • Integration into the vehicle data bus

An infrastructure-based device is roadside equipment with dedicated short-range communication (DSRC) devices that allow vehicles to receive information from the infrastructure and allow vehicles to update their security certificates. Communications infrastructure physically helps get the messages from the vehicles and from the Security Credentials Management System (SCMS), and helps get new certificates and the CRL from the SCMS to the V2V-equipped fleet. The communications infrastructure includes roadside equipment units (RSU), which contain a DSRC radio/cellular modem, a processor, connection ports, antennas, and software. The RSE uses wireless DSRC to send messages/materials to on-board unit (OBU). The RSE also connects to the SCMS via a wired connection (i.e., through the Internet), in order to support the transmission of reports from OBU through the RSU to the SCMS and the transmission of certificates from the SCMS through the RSU to the OBU. Security infrastructure helps ensure that the messages sent are trustworthy and helps remove malfunctioning devices from the system and protect against

outside threats. Generally speaking, security infrastructure will include computer hardware, software, and a physical location for all of the components of the SCMS, which will be connected via the Internet to the RSUs, which then connect to the V2V-equipped vehicles' OBU.

This literature review mainly concentrates on the on-board units (OBUs) including OEM devices and aftermarket devices, while the infrastructure-based device is not the focus of the literature review.

2.4.1. OBU Function Required of Connected Vehicle System Operation

In general, an integrated V2V system should be able to transmit an accurate and trustworthy BSM to the other vehicles. Also, it should be able to receive and interpret a BSM transmitted from another entity.

The signal sending and receiving is commonly based on DSRC technology. DSRC technology is a two-way short- to medium-range wireless technology that provides nearly instantaneous network connectivity and message transmission. DSRC has low latency and high reliability characteristics. It provides a 300-meter transmitting range and a 360-degree unobtrusive detection angle. With a designated licensed bandwidth of 5.9 GHz, DSRC permits reliable communication. In addition, it provides very high data transmission rates in high-speed vehicle mobility conditions which are critical characteristics for detecting potential and imminent crash situations (NHSTA, 2016).

In order to generate and send a BSM, we need a device which knows its own position. Thus, GPS antenna and receiver are needed. Then the device needs a computer processing unit to code its location and the information from other onboard sensors (e.g., speed, heading, and acceleration) to a qualified BSM data string with appropriate processing memory. When the BSM

is generated, a security module is needed to process and prepare the security information and certificates for transmission to provide the receiving vehicle assurance that the message is valid. This security information needs to be transmitted wirelessly as well. The security module is also called as the message authentication.

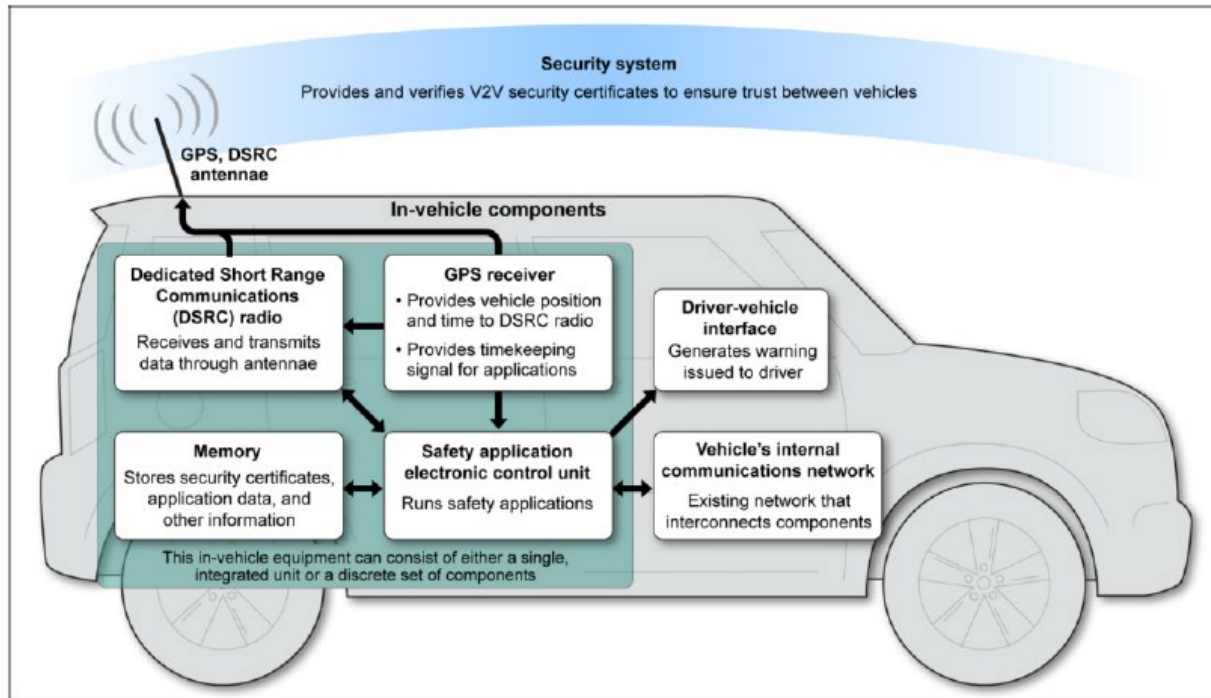
The primary message authentication approach proposed is a Public Key Infrastructure (PKI) that provides public-key encryption and digital signature services is implemented to ensure a trustworthy network environment and address the fundamentals of security: authentication, confidentiality, integrity, non-repudiation, and access control. The Notice of the Proposed Rulemaking (NPRM) also discusses two alternative approaches to message authentication. SCMS is a PKI system that is designed specifically for the V2V environment. SCMS thus is a comprehensive system that comprises the hardware, software, people, policies, standards, and procedures which are used to create, manage, distribute, monitor, and revoke digital certificates.

In order to receive and decode a BSM, a device must be capable of receiving the BSM that is transmitted from a nearby device and it must match the method of BSM transmission. The computer processing unit is also used here to decode the BSM properly. The GPS antenna and receiver are used to verify the relative distance between the sending device and the receiving device. Lastly, the device that receives the BSM must also have a security module that is capable of receiving and processing the security credential information.

In addition, in order to give drivers advisories and warnings, a driver-vehicle interface (DVI) is needed in the vehicle. This DVI also serves as an interface to the vehicle's sensors.

2.4.2. OBU Hardware Components of V2V System

According to the NHTSA report in 2016, a V2V device would require two DSRC radios (NHTSA, 2016) and a GPS receiver with a processor to derive information such as vehicle speed and predicted path from the device's GPS data. To improve the quality of the data that vehicle-based components could use to issue warnings, an inertial measurement unit to detect acceleration forces would be needed (Harding et al., 2014). The data collected by the vehicle is used by the safety applications to calculate the warning measurements. These warning measurements would be finally summarized and displayed by an interface which could be a screen or monitor. In addition, in order to store the data from multiple sources such as the security certificates, application data and other information, a memory device is needed. The whole V2V system is supported by the vehicle's internal communication network which enables different on-board components to connect to each other. Figure 20 illustrates the in-vehicle components needed for an integrated V2V system.



Sources: Crash Avoidance Metrics Partnership and GAO.

Figure 20. In-Vehicle components of a V2V system (Harding et al., 2014)

As for practical field testing research, there are three major wide-scale operation testing, including Transit Safety Retrofit Package (TRP) Project (Burt et al., 2014a, 2014b, 2014c), commercial connected vehicle test for safety applications (Howe et al., 2016a, 2016b, 2016c, 2016d; Stephens et al., 2014), and commercial vehicle test for retrofit safety device (LeBlanc et al., 2014a; LeBlanc et al., 2014b; Stephens et al., 2014).

The TRP was conducted by United States Department of Transportation (U.S. DOT). The aim is to conduct connected vehicle program to design and develop safety applications for transit vehicles that can communicate V2V as well as V2I for enhanced transit vehicle and pedestrian safety. The Transit Safety Retrofit Package (TRP) Project includes developing, testing, installing, and maintaining retrofit packages on three transit buses drawn from the University of Michigan transit fleets. Their proposed TRP system is illustrated in Figure 21.

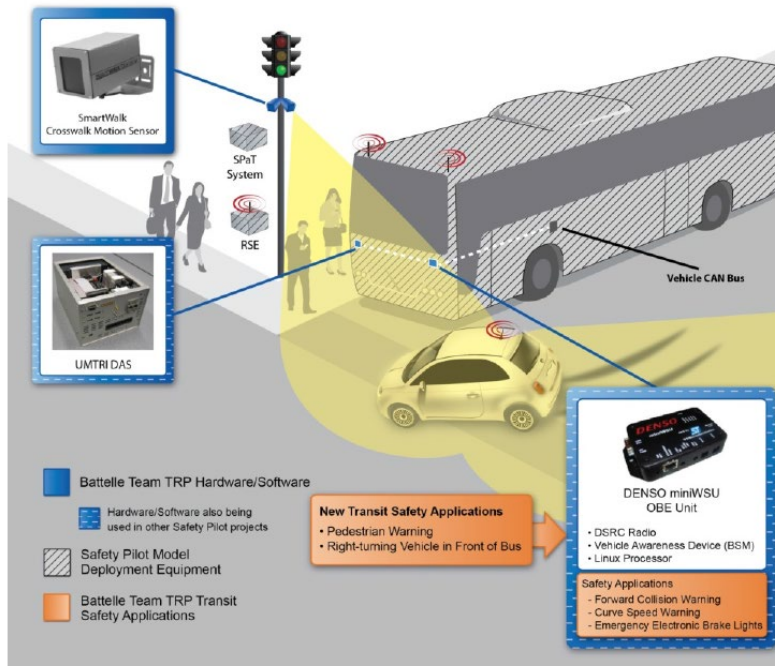


Figure 21. Illustration of the TRP system with all hardware components (Burt et al., 2014c)

As for the commercial connected vehicle test for safety applications (Howe et al., 2016a, 2016b, 2016c, 2016d; Stephens et al., 2014), the primary focus of this research was to develop the test procedures, with a secondary goal of evaluating the performance of the prototype V2V safety applications. As a critical part of achieving these goals, the U.S. DOT contracted with a Team led by Battelle to integrate and validate connected vehicle OBU and safety applications on selected commercial vehicles. The team was also required to support those vehicles in research and test activities that provide information and data needed to assess their safety benefits and support regulatory decision processes. Driver surveys are being conducted as a part of this project to evaluate acceptance of the connected vehicle technology and safety applications by drivers who were previously unfamiliar with the technology.

The U.S. DOT also conducted an associated project called Connected Commercial Vehicles—Retrofit Safety Device (CCV-RSD) Kit Project (LeBlanc et al., 2014a; Stephens et al., 2014). This project was to develop complete hardware and software that can be used in various

brands and models of heavy trucks. The RSD kits provide the functionality needed for cooperative V2V and vehicle-to-infrastructure (V2I) safety applications to support the Model Deployment and other USDOT connected vehicle projects. This project included testing and documentation needed for installation, operation, enhancement, and maintenance of the units. The OBUs are similar in these projects. The OBUs could either belong to the main hardware system or support system.

2.4.2.1. Main Hardware System

1) Wireless Safety Unit (WSU)

Wireless Safety Unit (WSU) is the core part of OBUs. It could interface with vehicle controller-area network (CAN) bus and other on-board vehicle equipment such as In-Vehicle display, data acquisition system, and DSRC antennas. It includes a DSRC radio which supports the basic safety message (BSM) communication via 5.9 GHZ DSRC. It also provides the platform to run safety applications like Forward Collision Warning. This device is being used in the development of transit safety retrofit package (TRP) project (Burt et al., 2014a, 2014b, 2014c), commercial connected vehicle test for safety applications (Howe et al., 2016a, 2016b, 2016c, 2016d; Stephens et al., 2014), and commercial vehicle test for retrofit safety device (LeBlanc et al., 2014a; LeBlanc et al., 2014b; Stephens et al., 2014). In these field test experiments, the WSU was customized by DENSO Company, who is the subcontractor of Battelle.

The DENSO WSU solution is a custom computing and communications platform specifically designed for the development, implementation, testing, and evaluation of 5.9 GHZ DSRC V2V/V2I applications. The device incorporates ST Microelectronics Cartesio+ chipset with an ARM11 application central processing unit (CPU), embedded Global Positioning System (GPS) receiver, and Atheros WAVE transceivers to facilitate the development of safety and non-

safety ITS applications. The software configuration uses Linux as a general purpose operating system (OS) (Burt et al., 2014b). The DENSO WSU includes a dual channel DSRC radio, GPS receiver and processing capability for the three basic safety applications.

The WSU may receive the following information: vehicle's current GPS position and time, vehicle's speed, vehicle's gear position, vehicle's brake status, vehicle's vehicle length, vehicle's vehicle type and basic safety messages including remote vehicle(s) path history and remote vehicle(s) position and heading.

The WSU may transmit the following information to the in-vehicle display: vehicle's current GPS position and time, vehicle's speed, vehicle's gear position, vehicle's foot brake status, remote vehicle(s) position and heading, remote vehicle(s) target classification, and safety application alert status like CSW alert and EEBL alert.

The WSU may transmit SAE J2735 Basic Safety Message (BSM) approximately once every 100 milliseconds on the DSRC radio. Figure 22 shows an example of the WSU.



Figure 22. WSU example

There may be more than one WSU in a V2V/V2I application depending on the multi-task requirements. For example, two WSUs are utilized sometimes. One is responsible for running

safety applications while the other operates as an alternative or channel-switching radio (Stephens et al., 2014).

Figure 23 shows the basic hardware architecture and electronic architecture of the WSU which was used in TRP project (Burt et al., 2014a, 2014b, 2014c). The miniWSU has a plastic molding that is approximately 106mm * 72mm * 25mm in dimension.

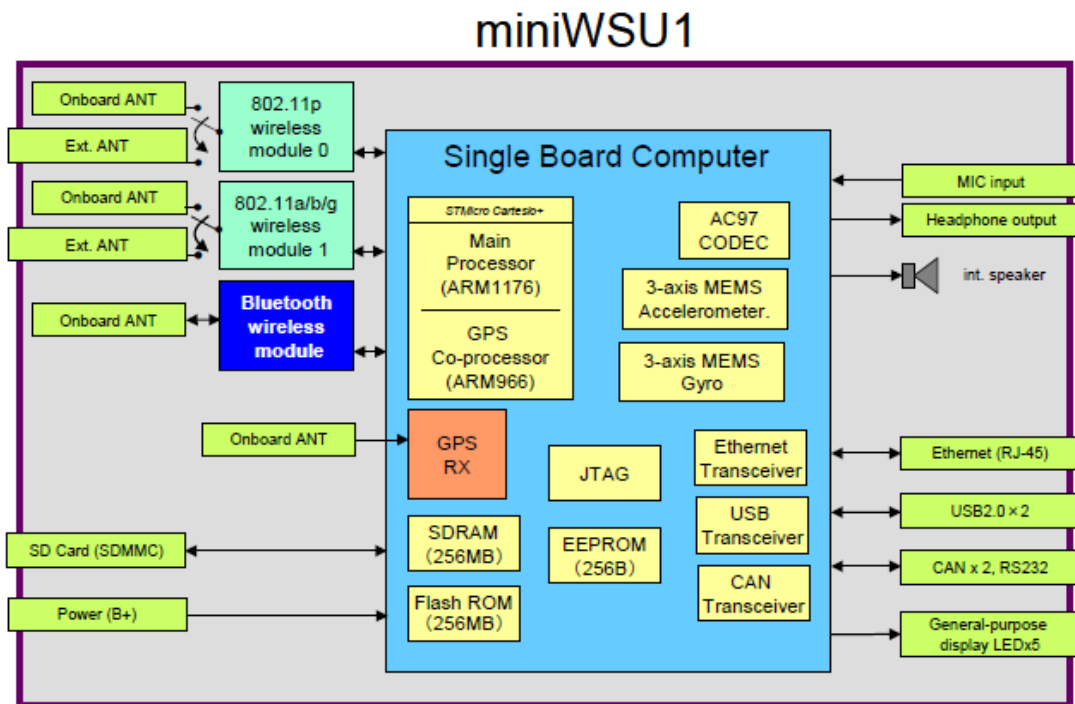


Figure 23. Construction of miniWSU1

In the project of commercial connected vehicle test for safety applications (Howe et al., 2016a, 2016b, 2016c, 2016d; Stephens et al., 2014) and the project, commercial vehicle test for retrofit safety device (LeBlanc et al., 2014a; LeBlanc et al., 2014b; Stephens et al., 2014), a particular version of WSU 1.5 was used for the project. It consists of a custom single board computer with various automotive and communications interfaces. Its size is 140 x 120 x 30 mm. A summary of the WSU 1.5 attributes is summarized in Table 16.

Table 16. Attributes of WSU 1.5

Attribute	Description
Operating Mode	WAVE (P1609.3, P1609.4, 802.11p)
Frequencies	5.85 - 5.925 GHz (ITS-RA band)
Data Rates	3 - 27 Mbps (10 MHz channels) 6 - 54 Mbps (20 MHz channels)
Transmission Output Power	10 to 18 dBm (rate-dependent), measured at antenna connector
Ambient Operating Temperature	-30 to +65 °C
Enclosure	System ground metal case
DENSO	

The WSU1.5 includes a 5.9 GHz DSRC radio, an applications processor, and a custom single board computer as a platform for V2X communications and applications research and development. Included are connections for power, ignition sense, RS-232 serial data, GPS serial NMEA and 1 pulse/second timing, 100BaseT Ethernet, two USB1.1 ports, three general purpose inputs, one general purpose output, supply ground, and two CAN2.0 ports. The internal WSU1.5 configuration file includes an element for entering the physical three-dimensional offset between the location of the GPS antenna and the geometrical center of the vehicle. Figure 24 shows two images of the device, the left details the electrical interface for the unit, the right shows a bottom view of a prototype unit with a ruler for scale.



Figure 24. DENSO WSU1.5 with automotive-grade and style connectors

2) DSRC Antennas and Receiver

The DSRC Antennas is used to send and receive BSMs. The number of DSRC antennas depends on the number of WSUs used in the V2V/V2I applications. If only one WSU is used, only two antennas are needed. In the TRP project (Burt et al., 2014a, 2014b, 2014c), since two DSRC Radios were built-in to the WSU, two DSRC Antennas were utilized. DSRC Antenna 1 is a “Whip” style antenna that mounts to the driver side mirror of the transit vehicle (Mobile Mark PN: EC012-5800). DSRC Antenna 2 is a glass mounted antenna that is mounted on the inside windshield of the transit vehicle (Mobile Mark PN: EDN137-1600), which is shown in Figure 25.

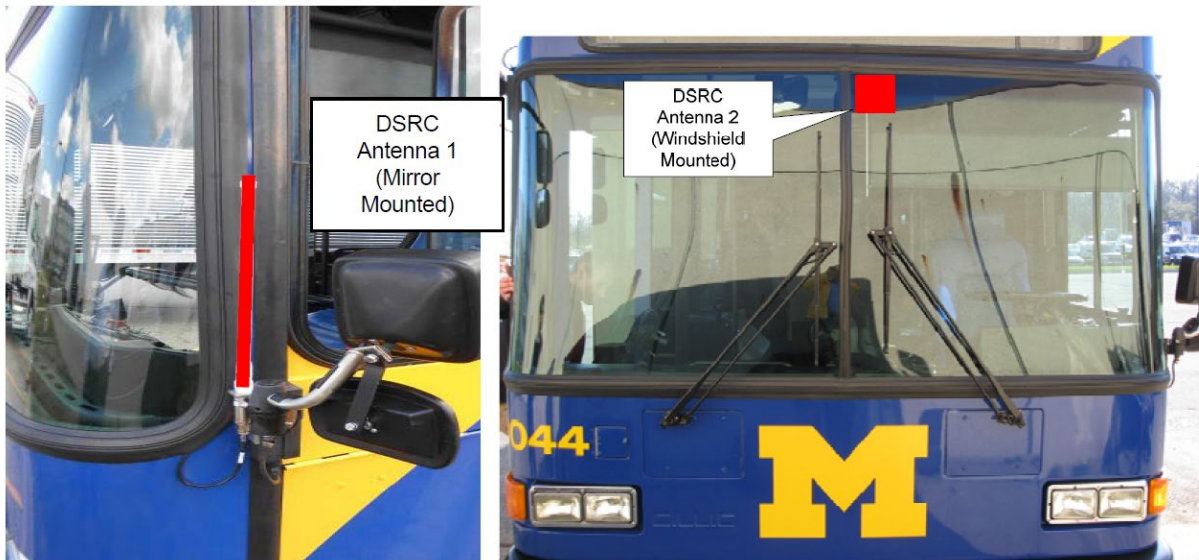


Figure 25. DSRC antenna 1 (left), DSRC antenna 2 (right)

In the project of commercial connected vehicle test for safety applications (Howe et al., 2016a, 2016b, 2016c, 2016d; Stephens et al., 2014) and the project commercial vehicle test for retrofit safety device (LeBlanc et al., 2014a; LeBlanc et al., 2014b; Stephens et al., 2014), the DGPS receiver was used, which is a Novatel model OEMV FLEXG2-V1-L1. The receiver is shown in Figure 26.



Figure 26. OBU DGPS receiver

If the V2V/V2I application uses two WSUs, there would be two DSRC antennas for the primary WSU and the secondary WSU would use a single shark-fin DSRC antenna which is identical to the two primary DSRC antennas. These two antennas were used by the primary WSU. The secondary WSU used a single sharkfin DSRC antenna, identical to those on light vehicles (Stephens et al., 2014), which was utilized in the project of commercial connected vehicle test for safety applications (Howe et al., 2016a, 2016b, 2016c, 2016d; Stephens et al., 2014) and the project commercial vehicle test for retrofit safety device (LeBlanc et al., 2014a; LeBlanc et al., 2014b; Stephens et al., 2014)

i. GPS Antenna and GPS Antenna Splitter

The GPS Antenna splitter is used to split the antenna signal to allow different hardware to have access to the GPS antenna. By using GPS Antenna splitter (for example, Mini-Circuits PN: ZAPD-2DC-S+), the number of antennas that are mounted on the exterior to the transit vehicle could be minimized (Burt et al., 2014b).

ii. Driver Display Platform

The Driver Display Platform is used to display the information processed by the on-board hardware components (i.e. WSU). In TRP project (Burt et al., 2014a, 2014b, 2014c), an android-based tablet was linked to the DSRC radio and the DAS, and this tablet included a Samsung Galaxy Tab™ 8.9 (SCH-I957) as illustrated in Figure 27. This tablet can be mounted in various configurations and included Wi-Fi® 802.11 a/b/g/n, and Bluetooth® Wireless Technology 3.0. This display unit measures 9.09 x 6.21 x 0.34 inches and contains: RAM: 1GB; ROM: 59.6GB; and 16GB Internal Memory



Figure 27. Samsung Galaxy tab (SCH-I957)

In the project of commercial connected vehicle test for safety applications and the project commercial vehicle test for retrofit safety device, the Driver Display Platform used a customized iPad2, as pictured in Figure 28, which hosted the display of visual information to the driver and supported four-channel audio. The cab's existing speakers were used to provide directional audio warnings.



Figure 28. Driver-Vehicle interface (DVI)

iii. Wireless Ethernet Interface Equipment

The Wireless Ethernet Interface Equipment allows both wireless and wired Ethernet connections to a local network on the vehicle. In TRP project, for wireless ethernet interface equipment, there is an Ethernet switch (Moxa PN: EDS-205A-T, or similar) and a wireless access point (AP) (Moxa PN: AWK-3121-US-T, or similar) installed on the transit vehicle. This allows both wireless and wired Ethernet connections to a local network on the transit vehicle.

Because different on-board units may use different power voltages, a DC/DC converter may be needed to convert vehicle power into regulated DC voltages that are used to power the safety application equipment. The DC/DC convert was utilized in Transit Safety Retrofit Package Development project. In TRP project, DC to DC Converter modules were used to convert vehicle power into regulated DC voltages that are used to power the TRP equipment. Two modules have been identified – a 12VDC to 12VDC conversion and 12VDC to 5VDC conversion.

2.4.2.2. Supporting System

1) Data Acquisition System (DAS)

Data Acquisition System (DAS) is a very important supporting system for a V2V/V2I application (Figure 29). This main function of this device is to store data from multiple source generated by the V2V/V2I application. The data includes Controller Area Network (CAN) bus data (i.e., the data transmitted in the communication networks that interconnects components inside a vehicle), video cameras, radar units, and the safety applications. A DAS usually has multiple working modes for different function usage like CPU-use and No CPU-use. A DAS is able to capture a wide range of signal frequencies, and a DAS has the ability to track multiple data source at the same time including radars, video streams, audio, GPS, and etc. The DAS is capable of recording those data at the variable sampling rates or based on the given sampling threshold or the trigger events (Burt et al., 2014b).



Figure 29. Illustration of the UMTRI GEN5 DAS (Burt et al., 2014c)

In TRP project, the DAS consists of the UMTRI GEN5 DAS. The GEN5 DAS application software runs on an UMTRI-configured version of Windows XP Embedded. The DAS can run different applications depending how the “Mode” connector is controlled by an external switch.

Automatic, Demo, Maintenance, graphical user interface (GUI), Upload, and No CPU are common modes used in testing. In “Automatic”, the DAS powers up when the ignition switch is turned on, data are collected until ignition is turned off and then the DAS powers down. In “GUI” mode, an experimenter/operator can start and stop data collection, observe a real-time display, and enter test parameters and other metadata. “Upload” triggers an automatic file transfer to a server. “No CPU” prevents the computer and peripherals from being powered and is used when a vehicle is being serviced or when no data are to be collected.

In TRP project, the DAS can capture hundreds of signals at 10 to 50 Hz, full target tracks from seven radars, five video streams, audio, GPS, and more. Data signal definitions are entered into a metadata database, using a GUI. This database is on-board the vehicle and is also available for analysis tools, enabling configurable and robust adaptation to different experiments. Video is collected onboard the DAS as well. The current system allows up to 8 separate image streams to be recorded separately, with frame size, frame rate, and compression parameters tunable to each image. Video imagery is collected by defining continuous data collection and/or triggered video collection using circular buffering, such that the triggered video (or other data) can be captured at higher rates, if desired. The DAS will, at minimum, be capable of recording up to 10 days of operational data before requiring a download. To capture data from the OBU and other onboard sensors, an existing UMTRI DAS (shown in Figure 30) from the Integrated Vehicle-Based Safety System (IVBSS) program was installed on each tractor. The DAS is approximately cube-shaped, nine inches on each side. In addition to an interface with the WSU via Ethernet switch, the DAS will also interface with the J1939 CAN bus, a suite of four cameras to capture video of the area around the tractor and inside the cab, a microphone, an external GPS and cell modem antenna,

and an independent inertial measurement unit (IMU) to capture the vehicle motion state measures of each tractor.

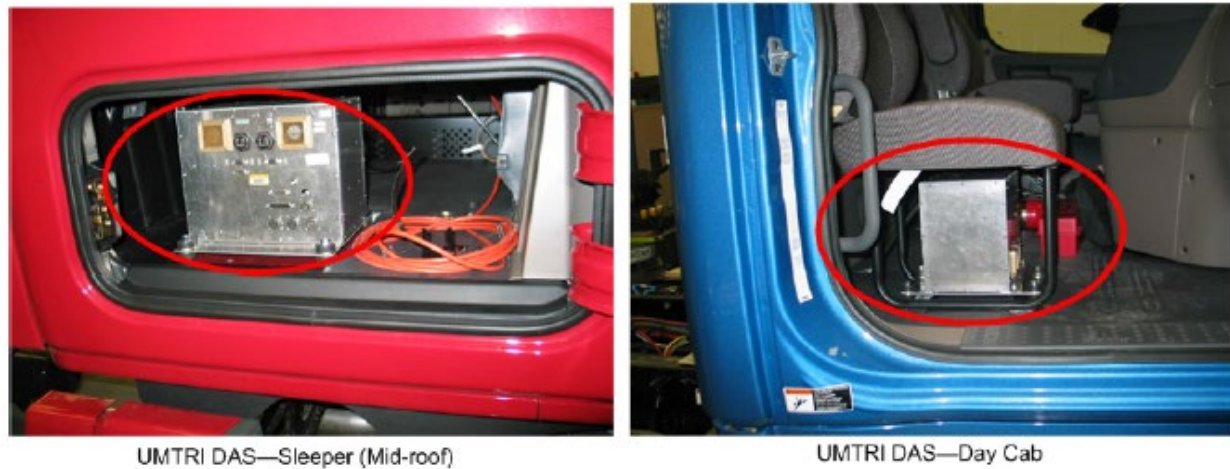


Figure 30. DAS mounted in a cargo compartment in the sleeper cabs (left) and under the passenger seat in the day cab (right)

2) Vehicle Information-- Inertial Measurement Unit (IMU)

The Inertial Measurement Unit (IMU) is a sensor that provides yaw rate, longitudinal and lateral acceleration that are used as inputs for DAS storage (Burt et al., 2014b). The interface to the IMU is a CAN bus interface. For the TRP project, a Local CAN Bus was installed to provide data to the DAS.

These IMU units were automotive grade and mounted in an enclosed weather-proof box located at the lateral center of the vehicle between the frame rails (approximately, at the center-of-gravity height of the vehicle to reduce roll and pitch effects on measured accelerations). A picture of the enclosure, mounted on an IVBSS tractor, is shown in Figure 31.

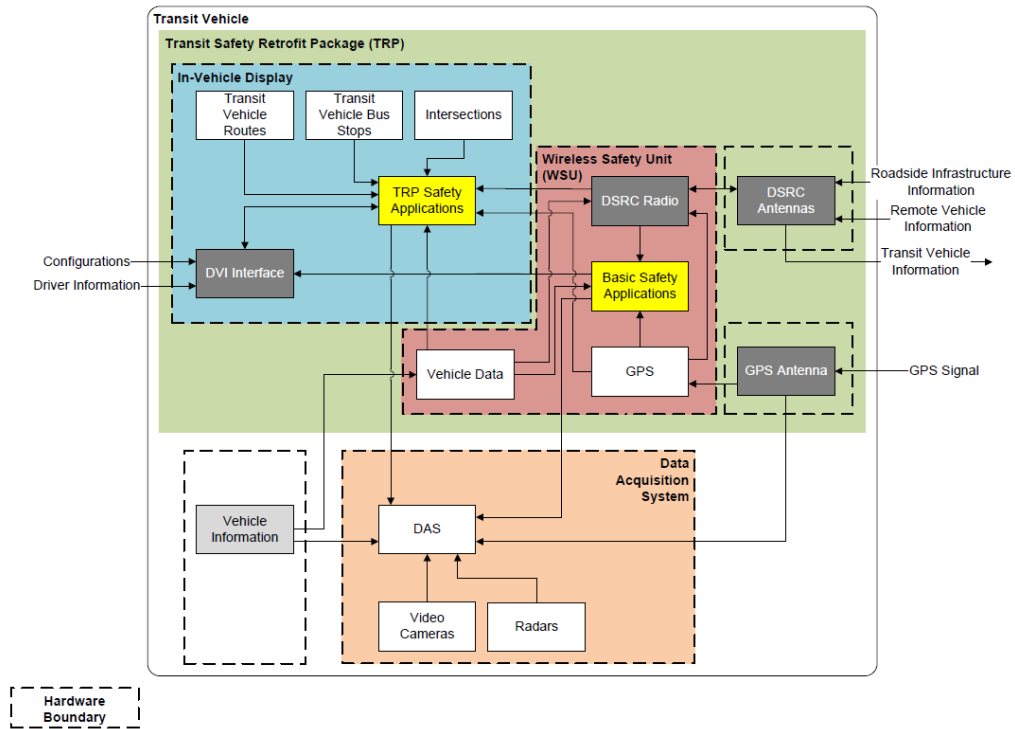


UMTRI

Figure 31. DAS IMU in sample location between frame rails behind cab on unsprung mass (LeBlanc et al., 2014a)

2.4.2.3. Overall Illustration of Relationship between On-Board Components

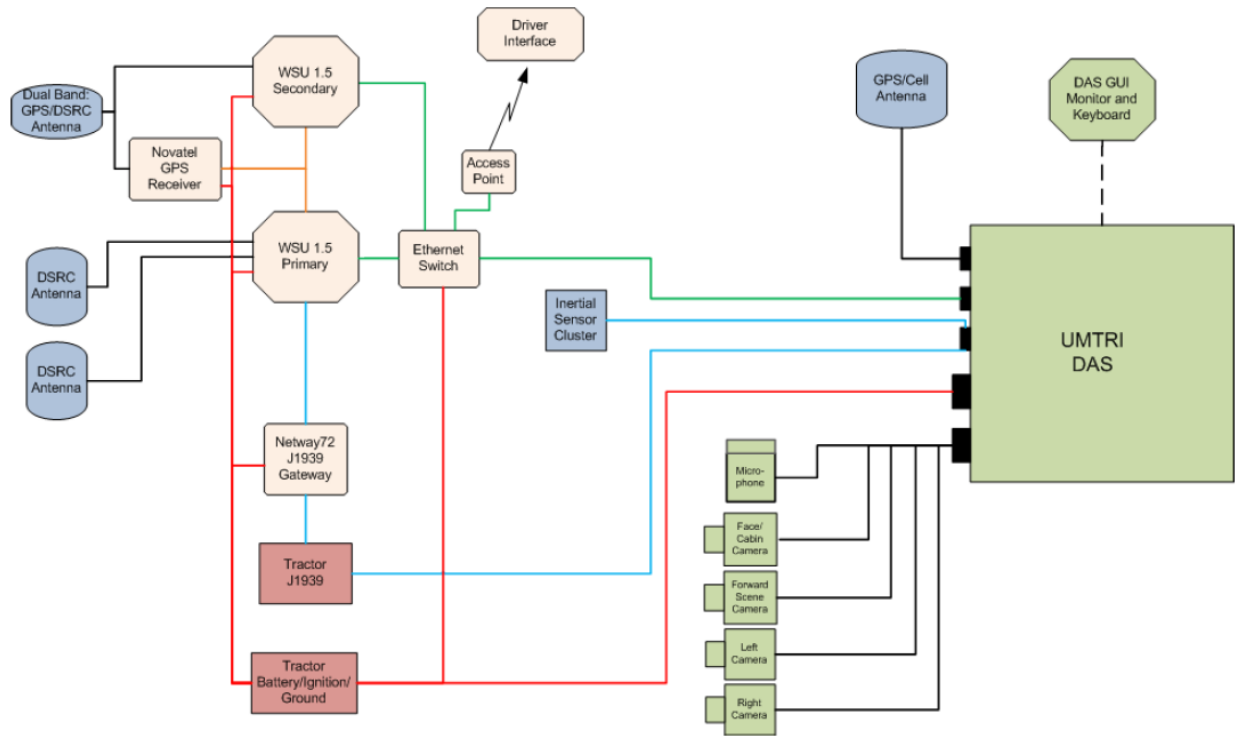
Figure 32 shows the relationship between requirements and architecture components in TRP projects. In Figure 32, the yellow-shaded architecture elements stand for the applications, and dark gray-shaded elements with white text represent the hardware. The other architecture elements are those elements pertaining to input data (e.g., transit vehicle routes) or elements that is not part of the TRP system (e.g., DAS). The TRP safety applications include pedestrian in crosswalk warning (PCW) and vehicle turning right in front of bus warning (VTRW). Basic safety applications include curve speed warning (CSW), emergency electronic brake lights (EEBL), and forward collision warning (FCW). Figure 32 clearly describes how the data is transmitted between different devices. The figure shows that the WSU is the core of the whole system and is a highly integrated system that it contains the function of generating and splitting the BSM, the function of positioning, and the function of processing data and running the safety applications.



Source: Battelle

Figure 32. Relationship between requirements and TRP architecture components (Burt et al., 2014a)

Figure 33 shows the OBE system architecture in the project of commercial connected vehicle test for safety applications. A pair of DSRC antennas broadcast and received BSMs. Communication with the host vehicle was through its J1939 data bus. Messages to be displayed by the Driver Vehicle Interface (DVI) were transmitted via Ethernet. All communication by the OBE and much information directly from the vehicle itself was stored by a Data Acquisition System (DAS) to support testing and analysis.



UMTRI

Figure 33. OBU system architecture (Stephens et al., 2014)

2.4.3. Summary

This chapter has reviewed the basic OBU applications for the connected vehicles. To ensure the success of the developed smartphone application, the mechanism of the OBU application should be fully understood and considered. The functions of OBU application for connected vehicles need to be realized by the smartphone applications.

2.5. Conclusions

In this chapter, the research team has conducted an extensive literature review of the existing studies and practices that are relevant to the smartphone sensors, data communication, OBU application, etc. Based on the findings from the reviewed practices and materials, the followings can be recommended and considered for using smartphone to emulate OBU applications:

- The viability of using smartphone for emulating the OBU application should be validated by considering data flow, latency of data flow, data utilization, impact on battery life, etc.
- Evaluation tests should be conducted to validate the accuracy of the data obtained based on smartphone sensors including travel modes, users' locations, speeds, movements, etc. The data should be refined if needed
- To provide proactive pedestrian detection, appropriate complementary sensors should be considered, and the corresponding detection and tracking methods should be referred to
- OBU applications should be fully understood and incorporated while developing the applications

CHAPTER 3. VALIDATION OF THE FEASIBILITY OF USING SMARTPHONE AS OBU

To emulate On-board units (OBUs), it is necessary to set up the smartphone data communication. As shown in Figure 34, the smartphone data communication involves three components: road users, smartphone, and cloud server. By using developed application (apps), the statuses such as locations, speeds, and movements could be collected by the smartphone sensors. Through wireless communication, the collected data could be uploaded to the server. The server could identify the potential conflicts through cloud computation and then send warning to road users through the developed apps. To realize the data communication, the research team developed the smartphone application (app) for both android and iOS systems and set up the cloud server in this task.

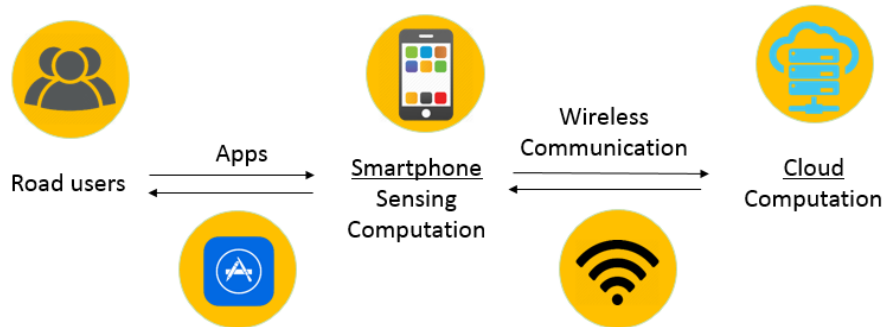


Figure 34. A framework of smartphone data communication

Section 3.1 discusses the process of apps and server development. The different sensor data and the corresponding traffic parameters are also investigated.

Section 3.2 presents the feasibility test of using smartphones as OBU emulators. In this task, the research team set up the developed app and the server for data communication from a smartphone to the cloud and vice versa. Extensive experiments are conducted to test the

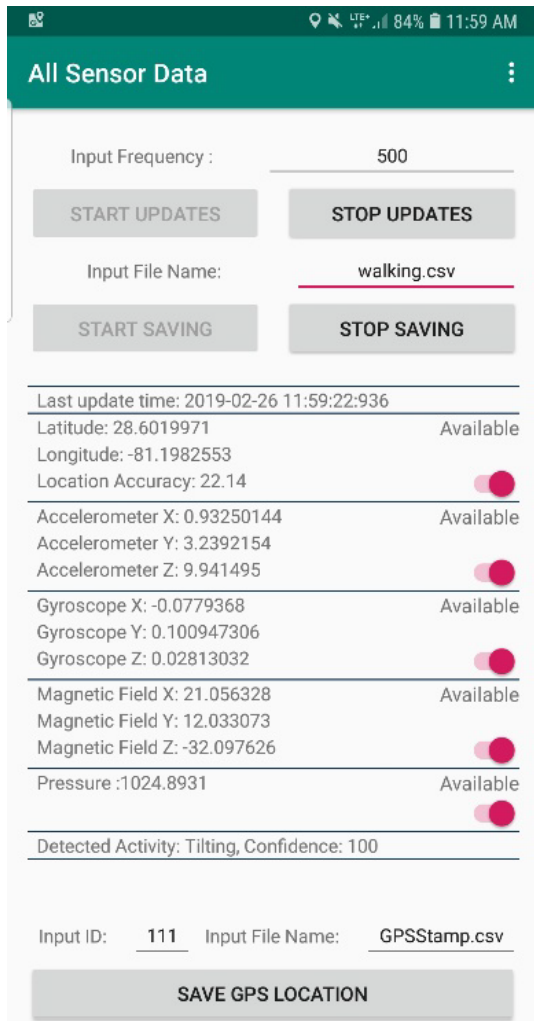
communication latency and battery consumption of the developed apps. The experiments considered the different transportation modes, speeds, sampling duration, etc.

Section 3.3 concludes the feasibility of using smartphones as OBU emulators. Meanwhile, the plans for the following tasks about validating the accuracy of smartphone data and test of complementary sensors are also presented.

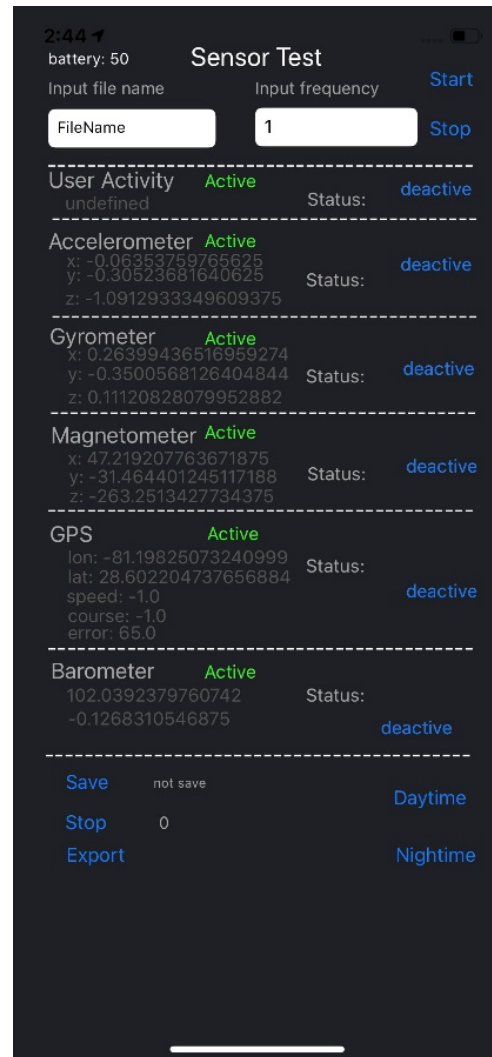
3.1. Development of Apps for Data Collection

3.1.1. Develop Application to Collect Data

There are two popular phone operation systems in the market, i.e., Android and iOS. Hence, the research team developed smartphone apps both for Android and iOS operating system to collect smartphone sensor data. The developed smartphone apps could collect sensors data for different traffic modes and then upload these data to the server. Figure 35 illustrates the developed apps of Android and iOS systems. It is shown that different sensors' data such as GPS (Global Positioning System), Accelerometer, Gyroscope, Magnetometer, and Barometer could be collected through the apps. The user can turn on/off each sensor. After collecting data, it can be saved in local storage in the smartphone and sent to the server. The user can select different sampling durations to test the data communication between the apps and the server. Depending on the sampling duration, the total amount of sensors' data is different. For example, if the sampling duration is 20ms (0.02 second), fifty samples could be recorded and transmitted to the server through the apps in one second. Thus, the higher the sampling duration value means the lower the data samples. Users can save the data samples by using 'save' options.



(a) Android App



(b) iOS App

Figure 35. Screen-shot of developed apps

3.1.2. Data Stream

Table 17 lists the sensors and the data that can be collected. Based on the data, road users' data information such as location, speed, movement, acceleration, transportation mode could be identified. For example, the GPS could directly provide road users' locations and the speed could be calculated based on the distance difference of two locations and the time interval. Based on smartphone accelerometer data, the acceleration of the vehicle could be calculated (Ferreira et al., 2017). Using gyroscope and GPS data, right/left/U-turning movements of vehicles could be

identified (Renaudin and Combettes, 2014). The vehicle's location (e.g., regular road, under pass, and fly-over) could be determined by using barometer and magnetometer data (Johnson and Trivedi, 2011). In addition to sensors data, researchers have also included the activity recognition API in Android and iOS app to detect travel mode of the app user. User activities are classified in some specific classes such as walking, bicycling, driving, etc.

Table 17. List of sensors used in apps for data collection

Sensor Name	Data Collected	Dimensions	Unit
GPS	Geographical descriptions of current location	Latitude, Longitude	Degree
Accelerometer	Acceleration	x, y, z	g-force
Gyroscope	Rotation Rate	x, y, z	radian/s
Magnetometer	Magnetic Field	x, y, z	μ T
Barometer	Ambient Air Pressure	1	hPa

All collected sensor data can be saved at the smartphone device and sent to the server. To save the sensor data in smartphone, the apps user needs to activate the saving option in the app and users can specify the data file name. Data format in the smartphone is a CSV (comma-separated-value) file while the data is saved as a JSON file. Figure 36 shows the screenshot of the collected data file for the smartphone app. In the data file, all selected sensors data are included.

Datetime	Accelerometer			Gyroscope			Magnetometer			GPS		Barometer	Mode
	x	y	z	x	y	z	x	y	z	Longitude	Latitude		
3-24-2019 11:59:05.963	0.11731084	-0.15920755	9.672159	-0.00237665	0.00100602	-4.91E-04	-97.3188	-7.12949	-145.792	-81.1982342	28.6019909	1023.614	Walking
3-24-2019 11:59:06.464	0.11012854	-0.16399577	9.676947	-0.00237665	3.95E-04	-7.96E-04	-97.1289	-7.14932	-145.834	-81.1982342	28.6019909	1023.6245	Walking
3-24-2019 11:59:06.964	0.12090199	-0.1627987	9.668568	-0.00268208	0.00100602	-7.96E-04	-97.5644	-7.54577	-145.649	-81.1982342	28.6019909	1023.61426	Walking
3-24-2019 11:59:07.463	0.11252264	-0.1627987	9.668568	-0.00298752	7.01E-04	-4.91E-04	-97.0877	-7.58041	-145.673	-81.1982342	28.6019909	1023.626	Walking
3-24-2019 11:59:07.964	0.11012854	-0.1627987	9.67575	-0.00268208	7.01E-04	-4.91E-04	-97.2309	-7.60513	-145.513	-81.1982342	28.6019909	1023.61743	Walking
3-24-2019 11:59:08.464	0.11371969	-0.16040462	9.670962	-0.00268208	7.01E-04	-1.85E-04	-97.0679	-7.51775	-145.647	-81.1982342	28.6019909	1023.6199	Walking
3-24-2019 11:59:08.966	0.11850789	-0.16160166	9.676947	-0.00268208	0.00100602	-7.96E-04	-97.2145	-7.34271	-145.387	-81.1982342	28.6019909	1023.6223	Walking
3-24-2019 11:59:09.466	0.11850789	-0.16519281	9.674553	-0.00298752	0.00100602	-1.85E-04	-96.5282	-7.54811	-145.769	-81.1982342	28.6019909	1023.62085	Walking
3-24-2019 11:59:09.966	0.11611379	-0.16040462	9.676947	-0.00237665	3.95E-04	-1.85E-04	-97.5709	-7.17851	-145.265	-81.1982342	28.6019909	1023.62695	Walking
3-24-2019 11:59:10.466	0.11252264	-0.16519281	9.664977	-0.00298752	0.00100602	-4.91E-04	-97.4243	-7.56724	-145.849	-81.1982342	28.6019909	1023.62744	Walking
3-24-2019 11:59:10.967	0.11012854	-0.16519281	9.672159	-0.00268208	7.01E-04	-7.96E-04	-97.1341	-7.48806	-145.99	-81.1982342	28.6019909	1023.61646	Walking
3-24-2019 11:59:11.468	0.11491674	-0.1627987	9.666174	-0.00298752	7.01E-04	-1.85E-04	-97.227	-7.1856	-145.853	-81.1982357	28.6019911	1023.6228	Walking
3-24-2019 11:59:11.969	0.11970494	-0.15441936	9.676947	-3.63E-05	3.11E-04	2.92E-04	-97.0635	-7.70202	-145.744	-81.1982357	28.6019911	1023.62354	Walking
3-24-2019 11:59:12.468	0.11371969	-0.15681347	9.668568	2.69E-04	-0.001216	2.92E-04	-96.7968	-7.05815	-145.193	-81.1982357	28.6019911	1023.6328	Walking
3-24-2019 11:59:12.969	0.11611379	-0.16160166	9.678144	2.69E-04	-6.05E-04	-3.19E-04	-97.5404	-7.15971	-145.486	-81.1982357	28.6019911	1023.6321	Walking
3-24-2019 11:59:13.470	0.11731084	-0.15920755	9.67575	2.69E-04	3.11E-04	-3.19E-04	-97.3377	-7.51857	-145.509	-81.1982357	28.6019911	1023.62695	Walking
3-24-2019 11:59:13.970	0.11491674	-0.16160166	9.668568	-3.63E-05	5.73E-06	2.92E-04	-96.8167	-7.0728	-145.45	-81.1982357	28.6019911	1023.6266	Walking
3-24-2019 11:59:14.470	0.11850789	-0.16160166	9.670962	-3.42E-04	-3.00E-04	-3.19E-04	-97.1286	-7.47529	-145.589	-81.1982357	28.6019911	1023.6277	Walking
3-24-2019 11:59:14.970	0.11850789	-0.16040461	9.67934	-3.63E-05	3.11E-04	-3.19E-04	-96.9733	-6.97186	-145.681	-81.1982357	28.6019911	1023.6255	Walking
3-24-2019 11:59:15.471	0.10893149	-0.16279872	9.669765	-3.42E-04	5.73E-06	2.92E-04	-96.9176	-6.95767	-145.091	-81.1982357	28.6019911	1023.61694	Walking
3-24-2019 11:59:15.972	0.11611379	-0.16040462	9.673356	-3.63E-05	5.73E-06	2.92E-04	-97.0388	-7.65827	-145.211	-81.1982357	28.6019911	1023.6245	Walking
3-24-2019 11:59:16.473	0.12449314	-0.16040462	9.674553	2.69E-04	3.11E-04	-3.19E-04	-97.1966	-7.27885	-145.536	-81.1982357	28.6019911	1023.6233	Walking
3-24-2019 11:59:16.974	0.12329608	-0.15322231	9.675751	2.69E-04	-3.00E-04	-3.19E-04	-97.162	-6.92077	-145.681	-81.1982357	28.6019911	1023.6426	Walking
3-24-2019 11:59:17.474	0.11012854	-0.15920757	9.673356	-3.63E-05	3.11E-04	2.92E-04	-97.0896	-7.05578	-145.66	-81.1982357	28.6019911	1023.6465	Walking
3-24-2019 11:59:17.974	0.11850789	-0.16040462	9.67575	-3.63E-05	-3.00E-04	2.92E-04	-97.4574	-7.64845	-145.56	-81.1982357	28.6019911	1023.6299	Walking
3-24-2019 11:59:18.474	0.11611379	-0.15681346	9.67934	2.69E-04	-3.00E-04	-3.19E-04	-96.7809	-7.53266	-145.613	-81.1982357	28.6019911	1023.6289	Walking
3-24-2019 11:59:18.974	0.11491674	-0.16160166	9.673356	-3.63E-05	3.11E-04	-1.34E-05	-97.2759	-7.02209	-145.376	-81.1982357	28.6019911	1023.6338	Walking
3-24-2019 11:59:19.474	0.12090199	-0.15801051	9.67934	2.69E-04	-3.00E-04	-1.34E-05	-96.982	-6.89706	-145.428	-81.1982357	28.6019911	1023.6272	Walking
3-24-2019 11:59:19.974	0.11731084	-0.15801051	9.670962	-3.42E-04	5.73E-06	-3.19E-04	-96.8509	-7.31903	-145.811	-81.1982357	28.6019911	1023.6323	Walking
3-24-2019 11:59:20.474	0.11731084	-0.16399576	9.678144	-3.63E-05	3.11E-04	2.92E-04	-97.4428	-7.36893	-145.67	-81.1982357	28.6019911	1023.64355	Walking
3-24-2019 11:59:20.975	0.11850789	-0.15801051	9.669765	2.69E-04	5.73E-06	-1.34E-05	-96.6937	-7.11053	-145.668	-81.1982357	28.6019911	1023.63306	Walking
3-24-2019 11:59:21.479	0.10773444	-0.15801051	9.673356	2.69E-04	3.11E-04	-1.34E-05	-96.7223	-7.5418	-146.015	-81.1982316	28.6019886	1023.63184	Walking
3-24-2019 11:59:21.978	0.12449313	-0.14843412	9.674553	-0.00156343	0.00306005	2.92E-04	-97.3507	-7.19544	-145.467	-81.1982316	28.6019886	1023.6367	Walking
3-24-2019 11:59:22.478	0.11850789	-0.16399577	9.678144	-3.42E-04	5.73E-06	-3.19E-04	-96.692	-7.507	-146.313	-81.1982316	28.6019886	1023.635	Walking
3-24-2019 11:59:22.979	0.10773444	-0.16160166	9.668568	-3.42E-04	3.11E-04	-3.19E-04	-97.1277	-7.31575	-146.279	-81.1982316	28.6019886	1023.6362	Walking
3-24-2019 11:59:23.479	0.10893149	-0.15920757	9.678144	-3.63E-05	3.11E-04	-3.19E-04	-97.1191	-7.22478	-145.992	-81.1982316	28.6019886	1023.63403	Walking

Figure 36. Screenshot of data file generated by the smartphone app

3.1.3. Summary

In this chapter, the development of smartphone apps and the data stream generated by the apps were discussed. The developed apps are able to read, upload, and save sensors' data. Data of five sensors including GPS, accelerometer, gyroscope, magnetometer, and barometer could be collected. The traffic parameters corresponding to different sensor data were investigated.

3.2. Feasibility Test

3.2.1. Demonstration of Data Communication

The server was set up for collecting data from the smartphone apps. Through an Application Programming Interface (API), the apps could upload the collected data to the server and download data from the server. An option of changing sampling duration was set up in the app to verify the effects of sampling duration on the battery consumption and communication loads for the apps. The Android app could be connected to the Android Studio to observe the data communication between the smartphone app and the server. As shown in Figure 37, two signals could be observed: yellow signal representing the data communication from smartphone to the server and blue signal for the acknowledgment signal from the server to a smartphone. After receiving a data packet, the server sends an acknowledgment signal to the server. Hence, two signals (data: apps to server, acknowledgment: server to apps) are shown in the figure for data communication. Figure 38 indicates that the iOS app is connected to the iOS profiler. It could be observed that the smartphone apps could successfully upload data to the server and receive data from the server. In other words, both the Android and iOS apps could successfully communicate data with the server.

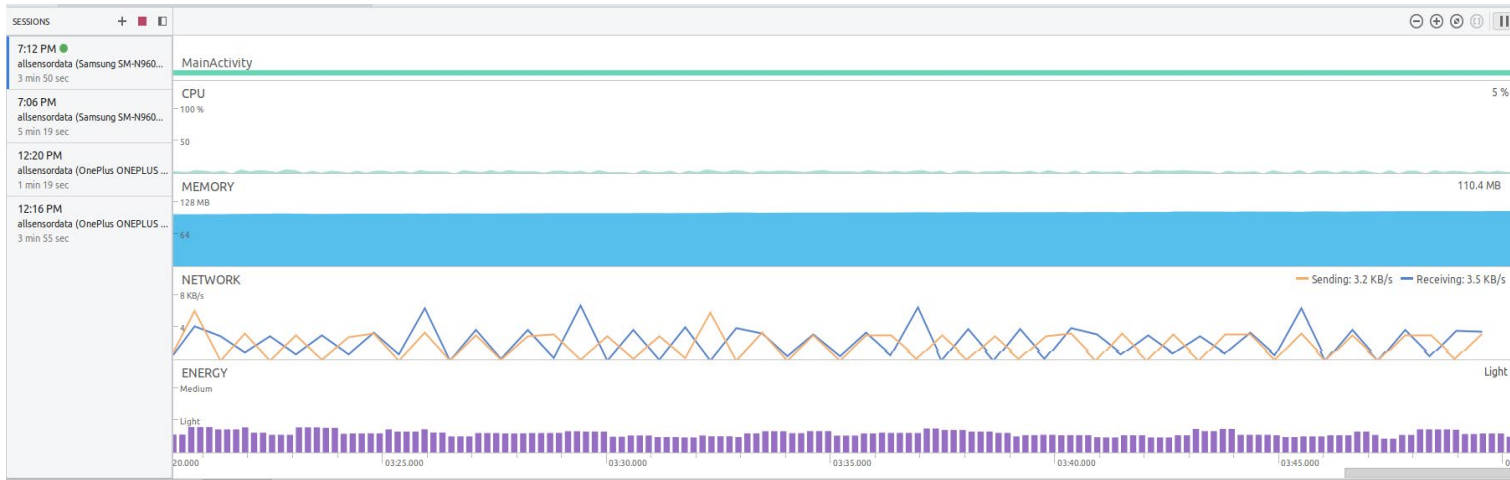


Figure 37. Visualization of data communication (through Android studio) for Android

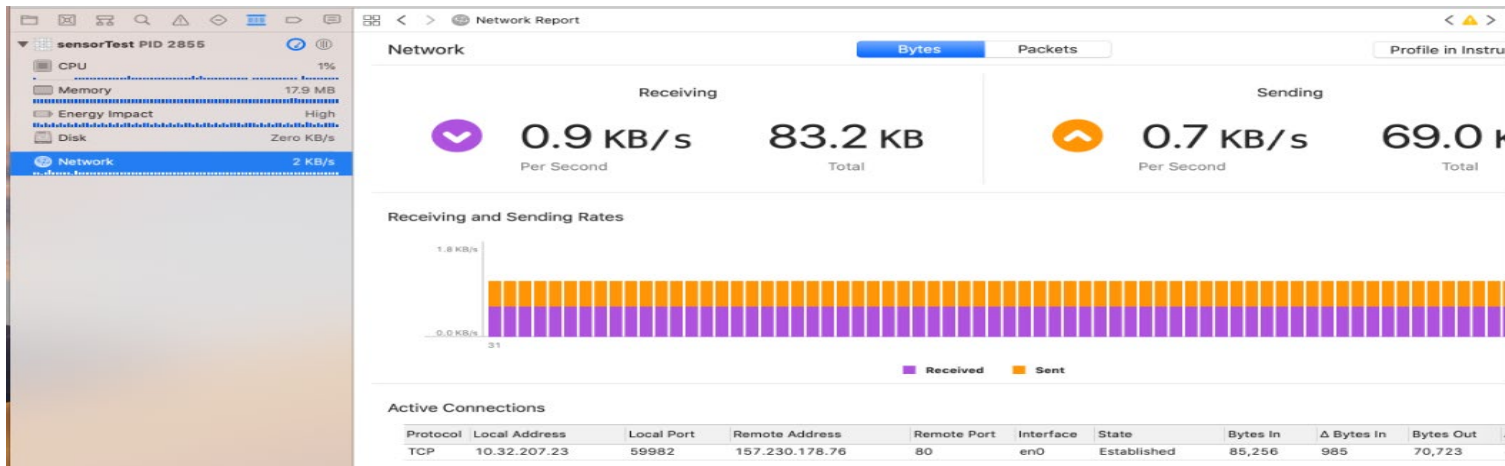


Figure 38. Visualization of data communication for iOS

3.2.2. Latency of Data Transmission

The data transmission involves two processes: (1) transfer data from smartphones to the server (uploading data); (2) obtain data from the server to the smartphone (downloading data). It is required to analyze the latency of data transmission for the two processes. Latency is an important indicator to measure the performance of a communication system, which basically depends on the communication delay. It should be noted that the information of each user will be uploaded from the app to the server in real time. Meanwhile, the server will send the warning message to the related users if a potential conflict is identified. Through the round-trip communication is tested for the same user, the round-trip latency could represent two cases: (1) uploading the app data and receiving the warning message by the same user; (2) uploading the app and receiving the warning message by different users. The overall communication delay for any communication technology is divided into four sections: transmission delay, propagation delay, queuing delay, and processing delay (Martin et al. 1997).

Transmission delay depends on the data packets lengths and the transmission rate of communication technology. Depending on communication technology, transmission delay can be different. Since our apps are designed for drivers and pedestrians who stay outside, the most likely communication technology used by the app user will be cellular communication (outside). Hence, delays with 4G (LTE) communication was investigated for both walking and driving statuses.

Propagation delay depends on the distance between transmitter and receiver, and propagation speed. For wireless communication technology, data packets are multiplexed with an electromagnetic wave, which is a carrier signal of gigahertz frequency in the speed of light. Hence, propagation delay is a minimum delay in the overall communication delay which could be ignored.

Queuing delay depends on the duration that a packet waits in the queue to be executed. Queuing delay is related to the data volume (communication load). Large data volume would increase the queue length and hence increases the queuing delay. Data volume depends on several issues such as the number of sensors used in the app, sampling duration, computational load at the a As the size of all sensor data is very small (approximately 3.2KB/s), only the effect of sampling frequency was investigated.

Processing delay is the delay introduced at the transmitter and intermediate nodes for multi-hop communication. Depending on the communication technology and processor design, processing delay can be different. This task experimented the latency test for two phones: Samsung Note9 (Android smartphone) and iPhone XS (iOS smartphone). The two phones can be regarded as the most advanced smartphones currently in the market.

3.2.2.1. Experimental Design

Two parameters including sampling duration and travel mode were investigated by using both the Samsung and iPhone for the experiment. Four sampling durations (i.e., 100 millisecond (ms), 300ms, 500ms, and 1,000ms) were investigated for the latency test and two different travel modes were investigated: walking and driving a vehicle. Four different constant velocities have been tested for the driving scenario: 20 mph, 30 mph, 40 mph, and 50 mph. During the latency test experiments, all the sensors were activated.

3.2.2.2. Experimental Results

The experiment was conducted by using Samsung Note9 for Android app and iPhone XS for iOS app. In each experiment, one thousand data samples were collected and each experiment was conducted twice. The average uploading, downloading and total latencies were calculated for

each case. In two travel modes, total walking duration for each application (Android/iOS) was more than two hours and total driving distance was approximately 180 miles.

The uploading, downloading, and round-trip durations for walking and driving modes are shown in Tables 18 and 19. The uploading duration is the time difference between the starting time to upload data from the app and the receiving time of data in the server. The downloading duration is the time difference between the data generating time at the server and the receiving time at the app. Round-trip duration is the summation of uploading and downloading durations. Several observations could be made from the results presented in the two tables. First, there is no significant effect of sampling rate, transportation mode, and speed on the latency. The round-trip latency under different conditions for the two smartphones ranges from 85ms to 150ms. Second, the uploading latency is longer than the downloading latency for the iPhone while the opposite result could be found for the Samsung. For different sampling durations and transportation modes, the average uploading latencies of iPhone are between 50ms and 100ms and the average downloading latencies are between 27ms and 35ms. For Samsung, the average uploading latencies are between 28ms and 33ms while the average downloading latencies are between 75ms and 130ms. The opposite performance of Samsung (Android) and iPhone (iOS) is because of the different communication design and developing environment of the a Finally, for most cases, iPhone has shorter latency compared to Samsung.

Latency requirement for V2X (vehicle-to-everything) for safety is between 100ms and 1 second (Deyet al. 2016; Xu et al. 2017). Hence, the average 4G LTE communication latency range of the developed application could generally meet the latency requirement. Meanwhile, it is expected that the latency will be drastically reduced with 5G communication technology, which is an imminent communication technology (Shah et al., 2018).

Table 18. Uploading, downloading and round-trip duration for the walking mode

Sampling duration	Walking - iOS			Walking -Android		
	Uploading Latency (ms)	Downloading Latency (ms)	Round-trip Latency (ms)	Uploading Latency (ms)	Downloading Latency (ms)	Round-trip Latency (ms)
100ms	114.89	31.85	146.73	32.01	79.93	111.94
300ms	106.99	29.32	136.31	29.30	95.21	124.52
500ms	62.68	32.71	95.39	29.08	101.38	130.46
1000ms	61.07	29.88	90.95	29.81	104.72	134.53

Table 19. Uploading, downloading, and round-trip duration for the driving mode

Speed	Sampling duration	In Vehicle - iOS			In Vehicle - Android		
		Uploading Latency (ms)	Downloading Latency (ms)	Round-trip Latency (ms)	Uploading Latency (ms)	Downloading Latency (ms)	Round-trip Latency (ms)
20mph	100ms	62.81	34.27	97.08	30.55	98.82	129.37
	300ms	85.78	34.42	119.21	31.2	101.674	133.58
	500ms	72.09	31.19	103.29	31.56	106.001	137.56
	1000ms	78.53	30.87	109.39	31.12	111.48	142.6
30mph	100ms	74.64	29.09	103.73	32.13	87.41	119.54
	300ms	59.97	28.26	88.23	30.95	92.23	123.18
	500ms	69.28	27.73	97.01	30.96	91.42	122.02
	1000ms	72.44	29.13	101.57	31.31	122.805	154.11
40mph	100ms	57.188	28.69	85.87	31.42	80.145	111.56
	300ms	60.36	30.72	91.08	31.54	91.47	123.02
	500ms	59.74	31.28	91.03	30.29	107.6	137.89
	1000ms	61.22	29.76	90.98	28.97	113.48	142.49
50mph	100ms	63.63	27.68	91.31	32.32	91.11	123.42
	300ms	61.17	27.89	89.06	31.32	100.49	132.01
	500ms	60.47	28.29	88.76	32.04	103.2	135.24
	1000ms	69.84	28.18	98.02	30.46	111.79	142.25

3.2.3. Battery Consumption

Battery consumption by the technology and hardware in the smartphone is also a key factor in the design of new service and application. Both sensing and processing data require a significant level of energy (Carroll and Heiser, 2010). The selection of sensors and methods to optimize sensing using suitable sampling frequency to leverage battery consumption and system performance should be considered in the smartphone-based application. Data processing energy can be reduced using effective and optimized algorithms (Lane et al. 2013). The detailed specification of both Samsung Note9 and iPhone XS is shown in Table 21. The Apple company didn't report the battery capacity of iPhone XS. According to the test of other companies (BGR Media LLC, 2018; Techradar, 2018), the battery capacity of iPhone XS is 2,658 mAh. It is reported by Samsung that Note9 has larger battery of 4,000 mAh, compared to the iPhone XS.

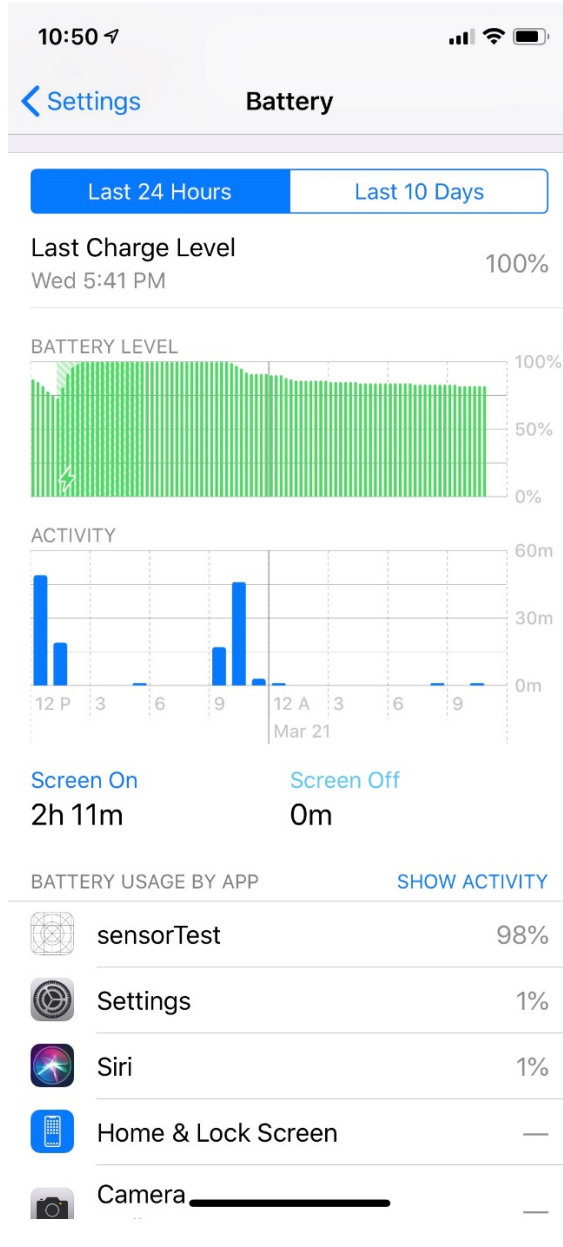
Table 20. Specifications of Samsung Note 9 and iPhone XS

Specifications	Samsung Note9	iPhone XS
Model	SM-N960U1	MT942LL/A
Battery Capacity	4,000mAh	2,658mAh
Capacity	128 GB	64 GB
Screen	FHD+(2220x1080)	1125 x 2436 pixels, 19.5:9 ratio (~458 ppi density)
Memory	6 GB	4 GB

Figure 6 presents screenshots of battery consumption status for both Samsung Note9 and iPhone XS. As shown in Figure 6, Samsung Note9 provides battery consumption at the mAh level. Meanwhile, iPhone only provides the percentage of battery consumption and the percentage is only at the integer level. The detailed estimation of battery consumption for Samsung could be collected in mAh and no accurate battery consumption for iPhone could be collected.



(a) Samsung Note9



(b) iPhone XS

Figure 39. Screen-shot of battery consumption status

3.2.3.1. Experimental Design

The experimental design can be described in two cases. First, battery consumption by the app was calculated for different sampling rate with activating all sensors. Ten different sampling durations from 100ms to 1000ms in 100ms increments. Second, battery consumption by smartphone app for different number of sensors was estimated (keeping a constant sampling rate). In the second case, several scenarios have been considered: (1) a single sensor was in activated mode. Five sensors were tested separately. (2) all sensors were activated. (3) no sensor was activated to test the pure battery consumption of the smartphone. Hence, totally seven different scenarios were included for the second case. The test for each scenario was conducted twice and one hour for each.

3.2.3.2. Experimental Results

In Figure 40, battery consumption by the Android Application for different sampling rate is shown. It is indicated that battery consumption slightly decreases with the increase of sampling durations. For different sampling durations, the battery consumption ranges from 249 mAh per hour to 273 mAh per hour, lower than 7% of the total batter capacity.

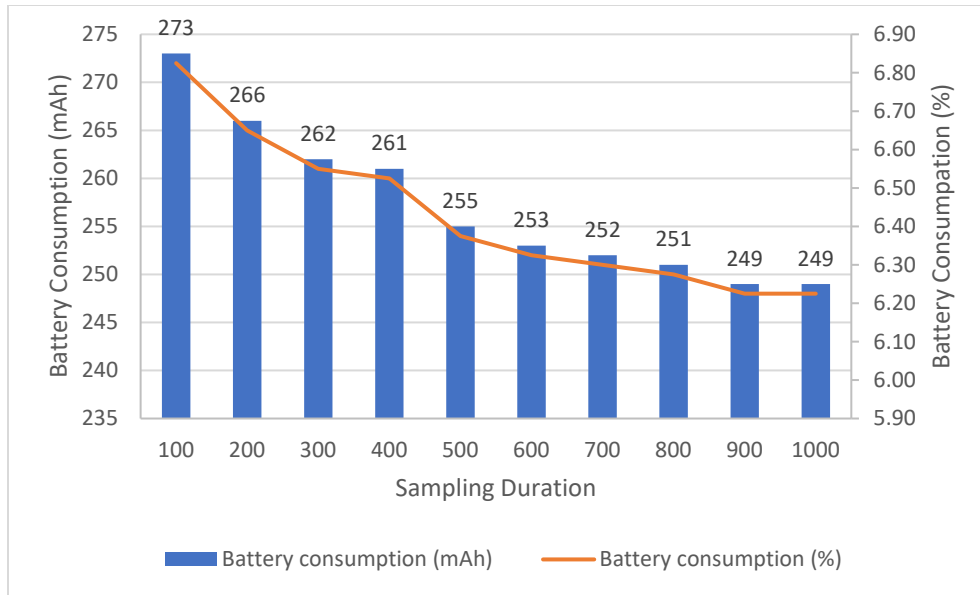


Figure 40. Battery consumption for Android for different sampling rate

Figure 41 shows battery consumptions for different sensors with the sampling duration of 500ms. The battery consumptions were 194mAh per hour by the app itself and was 255mAh per hour with activating all sensors. Among different sensors, GPS (Global Positioning System) consumes highest battery energy. Battery consumptions by other sensors could be negligible (less than 10 mAh).

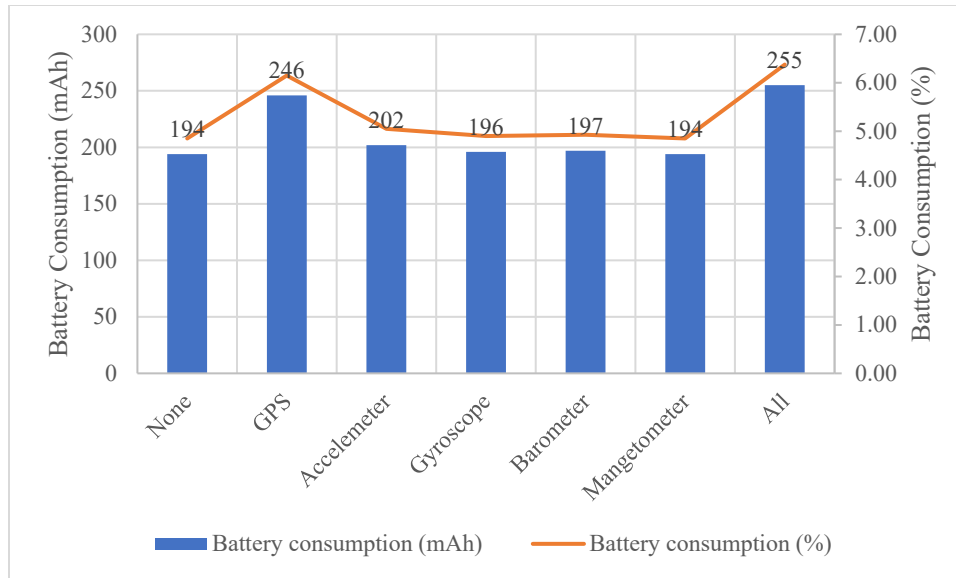


Figure 41. Battery consumption for different sensors

For the iPhone, the battery consumption with all sensors activated was around 12% if the sampling duration was lower than 500ms and the app with the sampling duration equal to and over 500ms would consume around 11% of the battery. Meanwhile, the app itself consumed around 10% of the battery.

It could be concluded that the smartphone could have sufficient battery capacity for using the developed apps to emulate the OBU.

3.2.4. Summary

In this chapter, data communication capabilities of the applications, latency requirement for the communication, and the battery consumptions for the application were investigated. The data communication between the developed apps and the server was set up. The data communication latency was investigated by considering the sampling duration, transportation mode (walking and driving), speed for both Android and iOS systems. The experiment results suggested that the latency of data communication could meet the latency requirement of V2X applications. Experiments were conducted to test the battery consumption by the developed apps with different

sensors activated. It was found that the battery consumptions per hour of the developed apps was lower than 7% of the total battery capacity for Samsung and around 10% for iPhone. It indicated that the smartphone could have sufficient battery capacity for using the developed app. The GPS consumes the highest battery compared to other sensors.

3.3. Conclusions

In this chapter, the research team has developed smartphone applications (apps) for both Android and iOS operating systems. The developed smartphone apps could collect smartphone sensor data. The data of five sensors including GPS, accelerometer, gyroscope, magnetometer, and barometer were included in the developed apps. The collected data could be used to obtain transportation related information of users including transportation mode, location, speed, movement, etc. The server was set up to realize the communication between the developed apps and the server. Extensive experiments were conducted to test the latency of data transmission and battery consumption of the developed apps for Samsung (Android smartphone) and iPhone (iOS smartphone). The results suggested that the latency of data communication could meet the requirement of V2X applications. Besides, both smartphones could have enough battery capacity for using the developed apps. Hence, it could be concluded that it is feasible to use the smartphone to emulate the OBU.

CHAPTER 4. VALIDATION OF THE DATA ACCURACY AND REFINEMENT OF THE DATA STREAM

Smartphone data communication involves three components: road users, smartphones, and the cloud server. By using developed applications (apps), the smartphone data such as Global Positioning System (GPS) coordinates, speed, and accelerometer can be collected and uploaded to the server. Smartphone data can be used to compute traffic parameters such as locations, speeds, and movements associated with. To emulate the OBU (on-board unit) applications, traffic parameters need to accurately reflect the users' statuses. This task attempted to compute different traffic parameters based on the smartphone data and validate the accuracy.

Section 4.1 presents the architecture design of the cloud server and smartphone apps. The designed architectures can better use the smartphones to emulate the OBU applications. The smartphone sensor data are also discussed.

Section 4.2 presents extensive smartphone data analysis and evaluation. Computation methods were proposed to obtain six different traffic parameters, including position, speed, localization, transportation mode, acceleration, and movement. The methods were programmed into either smartphone apps or the cloud server. Multiple experiments were conducted to evaluate the obtained traffic parameters.

Section 4.3 concludes the smartphone data analysis results and the developed smartphone apps cloud server. It suggested that the computed traffic parameters could well reflect the user status. The following tasks about emulating the OBU with the computed traffic parameters were also presented.

4.1. Development of Cloud Server and Apps

To emulate the OBU applications by using smartphones, the research team set up the server and developed smartphone apps. The architectures of the cloud server and apps were well designed to realize different functions such as data communication, data filtering, and computation.

4.1.1. Architectures of Cloud Server and Apps

4.1.1.1. Architecture of Cloud Server

The research team set up the cloud server with DigitalOcean. It is a Django webserver application which is a python-based framework. The main reason to proceed with a python server is that it is more convenient for data processing and machine learning methods. The delay associated with this type of server is also within the tolerable limits for P2V (Pedestrian-to-Vehicle) and I2V (Infrastructure-to-Vehicle) applications as was discussed in Chapter 3. As shown in Figure 42, the server has three main components including Django Application, databases, and algorithms. The description of each is mentioned below.

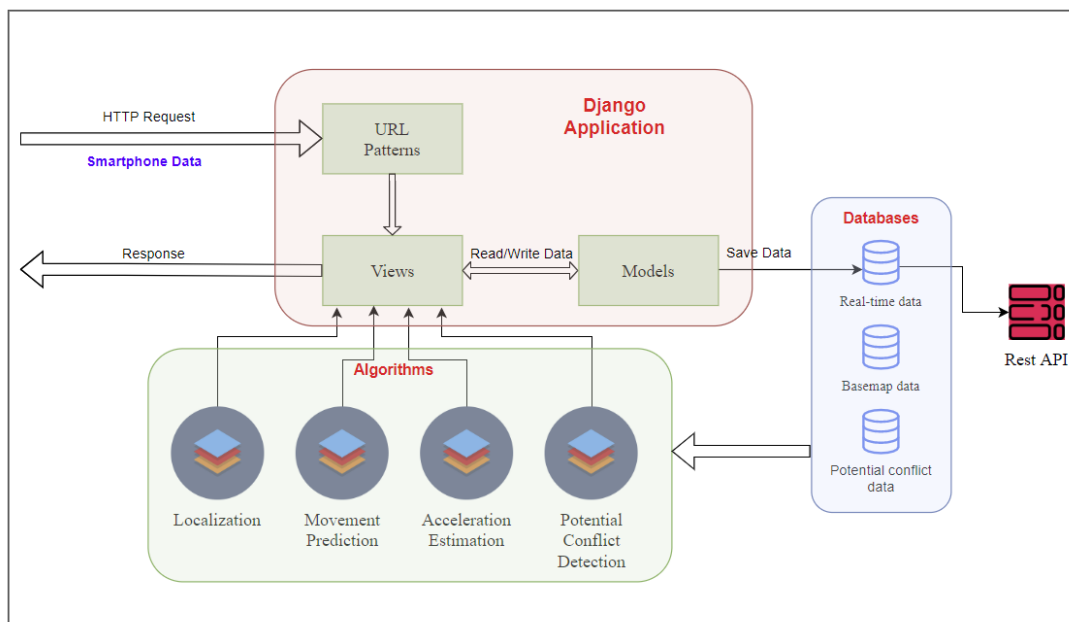


Figure 42. Server architecture

(1) Django Application

This is the core of any Django-based webservers. It processes data upload requests from smartphones and transfers the requests to appropriate Views. The Views are the place where the server decides what sort of data is coming in and where to save it in the database. This is then forwarded to Models. This is middleware and data handler between the data and database. Django provides this middle layer so that the developer can write/read data from the database without explicitly interfering with it.

(2) Databases

The databases associated with the Django Application is in PSQL format which is a SQL database. The real-time data coming from the smartphone is saved in one database. Base map database contains information of roads, intersections, curves and crosswalks while the potential conflict database contains information about the interference between pedestrians and drivers.

(3) Algorithms

This is the main part of the server where data is filtered, potential conflicts are identified, and the warning messages are sent back to the smartphone. The warning algorithms will be developed in the future tasks.

In addition, the research team has also established a Rest API. This allows the team to query data from the database and perform analysis of the data. In the future, this can be converted into a public API so that other users can benefit from this data as well.

4.1.1.2. Architecture of Smartphone Apps

The research team developed apps to emulate OBU applications. The environment used is the latest version available for the development platforms. Figure 43 shows the main components

of the developed smartphone including activity files, XML files, database, sensors, and data processing.

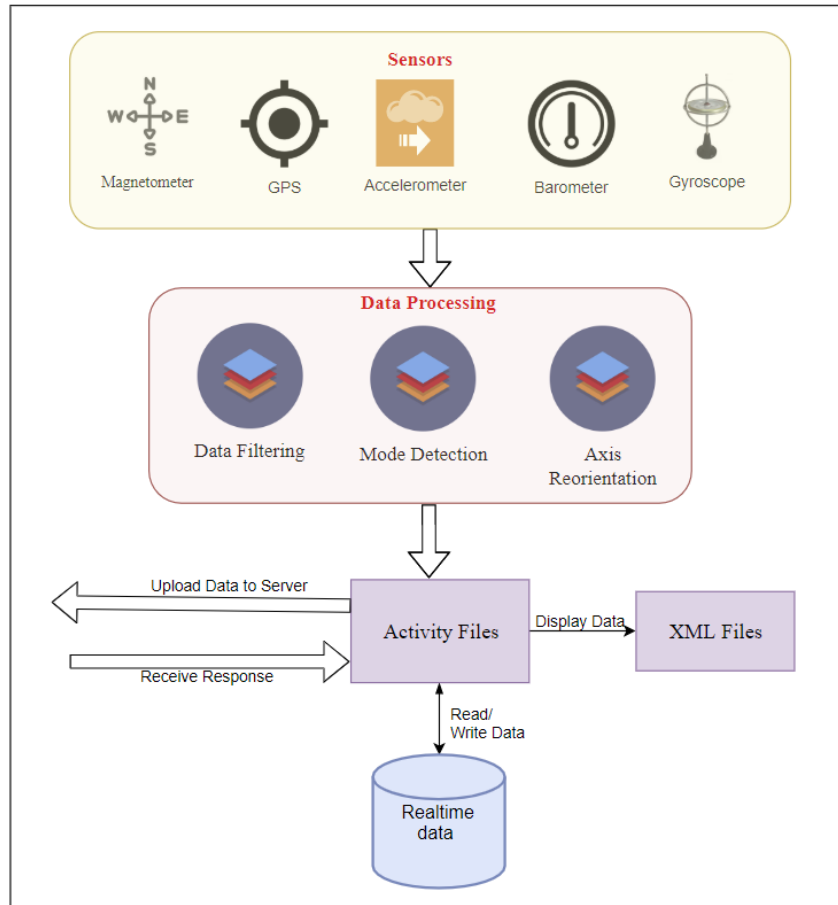


Figure 43. Mobile application architecture

(1) Activity Files

This is the core of any mobile apps. An Activity is the place which holds all the codes needed to run the app. In fact, some of the data processing and analysis is also carried out inside an Activity file. This is essentially the backend of the application. Each Activity is associated with an XML file.

(2) XML Files

While most codes run in an Activity, the developer needs to decide what to display to the users of the apps. This is accomplished with the help of XML files. It can also be referred to as the frontend of the application.

(3) Databases

There is also a database in the mobile app which stores the different data that can be useful for the application.

(4) Sensors

Almost every smartphone has a collection of various sensors. With the development of technology, these sensors have been becoming more and more accurate as well. The research team identified different sensors needed for the project, including GPS, magnetometer, accelerometer, barometer, gyroscope, etc. The data of these sensors will be introduced in the following section.

(5) Data Processing

These processes are usually carried out within an Activity. The raw sensor values could not directly reflect users' conditions. To understand smartphone users' different statuses such as locations, speeds, transportation modes, movements, different algorithms were proposed in this task to modify, filter, and fuse different sensor data. The data processing techniques will be discussed in Chapter 3.

After all the processes, the data is now ready to be uploaded to the server. The mobile app creates multiple threads while uploading the data. This is important because data does not have

to wait in a queue. Whenever there is a free thread, data can be uploaded to the cloud server and hence the communication latency can be reduced.

4.1.2. Overview of Smartphone Sensor Data

A smartphone is equipped with multiple sensors, including motion sensors, environmental sensors, position sensors, and connection sensors. In this report, the primary sensors used for traffic parameters estimation are accelerometer, gyroscope, GPS, and magnetometer.

(1) Accelerometer

Accelerometer sensors measure the acceleration of smartphone on different axes with the unit of m/s^2 (Figure 44). Since the sensor can represent the motion of a subject, the accelerometer has been widely used in driving movement detection, activity recognition, etc. (Paefgen et al., 2012, Vlahogianni and Barmounakis 2017). For example, if a vehicle makes hard acceleration, it will be reflected by an abrupt increase of the acceleration reading of y axis.

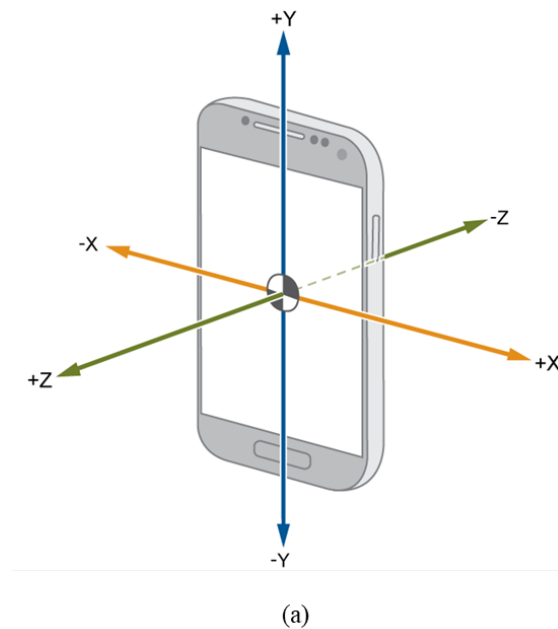


Figure 44. Accelerometer of a smartphone

(2) Gyroscope

Gyroscope sensor reflects smartphone's rotation rate on three axes with the unit of rad/s (Figure 45). Gyroscope is helpful in the navigation applications as well as some smartphone games which use the rotation data (Su, 2017). At the same time, it can also be used to detect vehicle's movements, which will be elaborated on in the following section.

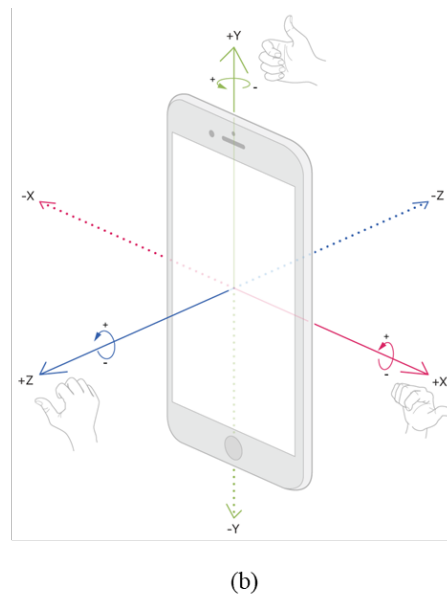


Figure 45. Accelerometer and gyroscope of smartphone

(3) Magnetometer

Magnetometer estimates magnetic field at a given point on Earth. In this report, it was mainly used to get the angle between the horizontal component of the magnetic field and the magnetic north, in degrees.

(4) GPS

GPS sensor can provide location information of the device. From the sensor, the data including longitude, latitude, bearing, and speed could be obtained. Specifically, the longitude and latitude

information tell the location of the device in degrees. Bearing is the horizontal direction of travel of this device, from 0 to 360 degrees. The speed is not calculated with the function of position against time. Instead, the speed is measured based on the Doppler shift in the signals coming from the satellites. Hence the speed is instantaneous speed, not the space mean speed.

4.1.3. Summary

The research team set up the server and developed smartphone apps to realize the data communication between smartphones and the server. In this chapter, the architecture design of server and apps were presented. For the server, the designed architecture could realize the functions of receiving data from smartphones, computation, storing data, and sending data to smartphones. For smartphone apps, the designed architecture could help collect the raw sensor data, initially process data, upload data to the server, and receive message from the server. In addition, the primary smartphone sensor data were discussed.

4.2. Smartphone Data Analysis and Validation

To better emulate the OBUs, it is essential to understand the statuses of smartphone users in real time. The statuses include the location, speed, traffic environment, transportation mode, acceleration, and movement. In the following sections, all these statuses were obtained with the smartphone sensor data and experiments were conducted to validate their accuracy.

4.2.1. Smartphone Position Data Accuracy Validation

The smartphone position data is very important for the system to know the location of smartphone users and send important information. To correctly measure the position accuracy, ground truth data were collected from RTK (Real Time Kinematics) GNSS (Global Navigation Satellite Systems). RTK positioning is a satellite navigation technique used to enhance the precision of position data derived from multiple satellite-based positioning systems (GNSS) including GPS

(Global Positioning System), GLONASS (GLObal NAVigation Satellite System), BeiDou, Galileo, QZSS (Quasi-Zenith Satellite System), and SBAS (Satellite-based Augmentation System). It uses measurements of the phase of the signal's carrier wave in addition to the information content of the signal and relies on a single reference station or interpolated virtual station to provide real-time locations with the centimeter-level accuracy.

The two phones used in this experiment are iPhone Xs and Samsung Note 9. As discussed in Chapter 3, two different applications were developed for reading the position data from the two different smartphones. Position data are acquired using the smartphone apps. First, the accuracy of two different RTK GNSS devices (Emlid Reach RS+ and CHC X91+ RTK GNSS Receiver) were validated. Figure 46 shows Reach RS+ and CHC X91+ RTK GNSS receiver, respectively. Notably, each GNSS module requires two GNSS devices: one is the base and the other is the rover.



(a) Emlid Reach RS+ RTK GNSS



(b) CHC X91+ RTK GNSS

Figure 46. RTK GNSS

The GNSS modules were validated by comparing with Google map. A volunteer walked along a crosswalk with GNSS rover and then the GNSS data points were plotted on Google Map (Figure 47). The validation results suggested that the Emlid Reach RS+ could provide positioning

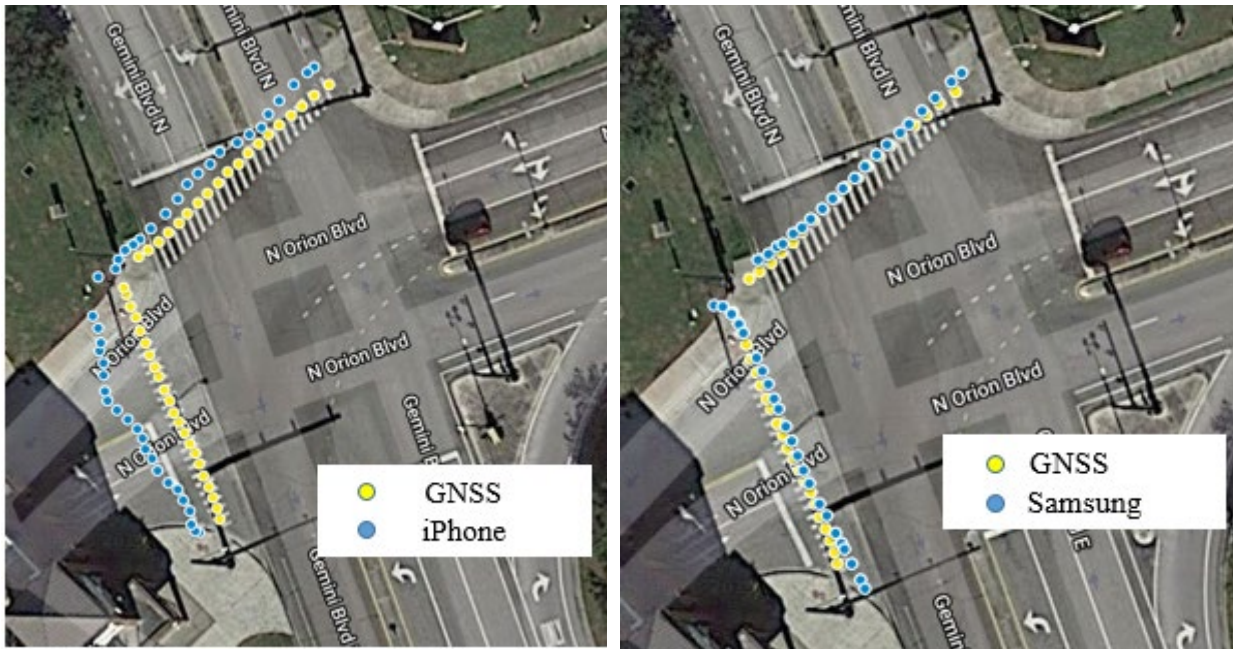
information with higher accuracy and better convenience to carry for the test. Hence, the Emlid Reach RS+ was used for validating the accuracy of smartphone positioning data.



Figure 47. Validating GNSS data using Google map

4.2.1.1. Validation of Pedestrians' Position Data

The volunteers walked along the crosswalk with a smartphone on one hand and the RTK GNSS receiver over in the other hand. Three experiments were conducted for each phone. Smartphone and GNSS data points were plotted on Google map and the error was calculated by measuring the distance between the two corresponding points from smartphones and the RTK GNSS. Figure 48 shows different data points from iPhone, Samsung, and the ground truth (RTK GNSS). It demonstrates that pedestrian trajectories from the Samsung phone is more close to the ground truth compared with the iPhone.



a) RTK GNSS data vs iPhone position data

b) RTK GNSS data vs Samsung position data

Figure 48. GNSS data points vs smartphone data points

For each experiment, the distances between the ground truth points from the RTK GNSS and the points based on smartphone data were calculated while the results are summarized in Table 21. As shown in Table 21, the mean distances of Samsung and iPhone are 7.33 feet and 11.33 feet, respectively. Maximum GPS error for the iPhone can go up to 32 feet, whereas it is 25 feet for the Samsung phone. Minimum errors for both smartphones are 1 foot. Hence, it could be concluded that the position data from the two phones could reflect the position information for pedestrians while Samsung is more accurate than iPhone. Meanwhile, it is expected that newer models of phone will achieve better accuracy.

Table 21. Validation results for iPhone

Smartphone	Average error	Max error	Min error
iPhone	11.33 feet	32 feet	1 foot
Samsung	7.33 feet	25 feet	1 foot

4.2.1.2. Validation of Vehicles' Position Data

To validate the position data for vehicles, the driver drove a car along the road with the two smartphones and the RTK GNSS. As shown in Figure 49a), the two smartphones were mounted in the front of the vehicle. Meanwhile, the GNSS rover was put on the roof of the vehicle with a tripod since the GNSS receiver must face the sky for high-accuracy localization outputs (Figure 49b)). The validation tests were conducted at 3 different speeds, i.e., 20, 25, and 30 mph. For each speed, three experiments were conducted with a mile for each experiment. The driver tried to maintain a constant speed every time. Sometimes, the speed fluctuated due to the influence of other surrounding cars and red lights. The data under these influences were excluded for the validation.



a) Two smartphones in the vehicle

b) GNSS rover on top of the vehicle

Figure 49. Locations of smartphones and GNSS

Similar to the validation of pedestrians' locations, the distances between GNSS points and smartphone points were calculated and summarized in Table 23. Figure 50 demonstrates the GNSS, iPhone and Samsung data points for the vehicle. The mean errors between the GNSS and

smartphones are 4 feet for Samsung and 11 feet for iPhone, respectively. The mean and maximum errors for the iPhone are much greater than that of the Samsung. The results indicate that both smartphones could provide acceptable position information for vehicles.

Table 22. Results for vehicle localization data validation

Device	Mean error (feet)	Max error (feet)	Min error (feet)
iPhone	11	73	1
Samsung	4	12	1

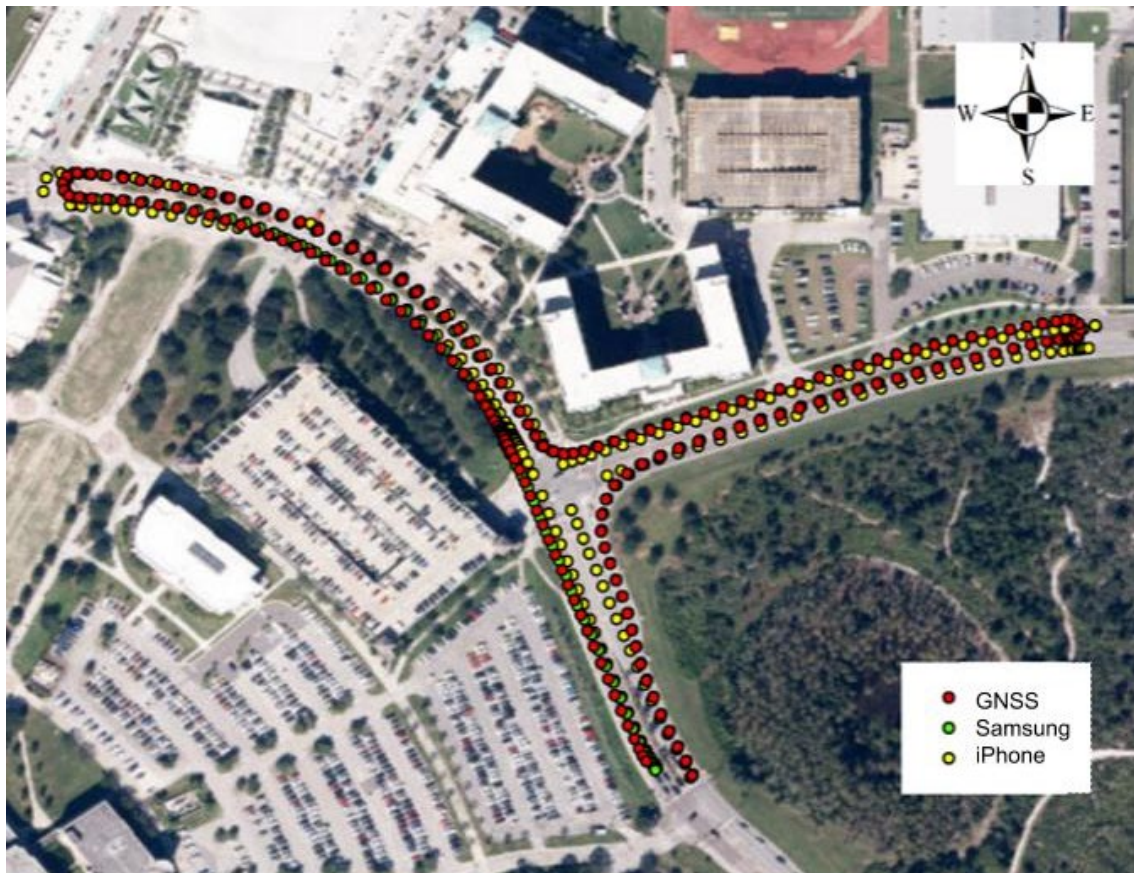


Figure 50. GNSS and smartphone points for the vehicle

4.2.2. Validation of Smartphone Speed

4.2.2.1 Validation of Pedestrians' Speed

Two different statuses of pedestrians were taken into account for pedestrian speed validation: walking and jogging. Average walking speed for pedestrians on crosswalks is from 4.1 feet/sec to 4.3 feet/sec while the speed of jogging is from 6 to 8 feet/sec (Seyfarth et al., 2009). Since it is not very easy to measure instantaneous pedestrian speed, the volunteer walked and jogged along the crosswalk with the smartphone and a stopwatch. Five experiments were conducted for each walking and jogging. Figure 51 illustrates the trajectory of the pedestrian at the test crosswalk during an experiment. The ground truth speeds were calculated by dividing the crosswalk distance (73.52 feet) and time from the stopwatch. The smartphone app provides an instantaneous speed at every second. These speeds are averaged and compared with the ground truth.



Figure 51. Location for pedestrian speed validation

For each status and each smartphone, the validation result is summarized in Table 23. The measure MAPE (Mean Absolute Percentage Error) was computed. According to Table 23, the errors of speed data from the two smartphones were between 5 and 10 percent, which indicates that the smartphone speed data could reflect pedestrians' status. Also, the speeds from the two smartphones were consistently lower than ground truth. Hence, the original speed data from the smartphones were weighted by +10% to obtain better speed readings for pedestrians.

Table 23. Smartphone walking speed validation results

Status	Device	Average speed (feet/sec)		MAPE (%)
		Ground Truth	Smartphone	
Walking	iPhone	4.038	3.794	6
	Samsung	4.012	3.634	9
Jogging	iPhone	8.176	7.696	6
	Samsung	8.318	7.704	7

4.2.2.2 Validation of Vehicles' Speed

To validate the vehicle speeds, the driver drove the vehicle at a constant speed using adaptive cruise control. Five different speeds (i.e., 25, 30, 35, 40, 45, 50 mph) were tested. At least 7 minutes were tested for each speed. Figure 52 shows the dashboard of the car using adaptive cruise control to maintain the speed at 25 mph.

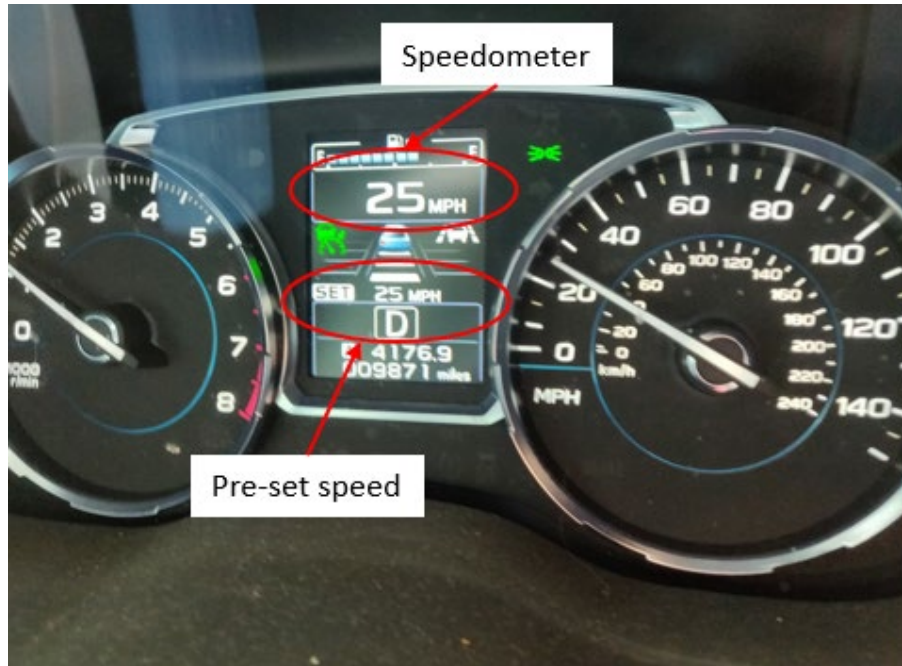


Figure 52. Validation of vehicle speed using adaptive cruise control

Table 24 demonstrates excellent accuracy of smartphone app speed reading. For each speed, the smartphones show very precise readings taking the speedometer as the ground truth. Except for one case (Samsung, 45 mph speed), the percentage of error for different cases is only 1 percent.

Table 24. Vehicle speed validation results

Speedometer reading in mph (ft/sec)	Device	Speed reading in mph (ft/sec)	MAPE (%)
25 (36.67)	iPhone	24.71 (36.24)	1
	Samsung	24.47 (35.89)	1
30 (44.00)	iPhone	30.15 (44.22)	1
	Samsung	29.95 (43.92)	1
35 (51.33)	iPhone	35.23 (51.67)	1
	Samsung	34.82 (51.07)	1
40 (58.67)	iPhone	39.54 (57.992)	1
	Samsung	40.10 (58.81)	1
45 (66.00)	iPhone	46.32 (67.93)	1
	Samsung	45.07 (66.10)	3
50 (73.33)	Samsung	49.83 (73.08)	1
	iPhone	49.45 (72.52)	1

4.2.3. Localize Users

4.2.3.1. Motivation for Localization

The GPS data from smartphones provide information about the position of a user. The research team also needs to get the localization (Sivaraman et al., 2013) information to successfully emulate the OBU. The fundamental difference between positioning and localization can be understood from their definitions:

Positioning: only gives information about a user coordinates; no information about environment such as type of road it is on and whether it is approaching a curve or an intersection.

Localization: gives information about receiver coordinates and also the environment

The research team attempted to create their own database with the appropriate localization information (such as curves, intersections, crosswalks, etc.). The positioning information (GPS coordinates) received from the smartphone can then be mapped to extract the appropriate localization information.

4.2.3.2. Localization for Smartphone Users

To localize a smartphone user, the Radius Neighbor Classifier algorithm was used, which is a variant of the popular K-Neighbor Classifier (Keller et al., 1985). While the K-Neighbor classifies according to the nearest K neighbors, the Radius Neighbor classifies a point only if the values fall within a particular radius.

After numerous tests, the research team has decided to select a radius of 30 feet for this algorithm. The motivation behind choosing this specific value is that, this method could localize points successfully even if the GPS points are received with large errors. Also, this method is extremely fast in classification. The localization process is in the server and the average time

taken to localize a newly uploaded GPS point is 1ms. Being efficient and fast is very important for the purpose of the project, since the research team aims to identify and send P2V and I2V warnings to avoid potential conflicts.

The flowchart of the algorithm used is shown in Figure 53. The GPS data from smartphones are uploaded to the server in real time. The task of localization then takes place in the server. First, GPS points and direction information are encoded properly to feed into the Radius Neighbor Classifier. The classifier can then classify points based on the pre-defined radius, which is 30 feet in this project. When the localization process is completed, the server generates appropriate labels with the localization information. The labels are now ready to be used for other calculations.

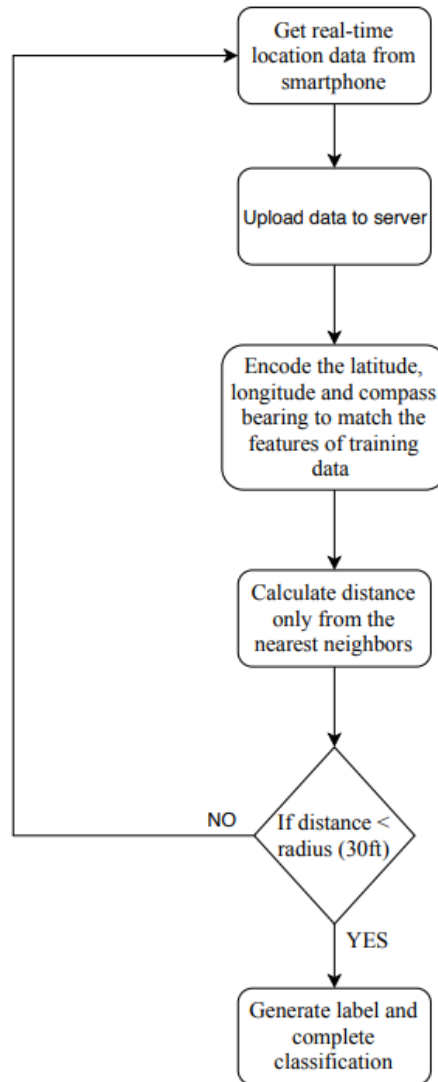


Figure 53. Flowchart for radius neighbor classifier

4.2.3.3. Evaluation Experiments and Results

A base map was prepared with the appropriate localization information by using the ArcGIS software. As shown in Figure 54, the base map contains certain points accurately mapping different roads and crosswalks. The points were automatically generated along lines with a certain distance (a distance of 10 feet was used in this project). Hence, the point base map could be easily generated with the roadway base map from FDOT and its coverage could be easily expanded to other locations of interest. The research team has programmed the training algorithm in the server.

If a new training dataset is collected, the server system could update the localization algorithm automatically.



Figure 54. Base map points captured by GNSS

Around 600 points were used for training the model and about 2,000 points collected from the smartphones at different locations were used for the validation. The accuracy was calculated as the percentage of correct labels. The results for vehicles and pedestrians are summarized in Table 25. It can be seen from the table that the accuracy of the two smartphones for vehicles and pedestrians is approximately 95%. Hence, it could be concluded that the proposed method has obtained high accuracy for all cases of localization. It should be noted that the errors are due to the error of the smartphone GPS sensor. Although the GPS sensor could provide good positioning information, it still introduces some errors. The OBU could provide more GPS information, but is very expensive. On the other hand, the smartphone approach could provide large benefits with the high market penetration rate. Meanwhile, more accurate GPS sensors such as dual-frequency

GNSS will be adopted in the smartphones in the near future, and the localization accuracy could be improved as well.

Table 25. Results from the localization algorithm for vehicles

Mode	Data Sources		Sample Size	Correctly Identified Labels	Accuracy (%)
Vehicle	Training Data	GNSS	539	-	-
	Smartphone Data	Samsung	579	570	98
		iPhone	450	427	95
Pedestrian	Training Data	GNSS	92	-	-
	Smartphone Data	Samsung	433	415	95
		iPhone	450	427	94

4.2.4. Identification of User Travel Modes

It was necessary, for the purpose of this project, to successfully identify the transportation modes of the smartphone app users. As this project focuses on the I2V and P2V applications, only two transportation modes needed to be identified: (1) driving; (2) walking. The research team wanted to identify these two user modes based on different sensor readings. In the next phase, the biking mode could also be added in the identification if the B2V (Bike-to-Vehicle) application is included.

The transportation modes could be identified by fusing different sensor data. Figure 55 shows example data of accelerometer z-axis (marked in blue), gyroscope z-axis (marked in red), and speed (marked in yellow). The time labels the x-axis, and the y-axis shows different sensor data. The first part of the data shows sensor readings for walking, and the latter shows those for driving. It indicates that the sensor data for pedestrians and vehicles are quite different. Therefore, given various sensor data, it is fairly straightforward to detect various modes.

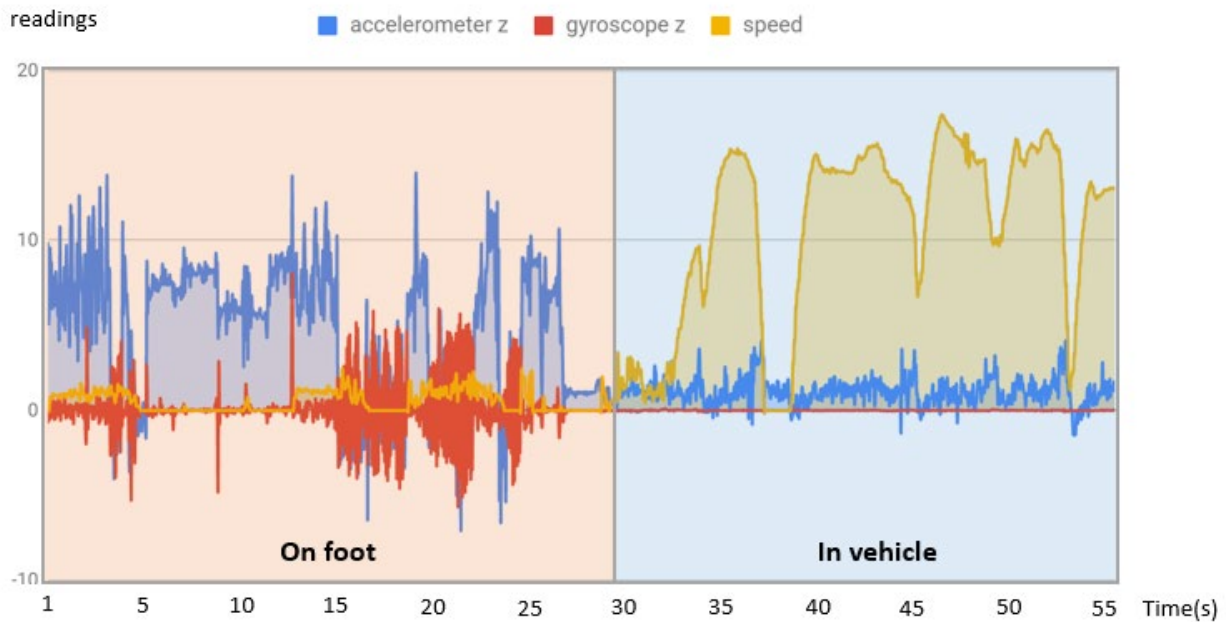


Figure 55. Example data of accelerometer, gyroscope and speed sensors from the smartphone

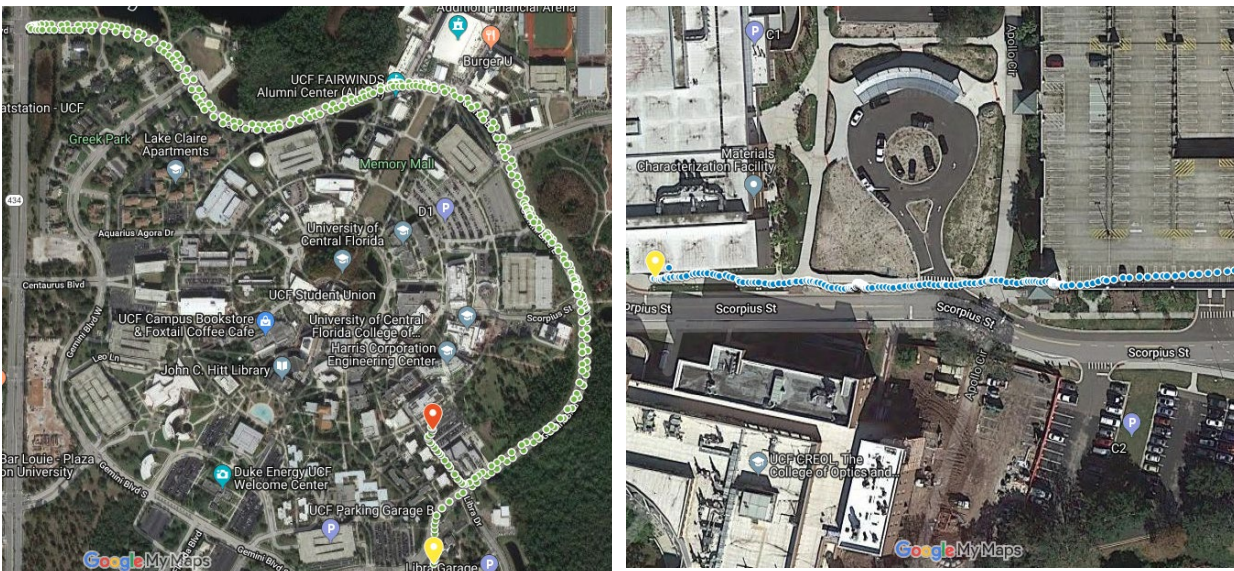
4.2.4.1. App Development for Activity Detection

For Android smartphones, Google has released its own version of Activity Recognition, popularly known as Activity Transition API. The developers have made sure that they use the least amount of sensor readings (which means lower battery usage) but also get the maximum accuracy of detecting an activity. Also, there is a similar API provided by Apple that can provide information about the user's activity, which is called CMMotionActivity.

The proposed mobile app could take short bursts of sensor data to detect the users' modes. Hence, the two APIs were coded in the apps for the Android and iOS systems. Transportation modes could be automatically identified by the two apps and the need for manually selecting the mode by the user could be avoided.

4.2.4.2. Validation Experiments and Results

Two experiments were conducted for testing the APIs for driving and walking. In the first experiment, data was collected while driving. Figure 56a) illustrates the performance for correctly identifying driving mode. The selected driving route had several intersections, one curve and one U-turn. In the second experiment, data was collected while walking. Figure 56b) shows the mode detection for walking. The blue dots show which points have been correctly identified as walking. It was found that the APPs of both Android and iOS systems could accurately identify the transportation modes even if the user stopped at intersections for a while.



a) Mode detection for driving

b) Mode detection for walking

Figure 56. Experiment routes for driving and walking

While the APPs with the APIs were able to correctly differentiate between walking and driving modes, a transition time is needed when the user's mode changes from one mode to the other. For example, a user drives a vehicle, parks the vehicle in a garage, and walks. There is a transition time to detect such changes and it takes around 25 seconds for the Android app and 5 seconds for the iOS app. Such change usually takes place in the parking garages or parking lots

and generally no warning information is needed. Hence, the transition time would not affect the usage of the developed APPs.

4.2.5. Acceleration Estimation

A vehicle's motion can be described by its speed or acceleration. Specifically, the longitudinal and lateral accelerations on its X and Y-axes (Figure 57) are two crucial factors for driving behavior identification. For example, negative values on longitudinal acceleration represent braking, which is a safety indicator for the rear-end collision risk evaluation. In this part, the research team aimed to use the sensors of a smartphone to estimate vehicles' acceleration.

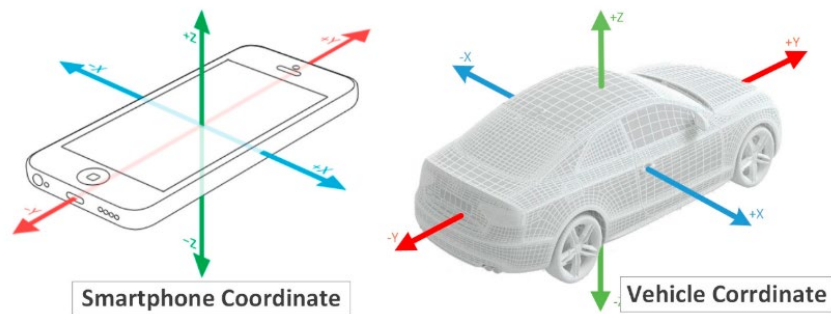


Figure 57. Coordinate systems of smartphone and vehicle (Eboli et al., 2016)

4.2.5.1. Coordinate Reorientation

As shown in Figure 57, smartphone and vehicle have similar coordinate systems. However, the position of a smartphone inside a vehicle is usually arbitrary. For example, a driver may place his/her smartphone in the pocket or use a mount to fix it. Thus, the accelerometer reading of a smartphone cannot truly represent the acceleration of a vehicle. In order to estimate the acceleration of the vehicle by the smartphone. The first task is to align the coordinate system. Generally, two methods are available for this problem. One is based on Euler angles (Tak et al., 2015) and the other is based on rotation matrix (Liu et al., 2017).

The Euler angles are three angles used to describe the orientation of a rigid body with the respect of a fixed coordinate system, which is shown in Figure 58. The red coordinate system is the orientation of a smartphone while the blue one indicates the orientation of a vehicle. The basic idea of Euler angles is any orientation of the accelerometer can be represented by three values. The angle β is the angle between Z axis and z axis, the angle α is the angle between the x axis and the N axis (x-convention), and the angle γ is the angle between the N axis and the X axis (x-convention).

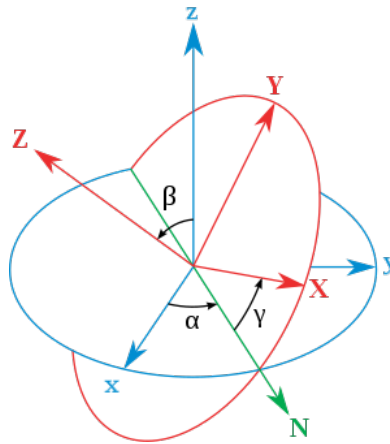


Figure 58. Euler angles illustration

The three angles are the fundamental parameters for this method. However, the smartphone needs to be stable for angle calculation. Hence it should not be placed to any kind of acceleration during the calculation, which makes this method not flexible enough for the vehicle application.

Another common way is based on the rotation matrix. It is done in two steps (Figure 59): (1) converting the smartphone coordinate system to the geometric coordinate system using rotation matrix; (2) converting the geometric coordinate system to the vehicle coordinate system using bearing and magnetic declination, which can be obtained from GPS and magnetometer sensors of smartphones (Mohan et al., 2008).

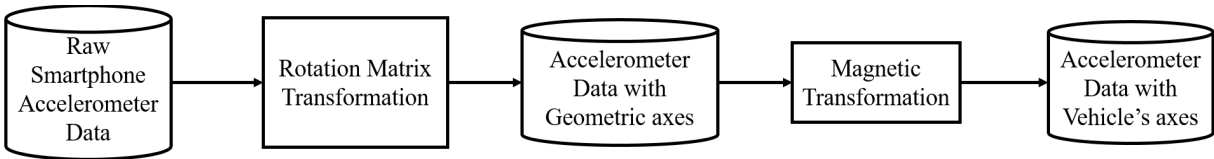


Figure 59. Coordinate reorientation

For the first step, the smartphone system provides a build-in rotation matrix sensor, a 9 by 9 matrix transforms accelerometer readings from the smartphone's coordinate system to the geometric coordinate system which is defined as a direct orthonormal basis (Figure 60). Specifically, X is tangential to the ground at the device's current location and roughly points East, Y is tangential to the ground at the device's current location and points towards the magnetic North Pole, and Z points towards the sky and is perpendicular to the ground.

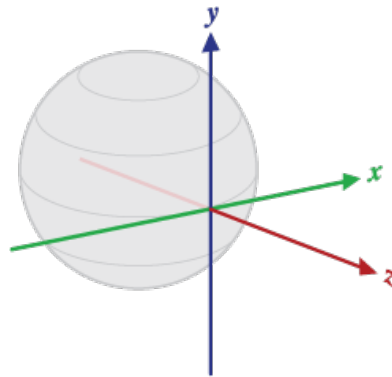


Figure 60. World's coordinate system

The second step is the transformation from world's coordinate system to vehicle's coordinate system. The apps collect magnetic sensors for magnetic declination, which is the deviation of magnetic north from true north. Moreover, bearing can be obtained from two continuous GPS points, which is the angle α in Figure 61. The T-North represents the true north while Y' is the direction of vehicle.

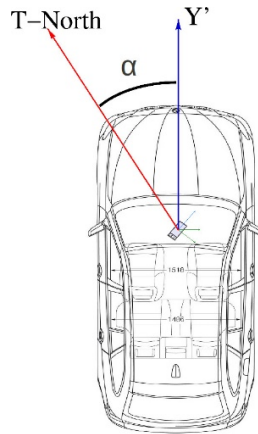


Figure 61. Bearing illustration (Bhoraskar et al. 2012)

If the accelerometer is well oriented, the reading from smartphone accelerometer can better reflect vehicles' acceleration on different axes. The reading of Z axis is very close to 32.17 feet/s^2 , which is the gravity. The reading of Y axis can represent the longitudinal acceleration of vehicle, which will be very close to the acceleration obtained from speed. Meanwhile, the reading of X axis can reflect the lateral acceleration of vehicle.

4.2.5.2. Validation Experiments and Results

The proposed method was programmed in our smartphone application and the reoriented data were uploaded to the server. In order to validate the results from coordinate reorientation, the research team conducted 27 driving experiments on UCF campus, and the trajectory of one experiment is shown in Figure 62. Besides, the smartphone's positions are shown in Figure 63. Different smartphone positions were tested in the experiments.

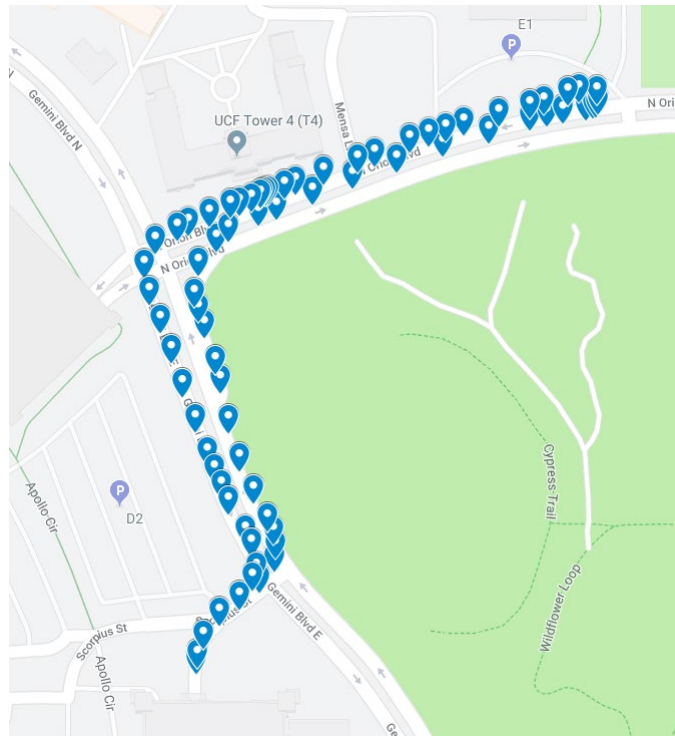


Figure 62. Vehicle's trajectory

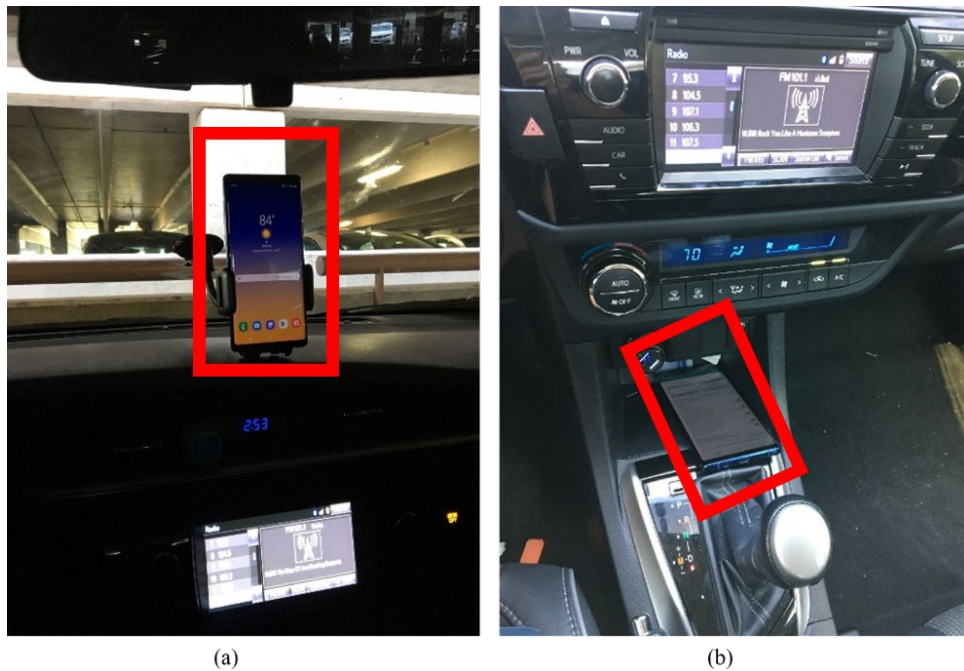


Figure 63. Smartphone's positions

After the coordinate reorientation, the readings of smartphone's acceleration can better reflect vehicle's motion. The Z axis of acceleration is not shown here as it was a constant of

gravity. The readings of Y axis during acceleration and deceleration are shown in Figure 64 (a) and Figure 64 (b). The blue lines are the speeds and the red lines are the acceleration rates. The driver kept accelerating in Figure 64(a) and made a hard brake in Figure 64(b). It shows the accelerometer after the coordinate reorientation successfully reflect the true acceleration of the vehicle. The acceleration was positive when the speed continued increasing while the acceleration was negative when the driver decelerated. Besides, the acceleration rate had a similar trend as the speed.

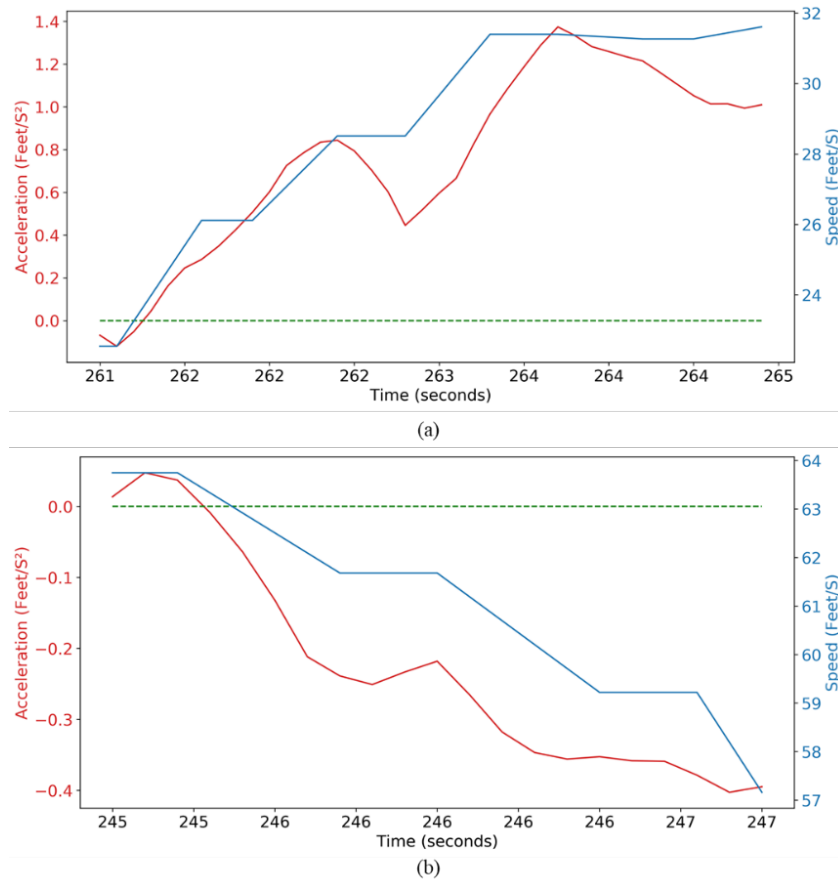
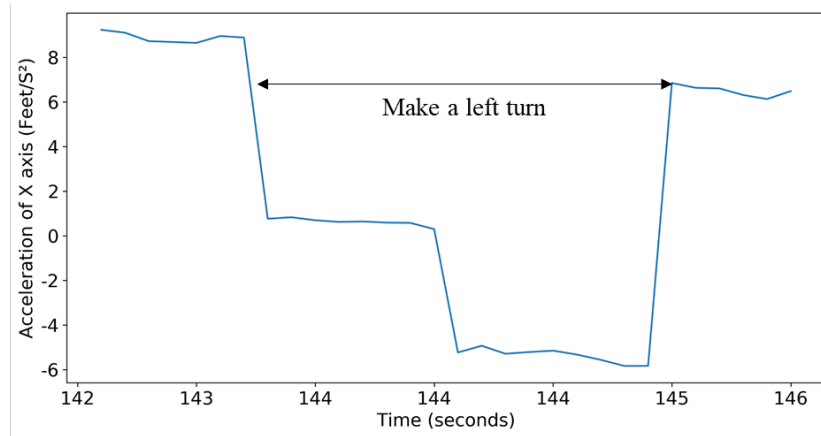


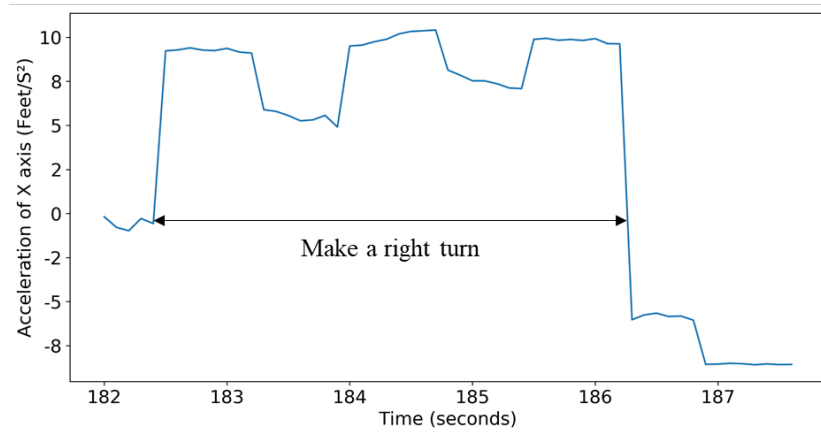
Figure 64. Reorientation results of Y axis

Similarly, after reorientation, the X axis of accelerometer has the capacity to capture vehicles' turning movements. The result of one driving test is shown in Figure 65. When the

vehicle makes a left turn, the value of X axis dropped from 8 to -6 feet/s². On the contrary, its values increased dramatically from 0 to 9 feet/s² during right turn.



(a)



(b)

Figure 65. Reorientation results of X axis

This part showed the possibilities of using the rotation matrix and bearing for coordinate reorientation. The reading from accelerometer after the reorientation can successfully represent vehicle's different movements.

4.2.6. Vehicle Movement Detection

A vehicle has several different movements, such as going straight, left (right) turn, and U-turn. Each movement has its own unique characteristics. The identification of movements could be

very important at intersections since vehicles could change the paths. Without the movement identification, it could be difficult to identify the potential conflicts between vehicles and pedestrians at intersections. In this part, different smartphone's sensors were used to identify vehicle movements based on machine learning methods.

4.2.6.1. Data Description

After the coordinate system reorientation, the readings from smartphone's sensors can much better reflect vehicle's information. In this part, smartphone sensors including accelerometer, gyroscope, and speed were used as inputs to detect vehicle movements (Bhoraskar, Vankadhara et al. 2012). Gyroscope has the same coordinate system as accelerometer but with a different unit (rad/s), which can be found in Figure 66 (a). The vehicle has its own motion system, pitch, roll and yaw (Figure 66 (b)). These three parameters can be reflected by the values of smartphone's gyroscope after the coordinate reorientation.

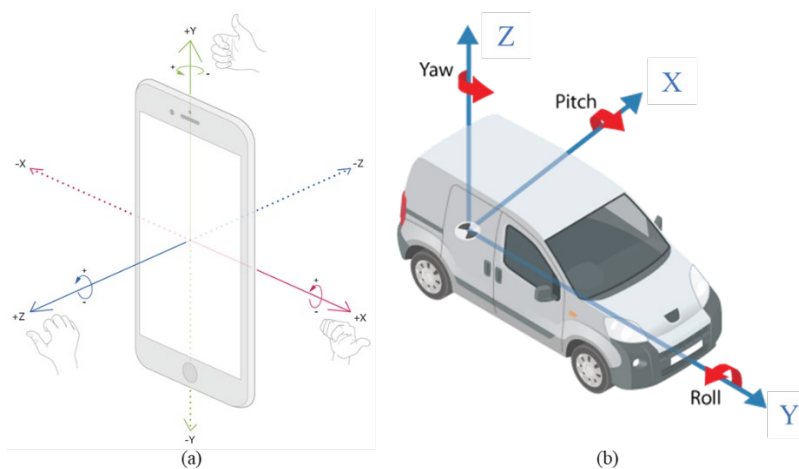


Figure 66. Gyroscope of smartphone and motion of car

Generally, when a car is making a turning movement, the value of Z axis (yaw) will change significantly than other two values. Specifically, it will be positive for left turn and negative for right turn, which are shown in Figure 67.

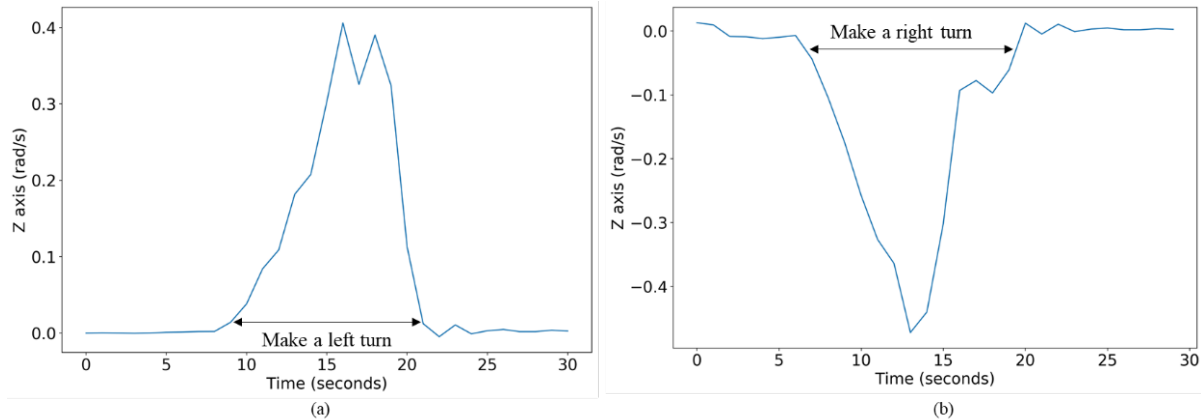


Figure 67. value of Z axis for left turn and right turn

4.2.6.2. Data Pre-Processing

Raw sensor data have a lot of noise. Take the x axis of accelerometer as an example, the blue line in Figure 68 shows the raw readings, which fluctuated frequently. In order to remove sensor noise, a smoothing filter was applied. There are many available methods, such as moving average filter, Kalman filter, and linear least squares filter. The research team selected moving average as the filtering method since it's the most common for sensor data cleaning (Paefgen et al., 2012; Vlahogianni and Barmponakis, 2017). Meanwhile, the computation cost for this method is low.

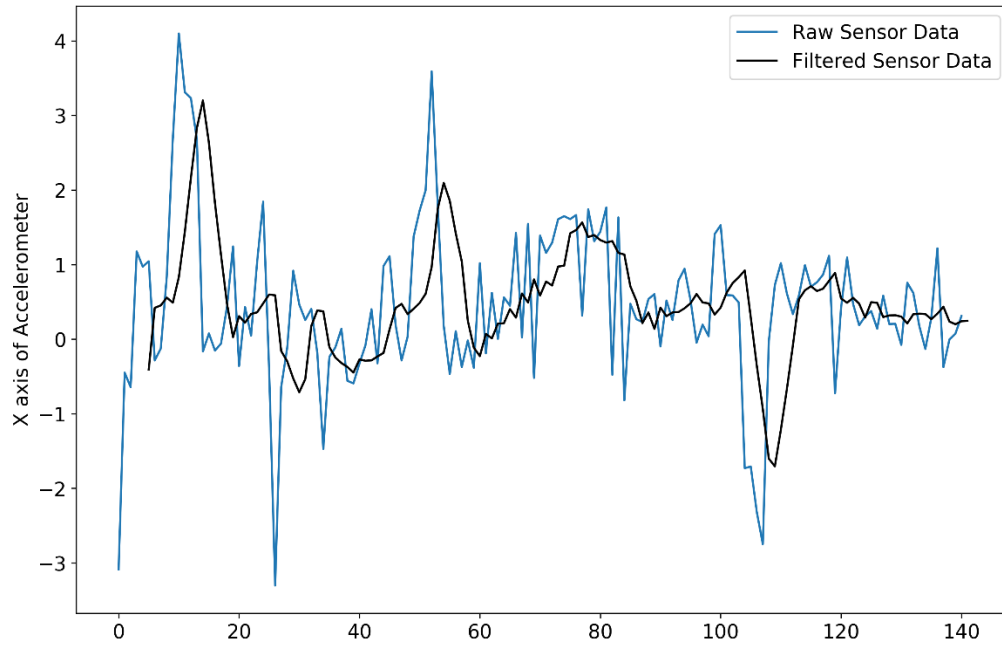


Figure 68. Value of X axis of accelerometer before and after filtering

Moving average is a very straightforward method. For a given time period, it calculates the average value over a number (N) of variables and replace the Nth variable with the new value. The equation is shown in the following equation:

$$SMA = \frac{\sum_{i=1}^N A_i}{N}$$

where A is the variable for each time slice, and N is the value for predefined time periods. The research team selected 5 seconds as the time period since it can successfully smooth the data without losing too much information. All the sensor values were applied this filter accordingly. The result of x axis of accelerometer is shown as the dark line in Figure 68. It clearly shows that the filtered data is smoother compared to the raw data (blue line).

In total, we have six features for movement classification. The Pearson correlation coefficient was computed to check the correlation between features. The result is shown in Figure 69. X_acc and Y_acc mean the value of X and Y axis of accelerometer, while X_gyro, Y_gyro,

and Z_gyro mean the value of X, Y, and Z axis of gyroscope. Again, the constant Z axis of accelerometer (gravity) was not included. All of the features were obtained after the coordinate reorientations. For each pair of sensor data, very small correlations were found. Hence, all the sensor data were used for the movement identification.



Figure 69. Feature correlation

4.2.6.3. Model Development and Validation

Real-world driving data were collected from May, 2019 to June, 2019. Four trained volunteers were asked to collect during daily commutes using the developed APPs developed by the research team. The positions of the smartphones were random in order to improve the transferability of the proposed methods. The data update frequency is 1 Hz. In total, the research team collected around 17,000 readings. One of the trajectories is shown in Figure 70.

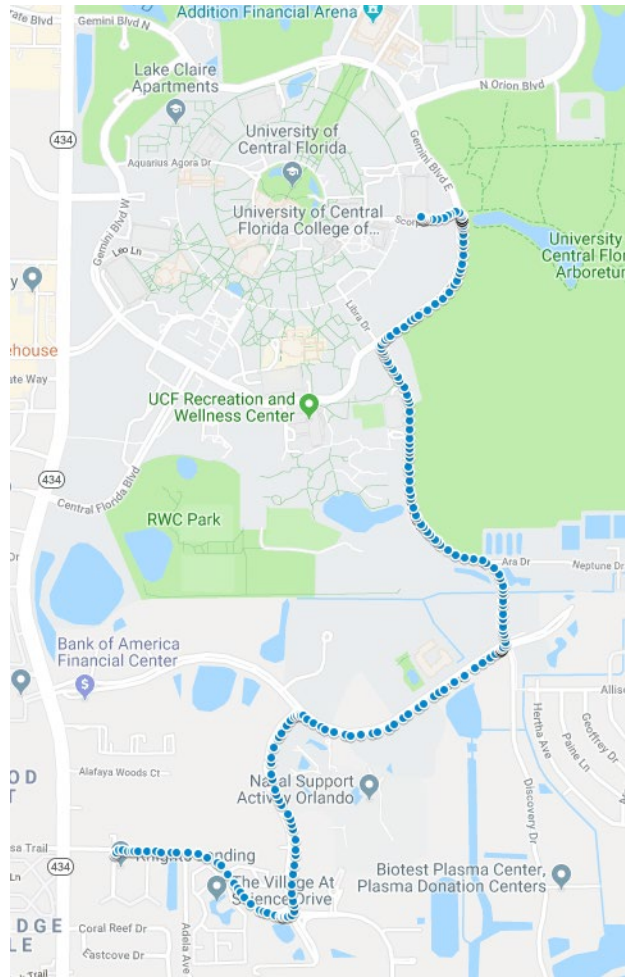


Figure 70. Driving trajectory

The research team selected support vector machine (SVM), Random Forest, KNN-Neighbor, and Gradient Boosting as the classification methods. All methods have been applied in driving movement classification previously (Eboli et al., 2016; Liu et al., 2017). The process of classification is shown in Figure 71.

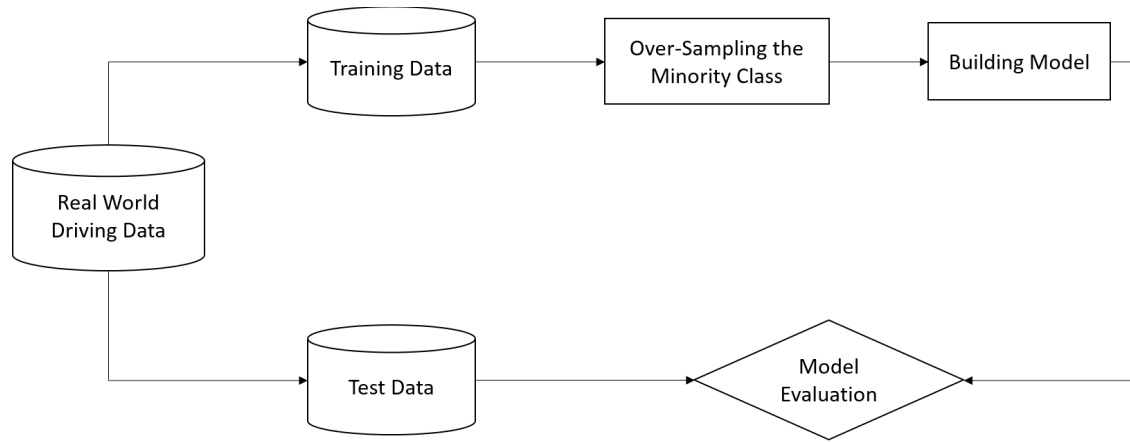


Figure 71. Classification process

The real-world driving data include four types of movements, driving through, U turn, right turn and left turn. These four events were labelled manually based on the trajectory. Then, the dataset was divided into training and test data based on the ratio of 7:3. However, since driving through is the most common event during daily driving, the dataset contains much more go straight events than others. In order to solve this problem, the research team applied Synthetic Minority Over-sampling Technique (SMOTE) as over-sampling method to increase the sample number of minor classes in training dataset (Yu et al., 2017), which is shown in Figure 71. After the over-sampling process, in the training dataset, the number of straight, left turn, right turn and U-turn changed from 1328, 75, 57 and 15 to 1313, 1322, 1320 and 1327. The pre-trained model was programed at the server side. Hence, the movement could be identified in the server based on the uploaded sensor data from smartphones. Precision and recall were selected as the metrics for model evaluation. They are useful measures of success of classification especially when the classes are very imbalanced. Precision is a measure of result relevancy, while recall is a measure of how many truly relevant results are returned. Precision (P) is defined as the number of true positives (T_p) over the number of true positives plus the number of false positives (F_p). Recall (R)

is defined as the number of true positives (T_P) over the number of true positives plus the number of false negatives (F_N). The formulation of these two measures are shown as following:

$$P = \frac{T_P}{T_P + F_P} \quad R = \frac{T_P}{T_P + F_N}$$

The performances of different models are shown in Table 26. Overall, the random forest model outperforms other models for the classifications of straight, left turn, and right turn movements. All of these three movements have good precision and recall results. The only exemption is U-turn event, the recall of random forest is relatively low (the value is 0.5) while the precision is much better than other methods (the value is 1). Although SVM could provide good recall performance for the U-turn movements, its precision is very low (the value is 0.27). The reason that the U-Turn movement has low accuracy is that the event is rare. Hence, considering both precision and recall, the random forest method could provide the best classification for the four types of movements. Hence, the method was adopted and programmed in the server to classify vehicles' movements.

Table 26. Classification results

Indicator	Precision				Recall			
	Methods	KNN	SVM	Gradient Boosting	Random Forest	KNN	SVM	Gradient Boosting
Straight	0.99	0.99	0.99	0.99	0.86	0.86	0.97	0.98
Left-Turn	0.36	0.31	0.68	0.86	0.74	0.68	0.89	0.95
Right-Turn	0.40	0.52	0.93	0.99	0.71	0.93	0.99	0.99
U-Turn	0.19	0.27	0.50	1.00	0.75	1.00	0.50	0.50

Figure 72 shows the driving trajectory of an experiment with different movements. It indicates that the developed method in the sever could successfully identify the corresponding movement.



Figure 72. Classification result

4.2.7. Summary

Different traffic parameters including positions, speeds, localizations, transportation modes, accelerations, and movements were computed based on the smartphone data such as GPS coordinate, GPS speed, compass bearing, accelerometer, gyroscope. The computation is summarized in Table 27. For different traffic parameters, different smartphone data were used. Four traffic parameters including position, speed, transportation mode, and acceleration are computed in the smartphone app, and then uploaded to the server. On the other hand, the localization and movement are determined in the cloud server once the smartphone data are uploaded. Except for the position and speed parameters, multiple models or methods were proposed to obtain traffic parameters. The validation results indicated that the all obtained smartphone data could well reflect users' statuses with good accuracy.

Table 27. Summary of traffic data computation with smartphone data

Traffic parameters	Required data for computation		Computation location	Method	Latency (ms)
	Smartphone data	Other data			
Position	GPS coordinate	-	Smartphone app	Directly obtained	instantaneous
Speed	GPS speed	-	Smartphone app	Directly obtained	instantaneous
Localization	GPS coordinate, compass bearing	Base map	Cloud server	Radius Neighbor Classifier	3
Transportation mode	Accelerometer, gyroscope, GPS speed	-	Smartphone app	API from Google and Apple	instantaneous
Acceleration	Accelerometer, gyroscope, compass bearing	Rotation matrix	Smartphone app	Coordinate Reorientation	instantaneous
Movement	Accelerometer, gyroscope, and GPS speed	-	Cloud server	Random forest classification method	3

4.3. CONCLUSIONS

To emulate the OBU implementations, the research team has designed the architectures of the server and developed smartphone applications (apps) to realize the communication between the cloud server and smartphones. In this task, multiple methods were proposed to obtain different traffic parameters of smartphone users based on the extensive understanding of the smartphone data. Numerous experiments were conducted to evaluate the obtained traffic parameters. The results suggested that the traffic parameters based on the smartphone data could reflect users' traffic statuses successfully. With the input of traffic parameters, the smartphones could be used to emulate the OBU implementations. In the following tasks, algorithms will be proposed and programmed in the server to emulate the OBU implementations for the P2V and I2V applications.

CHAPTER 5. TEST OF COMPLEMENTARY SENSORS

Pedestrians are the most vulnerable road users (FHWA, 2018). While the smartphone apps could successfully identify pedestrians, not all pedestrians would like to have the developed apps. To detect pedestrians without the developed apps and send approaching drivers warning messages if possible conflicts are identified, it is necessary to test other complementary sensors which could detect pedestrians proactively. In this task, the research team made a prototype by developing a real-time pedestrian detection system.

Section 5.1 discusses the structure of the P2V (Pedestrian-to-Vehicle) warning system based on the complementary sensor. The feasibilities of two emerging sensors (i.e., camera and Lidar) were validated and the proper sensor was adopted. With the adopted sensor, a prototype of a real-time pedestrian detection system was developed. Then, multiple experiments were conducted to test the system's robustness.

Section 5.2 extends the pedestrian detection system to a vehicular queue detection system at intersections. The queue detection system could be used for the I2V queue warning application. Experiments were conducted to validate the developed queue detection system.

Section 5.3 concludes the feasibility of using an external sensor to proactively detect pedestrians at different locations as well as queue length at intersections. Meanwhile, the plans for the following tasks about emulating the OBU applications with the smartphones by using the detected information, are presented.

5.1. Developing a Real-Time Pedestrian Detection System

5.1.1. The Structure of the P2V Warning System with Complementary Sensors

Although the developed smartphone apps could automatically detect pedestrians, not all pedestrians might use the developed apps. To proactively detect pedestrians even they do not have the develop apps, the research team developed a prototype of system to detect pedestrians in real time with complementary sensors.

In this task, a collision warning system with an external sensor was proposed. As shown in Figure 73, the detection unit is used to detect pedestrians. The detection data about the pedestrians are uploaded to the server through the cloud internet. All pedestrians under the detection area could be detected. Meanwhile, the vehicles' statuses could be collected by the drivers' smartphones and uploaded to the cloud server. The cloud server combines all information and determines whether potential conflicts exist. If a potential conflict is identified, the warning message will send to the corresponding drivers' smartphone in real time.

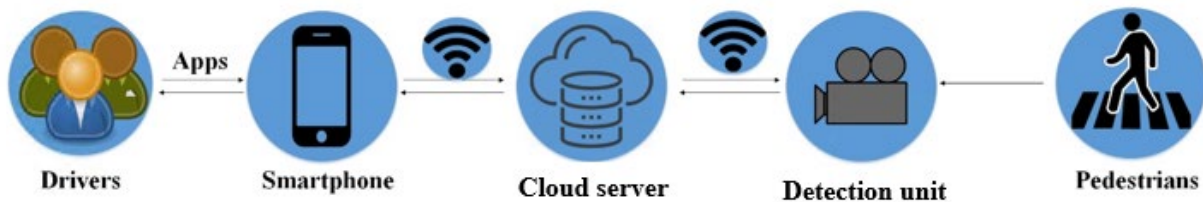


Figure 73. A framework of collision warning system

Hence, in addition to the smartphone and the server discussed in Task 3, the detection unit should be added in the system for the proactive P2V warning. The detection unit should be installed at the roadside to monitor the roads. The detection unit could be regarded as a Roadside Unit (RSU), similar to the RSU for the On-Board Unit (OBU) with Dedicated Short Range Communications (DSRC). The detection unit could be installed at an intersection to monitor the

crosswalks and other intersection areas. If a pedestrian is detected at a crosswalk of an intersection, the drivers on the corresponding entering approaches which have potential conflicts should receive the warning. Figure 74 shows a typical 4-leg intersection. For the crosswalk marked in blue, the Approach 1 right-turn vehicles, Approach 2 through vehicles, and Approach 3 left-turn vehicles could have potential conflicts. If a pedestrian presents on the crosswalk marked in blue, the drivers from these three approaches should be warned ahead. On the other hand, the detection unit could be installed at a segment to monitor the jaywalking pedestrians if the segment is a hotspot of pedestrian crashes. If a jaywalking pedestrian is detected, all approaching vehicles should receive the warning messages.



Figure 74. Illustration of P2V warning at an intersection

Hence, the system is composed of the following three parts:

1. Detection Unit: detecting pedestrians and sending information to the cloud server in real time. This unit is similar to the RSU of DSRC.

2. Smartphone app: collecting drivers' location information and uploading information to the cloud server; receiving messages from cloud server and displaying warning. The app is an OBU emulator.
3. Cloud server: receiving data from both the detection units and smartphones; identifying the potential risks and sending the warning to smartphones. The server is a computing center. Meanwhile, the internet is for data communication, similar to the DSRC.

5.1.2. The Real-Time Pedestrian Detection System

As discussed above, it is required that the detection unit could provide the real-time pedestrian presence information to the server. Generally, the detection unit includes three main parts: (1) sensor; (2) computing unit; (3) communication unit. The sensor could be a camera, lidar, or radar, which provide raw data. The raw data have different formats for different sensors. For example, it could be a live video from a camera and a point cloud from a LiDAR. Although the size of the raw data is big, the raw data could not directly tell whether a pedestrian is present or not. Instead, a computing unit is needed to process the raw data from the sensor to detect the presence of pedestrians. The processed data are much smaller than the raw data, and could be uploaded to the server through the communication unit (internet device) with much smaller delays. Hence, this section presents the selection of appropriate sensors and setup of the detection system.

5.1.2.1. Selection of Appropriate Sensors

In the recent years, there are several emerging sensors that could better detect objects. The camera and LiDAR are two of the most popular sensors and have been widely applied in the fields of autonomous vehicles, roadway monitoring, etc. In this task, the two types of sensors were compared and the most appropriate sensor for this project was selected.

LiDAR is an exceptional device as it can literally provide a 3D view of the world as the infrared pulses bounce off surrounding objects and return to the sensor with precise distance information. In the market, the major LiDAR vendors are Quanergy, Velodyne, LeddarTech, etc. The research team purchased the Quanergy M8 LiDAR (Figure 75) and tested its feasibility. The M8 LiDAR is a 360-degree long range LiDAR sensor enabling ubiquitous uses of smart sensing in dynamic situations-made and tested for 3D mapping, security, and harsh industrial environments. The M8 has a detection range that exceeds 200 meters on high reflectivity surfaces. The M8 has been used at moving objects such as autonomous vehicles and Unmanned Aerial Vehicles (UAV) for the detection. However, the sensor has not been applied at a fixed position to monitor traffic.



Figure 75. Quanergy M8 LiDAR

In this task, the M8 LiDAR was mounted at a high tripod along a road to monitor the traffic (Figure 76). It required a power supply, a computer to process the 3D points data, and cables to connect the LiDAR with the power supply and computer.



Figure 76. Setup for the M8 LiDAR

The Qortex software provided by the Quanergy company was used to process the data. The detection data is shown in Figure 77. It shows the data points from the LiDAR are not very promising. Only eight layers of data points could be provided by the LiDAR. Although some moving objects could be detected, it is difficult to distinguish whether the moving objects are vehicles or pedestrians. Hence, the LiDAR would not be applicable to detect pedestrians at a fixed location, which is required by the detection system in this project.

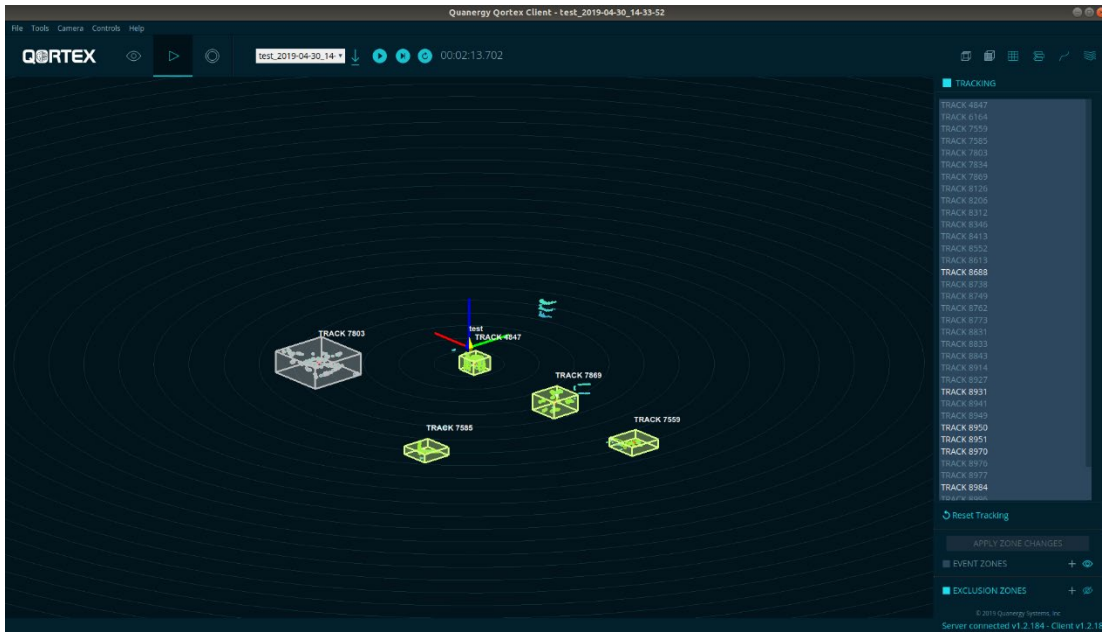


Figure 77. Screenshot of Qortex interface with LiDAR detection results

The camera could provide a live video data, from which pedestrians could be detected using computer vision methods. In the recent years, computer vision has been gaining a lot of attention and the video-based detection technology has been widely used in the fields of autonomous vehicles, traffic monitoring, security, etc. On freeways, cameras have usually been used to provide live videos for monitoring traffic. At intersections, the traffic parameters such as queue length and vehicle movement counts could be obtained by cameras. Figure 78 illustrates an example of a view of an intersection from a camera mounted at an intersection at UCF. The camera could provide a good view about what is happening in real time. With advanced computer vision technologies, pedestrians could be detected.



Figure 78. Screenshot of video from the camera

This study is mainly focused on detection algorithms with a roadside-mounted camera. A good video-based detection program should run in real time with the resistance to common problems like appearance of shadow, camera vibration and low visibility. Generally, the computer vision algorithms could be divided into the following two types:

1. Motion-based detection and segmentation
2. Deep learning-based object detection

The detailed literature review about computer vision could be found in the Task 1 report. For motion-based detection and segmentation approaches, there are three types: background subtraction methods, feature-based methods, and frame differencing and motion based methods. It was found that motion-based detection has shortcoming in detecting low-speed moving objects. On the other hand, deep learning methods are recently widely used due to its high accuracy and the rapid development of computer computation ability. Deep learning methods were divided into two-stage and one-stage approaches.

Two-stage algorithms, detection procedure contains two modules, region proposal and object classification:

1. Region proposal: this module is used for estimating the area in the image where the object is likely to be located.
2. Object classification: this module is used for classification of object categories in each area.

Convolutional Neural Network (CNN) is applied to identify the objects in the image, which is computationally expensive and difficult for the real-time detection.

One-stage approaches such as Single Shot Multi-Box Detector (SSD) (Liu et al., 2016d) and YOLO (Redmon et al., 2016; Redmon and Farhadi, 2018), combine two procedures together. This project takes YOLOv3 implemented in Keras (TensorFlow framework) as the pedestrian detection algorithm (Github, 2018). OpenCV (Bradski, 2019) is used to acquire video data from cameras. As Figure 79 (a) shows, it first resizes the input image to resolution of 448×448 (pixel) and runs CNN on the image. CNN is used for predicting bounding boxes, as well as object class in these boxes. Figure 79(b) shows the architecture of CNN. This CNN network has 24 convolutional layers followed by 2 fully connected layers. CNN could output class and the confidence score of predicted objects. Thus, compared with other deep learning detection techniques, YOLOv3 is an effective detection algorithm with the pedestrian classification and performs well with high average precision. As this task only focuses on detecting pedestrians, only the classification of pedestrians was enabled in the system.



Figure 80. Setup for the camera

The details of the devices used in the system are listed in Table 28. GPU GTX 1080 was used in the computer with an Ubuntu 18.04 LTS system to support the real-time pedestrian detection. A detection rate of 5 frame per second (FPS) (i.e., 0.2 second for a detection) was used in the system. Logitech Webcam was used as it can provide a live video stream. Besides, a USB cable with USB port 3.0 was used to ensure faster data transmission between the camera and computer. The detection results are uploaded to server in real time via a portable Wi-Fi device. The API was created in the server to accept the data from the computer.

Table 28. List of devices

Device	Model	System
Computer	ASUS ROG G703 GI (GTX 1080)	Ubuntu18.04 LTS
Web camera	Logitech C922x Pro Stream Webcam	-
Wi-fi portable devices	Netgear 4G LTE Mobile Wi-Fi Hotspot	-
USB cable	USB port 3.0	-
Tripods	-	-

5.1.3. Evaluation Experiments and Results

5.1.3.1. Experiment Design

A good detection program should run in real-time, as latency requirement for V2X (vehicle-to-everything) for safety is between 100ms and 1 second (Dey et al., 2016; Xu et al., 2017). The experiment was designed to test the detection accuracy of the algorithm and whether results could be received by the cloud server in real time.

Since it is easier to find pedestrians to cross the road at intersections, the experiments were conducted at intersections to validate the detection system more efficiently. Once the system is validated, it could also be applied at segments for detecting jaywalking pedestrians.

Intersections are recognized as dangerous locations due to the complexity and mixed-traffic environment. Crosswalks at intersections with zebra and pavement markings, are designed for pedestrians to pass the intersections safely. However, potential collisions could happen when drivers fail to yield for pedestrians, especially when the drivers are distracted or do not expect pedestrians. Since pedestrians would wait at sidewalks close to the crosswalks for the signal light, the detection area should include both crosswalks and surrounding sidewalks.

Figure 81 illustrates three detection zones (i.e., Zones 311, 312, and 342) for pedestrians at an intersection. Corresponding to the three zones, Approaches 11, 12, and 13 are approaches of coming vehicles which may have potential conflicts with the pedestrians on the zones. Red

stars denote potential conflict points between vehicles and pedestrians on certain directions. Black dotted lines denote the supposed trajectories of vehicles. For each detection zone, the computer will upload the number '1' to the server if pedestrians are detected. Otherwise, the number '0' will be uploaded to the server.

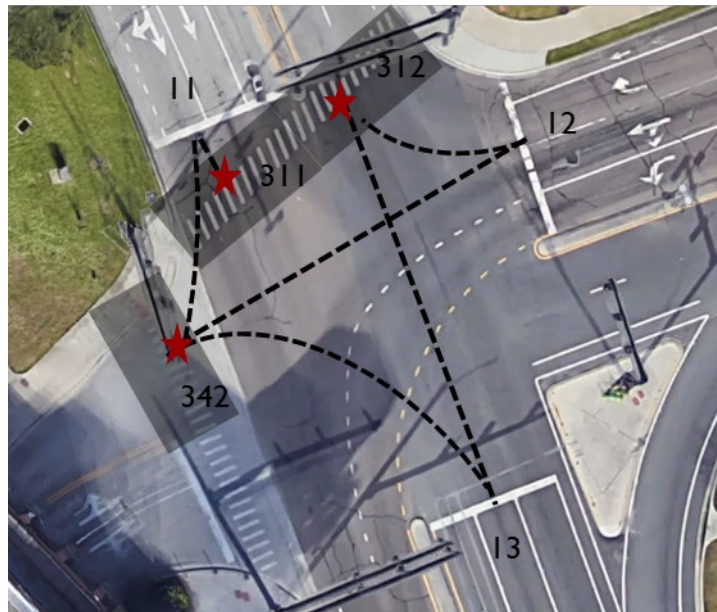


Figure 81. Detection zones and conflict points (spatial)

* 311, 312, 342: Detection zone

11, 12, 13: Vehicle approach

★: Potential conflict points

---: Vehicle moving trajectory

The experiments were designed to be carried out on three intersections, two were on the UCF campus and one is off campus, as listed as below:

1. Location 1-Intersection Gemini Blvd & Orion Blvd (on UCF campus): four-lag intersection with two crosswalks.

2. Location 2-Intersection Gemini Blvd & Hydra Ln (on UCF campus): three-lag intersection with three crosswalks.
3. Location 3-Intersection Research Pkwy & Libra Drive (off campus): four-lag intersection with four crosswalks.

Figure 82 shows studied locations on the Google Map.

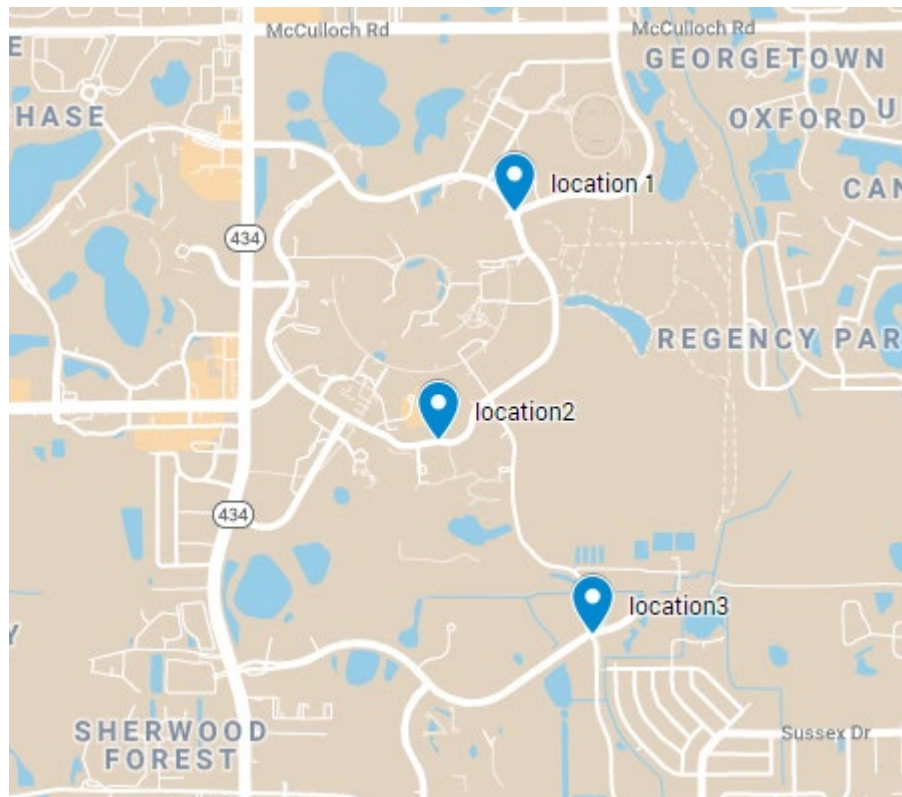


Figure 82. Experiment locations on map

5.1.3.2. Experiment Results

Figure 83 shows some screenshots of the detection results. As discussed above, the algorithm classifies detection zones as “1” (has pedestrian) and “0” (does not have pedestrian). The blue boxes indicate the bounding boxes for the pedestrians. It shows that the system could accurately identify pedestrians. For the corresponding zones, the cloud server will receive values “1” with zone number to indicate the presence of pedestrians.



(a) location 1



(b) location 2



(c) location 3

Figure 83. Screenshots of detection results

As the detection can be regarded as a binary classification problem, sensitivity, specificity and accuracy were used for the evaluation. As shown in Equation 1, Equation 2 and Equation 3, sensitivity measures how good the model is among all the positives, i.e., proportion of actual positives that are correctly identified by the model. Specificity measures the proportion of actual negatives that are correctly identified by the model. Accuracy value measures the proportion of true positives and negatives in all detected results.

$$\text{Sensitivity} = \frac{TP}{TP+FN} \quad \text{Equation 1}$$

$$\text{Specificity} = \frac{TN}{TN+FP} \quad \text{Equation 2}$$

$$\text{Accuracy} = \frac{TP+TN}{TP+FP+FN+TN} \quad \text{Equation 3}$$

The output videos at the three intersections were sliced into images for counting related parameters. And detection results of crosswalk zones are categorized as positive (“1”) and negative (“0”). Ground truth were categorized as “has person” and “does not have person” by manual observations. “True positive”, “False positive”, “False negative”, and “True negative” are four outcomes formulated in a 2 × 2 contingency table, as shown in Table 29.

Table 29. Diagram for metrics calculation

	Ground truth	
Detection result	Has person	Does not have person
Positive	True positive(TP)	False positive(FP)
Negative	False negative (FN)	True negative(TN)

As Table 30 shows, there are totally 962 detection observations collected from the three intersections. The detection results with different pedestrian presences are listed. Measures including sensitivity, specificity, and accuracy were calculated below. The sensitivity value of 0.831 indicates that 83.1% cases that pedestrians present at the detection zone could be detected. Besides, specificity value of 0.978 denotes there is seldom false alarm. And accuracy value indicates the output results of the algorithm is correct under 90.6% cases. Note that the metrics used here are usually used in statistics for binary classification problems. The evaluation results suggested the real-time pedestrian detection system could detect pedestrians in zones of interest

with a high accuracy. The evaluation results also apply to other areas such as segments and other zones of intersections.

Table 30. Experiment result of detection algorithm

Detection result	Ground truth		Total
	Presence of pedestrians (1)	No pedestrian (0)	
Presence of pedestrians (1)	388	11	399
No pedestrian (0)	79	484	563
Total	467	495	962
Measurement	Sensitivity= $388/467=0.831$	Specificity= $484/495=0.978$	-
	Accuracy = $(388+484)/(388+11+79+484) = 0.906$		

The latency of uploading the detection result to the server was also tested. It is suggested that the latency of data transmission is less than 200ms with the wireless internet. The latency could be even reduced if the wired internet or the coming 5G internet. Meanwhile, the detection frequency of camera is 5HZ and pedestrians could be detected before they enter the locations where they could have potential conflicts with vehicles. Hence, the develop prototype of real-time pedestrian detection system could meet the latency requirement of P2V (Pedestrian-to-Vehicle) for safety.

5.1.4. Summary

In this chapter, the structure of P2V warning system based on complementary sensors was presented. Two emerging sensors (i.e., LiDAR and camera) were validated for detecting pedestrians. It suggested that the camera could provide better detection results and YOLO computer vision algorithm was selected to detect pedestrian in real time. A prototype of a real-time pedestrian detection system based on a camera was proposed. Data communication between the detection system, the cloud server as well as smartphones were discussed. Experiments were conducted to validate the detection accuracy and latency. The results suggested that the suggested

detection system could reach a good accuracy and the latency could meet the safety requirement of P2V applications.

5.2. Video-Based Queue Detection System

Vehicular queue length is an important measurement for vehicle delay and signal performance at signalized intersection (National Research and Transportation Research, 1996; Webster, 1966). Meanwhile, a rear end crash could happen if a driver approaching an intersection fails to realize that a queue is ahead. Hence, queue warning is an important part of an Intersection Collision Warning System (ICWS). Similar to the P2V warning logic, the queue warning (a type of I2V (Infrastructure-to-Vehicle warnings) could be sent to drivers if a long queue is detected at an intersection. Traditional approaches use physical sensors like loop detector to detect the high occupation rate of road lane, thus identify the existence of vehicle queue. In this task, the research attempted to extend the computer vision algorithm to detect the existence of long queue through cameras.

5.2.1. Queue Detection Algorithm

The camera could cover an entering approach of an intersection. Then, as illustrated in Figure 84, virtual detectors could be set at the upstream from the stop line. When the vehicle enters the detector, the detector could be detected as “occupied” with the computer vision detection algorithm. Else, the detector is detected as “unoccupied”. When one detector is occupied in a continuous several seconds, it could be regarded as that the vehicle stops at the virtual detector locations, which means the queue has reached the detection location. In our study, a threshold of 3 seconds and a detection rate of 5 FPS was used to determine if a vehicle stops. Meanwhile, the

queue beyond the detection location could be simply calculated considering the arrival rate at the intersection. In this task, a rate of 280 vehicle/h was used for the test intersection.



Figure 84. Virtual loop detector (spatial)

Figure 85 shows the screenshot of queue detection system. It suggests that the system could successfully detect there was a queue on the middle lane. The queue warning would be sent to the approaching drivers' smartphones.



Figure 85. Screenshot of vehicular queue detection algorithm

*FPS: frame numbers processed per second.

Lane occupied: “0” denoting “non-occupied”, “1” denoting “occupied” for each lane

Queue length: estimated queue length for each lane.

Queue warning: “no” denoting “no queue”, “yes” denoting “has queue” for each lane.

5.2.2. Evaluation Experiment and Results

An experiment was conducted at the west bound approach at Intersection Gemini Blvd & Orion Blvd (on the UCF campus) during the morning peak time on June 6th, 2019. The setup of the detection system is same as the pedestrian detection system. The detection results were uploaded and saved to the cloud server. Meanwhile, the traffic was recorded in a video and the video was sliced into images. The queue status in each image was determined by manual observations and compared to the data saved in the server. The results are summarized in Table 31. A total of 6,000 images, among which 1,001 (16.7%) images had queues. The queue status was determined when the queue was beyond the virtual detector location (vehicle numbers are equal to or larger than

4). It was suggested that the algorithm could detect queue in 706 images and the sensitivity value was 0.705. On the other hand, there was no long queue in 4,999 images, of which 4,759 images could be correctly identified by the algorithm. Thus, the specificity rate was 0.952. The accuracy for total positive and negative cases was 0.911. It could be concluded that the suggested detection algorithm could provide an acceptable queue detection result. The latency of detection result communication is same as the pedestrian detection system, which could meet the requirement of I2V applications.

Table 31. Experiment result of queue detection algorithm

Detection result	Ground truth		Total
	Has long queue (1)	No long queue (0)	
Has long queue (1)	706	240	946
No long queue (0)	295	4759	5,054
Total	1,001	4,999	6,000
Measurement	Sensitivity=706/1001=0.705	Specificity=4759/4999=0.952	-
	Accuracy = (706+4759)/ (706+240+295+4759) = 0.911		

5.2.3. Summary

In this chapter, a real-time video-based vehicular queue detection system was proposed at signalized intersection by extending the pedestrian detection system. The queue status could be detected according to the occupancy status. Evaluation experiments were conducted at an intersection. The results suggested that the algorithm could provide a good queue status detection in real time.

5.3. Conclusions

In this task, the research team has developed a prototype of a real-time pedestrian detection system and vehicular queue detection system with an external complementary sensor. Two

emerging sensors including LiDAR and camera were evaluated for the detection at fixed locations and the camera was finally selected. The prototype of a real-time detection system was set up. Extensive experiments were conducted to evaluate the proposed system. The results suggested that the detection system could detect pedestrians and queue statuses at an accuracy of over 90%. The cloud server was utilized to realize the communication between external video sensor and drivers with smartphone (application). Meanwhile, detection results could be sent to the server in real time with a latency less than 200ms. Hence, the suggested pedestrian detection system could proactively detect the pedestrians in real time even the pedestrians do not have the developed app. Meanwhile, the system could successfully identify the queuing status at intersections.

CHAPTER 6. DEVELOPMENT OF PRELIMINARY SMARTPHONE-BASED I2V APPLICATION

In Chapter 4, the research team has developed the smartphone app and set up a cloud server to collect different smartphone sensor data to understand the users' statuses including position coordinates, locations (e.g., intersection or curve), speeds, transportation modes, accelerations, and turning movements. Meanwhile, a prototype system with cameras was proposed in Task 5 to automatically detect the presence of queue at intersections in real time. Based on the data obtained from Tasks 3 and 5, the research team developed the app to use the smartphones to emulate On-board unit (OBU) for the I2V warnings. Two I2V applications were developed including: (1) curve warning, and (2) queue warning.

Section 6.1 describes the design of database for the two I2V warning applications. Historical and real-time databases are set up for different data types. Different data from smartphone and cameras are collected in the server and utilized for the applications.

In Section 6.2, the I2V warning logic is presented for the curve warning and queue warning. Different types of warning messages are suggested to ensure sufficient warnings. Multiple experiments are conducted to evaluate the efficiency of the developed I2V warning applications.

Section 6.3 concludes the smartphone I2V warning applications and experiment results. It suggests the developed app could successfully emulate the OBU for the I2V warnings.

6.1. Preparation of Database

Lane-departure crashes at horizontal curves represent a significant portion of fatal crashes. Highway curves account for 25 percent of all highway fatalities in the US and produce an average crash rate three times of other highway segments (Pisano et al., 2008). Hence, solutions are needed to aid drivers in identifying upcoming curves and suggesting them of a safe speed.

The most common method for warning drivers about hazardous horizontal curves is with infrastructure-based systems ranging from standard curve warning signs (Lusetti et al., 2008) to sensor-triggered dynamic warning displays (Glaser et al., 2007). Signing is a common warning method for curves and commonly includes curve warning, advisory speed, and chevron signs. While signing curves can help, static signage is frequently disregarded by drivers and can only offer support to drivers who get alerts and look for curve information. Besides, the coverage of infrastructure-based systems is limited due to the cost of installing the infrastructures.

This task seeks to determine the feasibility of in-vehicle dynamic curve-speed warnings as deployed on the smartphone app. The selected driver warning methods must demonstrate effectiveness without distracting the driver from safely navigating through the curve. To maximize efficiency, the system designed in this study incorporated human factors principles to result in high usability and trust in the delivery and content of the warnings. The system was incorporated into a smartphone app capable of displaying warnings to drivers based on their distance to the curve.

Rear-end crashes should be the most common crash types among the multiple vehicle involved crashes. Rear-end crashes usually occur if vehicles approach downstream queueing traffic. This task also developed the application of queue warning to alert drivers so that they can

avoid rear-end collisions at intersections. The queue detection system architecture with cameras has already been discussed in detail in Task 5. By using computer vision technologies, virtual loops are created to detect the presence of queue. This information is then processed and uploaded into the cloud server in real time. This task will discuss the details regarding how this warning is processed in the server and displayed on the smartphone app. Noteworthy, the queue warning application could also be extended to segments as long as queueing status could be detected.

The server needs to have proper information in order to send warnings to any smartphones. Hence, the team used two databases in the cloud server (Figure 86):

1. Historical Database
2. Real-time Database

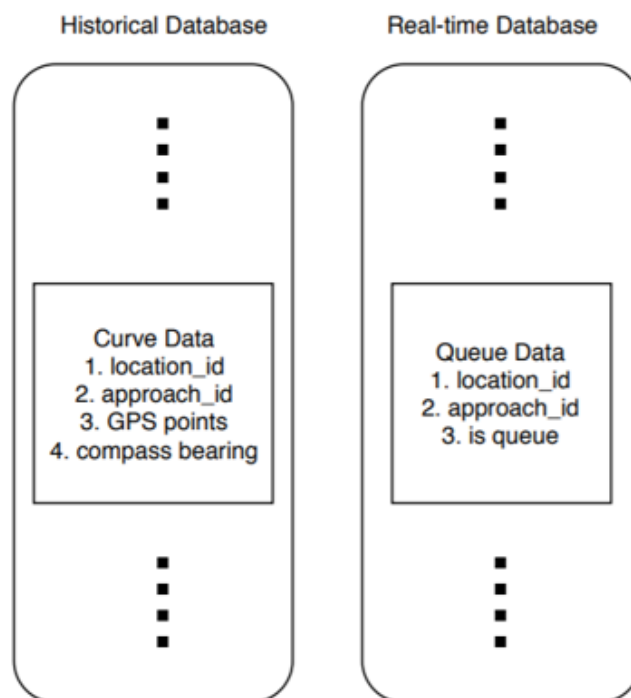


Figure 86. Structure of the database

The Historical Database contains information about roadway geometry such as information about curves, which is expected to remain the same over a long period of time. It contains information about the curve speed, curve radius, and minimum distance to send the warning. Data was collected based on ArcGIS. GPS points were collected at an even interval. Creating this database reduces the dependency on other paid online map services (i.e., Google Maps, HERE, etc.).

The Real-Time Database, on the other hand, contains information that needs to be updated very often, sometimes every second. The proposed system updates this database every half of a second as mentioned in Chapter 3. The information about current traffic conditions such as queue can be saved in this database. As the new queue information is uploaded from the local detectors (i.e., cameras), the information gets replaced. In Figure 73, the dots indicate that this database can be extended to include other historical or real-time data for other I2V applications.

Figure 87 shows how a curve is mapped in the database. For most cases, vehicles should approach a curve from the upstream of the roadway. However, there might be other accesses for vehicles to enter roadways. Hence, high-resolution location information is prepared to guarantee that all drivers could receive the curve warning. The points are collected for each direction separately. While it would involve less work to take points along the median of the road, it would be difficult to identify which direction a vehicle is approaching the curve. Also, proper signs need to be shown to drivers regarding the direction of the curve. By using the localization algorithm proposed in Chapter 4, whether a vehicle is approaching a curve could be determined in real time in the server.

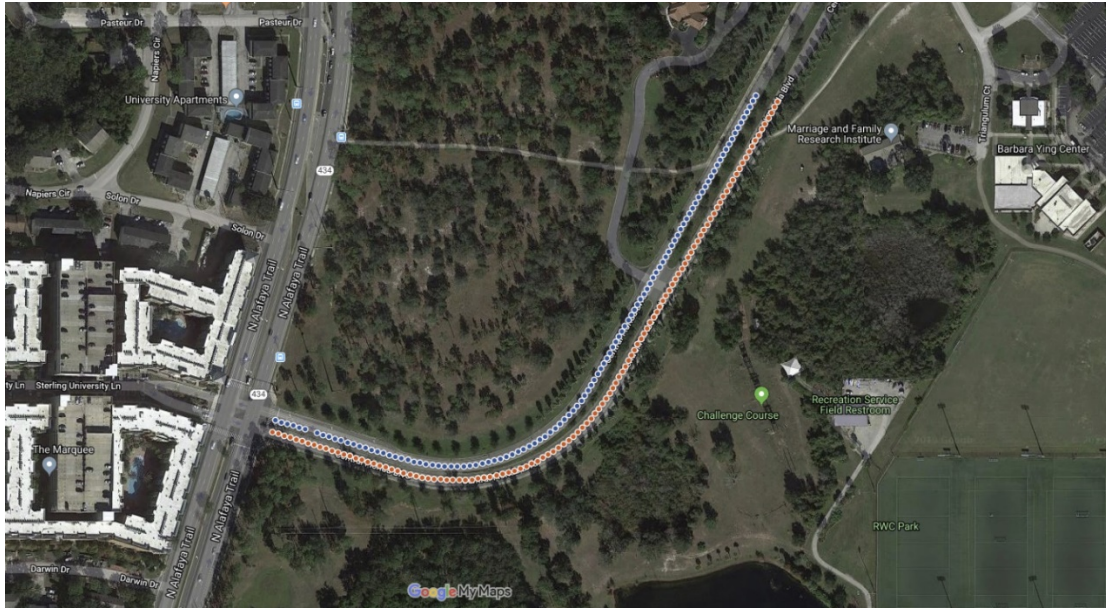


Figure 87. An example of a curve data in the database

6.2. Development of Warning Logic

6.2.1. Description of Data Transfer between Smartphone and Server

To ensure the efficient warning applications, it is of immense importance to keep the data transfer between the smartphone app and the server to a minimum. This is because that the time taken to transfer data is significantly higher (average latency is 160ms as mentioned in Chapter 3) than the time required for computations in the server or smartphone alone.

To ensure the data communication between the smartphone and server, it is essential to understand the data communication mechanism and establish an efficient way to send and receive data. Figure 88 shows how this is accomplished. The smartphone data sent to the cloud server also contains a field called the “warning_state”. This is usually null (no warning). However, the state of this field could be changed in the server to a warning such as ‘Curve Ahead’. The server can only make a decision if other calculations can be completed as swiftly as possible. The approach used was innovative in the way that the research team tried to identify and send the

warning in the client-server communication protocol. Additionally, this approach could help identify the warning as soon as new data is available.

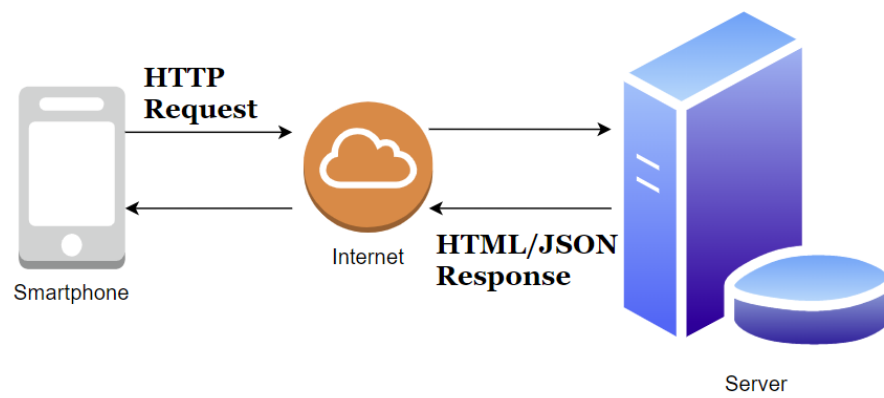


Figure 88. Client-Server communication

6.2.2. Warning Estimation in Server

Once smartphone data is received by the server, it takes the GPS points and localizes the corresponding users by using Radius Neighbor Classifier, which was suggested in Chapter 4. The localized points are matched with the curve database to see if the GPS points are in a curve. Once this is ensured, it is necessary to ensure that warning is sent out at the right time; not too early, not too late. For this task, the team used distance from the curve to send the warning. If the distance between the vehicle and the start of the curve reaches the threshold, the “warning_state” field is changed to “Curve Ahead”. As shown in Table 32, the threshold is dynamic as a function of speed limit, based on the Manual on Uniform Traffic Control Device (MUTCD) (FHWA, 2009).

It should be noted that similar logic could be applied for the “Work Zone Warning” if the locations of the work zone is available. If the plan of work zone could be connected to our server in the following phase, the “Work Zone Warning” application could easily be added. Hence, we developed instead queue warning which requires a different I2V warning logic.

Table 32. Distance as a function of speed

Speed	Distance
20 mph	115 feet
25 mph	155 feet
30 mph	200 feet
35 mph	250 feet
40 mph	305 feet
45 mph	360 feet
50 mph	425 feet
55 mph	495 feet
60 mph	570 feet
65 mph	645 feet
70 mph	730 feet
75 mph	820 feet

For the queue warning system, the camera sends information to the server regarding the position and existence of a queue. The server takes this information along with the location information from the smartphone and check if the vehicle is at the upstream of the queue. The driver will receive the queue warning message when these conditions are met.

6.2.3. Warning Display on the Smartphone

It is important to display the warnings to properly grab the drivers' attention but not so much as to distract his attention from the road (Ben-Yaacov et al., 2002). With this principle in mind, the team has decided to use both visual and auditory messages to warn drivers about curves (Scott and Grey, 2008). The driver can keep his eyes on the road and still receive warnings via the auditory message. Meanwhile, the smartphone display illuminates to show the proper warning and the direction of the curve.

The smartphone uploads sensor data and receives the response visa HTML/JSON response. It checks the field called "warning_state" to identify warnings and displays it on the

phone. Once the “warning_state” is null again, the warning disappears. The Figure 89 depicts this scenario.

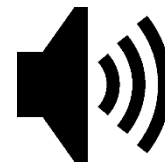


Figure 89. Screenshot of curve warning display in the smartphone app along with auditory warning message

The idea to displaying queue warning in the smartphone, which is similar to the curve warning. Figure 90 shows the screenshot taken when a queue warning was received at the driver’s smartphone.



Figure 90. Screenshot of queue warning displayed in the smartphone app

6.2.4. Results and Discussions

Experiments were conducted to validate the developed two I2V applications (i.e., curve warning and queue warning). Table 33 shows the summary of the experiments. The queue warning experiments were carried out at peak hours when there was a queue. The curve warnings were tested for a single curve but from both directions. It could be concluded from the experiments that the system has been successfully able to receive and display the warnings. Actually, the success of warning depends on the localization accuracy. As discussed above, the first computation of the

warning process is the localization. Once users' location is correctly clarified, all other aspects are only comparison of various data. Noteworthy, even though the localization algorithm may falsely classify some points, the accuracy of detecting a curve or intersection still does not fail because the curve is a series of GPS points, which means the proposed method could have the tolerance of certain false localizations. Figures 91 and 92 illustrate examples of curve warning and queue warning received in a car.

Table 33. Results of I2V warning system

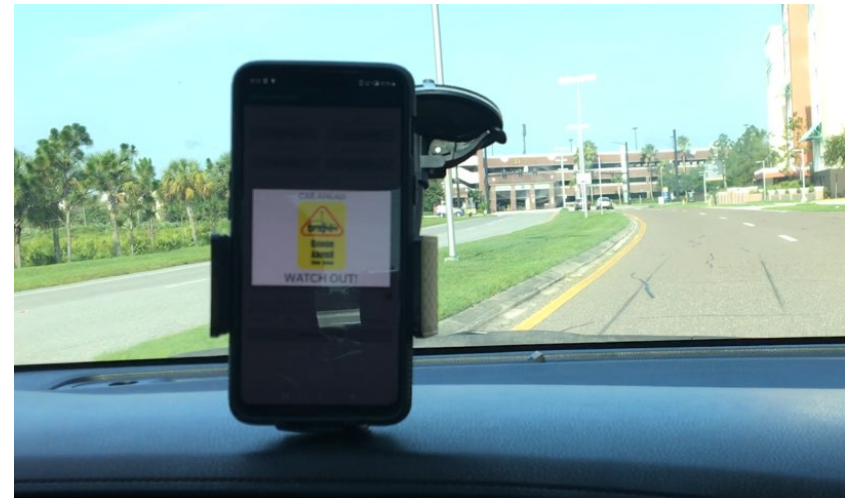
Warning Type	Location of Experiments	Number of Experiments	Number of Warnings Received Successfully
Queue	N Orion Blvd	12	12
Curve	Central Florida Blvd	7	7



Figure 91. Illustration of curve warning received in car



(a) Queue detection



(b) Illustration of queue warning received in car

Figure 92. Illustration of queue warning

6.3. Conclusions

In this chapter, the research team developed two I2V smartphone applications including curve warning and queue warning. Warning logics were proposed and programed in the server. The smartphone app was further programed to ensure receiving the warning from the server in real time. Experiments were conducted in field and it could be concluded that the developed system could successfully emulate the OBU for the I2V applications.

CHAPTER 7. DEVELOPMENT OF PROACTIVE SMARTPHONE-BASED P2V APPLICATION

In Chapter 4, the research team has developed the smartphone app and set up a cloud server to collect different smartphone sensor data. The data are used to identify users' statuses including position coordinates, location (e.g., intersection or curve), speed, transportation modes, acceleration, and turning movements. Meanwhile, a prototype of a detection system based on a roadside camera was proposed in Task 5 to proactively detect pedestrians' presence at intersections in real time. In this task, the research team developed the smartphone app and server to emulate the On-Board Unit (OBU) for the P2V warning. The system is based on the models and techniques embedded in the smartphone and relevant system components.

Section 7.1 describes the P2V warning logic based on smartphones. Two methods are developed for intersections and road segments, separately. The first one is designed using a basemap-based method, while the second one is achieved based on the proximity method.

Section 7.2 illustrates the warning logic of the proposed P2V system based on cameras. With the detection unit embedded with a web camera, potential conflicts between pedestrians and vehicles can be identified. Drivers on specific oncoming approaches can receive the warning in real time.

In Section 7.3, extensive experiments are conducted to test the accuracy and efficiency of the developed P2V warning system. Experiment results suggest that the developed system could successfully emulate OBU for the P2V warning.

Finally, Section 7.4 concludes the smartphone P2V warning applications.

7.1. Smartphone-Based P2V Warning Logic

In Chapter 4, the smartphone data reflecting users' statuses such as location, transportation modes, movement, and speed could be obtained and uploaded to the cloud server. Based on the smartphone data, the research team developed two different methods for the smartphone-based warning system. The first one uses the basemap, while the second one is based on the proximity between the road users. These two methods are designed for intersections and road segments, respectively. The intersections usually have more complicated traffic environment than road segments. For example, a vehicle may have different movements at an intersection, such as going straight, turning left, and turning right. Vehicles of different movements could have different potential conflicts with pedestrians at different crosswalk locations. Thus, it is better to utilize the basemap to divide the intersection into different parts. The implementation of basemap could help the system send warning more accurately and specifically. For the road segment, the vehicle usually has simpler movements since it can only go straight. A vehicle could hit a pedestrian when the pedestrian is walking along the roadside or crossing the segment such as walking at a mid-block crossing or jaywalking. A proximity-based method is used to identify potential conflicts on segments.

7.1.1. Smartphone-Based P2V Warning Logic Based on Basemap

P2V warning system using basemap is developed for intersections, which have relatively complicated conditions with mixed traffic. Three major modules in the cloud server are developed accordingly, which are vehicle localization, vehicle movement detection, and conflict pair module.

At the beginning, the vehicle/pedestrian is localized on the basemap according to the GPS information from the smartphones. Then, the movement of the vehicle is identified based

on the pre-trained machine learning model developed in Task 3. Finally, warning will be sent if the localization and movement information match the pre-defined potential conflict pair list.

A basemap in the system is the collection of several GPS points stored in the cloud server, which could help localize the users. One example of intersection basemap is shown in Figure 93. For vehicles, the roads heading to an intersection are divided into two sections, which are 'Approaching Intersection' and 'At Intersection'. Each one is assigned a unique two-digit 'Approach ID'. For example, the numbers '11' and '12' in Figure 93 indicate 'At Intersection' and 'Heading to Intersection' for one road. For pedestrians, a crosswalk at intersection is divided into two parts. Each part has a unique three-digit 'Approach ID' to distinguish from the vehicle approach. For example, '311' and '312' represent two parts of one crosswalk. It should be noted that the proposed system uses the crosswalk as a prototype. The proposed logic could also be extended to pedestrians at other locations as long as the map information was developed and saved in the cloud server.

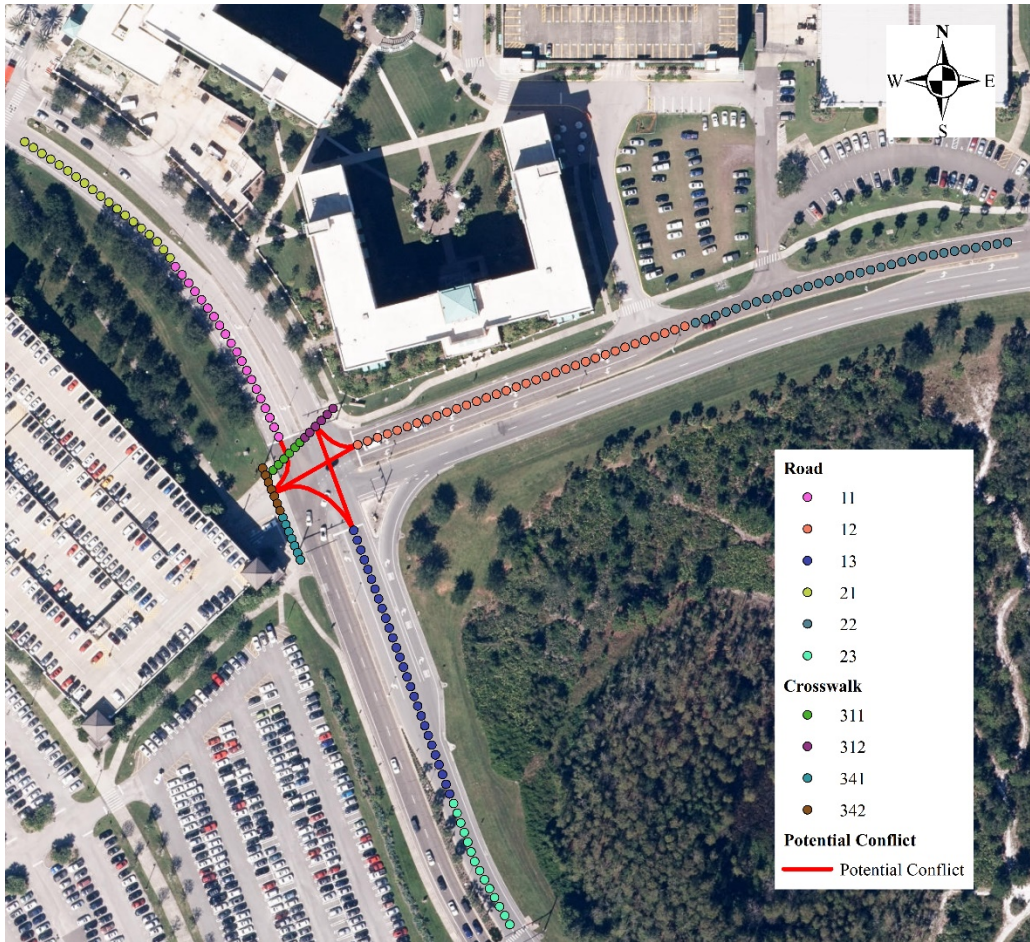


Figure 93. Illustration of base map and potential conflicts

Another important component of the warning system is the potential conflict pair module. Each potential conflict pair has four attributes, which are ‘Intersection ID’, ‘Approach ID (pedestrian)’, ‘Approach ID (vehicle)’, and ‘Movement ID’. The potential conflict pairs are shown by the red lines in Figure 93. Examples of the conflict pairs are shown in Table 34. For example, the vehicle turning left from approach ‘13’ could have a potential conflict with the pedestrian on the crosswalk ‘342’.

Table 34. Potential conflict pairs

Approach ID (pedestrian)	Approach ID (vehicle)	Vehicle Movement
311	11	Right Turn
312	12	Right Turn
342	13	Left Turn
312	13	Through

7.1.2. Smartphone-Based P2V Warning Logic Based on Proximity

For road segments, the research team designed a P2V warning logic based on proximity between vehicles and pedestrians. The proximity could be estimated based on their GPS data in the server. Once the distance is less than a certain threshold, the warning message will be sent to both the driver's and pedestrian's smartphones. In our study, the distance of 300 feet was selected as the threshold. In the future phase of the project, the threshold needs to be customized according to different factors such as vehicles' distance, speed, and roadway types. During the experiments, a driver was asked to drive a vehicle along a segment while a pedestrian was walking along the segment (see Figure 94). Although the pedestrian was not crossing the segment, the test scenario could reflect the conditions when a pedestrian is walking along the roadside or crossing the segment (e.g., walking at a mid-block crossing or jaywalking).



Figure 94. Illustration of the experiment on a segment

7.2. Camera-Based P2V Warning Logic

To proactively detect pedestrians even when they do not have the developed apps, the research proposed a prototype of collision warning system with the roadside camera detection unit. In Task 5, the data stream and algorithms inside the system were illustrated. The prototype contains three major parts, detection unit, cloud server and smartphone app:

1. Detection unit: detecting pedestrians and sending information to the cloud server in real time. This unit is similar to the Road Side Unit (RSU) in a Connected Vehicle device.
2. Cloud server: receiving data from both the detection unit and smartphones; identifying the potential conflicts and sending the warning to smartphones. The server is a computing center. Meanwhile, the internet is for data communication, like the DSRC.
3. Smartphone app: collecting drivers' location information and uploading information to the cloud server; receiving messages from cloud server and displaying warnings. The app is an OBU emulator.

The locations of the vehicles are collected through the smartphone app developed by the research team, while the detection unit at the road side can capture the pedestrians' presence based on the video processing. These two kinds of messages are uploaded to the cloud server simultaneously in real time.

As illustrated in the previous section, the potential conflict pairs between vehicles and pedestrians are stored in the server. Each conflict pair has four attributes, which are 'Intersection ID', 'Approach ID (Pedestrian)', 'Approach ID (Vehicle)', and 'Movement ID'. The detection unit returns "Approach ID (Pedestrian)" to the cloud server and then the cloud

server pairs pedestrians and vehicles. If there is a potential conflict between them, warning will be sent to the driver through the app.

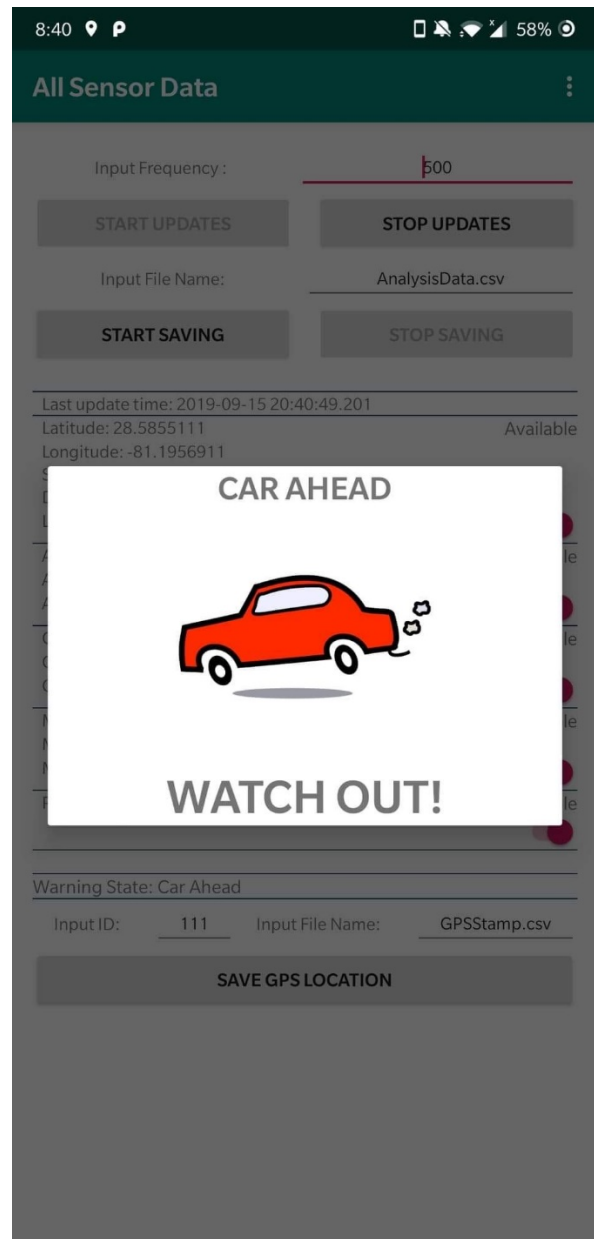
7.3. Experiments and Results

7.3.1. Warning Display on the Smartphone

The smartphone uploads sensor data and receives the response via HTML/JSON response. It checks the field called “warning_state” to identify warnings and displays it on the phone. Once the “warning_state” is null again, the warning stops. Figure 95 shows screenshots of warnings. Specifically, Figure 95 (a) is the warning message sent to the driver, while Figure 95 (b) is the warning message sent to the pedestrian. If a potential conflict is identified, the warning could be sent by the server to drivers and pedestrians in real time.



(a) Warning message received by driver



(b) Warning message received by pedestrian

Figure 95. Screenshots of warning display on the driver's and pedestrian's smartphones, respectively

7.3.2. Experiment Design

The research team conducted extensive experiments to evaluate the performance of two P2V warning systems. As intersections are regarded as dangerous locations for pedestrians with mixed traffic, experiments are designed and conducted at three intersections at the University

of Central Florida's (UCF) main campus. Their locations are shown in Figure 96 and the detailed information is as follows.

1. Location 1-intersection at Gemini Blvd and Orion Blvd (on UCF campus): four-lag intersection with two crosswalks.
2. Location 2- intersection at Gemini Blvd and Hydra Ln (on UCF campus): three-lag intersection with three crosswalks.
3. Location 3-intersection at Research Pkwy and Libra Drive (off campus): four-lag intersection with four crosswalks.

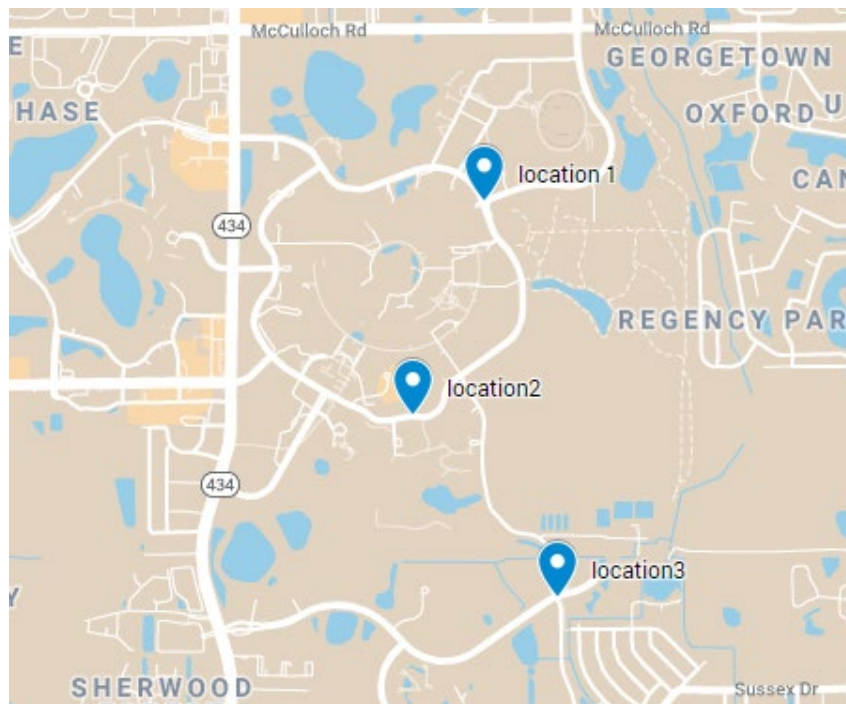


Figure 96. Experiment locations

At each intersection, the research team conducted 5 experiments for each vehicle movement, which are going through, turning left, and turning right. Both smartphone- and camera-based P2V warning systems were tested, separately. In total, there are 30 experiments at each intersection. The experiments are conducted from July to September 2019. Another 10 experiments were conducted on road segments based on the proximity method.

7.3.3. Experiment Results

The results of the experiments are summarized in Table 35. As shown in Table 35, among 93.3% of the basemap experiments, the warning could be successfully sent. Besides, the proximity based method could send the warning successfully for all cases (100%). The results are expected since the proximity method is relatively straightforward. For camera-based P2V warning, 45 experiments are conducted, while drivers receive warnings successfully in 42 cases (95.6%). Hence, it could be concluded that the two P2V systems achieved high accuracy from the perspective of warnings.

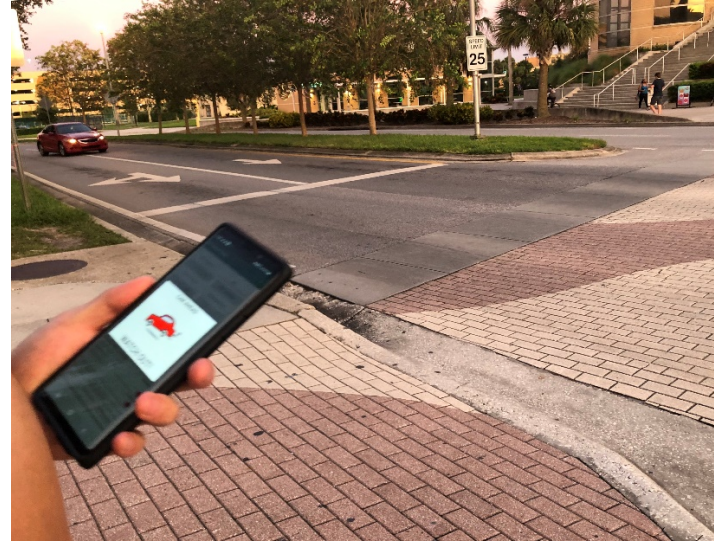
Table 35. Experiment results

Method	Movement	Number of Experiment	Number of Successful Warning	Successful Warning Rate	Reason
Basemap	Left-turn	15	13	86.7%	Activity recognition failure
Basemap	Through	15	14	93.3%	Vehicle localization failure/ Activity recognition failure
Basemap	Right-turn	15	15	100%	
Camera	Left-turn	15	13	86.7%	Vehicle localization failure
Camera	Through	15	15	100%	
Camera	Right-turn	15	15	100%	
Proximity	Segment	10	10	100%	

Furthermore, some examples of the experiment results are shown in Figure 97. Specifically, Figure 97 (a) and (b) show the warning messages received by the driver and pedestrian, while Figure 97 (c) and (d) show screenshots of the camera detection results.



(a) Vehicle warning



(b) Pedestrian warning



(c) Screenshot of camera detection result



(d) Screenshot of camera detection result

Figure 97. Experiment results

Moreover, the reasons of warning failures are concluded in om the perspective of warnings.

Table 35. Activity recognition failure and vehicle localization failure are the major two reasons. The activity recognition failure mainly happens when the API provided by Google Map failed to recognize the traffic mode of the road user. The research team plans to improve the activity recognition algorithm in the future phase to better identify the traffic mode. For localizing the vehicle, one reason of failure is the difference in the GPS chips between different smartphones. This problem could be resolved in the future as more advanced smartphones tend to have better localization accuracy. Another reason of failure is the radius for localization is too small since the GPS points fluctuate sometimes. The research team also plans to improve the localization module accordingly.

7.4. Conclusions

In this task, the research team proposed the P2V warning applications using the smartphone and camera. Warning logics based on traffic conflict identification were proposed under two conditions (i.e., smartphone only and smartphone & camera) and programmed in the cloud server. Totally 100 experiments were conducted at intersections and road segments to validate the developed system. It suggested that the developed system could identify the potential conflicts between vehicles and pedestrians with high accuracy and send the warning in real time. Through the completed tasks, it could be concluded that the developed smartphone applications and the warning system could successfully emulate the OBU for the P2V applications.

CHAPTER 8. SUMMARY AND CONCLUSIONS

8.1. Summary

In this research project, there were six major objectives. All the main objectives of this project have been achieved as follows:

1. Review studies and practice about smartphone-based communication and OBU applications

The research team has conducted a comprehensive literature review of the existing studies and practices that are relevant to the smartphone sensors, roadside detection sensor, data communication technology, On-board unit (OBU) application, etc. (Chapter 2)

2. Test the feasibility of using smartphones as OBU emulators

In Chapter 3, the research team developed smartphone apps to collect smartphone sensor data. The data of five sensors including GPS, accelerometer, gyroscope, magnetometer, and barometer were included in the developed apps. The server was set up to realize the communication to enable smartphone apps to upload/receive data to/from the cloud server. A variety of experiments were conducted to test the feasibility of using smartphones as OBU emulators based on two measures: (1) latency of data transmission; and (2) battery consumption.

3. Explore smartphone sensor data

In Chapter 4, multiple methods were proposed to obtain different traffic parameters of smartphone users based on the extensive understanding of the smartphone data. Numerous experiments were conducted to evaluate the accuracy of traffic parameters including position, location, speed, transportation mode, turning movement, and acceleration rate.

4. Test other roadside sensors to proactively detect pedestrians and queue lengths

Chapter 5 presents a prototype of a real-time pedestrian detection system and vehicular queue detection system with an external complementary sensor. Two emerging sensors including LiDAR and camera were evaluated for the detection at fixed locations and the camera was finally selected. The prototype of a real-time detection system was set up. Extensive experiments were conducted to evaluate the proposed system for proactively detecting pedestrians and queue length.

5. Develop the smartphone-based OBU emulator for I2V warning

The research team developed two I2V smartphone applications including curve warning and queue warning in Chapter 6. Warning logics were proposed and programed in the server. The smartphone app was further programed to ensure receiving the warning from the server in real time. Multiple experiments were conducted in field to evaluate the developed I2V warning based on smartphones.

6. Develop the smartphone-based OBU emulator for P2V warning

In Chapter 7, the research team proposed the P2V warning applications using the smartphone and camera. Methods were proposed to identify potential conflicts between pedestrians and vehicles at intersection and segments. The methods consider two different conditions, which are the condition when both drivers and pedestrians have the developed apps and the condition when drivers have the smartphone apps and pedestrians were proactively detected by the camera. Warning logics based on traffic conflict identification were proposed and programmed in the cloud server. Totally 100 experiments were conducted at intersections and road segments to validate the developed system.

8.2. Conclusions

The On-board unit (OBU) enables the in-vehicle communication by capturing information from multiple sensors and managing large amounts of data at high computation speed. An OBU is integrated in a vehicle for interaction with drivers by displaying warning, issuing alerts, offering automotive services, and managing the communication with a vehicle's surroundings. A lot of ongoing efforts are conducted to use OBU for the Infrastructure-to-Vehicle (I2V) and Pedestrian-to-Vehicle (P2V) applications. The applications could send early warning to drivers through OBU, which could help alleviate the misjudgment of the driver, avoid traffic crashes, and layout the efficient driving route. Given the current market penetration of OBUs, the benefits from OBU implementation are limited. The research team from the University of Central Florida attempts to take the benefits of OBU applications to a new level by using smartphones as OBU emulators.

First, the research team has reviewed a variety of existing studies and practices about the smartphone sensors, detection technologies, data communication methods, and OBU applications. The methods using smartphone sensors to detect users' traffic statuses and OBU applications have been well summarized. Based on the findings from the reviewed practices and materials, the important concepts should be considered for using smartphones to emulate OBU applications were summarized, which guided the following development of apps.

Subsequently, the feasibility of using smartphone as OBU emulators were validated. The research team developed smartphone apps to collect sensor data including data of GPS, accelerometer, gyroscope, magnetometer, and barometer. The server was set up to realize the communication between the developed apps and the server. The latency of data transmission was examined under the conditions of different data communication frequency, transportation modes, and speed. Besides, battery consumption of different sensors was explored. The results

suggested that the latency of data communication could meet the requirement of vehicle to everything (V2X) applications. Besides, smartphones could have enough battery capacity for using the developed apps. Hence, it could be concluded that it is feasible to use the smartphone to emulate the OBU.

The designed system with the cloud server and smartphone apps was further enhanced to upload and save the sensor data. The smartphone sensors were used to compute different traffic parameters including positions, speeds, localizations, transportation modes, accelerations, and movements. The computation is summarized in Table 36. Different traffic parameters were obtained by using different smartphone data. Four traffic parameters including position, speed, transportation mode, and acceleration are computed in the smartphone app, and then uploaded to the server. On the other hand, the localization and movement are determined in the cloud server once the smartphone data are uploaded. Except for the position and speed parameters, multiple models or methods were proposed to obtain traffic parameters. Numerous experiments were conducted to evaluate the accuracy of the obtained traffic parameters. The evaluation results indicated that the all obtained traffic parameters could reflect users' statuses with good accuracy.

Table 36 Summary of traffic data computation with smartphone data

Traffic parameters	Required data for computation		Computation location	Method	Latency (ms)
	Smartphone data	Other data			
Position	GPS coordinate	-	Smartphone app	Directly obtained	instantaneous
Speed	GPS speed	-	Smartphone app	Directly obtained	instantaneous
Localization	GPS coordinate, compass bearing	Base map	Cloud server	Radius Neighbor Classifier	3
Transportation mode	Accelerometer, gyroscope, GPS speed	-	Smartphone app	API from Google and Apple	instantaneous
Acceleration	Accelerometer, gyroscope, compass bearing	Rotation matrix	Smartphone app	Coordinate Reorientation	instantaneous
Movement	Accelerometer, gyroscope, and GPS speed	-	Cloud server	Random forest classification method	3

In addition to the smartphone data, the research team developed a prototype of a real-time pedestrian detection system and vehicular queue detection system with an external complementary sensor. The camera was used as the external sensor to set up the real-time detection system. Computer vision technologies were applied to detect the presence of pedestrians and queue length in real time. Multiple experiments were conducted to evaluate the proposed system. The results suggested that the detection system could detect pedestrians and queue statuses at an accuracy of over 90%. The cloud server was utilized to realize the communication between external video sensor and drivers with smartphone apps. Besides, it was confirmed that the detection data could be sent to the server in real time with a latency less than 200ms. The results indicated that the developed pedestrian detection system could proactively detect the pedestrians even the pedestrians do not have the developed app. Meanwhile, the system could successfully detect the queuing status at intersections.

Based on the data from smartphone and external cameras, the research team first used smartphone as OBU emulators for two I2V applications including curve warning and queue warning. Warning logics for both applications were proposed and programed in the server. The smartphone app was further programed to ensure receiving the warning from the server in real time. Experiments were conducted in field and it could be validated that the developed system could successfully emulate the OBU for the I2V applications.

Finally, the research team developed the apps to emulate OBUs for the P2V warning applications. The potential conflicts between vehicles and pedestrians are identified under two conditions: (1) both drivers and pedestrians have the developed apps and could receive message from the cloud server; (2) only drivers have the developed apps and pedestrians could be detected by the external camera. For both conditions, the warning logics were developed and programmed in the cloud server. Once a potential conflict is identified, the warning message could be received by smartphone apps. Totally 100 experiments were conducted at intersections and road segments to validate the developed system. It suggested that the developed system could identify the potential conflicts between vehicles and pedestrians with high accuracy and send the warning in real time. Through the completed tasks, it could be concluded that the developed smartphone applications and the warning system could successfully emulate the OBU for the P2V applications. The summary of developed applications is presented in Table 37. Totally 9 different applications have been developed for different scenarios.

Table 37 Summary of developed applications

Application type	Application scenario	Application environment	Vehicle movement	Equipment	Warning Receiver
I2V application	Curve warning	<ul style="list-style-type: none"> Vehicles are approaching curves 	-	<ul style="list-style-type: none"> Smartphones of drivers Cloud server 	Drivers
	Queue warning at intersections	<ul style="list-style-type: none"> Vehicles are approaching the ends of queues at intersections 	-	<ul style="list-style-type: none"> Smartphones of drivers Cameras at intersection Cloud server 	Drivers
P2V application	P2V warning at segments	<ul style="list-style-type: none"> Pedestrians are jaywalking or walking along segments There are potential conflicts between pedestrians and vehicles 	-	<ul style="list-style-type: none"> Smartphones of drivers Smartphones of pedestrians Cloud server 	Drivers and pedestrians
	P2V warning at intersections	<ul style="list-style-type: none"> Pedestrians are crossing intersections Pedestrians have the developed app There are potential conflicts between pedestrians and vehicles 	Straight	<ul style="list-style-type: none"> Smartphones of drivers Smartphones of pedestrians Cloud server 	Drivers and pedestrians
			Left-turn		
			Right turn		
P2V warning at intersections	<ul style="list-style-type: none"> Pedestrians are crossing intersections Pedestrians don't have the developed app There are potential conflicts between pedestrians and vehicles 	Straight	<ul style="list-style-type: none"> Smartphones of drivers Cameras at intersection Cloud server 	Drivers	
		Left-turn			
		Right turn			

With all efforts, the products of implementation from this project are summarized as follows:

- Smartphone apps which could collect smartphone sensors data with high sampling frequency.
- A architecture which allows smartphones apps to connect the cloud server with low latency.
- A system including multiple algorithms to compute users' traffic parameters such as turning movements and transportation modes based on smartphone sensors in real time.
- A framework of using cameras to automatically detect pedestrians and queue length.
- A system which enables to use smartphones as OBU emulators for I2V and P2V applications.

PROJECT SCHEDULE

Project Title	Using Smartphone as OBU Emulator Implementation Study
Research Agency	University of Central Florida
Principal Investigator	Mohamed Abdel-Aty

- Task 1: Review of Studies about Smartphone-Based Vehicle Communication and OBU Applications
- Task 2: Development of PWA and Validation of the Feasibility
- Task 3: Validation of the Data Accuracy and Refinement of the Data Stream
- Task 4: Development of Preliminary Smartphone-Based Application
- Task 5: Test of Complementary Sensors for P2V
- Task 6: Development of Proactive Smartphone-Based P2V Application
- Task 7: Phase 1 Work Evaluation, Draft Final Report, and Closeout Teleconference Plan
- Task 8: Final Report

	Anticipated Timeframe (in months)	2018				2019												2020		Estimated Completion (%)
		Sep	Oct	Nov	Dec	Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec	Jan	Feb	
		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	
Task 1	3																			100
Task 2	5																			100
Task 3	5																			100
Task 4	4																			100
Task 5	5																			100
Task 6	2																			100
Task 7	2																			100
Task 8	3																			100

REFERENCES

- Akherfi, K., Gerndt, M., & Harroud, H. (2018). *Mobile cloud computing for computation offloading: Issues and challenges*. Applied Computing and Informatics, 14(1), 1-16.
- Anaya, J. J., Merdrignac, P., Shagdar, O., Nashashibi, F., & Naranjo, J. E. (2014). *Vehicle to pedestrian communications for protection of vulnerable road users*. Proceedings of 2014 IEEE Intelligent Vehicles Symposium Proceedings, 1037-1042.
- Angelova, A., Krizhevsky, A. and Vanhoucke, V. (2015). *Pedestrian detection with a large-field-of-view deep network*. Proceedings of 2015 IEEE international conference on robotics and automation (ICRA), 704-711.
- Arras, K.O., Mozo, O.M. and Burgard, W. (2007). *Using boosted features for the detection of people in 2d range data*. Proceedings of 2007 IEEE International Conference on Robotics and Automation, 3402-3407.
- Artail, H., Khalifeh, K., & Yahfoufi, M. (2017). *Avoiding car-pedestrian collisions using a VANET to cellular communication framework*. Paper presented at the 2017 Wireless Communications and Mobile Computing Conference (IWCMC).
- Arulampalam, M.S., Maskell, S., Gordon, N. and Clapp, T. (2002). *A tutorial on particle filters for online nonlinear/non-Gaussian Bayesian tracking*. IEEE Transactions on signal processing, 50(2), 174-188.
- Baumberg, A. and Hogg, D. (1997). *Learning deformable models for tracking the human body*. In Motion-Based Recognition (39-60). Springer, Dordrecht.

Baumberg, A. (1998). *Hierarchical shape fitting using an iterated linear filter*. Image and vision computing, 16(5), 329-335.

Benenson, R., Mathias, M., Timofte, R. and Van Gool, L. (2012). *Pedestrian detection at 100 frames per second*. In 2012 IEEE Conference on Computer Vision and Pattern Recognition (2903-2910).

Bareth, U., & Kupper, A. (2011). *Energy-Efficient Position Tracking in Proactive Location-Based Services for Smartphone Environments*. Paper presented at the 2011 IEEE Annual Computer Software and Applications Conference.

Bellotti, C., Bellotti, F., De Gloria, A., Andreone, L., & Mariani, M. (2004). *Developing a near infrared based night vision system*. Paper presented at 2014 IEEE Intelligent Vehicles Symposium.

Ben Abdesslem, F., Phillips, A., & Henderson, T. (2009). *Less is more: energy-efficient mobile sensing with senseless*. Paper presented at the Proceedings of the 2009 ACM workshop on Networking, systems, and applications for mobile handhelds.

Ben-Yaacov, A., Maltz, M., & Shinar, D. (2002). *Effects of an in-vehicle collision avoidance warning system on short-and long-term driving performance*. Human Factors, 44(2), 335-342.

Bhoraskar, R., Vankadhara, N., Raman, B., & Kulkarni, P. (2012). *Wolverine: Traffic and road condition estimation using smartphone sensors*. Paper presented at the 2012 International Conference on Communication Systems and Networks.

Biljecki, F., Ledoux, H., & van Oosterom, P. (2013). *Transportation mode-based segmentation and classification of movement trajectories*. *International Journal of Geographical Information Science*, 27, 385-407.

Boy Genius Report (BGR) Media LLC. (2018). *We finally know exactly how big the iPhone XS Max and iPhone XR batteries are*. (<https://bgr.com/2018/09/19/iphone-xs-max-vs-iphone-xr-battery-size-and-battery-life-compared/>, access on 03/24/2019).

Bradski, G. (2019). *Open source computer vision*. (<http://www.drdoobs.com/open-source/the-opencv-library/184404319>, accessed on 06/20/2019).

Burt, M., Zimmer, R. E., Zink, G. J., Valentine, D. A., & Knox Jr, W. J. (2014a). *Transit Safety Retrofit Package Development: TRP Concept of Operations* (Report No. FHWA-JPO-14-117). U.S. Department of Transportation, Intelligent Transportation Systems Joint Program Office, Washinton D.C.

Burt, M., Zimmer, R. E., Zink, G. J., Valentine, D. A., & Knox Jr, W. J. (2014b). *Transit safety retrofit package development: architecture and design specifications*, (Report No. FHWA-JPO-14-119). U.S. Department of Transportation, Intelligent Transportation Systems Joint Program Office, Washinton D.C.

Burt, M., Zimmer, R. E., Zink, G. J., Valentine, D. A., & Knox Jr, W. J. (2014c). *Transit Safety Retrofit Package Development: Applications Requirements Document*, (Report No. FHWA-JPO-14-118). U.S. Department of Transportation, Intelligent Transportation Systems Joint Program Office, Washinton D.C.

Carroll, A., & Heiser, G. (2010). *An Analysis of Power Consumption in a Smartphone*. In USENIX annual technical conference, 14, 21-21.

Chachich, A.C., Pau, A., Barber, A., Kennedy, K., Olejniczak, E., Hackney, J., Sun, Q. and Mireles, E. (1997). *Traffic sensor using a color vision method*. In *Transportation Sensors and Controls: Collision Avoidance, Traffic Management, and ITS*, 2902, 156-165.

Chandra, R., Padhye, J., Ravindranath, L., & Wolman, A. (2007). *Beacon-stuffing: Wi-fi without associations*. Paper presented at the 2007 IEEE Mobile Computing Systems and Applications.

Chen, Z., Zou, H., Jiang, H., Zhu, Q., Soh, Y., & Xie, L. (2015). *Fusion of Wi-Fi, smartphone sensors and landmarks using the Kalman filter for indoor localization*. *Sensors*, 15(1), 715-732.

Chowdhury, A., Chakravarty, T., & Balamuralidhar, P. (2014). *A novel approach to improve vehicle speed estimation using smartphone's INS/GPS sensors*. Paper presented at the 2014 International Conference on Sensing Technology (ICST).

Coifman, B., Beymer, D., McLauchlan, P. and Malik, J. (1998). *A real-time computer vision system for vehicle tracking and traffic surveillance*. *Transportation Research Part C: Emerging Technologies*, 6(4), 271-288.

Cutler, R., & Davis, L. S. (2000). *Robust real-time periodic motion detection, analysis, and applications*. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8), 781-796.

Dalal, N., & Triggs, B. (2005). *Histograms of oriented gradients for human detection*. Paper presented at the IEEE Computer Vision and Pattern Recognition.

Daniel Costea, A. and Nedeveschi, S. (2014). *Word channel based multiscale pedestrian detection without image resizing and using only one classifier*. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (2393-2400).

David, K., & Flach, A. (2010). *Car-2-x and pedestrian safety*. IEEE Vehicular Technology Magazine, 5(1), 70-76.

Deng, Z. A., Hu, Y., Yu, J., & Na, Z. (2015). *Extended Kalman filter for real time indoor localization by fusing Wi-Fi and smartphone inertial sensors*. Micromachines, 6(4), 523-543.

Dey, K. C., Rayamajhi, A., Chowdhury, M., Bhavsar, P., & Martin, J. (2016). *Vehicle-to-vehicle (V2V) and vehicle-to-infrastructure (V2I) communication in a heterogeneous wireless network—Performance evaluation*. Transportation Research Part C: Emerging Technologies, 68, 168-184.

Dhondge, K., Song, S., Choi, B.-Y., & Park, H. (2014). *Wi-FiHonk: smartphone-based beacon stuffed Wi-Fi Car2X-communication system for vulnerable road user safety*. Paper presented at the 2010 IEEE Vehicular Technology Conference.

Eboli, L., G. Mazzulla and G. Pungillo (2016). *Combining speed and acceleration to define car users' safe or unsafe driving behaviour*. Transportation Research Part C: Emerging Technologies 68: 113-125.

Eftekhari, H. R., & Ghatee, M. (2016). *An inference engine for smartphones to preprocess data and detect stationary and transportation modes*. Transportation Research Part C: Emerging Technologies, 69, 313-327.

Eren, H., Makinist, S., Akin, E., & Yilmaz, A. (2012). *Estimating Driving Behavior by a Smartphone*. Paper presented at the 2012 IEEE Intelligent Vehicles Symposium (Iv), 234-239.

- Fang, S. H., Liao, H. H., Fei, Y. X., Chen, K. H., Huang, J. W., Lu, Y. D., & Tsao, Y. (2016). *Transportation modes classification using sensors on smartphones*. *Sensors*, 16(8), 1324-1339.
- Fernando, N., Loke, S. W., & Rahayu, W. (2013). *Mobile cloud computing: A survey*. *Future generation computer systems*, 29(1), 84-106.
- Ferreira, J., Carvalho, E., Ferreira, B. V., de Souza, C., Suhara, Y., Pentland, A., & Pessin, G. (2017). *Driver behavior profiling: An investigation with different smartphone sensors and machine learning*. *PLoS One*, 12(4), 1-15.
- Federal Highway Administration (FHWA). (2009). *Manual of uniform traffic control devices*. U.S. Department of Transportation, Washington D.C.
- Federal Highway Administration (FHWA). (2018). *Safety, Pedestrian & Bicycle Safety*. U.S. Department of Transportation, Washington D.C. (https://safety.fhwa.dot.gov/ped_bike/, accessed on 06/15/2019).
- Fod, A., Howard, A. and Mataric, M.A.J. (2002). *A laser-based people tracker*. In *Proceedings 2002 IEEE International Conference on Robotics and Automation* (Cat. No. 02CH37292), 3, 3024-3029.
- Friedman, N., & Russell, S. (1997). *Image segmentation in video sequences: A probabilistic approach*. Paper presented at the Proceedings of the 1997 Conference on Uncertainty in Artificial Intelligence.
- Fuerstenberg, K., & Willhoeft, V. (2001). *Object tracking and classification using laserscanners-pedestrian recognition in urban environment*. Paper presented at the 2001 IEEE Intelligent Transportation Systems.

Fuerstenberg, K., Dietmayer, K., & Willhoeft, V. (2002). *Pedestrian recognition in urban traffic using a vehicle based multilayer laserscanner*. Paper presented at the 2002 IEEE Intelligent Vehicle Symposium.

Galata, A., Cohn, A., Magee, D. and Hogg, D. (2002). *Modeling interaction using learnt qualitative spatio-temporal relations and variable length markov models*. In ECA, 741-745.

Gavrila, D.M. and Munder, S. (2007). *Multi-cue pedestrian detection and tracking from a moving vehicle*. International journal of computer vision, 73(1), 41-59.

Gavrila, D.M. and Philomin, V. (1999). *Real-time object detection for "smart" vehicles*. In Proceedings of the Seventh IEEE International Conference on Computer Vision, 1, 87-93.

Giusti, A., Cireşan, D.C., Masci, J., Gambardella, L.M. and Schmidhuber, J. (2013). *Fast image scanning with deep max-pooling convolutional neural networks*. In 2013 IEEE International Conference on Image Processing, 4034-4038.

Grimson, W.E.L., Stauffer, C., Romano, R. and Lee, L. (1998). *Using adaptive tracking to classify and monitor activities in a site*. In Proceedings of 1998 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (Category No. 98CB36231), 22-29.

Ghose, A., Chowdhury, A., Chandel, V., Banerjee, T., & Chakravarty, T. (2016). *An enhanced automated system for evaluating harsh driving using smartphone sensors*. Paper presented at the Proceedings of the 2016 International Conference on Distributed Computing and Networking.

Girshick, R. (2015). *Fast r-cnn*. Paper presented at the Proceedings of the 2015 IEEE International Conference on Computer Vision.

Girshick, R., Donahue, J., Darrell, T., & Malik, J. (2014). *Rich feature hierarchies for accurate object detection and semantic segmentation*. Paper presented at the Proceedings of the 2014 IEEE Conference on Computer Vision and Pattern Recognition.

Grimson, W. E. L., Stauffer, C., Romano, R., & Lee, L. (1998). *Using adaptive tracking to classify and monitor activities in a site*. Paper presented at the Proceedings of the 1998 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 22-29.

GitHub (2018). *A Keras implementation of YOLOv3 (Tensorflow backend)*. (<https://github.com/qqwweee/keras-yolo3>, access on 06/15/2019).

Glaser, S., Nouveliere, L., & Lusetti, B. (2007). *Speed limitation based on an advanced curve warning system*. Paper presented at the Proceedings of the 2007 IEEE Intelligent Vehicles Symposium, 686-691.

Hadi, R.A., Sulong, G. and George, L.E. (2014). *Vehicle detection and tracking techniques: a concise review*. arXiv preprint arXiv:1410.5894.

Harding, J., Powell, G., Yoon, R., Fikentscher, J., Doyle, C., Sade, D., Wang, J. (2014). *Vehicle-to-vehicle communications: Readiness of V2V technology for application*. National Highway Traffic Safety Administration, Washington D.C.

Hare, S., Golodetz, S., Saffari, A., Vineet, V., Cheng, M. M., Hicks, S. L., & Torr, P. H. (2015). *Struck: Structured output tracking with kernels*. IEEE Transactions on Pattern Analysis and Machine Intelligence, 38(10), 2096-2109.

Heap, T. and Hogg, D. (1998). *Wormholes in shape space: Tracking through discontinuous changes in shape*. In Sixth International Conference on Computer Vision (IEEE Cat. No. 98CH36271), 344-349.

Howe, G., Xu, G., Hoover, D., Elsasser, D., & Barickman, F. (2016a). *Commercial Connected Vehicle Test Procedure Development and Test Results–Forward Collision Warning (Report No. DOT-HS-812-298)*, U.S. Department of Transportation, Washington D.C.

Howe, G., Xu, G., Hoover, D., Elsasser, D., & Barickman, F. (2016b). *Commercial Connected Vehicle Test Procedure Development and Test Results–Blind Spot Warning/Lane Change Warning (Report No. DOT-HS-812-317)*, U.S. Department of Transportation, Washington D.C.

Howe, G., Xu, G., Hoover, D., Elsasser, D., & Barickman, F. (2016c). *Commercial Connected Vehicle Test Procedure Development and Test Results–Emergency Electronic Brake Light (Report No. DOT-HS-812-327)*, U.S. Department of Transportation, Washington D.C.

Howe, G., Xu, G., Hoover, D., Elsasser, D., & Barickman, F. (2016d). *Commercial Connected Vehicle Test Procedure Development and Test Results–Forward Collision Warning (Report No. DOT-HS-812-298)*, U.S. Department of Transportation, Washington D.C.

Hussein, A., García, F., Armingol, J. M., & Olaverri-Monreal, C. (2016e). *P2V and V2P Communication for Pedestrian Warning on the basis of Autonomous Vehicles*. Paper presented at the 2016 IEEE Intelligent Transportation Systems (ITSC) Conference.

Jing, P., Huang, W., & Chen, L. (2017). *Car-to-pedestrian communication safety system based on the vehicular ad-hoc network environment: A systematic review*. *Information*, 8(4), 127-146.

Johnson, D., & Trivedi, M. (2011). *Driving Style Recognition Using a Smartphone as a Sensor Platform*. Paper presented at the 2011 International IEEE Conference on Intelligent Transportation Systems, 1609-1615.

- Kalal, Z., Mikolajczyk, K., & Matas, J. (2010). *Face-tld: Tracking-learning-detection applied to faces*. Paper presented at the 2010 IEEE Image Processing (ICIP) International Conference.
- Karman, K.P. (1990). *Moving object recognition using an adaptive background memory*. Proceedings of Time Varying Image Processing, 289-296.
- Kang, W., & Han, Y. (2014). *SmartPDR: Smartphone-based pedestrian dead reckoning for indoor localization*. IEEE Sensors journal, 15(5), 2906-2916.
- Keller, J. M., Michael R. Gray, and James A. Givens (1985). *A fuzzy k-nearest neighbor algorithm*. IEEE transactions on systems, man, and cybernetics, 4, 580-585.
- Kilger, M. (1992). *A shadow handler in a video-based real-time traffic monitoring system*. Proceedings IEEE Workshop on Applications of Computer Vision, 11-18.
- Kohle, M., Merkl, D., & Kastner, J. (1997). *Clinical gait analysis by neural networks: issues and experiences*. Paper presented at the 1997 IEEE Computer-Based Medical Systems.
- Koller, D., Daniilidis, K., & Nagel, H. H. (1993). *Model-based object tracking in monocular image sequences of road traffic scenes*. International Journal of Computer Vision, 10(3), 257-281.
- Koller, D., Weber, J., Huang, T., Malik, J., Ogasawara, G., Rao, B., & Russell, S. (1994). *Towards robust automatic traffic scene analysis in real-time*. Paper presented at the 1994 IAPR Pattern Recognition.
- Lane, N., Chon, Y., Zhou, L., Zhang, Y., Li, F., Kim, D., & Cha, H. (2013). *Piggyback CrowdSensing (PCS): energy efficient crowdsourcing of mobile sensor data by exploiting*

smartphone app opportunities. Paper presented at the Proceedings of the 2013 ACM Conference on Embedded Networked Sensor Systems.

LeBlanc, D., Bogard, S., & Gilbert, M. (2014a). *Connected commercial vehicles-integrated truck project: vehicle build and build test plan* (Report No. FHWA-JPO-13-112), U.S. Department of Transportation, Washington D.C.

LeBlanc, D., Bogard, S. E., & Goodsell, R. (2014b). *Connected commercial vehicles—retrofit safety device kit project model deployment operational analysis report* (Report No. FHWA-JPO-14-110), U.S. Department of Transportation, Washington D.C.

Lee, S., & Lim, A. (2013). *An empirical study on ad hoc performance of DSRC and Wi-Fi vehicular communications*. *International Journal of Distributed Sensor Networks*, 9(11), 1-12.

Lendino, J. (2012). “*iOnRoad Augmented Driving (for Android)*”. *PC Mag*.

Liebner, M., Klanner, F., & Stiller, C. (2013). *Active safety for vulnerable road users based on smartphone position data*. Paper presented at the 2013 IEEE Intelligent Vehicles Symposium.

Lin, A., Zhang, J., Lu, K., & Zhang, W. (2014). *An efficient outdoor localization method for smartphones*. Paper presented at the 2014 International Conference on Computer Communication and Networks (ICCCN).

Lipton, A. J., Fujiyoshi, H., & Patil, R. S. (1998). *Moving target classification and tracking from real-time video*. Paper presented at the 1998 IEEE Applications of Computer Vision.

Liu, Y., Shan, S., Zhang, W., Chen, X. and Gao, W. (2009). *Granularity-tunable gradients partition (GGP) descriptors for human detection*. In 2009 IEEE Conference on Computer Vision and Pattern Recognition, 1255-1262.

Liu, J., Priyantha, B., Hart, T., Jin, Y., Lee, W., Raghunathan, V., & Wang, Q. (2016a). *CO-GPS: Energy Efficient GPS Sensing with Cloud Offloading*. IEEE Transactions on Mobile Computing, 15(6), 1348-1361.

Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C. Y., & Berg, A. C. (2016). *SSD: Single shot multibox detector*. In European conference on computer vision, 21-37. Springer, Cham.

Liu, Z., Liu, Z., Meng, Z., Yang, X., Pu, L., & Zhang, L. (2016c). *Implementation and performance measurement of a V2X communication system for vehicle and pedestrian safety*. International Journal of Distributed Sensor Networks, 12(9), 1550147716671267.

Liu, Z., Wu, M., Zhu, K., & Zhang, L. (2016d). *SenSafe: A smartphone-based traffic safety framework by sensing vehicle and pedestrian behaviors*. Mobile Information Systems.

Liu, X., Mei, H., Lu, H., Kuang, H., & Ma, X. (2017). *A vehicle steering recognition system based on low-cost smartphone sensors*. Sensors, 17(3), 633.

Lou, J., Yang, H., Hu, W., & Tan, T. (2002). *Visual vehicle tracking using an improved EKF*. Paper presented at the Proceedings of the Asian Conference of Computer Vision.

Lou, J., Yang, H., Hu, W.M. and Tan, T. (2002). *Visual vehicle tracking using an improved EKF*. Proceedings of Asian Conference Computer Vision, 296-301.

Lowe, D. G. (2004). *Distinctive image features from scale-invariant keypoints*. International Journal of Computer Vision, 60(2), 91-110.

Lu, D., Nguyen, D., Nguyen, T., & Nguyen, H. (2018). *Vehicle Mode and Driving Activity Detection Based on Analyzing Sensor Data of Smartphones*. Sensors (Basel), 18(4), 1036.

Luo, P., Tian, Y., Wang, X. and Tang, X. (2014). *Switchable deep network for pedestrian detection*. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 899-906.

Lusetti, B., Nouveliere, L., Glaser, S., & Mammari, S. (2008). *Experimental strategy for a system based curve warning system for a safe governed speed of a vehicle*. In 2008 IEEE Intelligent Vehicles Symposium, 660-665.

Malik, J., Beymer, D., Coifman, B., Huang, T., Lyddy, D., McLauchlan, P., Russell, S., Weber, J., 1997. *Traffic Surveillance and Detection Technology Development: New Traffic Sensor* (Report No. UCB-ITS-PRR-97-6), University of California, Berkeley, CA.

Martin, R.P., Vahdat, A.M., Culler, D.E. and Anderson, T.E. (1997). *Effects of communication latency, overhead, and bandwidth in a cluster architecture*. In ACM SIGARCH Computer Architecture News, 25(2), 85-97.

Mendes, A., Bento, L.C. and Nunes, U. (2004). *Multi-target detection and tracking with a laser scanner*. In IEEE Intelligent Vehicles Symposium, 796-801.

Meyer, D., Denzler, J., & Niemann, H. (1997). *Model based extraction of articulated objects in image sequences for gait analysis*. Paper presented at the Proceedings of International Conference on Image Processing.

Meyer, D., Pösl, J., & Niemann, H. (1998). *Gait Classification with HMMs for Trajectories of Body Parts Extracted by Mixture Densities*. Paper presented at the British Machine Vision Conference (BMVC).

Mohan, P., Padmanabhan, V., & Ramjee, R. (2008). *Nericell: Rich Monitoring of Road and Traffic Conditions using Mobile Smartphones*. ACM Sensys: Association for Computing Machinery, Inc.

Mohan, S. and Thondiyath, A. (2011). *A non-linear tracking control scheme for an under-actuated autonomous underwater robotic vehicle*. International Journal of Ocean System Engineering, 1(3), 120-135.

Montemerlo, M., Thrun, S. and Whittaker, W. (2002). *Conditional particle filters for simultaneous mobile robot localization and people-tracking*. In Proceedings 2002 IEEE International Conference on Robotics and Automation (Cat. No. 02CH37292), 1, 695-701.

National Highway Traffic Safety Administration (NHTSA), 2016. *Vehicle-to-Vehicle Communication Technology for Light Vehicles*. U.S. Department of Transportation, Washington D.C.

Nguyen, H. T., & Smeulders, A. W. (2006). *Robust tracking using foreground-background texture discrimination*. International Journal of Computer Vision, 69(3), 277-293.

Oshin, T. O., Poslad, S., & Ma, A. (2012). *Improving the Energy-Efficiency of GPS Based Location Sensing Smartphone Applications*. 2012 IEEE 11th International Conference on Trust, Security and Privacy in Computing and Communications, 1698-1705.

Oshin, T. O., Poslad, S., & Ma, A. (2012). *Improving the Energy-Efficiency of GPS Based Location Sensing Smartphone Applications*. Paper presented at the 2012 IEEE International Conference on Trust, Security and Privacy in Computing and Communications.

Ouyang, W. & Wang, X. (2013). *Joint deep learning for pedestrian detection*. In Proceedings of the IEEE International Conference on Computer Vision, 2056-2063.

Paefgen, J., Kehr, F., Zhai, Y., & Michahelles, F. (2012). *Driving behavior analysis with smartphones: insights from a controlled field study*. Paper presented at the Proceedings of the 2012 International Conference on Mobile and Ubiquitous Multimedia.

Papageorgiou, C. & Poggio, T. (2000). *A trainable system for object detection*. International Journal of Computer Vision, 38(1), 15-33.

Pisano, P. A., Goodwin, L. C., & Rossetti, M. A. (2008). *US highway crashes in adverse road weather conditions*. Proceedings of 2008 Conference on International Interactive Information and Processing Systems for Meteorology, Oceanography and Hydrology.

Prelipcean, A. C., Gidófalvi, G., & Susilo, Y. O. (2014). *Mobility collector*. Journal of Location Based Services, 8(4), 229-255.

Radu, V., & Marina, M. (2013). *Himloc: Indoor smartphone localization via activity aware pedestrian dead reckoning with selective crowdsourced Wi-Fi fingerprinting*. Paper presented at the Indoor Positioning and Indoor Navigation (IPIN).

Rappaport, T.S. (1996). *Wireless communications: principles and practice (Vol. 2)*. New Jersey: prentice hall PTR.

Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2016). *You only look once: Unified, real-time object detection*. Paper presented at the Proceedings of the 2016 IEEE conference on computer vision and pattern recognition.

Redmon, J. & Farhadi, A. (2018). *Yolov3: An incremental improvement*. arXiv preprint arXiv:1804.02767.

Ren, S., He, K., Girshick, R., & Sun, J. (2015). *Faster r-cnn: Towards real-time object detection with region proposal networks*. Paper presented at the 2015 Advances in Neural Information Processing Systems.

Renaudin, V., Demeule, V. & Ortiz, M. (2013). *Adaptative pedestrian displacement estimation with a smartphone*. In International conference on indoor positioning and indoor navigation, 1-9.

Renaudin, V., & Combettes, C. (2014). *Magnetic, acceleration fields and gyroscope quaternion (MAGYQ)-based attitude estimation with smartphone sensors for indoor pedestrian navigation*. Sensors, 14(12), 22864-22890.

Renaudin, V., Demeule, V., & Ortiz, M. (2013). *Adaptative pedestrian displacement estimation with a smartphone*. Paper presented at the International Conference on Indoor Positioning and Indoor Navigation (IPIN).

Ridder, C., Munkelt, O., & Kirchner, H. (1995). *Adaptive background estimation and foreground detection using kalman-filtering*. Proceedings of 1995 International Conference on recent Advances in Mechatronics.

Sabzmeydani, P. & Mori, G. (2007). *Detecting pedestrians by learning shapelet features*. In 2007 IEEE Conference on Computer Vision and Pattern Recognition, 1-8.

Saiprasert, C., Pholprasit, T., & Pattara-Atikom, W. (2013). *Detecting driving events using smartphone*. Paper presented at the Proceedings of the 2013 ITS World Congress.

Sanaei, Z., Abolfazli, S., Gani, A., & Buyya, R. (2014). *Heterogeneity in Mobile Cloud Computing: Taxonomy and Open Challenges*. IEEE Communications Surveys & Tutorials, 16(1), 369-392.

Schiele, B. (2006). *Model-free tracking of cars and people based on color regions*. Image and Vision Computing, 24(11), 1172-1178.

Scott, J. J., & Gray, R. (2008). *A comparison of tactile, visual, and auditory warnings for rear-end collision prevention in simulated driving*. Human factors, 50(2), 264-275.

Semanjski, I., & Gautama, S. (2016). *Crowdsourcing mobility insights: reflection of attitude based segments on high resolution mobility behaviour data*. Transportation Research Part C- Emerging Technologies, 71, 434-446.

Seyfarth, A., Iida, F., Tausch, R., Stelzer, M., von Stryk, O., & Karguth, A. (2009). *Towards bipedal jogging as a natural result of optimizing walking speed for passively compliant three-segmented legs*. The International Journal of Robotics Research, 28(2), 257-265.

Shah, S.A.A., Ahmed, E., Imran, M. and Zeadally, S. (2018). *5G for vehicular communications*. IEEE Communications Magazine, 56(1), 111-117.

Singh, G., Bansal, D., & Sofat, S. (2017). *A smartphone based technique to monitor driving behavior using DTW and crowdsensing*. Pervasive and Mobile Computing, 40, 56-70.

Sivaraman, S., and Mohan Manubhai Trivedi. (2013). *Integrated lane and vehicle detection, localization, and tracking: A synergistic approach*. IEEE Transactions on Intelligent Transportation Systems, 906-917.

Stauffer, C., & Grimson, W. (1999). *Adaptive background mixture models for real-time tracking*. Paper presented at the 1999 IEEE Computer Society Conference on Computer Vision and Pattern Recognition.

Steinfeld, A., Duggins, D., Gowdy, J., Kozar, J., MacLachlan, R., Mertz, C., Wang, C. (2004). *Development of the side component of the transit integrated collision warning system*. Paper presented at the 2004 IEEE Intelligent Transportation Systems.

Stephens, D., Pape, D., LeBlanc, D., Bogard, S., Peredo, G., Berg, R., & Wells, B. (2014). *Connected commercial vehicles—Integrated Truck project driver clinics, performance tests, and lessons learned* (Report No. FHWA-JPO-13-112).

Su, X. (2017). *Travel Mode Identification with Smartphone Sensors*. University of New York, Ph.D. Dissertation.

Sun, H., Feng, T., & Tan, T. (2000). *Robust extraction of moving objects from video sequences*, Proceedings of the Fourth Asian Conference on Computer Vision, 2, 961-963.

Szarvas, M., Sakai, U., & Ogata, J. (2006). *Real-time pedestrian detection using LIDAR and convolutional neural networks*. Paper presented at the 2006 IEEE Intelligent Vehicles Symposium.

Tahmasbi-Sarvestani, A., Mahjoub, H. N., Fallah, Y. P., Moradi-Pari, E., & Abuchaar, O. (2017). *Implementation and evaluation of a cooperative vehicle-to-pedestrian safety application*. IEEE Intelligent Transportation Systems Magazine, 9(4), 62-75.

Tak, S., Kim, S., & Yeo, H. (2015). *Development of a deceleration-based surrogate safety measure for rear-end collision risk*. IEEE transactions on intelligent transportation systems, 16(5), 2435-2445.

Tan, T., Sullivan, G. D., & Baker, K. D. (1994). *Pose determination and recognition of vehicles in traffic scenes*. Paper presented at the 1994 European Conference on Computer Vision.

Techradar. (2018). *iPhone XS Review*. (<https://www.techradar.com/reviews/iphone-xs-review/4>, access in 03/24/2019).

Tian, Z., Zhang, Y., Zhou, M., & Liu, Y. (2014). *Pedestrian dead reckoning for MARG navigation using a smartphone*. EURASIP Journal on Advances in Signal Processing, 2014(1), 65.

Tyagi, V., Kalyanaraman, S., & Krishnapuram, R. (2012). *Vehicular traffic density state estimation based on cumulative road acoustics*. IEEE Transactions on Intelligent Transportation Systems, 13(3), 1156-1166.

Viola, P. & Jones, M.J. (2004). *Robust real-time face detection*. International journal of computer vision, 57(2), 137-154.

Viola, P., Jones, M. J., & Snow, D. (2005). *Detecting pedestrians using patterns of motion and appearance*. International Journal of Computer Vision, 63(2), 153-161.

Vlahogianni, E., & Barmponakis, E. (2017). *Driving analytics using smartphones: Algorithms, comparisons and challenges*. Transportation Research Part C: Emerging Technologies, 79, 196-206.

Wahlström, J., Skog, I., & Händel, P. (2017). *Smartphone-based vehicle telematics: A ten-year anniversary*. IEEE Transactions on Intelligent Transportation Systems, 18(10), 2802-2825.

Wang, T., Cardone, G., Corradi, A., Torresani, L., & Campbell, A. (2012). *WalkSafe: a pedestrian safety app for mobile phone users who walk and talk while crossing roads*. Paper presented at the Proceedings of the 2012 Workshop on Mobile Computing Systems, San Diego, California.

Webster, F.V. (1966). *Traffic signals*. Road research technical paper, 56.

Wells, B., & Berg, R. (2014). *Connected commercial vehicles—retrofit safety device kit project: safety applications and development plan* (Report No. FHWA-JPO-14-106). U.S. Department of Transportation. Intelligent Transportation Systems Joint Program Office. Washington D.C.

Whaiduzzaman, M., Sookhak, M., Gani, A., & Buyya, R. (2014). *A survey on vehicular cloud computing*. Journal of Network and Computer applications, 40, 325-344.

White, J., Thompson, C., Turner, H., Dougherty, B., & Schmidt, D. C. (2011). *WreckWatch: Automatic Traffic Accident Detection and Notification with Smartphones*. Mobile Networks and Applications, 16(3), 285.

Wu, X., Miucic, R., Yang, S., Al-Stouhi, S., Misener, J., Bai, S., & Chan, W. (2014). *Cars talk to phones: A DSRC based vehicle-pedestrian safety system*. Paper presented at the IEEE 80th Vehicular Technology Conference.

Wu, B., & Nevatia, R. (2005). *Detection of multiple, partially occluded humans in a single image by bayesian combination of edgelet part detectors*. In Tenth IEEE International Conference on Computer Vision (ICCV'05), 1, 90-97.

Wu, B., & Nevatia, R. (2007). *Cluster boosted tree classifier for multi-view, multi-pose object detection*. In 2007 IEEE 11th International Conference on Computer Vision, 1-8.

Wu, Y., & Huang, T. S. (2001). *A co-inference approach to robust visual tracking*. In Proceedings Eighth IEEE International Conference on Computer Vision, 2, 26-33.

Xavier, J., Pacheco, M., Castro, D., Ruano, A., & Nunes, U. (2005). *Fast line, arc/circle and leg detection from laser scan data in a player driver*. In Proceedings of the 2005 IEEE International Conference on Robotics and Automation, 3930-3935.

Xiao, Y., Low, D., Bandara, T., Pathak, P., Lim, H. B., Goyal, D., Ben-Akiva, M. (2012). *Transportation activity analysis using smartphones*. Paper presented at the 2012 IEEE Consumer Communications and Networking Conference (CCNC).

Xu, Z., Li, X., Zhao, X., Zhang, M. H., & Wang, Z. (2017). *DSRC versus 4G-LTE for connected vehicle applications: A study on field experiments of vehicular communication performance*. Journal of Advanced Transportation, 1-10.

Yamabe, T., & Kiyohara, R. (2014). *A study of on-vehicle information devices using a smartphone*. Paper presented at the 2014 IEEE Computer Software and Applications Conference Workshops (COMPSACW).

Yoon, J., Noble, B., & Liu, M. (2007). *Surface Street Traffic Estimation*. Proceedings of the 5th International Conference on Mobile Systems, Applications and Services. New York, NY, USA: ACM.

Yu, J., Chen, Z., Zhu, Y., Chen, Y. J., Kong, L., & Li, M. (2016). *Fine-grained abnormal driving behaviors detection and identification with smartphones*. *IEEE transactions on mobile computing*, 16(8), 2198-2212.

Yu, R., Zhang, Y., Gjessing, S., Xia, W., & Yang, K. (2013). *Toward cloud-based vehicular networks with efficient resource management*. arXiv preprint arXiv:1308.6208.

Zhao, H., Shao, X. W., Katabira, K., & Shibasaki, R. (2006). *Joint tracking and classification of moving objects at intersection using a single-row laser range scanner*. In 2006 IEEE Intelligent Transportation Systems Conference, 287-294.

Zhao, H., & Shibasaki, R. (2005). *A novel system for tracking pedestrians using multiple single-row laser-range scanners*. *IEEE Transactions on systems, man, and cybernetics-Part A: systems and humans*, 35(2), 283-291.

Zhu, Q., Yeh, M. C., Cheng, K. T., & Avidan, S. (2006). *Fast human detection using a cascade of histograms of oriented gradients*. In 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06), 2, 1491-1498.

Zhu, X., Li, Q., & Chen, G. (2013). *APT: Accurate outdoor pedestrian tracking with smartphones*. Paper presented at the 2013 IEEE INFOCOM Conference.

Zhuang, Z., Kim, K., & Singh, J.(2010). *Improving energy efficiency of location sensing on smartphones*. Paper presented at the Proceedings of the 8th International Conference on Mobile systems, applications, and services, San Francisco, California, USA.

Zimmer, R. E., Burt, M., Zink, G. J., Valentine, D. A., & Knox Jr, W. J. (2014). *Transit safety retrofit package development* (Report No. FHWA-JPO-14-142). U.S. Department of Transportation, Intelligent Transportation Systems Joint Program Office, Washinton D.C.