RESEARCH

WIRELESS DATA COLLECTION SYSTEM FOR TRAVEL TIME ESTIMATION AND TRAFFIC PERFORMANCE EVALUATION

Final Report

SPR 737
OTREC-RR012-06

ODOT/OTREC

This publication is a result of joint funding by Oregon Department of Transportation (ODOT) and Oregon Transportation Research and Education Consortium (OTREC)

WIRELESS DATA COLLECTION SYSTEM FOR TRAVEL TIME ESTIMATION AND TRAFFIC PERFORMANCE EVALUATION

Final Report

SPR 737
OTREC-RR012-06

by

David S. Kim[1], J. David Porter[1], & Mario E. Magaña[2]
SeJoon Park[1], Amirali Saeedi[1]

[1]School of Mechanical, Industrial and Manufacturing Engineering
[2]School of Electrical Engineering and Computer Science
Oregon State University

for

Oregon Department of Transportation
Research Section
200 Hawthorne Ave. SE, Suite B-240
Salem OR 97301-5192

and

Federal Highway Administration
400 Seventh Street, SW
Washington, DC 20590-0003

May 2012

Technical Report Documentation Page

1. Report No.	2. Government Accession No.	3. Recipient's Catalog No.
FHWA-OR-RD-12-13		

4. Title and Subtitle	5. Report Date / 6. Performing Organization Code
Wireless Data Collection System for Travel Time Estimation and Traffic Performance Evaluation	5. Report Date: May 2012 6. Performing Organization Code
7. Author(s) J. David Porter, David S. Kim, SeJoon Park, and Amirali Saeedi School of Mechanical and Industrial Engineering 204 Rogers Hall Oregon State University, Corvallis, OR 97331 Mario E. Magaña School of Electrical Engineering and Computer Science 1148 Kelley Engineering Center Oregon State University, Corvallis, OR 97331	8. Performing Organization Report No.
9. Performing Organization Name and Address Oregon State University Corvallis, OR 97331	10. Work Unit No. (TRAIS) 11. Contract or Grant No. SPR 737
12. Sponsoring Agency Name and Address Oregon Department of Transportation Research Section and Federal Highway Administration 200 Hawthorne Ave. SE, Suite B-240 / 400 Seventh Street, SW Salem, OR 97301-5192 / Washington, DC 20590-0003	13. Type of Report and Period Covered Final Report 14. Sponsoring Agency Code

15. Supplementary Notes

16. Abstract

This report presents the results of the third and final research and development project of an implementable wireless travel time data collection system. Utilizing Bluetooth wireless technology as a platform, the prior projects focused on data collection hardware and network communication implementation. In this project the processing and synthesis of collected data was addressed. Results and procedures that address data filtering, travel time sample calculation, calculation of travel time statistics, travel time forecasting are described. Results for the use of collected data to estimate intersection performance are presented, as are the design and requirements for an automated travel time data collection system. The report includes a users' manual.

17. Key Words	18. Distribution Statement
Bluetooth; travel time estimation; traffic performance	Copies available from NTIS, and online at http://www.oregon.gov/ODOT/TD/TP_RES/

19. Security Classification (of this report)	20. Security Classification (of this page)	21. No. of Pages	22. Price
Unclassified	Unclassified	127	

Technical Report Form DOT F 1700.7 (8-72) Reproduction of completed page authorized Printed on recycled paper

SI* (MODERN METRIC) CONVERSION FACTORS

APPROXIMATE CONVERSIONS TO SI UNITS					APPROXIMATE CONVERSIONS FROM SI UNITS				
Symbol	When You Know	Multiply By	To Find	Symbol	Symbol	When You Know	Multiply By	To Find	Symbol
		LENGTH					**LENGTH**		
in	inches	25.4	millimeters	mm	mm	millimeters	0.039	inches	in
ft	feet	0.305	meters	m	m	meters	3.28	feet	ft
yd	yards	0.914	meters	m	m	meters	1.09	yards	yd
mi	miles	1.61	kilometers	km	km	kilometers	0.621	miles	mi
		AREA					**AREA**		
in^2	square inches	645.2	millimeters squared	mm^2	mm^2	millimeters squared	0.0016	square inches	in^2
ft^2	square feet	0.093	meters squared	m^2	m^2	meters squared	10.764	square feet	ft^2
yd^2	square yards	0.836	meters squared	m^2	m^2	meters squared	1.196	square yards	yd^2
ac	acres	0.405	hectares	ha	ha	hectares	2.47	acres	ac
mi^2	square miles	2.59	kilometers squared	km^2	km^2	kilometers squared	0.386	square miles	mi^2
		VOLUME					**VOLUME**		
fl oz	fluid ounces	29.57	milliliters	ml	ml	milliliters	0.034	fluid ounces	fl oz
gal	gallons	3.785	liters	L	L	liters	0.264	gallons	gal
ft^3	cubic feet	0.028	meters cubed	m^3	m^3	meters cubed	35.315	cubic feet	ft^3
yd^3	cubic yards	0.765	meters cubed	m^3	m^3	meters cubed	1.308	cubic yards	yd^3
NOTE: Volumes greater than 1000 L shall be shown in m^3.									
		MASS					**MASS**		
oz	ounces	28.35	grams	g	g	grams	0.035	ounces	oz
lb	pounds	0.454	kilograms	kg	kg	kilograms	2.205	pounds	lb
T	short tons (2000 lb)	0.907	megagrams	Mg	Mg	megagrams	1.102	short tons (2000 lb)	T
		TEMPERATURE (exact)					**TEMPERATURE (exact)**		
°F	Fahrenheit	(F-32)/1.8	Celsius	°C	°C	Celsius	1.8C+32	Fahrenheit	°F

*SI is the symbol for the International System of Measurement

ACKNOWLEDGEMENTS

The authors would like to thank the members of the Oregon Department of Transportation Intelligent Transportation Systems Unit for their assistance in helping us with equipment installation, and providing us with the test location. We would like to thank OTREC and ODOT for their support of this research as well as the members of the project Technical Advisory Committee: Don Crownover, Galen McGill, Dennis Mitchell, Nathaniel Price, Willie Rotich, Jason Shaddix, ad Doug Spencer. Finally we would like to thank June Ross of ODOT Research for managing the project.

DISCLAIMER

TABLE OF CONTENTS

LIST OF TABLES

LIST OF PHOTOS/FIGURES

1.0 INTRODUCTION

This report presents the results of the third research project in a series of projects focusing on an implementable wireless travel time data collection system. The first project was an Oregon Department of Transportation (ODOT) Intelligent Transportation Systems Unit funded project that focused on the initial development of inexpensive data collection units (DCUs) (*Porter et al. 2009*). The second project was an ODOT/Oregon Transportation Research and Education Consortium (OTREC) funded project that focused on the hardware and communications infrastructure for the wireless travel time data collection system (*Porter et al. 2011*). As part of the second project five DCUs and the supporting communications infrastructure were setup on a five mile section of Highway 99W in Tigard, Oregon. The project described in this report addressed the processing and synthesis of data collected by the installed DCUs for the purposes of generating travel time samples, estimating historical travel time performance, short-term travel time forecasting, and using data collected from a single DCU at an intersection to estimate intersection performance.

The method of travel time data collection made possible by wireless technology is vehicle re-identification. This requires identifiable wireless devices associated with vehicles, and roadside DCUs to detect these devices. When a vehicle is identified at two separate fixed points along a road (DCU locations) the time difference between the identifications constitutes a travel time sample. Bluetooth was the specific wireless technology utilized in this project due to the common use of Bluetooth devices in vehicles, and prior research utilizing this wireless technology for travel time data collection purposes. The media access control (MAC) addresses read from Bluetooth-enabled devices present in vehicles serves as the device identifier. Travel time data collection based on the collection of MAC addresses from Bluetooth-enabled devices has been examined by multiple states (*Wasson et al. 2008; Barcelo et al. 2010; Haghani et al. 2010; Quayle et al. 2010*). Past and ongoing research has demonstrated the feasibility of reading MAC addresses from Bluetooth-enabled devices present in vehicles moving past a fixed location and recording the time of this reading.

Since Bluetooth wireless technology was developed for short-range two way communication between devices (e.g., a cell phone and a hands-free cell phone device), the use of Bluetooth for the purposes of vehicle identification generates a number of issues related to the processing of collected data. To better understand specific research issues, background information on MAC address data collection is presented below

1.1 BACKGROUND INFORMATION

In this section, background information is presented to clarify some of the issues involved in utilizing collected MAC address data to generate travel time samples. Actual data from Bluetooth-based data collection units (DCUs) installed at two locations on Wallace Road in Salem, Oregon, will be utilized in examples of specific issues.

1.1.1 Description and Explanation of the Sample Data

A sample of data collected from one DCU is shown in Figure 1.1. This data is a sample of MAC addresses recorded over a seven minute period starting at approximately 5:00PM on a Tuesday afternoon. Truncated MAC addresses are shown in the first column and the date and time when the read was recorded are shown in the second and third columns, respectively. The table on the left in Figure 1.1 shows the MAC addresses in the sequence that they were read by the DCU. The table on the right in Figure 1.1 shows these same records sorted by MAC address. Nine different MAC addresses were read over the seven minute period.

MAC Addresses in Sequence		
MAC Add.	**Day**	**Time**
2:83:AC:3	8/17/2010	17:00:09
0:D1:BA:6	8/17/2010	17:00:13
0:D1:BA:6	8/17/2010	17:00:21
5:F9:FB:8	8/17/2010	17:00:53
5:F9:FB:8	8/17/2010	17:00:57
A:5E:22:8	8/17/2010	17:00:57
5:F9:FB:8	8/17/2010	17:01:02
5:F9:FB:8	8/17/2010	17:01:06
5:F9:FB:8	8/17/2010	17:01:10
5:F9:FB:8	8/17/2010	17:01:14
5:F9:FB:8	8/17/2010	17:01:19
A:CC:9B:5	8/17/2010	17:01:27
A:CC:9B:5	8/17/2010	17:01:31
A:CC:9B:5	8/17/2010	17:01:47
A:CC:9B:5	8/17/2010	17:01:51
A:5E:22:8	8/17/2010	17:01:59
A:5E:22:8	8/17/2010	17:02:15
C:2B:BD:0	8/17/2010	17:03:06
9:C0:9B:2	8/17/2010	17:03:10
C:2B:BD:0	8/17/2010	17:03:10
9:C0:9B:2	8/17/2010	17:03:15
C:2B:BD:0	8/17/2010	17:03:15
C:21:14:6	8/17/2010	17:04:33
C:21:14:6	8/17/2010	17:04:37
C:21:14:6	8/17/2010	17:04:41
8:6F:2D:F	8/17/2010	17:05:32
A:5E:22:8	8/17/2010	17:06:08
A:5E:22:8	8/17/2010	17:07:02

MAC Addresses Sorted		
MAC Add.	**Day**	**Time**
0:D1:BA:6	8/17/2010	17:00:13
0:D1:BA:6	8/17/2010	17:00:21
2:83:AC:3	8/17/2010	17:00:09
5:F9:FB:8	8/17/2010	17:00:53
5:F9:FB:8	8/17/2010	17:00:57
5:F9:FB:8	8/17/2010	17:01:02
5:F9:FB:8	8/17/2010	17:01:06
5:F9:FB:8	8/17/2010	17:01:10
5:F9:FB:8	8/17/2010	17:01:14
5:F9:FB:8	8/17/2010	17:01:19
8:6F:2D:F	8/17/2010	17:05:32
9:C0:9B:2	8/17/2010	17:03:10
9:C0:9B:2	8/17/2010	17:03:15
A:5E:22:8	8/17/2010	17:00:57
A:5E:22:8	8/17/2010	17:01:59
A:5E:22:8	8/17/2010	17:02:15
A:5E:22:8	8/17/2010	17:06:08
A:5E:22:8	8/17/2010	17:07:02
A:CC:9B:5	8/17/2010	17:01:27
A:CC:9B:5	8/17/2010	17:01:31
A:CC:9B:5	8/17/2010	17:01:47
A:CC:9B:5	8/17/2010	17:01:51
C:21:14:6	8/17/2010	17:04:33
C:21:14:6	8/17/2010	17:04:37
C:21:14:6	8/17/2010	17:04:41
C:2B:BD:0	8/17/2010	17:03:06
C:2B:BD:0	8/17/2010	17:03:10
C:2B:BD:0	8/17/2010	17:03:15

Figure 1.1: Sample of MAC address data from an installed Bluetooth DCU

The data shown in Figure 1.1 highlights several features that are common when reading MAC addresses from Bluetooth-enabled devices:

- There are multiple MAC addresses read over a fixed time period.
- A single MAC address may be read multiple times.
- The number of times a single MAC address is read differs for the different addresses.
- Different MAC addresses may have the same time stamp (i.e., date and time).

If the features of this data are considered along with the characteristics of the DCU, the characteristics of Bluetooth devices present in vehicles, and the antenna type utilized, the reasons for these features can be explained. The DCU utilized conducted an inquiry procedure (identifying the presence of enabled Bluetooth devices) every 3.84 seconds. The antenna attached to the DCU utilized to collect the sample data in Figure 1.1 has a road coverage area length of approximately 1,200 feet. Since a vehicle traveling on Wallace Road at 45 miles per hour (i.e., 66 feet per second) will be in the antenna coverage area for approximately 18 seconds, multiple reads (identifications) of MAC addresses should occur. The fact that the Bluetooth inquiry procedure is probabilistic in nature and that Bluetooth devices vary with respect to radio signal strength, combined with the unknown impacts of uncontrollable factors on radio frequency communications (e.g., interference, multipath) explains why active Bluetooth devices in different passing vehicles moving at the same speed may be read a different number of times.

In the data shown in Figure 1.1, the smallest time interval observed between records with the same MAC address is four seconds. The DCUs installed on Wallace Road were programmed to repeat the initiation of an *inquiry mode* that is 3.84 seconds in length. The duration of the inquiry mode determines the smallest time interval between reads of the same MAC address. Figure 1.1 also shows different MAC addresses that have the same time record (e.g., 9:C0:9B:2 and C:2B:BD:0). These MAC addresses were read (discovered) in the same inquiry mode period, and represent either multiple devices in the same vehicle or multiple vehicles with active devices passing the reader at about the same time.

1.1.2 Specific Issues Related to Computing Travel Time Samples

Although the features of the data obtained from the DCUs are explainable, specific questions arise about what data to save and utilize, and how data from one DCU is paired with data from another DCU for the purpose of generating accurate travel time samples. These questions relate to data filtering, the generation of travel time samples, and the accuracy and precision of travel time samples and travel time measures.

1.1.2.1 Data Filtering

In the context of this research, *data filtering* will refer to the removal of data. There are two different levels at which data filtering was examined. The first level was data filtering at the individual DCU. The second level is the removal of outlier travel time samples after these have been computed from MAC address data collected at two adjacent DCUs.

The objective of data filtering at the DCU level is to filter MAC addresses that are not representative of Bluetooth devices in vehicles traveling past the DCU. As an example of data filtering issues at the DCU level, consider MAC address 5:F9:FB:8 shown in the table of sorted addresses on the right of Figure 1.1. This particular MAC address was read seven times over a 26 second period (elapsed time between the first and last reads). The time differences between successive reads is either four or five seconds, which corresponds to the length of the inquiry mode programmed into the DCU. Should this MAC address data be filtered? If it represents a traveling vehicle, it is traveling at a slow speed through the antenna coverage area of the DCU. In this example, other MAC addresses are read just before and after address 5:F9:FB:8 is read (i.e., 0:D1:BA:6 and A:5E:22:8), and these addresses may generate data in a pattern that is more typical for a Bluetooth-enabled device in a vehicle moving at a speed close to the speed limit. Therefore, address 5:F9:FB:8 may not be in a moving vehicle and can be removed. In this particular seven minute period, address 5:F9:FB:8 makes up 25% of the data. The fact that address 5:F9:FB:8 was read seven times in 26 seconds is not enough information to indicate that the data should be filtered since congestion causing slower vehicle speeds could also generate such data. In this example, address 5:F9:FB:8 would not be filtered. However there are instances where a MAC address is detected many times over a long period of time indicating that the address was not from a Bluetooth device in a passing vehicle. Data of this type should be filtered at the DCU.

Next consider MAC address A:5E:22:8. This particular address was read five times in the seven minute period considered. Unlike 5:F9:FB:8, the longer time interval between reads indicates that it may not represent a Bluetooth-enabled device in a passing vehicle. The minimum time between reads of A:5E:22:8 is 16 seconds, but four of the five time intervals are close to one minute. It would appear that this MAC address should be filtered at the DCU level.

After MAC address data is filtered at DCUs and travel time samples are computed from this data (see the next section), some of these travel time samples will likely represent non-vehicle travel times. The next level of filtering is performed at the central server with the objective of filtering non-vehicle travel times. At this filtering level the goal is to filter non-vehicle travel time samples while at the same time keeping those travel time samples that reflect congestion and random events that may disrupt travel (e.g., accidents).

1.1.2.2 Generation of Travel Time Samples

Assuming that data from adjacent DCUs is filtered, the next data processing step is to "match" data records with the same MAC address collected at adjacent DCUs to generate travel time samples. To facilitate the description of issues related to computing such travel time samples, the concept of a "**group**" of MAC addresses will be defined and illustrated.

A **group** is defined as a collection of MAC address reads for the same MAC address that represent one trip (in a single direction) of the corresponding Bluetooth-enabled device through the coverage area of the DCU along the road that it is monitoring.

For example, the two reads of MAC address 0:D1:BA:6 shown in Figure 1.1 would constitute a group size of two. The seven reads of MAC address 5:F9:FB:8 would constitute a group size of seven. Groups can generally be identified as a collection of reads of the same MAC address with a time difference between successive reads that is no greater than some reasonable fixed value. With this fixed value set at 30 seconds or 60 seconds, the two groups just mentioned remain unchanged. However, MAC address A:5E:22:8 generates four groups with a 30 second value and three groups with a 60 second value.

Given groups of MAC addresses, the basic issue is to determine how MAC address reads within a group should be utilized to compute travel time samples. If address 0:D1:BA:6 also has a group size of two at the adjacent reader, then there are four possible travel time samples that may be computed. Travel time differences in these travel time samples may be relatively small; however, over shorter road segments found in typical arterial road applications, the travel time differences may be large relative to the actual travel time.

1.1.2.3 Accuracy and Precision of Travel Time Samples and Travel Time Measures

As was explained in the prior section, the existence of groups in the collected MAC address data leads to multiple ways that travel time samples can be computed. The central issue here is to identify the best method for computing travel time samples with respect to accuracy and determine if any bias exists in the method.

Once travel time samples are computed, generating long-term and short-term estimates of travel times and the variability in the estimates will need to be addressed. The main issue is the presence of correlation of the travel time samples that will change over time (i.e., high correlation with increased congestion and low or no correlation with low congestion).

1.2 RESEARCH OBJECTIVES

The objective of this project was to complete the research and development so that wide-scale implementation of a Bluetooth-based travel time data collection system may be started. The results of this research will permit ODOT to focus on the implementation of known hardware, software, data processing and analysis procedures, and database structure and reports.

To achieve this objective, this research included the following:

1. The development of data processing and travel time performance estimation procedures (implemented in prototype software) that utilize MAC address data collected by Bluetooth-based DCUs. In combination with the hardware and data collection system design completed in an earlier project, ODOT will have an implementable travel time data collection system utilizing the wireless reading of MAC addresses.
2. The use of MAC address data from single Bluetooth-based DCUs (installed at intersections) to compute intersection performance measures.
3. The development of a framework for travel time and intersection performance data reports that can be utilized to support planning, operations and traveler information functions.

Through this research, a more complete understanding was developed of the capabilities and limitations of travel time and intersection performance measure estimation based on the collection of MAC addresses from Bluetooth-enabled devices.

1.3 REPORT ORGANIZATION

The remainder of this report begins with a literature review. In this literature review, research addressing the use of Bluetooth technology for travel time data collection, data filtering, the analysis of non-independent travel time data, travel time forecasting, road system performance measures, and information systems design and implementation is reviewed. The literature review is followed by separate sections presenting the analysis and results for: data filtering and travel time sample generation; travel time statistics; intersection performance analysis; implementation procedures for the results of earlier sections; system design and functional requirements. A user manual on how to assemble and configure a Bluetooth DCU is included as Appendix A.

2.0 LITERATURE REVIEW

The literature review focused on a number of technical issues relevant to generating travel time samples from Bluetooth MAC address data. These topics are:

1. Existing methods of using Bluetooth MAC address data for generating travel time samples.
2. The accuracy of travel time samples generated from collected MAC addresses.
3. Data filtering techniques.
4. Statistical analysis of collected non-independent travel time samples.
5. Travel time prediction and forecasting methods.
6. Current and future ODOT travel time performance measures.
7. Information system design for applications with high volumes of data networked into central locations to examine network operational protocols, data volume issues, and data storage issues.

No published work was found that explored using collected MAC address data to estimate intersection performance. Intersection performance measures will be covered briefly in Section 2.5.

2.1 EXISTING BLUETOOTH TRAVEL TIME SAMPLE GENERATION METHODS

A number of researchers have published articles about reading Bluetooth MAC addresses for the purposes of estimating travel times. Only a subset of these articles discuss multiple detection of the same MAC address as a vehicle housing a Bluetooth-enabled device passes a DCU.

Malinovskiy et al. (*2010; 2011*) and Puckett et al. (*2010*) have addressed the issue of MAC address groups that are produced by the DCUs. Their simple solution is to utilize the time stamp for the first MAC address in a group (i.e., the time of the first discovery of the Bluetooth device) for all travel time calculations regardless of the number of records in a group. No justification or mention of the accuracy of this method with respect to other methods is presented. Quayle et al. (*2010*) also acknowledged that a Bluetooth equipped device may be detected multiple times as it passes by a DCU. They note that MAC address group sizes depend on the proximity of the DCU to the road and the amount of time the device is within the range of the DCU. The type of antenna used with the DCU is not mentioned. Quayle and Koonce (*2010*) mention that first-to-first and last-to-last detection times in MAC address groups were used to compute the travel time samples, but do not report differences in the accuracy of the methods. Tarnoff and Young (*2010*) mention the issue of multiple detections in a DCU's detection area and present a more detailed discussion on the effects of detection zone geometry and different traveling speeds. They suggest using appropriate DCU spacing in order to minimize redundant detections. For travel time calculations, when multiple detections exist, they suggest using an average of the detection times as the best estimate of the time when the vehicle has just passed the DCU.

2.2 TRAVEL TIME ACCURACY

A number of researchers have examined or considered the accuracy of travel time samples generated from collected MAC address data. Some of these researchers have examined travel time sample accuracy from the perspective of bounds and the relative magnitude of errors, while others have conducted accuracy experiments.

Sadabadi et al. (*2010*) used the relationship between the segment length, average speed, and travel time to come up with an upper bound for the error in speed estimates generated from MAC address data. In this analysis, a maximum "distance error" of 600 feet was assumed. The distance error is defined to be the actual distance that a vehicle is from the DCU when a MAC address is read. The 600 feet distance is equal to two times the estimated coverage distance of the DCU. Wasson et al. (*2008*) have acknowledged the existence of travel time sample inaccuracy. However, they claim that for the distances between DCUs that they examined (2-3 miles), the errors in travel time samples are negligible. There was no data presented to justify this claim.

Research that examined the accuracy of travel time samples generated with MAC address data utilized a variety of methods to establish "ground truth" travel times. GPS tracking, automatic license plate recognition (ALPR) technology, and automated vehicle identification (AVI) technology are examples of such methods. Travel time sample errors were reported as travel speed errors or as travel time errors.

Haghani et al. (*2010*) collected travel speeds of GPS equipped vehicles in Maryland and Northern Virginia for nine days during the morning and afternoon rush hours. These same vehicles contained Bluetooth-enabled devices so that their travel speeds could also be estimated using collected Bluetooth MAC address data. The average absolute error between speeds obtained from GPS tracking and speeds obtained from MAC address data was computed for different distances between the Bluetooth DCUs and for different speed ranges. The average absolute error was 4.66 miles/hour for all speed samples, which contain 78, 94, 241, and 349 speed samples for speeds ranges of 0~30, 30~45, 45~60, and more than 60 miles/hour, respectively. The average speed difference for all samples was only 0.4 miles/hour showing that there is little or no bias in the speed samples. Travel speed accuracy results were also presented for different distances between Bluetooth DCUs and different speed ranges. If the distance was more than one mile, the average absolute errors were 2.3, 4.0, 4.9, and 3.9 miles/hour for speed ranges of 0~30, 30~45, 45~60, and more than 60 miles/hour, respectively. If the distance was less than one mile, the average absolute errors were 3.1, 6.9, 5.5, and 6.4 miles/hour for speed ranges of 0~30, 30~45, 45~60, and more than 60 miles/hour, respectively. This result shows that as the distance between Bluetooth detectors increases and vehicle speed decreases, the average travel speeds generated from collected MAC addresses are more accurate. Results for individual vehicle speeds were not presented.

Quayle et al. (*2010*) collected 46 travel time samples using GPS tracking and 109 travel time samples from collected MAC addresses from 7-9 AM and 4-6 PM on February 11, 2009, between two points along Tualatin-Sherwood Road, Tualatin, Oregon. Average travel times generated from both methods were computed for each traffic direction and time period. The average travel times from GPS tracking were 403, 387, 428, and 762 seconds for 7-9 AM westbound, 7-9 AM eastbound, 4-6 PM westbound, and 4-6 PM eastbound, respectively. The difference between the average travel times from MAC address data and the GPS based travel times is shown in Table 2.1. Results for individual travel time samples were not presented, and statistical tests of travel time differences were not conducted.

Table 2.1: Travel time differences between MAC address and GPS generated travel times

Time	Direction	Average GPS Avg Travel Times	Difference Between Avg Travel Times	Percent Difference
7-9 AM	West	403 seconds	10 seconds	2.5%
7-9 AM	East	387 seconds	10 seconds	2.6%
4-6 PM	West	428 seconds	31 seconds	7.2%
4-6 PM	East	762 seconds	10 seconds	1.3%

Malinovskiy et al. (*2010*) compared travel times generated from MAC address data to travel times obtained using automatic license plate recognition (ALPR). Data was collected on a 0.98 mile six-lane portion of a state highway during a nine hour period. The results indicate that average travel times generated from collected MAC addresses usually overestimated the travel time for the road segment (the average ALPR based travel time was five minutes). The reasoning given for the overestimated travel times is that a faster vehicle has a better chance of moving through the Bluetooth DCU coverage area undetected. The conclusion is that the detection area on the road should be large enough to detect nearly all vehicles with Bluetooth-enabled devices traveling at different speeds. Malinovskiy et al. (*2011*) filtered travel times generated from MAC addresses by utilizing a moving window of travel times and an estimated standard deviation of travel times within the window (filtering mechanism details such as window size and confidence level were not specified). After applying this filter and increasing the DCU road coverage area, the average of the remaining travel times generated from MAC addresses still overestimated travel times.

In a study conducted by KMJ Consulting, Inc. (*2010*), Bluetooth travel times were compared with travel times from EZ Pass toll technology. EZ Pass is an AVI technology based on radio frequency identification (RFID). Travel time data was collected from a 2.9 mile portion of I-76 from August 13, 2009, to August 15, 2009. The average travel time difference between the EZ Pass travel times and travel times generated from MAC address data was 16 seconds (7%).

It can be concluded from published research that average travel times generated from collected MAC address data will generally be less than 10% different than "ground truth" travel times.

2.3 DATA FILTERING

Data filtering research was examined at two different levels. The first level is data filtering at the individual DCU. The second level is the removal of travel time samples after these have been computed from MAC address data collected at two adjacent DCUs.

2.3.1 MAC Address Filtering

There was no discovered published research that discussed the filtering of MAC addresses captured by a single DCU. Several papers (see Section 2.1) mention the reading of the same MAC address multiple times as a vehicle moves within a DCU's coverage area. However, there is no discussion of the MAC addresses collected being non-vehicle related.

2.3.2 Travel Time Sample Filtering

Travel time filtering is used to eliminate travel time samples that represent non-vehicle travel times, or vehicle travel times that are outliers with respect to the corresponding traffic flow conditions. For example, if a vehicle were to pass a DCU, stop at a parking lot temporarily, and then continue on to pass the next DCU, the system should recognize the travel time as an outlier and not include it when estimating the average travel time for that road segment.

A variety of filtering methods applied to travel times generated from MAC address data are found in the literature. Quayle et al. (*2010*) used a "moving standard deviation algorithm." The travel time mean and standard deviation are calculated for a fixed number of neighboring (in time) travel time data samples. If a travel time is a pre-established number of standard deviations above the mean, the obtained travel time is determined to be an outlier. Haghani et al. (*2010*) have proposed a two-step travel time filtering mechanism. The first step is based on a histogram of travel times (they used speed instead of travel times) generated from collected MAC addresses. The histogram is "smoothed" by computing the average of consecutive histogram frequencies. An average of eight consecutive frequencies was used. The "smoothed" histogram is an estimate of the distribution of speeds that occur along a particular road segment, which includes outlier travel times. The outliers are identified as those speeds that occur outside of lower and upper limits. The lower limit is found from the smoothed histogram by starting at the speed with the highest smoothed frequency and moving towards slower speeds. The speed where the smoothed histogram first increases in value is defined as the lower limit. The process is repeated in the direction of faster speeds to find the upper limit. This filtering step is intended to eliminate outliers that are generated from matching MAC addresses that are generated on different trips. A second filtering step is then applied for travel time outlier identification that is similar to the moving standard deviation method of Quayle et al. (*2010*).

The filtering algorithm used in Puckett and Vickich (*2010*) compares each new calculated travel time sample to the most recent average for the road segment. If the new travel time differs by more than a certain percentage (e.g., 25 percent), the new obtained travel time would be labeled "invalid" and would be discarded. The authors have claimed that this filtering method proved to be successful particularly on high volume freeways which do not have much variance in speed.

However, it tends to discard many potentially valid records on arterials where the travel speeds were more varied.

Dion and Rakha (*2006*) suggest several desirable features of an AVI-based Travel Time Filtering Mechanism. First, the algorithm should be able to handle both stable (constant mean) and unstable (varying mean) conditions. Second, it should be applicable to low levels of vehicles equipped with AVI technology (less than one percent). The algorithm should work for both freeway (un-interrupted) and signalized arterial roadways (interrupted). Based on these features, Dion and Rakha (*2006*) introduced a number of existing travel time filtering algorithms and provided brief descriptions for each. The TransGuide algorithm uses a moving average that automatically filters out all recorded travel times that exceed a user-defined threshold. The TranStar algorithm is similar to the TransGuide algorithm. The main difference is that travel times are updated each time new travel information is obtained from a passing vehicle (in most cases, a vehicle equipped with an electronic tag), instead of being updated at fixed intervals. The Transmit algorithm is another filtering method. This algorithm is similar to the TransGuide and TranStar systems. However, instead of using a moving average to obtain estimates of current travel times between vehicle detection stations, this system uses fixed 15-minute observation intervals. For each interval k, the system collects a sample $n(k)$ individual travel times, and uses these samples to estimate a current average travel time. The travel times estimated in previous intervals can be used to smooth new estimated travel times against historical data (i.e., same 15-minute interval in the previous week).

Dion and Rakha (*2006*) conclude that the Transmit algorithm can better reflect short-term fluctuations in traffic conditions than the other algorithms. However, they have claimed that none of the various algorithms reviewed are not capable of tracking abrupt changes in observed travel times without requiring a high sample rate. The new travel time filtering algorithm proposed by Dion and Rakha (*2006*) uses a dynamically varying validity window to identify valid travel times. The validity window is a function of the number of observations within the current sampling interval, the number of observations in the previous interval, and the number of consecutive observations outside the validity window.

Barceló et al. (*2010*) developed a set of techniques to filter travel times generated using collected MAC address data. Their method starts with a lower threshold for the free flow speed, estimated from previous traffic studies. This lower bound is used to remove abnormal travel speeds. The system monitors the aggregated average speed of the vehicles and modifies the current lower bound. Barceló et al. (*2010*) also apply the Kalman Filtering technique. This technique is a forecasting technique that can also be used to identify outliers.

2.3.3 Other Statistical Filtering Methods

A review of statistical techniques that may be applied to filtering travel time samples was conducted. Chandola et al. (*2009*) reviewed statistical procedures for anomaly detection. They divided existing anomaly detection techniques into seven general categories: classification based, nearest neighbor-based, clustering-based, statistical-based, information-theoretic-based, and spectral-based methods. The statistical-based anomaly detection techniques are applicable to travel time data.

Statistical anomaly detection techniques can be categorized as parametric techniques or non-parametric techniques (*Chandola et al. 2009*). Parametric statistical anomaly detection techniques detect outliers by assuming collected data are observations from a specific distribution. In this technique, parameters of an assumed or known distribution are estimated from collected data. Outliers are detected by computing probabilities from the assumed distribution. Schneider et al. (*2009*) filtered outliers of travel time data collected from a tolling system by using distribution-based techniques. They tested two different rules to filter outliers called box plot filtering and z-score filtering. Box plot filtering identifies outliers based on quartile values of a distribution, while z-score filtering identifies outliers based on a normalized measure that is the same as computing a standard normal measure.

Another class of parametric anomaly detection techniques is regression model-based methods. These techniques are typically applied to time series data and are based on the assumption that the time series data follow a specific stochastic process. The moving average process (a specific stochastic process) and the autoregressive process are commonly assumed processes for time series data. Each process has its own mathematical form and after estimating the parameters of the stochastic process, confidence intervals can be constructed. The major assumption of these two processes is that they are *mean stationary*, implying a constant mean over time. This is also referred to as a stationary time series (*Cryer and Chan 2008*). These methods may be applied to travel time data but only over intervals of time where the mean travel time is assumed to be equal.

The last class of parametric techniques consists of methods that assume different distributions for different components represented in collected data. These techniques use multiple distributions and may assume that outliers and non-outlier data are observations from different specific distributions. Some methods assume that non-outlier data consists of observations from multiple distributions. These techniques can be applied for filtering MAC addresses and travel times where different distributions may be assumed for different traffic conditions.

Non-parametric anomaly detection techniques do not rely on any assumptions regarding the probability distribution of the data. The histogram-based method is a simple non-parametric technique where data is assigned to bins and bins are given a score (usually based on frequency). Outliers can then be filtered based on their score. Haghani et al. (*2010*) used a histogram-based technique to filter travel times generated from MAC address data. Quayle et al. (*2010*) used the sample standard deviation measure as a basis to filter outlier travel time data. Another non-parametric technique is a “kernel function” based technique. The key feature of a kernel function based technique is the use of a simple function to approximate the probability density function of the data.

2.4 ANALYSIS OF NON-INDEPENDENT TRAVEL TIME SAMPLES

Assuming that accurate travel time samples are generated from MAC address data, an important next step is to utilize this data to compute estimates of average travel times, variability of travel times, and confidence intervals for average travel times. This will provide personnel utilizing travel time estimates with information on the certainty associated with travel time measures.

A key feature of travel time data that has not been discussed in other Bluetooth–based travel time data collection research is the presence of correlation in the travel time samples. The presence of correlation in collected travel times is easier to see in congested traffic conditions, where slower travel times of multiple samples collected sequentially will indicate that a slow travel time for the next collected travel time sample is likely. With correlated travel times, many "standard" normal theory or large sample-based statistical methods are no longer directly applicable since these methods assume that the data are independent.

There are however, multiple statistical methods that have been developed for discrete event simulation output analysis that can be applied to correlated travel time data. In discrete event simulation, it is common to collect data on system performance measures over simulated time. An example is the hourly throughput of a production line. Such performance measures are typically correlated, so a large amount of research has been devoted to developing statistical procedures for constructing confidence intervals for the mean under such conditions.

Examples of such methods are:

- Batch means - Law and Kelton (*2000*).
- Autoregressive methods - Law and Kelton (2000), Fishman (*1973*).
- Spectrum analysis - Law and Kelton (*2000*), Fishman (*1973*).
- Regenerative method - Law and Kelton (*2000*).
- Standardized time series method - Law and Kelton (*2000*).

Like the regression-based parametric anomaly detection techniques, all of these methods assume that the data analyzed are from a stationary process. To be applicable to travel time analysis, they should only be applied to travel times collected over time intervals where it is reasonable to assume the mean travel time and variance are constant.

2.5 TRAVEL TIME PREDICTION AND FORECASTING METHODS

There has not been a large amount of published research addressing short term traffic condition forecasting based on collected travel time data. Some of this can be attributed to the fact that automated travel time data collection is not widespread and is only available on certain roadways (e.g., road segments with electronic tolling).

Barceló et al. (*2010*) developed forecasts of short term traffic conditions based on collected travel time data. Barceló et al. (*2010*) applied the Kalman Filtering technique to develop forecasts of travel times. Kalman Filtering is a recursive forecasting technique which uses actual measurement data along with predicted data to generate an estimate of future system performance. Barceló et al. (2010) used Kalman filtering to estimate travel speeds in an Origin-Destination (OD) matrix for future time intervals. The traffic speed data was generated from collected MAC address data. These matrices were used to evaluate the performance of a road segment and the behavior of traffic in the roadway segment.

Multiple researchers have examined travel time forecasting from archived data. Wouters et al. (*2005*) constructed a database of historical travel times. The travel time data is segregated by day and season. Types of days include days of the week, school vacation days, and public holidays. They have developed web-based applications that provide estimated travel times for multiple departure times and days, and travel routes of different lengths. Logendran and Wang (*2008*) developed a regression tree based approach for predicting travel times using archived loop detector data. Multiple factors, including weather conditions and day of the week, were used to predict travel conditions.

More research has been published that addresses short-term traffic condition forecasting based on traffic volumes and travel times estimated from loop detector data. Travel times can be estimated by traffic flow (the number of vehicles passing a reference point per unit time) and vehicle density. The three major approaches applied for short-term traffic condition are time series analysis, Kalman filtering, and neural networks.

Chandra and Al-Deek (*2009*) applied autoregressive integrated moving average (ARIMA) models for short-term traffic volume prediction. An ARIMA (p,d,q) model is applied to the d^{th} difference of the time series data, with p autoregressive terms and q moving average terms. Both the autoregressive and moving average models assume a stationary mean, however the stationary mean assumption is not required for the ARIMA model (the d^{th} difference of the time series is assumed to have a stationary mean). ARIMA models can be applied to some non-stationary time series data such as traffic volumes over time. Other time series models have been developed to improve the performance of ARIMA models. Cetin and Comert (*2006*), Kamarianakis et al. (*2005*), and Tsekeris and Stathopoulos (*2006*) are examples of such ARIMA model extensions.

Kalman filters can be applied to non-stationary time series. Okutani and Stephanedes (*1984*) employ Kalman filters to predict short-term traffic volumes in Nagoya, Japan. The average prediction error was less than 9%, and the maximum error less than 30%. Stathopoulos and Karlaftis (*2003*) compare ARIMA with Kalman filtering, and conclude that Kalman filtering is more accurate than ARIMA for urban traffic volume predictions. Yang et al. (*2004*) used recursive least-squares (RLS) which is a special case of a Kalman filter. Xie et al. (*2007*) used a wavelet Kalman filter for traffic condition prediction. They used discrete wavelet decomposition analysis to divide the original data in order to reduce the variability used with a Kalman filter. Wu et al. (*2011*) also used Kalman filters for short-term traffic prediction utilizing data from loop detectors in Bellevue, Washington.

An artificial neural network is another approach that has been used for short-term traffic condition forecasting. Neural networks are closely related to empirical model fitting in statistics and are widely used for pattern matching. Smith and Demetsky (*1994*) applied a neural network for short-term traffic flow prediction. They compared their predictions to predictions from an ARIMA model as well as historical averages. Their neural networks resulted in the smallest errors among the three approaches. Jiang and Adeli (*2005*), Zheng et al. (*2006*), and Bashir and El-Hawary (*2009*) are other examples of the use of neural networks for short-term travel condition prediction.

2.6 PERFORMANCE MEASURES

The automated collection of travel time data using Bluetooth-based DCUs offers the opportunity to obtain very good estimates of travel time performance measures. Current travel-time based performance measures reported by ODOT are the Travel Time Index and Planning Time Index. These measures are defined by the Federal Highway Administration as follows:

- **Travel Time Index:** ratio of the average peak period travel time as compared to a free-flow travel time. The free-flow travel time for each road section is the 15th percentile travel time during traditional off-peak times (weekdays between 9AM-4PM, 7PM-10PM; weekends between 6AM-10PM), not to exceed the travel time at the posted speed limit (or 60 mph where the posted speed is unknown). For example, a value of 1.20 means that average peak travel times are 20% longer than free-flow travel times. In this report, the AM peak period is 6AM-9AM and the PM peak period is 4PM-7PM on non-holiday weekdays. Averages across road sections and time periods are weighted by vehicle miles traveled (VMT).

- **Planning Time Index:** a ratio of the total time needed to ensure 95% on-time arrival as compared to a free-flow travel time. This measure is computed for the AM peak period (6AM-9AM) and the PM peak period (4PM-7PM) for non-holiday weekdays. For example, a value of 40% means that a traveler should budget an additional 8-minute buffer for a 20-minute average peak trip time to ensure 95% on-time arrival. The planning time index is computed as the 95th-percentile travel time of the month divided by the free-flow travel time for each road section and time period. The free-flow travel time for each road section is the 15th percentile travel time during traditional off-peak times (weekdays between 9AM-4PM, 7PM-10PM; weekends between 6AM-10PM), not to exceed the travel time at the posted speed limit (or 60 mph where the posted speed is unknown). Averages across road sections and time periods are weighted by VMT.

The inputs for both of these measures are:

- Average free-flow speeds for each section during off-peak times.
- Five-minute section-level speeds and VMT for peak traffic periods (6AM-9AM, 4PM-7PM).

Both of these inputs can be obtained from the travel time data obtained from collected MAC addresses. In addition to the prior measures, much more detailed information will be available for road segments with DCUs installed.

Various measures are used to quantify intersection performance. *Control delay* is defined as the difference between the travel time that would have occurred in the absence of the intersection control, and the travel time that results because of the presence of the intersection control. Wolfe et al. (*2007*) describe other arterial and intersection performance such as traffic density, total or average delay, stops, and predicted travel time. Typically, engineers conducting an arterial analysis use the procedures outlined in the most recent *Highway Capacity Manual Update* and/or use a computer methodology that is designed to measure the performance from vehicle arrival rates and bandwidths for a fixed-time signal system. Required data to complete such analysis may come from conducting manual on-site observations or using advanced data collection sensors such as inductive loop detectors.

Section 5.0 will describe how data obtained from collected MAC addresses can be utilized to obtain some intersection performance measures. Because of the information available in this data, the performance measures will be similar to but not identical to previously described intersection performance measures.

2.7 INFORMATION SYSTEMS DESIGN AND IMPLEMENTATION

A potential information systems architecture to support a widespread Bluetooth-based travel time data collection system is depicted in Figure 2.1. In the context of this research, *architecture* is defined as "the organizational structure of a system, identifying its components, their interfaces, and a concept of execution among them" (*FHWA 2006*).

The components of the architecture include Bluetooth-based data collection units (DCUs) for capturing time-stamped MAC addresses, a database server that hosts a database management system (DBMS) to compute, filter, and store travel time data generated from the MAC addresses collected by the DCUs, and a server that hosts software applications to manage the interaction among the DCUs and to schedule the collection of the MAC address data. As illustrated in Figure 2.1, these components are connected via a network to facilitate remote monitoring and troubleshooting.

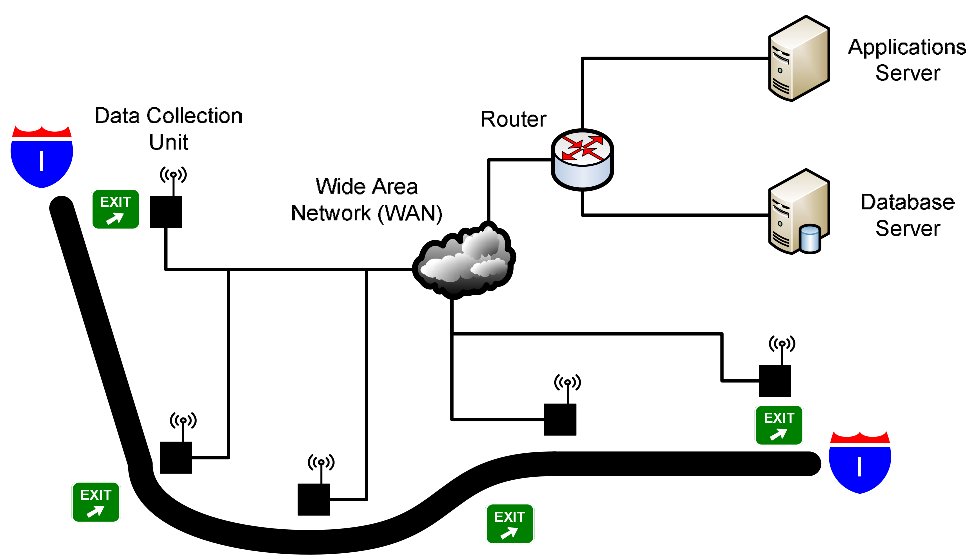

Figure 2.1: Architecture for travel time estimation system based on MAC addresses

At the heart of an information systems architecture is the notion of different layers, each of which has distinct responsibilities. The *presentation* layer is responsible for the display of information to system users and the handling of users' requests. Users may be humans or another system for which services are provided. The *infrastructure* layer is responsible for communicating with other systems to request services that may be needed. This layer is often dominated by the integration with DBMSs (to handle the data needs), but it can also include transactions processors, messaging, and legacy applications (i.e., older or highly customized software). The *domain* layer deals with the essence of what the information system is about, i.e., the business rules used to process the data that is relevant to the users. A *layered* information systems architecture is easier to maintain and administer since the hardware and software implemented at the different layers can be easily replaced or upgraded without affecting the layer above or below it (*FHWA 2006; Fowler 2002; Stallings 2010*).

2.7.1 Information Systems Challenges in a Widespread Bluetooth-based Travel Time Data Collection System

Researchers currently investigating Bluetooth-based travel time estimation have used different approaches to collect, filter, store, and process MAC address data for travel time estimation. These approaches are generally driven by the capabilities available in the DCU. For example, stand-alone battery-operated DCUs can collect and locally store MAC address data for several days (*Tarnoff and Young 2010*) or may have capabilities to enable online data retrieval via the Internet (*Malinovskiy et al. 2010*). The DCUs currently in use by ODOT may be powered with a direct current (DC) jack or by using a power over Ethernet (POE) injector, and store MAC address data locally in a file that is later retrieved by a software application running in a remote server.

It is evident that the operation of a widespread Bluetooth-based travel time data collection system composed of DCUs installed permanently on highways and arterial roads and at intersections would present a number of technical challenges to the information system architecture that supports it, particularly with regards to the volume of MAC address data being transmitted over the network by the multiple DCUs. Unfortunately, no published research was found on issues related to information flow volume in large-scale intelligent transportation systems (e.g., toll roads). This can be attributed to the fact that this information may be proprietary and, therefore, not publicly available.

Another important consideration is how frequently MAC address data should be extracted from the DCUs to generate travel time samples and which device (i.e., the DCU or the server) should initiate and control data transmission. In general, data communication systems can be distinguished according to whether they are based on a *push* or *pull* model. In a *push* model, data items are received from the clients (i.e., the DCUs) without explicit requests from the server. In contrast, in a *pull* model data items are broadcast by the clients in response to explicit requests received from the server (*Aksoy and Leung 2005*).

The next sections discuss the advantages and disadvantages of the *push* and *pull* models. It is important to note that the decision regarding which of these models would be implemented in a widespread Bluetooth-based travel time data collection system depends on each individual organization. Restrictions such as privacy, data security or organizational rules, and currently established practices will dictate which model is used.

2.7.1.1 Push Model

A *push* model is appropriate when transactions need to be distributed as they happen, making the applications *talk* when the transactions are available. The information systems that manage stock market and other types of financial institution transactions are employ a *push* technology model (*Aksoy and Leung 2005*). Table 2.2 shows the main advantages and disadvantages of the *push* model (adapted from *Nelson 2008*).

Table 2.2: Advantages and Disadvantages of the Push Model

Advantages	Disadvantages
1. Instant transaction communication. 2. Feedback on the transaction after the business processes are invoked. 4. Transaction by Transaction / Guaranteed delivery mechanisms. 5. Visual Business Rule Processing Engines are usually in place. 6. Tactical in nature. 7. New sources can come on-line and push out new transactions integrating with ease into existing layers.	1. Independent transactions (i.e., cannot rely on history or trends.) 2. Difficult to establish context. 3. Cannot transform massive sets of data at once due to current computing power limitations. 4. Once a transaction is sent, it is gone. No "recorded history."

2.7.1.2 Pull Model

In the *pull* model of data delivery, the transfer of data from clients to servers is initiated by a server pull. When a server request is received by a client (i.e., the DCU), it responds by locating the requested information. Also, in a pull model, clients must be interrupted continuously to deal with requests from servers. Furthermore, the information that servers can obtain from a client is limited to when and what servers know to ask for.

The nature of pull systems requires huge amounts of processing power to get the job done right. Acharya et al. *(1997*) showed that in highly asymmetric cases, the server can become a bottleneck and using the server to simply pull all packets (a unit of transmitted data) is not a good strategy. Table 2.3 shows the main advantages and disadvantages of the pull model (adapted from *Nelson 2008*).

Table 2.3: Advantages and Disadvantages of the Pull Model

Advantages	Disadvantages
1. Massive sets of transactions in parallel/ partitioned can be handled in ever smaller execution windows. 2. Increase in processing power means increase in data set that can be dealt with. 3. The information needed can be obtained when needed via scheduling. 5. Predictive support, predictive failures, predictive model (facilitates standardization and automation.) 6. Strategic in nature.	1. Requires massive sets of processing power (for large volumes of data). 2. Batch windows are continually shrinking while data sets are ever growing. 3. Little to no visibility into the transactions as they occur.

3.0 DATA FILTERING AND GENERATION OF TRAVEL TIME SAMPLES

In Section 1.1.2, background information was presented about data filtering and the complications that arise when computing travel times samples from collected MAC address data. In this section, the solutions developed to address data filtering and travel time sample generation will be described.

3.1 MAC ADDRESS DATA FILTERING

Because active Bluetooth devices detected by a DCU may not always be devices residing in a traveling vehicle, some of the MAC address data collected will not be utilized. To reduce the amount of data stored at the DCU and transferred to a central server, a procedure for filtering MAC addresses not contained in traveling vehicles has been developed and implemented. This procedure is based on the concept of a “group” of MAC address records. A group is defined as a collection of MAC address reads for the same MAC address. A group should represent one trip (in a single direction) of the corresponding Bluetooth-enabled device through the coverage area of the DCU along the road that it is monitoring.

Assuming MAC addresses not contained in traveling vehicles have been filtered, the remaining MAC address data can be organized into groups of MAC addresses. For efficiency and to increase the accuracy of travel time samples computed from MAC address data, a procedure has been developed, tested, and implemented that identifies a single MAC address from each group to keep. This single MAC address per group will be stored locally at the DCU and later transferred and used for travel time sample generation. The received signal strength indicator (RSSI) is the key parameter used in this procedure to select the single best MAC address record to keep from a group.

Both of these procedures will be described in this section. To simplify the presentation, a “non-vehicle MAC address” will refer to MAC addresses of Bluetooth devices detected but not contained within a passing vehicle, and a “vehicle MAC address” will refer to detected MAC addresses of Bluetooth devices contained within passing vehicles.

3.1.1 Forming MAC Addresses Groups

The concept and definition of a group of MAC addresses has been presented. Although the concept of a group is straightforward, a procedure for the identification of groups is required. The DCUs simply collect and store MAC address records in the order that they were discovered (see Figure 3.1, which is the same as Figure 1.1). The procedure for identifying groups will be explained assuming a file of MAC address records has been collected for a specific period of time (the implementation creates and finalizes groups as MAC addresses are collected).

Assuming a file of MAC addresses had been collected, the first step required to form groups is to sort the file by MAC address (Figure 3.1).

MAC Addresses in Sequence		
MAC Add.	Day	Time
2:83:AC:3	8/17/2010	17:00:09
0:D1:BA:6	8/17/2010	17:00:13
0:D1:BA:6	8/17/2010	17:00:21
5:F9:FB:8	8/17/2010	17:00:53
5:F9:FB:8	8/17/2010	17:00:57
A:5E:22:8	8/17/2010	17:00:57
5:F9:FB:8	8/17/2010	17:01:02
5:F9:FB:8	8/17/2010	17:01:06
5:F9:FB:8	8/17/2010	17:01:10
5:F9:FB:8	8/17/2010	17:01:14
5:F9:FB:8	8/17/2010	17:01:19
A:CC:9B:5	8/17/2010	17:01:27
A:CC:9B:5	8/17/2010	17:01:31
A:CC:9B:5	8/17/2010	17:01:47
A:CC:9B:5	8/17/2010	17:01:51
A:5E:22:8	8/17/2010	17:01:59
A:5E:22:8	8/17/2010	17:02:15
C:2B:BD:0	8/17/2010	17:03:06
9:C0:9B:2	8/17/2010	17:03:10
C:2B:BD:0	8/17/2010	17:03:10
9:C0:9B:2	8/17/2010	17:03:15
C:2B:BD:0	8/17/2010	17:03:15
C:21:14:6	8/17/2010	17:04:33
C:21:14:6	8/17/2010	17:04:37
C:21:14:6	8/17/2010	17:04:41
8:6F:2D:F	8/17/2010	17:05:32
A:5E:22:8	8/17/2010	17:06:08
A:5E:22:8	8/17/2010	17:07:02

MAC Addresses Sorted		
MAC Add.	Day	Time
0:D1:BA:6	8/17/2010	17:00:13
0:D1:BA:6	8/17/2010	17:00:21
2:83:AC:3	8/17/2010	17:00:09
5:F9:FB:8	8/17/2010	17:00:53
5:F9:FB:8	8/17/2010	17:00:57
5:F9:FB:8	8/17/2010	17:01:02
5:F9:FB:8	8/17/2010	17:01:06
5:F9:FB:8	8/17/2010	17:01:10
5:F9:FB:8	8/17/2010	17:01:14
5:F9:FB:8	8/17/2010	17:01:19
8:6F:2D:F	8/17/2010	17:05:32
9:C0:9B:2	8/17/2010	17:03:10
9:C0:9B:2	8/17/2010	17:03:15
A:5E:22:8	8/17/2010	17:00:57
A:5E:22:8	8/17/2010	17:01:59
A:5E:22:8	8/17/2010	17:02:15
A:5E:22:8	8/17/2010	17:06:08
A:5E:22:8	8/17/2010	17:07:02
A:CC:9B:5	8/17/2010	17:01:27
A:CC:9B:5	8/17/2010	17:01:31
A:CC:9B:5	8/17/2010	17:01:47
A:CC:9B:5	8/17/2010	17:01:51
C:21:14:6	8/17/2010	17:04:33
C:21:14:6	8/17/2010	17:04:37
C:21:14:6	8/17/2010	17:04:41
C:2B:BD:0	8/17/2010	17:03:06
C:2B:BD:0	8/17/2010	17:03:10
C:2B:BD:0	8/17/2010	17:03:15

Figure 3.1: Sample of MAC address data in order of discovery (left), and sorted by MAC address (right)

When examining the time-sorted records of the same MAC address (e.g., address 5:F9:FB:8 in Figure 3.1) groups are formed based on a pre-defined "group threshold time." The "group threshold time" is defined as the maximum time between consecutive MAC address records in a group. As an example, assume the group threshold time is set to 30 seconds. The seven records for MAC address 5:F9:FB:8 will form a single group since no consecutive records are separated by more than 30 seconds. However, for MAC address A:5E:22:8, four groups would be formed. The group sizes would be one, two, one, and one starting from the earliest record and moving to the last record.

Analysis was conducted on MAC address data obtained from DCUs installed on Highway 99 in Tigard, Oregon. The number of MAC address records collected by five DCUs installed on Highway 99 in Tigard, Oregon for approximately five days is shown in Table 3.1.

Table 3.1: The Quantity of MAC Address Data Collected to Analyze Group Threshold Time Effects

DCU #	DCU Location	Data Collection		Number of Collected Mac-address Records
		Start	End	
DCU 9	99W I-5 SB / 64th Ave	2/17/2011 18:22:25	2/20/2011 11:33:26	102,716
DCU 10	99W 217 NB	2/15/2011 13:07:49	2/20/2011 11:31:46	286,067
DCU 11	99W Main St / Johnson St	2/15/2011 12:38:39	2/20/2011 11:37:35	254,928
DCU 12	99W McDonald St / Gaarde St	2/15/2011 10:22:36	2/20/2011 11:29:33	394,760
DCU 13	99W Durham Rd	2/16/2011 9:39:05	2/20/2011 11:27:50	242,811

The analysis conducted examined how different group threshold times affected the total number of MAC address groups created from the data collected by each DCU. Figure 3.2 shows plots of the total number of groups as a function of group threshold time.

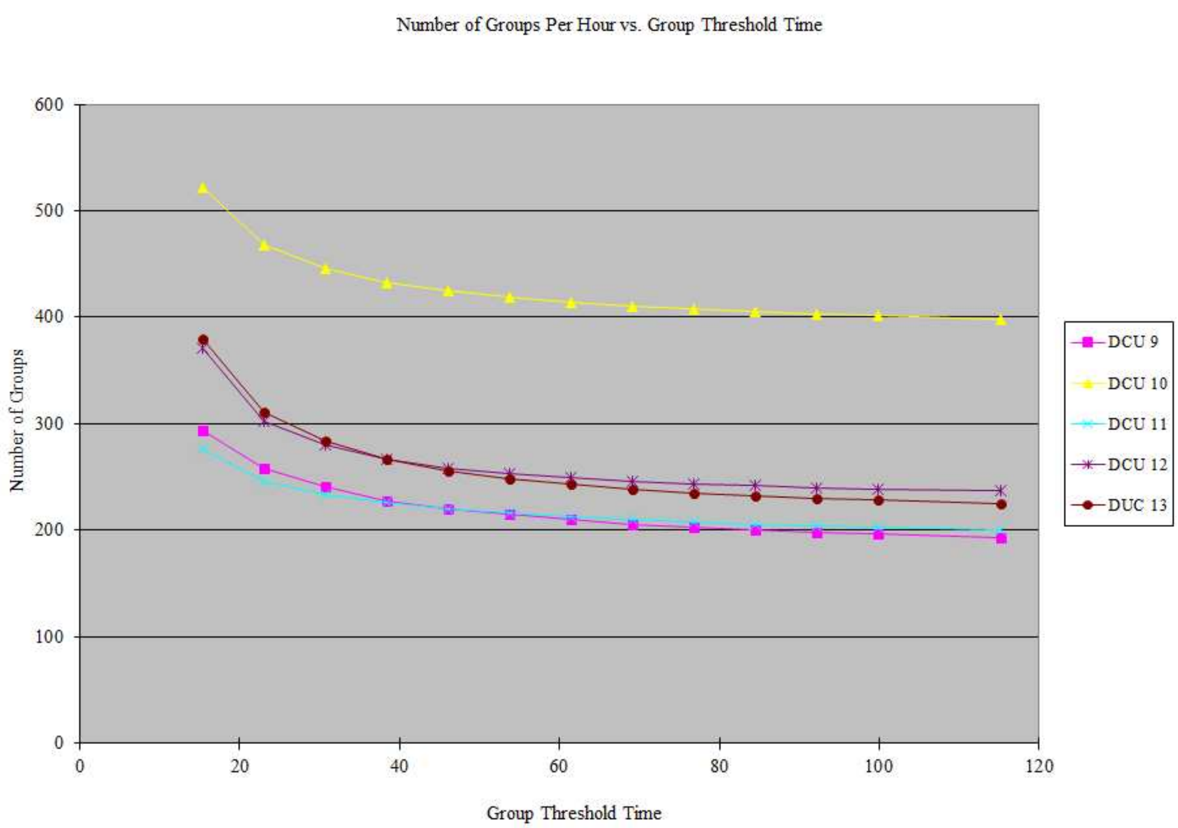

Figure 3.2: Total number of groups as a function of group threshold time

The slopes of the curves in Figure 3.2 stabilize around a group threshold time of 50 seconds. As the group threshold time is increased from a starting value of ten seconds the number of groups drops rapidly and then decreases at a steady and relatively low rate after approximately 45 seconds (the number of groups per hour decreases less than one per second). One explanation of this behavior is related to the coverage area of the DCUs. The total DCU coverage range along the highway is approximately 400-500 meters (approximately 200-250 meters north and south of the DCU). If vehicles travel through the intersections where the DCUs are installed at an average speed of 22.5 MPH, then the average vehicle is in the coverage area of a DCU between 40 and 50 seconds. If a vehicle MAC address is first detected as it enters the coverage area, and then next detected as it is leaving the coverage area a group threshold time of 45-50 seconds will put these two MAC address records in the same group, which in this hypothetical example is correct. The average speed over a period of several days on road segments between the DCUs on HW99W was approximately 27 MPH. Since the average speed was over the whole road segment between the DCUs, the average speed in the coverage area of a DCU will be lower since the DCU coverage areas each include a signalized intersection.

It is possible that some vehicles with active Bluetooth devices leave the DCU coverage area, turn around, and then re-enter the coverage area within a total time that is less than the group threshold time but this is most likely very infrequent. As the group threshold time increases, a component of the reduction in groups will occur due to combining MAC address records from separate trips past the DCU into a single group, which is not desired.

The recommended group threshold time should be close to the average time a vehicle requires to move through the DCU coverage area plus sufficient extra time to account for obstructions and other unforeseen events that may cause gaps in groups. The group threshold time will be based on the behavior of the total number of groups as a function of the group threshold time. Probe vehicle tests indicate that an effective group threshold time is in the 60-70 second range. Since the Bluetooth DCUs have the inquiry period length (shortest time interval between consecutive reads of the same MAC address) set at 3.84 seconds, an integer multiplier of 3.84 seconds should be used as the group threshold time. The implemented group threshold time is 65.28 seconds = 17 * 3.84 seconds.

3.1.2 Filtering Non-Vehicle MAC Addresses

The basic idea used to identify non-vehicle MAC addresses is that vehicles traveling on the road where a DCU is located will be in the coverage area of the DCU antenna for a specific time that is a function of vehicle speed. During this time, an active Bluetooth device may be detected multiple times forming a group of MAC address records (using the group threshold time of 65.28 seconds). The number of MAC address records in a group is primarily a function of the following factors:

- The time a vehicle (with a BT device) is in the DCU coverage area as determined by vehicle speed.
- The strength of the radio frequency signal emitted by the mobile BT device.
- Group threshold time – The maximum time between consecutive records within a group.

- The minimum time between consecutive reads of the same MAC address (i.e., the length of the Bluetooth inquiry period which is an integer multiple of 1.28 seconds).
- The number of non-vehicle MAC addresses detected.
- The probabilistic nature of the Bluetooth inquiry procedure.

Because of these factors, the size of a group is a random variable. The general form of the group size probability distribution can be estimated by examining a histogram of group sizes obtained from collected data. A histogram of group sizes observed from DCU12 (see Table 3.1) is shown in Figure 3.3.

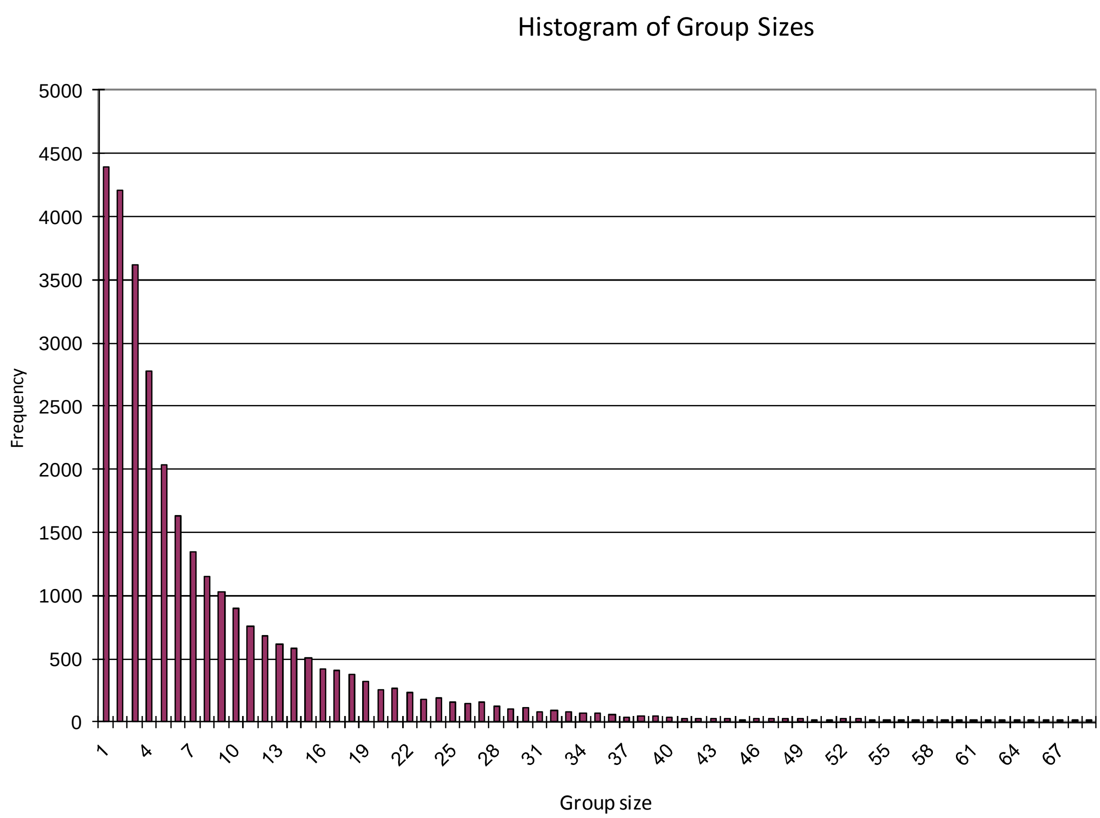

Figure 3.3: Group size histogram from DCU 12 data using a group threshold time of 65.28 seconds

Since a vehicle traveling past a DCU will be in the coverage area of the DCU for a limited amount of time, very large group sizes will indicate non-vehicle MAC addresses and thus a group size number can be utilized to identify and filter non-vehicle MAC addresses. Table 3.2 shows how the cumulative percentage of total groups and total MAC address records increases as a function of group size.

Table 3.2: The Cumulative Number of Groups and MAC Address Records as a Function of Group Size for DCU 11

Group Size	Number of Groups	Cumulative % of Groups	Number of MAC address Records	Cumulative % of Records
=< 10	22,342	85.31%	84,167	33.04%
=< 20	24,927	95.18%	121,250	47.60%
=< 30	25,692	98.10%	140,124	55.01%
=< 40	25,897	98.88%	147,227	57.80%
=< 50	25,984	99.21%	151,101	59.32%
=< 80	26,057	99.49%	155,647	61.10%
=< 100	26,077	99.57%	157,456	61.81%
=< 200	26,129	99.77%	164,495	64.57%
=< 7986 (Total)	26,190	100.00%	254,736	100.00%

When establishing a group size number to filter out non-vehicle MAC addresses, a trade-off exists between eliminating vehicle MAC addresses and keeping non-vehicle MAC addresses. Since many non-vehicle MAC addresses will be eliminated in the travel time generation procedure completed at the central server (if no matches at another DCU occur), a group size of 50 provides a good balance of MAC address elimination at the DCU (approximately 40% for DCU 11 in Table 3.2) and minimization of non-vehicle MAC addresses passed to the central server. Table 3.3 shows a similar analysis of groups for data collected from a single DCU for six days on Wallace Road in Salem, Oregon. The results from Wallace Road show that smaller group sizes are more predominant than on Highway 99W, which can be explained by the fact that the DCU is not near an intersection as are the DCUs on Highway 99W, and that the average speed on Wallace Road is greater than average speed on Highway 99W. Even though a group size of 50 for filtering non-vehicle MAC addresses eliminates fewer groups on Wallace Road, it does result in a higher fraction of data filtered at the DCU level.

Table 3.3: Cumulative number of groups and MAC address records as a function of group size for a Wallace Road DCU

Group Size	Number of Groups	Cumulative % of Groups	Number of MAC address Records	Cumulative % of Records
=< 10	8,367	95.34%	22,785	40.29%
=< 20	8,559	97.53%	25,522	45.13%
=< 30	8,625	98.28%	27,168	48.04%
=< 40	8,664	98.72%	28,551	50.49%
=< 50	8,686	98.97%	29,557	52.27%
=< 80	8,721	99.37%	31,779	56.20%
=< 100	8,734	99.52%	32,920	58.22%
=< 200	8,756	99.77%	35,997	63.66%
=< 5108 (Total)	8,776	100.00%	56,549	100.00%

Filtering of non-vehicle MAC addresses utilizing a group threshold time of 65.28 seconds and a group size filter value of 50 has been implemented on all Bluetooth DCUs that have been installed and operating for this project.

3.1.3 Identifying a Single MAC Address from Each Group

As a result of non-vehicle MAC address filtering, group sizes for the remaining vehicle MAC addresses will range from 1 to 50. The next step of DCU level data filtering implemented selects a single MAC address from each vehicle MAC address group to store and transfer to the central server. For the generation of accurate travel time samples, the MAC address selected from each group should be the record with the time stamp corresponding to when the vehicle was closest to the DCU. Here "closest" means the closest to an imaginary line extending from the DCU across and perpendicular to the road.

The method developed to select the MAC address record is based on the received signal strength indicator (RSSI). The RSSI is a measure of the strength of wireless communication occurring between the DCU and the mobile Bluetooth device (RSSI values are measured in units of decibels (dB)). The RSSI can be obtained during the Bluetooth inquiry procedure at the same time that the MAC address is read and thus does not add any additional processes and delays when reading MAC addresses. RSSI levels are known to be correlated with distance where a larger RSSI value indicates that the DCU and Bluetooth device are closer together than a lower RSSI value.

With this measure available, the method for selecting a single MAC address from a group differs for DCUs located at intersections and DCUs located adjacent to free-flowing roads as follows:

- For DCUs located at intersections, the method is to select the record after which the subsequent RSSI values decrease at the highest rate. In those cases where this decrease is not observed, the last record in a group is saved. Often the record saved is the record with the largest RSSI value.
- For DCUs located adjacent to free-flowing roads, the record saved is the record with the largest RSSI value.

Details of obtaining the RSSI values and validation of this method are presented next.

3.1.3.1 Obtaining RSSI Values

There are two types of possible solutions for acquiring the Bluetooth RSSI measurements: the connection-based solution and the inquiry-based solution. In the connection-based solution, a communication connection between the DCU and a mobile Bluetooth device must be established before RSSI measurements can be obtained. However in this case, the transmission power adjustment (used to adjust power based on RSSI to conserve power) eliminates the correlation between the RSSI measurement and the distance between a mobile Bluetooth device and the DCU.

The inquiry-based solution retrieves the RSSI measure from the inquiry response without establishing any connection to the mobile Bluetooth device and has been implemented on all DCUs. To obtain RSSI values, an inquiry command is sent to the Host Controller Interface (HCI) of the Bluetooth module. The controller will reply with a series of standard HCI event packets. The first packet is the “inquiry with RSSI event”. This packet contains information relevant to the inquiry process including the mobile device’s MAC address and the corresponding RSSI.

3.1.3.2 Validation Experiments

Experiments in three different environments were conducted to assess differences between the time stamp of the selected record, and the time the vehicle crosses an imaginary line extending from the DCU antenna across and perpendicular to the road. A schematic of the general physical experimental setup is presented in Figure 3.4. A DCU and antenna are installed at some location adjacent to the roadway being monitored (point A in Figure 3.4). The distance *d*, from point A in Figure 3.4 and the roadway will vary depending on the available mounting structures at the location where the DCU is placed.

In these experiments, a probe vehicle containing active Bluetooth devices with known MAC addresses travels past a DCU multiple times. At the same time there is a person operating a laptop computer that is communicating with the DCU. The communication of the DCU and laptop computer permits synchronization of the DCU time and laptop time. Software is provided to help the laptop operator record the exact time that a vehicle passes the DCU (crosses line AB in Figure 3.4). When the front of the vehicle just passes line AB in Figure 3.4, the operator will press the “Enter” key on the laptop keyboard. When this key is pressed a time stamp will be automatically generated and stored in a file.

If feasible in a particular test environment the vehicle will be driven past the DCU at a fixed speed. Different speeds can be examined to see if speed has an effect on the differences between the timestamp of the selected record, and the time the vehicle passes the DCU.

The DCU will be scanning continuously during the experiment. Knowing the time that the vehicle passed the DCU and assuming a constant travel speed, the time that the vehicle was a specific distance from the DCU can be computed. For example, the time that the vehicle is at points 1, 2, and 3 in Figure 3.4 can be computed from the time the vehicle passed the DCU (point x). The distance that each point is from point x (d1, d2, d3) is known. By matching the MAC address timestamps to various locations, it is possible to map RSSI values to different vehicle locations.

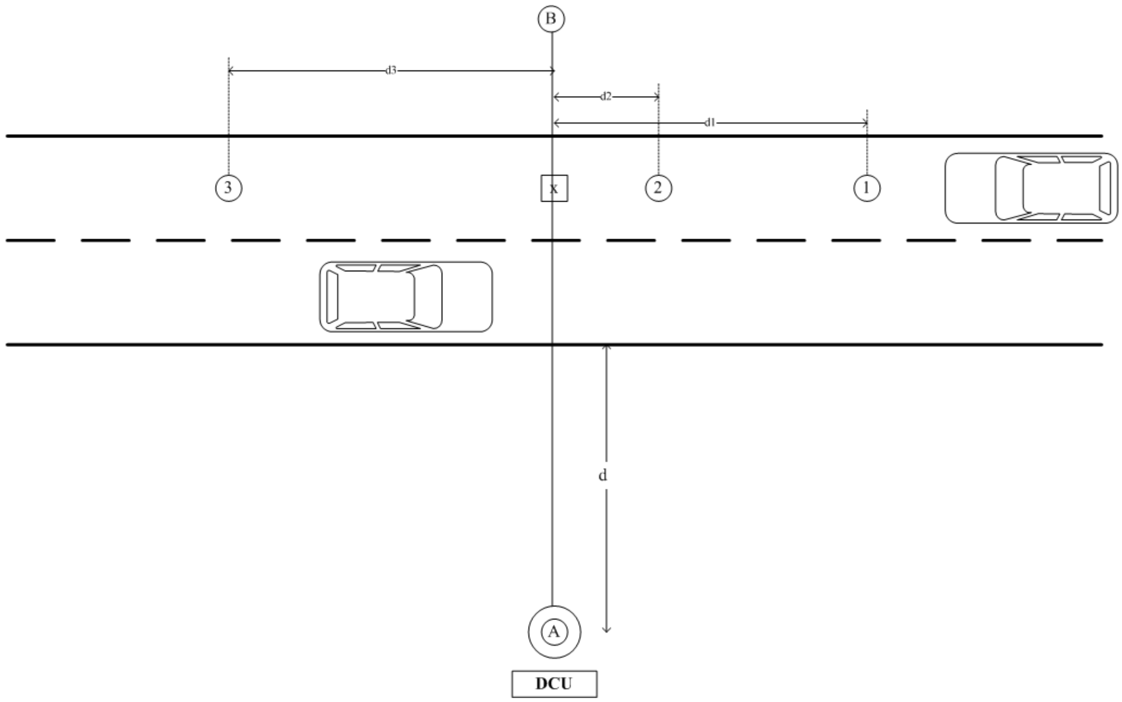

Figure 3.4: General configuration for testing if MAC address records with the largest RSSI values correspond to when the vehicle just passes the DCU

The first test environment used for validation was a low traffic, free flowing rural road near Corvallis, Oregon (Camp Adair Road adjacent to EE Wilson Wildlife Area – Figure 3.5). Due to low traffic volumes it was possible to drive at various constant speeds past the DCU. The probe vehicle traveled passed the DCU 30 times (15 times traveling east and 15 times traveling west) for each speed tested. The speeds tested were 25, 35, and 45 mph. The DCU was located 36 feet from the road and the antenna was mounted on a tripod at a height of 70 inches. Two different cell phones with Bluetooth communications active were used in the tests. The responses computed from the recorded data were the time differences between the MAC address records with the highest RSSI reading and the times the vehicle crossed the line drawn from the DCU perpendicular to the road. A histogram of all the test results is shown in Figure 3.6. Most time differences are less than two seconds, with a maximum difference of 7.5 seconds. The average time difference was 0.23 seconds, with a standard deviation of 1.37 seconds. Negative time differences occur when the time stamp of the highest RSSI record occurs after the vehicle passes the DCU.

Figure 3.5: DCU installation on Camp Adair Road, Benton County

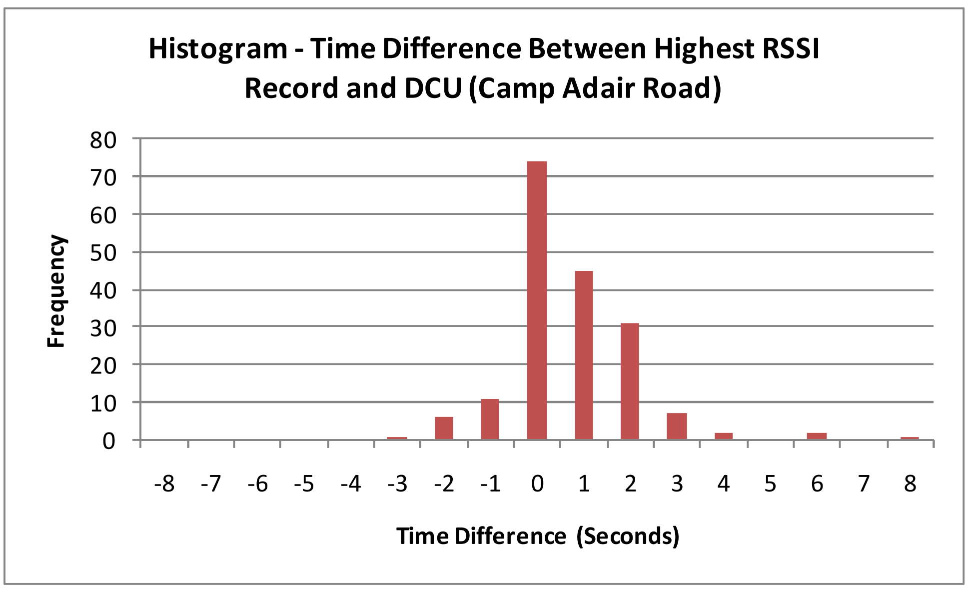

Figure 3.6: Histogram of the difference between the time the probe vehicle passed the DCU on Camp Adair Road and the MAC address record with the highest RSSI

The second test environment was Wallace Road, which is located on the west side of the city of Salem, Oregon. This road is also a free-flowing road with no signals located near the DCU, but a single signal between the DCU locations. The Oregon Department of Transportation (ODOT) Intelligent Transportation Systems (ITS) Unit operates a test site on this road located at latitude-longitude of N 44.972858 and W -123.066321 (see Figure 3.7). Wallace Road runs north and south with two lanes in both directions, and the speed limit is 45 MPH. Forty total probe vehicle runs (20 traveling northbound and 20 traveling southbound) were made past the DCU, with two different cell phones with active Bluetooth communications present in the vehicle.

The responses computed from the recorded data were the time differences between the MAC address records with the highest RSSI reading and the times the vehicle crossed the line drawn from the DCU perpendicular to the road. A histogram of all the test results is shown in Figure 3.8. Most time differences are less than two seconds, with a maximum difference of five seconds. The average time difference was 0.23 seconds (the same as for Camp Adair Road) with a standard deviation of 1.24 seconds. Negative time differences occur when the time stamp of the highest RSSI record occurs after the vehicle passes the DCU.

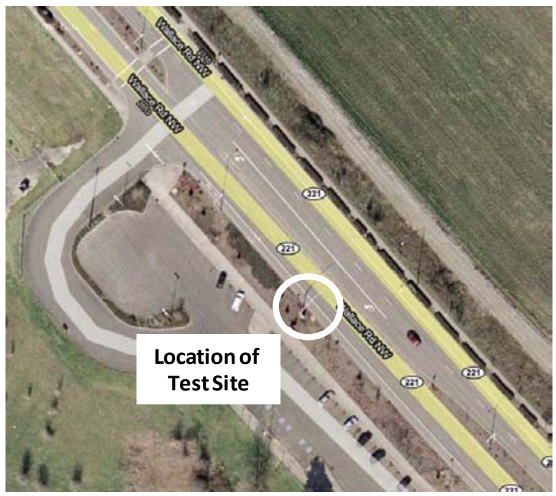

Figure 3.7: DCU location on Wallace Road, Salem, Oregon

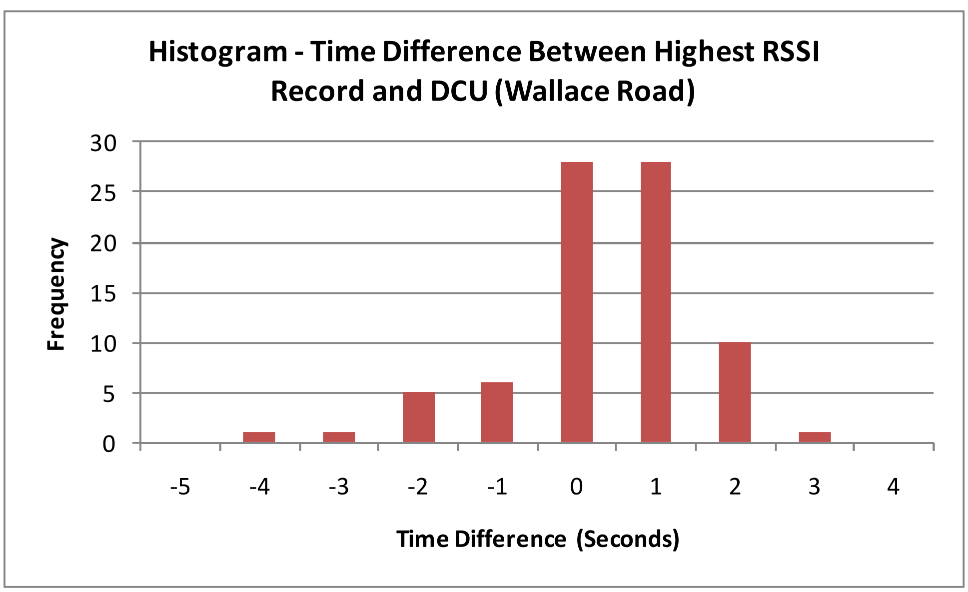

Figure 3.8: Histogram of the difference between the time the probe vehicle passed the DCU on Wallace Road and the MAC address record with the highest RSSI

The third test environment was along Highway 99W in Tigard, Oregon. Five DCUs have been installed at five different intersections (see Figures 3.9 and 3.10).

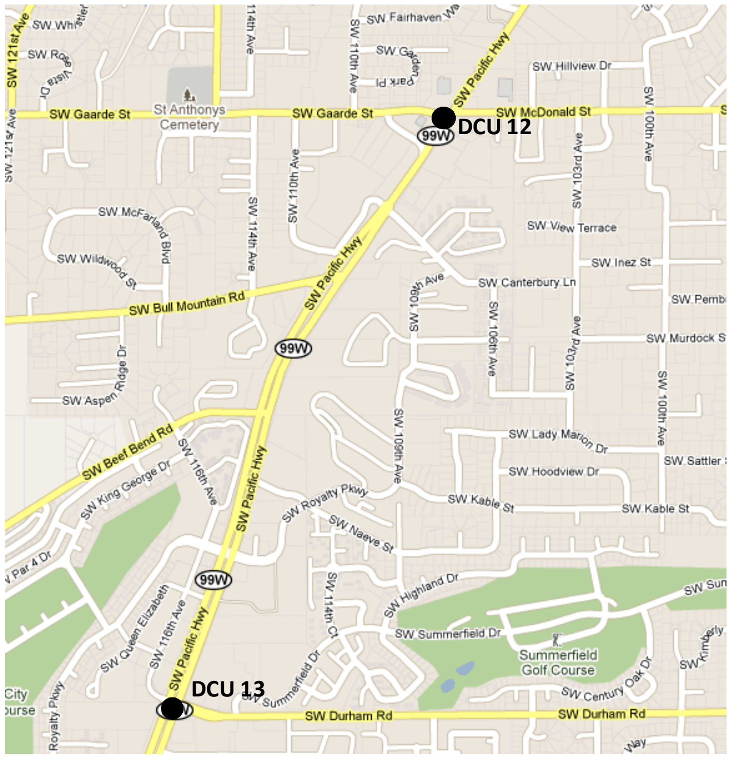

Figure 3.9: Southernmost DCU locations on Highway 99W, Tigard Oregon

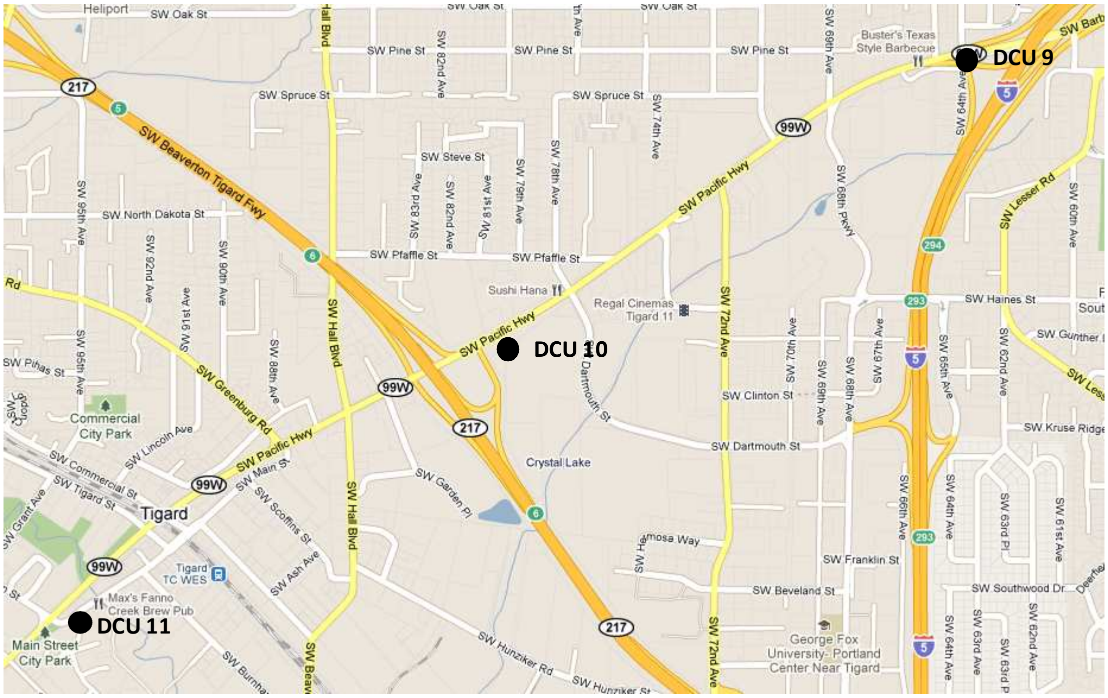

Figure 3.10: Southernmost DCU locations on Highway 99W, Tigard Oregon

The Highway 99W tests were conducted at DCU 12 which is located at a signalized intersection. Forty total probe vehicle runs (20 traveling northbound and 20 traveling southbound) were made past the DCU, with two different cell phones with active Bluetooth communications present in the vehicle. Two responses were recorded and analyzed.

The first response computed from the recorded data were the time differences between the MAC address records with the highest RSSI reading and the times the vehicle crossed the line drawn from the DCU perpendicular to the road. This is the same response as used in the prior test environments, which were both free-flowing roads. A histogram of all the test results is shown in Figure 3.11. Most time differences are less than two seconds, with a maximum difference of 43 seconds. The average time difference was 3.1 seconds with a standard deviation of 9.9 seconds. The results were similar to the other test environments except for five test runs where the response was greater than 11 seconds (42, 41, 18, 12, and 12 seconds). With these five responses removed, the average, maximum, and standard deviation of the response are -0.03 seconds, 3.3 seconds, and 1.3 seconds respectively.

In those instances where large time differences were observed, the probe vehicle was stopped by the signal and stationary. The stationary position of the probe vehicle was one or two vehicle lengths before the line drawn from DCU 12 perpendicular to the road. When stationary, yet close to the DCU, higher RSSI readings were obtained than when the probe vehicle was moving across the line drawn from the DCU. In this case, the MAC address record with the highest RSSI reading occurred when the probe vehicle was close in distance to the line drawn from the DCU, but still relatively far with respect to time since the vehicle was waiting for a green light to proceed. As seen in Figure 3.11, all of the relatively large time differences are positive differences, which occur when the MAC address with the highest RSSI reading occurs before the probe vehicle passes the line drawn from the DCU. Negative time differences occur when the time stamp of the highest RSSI record occurs after the vehicle passes the DCU. In such cases as just described the accuracy of the travel time samples generated will be reduced. The time interval between when the highest RSSI reading was recorded and when the vehicle passed the DCU will be added to the travel time along the road section that occurs *after* the signal. Similarly, this time difference will not be included in the travel time for the road section occurring before the signal.

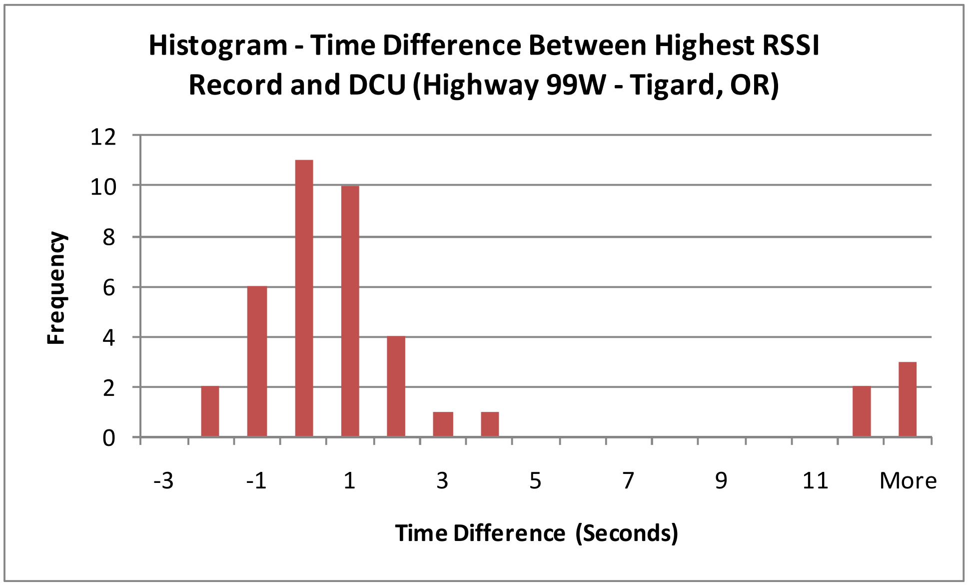

Figure 3.11: Histogram of the difference between the time the probe vehicle passed DCU 12 on Highway 99W and the MAC address record with the highest RSSI

The second response recorded and analyzed is based on a method to eliminate these occasional large errors caused when a vehicle stops just before passing the DCU. In this method the single record selected from a group of MAC addresses is the record after which the subsequent RSSI values decrease at the highest rate. In those cases where this decrease is not observed the last record in a group is saved. Since the highest RSSI record can often be when a vehicle stops close to, but before passing the DCU, the RSSI values should also decrease rapidly once a vehicle passes the DCU after the vehicle has a green signal.

Figure 3.12 shows the time stamps and associated RSSI for two Bluetooth devices (two cell phones) when a vehicle stops and waits close to the DCU as it travels through the intersection. The large circles represent the time stamps associated with the highest RSSI. Since the vehicle stops near the DCU the time stamps which are associated with the highest RSSI are considerably earlier than the time when the vehicle passes the DCU.

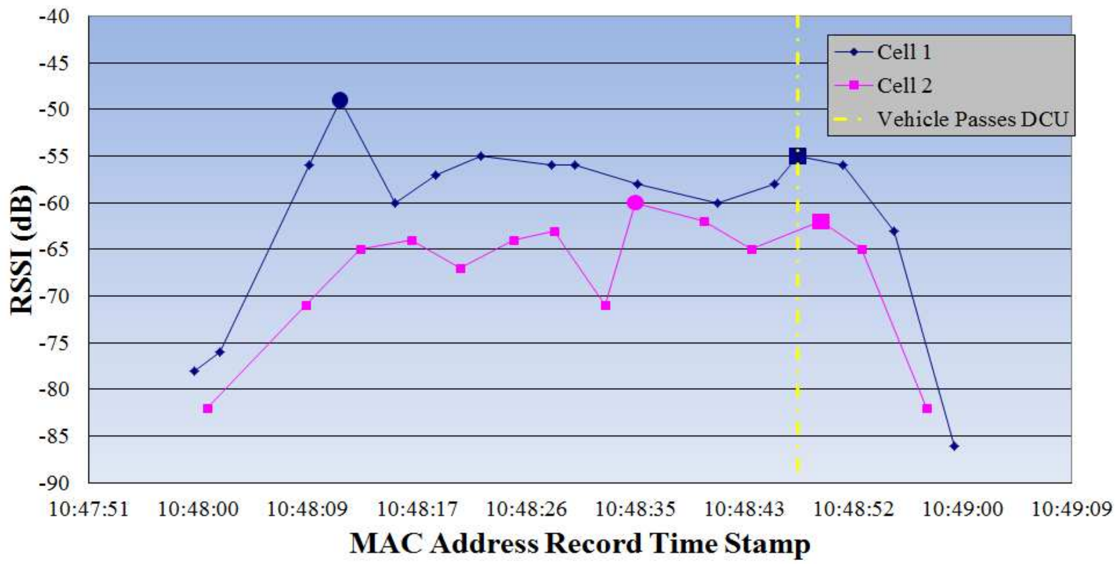

Figure 3.12: A sample RSSI profile for a vehicle stopping at an intersection

The responses computed using this method were compared to the times the probe vehicle crossed the line drawn from the DCU perpendicular to the road. A histogram of all the test results is shown in Figure 3.13. Most time differences are less than two seconds, with a maximum difference of 21 seconds. The average time difference was 2.2 seconds with a standard deviation of 4.1 seconds. The performance is more accurate and consistent than saving the record with the highest RSSI value.

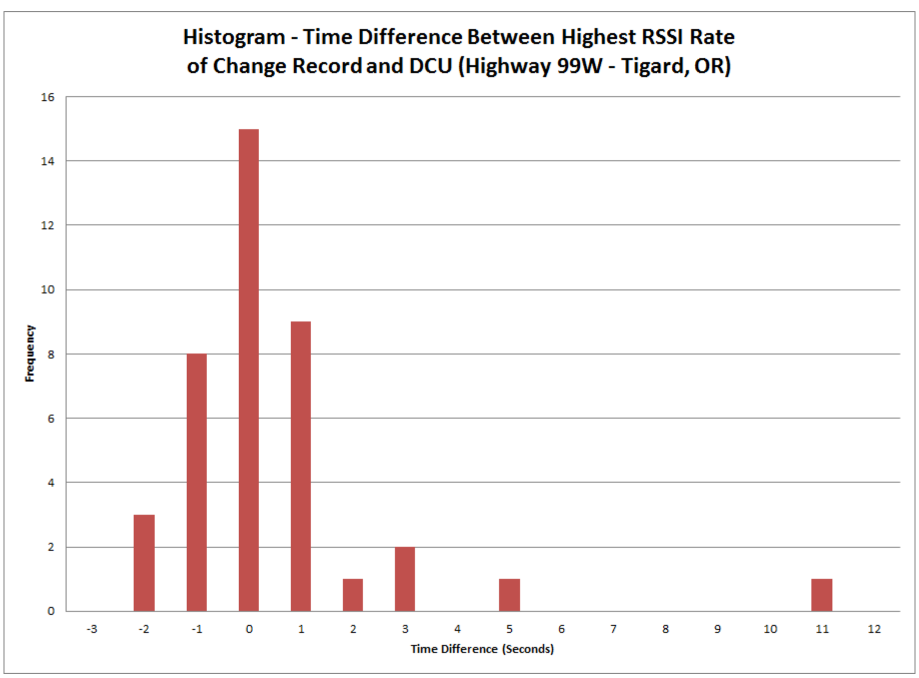

Figure 3.13: Histogram of accuracy of the time stamp before the RSSI values rapidly decrease

3.1.4 Filtering Performance

Filtering of non-vehicle MAC addresses, and saving one MAC address record per group based on RSSI values has been implemented on DCUs installed on Wallace Road in Salem, Oregon and Highway 99W in Tigard, Oregon. Storing a single MAC address per group results in 13% of the total vehicle MAC address records detected at a DCU being stored and transferred. When non-vehicle MAC addresses are filtered from these records, approximately 8% of the total MAC address records read by a DCU are stored and then transferred from each DCU. Monitoring of the Highway 99W DCUs has indicated that the total amount of data kept at the DCUs (to be transferred to the central server) is approximately 10% of the amount prior to implementation of MAC address filtering.

3.2 GENERATING TRAVEL TIME SAMPLES

Assuming two DCUs are collecting data at two ends of a road segment of interest, filtering of MAC addresses is occurring at each DCU, and the MAC address data is downloaded to a central server, travel time samples for the road segment need to be generated.. Travel time sample generation is one task executed as part of an overall Bluetooth DCU-based travel time performance estimation system. The details of travel time sample generation will be presented in this section. Travel time sample generation implementation specifics (pseudo-code) will be presented in Section 6.0.

3.2.1 Generating Travel Time Samples

Once it has been determined that travel time samples should be generated for a road segment between two DCUs (referred to as DCU 1 and DCU 2), the downloaded MAC address records will be utilized to generate these samples. As part of this process a maximum travel time parameter (T_{max}) must be specified. This parameter is used to prevent the generation of travel time samples that could result from MAC address records for the same vehicle that occur from different trips past the same DCUs over a relatively long period of time.

Since a single MAC address record is stored per MAC address group as a vehicle travels past DCU 1, if the same MAC address is read at DCU 2 there will typically be a single way to calculate a travel time sample. The travel time sample generated will be equal to the difference in time between the “matching” MAC address records at DCU 1 and DCU 2.

In some cases, it is possible for the same vehicle to make multiple trips past the DCUs in a time period that is less than T_{max}. An example is a delivery vehicle that travels along the road segment between DCU 1 and DCU 2 multiple times within a time period of T_{max} minutes. If both DCU 1 and DCU 2 detect the vehicle on every trip, the calculation of travel time samples will be straightforward since the MAC address records will form natural pairs of matching records. However, if either or both DCUs do not detect the vehicle on every trip past the DCU there will be a choice about which records from DCU 1 and DCU 2 are matched. In this case, the decision rule implemented matches the MAC address records that give the shortest travel time samples. The rationale for this decision rule is that if a vehicle makes multiple trips past DCU 1 and DCU 2 it must pass one DCU twice before returning and passing the other DCU for a second time. This can be seen in Figure 3.14 where multiple trips past DCU 1 and DCU 2 over time are depicted by the diagonal line. The small rectangles indicate that the MAC address of the Bluetooth device in the vehicle has been read by the DCU. Under the decision rule and assuming a constant travel speed, a travel time would be generated from the DCU records connected by the thick lines which indicate correct matches. The pairs of MAC address records which are circled would generate a travel time sample that would be from incorrectly matched records. Notice that in these instances, one DCU missed a detection of the MAC address, but the MAC address is detected after the vehicle turns around and passes the same DCU in the other direction. Assuming the time taken to turn around and return past the same DCU is large relative to travel times, these incorrectly generated travel times will be identified as outliers and removed.

From an overall system perspective, the fraction of travel time samples that involve the logic just described is very small (roughly 2.5%). Therefore, if an incorrect travel time sample is not removed as an outlier, it will have little impact on travel time performance measures.

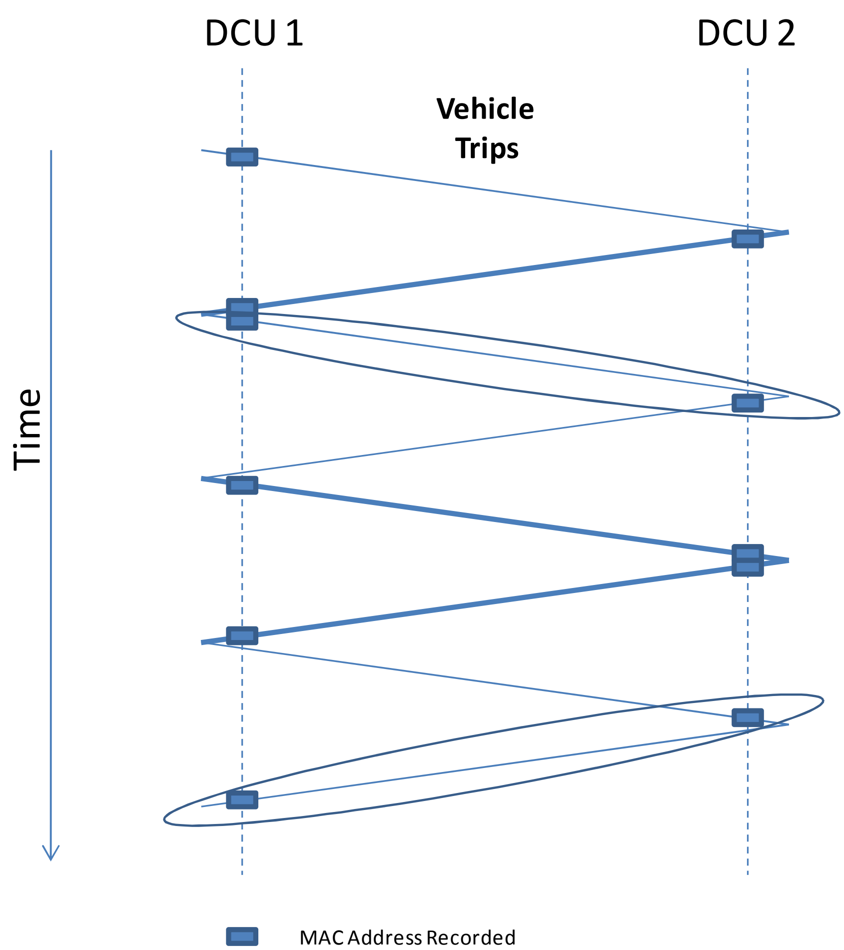

Figure 3.14: Travel time samples generated when a vehicle makes multiple trips in a short time period

Prior to generating travel time samples, there must be a check that a MAC address recorded at a DCU has also been recorded at an upstream or downstream DCU. If this is not the case, the MAC address record is discarded. Assuming records exist for a MAC address at both DCUs, the detailed logic implemented for generating travel time samples is described in Section 6.3. This logic must be repeated for all MAC addresses that have been read at both DCUs.

3.2.2 Accuracy Testing

Accuracy testing of travel times generated from collected MAC address data was assessed on two different types of road segments using probe vehicle data as "ground truth" travel times. Accuracy testing was conducted on a 0.75-mile segment on Wallace Road in Salem, Oregon (see Porter et al. 2011 for DCU placement and installation details), and on two different Highway 99W round trips routes encompassing DCU 12 (see Figure 3.9). For all travel time sample accuracy testing, the start and the end of a route (which can be a road segment) were defined by imaginary lines drawn from a DCU perpendicular to the road traveled.

The 0.75 mile Wallace Road segment has no stop signs or signals located near the DCU, but a single signal between the DCU locations. Traffic flows freely and normally there is no congestion. Twenty (20) total probe vehicle runs were conducted on Wallace Road (10 runs traveling north and 10 runs traveling south). The probe vehicle held two active Bluetooth enabled cell phones with known MAC addresses. The cell phones were placed within reach of the driver but otherwise there was no control over cell phone location within the vehicle. Summary results of the travel time sample accuracy are shown in Table 3.4. The first column gives the average travel time of the probe vehicle recorded manually by a passenger in the vehicle. The second and third columns show the differences for each probe vehicle run between the manually recorded travel time, and the travel time generated from the time stamped MAC address records for each cell phone present in the probe vehicle. The travel time samples were generated from a single MAC address record saved from each group using the highest RSSI value. As a comparison, travel time samples were also generated using the average time of the first and last records in a MAC address group. When utilizing this method to generate travel time samples the average, standard deviation, maximum, and minimum difference in seconds from the probe vehicle travel times are -1.49, 4.77, 6.30, and -10.8 respectively for cell phone 1. For cell phone 2, the corresponding statistics are 0.76, 3.29, 8.80 and -3.70. The accuracy of the travel time samples generated using the record with the highest RSSI value is clearly better for cell phone 1, and roughly the same for cell phone 2 when compared to using the average time of the first and last records in a MAC address group. Statistical tests on the paired differences in absolute error confirm at a 95% confidence level that the travel time sample errors are significantly less for cell phone 1 using the highest RSSI records. For cell phone 2, the null hypothesis that the mean travel time sample errors are the same when using both methods is not rejected.

Table 3.4: Comparison of probe vehicle travel times and travel times estimated from Bluetooth DCU data on Wallace Road

Trial	Manual Travel Time (Seconds)	Manual TT - BT TT Cell 1 (Seconds)	Manual TT - BT TT Cell 2 (Seconds)
1	50.60	-2.40	-2.40
2	78.30	0.30	-2.70
3	65.80	-0.20	6.80
4	57.00	1.00	0.00
5	54.70	0.70	-0.30
6	65.40	-1.60	-0.60
7	52.20	1.20	3.20
8	57.70	0.70	1.70
9	52.80	-0.20	-1.20
10	53.40	0.40	0.40
11	53.20	-0.80	-0.80
12	54.00	-1.00	1.00
13	52.20	2.20	-2.80
14	52.20	0.20	2.20
15	56.40	-1.60	-0.60
16	55.30	-2.70	-4.70
17	52.40	1.40	-0.60
18	53.40	-0.60	-1.60
19	51.10	2.10	9.10
20	56.00	0.00	-1.00
Avg.	**56.21**	**-0.04**	**0.26**
StdDev.	**6.62**	**1.36**	**3.22**
Max	**78.30**	**2.20**	**9.10**
Min	**50.60**	**-2.70**	**-4.70**

Highway 99W round trip route number 1 starts at DCU 12 and moves south along the 1.1 mile long road segment to DCU 13. The vehicle passes DCU 13, turns around and then the returns northbound past DCU 13 and back to DCU 12. Between the two signalized intersections at SW Durham Road and SW McDonald Street where the DCUs are located, there are four other signalized intersections. Highway 99W roundtrip route number 2 starts at DCU 12 and moves north for a short distance before turning around and returning south and back to DCU 12. These round trip routes were utilized because accurate and consistent measurements of when the probe vehicle traveled past DCU 12 were recorded by an outside observer at DCU 12.

Ten probe vehicle runs traveling north and ten runs traveling south passing both DCU 12 and DCU 13 on each run were conducted. The probe vehicle held two active Bluetooth enabled cell phones with known MAC addresses. The cell phones were placed within reach of the driver but otherwise there was no control over cell phone location within the vehicle. Nineteen (19) travel time samples for the round trip routes described were generated from the probe vehicle runs. Summary results of the travel time sample accuracy are shown in Tables 3.5 and 3.6. The travel time samples were generated from a single MAC address record saved per group. The saved MAC address records were the records after which the subsequent RSSI values decreased at the highest rate (see Section 3.1.3.2).

Table 3.5: Comparison of probe vehicle travel times and travel times estimated from Bluetooth DCU data on Highway 99W round trip route number 1

	Highway 99W Round Trip 1 (9 travel time samples)				
	Manual TT	Manual -BT TT		Manual -BT TT	
Trip	(seconds)	Cell 1 (secs)	Cell 1 (%)	Cell 2 (secs)	Cell 2 (%)
1	349.27	0.58	0.17%	2.18	0.62%
2	323.48	0.74	0.23%	2.03	0.63%
3	449.8	0.72	0.16%	2.14	0.48%
4	483.8	6.32	1.31%	2.51	0.52%
5	338.95	0.03	0.01%	0.05	0.01%
6	429.92	0.91	0.21%	0.45	0.10%
7	461.9	12.89	2.79%	3.05	0.66%
8	351.04	0.76	0.22%	0.04	0.01%
9	375.87	0.29	0.08%	0.28	0.07%
	Average	**2.58**	**0.58%**	**1.41**	**0.35%**
	Std Dev	**4.32**	**0.92%**	**1.19**	**0.29%**
	Max	**12.89**	**2.79%**	**3.05**	**0.66%**
	Min	**0.03**	**0.01%**	**0.04**	**0.01%**

Table 3.6: Comparison of probe vehicle travel times and travel times estimated from Bluetooth DCU data on Highway 99W round trip route number 2

	Highway 99W Round Trip 2 (10 travel time samples)				
	Manual TT	Manual -BT TT		Manual -BT TT	
Trip	(seconds)	Cell 1 (secs)	Cell 1 (%)	Cell 2 (secs)	Cell 2 (%)
1	118.09	-0.82	-0.69%	-1.99	-1.69%
2	162.76	-1.93	-1.19%	-1.53	-0.94%
3	109.39	-0.42	-0.38%	-3.57	-3.26%
4	132.52	-1.17	-0.88%	-2.27	-1.71%
5	154.49	-4.33	-2.80%	-3.50	-2.27%
6	152.71	-1.30	-0.85%	-0.10	-0.07%
7	142.70	-1.82	-1.28%	-0.02	-0.01%
8	159.69	-11.80	-7.39%	-3.87	-2.42%
9	223.05	0.38	0.17%	-1.14	-0.51%
10	124.99	-0.99	-0.79%	-3.56	-2.85%
	Average	**-2.42**	**-1.61%**	**-2.16**	**-1.57%**
	Std Dev	**3.52**	**2.17%**	**1.45**	**1.15%**

Max	**0.38**	**0.17%**	**-0.02**	**-0.01%**
Min	**-11.80**	**-7.39%**	**-3.87**	**-3.26%**

For comparison, Table 3.7 shows the summary statistics for the travel time samples over the same trips generated using a single MAC address saved per group based on the highest RSSI value. For DCUs installed at intersections, saving the MAC address record prior to a rapid decrease in RSSI results in better performance. However, with the variability in the travel time sample errors, the differences are not statistically significant at a 95% confidence level for the number of probe vehicle runs conducted.

Table 3.7: Travel time accuracy when using the highest RSSI to save one record per MAC address group

	Highway 99W Round Trip 1			
	Cell 1 (secs)	**Cell 1 (%)**	**Cell 2 (secs)**	**Cell 2 (%)**
Average	**7.05**	**1.53%**	**9.13**	**2.02%**
Std Dev	**13.90**	**2.85%**	**15.18**	**3.33%**
Max	**42.55**	**8.79%**	**45.06**	**10.02%**
Min	**0.29**	**0.08%**	**0.04**	**0.01%**
	Highway 99W Round Trip 2			
	Cell 1 (secs)	**Cell 1 (%)**	**Cell 2 (secs)**	**Cell 2 (%)**
Average	**-6.49**	**-4.22%**	**-8.06**	**-5.73%**
Std Dev	**12.45**	**8.02%**	**14.66**	**10.90%**
Max	**-0.42**	**-0.38%**	**3.56**	**2.85%**
Min	**-40.56**	**-26.25%**	**-45.44**	**-34.29%**

3.2.3 Travel Time Sampling Rates

Since only some vehicles will contain enabled Bluetooth devices that can be detected by the DCUs, data was collected on the quantity of travel time samples as a fraction of traffic volume on Highway 99W. The number of travel times generated by data collected from the DCUs between 10:30AM, December 6, 2011 and 11:50AM, December 19, 2011 was divided by the traffic volumes obtained from loop detector data for the same time period. The loop detector volumes were from loop detectors located at one end of the road segment. This was completed for the four road segments defined by the DCUs. This data is shown in Table 3.8.

Table 3.8: The volume of travel time samples generated compared to traffic volume

Road Segment	Travel Direction	Avg. Daily # of Travel Times	Avg. Daily Traffic Volume	Travel Time Vol./ Traffic Vol.
64th street -	North	1306	21192	6.20%
217 NB ramp	South	1369	23423	5.80%
217 NB ramp -	North	1243	25764	4.80%
Johnson	South	1118	19515	5.70%
Johnson -	North	1359	22424	6.10%
McDonald	South	1287	19735	6.50%
McDonald -	North	1243	20052	6.20%
Durham	South	1128	13375	8.40%

The percentages shown in Table 3.8 will underestimate the percentage of vehicles traveling the road segments for which a travel time sample was obtained. This is because the traffic volumes will include vehicles turning off of Highway 99W onto intersecting roads, and vehicles entering Highway 99W from intersecting roads that do not travel a whole road segment.

3.3 IDENTIFYING TRAVEL TIME SAMPLE OUTLIERS

Multiple methods for identifying outlier travel times generated from Bluetooth DCUs have been published. The moving standard deviation method, a simple rule-based method, and a smoothed histogram-based method have been applied by Quayle et al. (*2010*), Puckett and Vickich (*2010*), and Haghani et al. (*2010*), respectively. The moving standard deviation method assumes that travel times are independent and normally distributed. The simple rule-based method used by Puckett and Vickich (*2010*) identifies a travel time as an outlier if it is a fixed percentage (e.g., 25%) larger or smaller than the most recent average travel time (moving average). The histogram-based method in Haghani et al. (*2010*) estimates the travel time distribution by "smoothing" the frequencies in a histogram of travel times. No distributional assumptions are made in the latter two methods, but the histogram-based method is intended to identify "travel times" that should not have been generated (e.g., matching MAC addresses produced on different trips on different days). In this section three different methods for outlier detection are examined. One method is the moving standard deviation method and the second is similar to the simple rule based method of Puckett and Vickich (*2010*) but is based on travel time percentiles. The third method is a rule based method that was developed to exploit the characteristics found in travel time data collected.

3.3.1 Highway 99W Travel Time Data Characteristics

Figure 3.15 through Figure 3.17 show histograms of *unfiltered* travel time samples generated from collected MAC address data between DCU 12 and DCU 13 on Highway 99W for three different time periods. These were compiled to empirically examine the probability distribution of travel times between signalized intersections on Highway 99W.

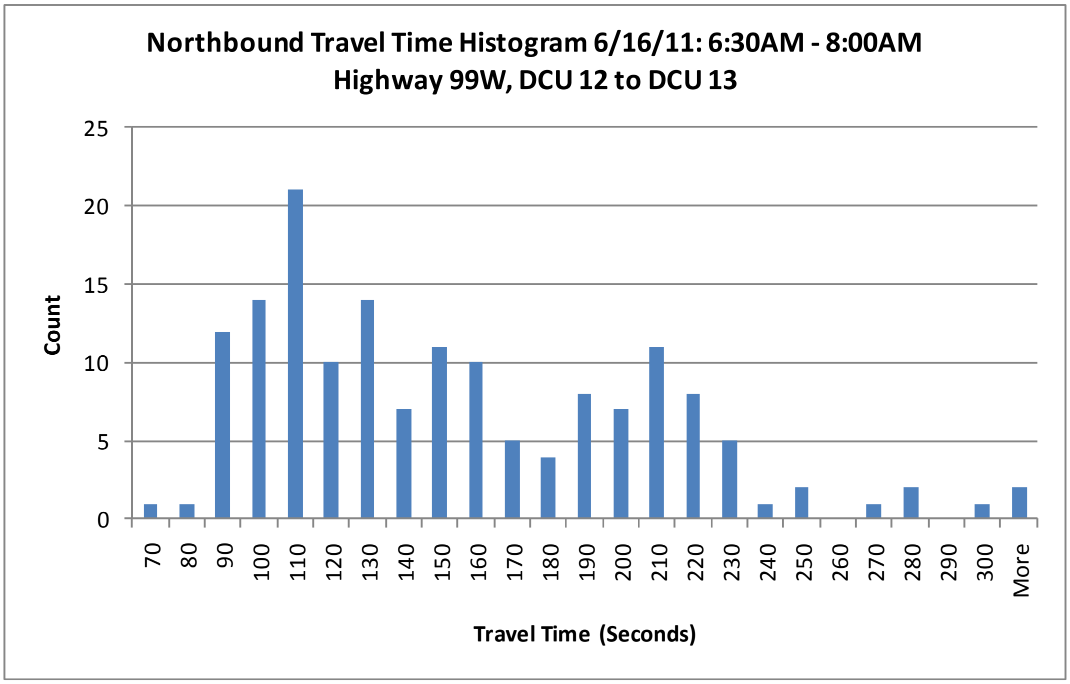

Figure 3.15: Histogram of travel times for trips from DCU 12 to DCU 13, 6:30 AM to 8:00AM

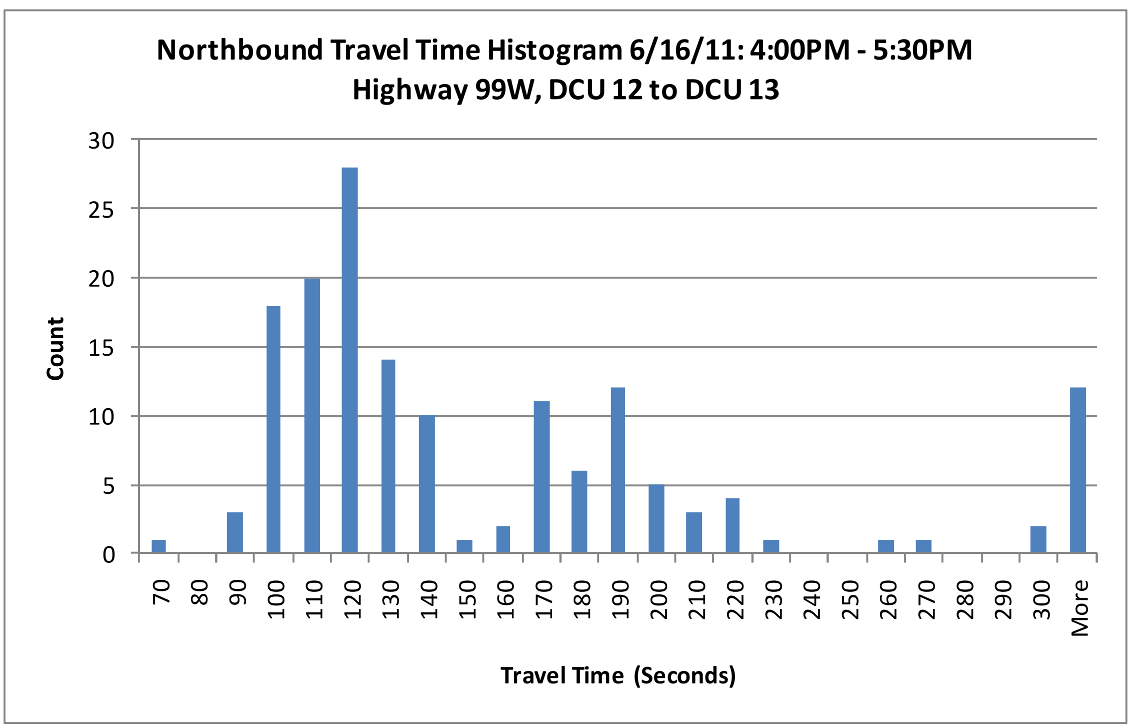

Figure 3.16: Histogram of travel times for trips from DCU 12 to DCU 13, 4:00 PM to 5:30 PM

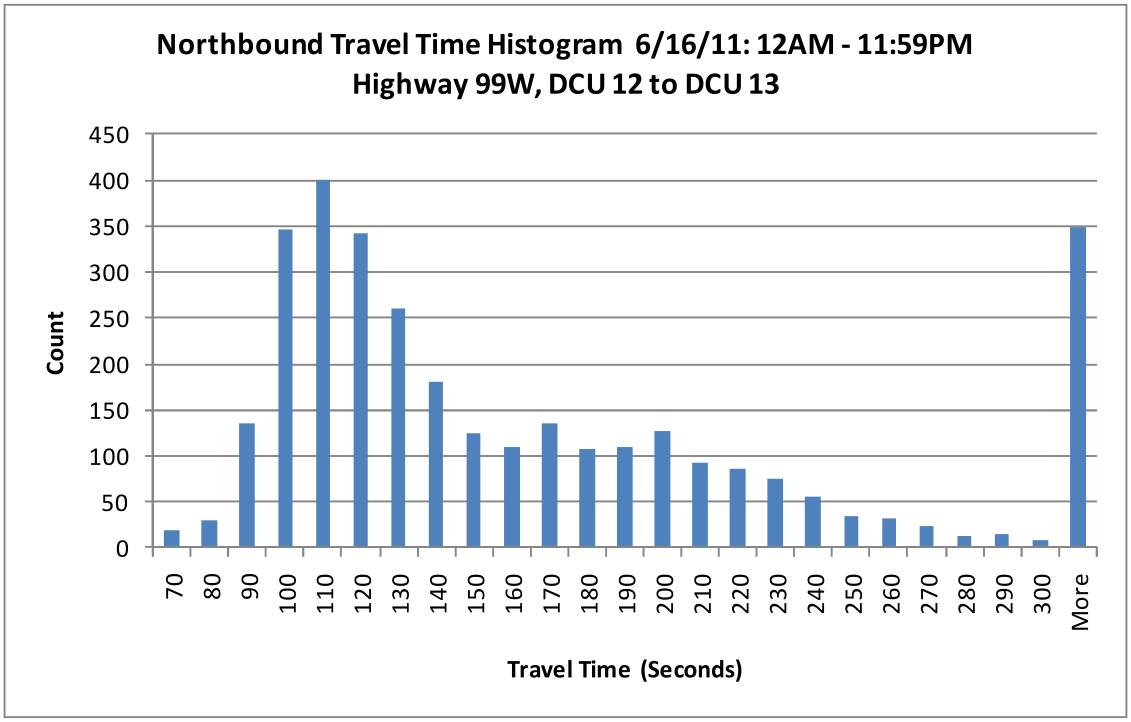

Figure 3.17: Histogram of travel times for trips from DCU 12 to DCU 13, 12:00AM to 11:59 PM

The histograms highlight two important facts. First, each histogram presents evidence that the travel times between DCU 12 and DCU 13 are not normally distributed. It appears that the signalized intersections between DCU 12 and DCU 13 are creating multiple modes in the travel time histograms (i.e., modes in the probabilistic sense – most frequently occurring values). The non-normally distributed travel times theoretically preclude the application of the moving standard deviation method (in particular to filter short travel times), and the multimodal travel time distribution does not allow the method of Haghani (*2010*) to be applied. The second important observation is that the histograms show that the general shape of the travel time distribution is similar for different non-overlapping time periods, but the average travel time changes.

3.3.2 Testing

Several travel time outlier filtering methods were tested on northbound travel time samples collected between DCU 12 and DCU 13 on June 16-17, 2011. For all methods, a moving window of the latest 100 travel time samples was used as a basis for computing the criteria by which outliers are identified and removed. The travel time outlier identification methods tested are listed next.

- **Moving standard deviation method.** Multiples of the sample standard deviation are added and subtracted from the sample average. A multiple of 1.65 (e.g., 90% of the observations from a normal distribution) will fall between the mean ± 1.65 * standard deviation.

- **Box-plot method.** This method identifies the time length (L) between the travel time samples at the 25th and 75th percentiles, and then considers any travel times that are less than the (25th percentile point – L), or the (75th percentile point + L) to be outliers.
- **Gap method.** This method is empirical and is used to identify long travel times that are outliers. It is based on the observation that there is often a large gap (i.e., a length of time with no travel time samples) between non-outlier and outlier travel times. The method tested defines the critical gap length to be equal to (0.5 * median travel time). Starting at the median travel time and sequentially moving to longer travel times, the time difference between the samples is computed. If a gap greater than the critical gap time is found, the last travel time used in the computations and all longer travel times are considered outliers.

Plots showing the computed travel times and those identified as outliers are shown in Figure 3.18 through Figure 3.20.

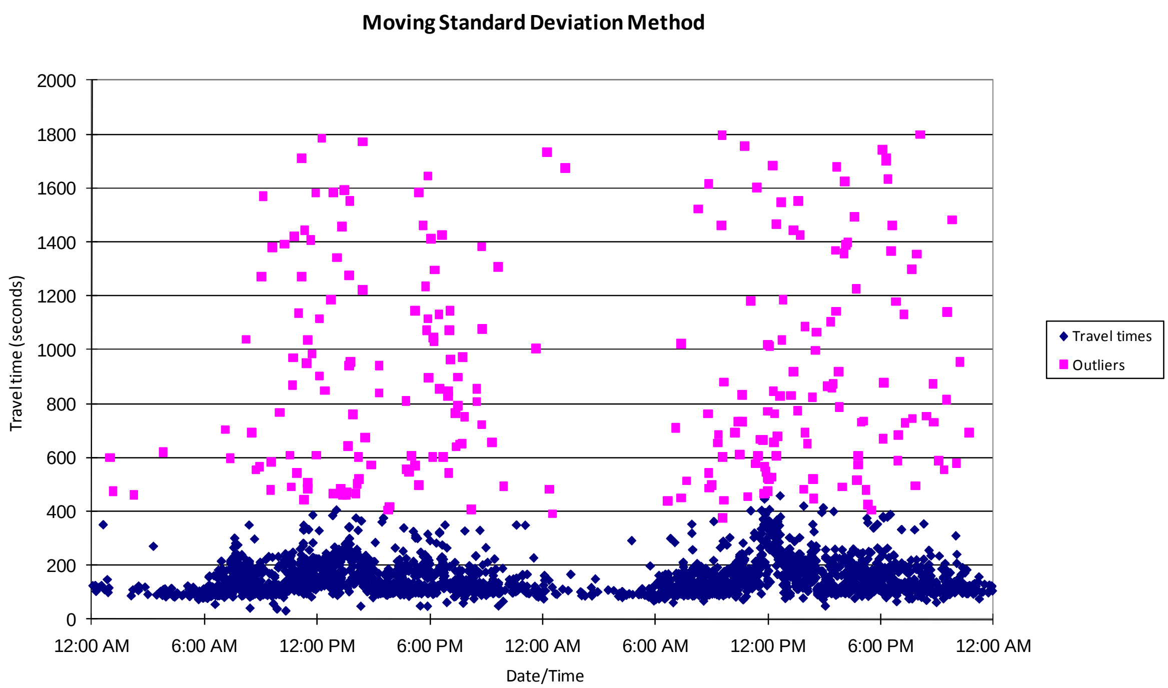

Figure 3.18: Travel time outliers identified using the moving standard deviation method

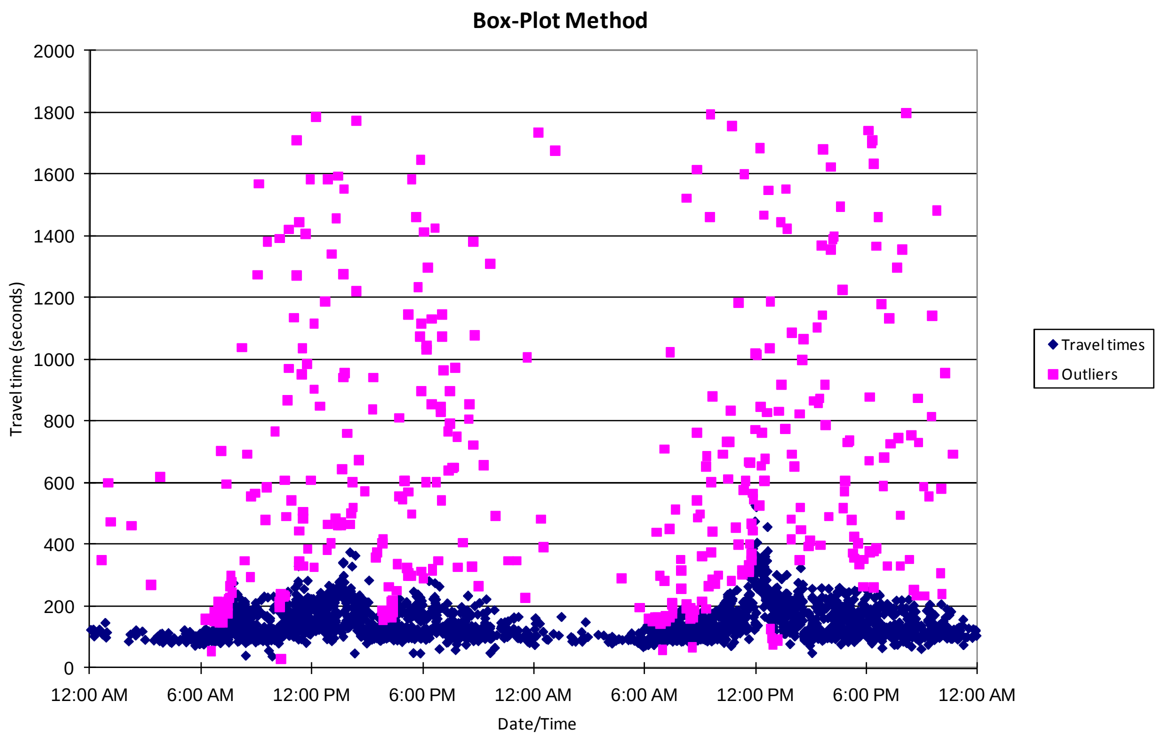

Figure 3.19: Travel time outliers identified using the box-plot method

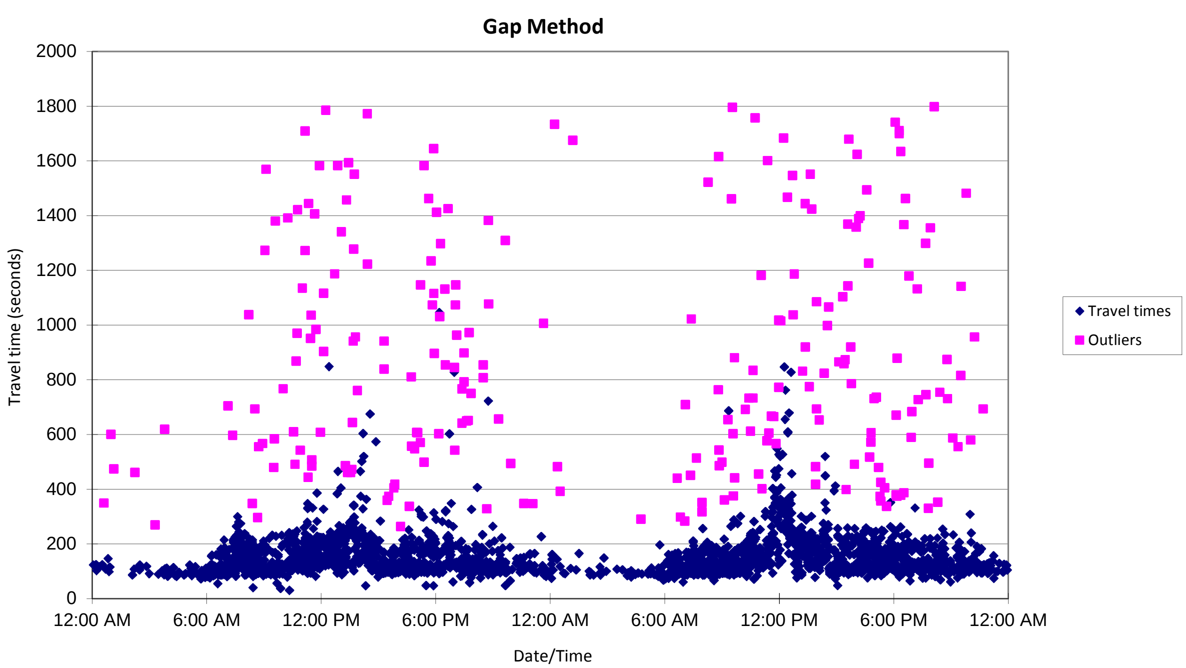

Figure 3.20: Travel time outliers identified using the gap method

The moving standard deviation method performs well with respect to identifying long travel times as outliers. Because the travel times are not normally distributed, this method does not filter out any short travel times since the mean minus 1.65 standard deviation units is negative or close to zero. The box-plot and gap methods both have similar undesirable features. The box-plot method is slow to adapt to increasing travel times so that longer travel times that occur at the start of a busier period are filtered. Both the box-plot method and the gap method adapt to longer outlier travel times so that if enough outliers occur in a period, new outliers are included.

3.3.3 Implemented Method

The outlier identification method that produces satisfactory results is a combination of the moving standard deviation method and a fixed lower bound on travel times. The moving standard deviation method utilizing the mean plus 1.65 times the sample standard deviation as an upper limit for travel times appears to do the best job of identifying long travel time outliers. Since the moving standard deviation method does not filter any short travel times, a fixed lower bound that is equal to a travel time at double the speed limit is used to filter short travel times. Figure 3.21 shows the number of travel time samples generated every 30 minutes (after filtering) over a two week period between DCU 12 and DCU 13. A moving window of 100 travel times allows the filtering method to adapt to the travel time behavior exhibited over 30-minute time intervals during peak traffic volumes.

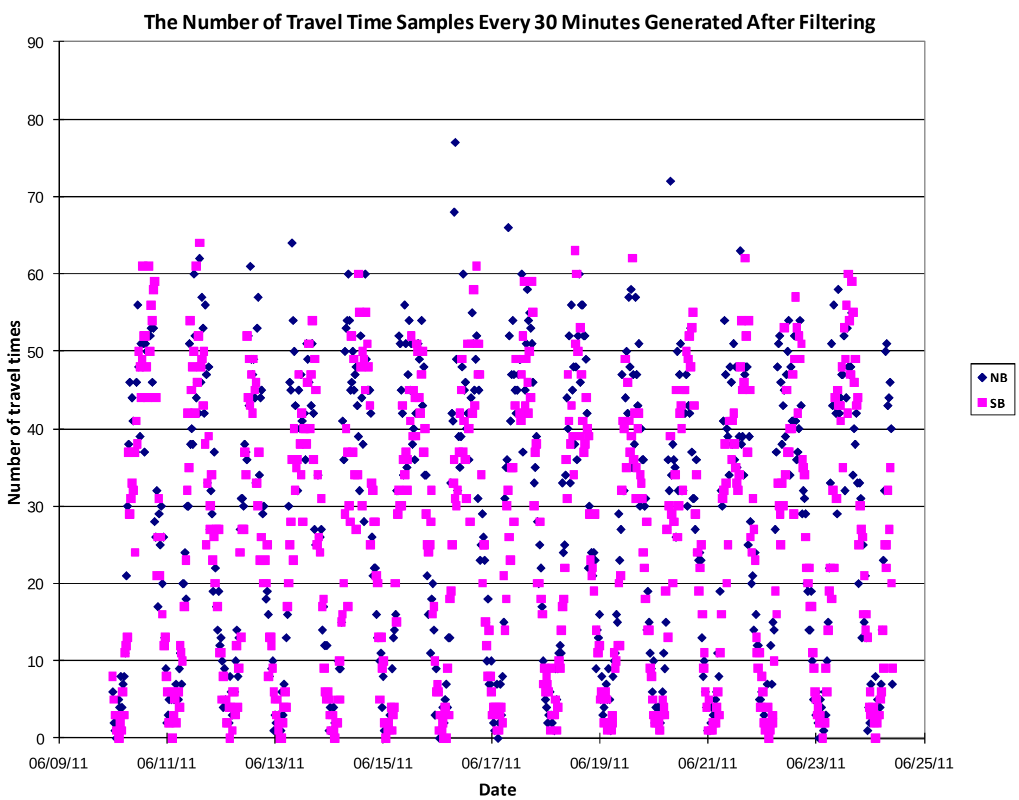

Figure 3.21: The number of travel time samples generated over time after filtering between DCU 12 and DCU 13

4.0 TRAVEL TIME STATISTICS

Assuming travel times samples have been generated and travel time outliers have been filtered travel time samples will be archived in a database (see Section 6.0). Various standard reports can be designed that utilize the archived data. For example, a report may be designed that generates basic statistics and graphs of mean travel times for specific time intervals and specific days of the week over some specified time period (e.g., 6-7AM on Fridays from August 1, 2011 through December 31, 2011). Other reports may focus on individual travel times over shorter periods of time. This latter type of report is more applicable to real time travel time updates that may be utilized to update variable message signs (Section 4.3).

4.1 STATISTICS FOR TRAVEL TIME DATA FROM SEPARATE TIME PERIODS

If statistics are to be computed for travel time data from separate time periods (e.g., the same hour over different days) the travel time results from separate time periods can be assumed to be independent. The assumption of independence does not affect sample average, maximum, and minimum statistics, but it does affect the variance estimate calculation. Assuming independent data, the sample variance is computed as:

$$s^2 = \frac{\sum_{i=1}^{n} x_i^2 - \frac{\left(\sum_{i=1}^{n} x_i\right)^2}{n}}{n-1} \tag{4-1}$$

The x_i are the data from each time period (e.g., the average travel time over time period i). n equals the total number of time periods. With sample averages and sample variances estimated from the travel time data, normal theory based confidence intervals may be computed. For example, if $\bar{x}$ is the sample average of n average travel times from 6-7AM on Fridays from August 1, 2011, through December 31, 2011, and s^2 is the sample variance of the 6-7AM average travel times, then a 95% confidence interval for the average Friday 6-7AM travel time is computed as:

$$\bar{x} \pm 1.96 * \frac{s}{\sqrt{n}} \tag{4-2}$$

4.2 SAMPLE VARIANCE AND CONFIDENCE INTERVALS FOR SHORT TIME PERIODS

When statistics are computed for individual travel time samples generated over a short time period such as 30 minutes to one hour, the calculation of sample variance is affected by the possible correlation of the travel time samples. Correlation is the ability to better predict a future travel time sample if the latest travel time sample is known. High positive correlation implies that if the most recent travel time is long, then it is likely that the next travel time is long. Since travel time samples occur over time, correlation may exist between two successive travel time samples (called lag 1 correlation), or in general between travel times that are j samples "apart" (called lag j correlation). In general, as j increases the correlation should approach zero.

Correlation affects the calculation of sample variance. Let $x_1, x_2, \cdots, x_n$ represent the individual travel time samples in the order that they were generated over a short period of time. Let $\bar{x}$ denote the sample average and s^2 denote the sample variance of these individual travel time samples. The estimator for lag j correlation ρ_j is:

$$\hat{\rho}_j = \frac{\sum_{i=1}^{n-j}[(x_i - \bar{x})(x_{i+j} - \bar{x})]}{s^2 * (n-j)} \tag{4-3}$$

Figure 4.1 is a plot of the lag 1 correlations for travel time samples generated in 30-minute intervals between DCU 12 and DCU 13 on Highway 99W from June 16-17, 2011.

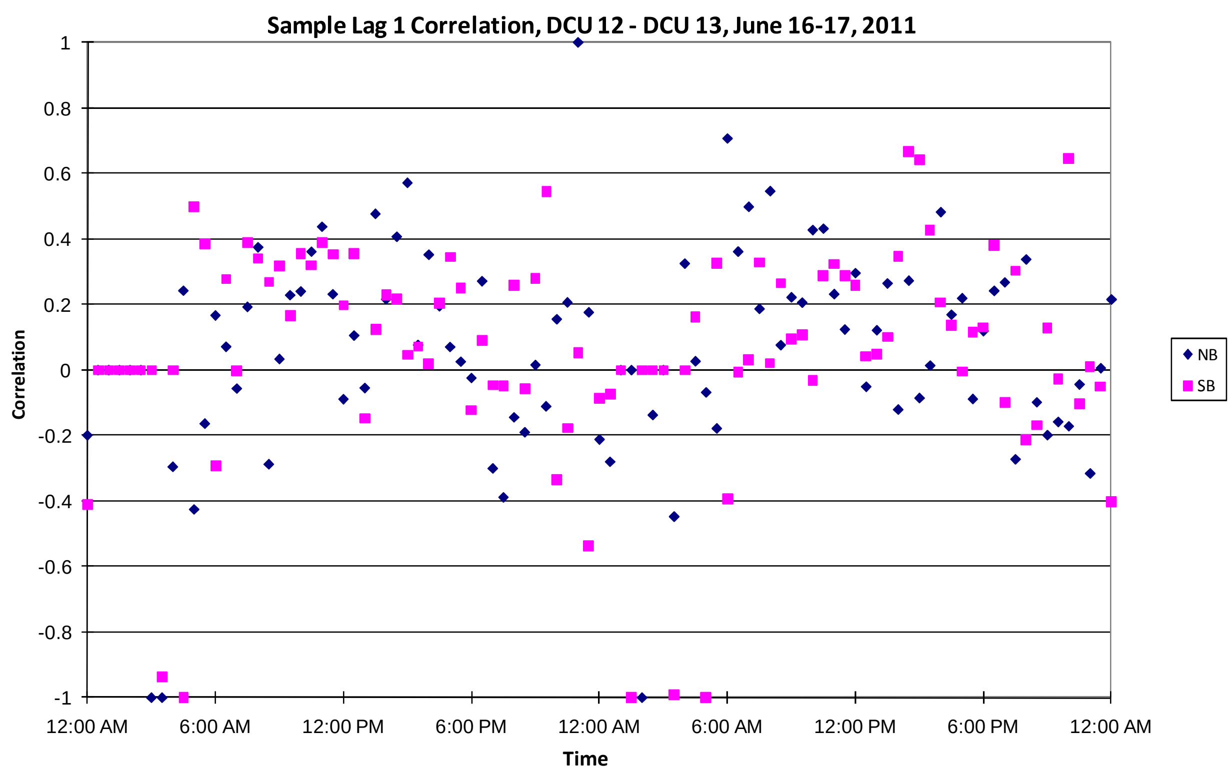

Figure 4.1: Lag 1 correlation estimates for travel time samples generated in 30-minute intervals on Highway 99W

The positive correlations in travel times will increase the estimate of travel time variance. The estimator for variance that accounts for correlation is:

$$\widehat{Var}(X) = s^2\left[1 + 2 * \sum_{k=1}^{n-1}\left(1 - \frac{k}{n}\right)\hat{\rho}_k\right] \tag{4-4}$$

Figure 4.2 shows the incorporation of the lag 1 correlation estimates into the standard deviation estimates (the square root of variance) for the same 30-minute interval of travel times as shown in Figure 4.1.

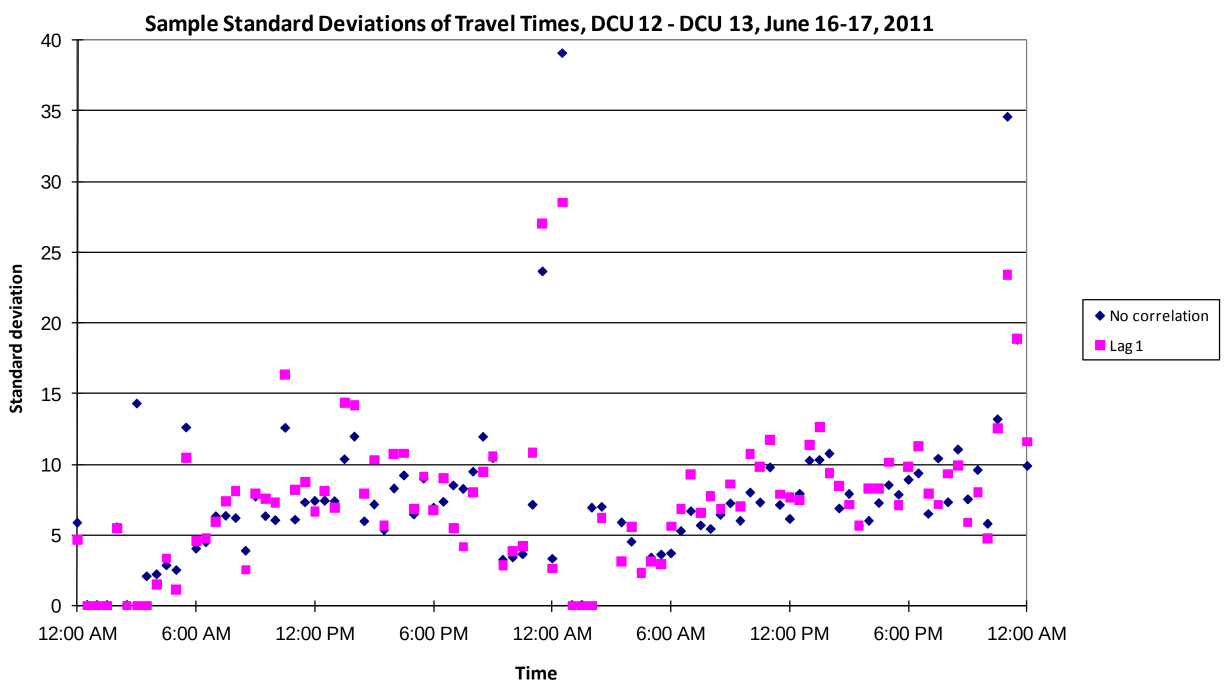

Figure 4.2: Travel time standard deviation estimates for travel time samples generated in 30-minute intervals on Highway 99W

As can be seen from Figure 4.2, most of the sample standard deviation estimates with the lag 1 correlation are greater than the standard deviation estimate assuming independent travel times. In practice, there must be a requirement for enough travel time samples (e.g., 20 or more) in a 30- minute interval for the inclusion of correlation in the standard deviation estimate. Including correlation estimates for lags greater than one will increase the standard deviation estimates (assuming positive correlation). However, since the correlation estimates themselves are highly variable, it is recommended that the inclusion of additional correlations with larger lags be limited. Figure 4.3 shows travel time standard deviation estimates with up to lag 3 correlations included.

An analysis of travel times collected in 30-minute intervals between June 17, 2011 and June 23, 2011 for three different road segments on Highway 99W show that if there are more than 10 travel time samples in the interval, then there is a positive estimated lag 1 correlation in over

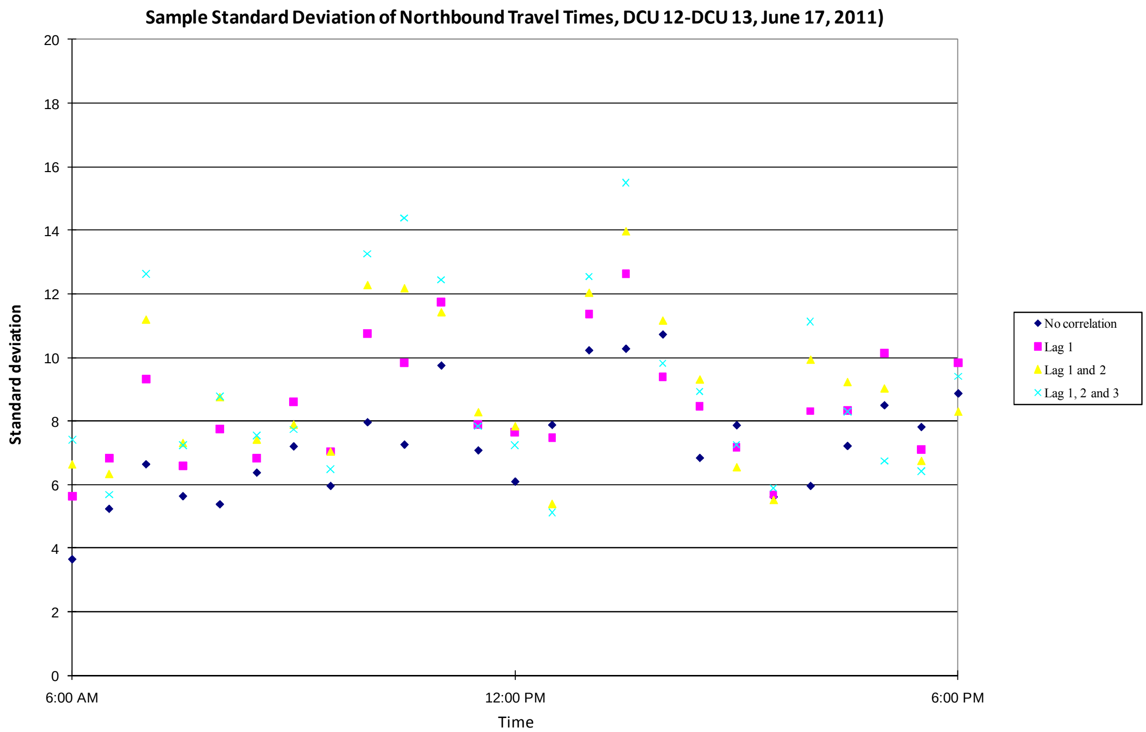

Figure 4.3: Travel time standard deviation estimates for travel time samples generated in 30-minute intervals on Highway 99W with up to lag 3 correlation

50% of these intervals. If there are over 30 travel time samples in an interval then there is positive estimated lag 1 and lag 2 correlations in over 50% of these intervals. If there are over 50 travel time samples in an interval, then there is positive estimated lag 1, lag 2, and lag 3 correlations in over 50% of these intervals. These values (10, 30, and 50) for the number of travel time samples in a 30-minute interval can be used as threshold values for the inclusion of estimated lag 1, lag 2, and lag 3 correlation into the estimated travel time standard deviation.

Once the travel time standard deviation estimate has been computed, the confidence interval for the average travel time within the time interval can be computed using the same formula presented in Section 4.1. Here, *s'* is the standard deviation estimate incorporating travel time correlation and the 95% confidence interval for the mean travel time is:

$$\bar{x} \pm 1.96 * \frac{s'}{\sqrt{n}} \tag{4-5}$$

The 95% confidence interval for individual travel times is computed as:

$$\bar{x} \pm 1.96 * s' \quad \textbf{(4-6)}$$

The lower confidence limit will be the maximum of zero and the lower value computed using the prior formulas.

4.3 SHORT TERM TRAVEL TIME FORECASTING

As travel time samples are obtained, statistics as described in Section 4.2 can be computed over very short time intervals (e.g., five-minute intervals) assuming a sufficient number of travel time samples were obtained. These statistics can then be used as the basis for short term travel time predictions which can be displayed on a variable message sign. The intent is to display the best prediction of travel time that drivers seeing the variable message sign will experience.

Three short term forecasting methods were examined using historical travel time data obtained on Highway 99W. The first method is the application of Kalman filtering. The second and third methods are simple moving interval based methods (non-overlapping and overlapping intervals). The basic principle behind these methods is that the best predictor for travel times in the next five minutes is the average travel time over the most recent time interval, assuming a sufficient number of travel time samples were obtained.

4.3.1 Kalman Filter

A brief description of the Kalman filter is that it is a set of mathematical equations to estimate the state of a process (e.g., travel time realization) that minimizes the squared errors of these estimates. The Kalman filter estimates are based on a state equation and measurement equation, which are used to generate an *a priori* estimate of the state (a prediction), and then an *a posteriori* estimate of the state after a measurement is obtained. Its appeal is that it provides an efficient recursive solution procedure and does not require that a process be stationary. The literature review cites multiple applications of the Kalman filter for travel condition predictions.

The Kalman filter applied for short term travel time forecasting is the same as utilized in Barceló et al. (*2010*). This application simplifies the general Kalman filter presentation since the system state (the average travel time over a five-minute interval) is a scalar, and it is assumed that the noise in the "process equation" is zero. Additionally, the process equation is a simple function of the average travel times computed from travel time samples in prior five minute time intervals. The specifics of the Kalman filter tested are described below:

- Notation

  x_k : Average travel time for a road segment over 5-minute time interval *k*.

  $\hat{x}_k^-$: A priori estimate of x_k.

  $\hat{x}_k$: A posteriori estimate of x_k.

  y_k : Sample average travel time for a road segment over 5-minute time interval *k* computed from collected travel time samples.

R_k : Travel time variance in time interval k.

F_k : State model for time interval k.

P_k : A posteriori estimate error variance for time interval k.

P_k^- : A priori estimate error variance for time interval k.

K : Gain or blending factor.

- Model

  State equation: $x_k = F_k x_{k-1}$.

  Measurement equation: $y_k = x_k + v_k, v_k \sim N(0, R_k)$.

- Calculations from collected travel times in interval k.

  y_k = Sample average travel time for a road segment over 5-minute time interval k.

  R_k = Sample variance of collected travel times in time interval k. $R_k = R_{k-1}$, if number of travel times is less than 2.

- State Model Equations

  $F_{k+1} = \dfrac{y_k}{y_{k-1}}$ if $y_k \neq 0$ and $y_{k-1} \neq 0$, $F_{k+1} = 1$ if $y_k = 0$, $F_{k+1} = \dfrac{y_k}{\hat{x}_{k-1}}$ otherwise.

- Initial Conditions

  $F_1 = 1$.

  P_0 = Sample variance of the travel times for the latest 5-minute time interval.

  $\hat{x}_0$ = Sample average of the travel times for the latest 5-minute time interval.

- A priori estimates (forecasts)

  $\hat{x}_k^- = F_k \hat{x}_{k-1}$

  $P_k^- = F_k P_{k-1} F_k^T = (F_k)^2 \cdot P_{k-1}$

- A posteriori estimates

  $K = P_k^- (P_k^- + R_k)^{-1} = \dfrac{P_k^-}{P_k^- + R_k}$

  $\hat{x}_k = \hat{x}_k^- + K(y_k - \hat{x}_k^-)$ if $y_k \neq 0$, $\quad \hat{x}_k = \hat{x}_k^-$ otherwise

  $P_k = (1 - K)P_k^-$ if $y_k \neq 0$, $\quad P_k = P_k^-$ otherwise.

4.3.2 Moving Window Average

The second and third forecasting methods tested are simple moving window averages. In the second method the forecasted travel time is updated every T minutes (non-overlapping T minute intervals), and this forecasted travel time is the sample average of the travel times obtained in the latest T minute time interval. If there are fewer than n travel time samples collected in the latest T minute time interval, then the forecast is a weighted average of the travel time samples in the latest T minute time interval, and the prior forecast. The parameters of this method are the time interval length T, and the required number of travel time samples in an interval n. Various parameter values were tested and results for $T = 1$ and $T = 5$ minutes with $n = 10$ are reported.

The third method generates a forecast as the average travel time from the last T minute time interval, and then updates the average every t minutes. With $t < T$, the time intervals used to generate the forecasts will overlap. Various parameter values were tested and results for $T = 5$ minutes and $t = 1$ minute are reported.

4.3.3 Accuracy Testing

The forecasting accuracy testing assessed the accuracy of forecasts displayed on a hypothetical variable message sign (VMS) located at Highway 99W and Durham and seen by drivers traveling north. The forecasts seen were compared to the actual travel times realized by the vehicles. Travel times were computed from MAC address data collected on Highway 99W from June 17, 2011 to June 23, 2011. Starting with the DCU located at Highway 99W and Durham and traveling north, travel times can be computed for four road segments defined by adjacent DCU pairs. The accuracy of travel time forecasts for the first road segment (Highway 99W from Durham to McDonald), and the total of all four road segments (Highway 99W from Durham to SW 64th Avenue) were evaluated. The forecasting methods tested were:

- Non-overlapping moving window average with $T = 1$ and 5 minutes, $n = 10$.
- Overlapping moving window average with $T = 5$ minutes, $t = 1$ minute.
- Kalman filter with 1- and 5-minute time intervals.

Figure 4.4 through Figure 4.7 show a sample of plots of the forecasted travel times using some of the forecasting methods tested. These figures also show the actual average travel times for those vehicles that would have observed the forecast. The performance of the methods is very similar. Figure 4.8 shows the same forecasting methods over a shorter time period as well as individual travel times. The individual travel time variability is very high and in this environment the forecasting methods perform similarly. Tables 4.1 and 4.2 show a numerical comparison of the forecasting methods tested. The percent differences are computed as (average travel time -forecasted time)/average travel time. A negative value implies that the forecast was greater than the actual average travel times. The forecasted methods were tested for data collected over six days (June 18-23, 2011) during time periods when traffic volumes were the highest.

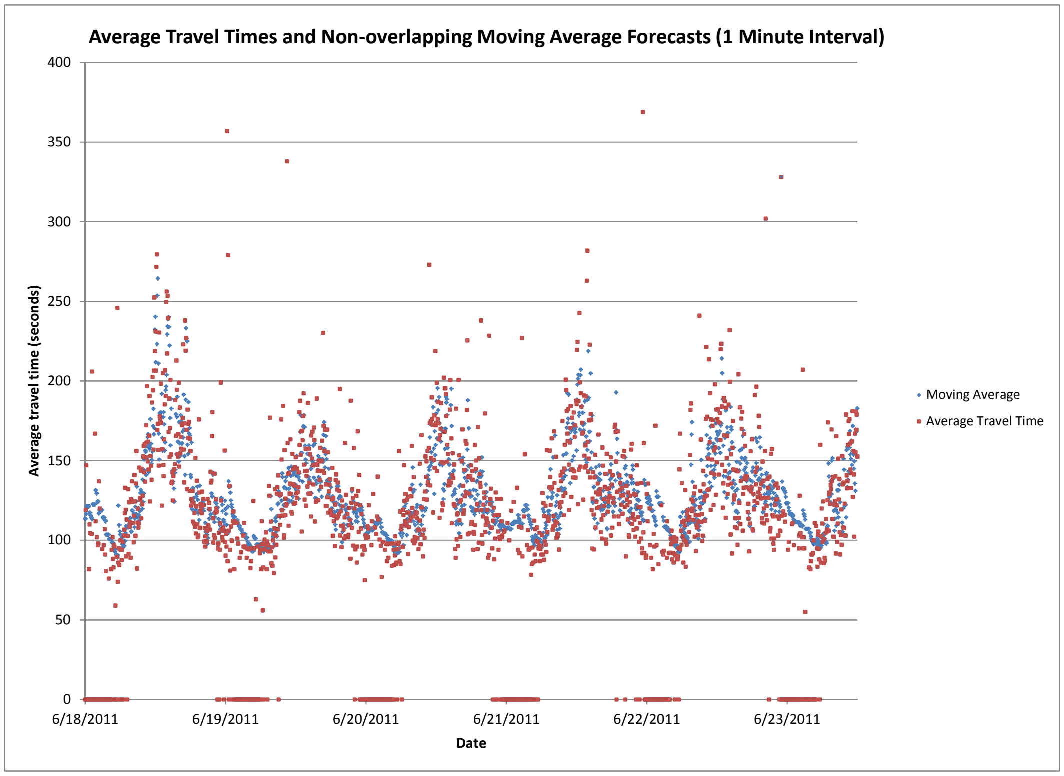

Figure 4.4: Moving window forecasted travel times and actual average travel times for one road segment using a one-minute interval

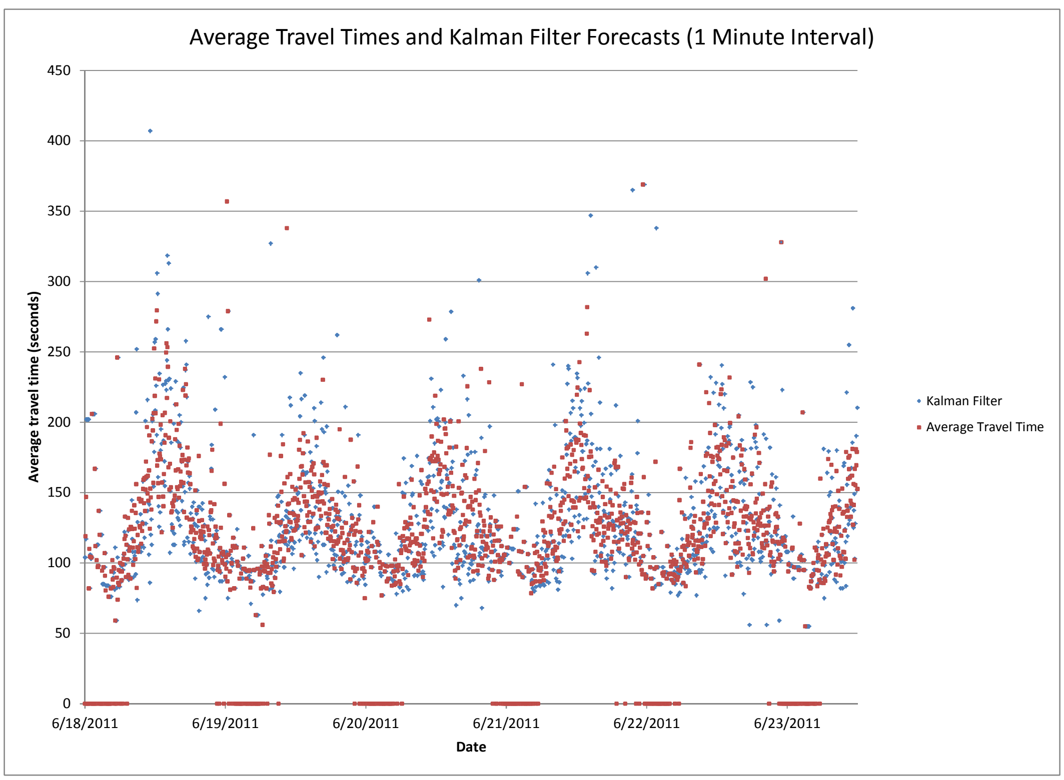

Figure 4.5: Kalman filter forecasted travel times and actual average travel times for one road segment using a one- minute interval

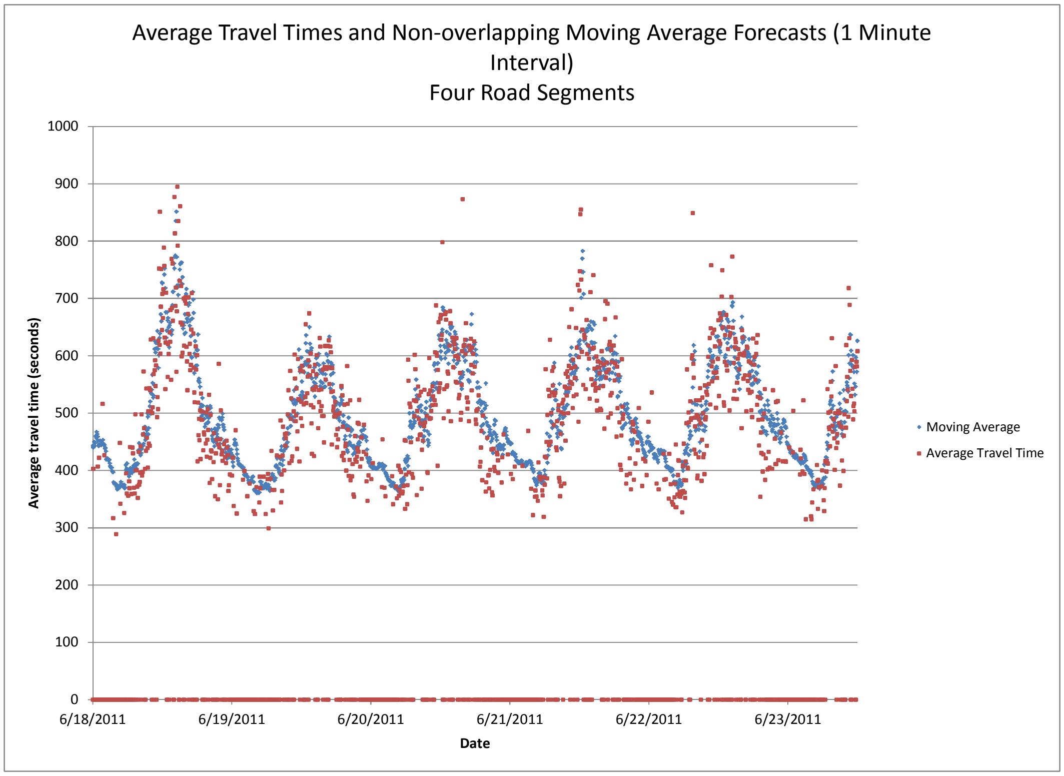

Figure 4.6: Moving window forecasted travel times and actual average travel times for four road segments using a five-minute interval

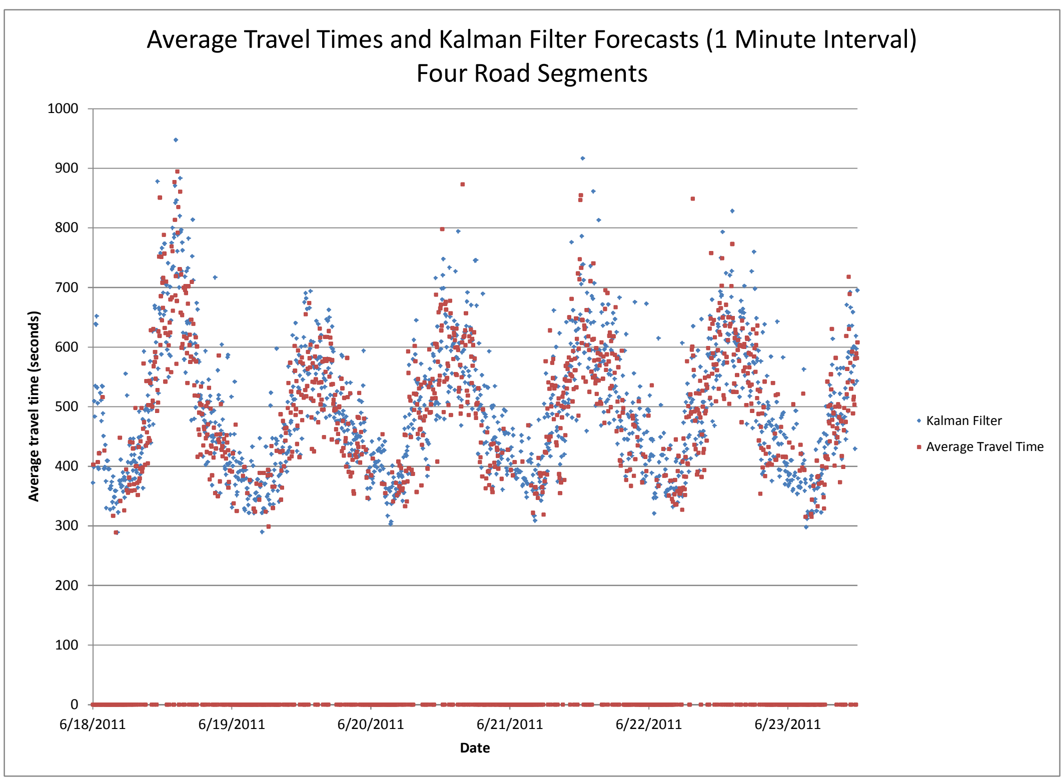

Figure 4.7: Kalman filter forecasted travel times and actual average travel times for four road segments using a five- minute interval

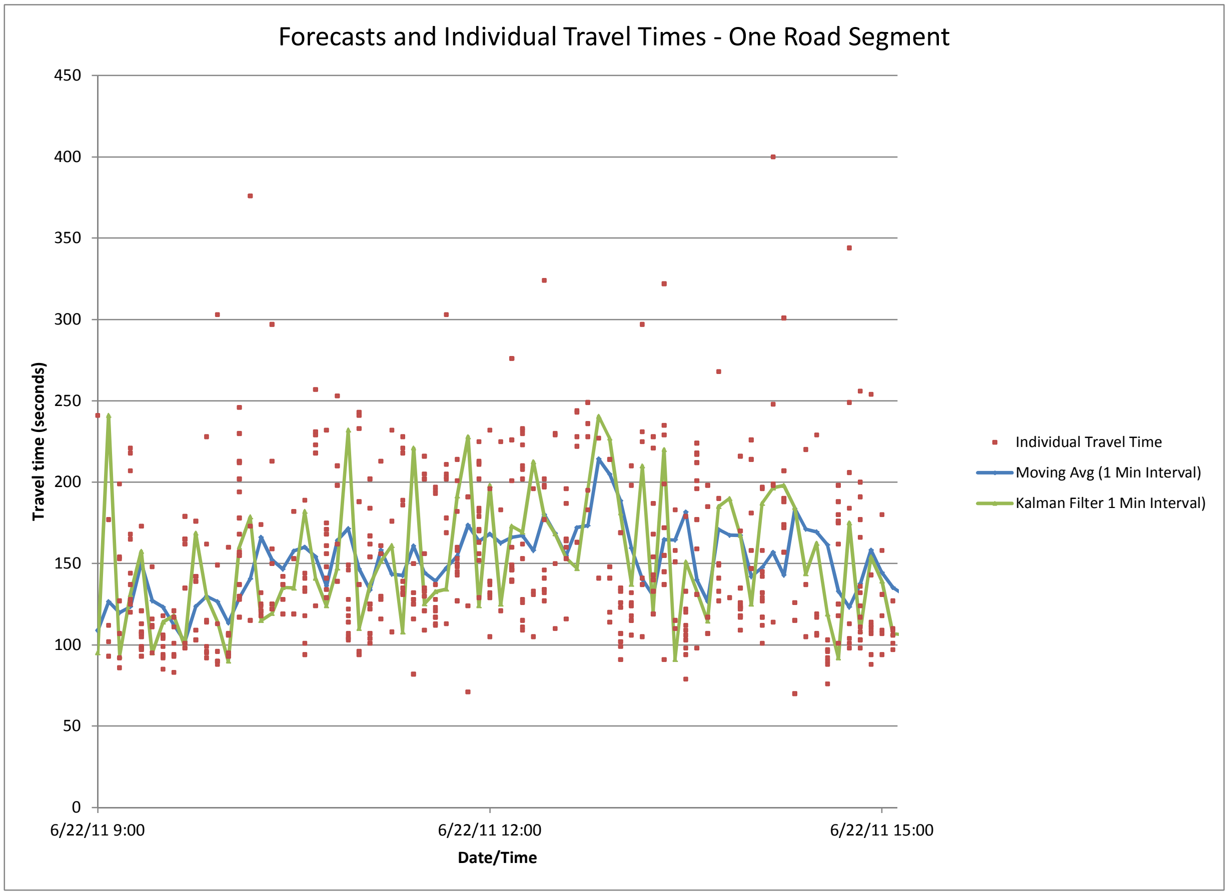

Figure 4.8: Kalman filter and moving window forecasted travel times and individual travel times for one road segment

Table 4.1: Differences between forecasts and average travel times for different forecasting methods over a single road segment (June 18-23, 2011, 6AM-9AM, 11AM-1PM, 4PM-6PM)

Method	Overlapping Moving Avg. 5 min	Kalman filter 1 min	Kalman filter 5 min	Non Overlapping Moving Avg. 1 min	Non Overlapping Moving Avg. 5 min
Avg.	-1.81%	-1.99%	-1.58%	-1.15%	-1.59%
Std. Dev.	21.75%	30.20%	21.70%	17.48%	18.34%
Median	0.83%	2.22%	0.87%	-0.18%	-0.25%
Min	-111.24%	-206.38%	-110.49%	-81.04%	-85.67%
Max	45.30%	68.58%	45.24%	40.69%	41.70%

Table 4.2: Differences between forecasts and average travel times for different forecasting methods over four road segments (June 18-23, 2011, 6AM-9AM, 11AM-1PM, 4PM-6PM)

Method	Overlapping Moving Avg. 5 min	Kalman filter 1 min	Kalman filter 5 min	Non Overlapping Moving Avg. 1 min	Non Overlapping Moving Avg. 5 min
Avg.	-0.83%	-0.20%	-0.30%	0.18%	-0.17%
Std. Dev.	15.31%	17.38%	15.25%	14.10%	14.36%
Median	-0.22%	1.07%	0.36%	0.90%	0.47%
Min	-71.37%	-66.30%	-71.36%	-48.75%	-53.72%
Max	43.23%	43.37%	43.54%	41.20%	40.46%

4.3.4 Recommended Forecasting Method

The forecasting method recommended for implementation is the non-overlapping moving average method using one-minute time intervals. The performance of this method and the five-minute non-overlapping moving average method are very similar and the method is simple to implement. While the performance of the five-minute non-overlapping moving average is similar to the one-minute non-overlapping method, it results in a smoothing of the forecasts due to the use of a larger time interval for the travel time average. It is therefore less responsive to changes in travel time that may occur. This can be seen in Figure 4.9.

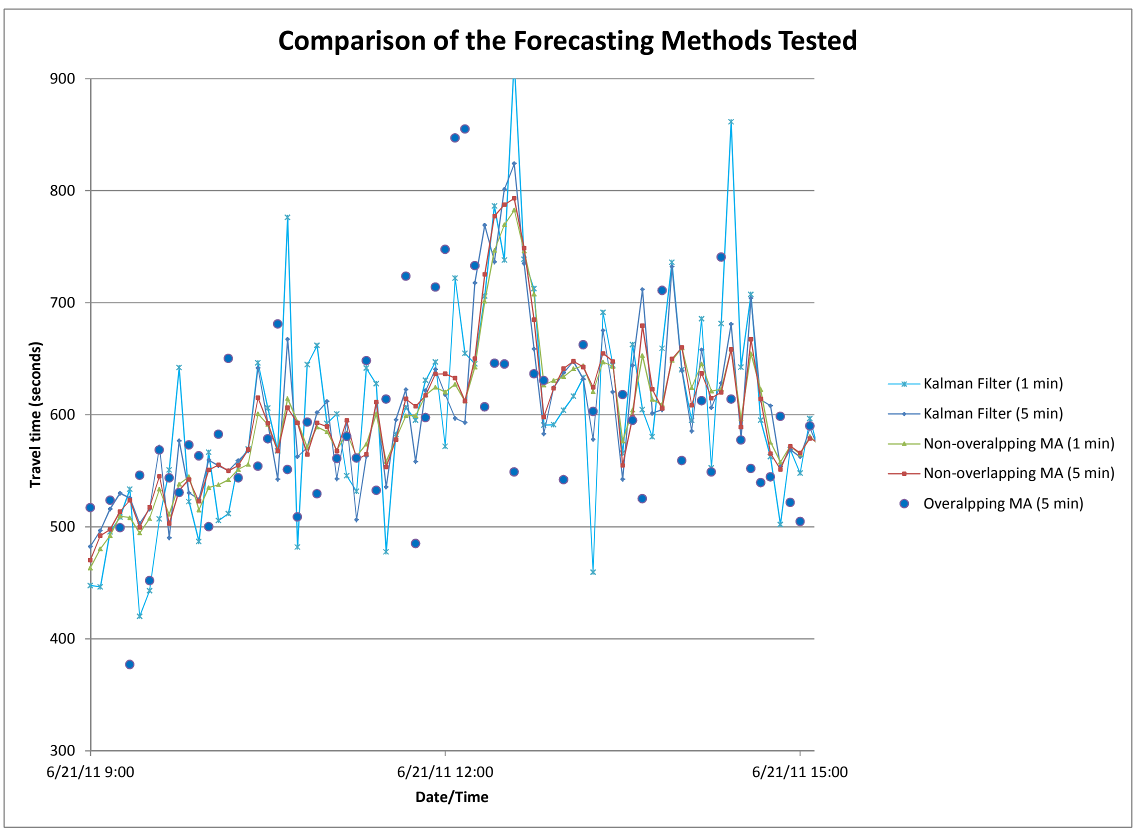

Figure 4.9: Kalman filter and moving window forecasted travel times for four road segments

4.3.5 Forecasting Under Heavy Congestion

Forecasting errors increase as a function of the travel time rate of change through a road segment. Forecasting errors are defined as the difference between the predicted travel time when a vehicle just starts travel on a road segment, and the travel time realized by this vehicle when it just finishes traveling the road segment. If the average travel times through a road segment do not change over time, then forecast accuracy will be good under both congested and uncongested situations.

However, if there is large change in the average travel times over a relatively short period of time (as can be caused by an incident) large forecasting errors will results. To understand the forecasting error magnitude caused by such travel time changes a deterministic simulation was conducted. In this simulation a one-mile road segment with a 45MPH speed limit was considered. Under non-congested conditions, the travel time for all vehicles traveling the road was assumed to be 1.33 minutes (based on the 45MPH speed limit). Two travel time rates of change scenarios were examined. In the first scenario vehicles enter the road segment every 30 seconds. The travel time for each vehicle was increased steadily over time with a maximum travel time of 20 minutes that occurs 15 minutes after the start of the simulation. After 15 minutes, travel times for vehicles entering the road segment steadily decrease from the maximum until they reach the uncongested travel time of 1.33 minutes, 30 minutes after the start of the simulation. In the second scenario the congestion period is 60 minutes, with the maximum travel time occurring for vehicles that enter the road segment 30 minutes after the start of the simulation.

Forecasting error percentages = (forecast - actual travel time)/(actual travel time) were computed for the five-minute non-overlapping interval forecasting method and are shown in Tables 4.3 and 4.4. The forecasts remain the same over a five-minute interval, so every vehicle entering the road segment during that five-minute period sees the "displayed" forecast.

Table 4.3: Forecast errors under a steady increase and decrease in travel times over a 30-minute interval

Five Minute Interval	Actual Avg. Travel Time (minutes)	Forecast (minutes)	Avg. Error (minutes)	% Error
[0,5)	4.13	1.33	-2.80	-68%
[5,10)	10.36	1.71	-8.65	-84%
[10,15)	16.58	2.93	-13.65	-82%
[15,20)	17.20	5.24	-11.96	-70%
[20,25)	10.98	7.29	-3.69	-34%
[25,30)	4.76	10.22	5.47	115%
[30,35)	1.33	12.51	11.18	839%

Table 4.4: Forecast errors under a steady increase and decrease in travel times over a 60-minute interval

Five Minute Interval	Actual Avg. Travel Time (minutes)	Forecast (minutes)	Avg. Error (minutes)	% Error
[0,5)	2.73	1.33	-1.40	-51%
[5,10)	5.84	1.64	-4.20	-72%
[10,15)	8.96	2.86	-6.10	-68%
[15,20)	12.07	4.46	-7.60	-63%
[20,25)	15.18	6.63	-8.55	-56%
[25,30)	18.29	8.40	-9.89	-54%
[30,35)	18.60	10.23	-8.37	-45%
[35,40)	15.49	12.08	-3.41	-22%
[40,45)	12.38	13.94	1.56	13%
[45,50)	9.27	15.80	6.53	71%
[50,55)	6.16	17.67	11.51	187%
[55,60)	3.04	15.96	12.91	424%
[60,65)	1.33	7.71	6.38	478%

As expected, the forecast errors are larger for the 30-minute congestion period since the travel time rate of change is greater. The largest forecast errors are from overestimated travel times caused by forecasts computed from large travel times that are displayed when the congestion is decreasing.

Given the potential for such large forecast errors a possible strategy for VMS displays is to suspend the display of forecasts as soon as a significant incident is detected on a road segment. Information on the severity and time of the incident can be displayed instead.

5.0 INTERSECTION PERFORMANCE ANALYSIS

The time stamped MAC address data collected at adjacent intersections has been utilized to generate travel time samples for vehicles traveling between two consecutive intersections. In this section, the use of MAC address data collected at a single DCU to obtain measures of intersection performance is examined. Because data from only a single DCU is utilized it is not possible to accurately estimate standard intersection performance measures such as control delay and stop delay. Instead, the objective focused on utilizing the MAC address data collected at a single DCU to compute measures that are correlated to the time a vehicle requires to pass through an intersection.

5.1 INTERSECTION DATA FILTERING

Because each individual DCU installed on Highway 99W is located at an intersection, the recorded MAC addresses represent vehicles traveling on both intersecting roads. Because of this, no discernible patterns could be recognized from the MAC address data captured. To make this task feasible, adjacent DCUs south and north of the DCU located at the intersection of Highway 99W and McDonald Avenue were utilized to identify MAC addresses in vehicles traveling north. These MAC addresses were utilized to obtain intersection performance measures for northbound vehicles.

5.2 INDIVIDUAL BLUETOOTH DEVICE DATA

Individual Bluetooth devices in vehicles are typically read multiple times as the vehicle travels through the coverage area of the DCU’s antenna. The software implemented on the DCUs will have a minimum time of consecutive MAC address reads of 3.84 seconds, which is the length of the repeated Bluetooth inquiry period. For vehicles that have stopped at the intersection, the number of times a MAC address is read and recorded will generally increase compared to when the vehicle is not forced to stop, since the vehicle is in the DCU antenna coverage area for a longer period of time.

Utilizing the methods described in Section 3.1.3, an estimate of the time a vehicle (with a Bluetooth device) just passes the signal can be obtained (since the DCU is located close to the signal). If the first time the MAC address is read is recorded, the difference in time between these two MAC address records should be correlated with the time a vehicle spends in the intersection. Here the intersection is defined to start approximately 300 feet before the signal, and ends just past the signal. This basic concept is used to obtain intersection performance measures from the MAC address data. However, the procedure for computing performance measures is complicated by the fact that Bluetooth devices differ with respect to the distance from the DCU at which they can be detected.

5.3 DATA FROM MULTIPLE BLUETOOTH DEVICES

Because different Bluetooth devices are different with respect to when they are first read by a DCU as a vehicle approaches the intersection signal, the time elapsed between when a vehicle is first detected and when it just passes the signal cannot be used directly for all vehicles as a measure of the time a vehicle stays in the intersection. However, the time when a Bluetooth device is first detectable by the DCU is related to the strength of the radio frequency signal emitted by the Bluetooth device as measured by the highest received signal strength indicator (RSSI) value. RSSI values are measured in units of decibels (dB) and each DCU records the RSSI associated with each MAC address read from Bluetooth devices as they pass by a DCU. From these RSSI values the highest RSSI values can be examined in more detail.

Tables 5.1 and 5.2 show the number of vehicles in each of five ranges of highest RSSI values recorded by the DCU located at Highway 99W and McDonald (DCU 12) during two different time periods.

Table 5.1: Number of vehicles by highest RSSI at DCU 12 between 9/20/2011 and 9/28/2011

	Highest RSSI Range (in dB)				
	[-40,-50)	[-50,-60)	[-60,-70)	[-70,-80)	[-80,-∞)
Number of vehicles	366	2,700	4,830	2,096	720

Table 5.2 Number of vehicles by highest RSSI at DCU 12 between 11/7/2011 and 11/15/2011

	Highest RSSI Range (in dB)				
	[-40,-50)	[-50,-60)	[-60,-70)	[-70,-80)	[-80,-∞)
Number of vehicles	375	2,423	4,299	1,827	667

Duration will be defined as the time between the first MAC address record and the MAC address record representing when the vehicle passes the DCU. Duration can then be examined for the MAC address records within a specific highest RSSI range. For each highest RSSI range the fraction of durations of different lengths can be examined. A graph of the cumulative fraction of duration records versus duration can be constructed. Figure 5.1 shows such a graph for MAC addresses collected at DCU 12 for vehicles traveling north. For a specific highest RSSI range, the plot in Figure 5.1 shows the fraction of durations that are less than or equal to a duration range. The empirical relationships shown in Figure 5.1 form the basis for equating durations that are associated with different highest RSSI values.

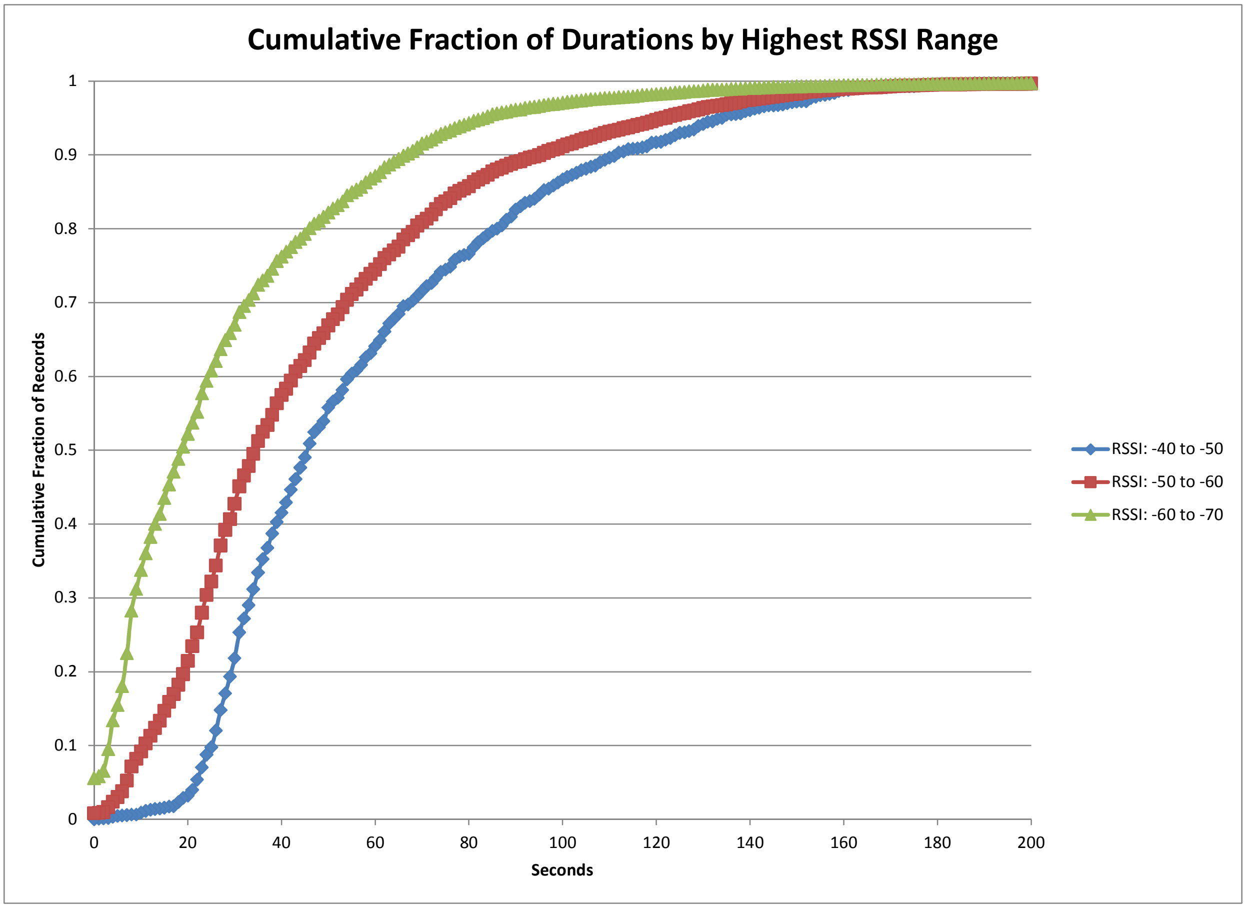

Figure 5.1: Cumulative fraction of durations versus duration for northbound MAC addresses recorded at DCU 12 between 12/6/11 and 12/19/11

Figure 5.1 does not include plots for highest RSSI values less than -70dB. Groups of MAC address records with such low RSSI values were almost always one or two records and thus contain minimal information related to duration. Such MAC address records were not utilized further to obtain intersection performance measures.

5.4 APPROACH FOR MEASURING INTERSECTION PERFORMANCE

The approach developed to obtain an intersection performance measure is to “normalize” observed durations to a specific RSSI range and then use the average of the normalized durations as the intersection performance measure. The highest RSSI range used as the “standard” range was arbitrarily selected as the -40 to -50dB range, but the specific RSSI range utilized should be calibrated to a specific intersection. Figure 5.2 shows conceptually how a19-second duration associated with a highest RSSI value in the -60 to -70dB range is normalized to a value close to 45 seconds in the -40 to -50dB highest RSSI range. Figure 5.3 shows plots of individual durations and normalized durations (normalized to the highest RSSI range) over a twelve-hour period.

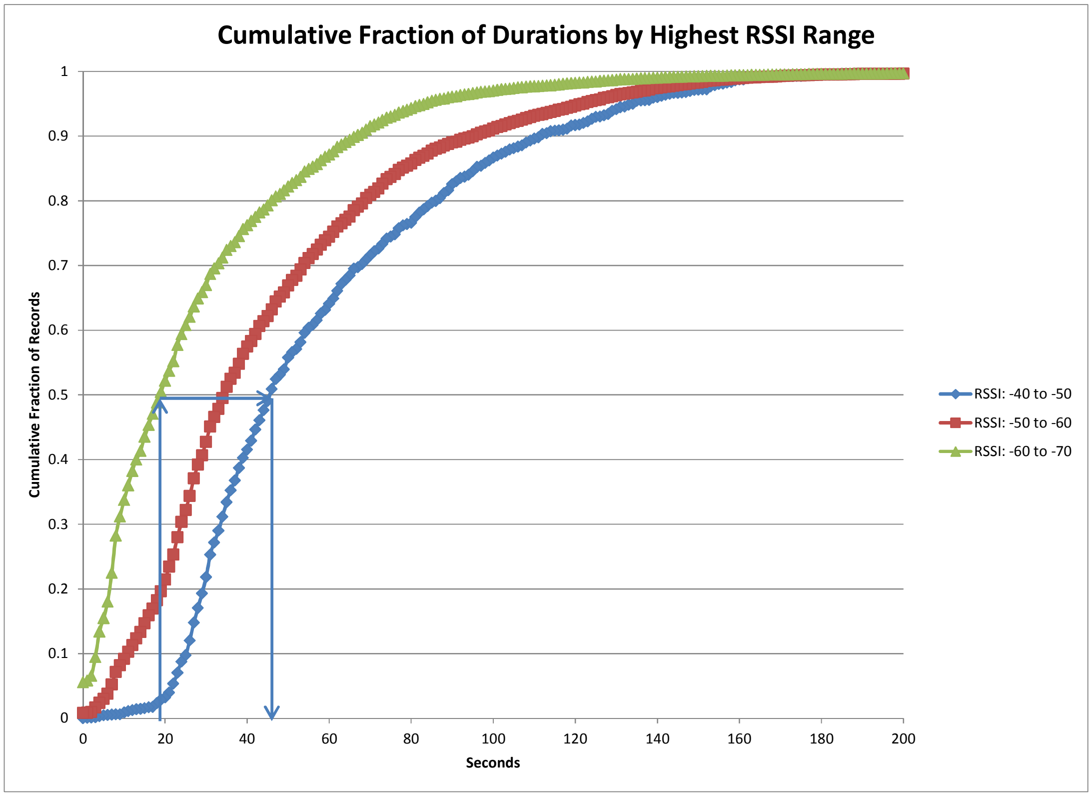

Figure 5.2: Example of the normalization of a duration to a specific RSSI range

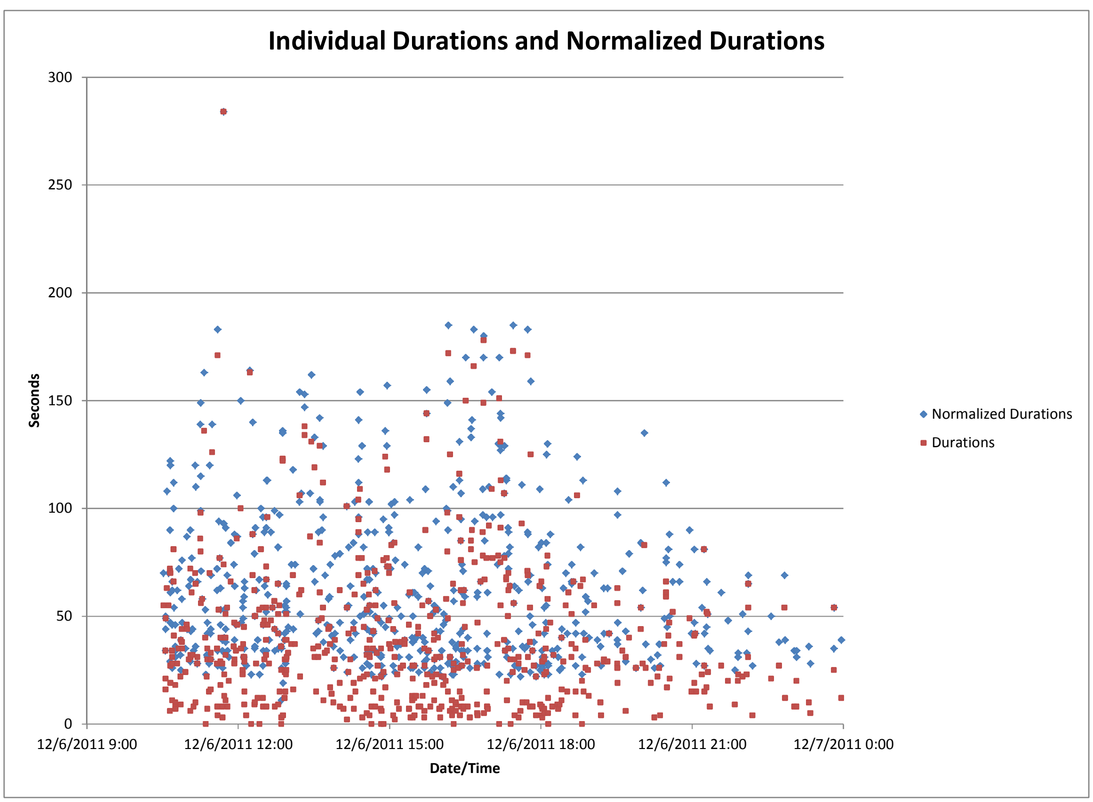

Figure 5.3: Plots of individual durations and normalized durations

Figure 5.4 shows the average durations for non-overlapping 30-minute intervals over time for the same data used in Figure 5.3.

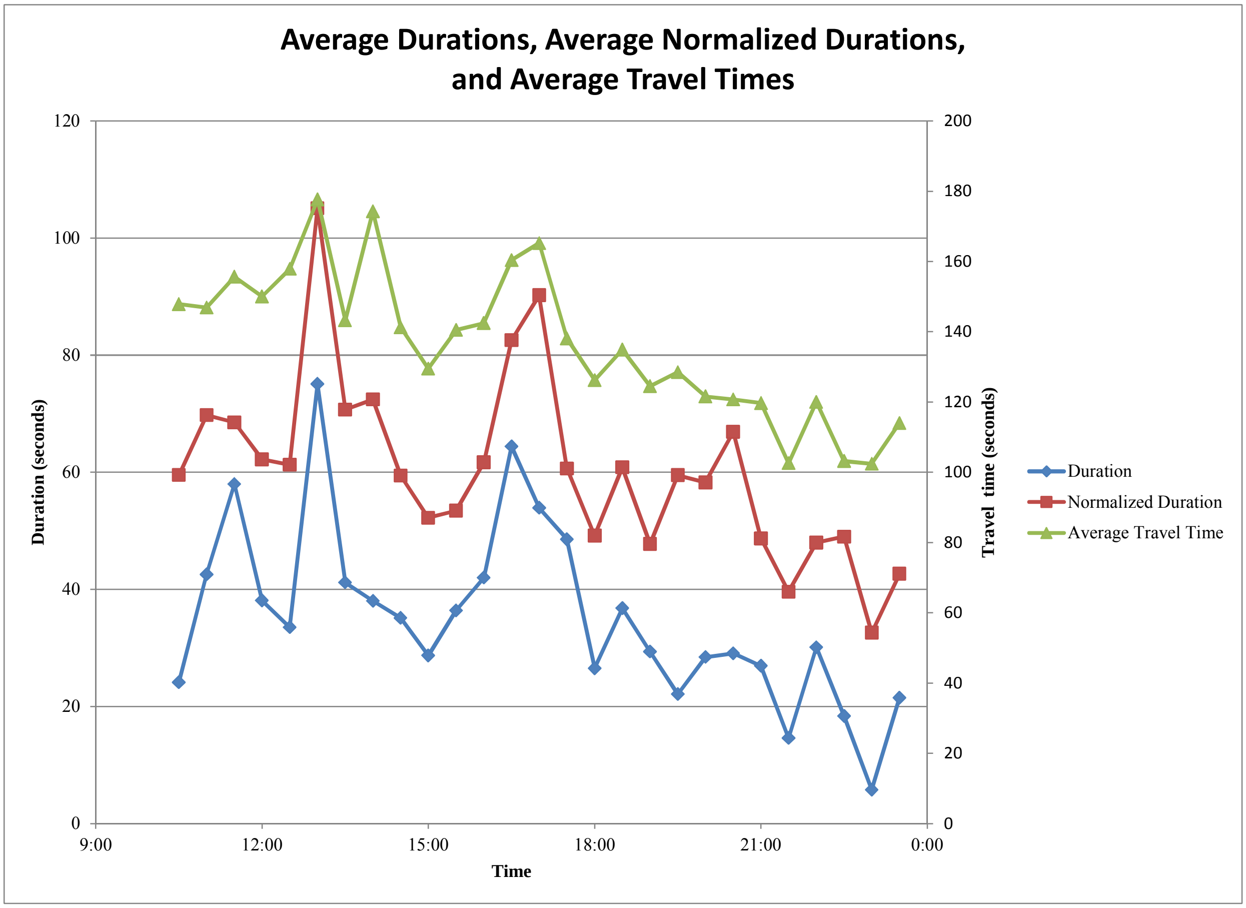

Figure 5.4: Plots of average durations, average normalized durations, and average travel times

Figure 5.4 demonstrates that there is a relatively large difference between the normalized and un-normalized average durations. The specific RSSI range utilized as the standard for normalization can be calibrated to samples of manually collected data. Figure 5.4 also shows that the average durations correspond to the average travel times as traffic volumes vary over time.

If calibration is not feasible, it also possible to compute an intersection performance measure that is a ratio between average durations during peak traffic periods to average durations at known low traffic periods.

6.0 IMPLEMENTATION PROCEDURES FOR DATA COLLECTION, DATA FILTERING AND DATA ANALYSIS

In this section, the implementation procedures required in the data collection unit (DCU) and the central server for data collection, data filtering and data analysis are presented in a format that facilitates their translation into computer code.

6.1 HIGH LEVEL SYSTEM DESCRIPTION

Figure 6.1 depicts the main components of the Bluetooth-based travel time data collection system which includes several DCUs that can be accessed by a central server via ODOT's network.

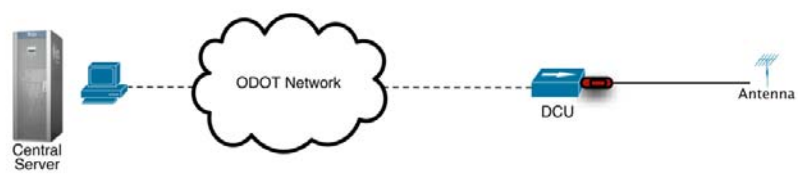

Figure 6.1: Bluetooth-based data collection system

Figure 6.2 illustrates the necessary steps to collect and process time-stamped MAC addresses. First, DCUs detect MAC addresses from Bluetooth-enabled devices in vehicles. Second, a time stamp is added to each MAC address to identify their actual detection time. Third, the time-stamped MAC addresses are inspected to identify and eliminate (i.e., filter out) outliers (i.e., non-moving vehicle or non-vehicle time-stamped MAC addresses). Fourth, only those time-stamped MAC addresses assumed to have been collected from moving vehicles are imported to the server based on a pre-defined schedule. Fifth, immediately after time-stamped MAC addresses are imported, travel time samples are calculated for road segments contained within two consecutive DCUs, as depicted in Figure 6.3. Sixth, each individual travel time sample calculated is checked to determine whether or not it is an outlier. Travel time samples that are not outliers are stored in a database; outlier travel time samples are filtered out. At this point, the time-stamped MAC addresses used to calculate travel time samples can be discarded. The final step in the process is to calculate travel time statistics from the travel time samples stored in the database. Standard reports and features for creating custom reports can be incorporated as part of processes established to utilize the travel time data.

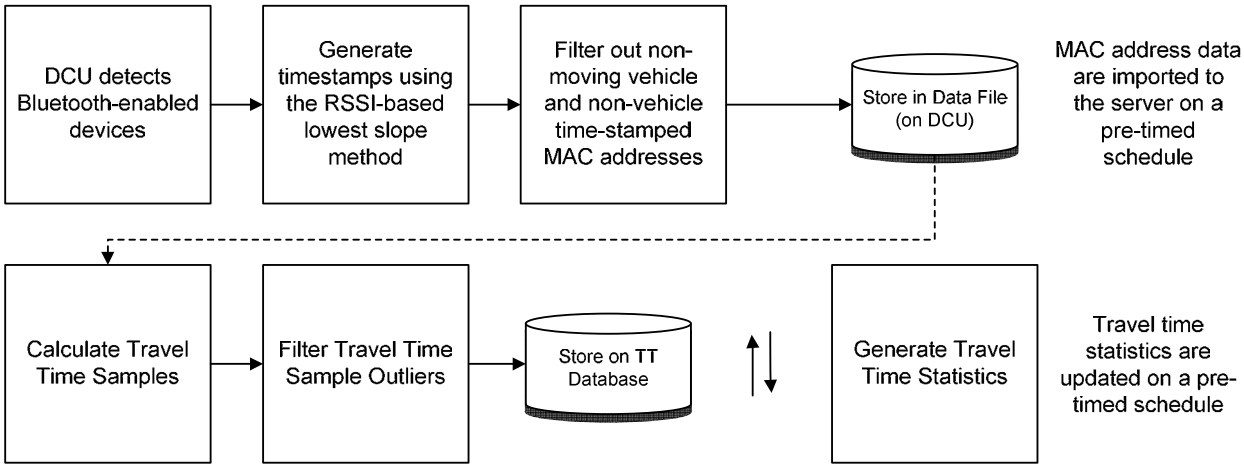

Figure 6.2: Data collection, data filtering and data analysis steps

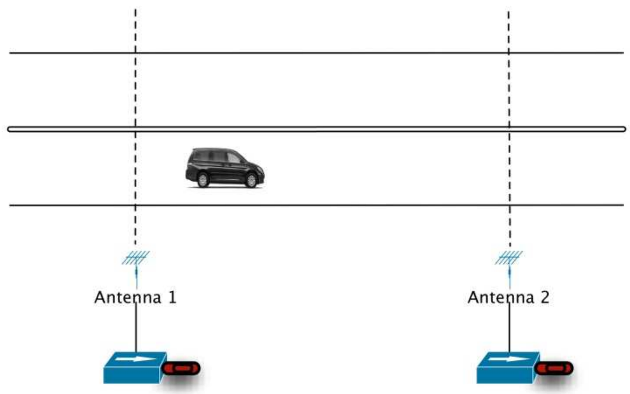

Figure 6.3: Consecutive DCUs covering a road segment

Several procedures were developed to accomplish the steps depicted in Figure 6.2. Some of these procedures are implemented and executed at the DCUs, while others are implemented and executed at the central server. The following subsections describe these procedures in more detail. First, a brief explanation of the DCU and server procedures is presented. Later, detailed pseudocode for these procedures is provided.

6.2 DATA COLLECTION UNIT PROCEDURES

6.2.1 Filtering MAC Address for Traveling Vehicles

The objectives of the filtering process are (1) to identify a single time-stamped MAC address that represents the point at which a vehicle carrying a Bluetooth-enabled mobile device is located on an imaginary line perpendicular to the DCU, and (2) to remove time-stamped MAC addresses assumed to have been collected from Bluetooth-enabled devices not contained in moving vehicles. The filtering technique used to accomplish these two objectives is referred to as the *RSSI-based smallest slope method*.

There are three processing steps involved in filtering time-stamped MAC addresses and they are repeated every 3.84 seconds. The 3.84-second interval is based on the length of the Bluetooth inquiry procedure currently utilized in the DCUs. Table 6.1 shows the data fields that are reported by a DCU for each Bluetooth-enabled device detected: (1) MAC address, (2) the time stamp that corresponds to the highest RSSI level detected in the 3.84-second interval, and (3) the value of the highest RSSI (referred to as the *current RSSI*). Table 6.2 shows the data fields that are used to construct a temporary data array in the DCU's memory. This temporary data array is used to identify and filter outliers and to perform the RSSI-based smallest slope method.

Table 6.1: Fields of detected MAC-addresses

Field #	Field Description
1	MAC address of detected Bluetooth device
2	Time stamp (date/time) of highest RSSI detected
3	Current RSSI

Table 6.2: Field in the temporary data array

Field #	Field Description
1	MAC address of detected Bluetooth device
2	Time stamp (date/time) for last RSSI
3	Last RSSI
4	Time stamp (date/time) for local highest RSSI
5	Current slope
6	Time stamp (date/time) for highest RSSI of smallest slope
7	Smallest slope
8	Number of detections
9	Number of consecutive non-detections
10	Slope trend indicator

The steps of the RSSI-based smallest slope method are as follows:

Step 1: Record all detected MAC addresses.

Step 2: Update the temporary data array.

Step 2a: If a time-stamped MAC address is already present in the temporary data array:

- Calculate the current slope as (*current RSSI – last RSSI*) divided by (Time stamp (date/time) of highest RSSI detected - Time stamp (date/time) for last RSSI). The values in the numerator correspond to field 3 in Table 6.1 and field 3 in Table 6.2, respectively, whereas the values in the denominator correspond to field 2 in Table 6.1 and field 2 in Table 6.2, respectively. If the value of the current slope is less than the value of the smallest slope, then set the value of the smallest slope to the value of the current slope and set the time stamp (date/time) for the highest RSSI of smallest slope to the time stamp (date/time) for the local highest RSSI.

- Increase the number of detections by one.
- Set the number of consecutive non-detections to zero.
- If the value of the current slope is greater than zero, then set the value of the slope trend indicator to one. Otherwise, set the value of the slope trend indicator to zero. If the value of the current slope is greater than or equal to zero and the slope trend indicator is one, then set the time stamp (date/time) for the local highest slope to the detection time.
- If the time-stamped MAC address is not in the temporary data after the latest 3.84-second interval, then increase the number of consecutive non-detections for this record by one.
- If the number of detections is greater than 1000, discard the time-stamped MAC address from temporary data array.

Step 2b: If the detected time-stamped MAC address is not in the temporary data array:

- Add the time-stamped MAC address to the temporary data array.
- Set the value of the current slope to zero.
- Set value of the smallest slope to zero.
- Set the number of detections to one.
- Set the number of consecutive non-detections to zero.
- Set the time stamp (date/time) for the last RSSI, the time stamp (date/time) for the local highest RSSI, and the time stamp (date/time) for the highest RSSI of smallest slope to the detection time.
- Set the last RSSI to current RSSI.
- Set the slope trend indicator to one.

Step 3: Check if a group is formed and filter out groups of time-stamped MAC addresses whose size is 50 (most likely not produced by a vehicle).

- If the number of consecutive non-detections equals 17 (17*3.84 seconds = group threshold time of 65.28 seconds) a group is formed. However, if the number of detections in this group exceeds 50, then filter this MAC address since it has been identified as a non-vehicle MAC address.

- Set the value of the current slope to (– 90 – last RSSI) / 3.84. If the current slope is less than the smallest slope, then set the value of the smallest slope to the current slope and set the time stamp (date/time) for the highest RSSI of the smallest slope to the time stamp (date/time) for the local highest RSSI.

- If a group composed of time-stamped MAC address generated by a vehicle has been formed, add this MAC address, time stamp (date/time) for the highest RSSI of the smallest slope, and the MAC address of the DCU to the output data file stored in the DCU's hard drive. The data fields contained in this file are shown in Table 6.3. This data file will be retrieved by the central server on a predefined schedule to produce travel time samples. Then, the associated time-stamped MAC address data is discarded from the temporary data array in memory.

Table 6.3: Fields of data file to send to central server

Field #	Field Description
1	MAC address of detected Bluetooth device
2	Time stamp (date/time) for highest RSSI of smallest slope
3	MAC address of detector (or ID of detector)

6.3 SERVER PROCEDURES

6.3.1 Calculation of Travel Time Samples

Travel time samples are calculated using paired, time-stamped MAC addresses collected by two consecutive DCUs. In order to prevent using the same time-stamped MAC address in multiple travel time calculations, time-stamped MAC addresses that have already been used are discarded. If multiple candidate pairs of time-stamped MAC addresses exist, then the time-stamped MAC address pair that results in the shortest travel time sample is selected. If existing time-stamped MAC addresses do not qualify to be paired with newly imported time-stamped MAC addresses to calculate travel time samples because they are outdated, they will also be discarded. Outdated time-stamped MAC addresses are identified by checking the maximum time required to travel the road segment between two consecutive DCUs ($T_{maximum}$). Since time-stamped MAC addresses are imported every $T_{interval}$, travel time samples can be calculated at most every $T_{interval}$.

The steps to calculate travel time samples are the following:

Step 1: Import a new data file from two adjacent DCUs that define a road segment (call them DCU1 and DCU2). For each DCU, add the newly imported time-stamped MAC addresses to a temporary data file according to the structure shown in Table 6.4. Mark field 3 of all newly imported time-stamped MAC address data as "unused" for both DCU1 and DCU2. Remove all time-stamped MAC addresses whose time stamp (date/time) is $T_{maximum}$ seconds older than the recently imported time-stamped MAC addresses.

Table 6.4: Field of data file to use for travel time calculation for each DCU

Field #	Field Description
1	MAC address of detected Bluetooth device
2	Time stamp (date/time)
3	Used/unused

Step 2: Select an unused time-stamped MAC address from DCU1. The associated time stamp will be denoted as T_1.

Step 3: Find an unused matching time-stamped MAC address from DCU2 with time stamp T_2 that results in the smallest travel time sample ($T_a = |T_1 - T_2| < T_{maximum}$) when paired with the selected time-stamped MAC address from DCU1. If a T_a exists, mark the DCU1 time-stamped MAC address as "used" and go to Step 4. Otherwise, go to Step 2.

Step 4: Find an unused time-stamped MAC address from DCU1 with time stamp T_3 that results in the smallest travel time sample ($T_b = |T_2 - T_3| < T_{maximum}$) when paired with a selected time-stamped MAC address from DCU2. If T_b does not exist or T_a is less than T_b, mark the DCU2 time-stamped MAC address as "used" and update the data file to send travel time samples to the database. Set the time stamp (date/time) to the latest time stamps of T_1 and T_2, set the travel time to T_a, and set the direction based on T_1 and T_2. Otherwise, go to Step 2.

Step 5: Repeat Steps 2 and 3 for all unused time-stamped MAC addresses from DCU1.

6.3.2 Filtering travel time sample outliers

In some cases, abnormal travel time samples might be calculated. For example, very large travel time samples can be obtained when vehicles stop between two consecutive DCUs for a period of time. In addition, non-vehicle Bluetooth devices (e.g., bicycles and pedestrians) might also generate abnormal travel time samples. Therefore, it is necessary to identify these abnormal travel time samples and filter them out to improve the accuracy of travel time estimates.

Every individual travel time sample is checked to verify whether or not it is an outlier. For this purpose, the latest *N* travel time samples (not including outliers) for each road segment and direction must be maintained. Travel time samples can be filtered immediately after completing the travel time sample calculation. The steps are as follows:

Step 1: Calculate the average travel time and standard deviation for *N* travel time samples. *N*=100 is recommended.

Step 2: An individual travel time sample will be filtered out if:

a) The value of the individual travel time sample is lower than the amount of time it would take a vehicle to travel the length of the segment at twice the speed limit for the segment, or
b) The value of the individual travel time sample is larger than the average travel time multiplied by 1.65 times the standard deviation.

Step 3: If the current travel time sample is not filtered out in Step 2, discard the oldest travel time sample and add the current travel time sample. This record will be saved to a data file to be stored later in a database based on the field structure shown in Table 6.5.

Table 6.5: Field of data file to send travel time database

Field #	Field Description
1	Time stamp (date/time) – for reference
2	Travel time sample
3	Direction of travel (i.e., northbound, southbound, eastbound, westbound)

6.4 CALCULATION OF TRAVEL TIME STATISTICS

Travel time statistics can be calculated for each road segment and each predefined time segment immediately after completing the calculation of travel time samples and the filtering of outlier travel time samples. To execute this process, it is also necessary to determine a recurring schedule for updating travel time statistics. The formulas to calculate several of these statistics are described in Section 4.0.

7.0 SYSTEM DESIGN AND FUNCTIONAL REQUIREMENTS

The generation of travel time samples, identification of travel time outliers, and the calculation of travel time statistics are executed at a central server. The execution of these functions, as described in the prior sections, are tasks that are part of an overall system that pulls time-stamped MAC address data from the DCUs, generates travel time samples, computes travel time performance measures and statistics, generates reports and graphics, provides data to other applications (e.g., variable message signs), and archives data. The architecture, operational parameters, and functional requirements of the system will be presented in this section.

7.1 SYSTEM ARCHITECTURE AND DATABASE DESIGN

The process for generating travel time performance measures and statistics from travel time samples is illustrated in Figure 7.1. The process begins when a software process residing at a central server collects time-stamped MAC addresses from the DCUs. These time-stamped MAC addresses are then stored in a central database. Another software process will use the time-stamped MAC address records from two DCUs to generate travel time samples for a given road segment. Finally, travel time statistics and other performance measures are calculated from the travel time samples and stored in the central database so that reports can be generated and to facilitate short-term travel time forecasting.

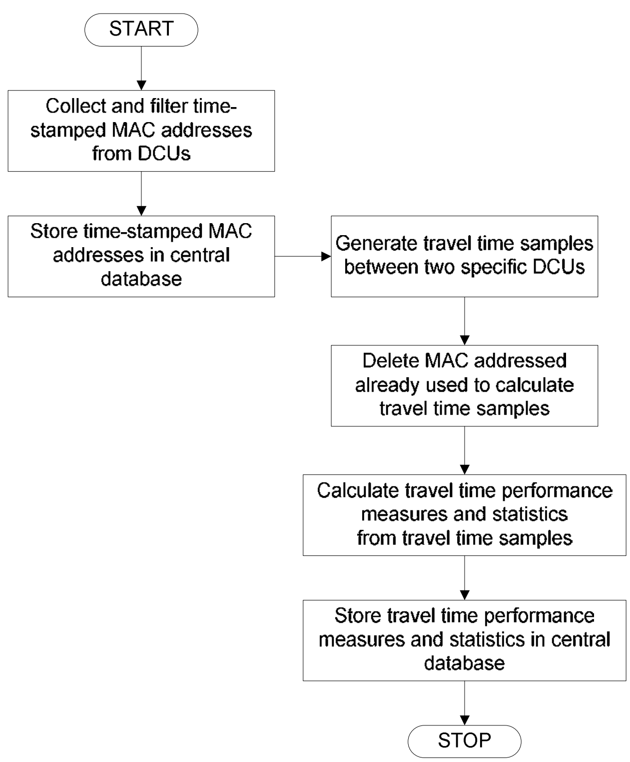

Figure 7.1: Process to generate travel time samples

The following subsections discuss in more detail the functional requirements of the central server software processes as well as of the central database needed to support a large-scale Bluetooth-based travel time data collection system.

7.1.1 Collecting and Storing Time-Stamped MAC Data from the DCUs

A data file containing only filtered time-stamped MAC addresses will be saved in the DCU and will be collected by a software process residing at the central server. The frequency at which the software process will perform this data transfer operation must be determined and it will be a parameter that can be adjusted depending on different performance requirements. Once a decision is made as to how frequently MAC address data files will be collected from the DCUs in the network (e.g., every two to three minutes), an automated service task can be scheduled by the central server. The central server software process will access each individual DCU on the network based on their internet protocol (IP) address. This information will be included in the central database table TBL_EQUIP_DESC, as shown in Table 7.1. As DCUs are added to the network, their information should be added to this central database table so that they are included in the process for generating travel time samples.

Table 7.1: Data elements and data types of table TBL_EQUIP_DESC

Attribute Name	Data Type	Length	Description
EQUIP_MAC_ADDR	Char	17	MAC address of the DCU
ODOT_DEVICE_ID	Numeric		ODOT assigned device number (not null)
DEVICE_IP_ADDRESS	Char	15	IP address of DCU
EQUIP_TYPE	Varchar	50	
EQUIP_MANUF	Varchar	50	
LAT_NO	Decimal	(9, 6)	Latitude coordinate of the DCU
LONGTD_NO	Decimal	(9, 6)	Longitude coordinate of the DCU
MP_NO	Decimal	(7, 4)	Approximate mile post where DCU is located

Once the software process at the central server is connected to a DCU, it shall do the following:

1. Copy the time-stamped MAC address records contained in the file *macs.txt* to a temporary database table in the central server.
2. Execute a subroutine to store the time-stamped MAC address records into the central database table TBL_RAW_DATA. The structure of this database table is shown in Table 7.2.
3. Delete the file *macs.txt* from the DCU.
4. Delete the time-stamped MAC address records from the temporary database table in the central server.

This four-step process shall be repeated with all DCUs registered in the central database table TBL_EQUIP_DESC.

Table 7.2: Data elements and data types of table TBL_RAW_DATA

Data Element	Data Type	Length	Description
EQUIP_MAC_ADDR	Char	17	MAC address of the DCU
REC_ID	Int		Record identification number
BT_MAC_ADDR	Char	9	MAC address collected by the DCU
MAC_ADDR_COLL_DT_TM	Datetime		Timestamp of the collected MAC address
RSSI_LEVEL	Int		Received signal strength indicator level

7.1.2 Generating and Storing Travel Time Samples

The basic procedure to generate travel time samples is described in Section 6.3.1. This process relies on the availability of a time-stamped MAC address recorded for the same vehicle at DCU1 and DCU2. When this procedure is implemented in a software process (whether as part of a stand-alone executable or as a procedure in the central database), the central database table TBL_RAW_DATA will be queried to identify the records that should be used to generate travel time samples for a road segment. A *road segment* is a specific traffic corridor contained between two DCUs and for which travel time samples need to be generated. Figure 7.2 illustrates two examples of the concept of a road segment. Note that in Figure (a), DCU1 and DCU2 define only one road segment. In contrast, Figure (b) shows an example of four road segments defined by five DCUs.

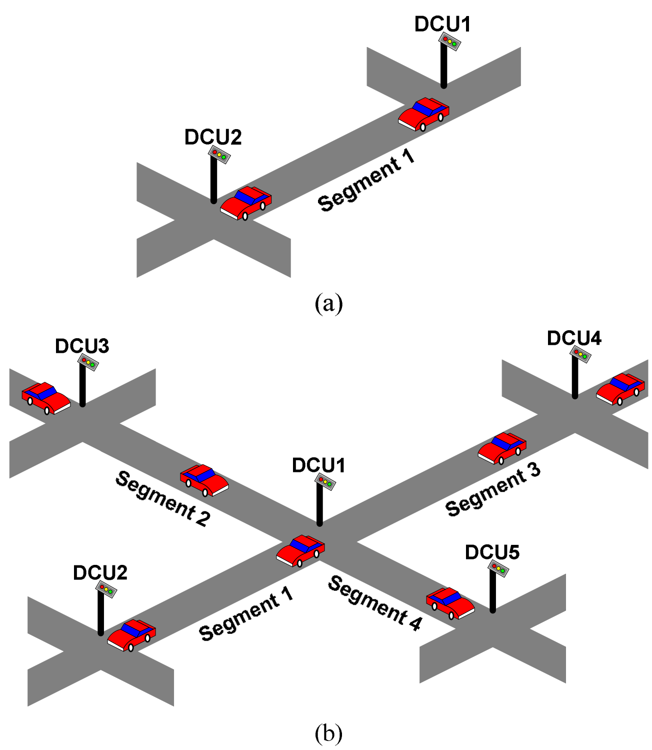

Figure 7.2: Examples of road segments

The structure of the central database table TBL_SEGMENTS that will store segment information is shown in Table 7.3 and is composed of four elements:

- ROAD_NUMBER: The road identifier (e.g., 99W, I-5, Powell Blvd., etc.)
- SEG_ID: The segment identification number.
- DCU1_MAC_ADDR: The MAC address of the Bluetooth interface of DCU1.
- DCU2_MAC_ADDR: The MAC address of the Bluetooth interface of DCU2.

Table 7.3: Data elements and data types of table TBL_SEGMENTS

Data Element	Data Type	Length	Description
ROAD_NUMBER	Char	15	Road number or name
SEG_ID	Int		Segment identification number
DCU1_MAC_ADDR	Char	17	MAC address of DCU1
DCU2_MAC_ADDR	Char	17	MAC address of DCU2

As an example, consider the five DCUs currently installed on Highway 99W in Tigard, Oregon. These five DCUs define four segments for which travel samples can be calculated. The records associated with this corridor would appear in the central database table TBL_SEGMENTS as shown in Table 7.4. In this case, the DCU located farther north in a segment on 99W (see Figure 3.9 and Figure 3.10) was considered DCU1. However, other criterion (or criteria) may be used when identifying the DCUs for the purpose of calculating travel time samples.

Table 7.4: Examples of records stored in table TBL_SEGMENTS

ROAD_NUMBER	SEG_ID	DCU1_MAC_ADDR	DCU2_MAC_ADDR
99W	1	00:01:95:0A:C9:8E	00:01:95:0A:C9:7F
99W	2	00:01:95:0A:C9:7F	00:01:95:09:5E:E8
99W	3	00:01:95:09:5E:E8	00:01:95:0A:C9:88
99W	4	00:01:95:0A:C9:88	00:01:95:0A:C9:5B

As DCUs are added or removed from the network and, as a consequence, segments are created or deleted, the information in the central database table TBL_SEGMENTS must be updated. It is critical that an interface is developed in the central server to allow a network administrator to enter or modify information in this table, especially since the records contained in it will guide the process of calculating travel time samples.

As travel time samples are generated for a specific segment, they will be stored in the table TBL_TRAVL_TIME_SAMPLES. The structure of this table is shown in Table 7.5. Each travel time sample will be uniquely identified by a composite primary key composed of the attributes road number (i.e., ROAD_NUMBER) and the segment identification number (i.e., SEG_ID).

Table 7.5: Data elements and data types of table TBL_TRAVL_TIME_SAMPLES

Data Element	Data Type	Length	Description
ROAD_NUMBER	Char	15	Road number or name
SEG_ID	Int		Segment identification number
TRVL_TIME_SAMPLE	Decimal	(6, 4)	Travel time data point for segment

7.1.3 Calculating Travel Time Performance Measures and Statistics

The travel time performance measures and other statistics that can be calculated using travel time samples are described in Section 6.4. The software process that will calculate these metrics will utilize the information in the central database tables TBL_RAW_DATA, TBL_SEGMENTS and TBL_TRAVL_TIME_SAMPLES. The steps that should be performed by the software process to complete these calculations are as follows:

1. Query the central database table TBL_SEGMENTS to identify the segments for which travel time performance measures must be calculated.

   Note: If desired, segments can be grouped into ***zones****. A zone can be defined as an area for which travel times performance measures must be calculated first. For example, there may be corridors in the Portland metro area that experience significant traffic congestion and therefore have a higher priority.*
2. Use each record retrieved in the prior step to query the central database table TBL_RAW_DATA to retrieve travel time samples for the corresponding DCU1 and DCU2.
3. Calculate travel time samples using the information contained in the table TBL_RAW_DATA to calculate travel time samples. Store travel time samples calculated for each road and each segment of this road in the table TBL_TRAVL_TIME_SAMPLES.
4. Use the travel time samples in the table TBL_TRAVL_TIME_SAMPLES to calculate the mean travel time for the selected records as well as the standard deviation, minimum and maximum.
5. Store the travel time performance measures and other statistics in the central database table TBL_TRAVEL_STATS.

7.1.4 Storing Travel Time Statistics

Travel time statistics will be saved in the central database table TBL_TRAVEL_STATS. The structure of this table is shown in Table 7.6.

Table 7.6: Data elements and data types of table TBL_TRAVEL_STATS

Data Element	Data Type	Length	Description
ROAD_NUMBER	Char	15	Road number or name
SEG_ID	Int		Segment identification number
TRVL_TIME _DT_TM	Datetime		Date and time when the travel time estimate was calculated
TRVL_TIME_AVG	Decimal	(6, 2)	Mean travel time for segment
TRVL_TIME_STDEV	Decimal	(6, 2)	Standard deviation of the mean travel time for segment
TRVL_TIME_MIN	Decimal	(6, 2)	Minimum travel time for segment
TRVL_TIME_MAX	Decimal	(6, 2)	Maximum travel time for segment

The information stored in the central database table TBL_TRAVEL_STATS can be used to produce reports that show mean travel time trends for specific travel corridors and also to perform short-term forecasting of expected mean travel times. Since this information does not contain MAC address data, it can be kept in the central database for as long as the system administrator deems reasonable.

7.1.5 Deleting MAC Address Data to Protect Privacy

There are two places in the Bluetooth-based data collection system where MAC address data is stored for some period of time. First, DCUs keep files of time-stamped MAC addresses so that the central server can pull these files on a pre-defined schedule. There are two ways in which these MAC address files can be removed from the DCUs:

- The file can be deleted from the DCU's compact memory flash (CMF) card every time the central server pulls the file.
- The file can be deleted from the DCU's CMF card every 24 hours by running a script.

Second, time-stamped MAC address data will be stored in the central database table TBL_RAW_DATA. The data in this table shall be kept for a maximum of 24 hours. Thus, a software process needs to be implemented at the central server to identify those records in the central database table TBL_RAW_DATA that exceed the 48-hour threshold and to permanently remove them from the database.

8.0 REFERENCES

Acharya, S., M. Franklin, and S. Zdonik. Balancing Push and Pull for Data Broadcast. Proceedings of ACM SIGMOD Conference, 1997, pp. 183-194.

Aksoy, D., and M.S.F. Leung. Pull vs push: a quantitative comparison for data broadcast. Presented at 2004 Global Telecommunications Conference, GLOBECOM '04. Vol. 3, 2005, pp. 1464 - 1468.

Barcelo, J., L. Montero, L. Marques, and C. Carmona. *Travel Time Forecasting and Dynamic OD Estimation in Freeways Based on Bluetooth Traffic Monitoring*. CD-ROM. Presented at TRB 2010 Annual Meeting Proceedings, Washington, D.C., 2010.

Bashir, Z. A. and M.E. El-Hawary. Applying wavelets to short-term load forecasting using PSO-based neural networks. *IEEE Transactions on Power Systems*, Vol. 24, No. 1, 2009, pp. 20-27.

Cetin, M. and G. Comert. Short-term traffic flow prediction with regime switching models, *Transportation Research Record – Traffic Flow Theory*, No. 1965, Transportation Research Board of the National Academies, Washington, D.C., 2006, pp. 23-31.

Chandola, V., A. Banerjee, and V. Kumar. Anomaly Detection: A Survey. *ACM Computing Surveys*, Vol. 41, No. 15, 2009, pp. 15:1 - 15:58.

Chandra, S. R. and H. Al-Deek. Prediction of freeway traffic speeds and volumes using vector autoregressive models. *Journal of Intelligent Transportation Systems*, Vol. 13, No. 2, 2009, pp. 53-72.

Cryer, J.D., and K. Chan. *Time Series Analysis with application in R*, 2nd Ed., Springer, New York, 2008.

Dion, F. and H. Rakha. Estimating dynamic roadway travel times using automatic vehicle identification data for low sampling rates, *Transportation Research Part B: Methodological*, Vol.40, No. 9, 2006, pp. 745-766.

FHWA. *Regional ITS Architecture Guidance Document*. Publication FHWA-HOP-06-112. Department of Transportation, Office of Operations. Washington, D.C., 2006.

Fishman, G.S. *Concepts and Methods in Discrete Event Digital simulation*, John Wiley, New York, 1973.

Fowler, M. Information Systems Architecture. ICSE '02 Proceedings of the 24th International Conference on Software Engineering, New York, NY, 2002.

Haghani, A., H. Masoud, F.S. Kaveh, S. Young, P. Tarnoff. Freeway Travel Time Ground Truth Data Collection Using Bluetooth Sensors. CD-ROM. Presented at TRB 2010 Annual Meeting Proceedings, Washington, D.C., 2010.

Jiang, X. and H. Adeli. Dynamic wavelet neural network model for traffic flow forecasting, *Journal of Transportation Engineering*, Vol. 131, No. 10, 2005, pp. 771-779.

Kamarianakis, Y., A. Kanas, and P. Prastacos. Modeling traffic volatility dynamics in an urban network, *Transportation Research Record – Network Modeling,* No. 1923, Transportation Research Board of the National Academies, Washington, D.C., 2005, pp. 18-27.

KMJ Consulting, Inc. Bluetooth Travel Time Technology Evaluation Using the Blue TOAD™. Report for E01271, PennDOT District 6-0 ITS & Support Open End Contract – Work Order #1, 2010.

Law, A.M., and W.D. Kelton. *Simulation Modeling and Analysis.* 3rd Ed., McGraw Hill, 2000.

Logendran, R., and L. Wang. *Dynamic Travel Time Estimation Using Regression Trees*. Report SPR 304-351. Oregon Department of Transportation Research Report , Salem, OR, 2008.

Malinovskiy, Y., Y. Wu, Y. Wang, and U. Lee. Field Experiments on Bluetooth-based Travel Time Data Collection. CD-ROM. Presented at TRB 2010 Annual Meeting Proceedings, Washington, D.C., 2010.

Malinovskiy, Y., U. Lee, Y. Wu, and Y. Wang. Investigation of Bluetooth-Based Travel Time Estimation Error on a Short Corridor, CD-ROM. Presented at TRB 2010 Annual Meeting Proceedings, Washington, D.C., 2011.

Nelson, R. Push and Pull, Lerner Books, 2008.

Okutani, I., and Y.J. Stephanedes. Dynamic prediction of traffic volume through Kalman filtering theory, *Transportation Research Part B*, Vol. 188, No. 1, 1984, pp. 1-11.

Porter, J.D., D.S. Kim, M.E. Magaña. Performance Monitoring for Transportation System Management and Operations. Unpublished manuscript. Oregon Department of Transportation, Salem, OR, 2009.

Porter, J.D., D.S. Kim, M.E. Magaña. *Wireless Data Collection System for Real-Time Arterial Travel Time Estimates*. ODOT Research Report Number OR-RD-11-10. Oregon Department of Transportation, Salem, OR, 2011.

Puckett, D.D., and M.J. Vickich. *Bluetooth-Based Travel Time/Speed Measuring Systems Development*. Final Report. University Transportation Center for Mobility Project Report #09-00-17, Texas Transportation Institute, 2010.

Puckett, D.D., and M.J. Vickich. Bluetooth-Based Travel Time Speed Measuring Systems Development. Presented at NATMEC, Seattle, WA, 2010.

Quayle, S.M., P. Koonce, D. DePencier, and D. Bullock. Arterial Performance Measures Using MAC Readers: Portland Pilot Study. CD-ROM. Presented at TRB 2010 Annual Meeting Proceedings, Washington D.C., 2010.

Quayle, S.M., and P. Koonce. Arterial Performance Measures Using MAC Readers--Portland's Experience. Presented at North American Travel Monitoring Exposition and Conference, Seattle, WA, 2010.

Sadabadi, K. F., M. Hamedi, and A. Haghani. Evaluating moving average techniques in short-term travel time prediction using an AVI dataset. CD-ROM. Presented at TRB 2010 Annual Meeting Proceedings, Washington D.C., 2010.

Schneider, M., Linauer, M., Hainitz, N., and H. Koller. Traveller information service based on real-time toll data in Austria. *IET Intelligent Transport Systems*, Vol. 3, Iss. 2, 2009, pp. 124-137.

Smith, B. L., and M.J. Demetsky. Short-term traffic flow prediction: neural network approach, *Transportation Research Record - Intelligent Transportation Systems: Evaluation, Driver Behavior, and Artificial Intelligence,* No. 1453, Transportation Research Board of the National Academies, Washington, D.C., 1994, pp. 98-104.

Stallings, W. *Data and Computer Communications*. Pearson Education, Inc.: Upper Saddle River, NJ, 2010.

Stathopoulos, A. and M.G. Karlaftis. A multivariate state space approach for urban traffic flow modeling and prediction, *Transportation Research Part C: Emerging Technologies. 2003*, pp. 121-135.

Tarnoff, P.J., and S.E. Young. Bluetooth Re-identification Technology and its application to Arterial Road Networks. 2010.

Tsekeris, T., and A. Stathopoulos. Real-time traffic volatility forecasting in urban arterial networks, *Transportation Research Record – Network Modeling.* No. 1964, Transportation Research Board of the National Academies, Washington, D.C., 2006, pp. 146-156.

Wasson, J.S., J.R. Sturdevant, and D.M. Bullock. Real-Time Travel Time Estimates Using Media Access Control Address Matching. *ITE Journal,* Vol. 78(6), 2008, pp. 20-23.

Wolfe, M., C. Monsere, P. Koonce, and R.L.Bertini. Improving Arterial Performance Measurement Using Traffic Signal System Data. Presented at ITE District 6 Annual Meeting, 2007.

Wouters, J.A.A., K. Chan, J. Kolkman, and R.W. Kock. Customized pretrip prediction of freeway travel times for road users, *Transportation Research Record – Data Initiatives,* No. 1917, Transportation Research Board of the National Academies, Washington, D.C., 2005, pp. 28-36.

Wu, Y., S. An, X. Ma, and Y. Wang. Development of a web-based arterial network system for real-time decision support. CD-ROM. Presented at TRB 2011 Annual Meeting Proceedings, Washington, D.C., 2011.

Xie, Y., Y. Zhang, and Z. Ye. Short-term traffic volume forecasting using Kalman filter with discrete wavelet decomposition. *Computer-Aided Civil and Infrastructure Engineering*, Vol. 22, 2007, pp. 326-334.

Yang, F., Z. Yin, H.X. Liu, and B. Ran. Online recursive algorithm for short-term traffic prediction, *Transportation Research Record – Information Systems and Technology,* No. 1879, Transportation Research Board of the National Academies, Washington, D.C., 2004, pp. 1-8.

Zheng, W., D. Lee, and Q. Shi. Short-term freeway traffic flow prediction: Bayesian combined neural network approach, *Journal of Transportation Engineering*, Vol. 132, No. 2, 2006, pp. 114-121.

APPENDIX A: USER MANUAL

A-1.0 INTRODUCTION

The User Manual includes instructions on how to assemble and configure the DCU to collect media access control (MAC) addresses from Bluetooth-enabled devices. It has been developed as part of the research project, *Wireless Data Collection System for Travel Time Estimation and Traffic Performance Evaluation*, which is the third of a series of projects focusing on an implementable Bluetooth-based travel time data collection system. The method of travel time data collection made possible by Bluetooth technology is vehicle re-identification, which requires identifiable Bluetooth-based devices associated with vehicles, and roadside data collection units (DCUs) to detect these devices

In addition to instructions for assembling and configuring the DCU, troubleshooting procedures are also included to address potential malfunctions that may occur when the unit is deployed in the field. The information has been organized in the following sections:

- Assembling the DCU
- Cloning the software image to a new compact flash card (CFC)
- Configuring the ALIX Board to Run the Bluetooth Script
- Configuring a Cellular Router to Access the DCU Remotely
- Accessing the DCU via a Remote Connection
- Accessing the DCU via a Direct Connection
- Troubleshooting the DCU

A-1.1 LIST OF ACRONYMS

There are several acronyms used throughout the User Manual. These acronyms are listed below along their meanings.

ACRONYM	**MEANING**
CFC	Compact flash card
DCU	Data collection unit
IP	Internet protocol
LAN	Local area network
MAC	Medium access control
POE	Power over Ethernet
SSH	Secure shell
TCP	Transmission control protocol
Telnet	Teletype network
USB	Universal serial bus
VGA	Video graphics array

A-2.0 ASSEMBLING THE DCU

Table A-1 shows the components required to assemble a DCU. Parts names, quantities needed of each part, and URLs to vendors' web sites are included. The last column in Table A-1 indicates the figure within this section that depicts the corresponding part.

Table A-1: Components Needed to Assemble DCUs

PART NAME	QTY	WHERE TO BUY IT	FIGURE #
ICF 4000 Industrial Compact Disk 1GByte	1	http://www.mini-box.com/1GB-Inndustrial-Compact-Disk-4000	Figure A-1
ALIX3D3 board	1	http://www.mini-box.com/Alix-3D-Board-3-LAN-1-MINI-PCI-1_3?sc=8&category=19	Figure A-2
Indoor enclosure for ALIX.3 Boards	1	http://www.mini-box.com/ALIX-3-Enclosure?sc=8&category=19	Figure A-3
I/O Bracket for Alix.3D3 (includes 8 screws)	2	http://www.mini-box.com/Bracket-for-Alix-3c2_2	Figure A-4
18w (15v/1.2A) AC-DC Power Adapter	1	http://www.mini-box.com/60w-12v-5A-AC-DC-Power-Adapter_3?sc=8&category=19	Figure A-5a
POE adapter	1	http://www.mini-box.com/s.nl/it.A/id.309/.f	Figure A-5b
SENA UD100 Bluetooth dongle	1	http://www.sena.com/where_to_buy/channel_partners/	Figure A-5c

Figure A-1: ICF 4000 industrial compact disk 1GB

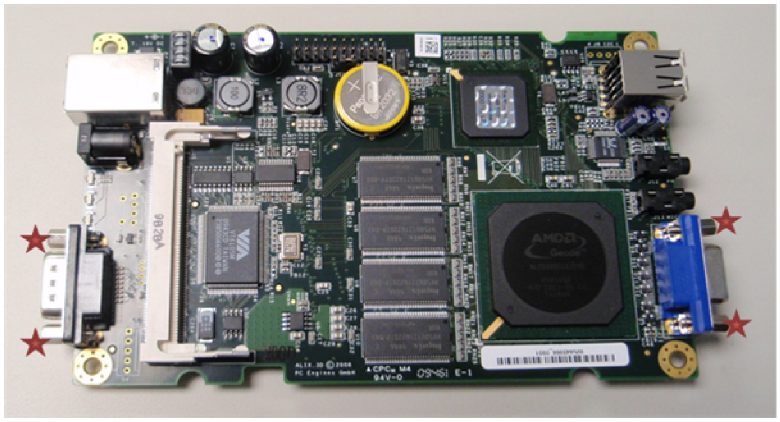

Figure A-2: ALIX3D3 board

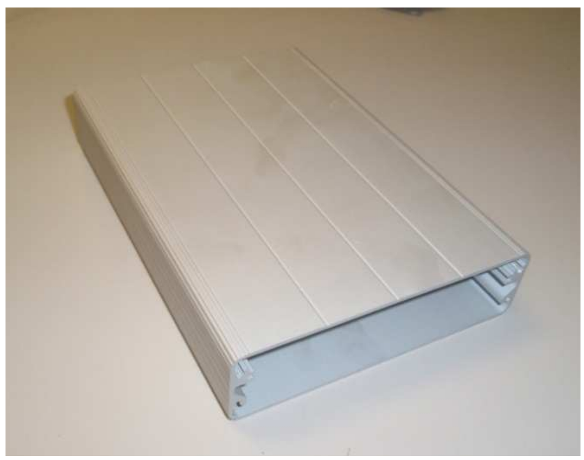

Figure A-3: Indoor enclosure for ALIX.3 Boards

Figure A-4: I/O bracket for ALIX.3D3

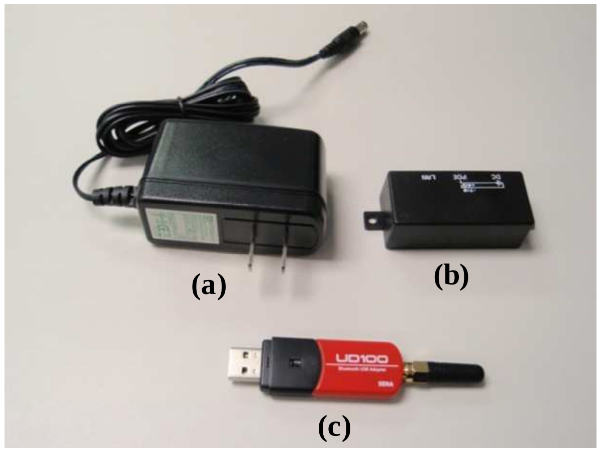

Figure A-5: DCU components: (a) power supply, (b) POE adapter, and (c) Bluetooth USB adapter

The following steps detailed the instructions to assemble a DCU from the parts listed in Table A-1.

1. Install the compact flash card (CFC) on the ALIX3D3 board.

2. Figure A-6 shows the top side of the board. The slot where the Compact Flash Card needs to be installed is on the bottom side. Slide the card in the direction of the red arrow, as depicted in Figure A-6.

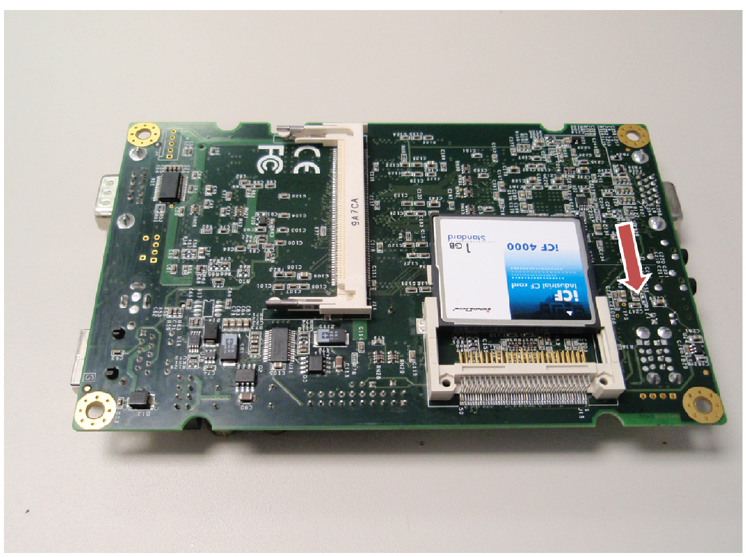

Figure A-6: Insert compact flash card into ALIX board

3. The ALIX board needs to be inserted into the indoor enclosure, as depicted in Figure A-7. The screws marked with red stars in Figure A-2 will need to be removed to install the I/O brackets.

Figure A-7: Slide ALIX board into enclosure

4. The I/O brackets can now be installed on both ends of the DCU's enclosure, as shown in Figure A-8. Note that the screws marked with red stars in Figure A-2 are now back in place.

Figure A-8: Installation of I/O brackets

5. The last step in completing the assembly of the DCU is to connect the Bluetooth USB adapter, the power supply, and the Ethernet cables. Two Ethernet cables are needed. Figure A-9 shows how the final DCU assembly looks. The yellow Ethernet cable goes from the POE to a device that provides network connectivity, such as a cellular router or a LAN router. The grey Ethernet cable provides the DCU with power and network communications. Finally, the power supply needs to be connected to a power source.

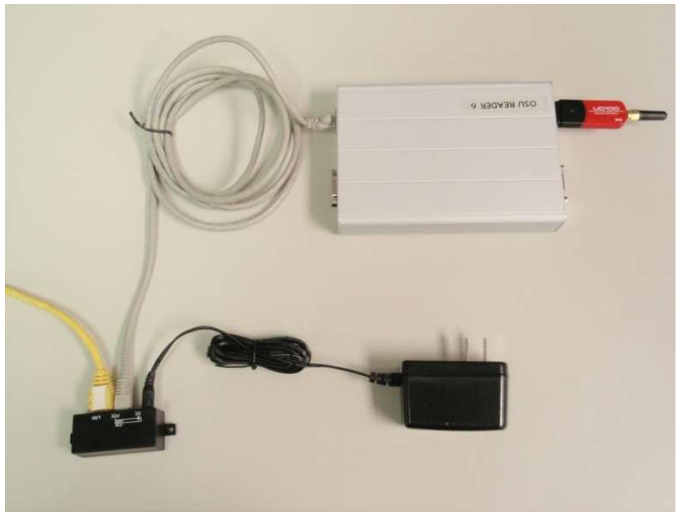

Figure A-9: DCU with POE and Bluetooth USB adapter

Depending on the final installation in the field, this configuration needs to be established in order for the device to work properly.

A-3.0 CLONING THE SOFTWARE IMAGE TO A NEW COMPACT FLASH CARD

All the software needed for the DCU to function properly, including the operating system and data collection script, is stored in a one gigabyte (GB) compact flash card (CFC). This section outlines the steps required to format and install the software onto a CFC. It is assumed that a blank CFC will be "cloned" using a completely formatted CFC. For the cloning procedure, a ready-to-go disk image is provided. The steps to clone the CFC are as follows:

1. Insert the CFC into a card reader and connect the card reader to a computer running Linux. It is highly recommended to use the Ubuntu Linux distribution.

2. Before installing the image in the new and blank CFC, the CFC needs to be formatted using Gparted. In Ubuntu, Gparted can be found at System > Administration > Gparted. Once in Gparted, select the CFC (i.e., "/dev/sdb"). Then make sure that the right CFC is selected and double-check its size. If there is any previous partition in the CFC, it must be erased. If there is no pre-existing partition, create a new partition. Right click on the partition and select "Unmount," as depicted in Figure A-10.

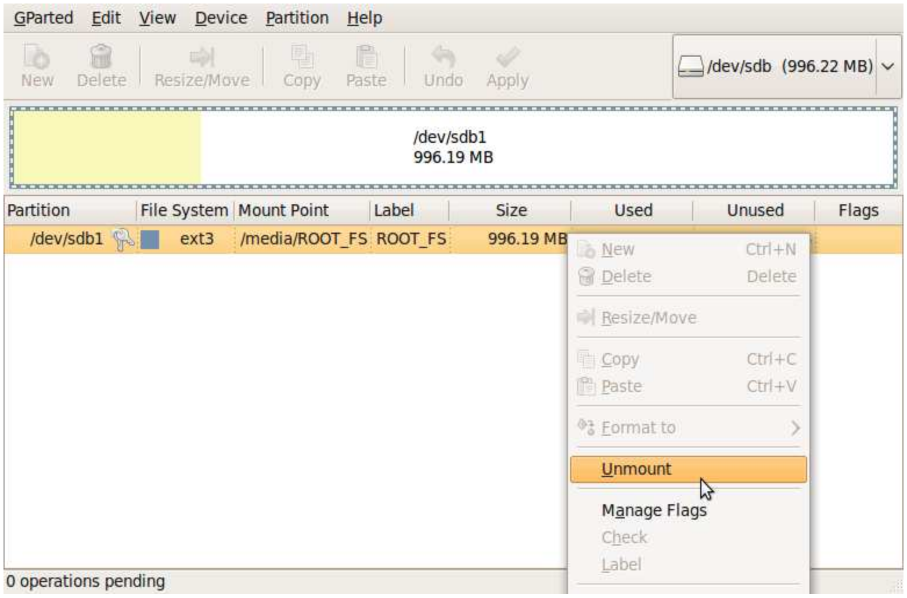

Figure A-10: Unmounting the CFC in Gparted

3. Left mouse click on the partition again to highlight it. The menu icon "Delete" will become active. Press the "Delete" button to delete the partition, as depicted in Figure A-11.

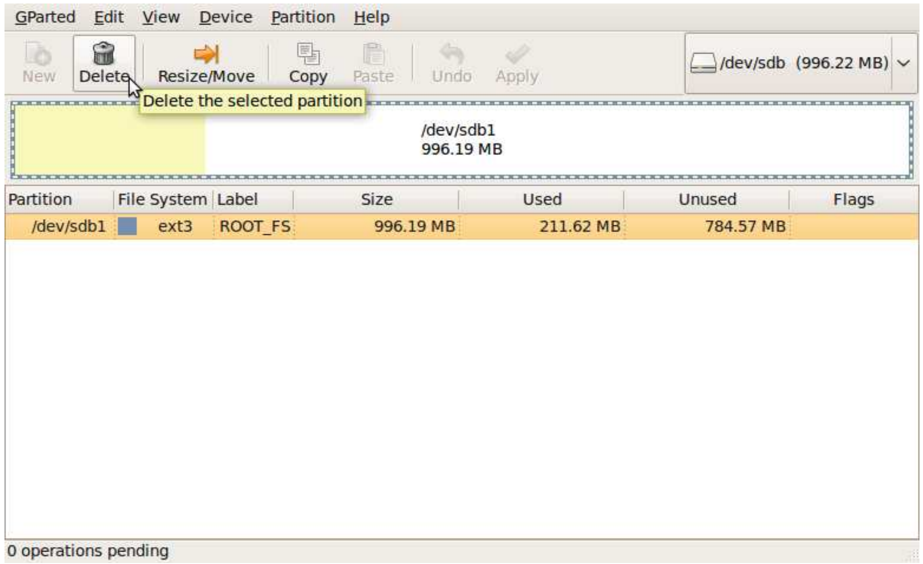

Figure A-11: Deleting the CFC's partition in Gparted

4. After clicking on the "Delete" icon, click on "Apply" to make the changes. Confirm the selection when prompted (see Figure A-12).

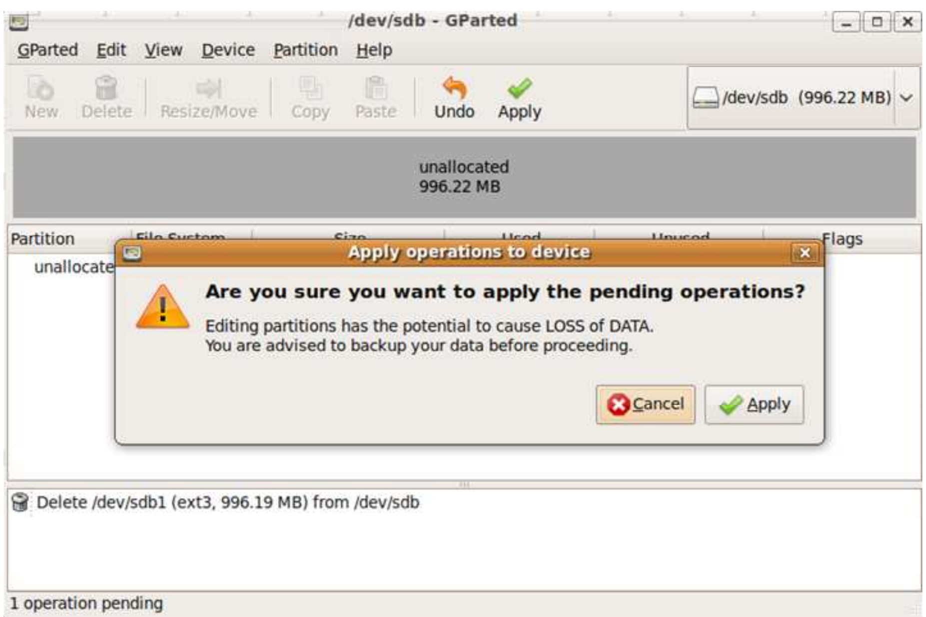

Figure A-12: Deleting the CFC's partition in Gparted (cont.)

5. After the configuration steps are completed in Gparted, a new partition must be created in the CFC. Highlight the partition and press the "New" button. A new window will appear. Select "ext3" from the drop down menu labeled "File System." Press the "Add" button to set the changes (see Figure A-13) and this window will close. Click on "Apply" to create the partition.

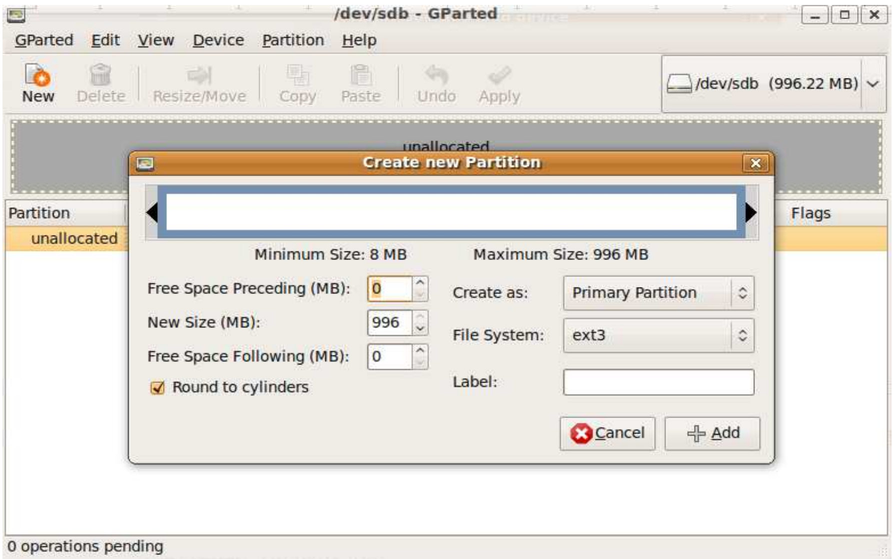

Figure A-13: Creating a partition in the CFC

6. Close Gparted and go to the Linux terminal window. In Ubuntu, the terminal window can be accessed via Applications > Accessories > Terminal. For this step, the user will need the provided ready-to-go disk image. The user must copy the disk image into the folder "/tmp/." In Ubuntu, the easiest way to copy the file is to first copy it to the computer Desktop. Then, use the move command in Linux terminal to move the disk image to the "/tmp/" folder by typing "mv ~/Desktop/BootCF.img /tmp/" and then pressing <Enter>.

7. Run the command enclosed by the red rectangle shown at the top of Figure A-14. Running this command may take several minutes and no feedback will be provided while the process is running. This command copies the image into the new CFC. In this example, it is assumed that the CFC image file is already copied to the folder /tmp/ of the local computer. If the image file is stored on a different location, the command presented in this example must also be modified to read the disk image from a different location.

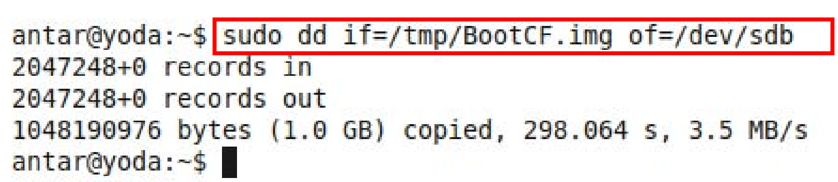

Figure 14: Copying image to the CFC

8. Once the output shown in Figure A-14 is displayed, the CFC is ready to be inserted into the ALIX board. For the DCU to be completely functional, the appropriate network and time server parameters need to be set. Network parameters (e.g., IP address, subnet mask, etc.) are necessary so that the unit can be accessed remotely. The time server parameters will allow the DCU to add accurate time stamps to the MAC address records it collects.

A-4.0 CONFIGURING THE ALIX BOARD TO RUN THE BLUETOOTH SCRIPT

This section details the steps needed to complete the configurations needed so the DCU unit can become operational. In particular, it describes the necessary steps to set the network and time server settings in the DCU. It is assumed that the DCU is connected to a LAN and that the following hardware is available:

- ALIX board with a cloned CFC.
- A VGA PC monitor.
- A USB keyboard.
- Ethernet cable.

It is important to note that the procedures explained in Section A-6.0 Accessing the DCU via a Remote Connection*)* and Section A-7.0(Accessing the DCU via a Direct Connection) can be used to gain access to the DCU instead of using the hardware listed above. If this is the case, skip to Step 2 in the detailed procedure that follows:

1. Connect the ALIX board to a keyboard and a PC monitor. Do not turn on the power until all hardware components are properly connected. Also, the ALIX board needs to be connected to a power source and needs access to the Internet through the Ethernet port. Once all the connections are completed, turn on the power of the ALIX board.
2. As soon as the ALIX board is powered up, the operating system will boot. The booting procedure should be displayed on the PC monitor. After the booting process is complete, the screen displayed in Figure A-15 will appear. Type the login and password (i.e., *root* and *depaT#ts910*, respectively).

Figure A-15: Voyage Linux interface

3. Set Network Configurations: Once logged in to the DCU, the user needs to modify

the configuration file “interfaces,” which is typically located in “/etc/network.” This can be accomplished by running the command shown in Figure 4.16 directly on the DCU interface. Executing this command will open the “vi” editor.

```
voyage:/usr/odot# vi /etc/network/interfaces
```

Figure 4.16: Opening the configuration file “interfaces”

4. The information depicted in Figure A-17 will be displayed. The text in Figure A-17 shown inside the red rectangle is an example for the network information that needs to be set. Information such as the IP address, gateway and the subnet mask must be provided by the IT department (see examples of these entries in Figure A-17). Use the arrow keys to move the cursor to the location where text needs to be edited. Then press the “i” key on the keyboard. This will change the editor’s mode to editing status. Finish updating the network information. Once all the network parameters have been entered, press the “Escape” key. Then, type “:wq” and press <Enter>. This should save the changes and get the user back to the Linux console.

```
# Used by ifup(8) and ifdown(8). See the interfaces(5) manpage or
# /usr/share/doc/ifupdown/examples for more information.
auto lo
iface lo inet loopback

auto eth0
iface eth0 inet static
address 192.168.1.253
netmask 255.255.255.0
gateway 192.168.1.100


#auto wlan0
#iface wlan0 inet static
#        address 10.1.10.1
#        netmask 255.255.255.0
#        broadcast 10.1.10.255
#        up iwconfig wlan0 essid voyage mode Master
#        up nat.sh wlan0 eth0 "10.1.10.0/24"
```

Figure A-17: Contents of the configuration file “interfaces”

5. After the configuration file “interfaces” has been edited and closed, the command “/etc/init.d/networking restart” needs to be executed to restart the network.

6. To configure the time synchronization of the DCU, the timeserver setting needs to be modified. This information is typically located in the file “/etc/ntp.conf.” Execute the command “vi /etc/ntp.conf” to open it. Use the same procedures explained before to operate within the vi text editor interface. One example for contents of file is depicted in Figure 18. The user can set more than one timeserver in this configuration file (one is required). The timeserver address must be obtained from the IT department.

```
# ---Server Address---
server pool.ntp.org
server 10.12.24.87

# ---Drift File---
driftfile /etc/ntp/drift
```

Figure A-18: Contents of the file “ntp.conf”

7. Once the changes to the file “ntp.conf” are saved, the user should restart the unit. It can be verified that the DCU’s clock is synchronized with the server, by running the command “/etc/init.d/ntp status” as depicted in Figure A-19.

```
NTP server is running..
voyage:~#
```

Figure A-19: Executing the command ntpdate-debian

A-5.0 CONFIGURING A CELLULAR ROUTER TO ACCESS THE DCU REMOTELY

The Bluetooth-based travel time data collection system can utilize cellular routers to enable remote connectivity to the DCUs for downloading data files and for running diagnostics. This section describes the steps needed to configure a cellular router to enable communications between a DCU and a remote computer.

The DCU needs to be configured first to establish a network connection through the cellular modem. The procedure is very similar to the configuration process to connect the DCU to a LAN network, explained in Section A-4.0 (Configuring the ALIX Board to Run the Bluetooth Script).This requires the user to log in into the DCU directly, as explained in Section A-7.0(Accessing the DCU via a Direct Connection). Once logged in, the user needs to modify the configuration file "interfaces," which is typically located in "/etc/network." This can be accomplished by running the command shown in Figure A-20 directly on the DCU interface. Executing this command will open the "vi" editor.

```
voyage:/usr/odot# vi /etc/network/interfaces
```

Figure A-20: Opening the configuration file "interfaces"

The information depicted in Figure A-21 will be displayed. The text in Figure A-21 shown inside the red rectangle is the new information that needs to be added. The gateway and subnet mask are the internal IP addresses from the cellular router. Please confirm the network configurations required for your cellular modem device with the IT department.

```
# Used by ifup(8) and ifdown(8). See the interfaces(5) manpage or
# /usr/share/doc/ifupdown/examples for more information.
auto lo
iface lo inet loopback

auto eth0
iface eth0 inet static
address 192.168.1.253
netmask 255.255.255.0
gateway 192.168.1.100


#auto wlan0
#iface wlan0 inet static
#        address 10.1.10.1
#        netmask 255.255.255.0
#        broadcast 10.1.10.255
#        up iwconfig wlan0 essid voyage mode Master
#        up nat.sh wlan0 eth0 "10.1.10.0/24"
```

Figure A-21: Contents of the configuration file "interfaces"

After the configuration file “`interfaces`” has been edited and closed, the command “`/etc/init.d/networking restart`” needs to be executed to restart the network.

The cellular router should have the configuration depicted in Figure A-22. In the cellular router’s web interface, select the “Network” option from the configuration menu (left side of the screen).

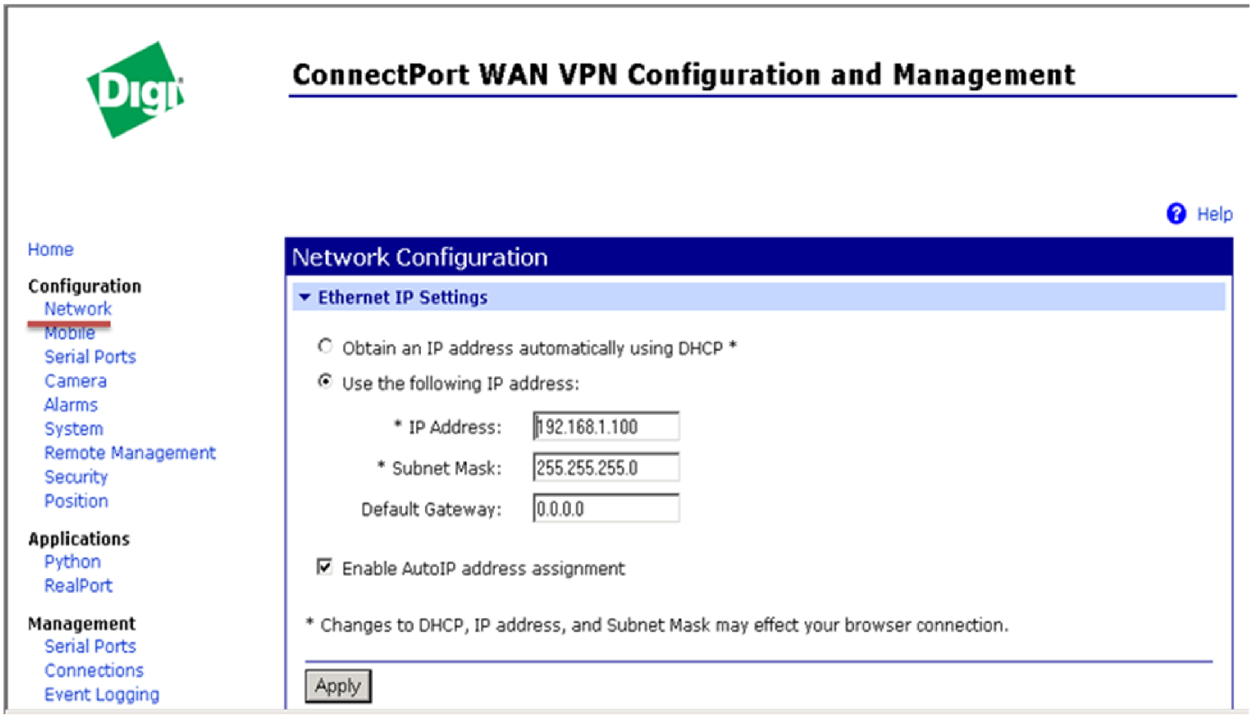

Figure A-22: Cellular router’s web interface configuration menu

Since data communication between the DCU and the remote computer will take place via port 22, the cellular router needs to be configured to forward all communications through this port. Scroll down on this window until the option “IP forwarding settings” is reached. Select this option and scroll down to find the table depicted in Figure A-23. An entry will be added to this table to forward a port to the DCU.

Enable	Protocol	External Port	Forward To Internal IP Address	Forward To Internal Port	Range Port Count	
☐	TCP	4000	10.0.46.52	80	1	Remove
☐	UDP	4000	10.0.46.52	80	1	Remove
☑	TCP	4000	192.168.1.101	80	1	Remove
☑	TCP	5000	192.168.1.101	5000	1	Remove
☑	TCP	23	192.168.1.253	23	1	Remove
☑	TCP	22	192.168.1.253	22	1	Remove
☑	FTP	21	192.168.1.253	23	1	Remove
☑	UDP	123	192.168.1.253	123	1	Remove
☑	UDP	5010	192.168.1.3	300	1	Remove
☑	TCP	5020	192.168.1.3	200	1	Remove
☑	TCP	22	192.168.1.253	22	1	Add

Apply

Figure A-23: IP forwarding table

In this example, the IP address of the DCU is 192.168.1.253, it uses port 22, and the transport protocol is TCP. To add the new entry, enter the following information:

1. Protocol: TCP
2. External port: 22
3. Forward to internal IP address: 192.168.1.253
4. Forward to internal port: 22
5. Press “Add”
6. Press “Apply”

The cellular router is now configured and should allow a remote computer to access the DCU for maintenance and troubleshooting operations.

A-6.0 ACCESSING THE DCU VIA A REMOTE CONNECTION

This section describes the process to remotely connect to the DCU. The connection process detailed here may be helpful if it is necessary to access the DCU to verify that the Bluetooth USB adapter is working properly or to determine whether or not the data collection script is running. It is important to note that the DCU and the computer (i.e., laptop or desktop) from which the remote communication is being established need to be in the same local area network (LAN).

A-6.1 CONNECTING REMOTELY VIA PUTTY

The DCU can be accessed remotely by using the program PuTTY. PuTTY is a Secure Shell (SSH) and teletype network (Telnet) client, developed originally by Simon Tatham for the Windows platform. PuTTY is open source software and is available for download at www.putty.org/.

Once the file PuTTY.exe has been downloaded and saved to the Desktop of the computer, follow these steps to remotely connect to the DCU:

1. Double click on the file PuTTY.exe to open it. The interface depicted in Figure 24 will appear.

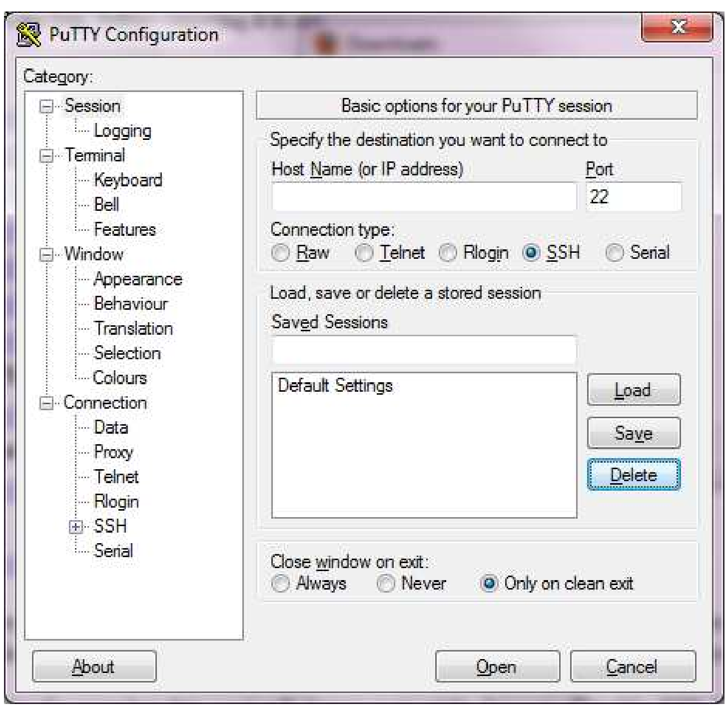

Figure A-24: PuTTY program interface

2. Enter the IP address of the DCU (e.g., 10.10.78.49) in the field “Host Name.” If a cellular router is being used, then the IP address of this device needs to be entered. Verify that the connection type selected is SSH and then press the “Open” button.
3. Another window will open. When prompted, enter “`root`” as the login and press <Enter>. Type “`depaT#ts910`” as the password and press <Enter> (Note: The cursor will not move while the password is being entered).
4. The DCU interface will open (see Figure 25). Now, the processes described in Section A-8.0 (Troubleshooting the DCU) can be executed.

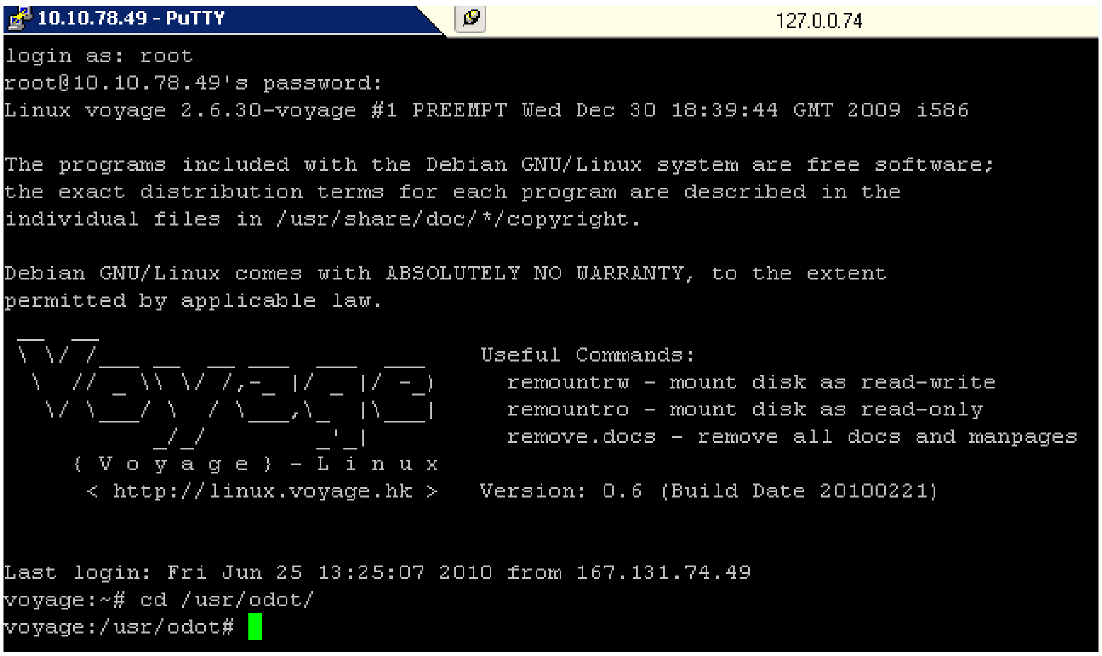

Figure A-25: DCU interface window

A-6.2 CONNECTING REMOTELY VIA A LINUX TERMINAL

A remote connection to the DCU can be established using a computer running the Ubuntu distribution of Linux.

1. Open a terminal window on the computer running Ubuntu.
2. Type the command “ssh root@[IP Address].” The [IP address] entered should be either the DCU’s IP address or the cellular router’s IP address (see Step 2 in Section A-6.1).
3. The DCU interface will open (see Figure 25). Now, the processes described in Section A-8.0 (Troubleshooting the DCU) can be executed.

A-7.0 ACCESSING THE DCU VIA A DIRECT CONNECTION

This section describes the process to connect directly to the DCU when a connection via a cellular router or a network is not available. The connection process outlined here may also be helpful if it is necessary to check the status of the DCU after completing the software installation process described in Section A-4.0 Configuring the ALIX Board to Run the Bluetooth Script.

. The hardware and software required includes:

- Laptop computer running either Windows or the Ubuntu distribution of Linux (version 9.04 or higher).
- Router. In this example, a DLINK WBR-2310 wireless router (http://www.dlink.com/products/?pid=470) was used.
- Ethernet cable.
- PuTTY program (www.putty.org/)

The steps to directly connect to the DCU are as follows:

1. Identify the slot labeled "LAN" on the power over Ethernet (POE) adapter depicted in Figure A-26. In a typical installation, the LAN slot is used to insert an Ethernet cable whose other end is connected to a cellular router or network router. If this is the case, unplug the end of the Ethernet cable connected to the cellular router or network router and connect it to the router being used to connect directly to the DCU.

Figure A-26: Power over Ethernet (POE) adapter

2. Connect the laptop computer to the router using an Ethernet cable.
3. Turn on the router.
4. Power cycle the DCU (i.e., unplug it from power and plug it back into power).

5. Open a web browser on the laptop computer and open the router's configuration page. In some laptop computers you will need to turn off the wireless network (if one is enabled). In this example, a D-Link wireless router is used and its configuration page is depicted in Figure A-27. The user name for this router is *admin* and there is no password.

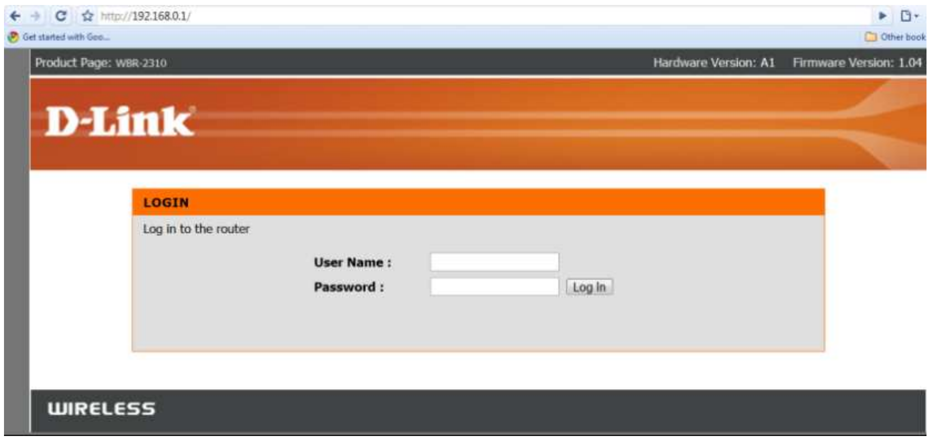

Figure A-27: D-Link configuration interface log in screen

6. Once inside the configuration menu of the D-Link wireless router, access the network settings page to see which devices are listed as being connected to the D-Link wireless router. An entry identifying the DCU should be listed as "voyage" (see Figure A-28). Copy the IP address shown for the DCU (in this example, it is 192.168.0.100).

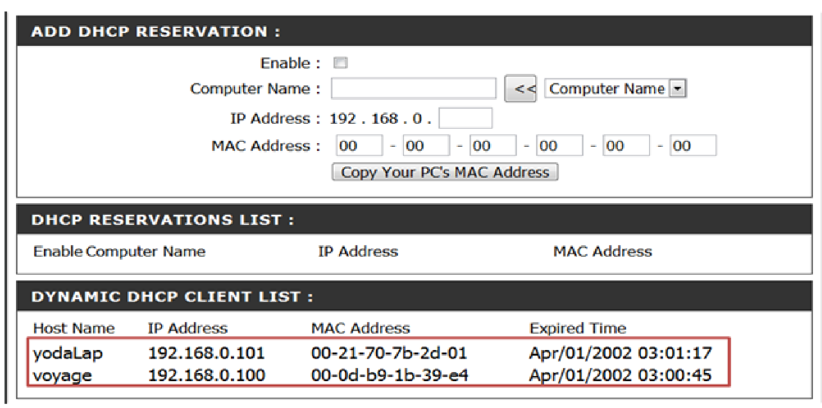

Figure A-28: Network settings page of the D-Link router

7. Open PuTTY (or if in Ubuntu open a console window) to connect to the DCU using the IP address copied from the D-Link wireless router (see Figure A-29). Once connected to the DCU, any of the troubleshooting tests can be performed on the unit.

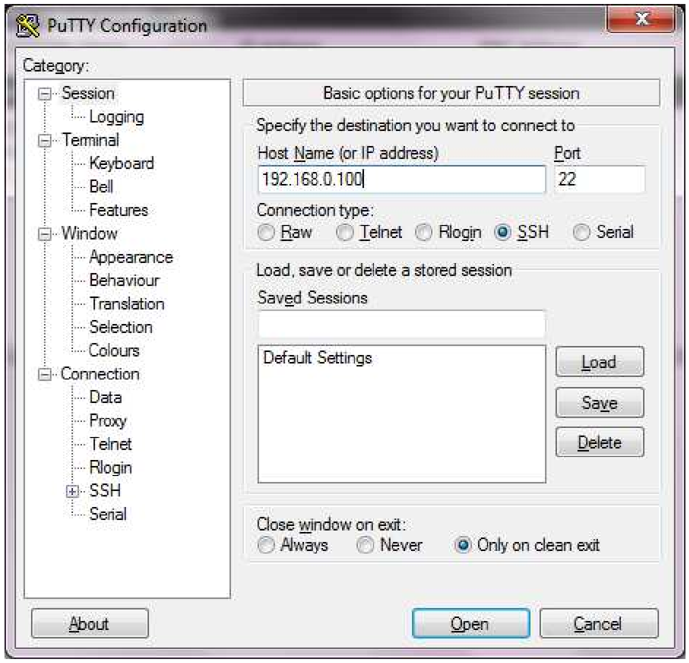

Figure A-29: PuTTY screen to connect to the router

A-8.0 TROUBLESHOOTING THE DCU

This section describes the steps needed to verify that:

- The Bluetooth USB adapter attached to the DCU is functioning properly, and
- The script in the DCU that collects MAC addresses from Bluetooth-enabled devices is running.

In order to perform these diagnostics, the user must be connected directly to the DCU or through a remote network connection. Please refer to Section A-6.0 (Accessing the DCu via a Remote Connection) or Section A- 7.0(Accessing the DCU via a Direct Connection**Error! Reference source not found.**) in this User Manual to review how to achieve connectivity to the DCU with either method.

A-8.1 RUNNING DIAGNOSTICS ON THE BLUETOOTH USB ADAPTER

The Bluetooth USB adapter connected to the DCU permits the collection of MAC addresses from Bluetooth-enabled devices. If there are reasons to suspect that the adapter is not working properly, execute the following steps:

1. Access the DCU via the PuTTY program, as described in Section A-6.0 (Accessing the DCU via a Remote Connection).
2. Navigate to the folder where the file that stores the collected MAC addresses is located by typing the command “cd /usr/odot” and then press the <Enter> key.
3. List all the files contained in the folder “/usr/odot” by typing the command “ls –all –h” and then press the <Enter> key (see Figure 30).

```
voyage:~# cd /usr/odot/
voyage:/usr/odot# ls -all -h
total 32K
drwxr-xr-x  2 root root 4.0K Jun 25 14:11 .
drwxr-xr-x 11 root root 4.0K May 12 13:49 ..
-rwxr-xr-x  1 root root   72 May 12 13:50 killscript.ssh
-rw-r--r--  1 root root  14K Jun 25 14:11 macs.txt
-rw-r--r--  1 root root    5 Jun 25 13:25 pid
-rw-r--r--  1 root root    0 Jun 25 14:11 rawSannedMacs.txt
voyage:/usr/odot#
```

Figure A-30: MAC address folder

4. Verify that the file ***macs.txt*** file resides in the folder “`/usr/odot`.” If it is not listed, then the Bluetooth USB adapter could be damaged.

5. First, verify that the Bluetooth USB adapter is working by typing the command "`hcitool dev`" followed by the <Enter> key. If the Bluetooth USB adapter is working properly, then its MAC address should be listed as part of the output of the "`hcitool dev`" command, as shown in Figure A-31.

Figure A-31: Output of the `hcitool dev` command

6. If the MAC address of the Bluetooth USB adapter is not shown in the output of the "`hcitool dev`" command, then the adapter is either off or not working properly. To verify that the adapter is powered, issue a reboot on the DCU by typing the command "`reboot -p`" followed by the <Enter> key.
7. Repeat steps 1 through 6 to verify that the Bluetooth USB adapter is working properly. If the problems persist, replace the Bluetooth USB adapter.

A-8.2 VERIFYING THAT THE LINUX SCRIPT IS RUNNING

A Linux script file running in the DCU enables the collection of MAC addresses from Bluetooth-enabled devices. To verify that the script is running, follow these steps:

1. Access the DCU via the PuTTY, as described in Section A-6.0 titled "*Accessing the DCU via a Remote Connection.*"
2. Type the command "`ps -ax`" followed by the <Enter> key. This command lists all the processes that are running in the DCU. The output depicted in Figure A-32 should be displayed on the DCU interface.

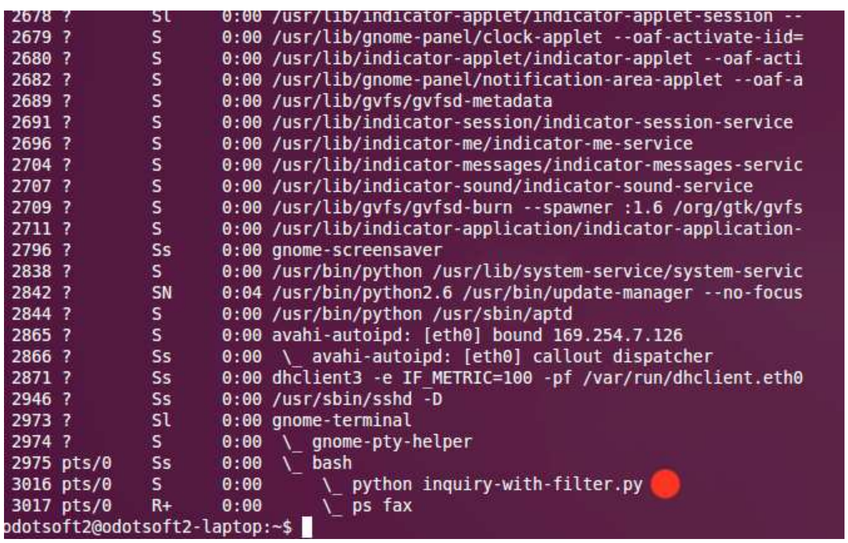

Figure A-32: Output of the command `ps -ax`

3. Verify that the process "`inquiry-with-filter.py`" is listed in the output. If is not listed, issue a reboot on the DCU by typing the command "`reboot -p`" followed by the <Enter> key.
4. Repeat Steps 1 and 2 and verify that the process "`inquiry-with-filter.py`" is listed.