A Smartphone-based System for Automated Congestion Prediction

Lance Fiondella[1], Swapna S. Gokhale[2], and Nicholas Lownes[3]
[1]Dept. of Electrical and Computer Engineering, University of Massachusetts, Darmouth, MA 02747
[2]Dept. of Computer Science and Engineering, University of Connecticut, Storrs, CT 06269
[3]Dept. of Civil and Environmental Engineering, University of Connecticut, Storrs, CT 06269

Abstract

Accurate collection of traffic data is essential, tactically for efficient highway operations and strategically for capacity planning purposes. Currently, this traffic data is collected by physical sensors, which suffers from many limitations. These sensors gather limited measurements of vehicle speeds and times from their fixed locations and cannot systematically acquire mobility dynamics of individual vehicles and the interactions among them. Also, because deploying and maintaining physical sensors is expensive and time consuming, typically they are installed to measure traffic only on the busier road segments within a transportation network. To overcome these issues, this paper reports on a general-purpose, location-aware, smartphone-based traffic monitoring system as a simple and an inexpensive alternative to collect dynamic vehicle data extensively through a transportation network. A privacy-preserving smartphone application was developed, deployed, and tested on the network surrounding the University of Connecticut (UConn) in Storrs. It securely transmits individual trip data over the Internet to the servers hosted at UConn. Preliminary experimentation suggests that crowdsourcing the collection of traffic data with smartphones can be cost effective and can lead to richer data sets spanning the entire web of roadways.

1 Introduction

Presently, many highway operations centers rely on physical sensors deployed at fixed locations to collect traffic data. Large volumes of video feeds collected by these sensors are then monitored and analyzed by human experts to identify evolving conditions that may disrupt traffic flow, including rush-hour congestion and accidents. These experts must also manually enter traffic alerts to be displayed on signs to caution motorists of the present and changing conditions that may negatively impact travel. Collecting traffic data using physical sensors suffers from three drawbacks:

- Sensors can collect measurements only from a set of fixed locations, with no means to either record or infer traffic conditions between these discrete collection points. Capturing traffic data on a continuum, however, can offer better insights into what is transpiring between these discrete locations. Based on such continuous data, we can infer traffic conditions that vehicles will encounter through the network, accurately predict impending congestion before it becomes severe, and issue warnings to motorists to seek alternate routes. Such proactive warnings may be superior compared to the current, reactive alerts that cannot prevent motorists from being engulfed in heavy congestion.
- Physical sensors collect measurements on the collective, aggregate traffic behavior rather than gathering data on the mobility dynamics of individual vehicles including their acceleration, decceleration, and steering. They also cannot report on the interactions among the vehicles. Vehicle-level measurements such as the acceleration and deceleration could be correlated to the safety of a road segment, while inter-vehicle interactions could offer new insights into how congestion forms and at what rate.
- Deploying and maintaining sensors on roadways can be expensive, time consuming, and hazardous. Because of these monetary and safety constraints, physical sensors may be installed only on busy roadways, and hence, automobile traffic traveling only on these road segments can be measured.

In the recent years, modern technologies such as Global Positioning System (GPS)-enabled smartphones have emerged as a promising alternative to overcome the data collection drawbacks posed by physical sensors. Such location-enabled smartphones can provide continuous streams of mobility data for individual vehicles that can support the construction of their time-varying models. These vehicle-level models can then be fused to form a more complete and nuanced picture of network conditions. An additional advantage of using smartphones is that the measurements are not limited to automobiles but can also be generated as a traveler navigates other modes including mass transit and pedestrian walkways. Therefore, smartphone measurements can support multi-modal models to characterize an individual's behavior traversing multiple means within a

(DOI reference number: 10.18293/SEKE2015-183)

transportation network. These models can thus provide a holistic view of how different modes interact, which have traditionally been studied in isolation. Finally, crowdsourcing data collection to smartphones is virtually free, incurs no deployment and maintenance costs, and promotes the safety of roadway workers.

Despite their promise, there is very little research on the suitability of GPS-enabled smart phones to monitor traffic conditions across the entire transportation network. Existing works demonstrate a narrow scope, such as: (i) measuring traffic from fixed or static locations [4, 2]; (ii) restricting to specialized networks such as highways [1] or university campuses [4]; or (iii) covering only one mode, for example, public transportation [5]. This paper reports on the architecture and non-functional properties of a general-purpose smartphone-based monitoring system to collect continuous streams of traffic measurements. The system captures a sequence of locations and transmits these data points over the public Internet to a server for storage and processing. Experimental deployment, data collection, and analysis on the transportation network around UConn demonstrates the promise of using crowdsourced technology for efficiently collecting richer traffic data through the web of roadways spanning the entire transportation network. Such detailed data can feed sophisticated models which can be used to plan capacity, enhance road safety, and reduce congestion.

The rest of the paper is organized as follows: Section 2 describes the system architecture and its properties. Section 3 presents the preliminary experiment and analysis. Section 4 surveys related research. Section 5 offers conclusions and directions for future research.

2 Traffic Monitoring System

We describe the architecture and non-functional properties of the smartphone system for collecting traffic data.

2.1 Architectural View

The following three major components of the smartphone system are organized as a pipeline. This section describes the functions and the technologies used to implement these components and the communication between them.

The **Smartphone Application** has a simplistic interface, with just two buttons. One button begins and the other one ceases the data collection. Clicking on the "Start Trip" button generates a random numeric identifier, which remains constant through a session. Unlike sensors that measure traffic only from static locations, this identifier will enable a traveler's path to be re-constructed because of its association with a sequence of data points that contain the traveler's time and GPS locations.

The application then runs a simple loop that periodically acquires the phone's longitude and latitude coordinates. The period or the time interval between the sampling of two data points denoted by *timeStep* is a configuration parameter of the application. A counter is initialized to zero and incremented each time the coordinates are sampled. The numeric values of longitude and latitude are concatenated to two separate comma delimited strings. When the counter reaches a preset threshold denoted by *dataPoints* in the code, the sequence of coordinates is transmitted to a database server through the Internet. After transmitting the current sequence, the application begins collecting a new one. This process repeats indefinitely or until the application is terminated by the user. The number of data points that the application must collect before transmitting the sequence is also a configuration parameter of the application.

We used Sencha Touch [6] and PhoneGap [7] to implement the smartphone application because professional developers at the University of Connecticut (UConn) recommended this combination of technologies, and offered technical support. Sencha Touch is a HTML5 mobile application framework to build apps for iOS, Android, BlackBerry, Windows Phone, and other devices. It exposes the native application programming interfaces (API) of various devices through its unified framework, hoping to achieve a high degree of platform independence. PhoneGap is an open source application container technology to create natively-installed applications for mobile devices with HTML, CSS, and JavaScript. It can output a binary application for deployment to a particular platform such as IPA file (iOS Application Archive) or an APK file (Android Package). Thus, Sencha Touch and PhoneGap support the development and deployment of programs that target multiple platforms with minimal effort.

The **Database Management System (DBMS)** uses MySQL database running on an Apache web server [8] to store the data collected by the smartphone application. The web server is hosted within the Department of Civil & Environmental Enginerring at UConn. This server resides behind the School of Engineering (SoE) firewall under the authority of ECS and is password protected to prevent unauthorized access. Both MySQL database and Apache web server are a part of XAMPP [9], which is an open source Apache distribution containing MySQL, PHP (PHP: Hypertext Preprocessor, formerly Personal Home Page) [10], and Perl. MySQL supports database creation and standard structured query language (SQL) commands to dynamically insert, update, and delete data. The Apache Project develops and maintains an open source HTTP server for operating systems, including UNIX and Windows NT.

The SQL table to store traffic data has four fields:

1. **vehicle_ID**: A numeric identifier of 10 digits, generated by the smartphone application and common to all data points in a single trip.

2. **t_stmp**: The timestamp at which a longitude/latitude pair was sampled by the smartphone.

3. **lon**: Longitude of a data point.

4. **lat**: Latitude of a data point.

We note that the smartphone only collects and transmits the latitude and longitude coordinates of the phone. We can convert these latitude and longitude sequences into the distance or displacement between successive points. For example, GeoDataSource (http://www.geodatasource.com/developers) provides source code to perform this conversion in 12 different programming and scripting languages. These displacements can be used to compute acceleration and deceleration.

A **PHP Web Page** was implemented and deployed on the server hosting the MySQL database to transmit data from the smartphone application to the DBMS. This PHP web page contains a form and code to process the data submitted through the form. Figure 1 shows the form, which creates a channel for the data as it is transmitted from the smartphone application to the MySQL database. The smartphone appli-

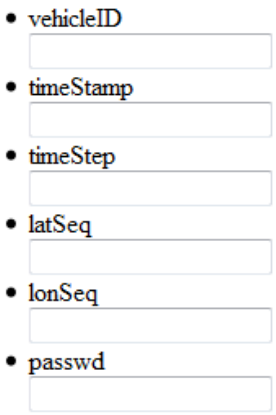

Figure 1. PHP Form for Data Channel

cation automatically populates the fields shown in Figure 1 to the form. *vehicleID* is the numeric identifier, *timeStamp* is the time at which the initial data point in a sequence was collected and *timeStep* is the constant interval between the data points. *lonSeq* and *latSeq* are comma delimited strings of signed double precision values representing the longitude and latitudes tuples. *passwd* is a hidden field that does not display on the web page. The password is embedded in the application to ensure that the data is in proper format before it is inserted into the database. This also prevents users from injecting garbage data. The PHP web page processes the data posted through the form only after authentication. It then dynamically constructs an SQL command to insert the sequence into the DBMS. Figure 2 shows how the components of the system communicate during a typical session of the application. Once the application is started, the traveler's location is sampled periodically and recorded. After collecting a certain number of data points, the application transmits this sequence to the database server via the PHP enabled web page, which builds a sequence of SQL queries that insert the location and time data. The database supports queries for analysis and modeling.

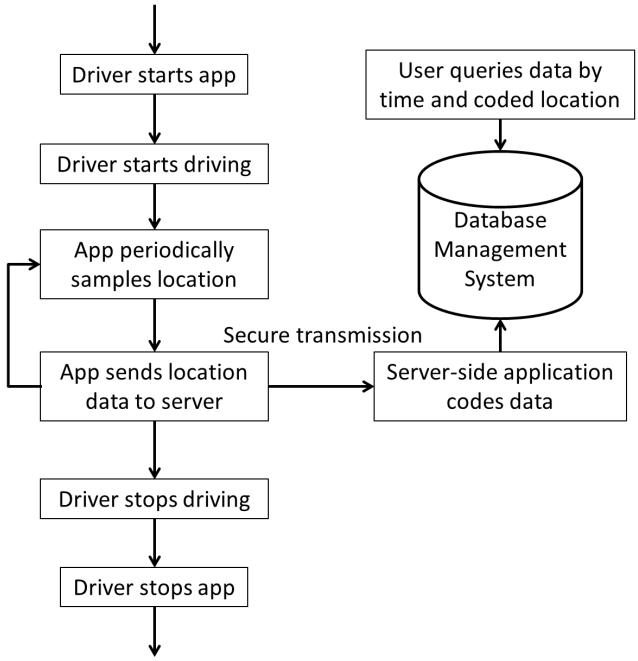

Figure 2. Communication Between Components of Smartphone System

2.2 Architectural Properties

In this section, we outline our design choices to achieve key non-functional attributes.

Performance is concerned with the timely delivery of the data. Sending each latitude-longitude pair as it is sampled, however, will be inefficient and drain a phone's battery rapidly. On the other hand, transmitting data in batches sacrifices the timeliness of delivery, which can be vital to predict congestion in real time. Thus, performance and energy efficiency conflict, and we balance these two competing goals by grouping a series of measurements and then transmitting this entire sequence.

Transmitting a group of measurements also reduces the number of bits transferred. This is because the application acquires the location data at fixed time intervals, and hence, the timestamps of the entire sequence of coordinates contained in the group can be inferred based on the initial one. Obviously, the savings in bits realized by this choice increases with the number of coordinate tuples grouped together in a sequence. Nevertheless, the size of the sequence transmitted grows with the number of tuples. The extreme case will involve transmitting the entire sequence at once at the end of the trip. However, this deferred delivery will delay the prediction of congestion, and in some cases even make it irrelevant. Moreover, transmitting a large group of data can be unreliable, if bit error rates cause parts or the entire sequence to be lost.

Reliability ensures that the data is delivered to the server without loss or corruption. Software and application fail-

ures, and poor wireless coverage can cause loss of data. Of these, failures of the wireless channel can be tolerated despite intermittent availability, which may often occur in locations such as underground tunnels with weak GPS signals. To prevent data loss from poor coverage, the application buffers the collected sequences on the phone, and then transmits these when the communication is re-established.

The application promotes **Safety** through simplicity of design that renders it non intrusive. It also directly encourages safe driving by explicitly instructing the user to start the application before and discouraging its use while driving. Thus, the application collects and transmits data in the background without active involvement from the user.

Privacy and security is achieved by not collecting any Personally Identifiable Information (PII) such as the username and password or the hardware identifier of the phone. The application captures the dynamics of each trip by an individual, but generates a new identifier per trip. This eliminates the need to collect PII without sacrificing the per-trip details. To ensure the security and privacy of the data during transit, the Advanced Encryption Standard (AES) [11], from the U.S. National Institute of Standards and Technology (NIST), was incorporated through Sencha Touch.

3 Experimental Study

This section details an experiment conducted by deploying the smartphone application on the Android phone of an undergraduate student. With this phone, she rode on the Orange Line bus route around the UConn campus. The random identifier for the trip was 7225472983 and consisted of 156 latitude-longitude tuples. Her trip commenced at 9:32:17 AM at $41°81'32.5792"$N $-72°24'45.6682"$W and concluded at 9:44:41 AM at $41°80'55.7682"$N $-72°24'86.3101"$W. The size of the sequence or the number of data points grouped into a single transmission was set to one. Thus, the data points were transmitted as they were sampled.

Table 1 lists the absolute time, and the latitude and the longitude coordinates of the first six points collected during this ride. The table also reports the delay between the two points, although this information is not explicitly transmitted and stored in the database.

Table 1. Sample Sequence of Trip Data

Time	Delay	Latitude	Longitude
9:25:41 AM	N/A	41.80562405	−72.24867839
9:25:46 AM	0 : 00 : 05	41.80553366	−72.24848601
9:25:51 AM	0 : 00 : 05	41.80553954	−72.24845334
9:26:01 AM	0 : 00 : 10	41.80556924	−72.24842785
9:26:06 AM	0 : 00 : 05	41.80568999	−72.24822567
9:26:12 AM	0 : 00 : 06	41.80595009	−72.24786536

Figure 3 shows the route with green dots, where each dot denotes a point where data was collected, transmitted and stored at the server. This data collection route closely approximates the bus route as shown in Figure 4. We note that

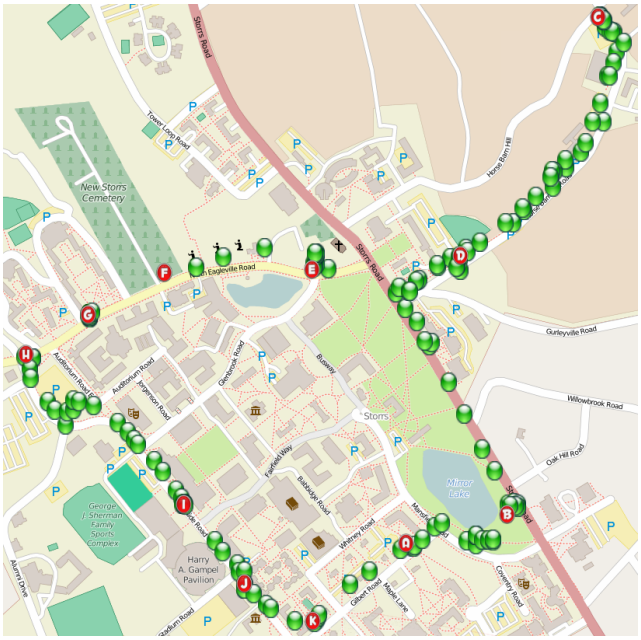

Figure 3. Data Collection Points Along Route

Figure 4. Orange Line Bus Route

the delay between successive coordinates varies slightly during the entire data collection duration. One reason for this variation is weak GPS reception at the phone. For example, the missing data point at 9:25:56 AM in Table 1, may

have been caused by an intermittent loss of signal. Also, the six-second delay between the last two data points in Table 1 may be caused by multiple applications running simultaneously and context switching on the smartphone, leading to non-uniform intervals.

We estimated the speed empirically by computing the distance between two successive geo-coordinates and dividing it by the number of seconds between the two points. The resulting speed is in miles per second, which we convert to miles per hour. We computed a moving average for the i^{th} speed over five points because we found that it was large enough to eliminate some noise, but small enough to prevent excessive smoothing of the data. Excessive smoothing is not desirable because it makes the moving average less responsive to variations, thereby making it harder to detect sudden changes. Figure 5 shows a plot of the moving average of the

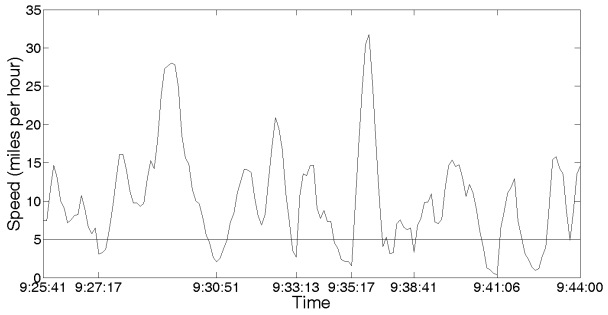

Figure 5. Moving Average of Trip Speed Data

estimated speed of the bus throughout the trip. The straight line in Figure 5 shows a constant speed of five miles per hour for the entire duration of the trip. The moving average of the speed fell below this value nine times. We note that the moving average eliminates speed estimates very close to zero because these are points immediately before a stop and correspond to the bus slowing down. Points immediately after the bus stop correspond to values when the bus was accelerating. We analyzed these local speed minima manually and discovered a strong correlation between the bus stops and the corresponding geo-locations. For example, the trip began at the Arjona stop north of the large lake in Figure 4, marked with a red letter A in Figure 3. The stop B at 9:27:17 was associated with the Shippee stop. The bus did not stop again until 9:30:51 (C) when it reached the Horse Barn Hill Arena at the southern most stop in Figure 4. Subsequent slow downs from 9:33:13 onward include the Young Building (D) Inbound stop, the stop light south east of the classics (CLAS) classroom building (E), slowing down at cross walk (F), a long delay at the North Campus stop at 9:37:34 (G), where many students cross the street from dorms to classroom buildings on the other side. The remaining four slow downs from 9:38:41 include the intersection of North Eagleville Road and Hillside Road (H) at the top right corner of Figure 4 (left of Figure 3), followed by the Field House (I), Co Op (J), and Alumni stops (K).

4 Related Research

This section compares our work to the conventional and modern approaches to studying transportation networks.

4.1 Conventional Approaches

Conventional approaches that explore transportation data [12] have built tools for examining historical incidents such as accidents. These tools analyze periodically updated archived data and separate data collection from exploration. However, this separation makes it difficult to glean detailed insights on the impact of these incidents on the network dynamics. For example, vehicle crash data often reports time, place, property damage, and fatalities but offers no information on the resulting delay or on how drivers rerouted themselves in response to the incident. In comparison, real-time data collection via smartphones can facilitate greater insights into how traffic changes with time, and how congestion develops and clears. Moreover, the impact of accidents, construction, speed and traffic volume on congestion can be predicted.

State-of-the-art transportation models [13, 14] employ stochastic techniques to estimate the utilization of roads considering travel demand and driver behavior, which is captured in terms of probabilistic selection of routes. Current congestion prediction models use demand data inferred from surveys or fixed sensors. This data includes link volume but not the trends such as acceleration, deceleration, and lane changes that can also indicate congestion. Smartphones can provide richer data to develop detailed models. Because data collection is automated, these models can be validated by collecting data from the same sites. Finally, predictive accuracy of these models can be assessed by collecting data on roads with similar characteristics.

In summary, automated data collection, integrated with exploration and modeling can offer several benefits. First, it can facilitate near real-time monitoring and detection of problems so that alerts can be issued to exercise caution around congestion or an accident and to offer guidance to emergency response officials. Enhanced congestion models can also assess how safety improvements such as new traffic lights impact the roads around the modified location.

4.2 Modern Approaches

Contemporary approaches have used the smartphone technology for the collection of transportation data. Feng *et al.* [1] acquired measurements from GPS-equipped vehicles and developed analytical models to optimally place probe vehicles to minimize the travel time prediction error. Another recent study [2] implemented virtual trip lines, where smartphones collected the location and speed of vehicles as they cross these lines. Lin [3] describes an Android smartphone application called the Toronto Buffalo Border Wait Time (TBBW) to share the waiting time among travelers

on the three Niagara Frontier border crossings. Davami *et. al.* [4] describe Kpark, a crowdsourced mobile application to monitoring parking availability on a university campus. Nandan *et al.* [5] identified common challenges in using crowdsourcing for public transportation, and implement an application for demand estimation and next arrival time.

The limitations of these approaches include: (i) studying only specialized transportation networks such as highways [1] and university campuses [4]; (ii) (still) collecting traffic data from fixed, static locations, similar to phyisical sensors [3, 2]; or (iii) considering only one mode, for example public transportation [5]. Our general-purpose smartphone-based monitoring system, however, can collect dynamic vehicle measurements on a continum, across the entire web of roadways within a transportation network. It can measure the dynamics of interacting public and private modes including pedestrian walkways and mass transit, facilitating an integrated study.

5 Conclusions and Future Research

This paper describes preliminary results from our efforts to engineer a smartphone-based system to crowdsource the collection of transportation data. An Android smartphone application for gathering user location information was developed and integrated with a server designed to receive, process, and store this data. Experimenting with data collection using the application around UConn campus, and subsequent example analysis of the collected data suggests that our approach can potentially collect richer data sets through the entire transportation network to test the validity of existing models and to develop new ones. Our future work includes enhancements to the application to minimize the involvement of users in data collection. Extensive data gathering experimentation using the application and subsequent analyses are also planned.

Acknowledgments

The authors would like to thank Tiffany and Linda Hoang, Orlando Echevarria and Louis Herman for their help with the smartphone application. This paper was supported by the New England University Transportation Consortium $USDOT/MIT - 5608530$.

References

[1] W. Feng, A. Bigazzi, S. Kothuri, and R. Bertini, "Freeway sensor spacing and probe vehicle penetration," *Transportation Research Record: Journal of the Transportation Research Board*, vol. 2178, no. 1, pp. 67–78, 2010.

[2] J. Herrera, D. Work, R. Herring, X. Ban, Q. Jacobson, and A. Bayen, "Evaluation of traffic data obtained via gps-enabled mobile phones: The mobile century field experiment," *Transportation Research Part C: Emerging Technologies*, vol. 18, no. 4, pp. 568–583, 2010.

[3] L. Lin, "Data science application in intelligent transportation systems: An integrative approach for border delay prediction and traffic accident analysis," PhD, SUNY at Buffalo, Buffalo, NY, 2015.

[4] E. Davami and G. Sukthankar, "Improving the performance of mobile phone crowdsourcing applications," in *Proc. of Intl. Conf. on Autonomous Agents and Multiagent Systems*, May 2015 (to appear).

[5] N. Naveen, A. Pursche, and X. Zhe, "Challenges in crowdsourcing real-time information for public transportation," in *Proc. of Intl. Conf. on Mobile Data Management*, Jul. 2014, pp. 67–72.

[6] A. Kumar, *Sencha Touch Cookbook: Over 100 Recipes for Creating HTML5-based Cross-platform Apps for Touch Devices*. Birmingham, UK: Packt Publishing, 2011.

[7] Adobe Systems, "Phone gap," *phonegap.com/*, Last accessed March 18, 2015.

[8] The Apache Software Foundation, "Apache HTTP server project," *http://httpd.apache.org/*, Last accessed March 18, 2015.

[9] Apache Friends, "XAMPP Apache + MySQL + PHP + Perl," *https://www.apachefriends.org/*, Last accessed March 18, 2015.

[10] A. Tarr and W. Mostrey, *PHP and MySQL 24-hour Trainer*. Indianapolis, IN: Wrox/John Wiley & Sons, 2012.

[11] J. Daemen and V. Rijmen, *The Design of Rijndael: AES–the Advanced Encryption Standard*. Berlin: Springer, 2002.

[12] M. Pack, K. Wongsuphasawat, M. VanDaniker, and D. Filippova, "Ice - visual analytics for transportation incident datasets," in *Proc. of the Intl. Conf. on Information Reuse & Integration*, Las Vegas, NV, Aug 2009, pp. 200–205.

[13] H. Mahmassani, J. Dong, and B. Park, "Existing traffic prediction and estimation models and systems: Review and summary," Northwestern University, Tech. Rep. US DOT/FHWA DTFH61-06-D-00005, 2008.

[14] B. Li, "Recursive estimation of average vehicle time headway using single inductive loop detector data," *Transportation Research Part B: Methodological*, vol. 46, no. 1, pp. 85–99, 2012.