STATE OF CALIFORNIA • DEPARTMENT OF TRANSPORTATION
TECHNICAL REPORT DOCUMENTATION PAGE
TR0003 (REV 10/98)

1. REPORT NUMBER	2. GOVERNMENT ASSOCIATION NUMBER	3. RECIPIENT'S CATALOG NUMBER
CA18-2652		

4. TITLE AND SUBTITLE	5. REPORT DATE
Development of an Adaptive Control Algorithm for Arterial Signal Control	03/31/2018
	6. PERFORMING ORGANIZATION CODE
7. AUTHOR	8. PERFORMING ORGANIZATION REPORT NO.
Zahra Amini, Michael Mauch and Alexander Skabardonis	
9. PERFORMING ORGANIZATION NAME AND ADDRESS	10. WORK UNIT NUMBER
California PATH program Institute of Transportation Studies University of California, Berkely 409A McLaughlin Hall, MC 1720 Berkeley, CA 94720-1720	11. CONTRACT OR GRANT NUMBER IA 65A0543
12. SPONSORING AGENCY AND ADDRESS	13. TYPE OF REPORT AND PERIOD COVERED
California Department of Transportation Division of Research Innovation, and System Information MS-83 1727 30th Street Sacramento CA 95816	final report 1/1/2015 - 6/30/2017 14. SPONSORING AGENCY CODE

15. SUPPLEMENTARY NOTES

16. ABSTRACT

An optimization algorithm was developed to optimize the signal offsets along arterials in real-time based on volume data from loop detectors. The algorithm performance was tested with the VISSIM microscopic model on three real-world arterials. The algorithm generated offsets had better performance than the offsets generated using the widely used SYNCHRO optimization tool. The real-time update of offsets generated by the algorithm was also evaluated using a custom made simulation testbed. The results indicate modest improvements in traffic performance over the SYNCHRO timings. A data filtering algorithm was developed to estimate the systematic error of detector data. Application of the algorithm at a real world intersection with multiple detectors resulted in significant error reduction on high volume approaches. A clustering algorithm was also applied to the intersection detector data to determine both the traffic demands for developing TOD signal timing plans, and the associated plan switching times. The new timing plans outperformed the conventional TOD plans based on traffic volumes obtained at predetermined time periods.

17. KEY WORDS	18. DISTRIBUTION STATEMENT
Traffic signals, mathematical models, optimization simulation, coordination, offsets, detectors	No restrictions.

19. SECURITY CLASSIFICATION *(of this report)*	20. NUMBER OF PAGES	21. COST OF REPORT CHARGED
Unclassified	44	

DISCLAIMER STATEMENT

CALIFORNIA PATH PROGRAM
INSTITUTE OF TRANSPORTATION STUDIES
UNIVERSITY OF CALIFORNIA, BERKELEY

Development of an Adaptive Control Algorithm for Arterial Signal Control

Zahra Amini
Michael Mauch
Alexander Skabardonis

California PATH Research Report
UCB-ITS-PRR-2018-01

This work was performed as part of the California PATH program of the University of California, in cooperation with the State of California Business, Transportation and Housing Agency, Department of Transportation, and the United States Department of Transportation, Federal Highway Administration.

Final Report for Agreement 65A0543

March 2018

CALIFORNIA PARTNERS FOR ADVANCED TRANSIT AND HIGHWAYS

ABSTRACT

An optimization algorithm was developed to optimize the signal offsets along arterials in real-time based on volume data from loop detectors. The algorithm performance was tested with the VISSIM microscopic model on three real-world arterials. The algorithm generated offsets had better performance than the offsets generated using the widely used SYNCHRO optimization tool. The real-time update of offsets generated by the algorithm was also evaluated using a custom made simulation testbed. The results indicate modest improvements in traffic performance over the SYNCHRO timings. A data filtering algorithm was developed to estimate the systematic error of detector data. Application of the algorithm at a real world intersection with multiple detectors resulted in significant error reduction on high volume approaches. A clustering algorithm was also applied to the intersection detector data to determine both the traffic demands for developing TOD signal timing plans, and the associated plan switching times. The new timing plans outperformed the conventional TOD plans based on traffic volumes obtained at predetermined time periods.

Keywords:

Traffic signals, mathematical models, optimization simulation

ACKNOWLEDGEMENTS

This work was performed by the California PATH Program at the University of California at Berkeley, in cooperation with the California State Transportation Agency, Department of Transportation (Caltrans), Division of Research, Innovation and Systems Integration (DRISI) under the Interagency Agreement #65A0543. The contents of this report reflect the views of the authors, who are responsible for the facts and the accuracy of the data presented herein. The contents do not necessarily reflect the official views or policies of the State of California.

The authors thank John Slonaker of Caltrans DRISI for his support and advice during the project. We would like to thank the project technical advisory committee members Ted Lombardi, Kai Leung, and Jorge Fuentes of Caltrans for their comments and suggestions throughout the study.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1
INTRODUCTION

1.1 Problem Statement

Considerable attention has been given to new approaches for improving the transportation system because of limited funding and environmental concerns for constructing new highway facilities. One promising approach is optimization of signal timing plans along arterials. This would reduce unnecessary delays and stops at traffic signals, improve travel times and cut fuel consumption and emissions. In many instances these arterial facilities also serve as reliever routes for congested freeways especially under incident conditions. Signal timing optimization is an important part of the Integrated corridor management (ICM) approach that focuses on the coordinated management of urban networks with adjacent highway facilities (e.g., nearby parallel freeways), that comprise travel corridors [1].

Traffic signals operate under specific timing plans (cycle length, and green times) to provide the right of way to conflicting traffic movements. A significant number of traffic signals operate as actuated and are able to respond to cycle-by-cycle variations in traffic volumes to better assign green times to conflicting movements. Most of the existing signal systems use fixed time timing plans prepared off-line based on historical data on traffic demand, and implemented by time of day (TOD), e.g., am, midday and pm peak periods. Fixed-time plans, however, cannot deal with the variability of traffic patterns throughout the day, and they become outdated because of the traffic growth and changes in traffic patterns.

Traffic responsive and adaptive traffic control systems update the timing plans based on data from detectors located on each intersection approach and travel lane. Traffic responsive systems adjust the signal settings at short time intervals based on detector data. Adaptive control policies use measured and predicted traffic data to continually optimize the signal settings. However, the implementation of adaptive systems is limited because of the complexity and the extensive and costly data requirements [2]. The adaptive control software Light (AVS-Light) [3] was developed by FHWA as a cost effective solution for adaptive control by directly utilizing data typically available from detectors in coordinated actuated signals to optimize the splits and offsets.

1.2 Objectives of the Study

The objective of this project is to develop and evaluate an adaptive control algorithm for optimizing the signal settings on arterial systems operated by the California Department of Transportation (Caltrans). The algorithm performance will be evaluated through simulation on real-world arterial systems.

Emphasis is placed on developing an algorithm that optimizes the signal offsets based on recent platoon arrival data at the intersection as characterized by advanced detectors. The study will also develop an algorithm that will be capable of accommodating detector errors. Furthermore, we will explore the availability of detector data to determine volumes for developing TOD timing plans and associated switching times.

1.3 Organization of the Report

This document is the final report for the study. It describes in detail the research performed and presents the findings and recommendations. The next Chapter describes a methodology for selecting TOD plans based on detector data. Chapter 3 describes the algorithm developed for identifying detector errors. The proposed offset optimization algorithm and the results from the simulation testing on three real-world arterials are described in Chapter 4. Chapter 5 presents the approach for implementation in a typical arterial control system. The final Chapter summarizes the study findings and discusses future research directions.

CHAPTER 2

IMPROVING SIGAL LTIMIG PLANS USING HR DATA

2.1 Introduction

High resolution (HR) data at signalized intersections refers to the continuous acquisition of detector data that provide information on the vehicles approaching and leaving the intersection plus simultaneous information on the signal status. This information can be obtained from multiple sensors and/or a signal controller with a high definition data logger [4,5]. Figure 2.1 shows an example of instrumentation for HR data at a signalized intersection in the city of Danville, CA. Each approach has stop bar detectors denoted by grey circles, and an advance detector (not shown). In addition there is a detector in each departure lane, denoted by red circles, which permits measurement of turn movements. The controller has a signal monitoring card that provides information about the signal phase. The vehicle detectors are magnetometers that send the detection events wirelessly to the Access Point (AP). The AP also receives the signal phase events. The AP time stamps all events with an accuracy of 10ms. The event stream is sent via a cellular modem to a server. The intersection is also equipped with a PTZ camera, used to verify the accuracy of the data collection process.

HR data are increasingly being used in deriving signal performance measures (delay, proportion of vehicles arriving during the green) [6] and in data driven timing plan development. The typical approach for developing fixed time of day plans is to collect turning movement counts for certain times of the day (typically AM, midday and PM peak) and determine the signal settings to minimize delay, stops or other chosen performance measures. These timing plans may not be optimal for other time periods or other days, because they don't account for the variability of traffic demand. Procedures have been proposed to develop robust timing plans to account for demand as well as supply variability but the high costs of collecting the required field data on turning movement counts makes most proposed approaches impractical [7]. The availability of continuous HR resolution data provides the opportunity to develop robust timing plans without additional costs and more importantly assess their performance.

The following section describes the methodology for determining TOD signal timing plans based on HR data, and the application at a real-world signalized intersection.

2.2 Application of HR data: Improving Signal Timing Plans

We collected and analyzed HR data at a signalized intersection in Beaufort, South Carolina. The Intersection has four approaches and there are three movements per approach. The sensor instrumentation includes upstream detectors on each travel lane located approximately 200 to 300 ft upstream from the stop-line, and stop bar detectors. Also, departure lane detectors are located on each lane downstream of the stop-line. There are protected left-turn phases on the main road, and split phasing on the cross streets. The signal operates as coordinated traffic actuated with fixed-time of day timing plans. There are a total of three timing pans for AM, midday and PM peak on weekdays.

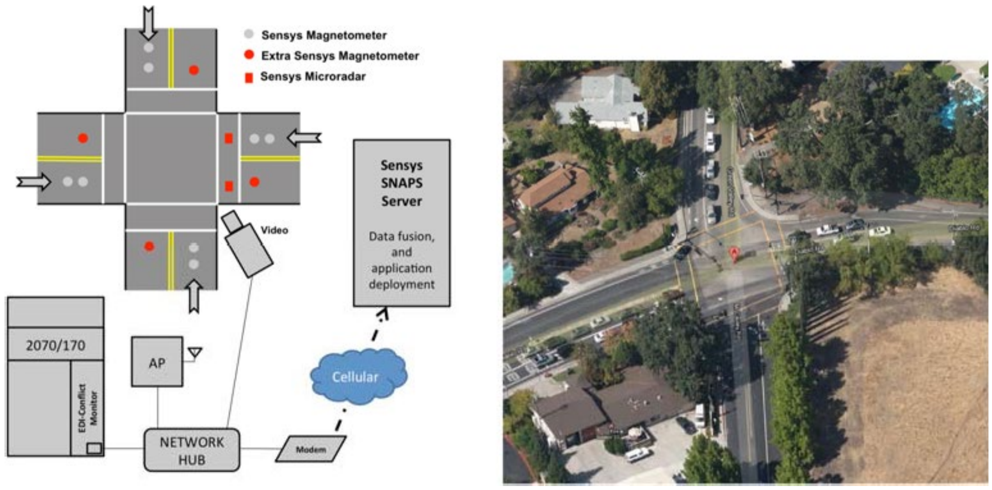


Figure 2.1 Sample Instrumented Intersection

Figure 2.2 shows the total 5 minute traffic volumes at the test intersection over the 12 hour analysis period. We use this information to group (cluster) the data into periods of similar conditions for developing the signal timing plans instead of predetermined time periods.

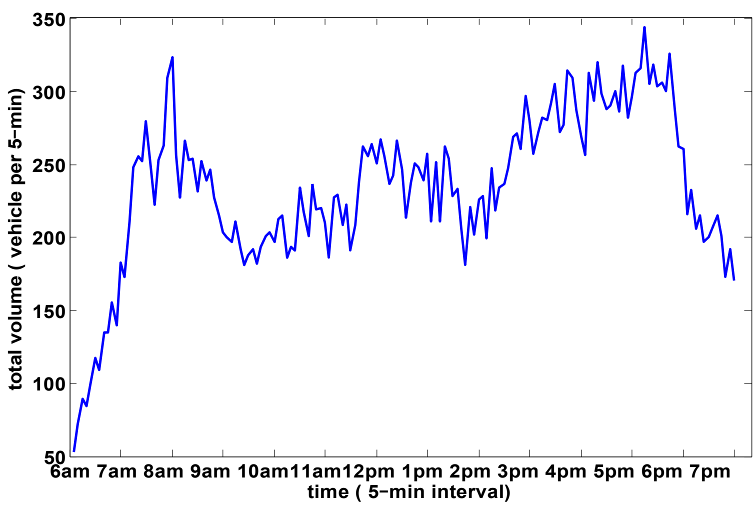


Figure 2.2 Total Traffic Volume at Test Intersection (#veh/5-min)

We used a K-means clustering method which partitions n observations (traffic volumes) into k clusters in which each observation belongs to the cluster with the nearest mean. The objective is to minimize the distance of every item in each cluster from the center of the cluster, as is shown in the following equation:

$$\underset{S}{\operatorname{argmin}} \sum_{i=1}^{k} \sum_{x \in S_i} \| x - \mu_i \| \qquad (2\text{-}1)$$

Where, x presents each point from group S_i.

In our case we have K=3 clusters because we have three timing plans. The process consists of the following steps: a) estimate the center of each group, b) calculate the distance of each point from the center points, c) put each point in the group with the least distance from the center, d) go back to step (b) and continue this process until no point leaves its group in step (c). The points in each cluster obtained from the application of the method are shown in Figure 2.3.

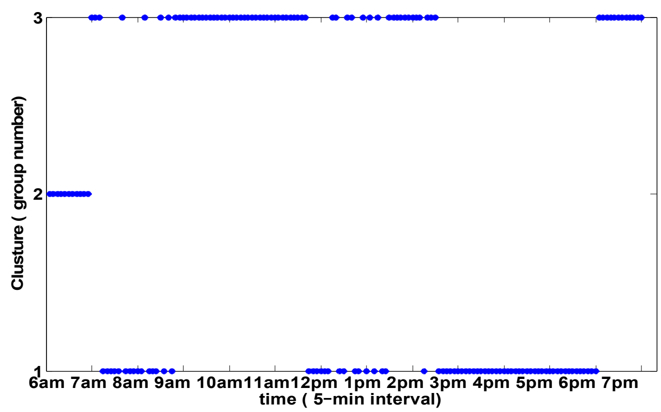


Figure 2.3 Clustering of Traffic Volume Data

We developed three signal timing plans using the clustered volumes, using the widely used SYNCHRO signal optimization software [8]. Figure 2.4 shows a comparison of existing and new timing plans. It can be seen that the proposed timing plans improved the intersection delay by 10% on average over the existing signal timing plans. As was expected the higher benefits were obtained during the time periods of the beginning and dissipation of the designated peak periods (e.g., 7 and 9 AM and 4 and 6 PM).

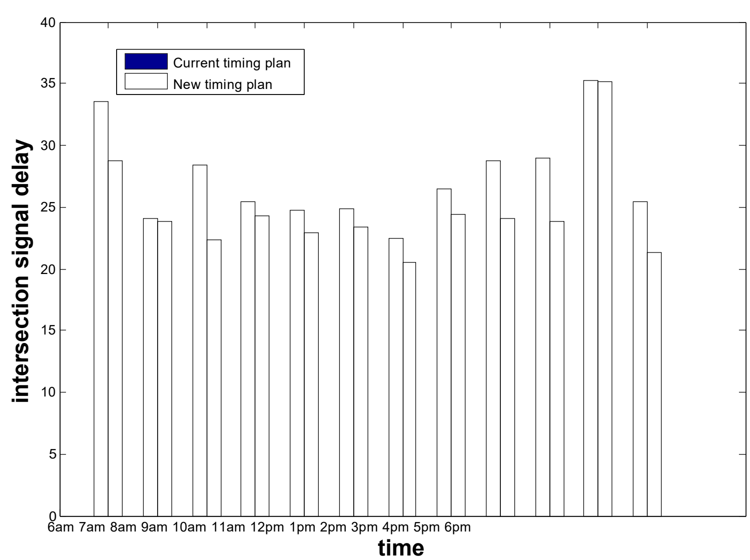


Figure 2.4 Intersection Delay: Existing vs. Proposed Timing Plans

CHAPTER 3
DETECTOR DIAGNOSTICS

3.1 Introduction

HR data are very useful in the assessment of operating conditions and the development and implementation of improved signal control strategies. However, detector failures cause missing and incorrect HR data that significantly affect the effectiveness of the traffic control system in place.

There are several types of detector data errors. Data losses occur because of equipment failures (e.g., broken loops) and communication failures. Incorrect calls cause unwarranted signal phase activations or extensions. HR data are also subject to systematic errors (e.g., undercounting or overcounting) because of detector tuning and environmental reasons. Existing procedures for detecting detector errors at signalized intersections mostly check only for state changes at the detector due to a vehicle passage and communications to the signal controller [9].

Most of the studies on detector diagnostics focus on loop detectors on freeways. A number of studies compare the detector data against thresholds (e.g., minimum and maximum vehicle occupancies) to verify the accuracy of the data [10,11]. Other approaches applied statistical techniques (historical average, nearest-neighbor algorithm, and autoregressive integrated moving average models) [12], as well as and Fourier transform-based techniques and genetic algorithms [13] to detect detector data errors. Most of these methods test the data at each time interval (30 seconds) with is computationally very hard for large detector systems. The data filtering algorithm uses in California PeMS (Performance Measurement System) [14] uses data from the entire day.

This Chapter describes the development and application of an algorithm for detecting detector errors. The emphasis in developing the algorithm is to correct for systematic errors of detectors that have implications for signal control strategies.

3.2 Research Approach

We developed an algorithm to estimate the systematic error of the detectors based on the flow conservation principle. At any time interval, the total number of the vehicles that enter the intersection is equal to the sum of the vehicles that exit the intersection. Our approach has several advantages over the above mentioned methodologies. Most importantly, it could be used in real time and it is possible to include the data-filtering algorithm in the data collection process. Also, it can be easily modified to work at intersections with different detector systems and geometrics. Finally, in addition to estimating the error value, it can substitute any missing value by using available data.

Figure 3.1 shows a typical signalized intersection with four approaches. We have 4 inputs, 4 outputs, and 12 movement flow values as is shown in Figure 3.1. The same figure also shows the location of the detectors. Advance detectors are located at each approach upstream of the intersection stopline and they measure the arriving (input) flows. The output flows are measured by the departure detectors located in the departure lanes, downstream of the intersection stopline. The stop-bar detectors located at the stop line, count the number of vehicles for each movement (right, left, through). For any time interval, we can obtain the traffic volume for each movement. Therefore, we will have 20 values in total and each value will have a different amount of error.

There are multiple sources of error in traffic data; some errors could be due to vehicle's movement such as lane change and driving over the lane, while some are due to environmental conditions such as rain. Incidents can also cause error in traffic data and interrupt the data collection process. Some detectors miscount the vehicles when they are very large or very small in dimension or when the vehicles are traveling too close to each other. In addition, there are systematic errors in some detector's data collection process that are hard to eliminate. Using different detectors to count the same vehicle several times would improve the counting process and provide accurate values. Figure 3.1 shows an example where each vehicle was counted three times and each measured value is equal to the true value plus or minus some error. The goal is to minimize the summation of all the error terms.

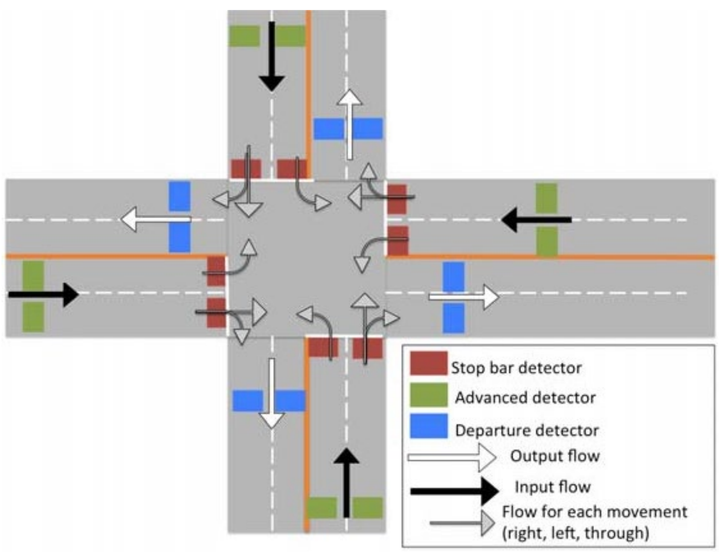


Figure 3.1 Typical 4 Legged Signalized Intersection

We assume that the measured flow value, $\boldsymbol{f}_i$, equals the estimated flow value, f_i, plus the error term, E_i (equation 3-1). A negative error term means the detected value is less than the actual value and the detector is undercounting. While a positive error term implies that the detector is over counting.

$$\boldsymbol{f} = \boldsymbol{f}_i + E_i \quad for\ all\ i \tag{3-1}$$

The estimated value f_i is not necessarily equal to the actual value because E_i only captures the amount of the systematic error and not all the errors that exist in the detected value. However, our claim is that the estimated value would be closer to the actual value compared to the detected value. In section 3.3 we will test this hypothesis by applying the algorithm on a real database with ground truth data available.

For the intersection shown in Figure 3.1, there are 20 variables. The corresponding cost function, equation (3-2), includes 20 error terms, one for each variable.

$$C = \sum_{all\, i} w_i * E_i^2 \qquad (3\text{-}2)$$

In the cost function C, we are summing the ε_i^2 values to include both positive and negative errors and making sure that they do not cancel each other out. Also the squared error is an appropriate fit for our cost function because it is differentiable. Moreover, each ε term has a corresponding ω_i value, which is the weight of each ε_i term and its value depends on the level of accuracy of f_i. In an equal situation ω_i would be the same for all the variables, but it is possible that we have more confidence in one or some detectors' performance than others. For example, it is reasonable to say that advanced detectors have a better accuracy in their counting process than stop-bar detectors, because of the lane changes and stops that happen at the stop lines. Based on available information about the detectors and previous studies we can assign an appropriate value for ω_i.

We have a set of constraints based on the flow conservation rule. Equation (3-3) requires the input flow in each leg to be equal the sum of turn-movement flows in the same leg. The next constraint in Equation (3-4) requires the output flow in each leg to equal the sum of turn-movement flows that are feeding the output direction. Finally, it is important to mention that all the detected and estimated flow values, f_i should be equal to or greater than zero.

$$f_{in} = \sum_{all\ movements} f_{movement} \qquad (3\text{-}3)$$

$$f_{out} = \sum_{all feeding\ movements} f_{movement} \qquad (3\text{-}4)$$

Ultimately, we have the detected flow values and we want to estimate the errors by minimizing C, which is a convex function, subject to linear constraints so we have a minimum point where the cost should be at its lowest value.

After formulating the cost function, assigning appropriate ω values, and inputting the detected flow values we can run the optimization problem. This process could be also done in real time. After minimizing C we get the e_i for each f_i value, then we can estimate f_i which is our imputed flow value. The time interval that we are running in the optimization needs to be long enough to minimize the impact of the incomplete trips, but at the same time it should be small enough so we can assume the traffic volume is uniform during each time step. Also, the cycle length can impact the length of the time interval.

3.3 Case Study

The proposed methodology was applied to estimate the detectors' systematic error in a real-world signalized intersection, Montrose Rd. and Tildenwood Dr., in Montgomery County, Maryland. Figure 3.2 shows the schematic of the detector layout for the test intersection. Each approach has stop bar detectors denoted by red rectangles, advance detectors denoted by green rectangles, and departure detectors denoted by grey rectangles. There is a network-monitoring card that provides the signal phase. The vehicle detectors are magnetometers that send the detection events wirelessly to an Access Point (AP). The AP also receives the signal phase events from the signal controller's split monitor. The AP time stamps all events with an accuracy of 10ms. The event data are sent via a cellular modem to a server. The synchronous data collection permits matching different vehicle movements and the signal phase(s) serving them.

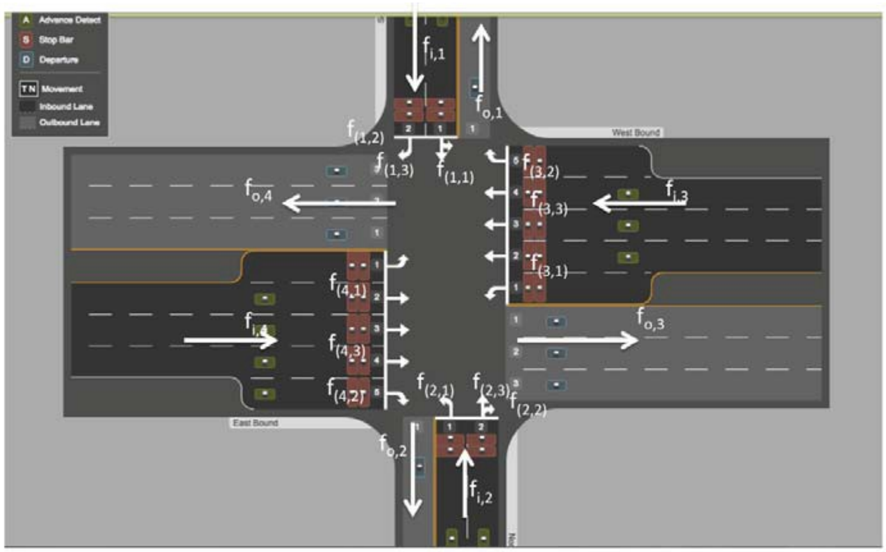


Figure 3.2 Test Intersection: Montrose Rd. and Tildenwood Dr.

The intersection has four approaches (legs) and there are three movements per approach. As shown in Figure 3.2, in each leg there are 3 different types of detectors and each one counts vehicles independently. Advanced detectors (located approximately 200-300 ft upstream from the stop-line) record the input values, which is shown by $f_{i,n}$ where n is the leg number. The output value, $f_{(o,n)}$ is recorded by departure detectors. On the same figure we can see the flows from the turn movements presented by $f_{(n,m)}$ where left, right, and through movement are respectively m=1,2, and 3. The intuition of our approach is to minimize the error corresponding to each variable. We have 4 inputs, 4 outputs, and 12 turn movement variables, which gives us 20 variables in total. The measured value f equals the estimated value f plus the error term, ε (Equations 3-5 through 3-7).

$$f_{(n,m)} = f_{(n,m)} + E_{f(n,m)} \quad \text{for all } (n,m), n = 1,2,3,4 \text{ and } m = 1,2,3 \tag{3-5}$$

$$f_{i,n} = f_{i,n} + E_{f_{i,n}} \quad \text{for all } n = 1,2,3,4 \tag{3-6}$$

$$f_{o,n} = f_{o,n} + E_{f_{o,n}} \quad \text{for all } n = 1,2,3,4 \tag{3-7}$$

The cost function in Equation (3-8) includes the square of all the error terms multiplied by the weights, a_n, $f3_n$, and $y_{(n,m)}$. As we mentioned in section 3.2, weights reflect confidence in the accuracy of the measurements. Flow values should be positive so we add a lower boundary condition, as shown in Equations (3-9) through (3-11). Also, we have a set of constraints based on the flow conservation rule, which shows each input count should equal the turn movement count of that approach and each output count is equal to the sum of all the movements that enter that approach (Equations 3-12 to 3-16). For example, in Equation 3-13 the output flow in leg 1 equals the right movement flow from leg 3 plus the left movement flow from leg 4 and the through movement from leg 1.

$$\min_{f_{i,n}, f_{o,n}, f_{(l,m)}} \sum_{n=1}^{4} a_n E_{f_{i,n}}^2 + \sum_{n=1}^{4} P_n E_{f_{o,n}}^2 + \sum_{n=1}^{4} \sum_{m=1}^{3} y_{(n,m)} E^2_{f_{(n,m)}} \quad (3\text{-}8)$$

$$\text{s.t. } 0 ::: f_{n,m} \text{ for all } (n, m) \quad (3\text{-}9)$$

$$0 ::: f_{i,n} \text{ for all } n \quad (3\text{-}10)$$

$$0 ::: f_{o,n} \text{ for all } n \quad (3\text{-}11)$$

$$f_{i,n} = \sum_{m=1}^{3} f_{(n,m)} \quad \text{for } n = 1, 2, 3, 4 \quad (3\text{-}12)$$

$$f_{o,1} = f_{(2,3)} + f_{(3,2)} + f_{(4,1)} \quad (3\text{-}13)$$

$$f_{o,2} = f_{(1,3)} + f_{(4,2)} + f_{(3,1)} \quad (3\text{-}14)$$

$$f_{o,3} = f_{(4,3)} + f_{(2,2)} + f_{(1,1)} \quad (3\text{-}15)$$

$$f_{o,4} = f_{(3,3)} + f_{(1,2)} + f_{(2,1)} \quad (3\text{-}16)$$

We implemented the data-filtering algorithm on the detector data for every 15 minutes starting at 6 AM until 7 PM on June 14, 2016 (a total of 52 time intervals). We set all the weights equal to 1 and assumed all detectors were in good condition and had the same level of accuracy. The ground truth turn movement flows collected manually by observers over the same time period (6AM to 7PM) for every 5 minutes were available for the same intersection and the same day. We aggregated the data to get the vehicle counts for every 15 minutes, and then compared it with the turn movement vehicle counts from the detectors and the counts obtained from the algorithm. Table 3.1 shows the average error value over the 13-hour period for each intersection leg separately for the major and minor movements. Major movements have very high traffic demand, while the minor approaches have minimal traffic volumes. The second column of the Table shows the percentage difference between the true and detected value, and the last column shows the percentage difference between the true and the algorithm estimated value.

The results show that the application of the algorithm can reduce the average error by 25% on the major movements. The results were mixed on the minor movements. Figure 3.3 shows the cumulative vehicle counts for one major movement obtained from the field measurements, the detectors and the algorithm over the 13-hour period. The proposed algorithm improves the accuracy of the detected flows over the entire analysis period. The improvements are larger under high volume conditions during the pm peak period as shown in Figure 3.4. This is significant for signal timing, because on high volume approaches, the differences between detected and true flows may be several vehicles per signal cycle which could affect the quality of timing plans.

The data filtering methodology can be easily adjusted for use with the configuration of stop bar and advance detectors typically employed by Caltrans. In this case we remove the equations corresponding to the departure detectors, and the error term for the departure detectors in the cost function. The algorithm checks that the input values from the advanced detectors and the output values from the stop-bar detectors are the same in each intersection approach.

Table 3.1 Comparison of Detected, Estimated, and Observed Values

Movement		Error for true vs. detected (%)	Error for true vs. estimated (%)
Major Movements	Leg 3 through	4%	3%
	Leg 4 through	6%	5%
Minor Movements	Leg 3 Right	6%	9%
	Leg 3 Left	-43%	-35%
	Leg 4 Left	-41%	-11%
	Leg 4 Right	-10%	7%
	Leg 2 Left	6%	2%
	Leg 2 Right	5%	-20%
	Leg 1 Left	-8%	-37%
	Leg 1 Right	7%	-19%
	Leg 2 through	-75%	115%
	Leg 1 through	-50%	198%

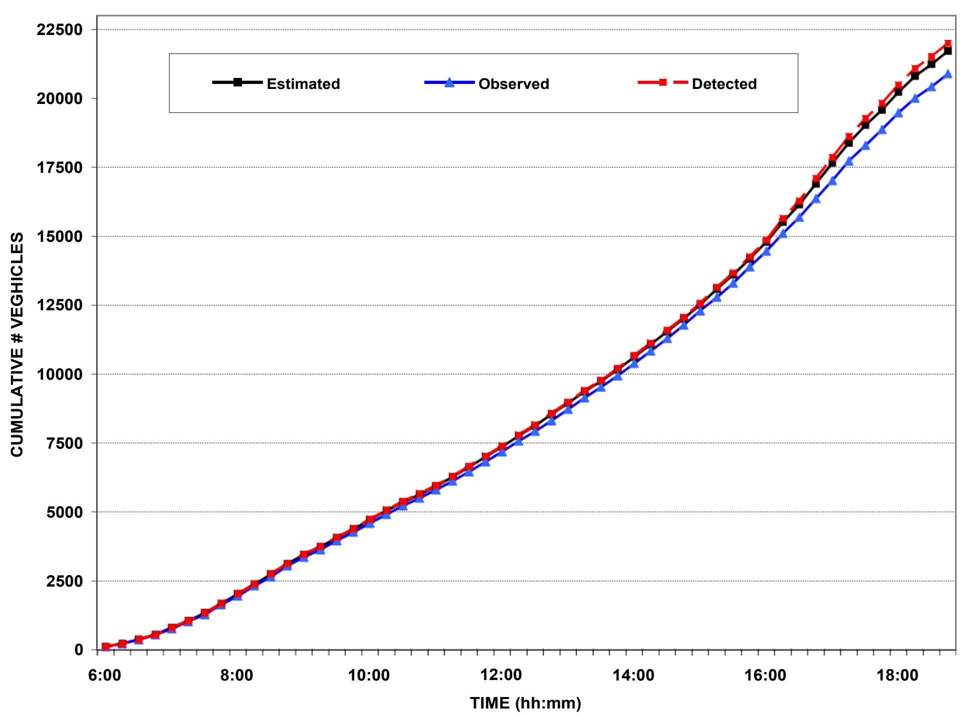


Figure 3.3 Cumulative Count Curves: Leg 3 Through Movement - 13hrTime Period

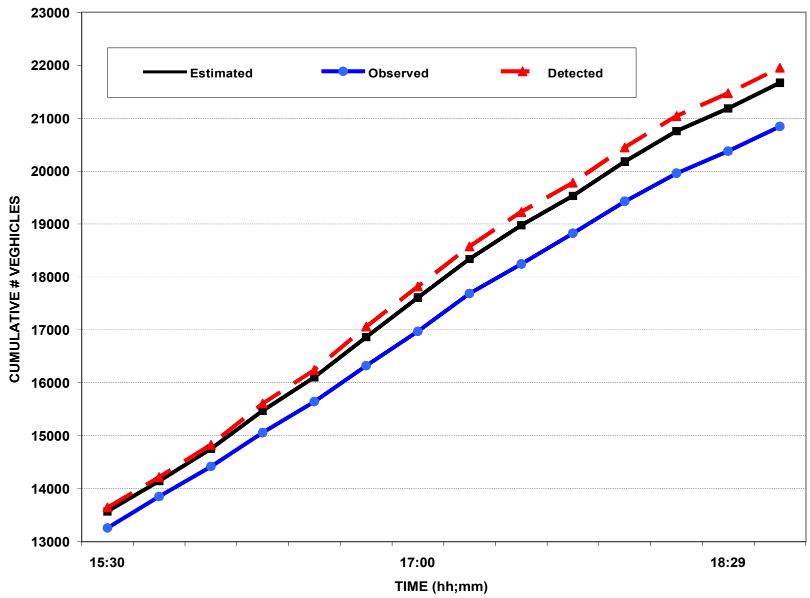


Figure 3.4 Cumulative Count Curves: Leg 3 Through Movement-Peak Period

On the minor movements, the algorithm produced inconsistent results with certain approaches having detected values closer to the true values compared to the imputed values (Table 3.1). We increased the weights ω_i of the minor movements, i.e., we assume that the detector detected vehicle counts are more accurate compared to the detected counts on major movements. The results shown in Figure 3.5 show that the algorithm calibration through the weights adjustments produced significant improvements to the minor movements.

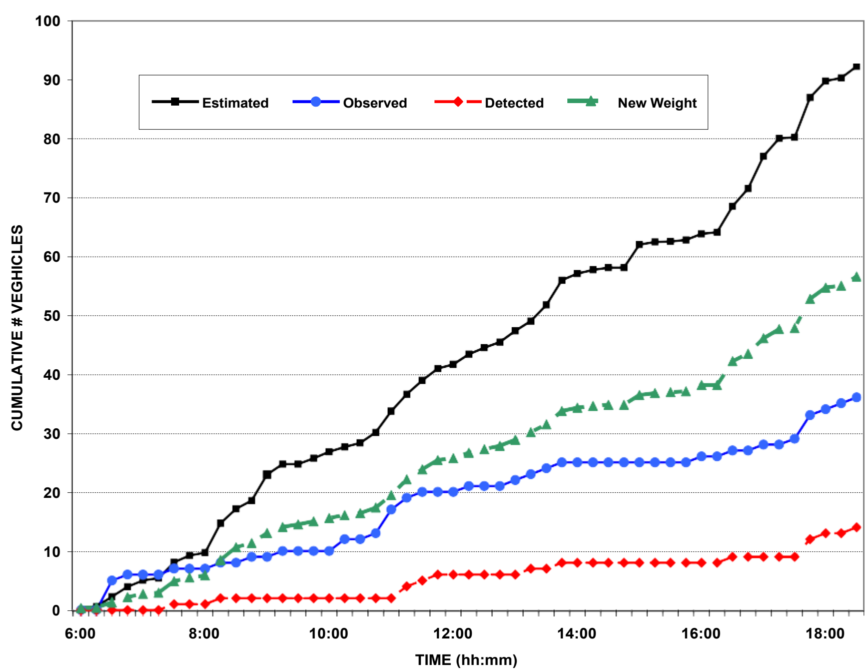


Figure 3.5 Cumulative Count Curves: Leg 2 Through Movement

CHAPTER 4.

PROPOSED OFFSET OPTIMIZATION ALGORITHM

4.1 Background

Traffic signals in urban arterials and networks operate as coordinated to facilitate the progression of major traffic movements. The signals operate on a common cycle length, typically the minimum cycle required to provide capacity at the critical intersection. The green times at each intersection are determined to minimize the approach delays subject to capacity and safety constraints (i.e., minimum times for pedestrians and vehicles). The offsets (the time for starting of the green light at the signal relative to adjacent signals) are the critical control variables for coordinated operation.

Existing offset optimization algorithms determine the offsets to maximize the green bandwidth, or to minimize a combination of delays and stops in the network. Green bandwidth is the portion of the green time that the through vehicles can travel along the arterial without stopping. MAXBAND [15] and PASSER [16] are arterial signal timing optimization tools that optimize the signal settings for maximum bandwidth. The TRANSYT model [17,18] has been the most widely used model to optimize the signal settings on arterial networks to minimize a combination of delays and stops. The SYNCHRO model optimizes the signal settings on arterials for minimum delay. SYNCHRO is the most widely used software for signal settings optimization largely because of the ease in data entry and visualization of data and results. Procedures also have been developed to determine signal settings for actuated controllers from optimized fixed-time plans [19].

These optimization tools assume fixed-time control and require historical data on traffic volumes for each signal approach to determine the signal settings, and they are not designed to make adjustments to the offsets to accommodate traffic volume variations based on real-time detector data. This Chapter describes the formulation and testing of a new offset optimization algorithm that can be used for real-time offset adjustments.

4.2 Proposed Signal Offset Optimization Algorithm

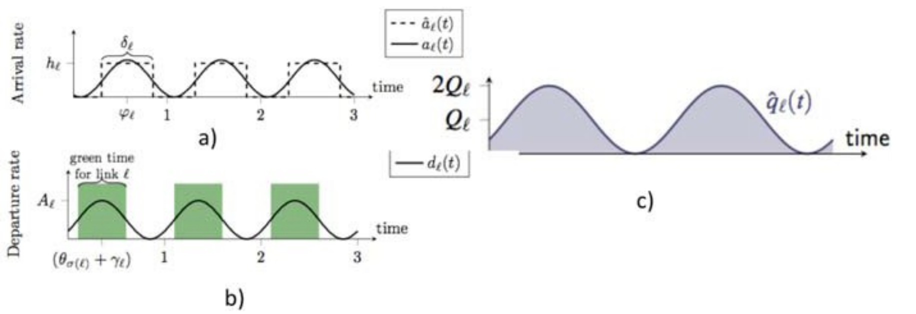
4.2.1 Formulation

The proposed optimization algorithm is an extension of an algorithm [20] that formulates offset optimization as a quadratically constrained quadratic program amenable to convex relaxation. The approach considers all the links in a network with arbitrary topology and is not restricted to a single arterial. The offset optimization problem is formulated as minimizing the queues at all intersections in the network, rather than of minimizing the sum of link delay functions. The flow of vehicles arriving at and deporting from a signalized intersection is approximated as sinusoidal functions of time. Figure 4.1 shows the sinusoidal arrival and departure of vehicles at the intersection approach in parts (a), and (b). Part (c) of the same figure presents the oscillation of average queue length.

The arrival of vehicles to the queue at time *t* on entry link *l* at signal *a(l)* is approximated by the function:

$$a_l(t) = A_l + a_l cos(wt - \varphi_l)$$

Where, A_l represents the average arrival rate of the vehicle to the queue, a_l is for periodic fluctuation in arrival rate, $<p_l$ is the offset of the periodic arrival of vehicles to the queue including the travel time on link l, and $w = 2rr/T$ is the cycle time of the signal in radians/second, when T is the cycle time in seconds, common to all intersections.



a) Vehicle arrival on non-entry links b) Vehicle departure over time c) Queue length over time

Figure 4.1 Sinusoidal Approximation of Arrivals and Departures

Similarly we can write a function for arrival of vehicles in non-entry links

$$a(t) = A_l + a_l cos(wt - (0_{r(l)} + <p_l))$$

Where, $0_{r(l)}$ is the offset at the arriving intersection. Unlike arrivals, the departure of the vehicles from the queue at time t in entry and non-entry links follows the following function:

$$_l(t) = A_l\ 1 + cos\ wt - (0_{\beta(l)} + y_l)$$

Where, y_l represents the green split offset of link l in radians and $0_{\beta(l)}$ is the offset at departure signal. If $q(t)$ is the approximate queuing process, then

$$Q^N{}_l(t) = a(t) - {}_l(t) = a_l\ cos(wt - 0_{r(l)} - <p_l) - A_l\ cos(wt - 0_{\beta(l)} - y_l)$$

Where, we can use Euler's equation and phasor to write

$$Q^N{}_l(t) = Q_l\ cos(wt - (_l)$$

For phase shift $(_l$ and amplitude Q_l, where

$$Q_l(0) = \sqrt{A_l^2 + a_l^2 - 2A_l a_l cos((0_{r(l)} + <p_l) - (0_{\beta(l)} + y_l))}$$

In order to minimize the sum of all the queues in all links in a set L for a set S of signalized intersections, $\sum Q_l$.

Since $A_l^2 + \alpha_l^2$ is constant, we can maximize the negative part and solve the following optimization problem:

$$\underset{\{\theta_s\}_{s\in\mathcal{S}}}{\text{maximize}} \sum_{l\in\mathcal{L}} A_l\alpha_l \cos(\theta_{\tau(l)} - \theta_{\sigma(l)} + \varphi_l - \gamma_l)$$

4.2.2 Storage Capacity Constraint

The formulation of the proposed algorithm assumes that all network links have infinite capacity and there is no limitation for the number of vehicles in the queue. However, in the real-world the queue length at the intersection approach may exceed the length of the link and create spillback as illustrated in Figure 4.2 for the first intersection of the arterial shown. Spillback happens when the queue from next intersection blocks the intersection and vehicles cannot use the green time and enter the intersection.

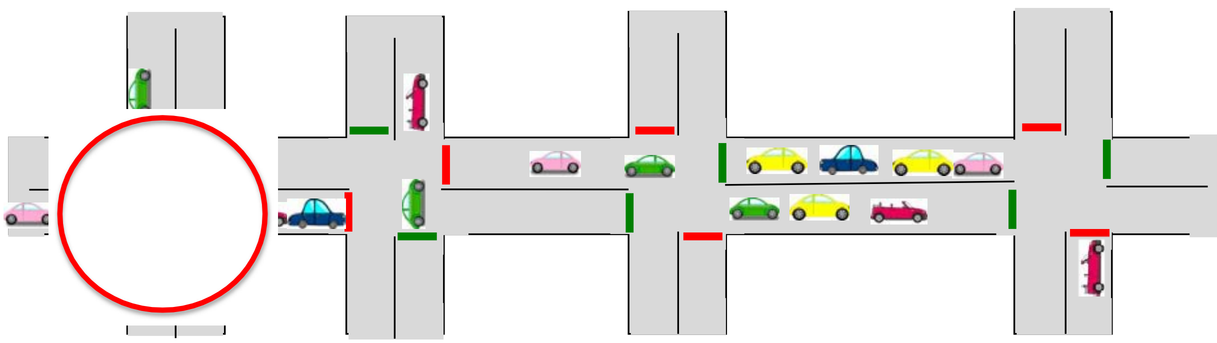


Figure 4.2 Queue Spillback at an Arterial Network

Therefore, to avoid the spillback occurrence, we added the following constraints to the optimization algorithm:

$$0 < Q_l < K_l$$

where K_l is the capacity of link l in number of the vehicles that we can fit in link l at jam density.

4.2.3 Algorithm Solution

This optimization problem is not convex and it is not possible to solve it with existing methods. In the original formulation, a semi-definite relaxation was proposed for approximately solving the problem, but after the capacity constraints were added, the same conditions do not hold anymore and we cannot use the same method to solve the problem. The original formulation set the equivalent quadratically constrained quadratic program (QCQP) of the problem:

$$\max_{z\in\mathbb{R}^{2|S|+2}} z^T W z$$

$$S.\,t\; z^T M_s z = 1 \quad \forall s \in S$$

Where, $L_{s\rightarrow u} = \{l \in L | r(l) = s \ an \ a(l) = u\}$ and

$$W_1[s,u] = \sum_{l \in L} A_l a_l \cos(\varphi_l - y_l)$$

$$W_2[s,u] = \sum_{l \in L} A_l a_l \sin(\varphi_l - y_l)$$

$$\underline{W} = \begin{bmatrix} W_1 & W_2 \\ -W_2 & W_1 \end{bmatrix}, \ W = \frac{1}{2}(\underline{W} + \underline{W}^T)$$

$$x_s = \cos\theta_s \ , y_s = \sin\theta_s \qquad \forall s \in S \quad an \ z = (x, y)$$

And the constraint is

$$E_s[v,u] = \begin{cases} 1 \ if \ u = v \\ 0 \ otherwise \end{cases}$$

$$M_s = \begin{bmatrix} E_s & 0 \\ 0 & E_s \end{bmatrix}$$

We modified the original problem by adding the following constraint

$$z^T C_l z < K_l \quad \forall l \in L$$

Where,

$$C_{l_1}[s,u] = \begin{cases} -2A_l a_l \cos(\varphi_l - y_l) & if \ s = r(l) \ an \ u = a(l) \\ 0 & otherwise \end{cases}$$

$$C_{l_2}[s,u] = \begin{cases} -2A_l a_l \sin(\varphi_l - y_l) & if \ s = r(l) \ an \ u = a(l) \\ 0 & otherwise \end{cases}$$

$$\underline{C_l} = \begin{bmatrix} C_1 & C_2 \\ -C_2 & C_1 \end{bmatrix}, \quad C_l = \frac{1}{2}(\underline{C_l} + \underline{C_l}^T).$$

$$K_l = -A_l^2 - \alpha_l^2 + \left(\tfrac{\omega k_l}{2}\right)^2$$

Where k_l is equal the number of the vehicles that we can fit in link l in jam density.

The original assumption was $Z = zz^T$ and $rank(Z) = 1$, then solve the relaxed convex semi-definite problem (SDP). But after we add the constraint the $rank(Z) = 1$ does not hold for the tested networks anymore. Therefore, the following process was applied to solve the problem:

Step 1:

Solve the original problem, using the semi-definite relaxation and find the optimum offset values for the scenario that all links have infinite capacity.

Step 2:

Estimate the maximum queue length for each link and compare that with the maximum capacity of the link. If there is a link with the maximum queue length bigger than the capacity then we go to step 3, if not, the optimum offset is achieved.

Step 3:

Run the optimization problem with capacity constraints using numerical method to find the optimum point. There will be 2 outcomes:

1. There is an optimum point that satisfies all the constraints, so the optimum offset value is achieved.
2. There is no point that would satisfy all the constraints, so it is impossible to avoid the spillback problem then we use the offset value from step 1.

In step 3, we used the suggested optimum value from step 1 to define the lower and upper bound for the result. Also we used Lagrangian multiplier to form a single function that includes the cost function and constraints. Next we used gradient descent to find the local minimum for several different initial points. Using Lagrange Multiplier and Gradient Descent to solve the modified offset problem.

$$L(z, A_1, \ldots, A_n, \mu_1, \ldots, \mu_m) = z^T W z + \sum_{s=1}^{n} A_n (z^T M_s z - 1) + \sum_{l=1}^{m} \mu_m (z^T C_l z - K_l)$$

Then create a Hessian function

$$\nabla^2_{xx} L(z, A_1, \ldots, A_n, \mu_1, \ldots, \mu_m) = \nabla^2 f(z) + \sum A_s \nabla^2 ceq_s(z) + \sum \mu_l \, c$$

The fmincon function in MATLAB was used to find a local optimum point. fmincon is a gradient-based method that is designed to work on problems where the objective and constraint functions are both continuous and have continuous first derivatives.

4.3 Algorithm Testing, Verification & Validation

We tested the proposed offset optimization algorithm to three real-world arterial networks. We compared through simulation the traffic performance of the algorithm derived offsets against the offsets derived from existing state of the art methodologies.

4.3.1 Design of Experiment

We used the SYNCHRO macroscopic simulation and optimization model to optimize the intersection offsets at each network while the rest of the signal control parameters (cycle time, green times, phase sequence) were kept fixed. As it was previously mentioned SYNCHRO was used because it is the most widely used simulation tool to optimize the signal settings on signalized arterials.

We coded each network into the VISSIM microscopic simulation model and calibrated each model using available data from previous studies. We obtained the following performance measures from each simulation run:

- average number of stops that each vehicle experienced when traveling in major routes
- average vehicle delay that each vehicle experienced when traveling in the major routes

Major routes are typically the routes used by the arterial through traffic on each travel direction, and also on major cross streets that are coordinated.

The average number of stops is equal to the average number of the times that a vehicle makes a complete stop during the trip. VISSIM provides the number of the stops S_i that vehicles experience in each route i, along with the number of the vehicles V_i that completed finished their trip during the simulation. Therefore we estimated the average number of the stops that vehicles in major route experience as

$$S_{total} = \frac{\sum_{i=all\ routes} S_i * V_i}{\sum_{i=all\ routes} V_i}$$

VISSIM estimates the vehicle delay by finding the difference between the travel time in free flow condition and actual travel time to complete its trip. This delay includes the time that the vehicle is stopped behind the traffic lights and the acceleration and deceleration time as well. The average vehicle delay for all vehicles in the major routes is

$$D_{total} = \frac{\sum_{i=all\ routes} D_i * V_i}{\sum_{i=all\ routes} V_i}$$

where D_i is the average delay that vehicles experience in route i.

We applied both SYNCHRO and the proposed algorithm to obtain the optimum signal offsets in each network. We evaluated the performance of the signal settings using the delays and stops from the VISSIM simulation model from the two control scenarios:

1. optimum offsets obtained from SYNCHRO optimization method
2. optimum offsets obtained from the offset optimization algorithm

We simulated the performance of the test networks for a one hour time interval under each scenario. The offset optimizations and simulations were also performed for several traffic conditions on the selected networks. A total of eighteen scenarios were simulated and analyzed (nine operating conditions in three test sites x two control scenarios). Also, a total of five VISSIM simulation runs were made for each scenario to determine the stochastic variability. The average values of the performance measures were used in the evaluation.

4.3.2 Application: San Pablo Avenue, Berkeley, California

Figure 4.3 shows the San Pablo Avenue test network. It is a ten-intersection segment of a major arterial (SR123) in Berkeley, California. There are two through lanes in each direction and left turn lanes at each intersection. The distance between intersections ranges from 450 to 1983 ft, for a total length of 2.02 miles. The posted speed limit is 35 mph. All signals operate as fixed-time coordinated with a common cycle length of 80 seconds. The arrows shown in the Figure represent the major traffic movements. Arrow 1 is the southbound volume (input 1), and arrow 2 is the northbound volume (input 2). Figure 4.4 shows the San Pablo network coded into the SYNCHRO and VISSIM models.

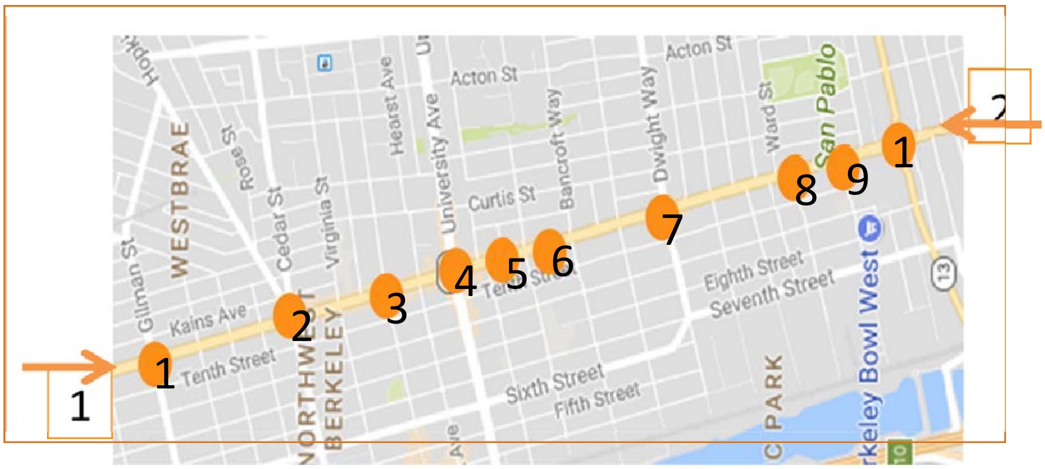


Figure 4.3 San Pablo Avenue Test Arterial

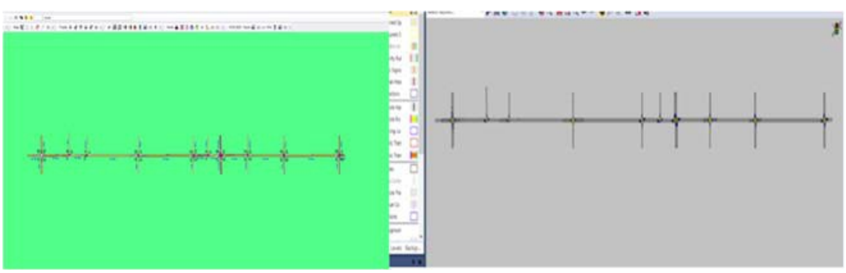


Figure 4.4 San Pablo Avenue coded in SYNCHRO and VISSIM models

We used three traffic profiles for the San Pablo Ave arterial to model different traffic conditions (Figure 4.5). Case 1 represents the AM peak conditions, where the southbound movement has the heavier volume. Case 2 represents the pm peak period with heavier traffic in the northbound direction and case 3 represents lower traffic demands in both directions, typical of the midday traffic conditions.

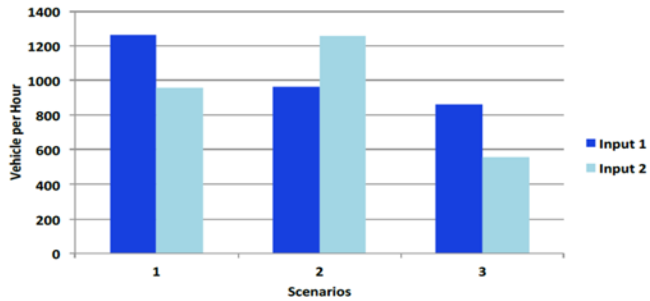


Figure 4.5 San Pablo Avenue Flow Profiles

Table 4.1 shows the simulated delays and stops resulting from the optimized offsets from SYNCHRO and the algorithm for each traffic scenario. The results show that the algorithm offsets improved the number of the stops in all three simulated traffic conditions in the network. The delays were the same for the pm period scenarios. The algorithm showed a small improvement in the off peak period.

The delays shown are the average delays of all vehicles completing the trip on the arterial. Analysis of the simulation results at each intersection and traffic movement indicate that the benefits from the algorithm offsets are higher in both delays and stops on the major traffic movements. The delay increases on the minor approaches are not significant but they do reduce the overall system arterial benefits.

Table 4.1 SYNCHRO vs. Algorithm-San Pablo Avenue

TRAFFIC SCENARIO	NUMBER OF STOPS			DELAY (sec/veh)		
	SYNCHRO	ALGORITHM	% DIFF	SYNCHRO	ALGORITHM	% DIFF
1	3.1	2.8	9.7	95	95	0.0
2	3.8	3	21.1	110	105	4.5
3	3.1	2.9	6.5	65	65	0.0

4.3.3 Montrose Rd, Montgomery County, Maryland

Figure 4.6 shows the second test network, a seven-intersection segment of Montrose Road in Montgomery County, Maryland. There are three through lanes in each direction and left turn lanes at each intersection. The distance between intersections ranges from 1,050 to 3,100 ft. for a total length of 1.4 miles (note that the length of the top portion and the bottom portion of the network is very similar, therefore, the distance from the leftmost intersection to either of the rightmost intersections is around 1.4 miles.) The posted speed limit is 40 mph. All signals operate as fixed-time coordinated with a common cycle length of 120 seconds.

Similar to the first network, Figure 4.6 shows the major inputs of the network. Most traffic on the eastbound approach is coming from input 1 and on the westbound is from input 2 from the top portion and input 3 from the bottom portion. Figure 4.7 shows the test network coded into the SYNCHRO and VISSIM models.

Figure 4.6 Montrose Road Test Network

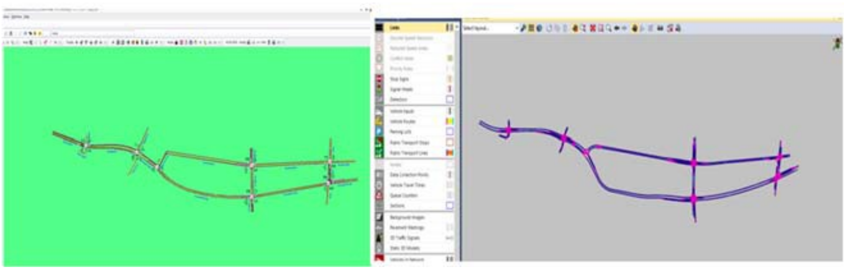


Figure 4.7 Montrose Road coded in SYNCHRO and VISSIM models

In the Montrose Road Network we tested five traffic profiles as shown in Figure 4.8. Cases 1 and 5 represent the AM-peak conditions while cases 2 and 4 are typical of PM peak period, and case 3 represents typical mid-day traffic conditions with lower traffic demands.

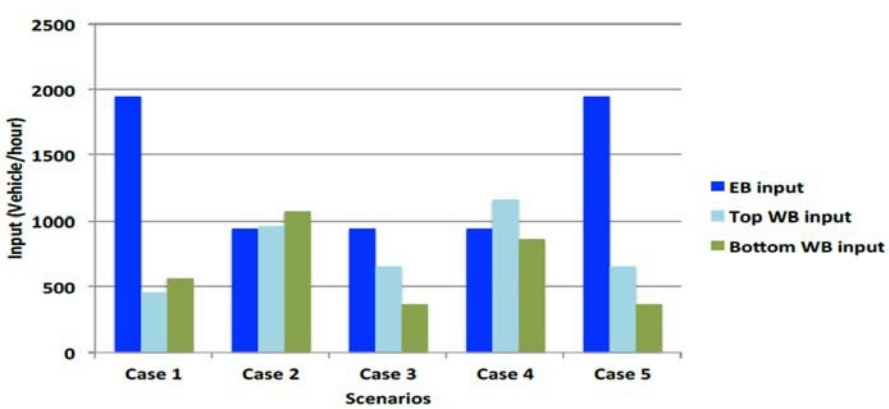


Figure 4.8 Montrose Road Flow Profiles

Table 4.2 shows the delays and stops from SYNCHRO and the algorithm. The offsets from the optimization algorithm resulted in improvements in both delays and stops for all traffic scenarios, even in the case of scenarios 1 and 5 that represent heavy directional traffic conditions where SYNCHRO optimization is effective.

The higher benefits obtained in the Montrose Road network compared to San Pablo Avenue are due to the network configuration and traffic patterns. The proposed algorithm can keep track of multiple traffic inputs, and accounts for the possibility of spillbacks compared to the SYNCHRO model.

Table 4.2 SYNCHRO vs. Algorithm-Montrose Road

TRAFFIC	NUMBER OF STOPS			DELAY (sec/veh)		
SCENARIO	SYNCHRO	ALGORITHM	% DIFF	SYNCHRO	ALGORITHM	% DIFF
1	3.3	3	9.1	153	123	19.6
2	3	2	33.3	126	88	30.2
3	2.2	1.6	27.3	98	81	17.3
4	2.6	2.1	19.2	120	89	25.8
5	2.7	2.6	3.7	128	115	10.2

4.3.4 Application: Live Oak, Arcadia, California

Figure 4.9 shows The Live Oak Avenue test network. This test site is an eight intersection section of the Live Oak Avenue located in Arcadia, California. There are 2 through lanes in each direction and left turn lanes (1 lane for turn) at each intersection. The distance between intersections ranges from 0.1 to 0.4, for a total length of 1.5 miles. The posted speed limit is 35 mph. All signals operate as fixed-time coordinated with a common cycle length of 120 seconds. Figure 4.10 shows the network coded into the SYNCHRO and VISSIM models.

We optimized the offsets and simulated the traffic performance for the AM-peak traffic conditions. The flow profile is shown in Figure 5.11. The average volumes are high and several of the intersections experience spillback problems because of the high traffic demands. The simulation results (Table 4.3) show that the algorithm offsets reduced both the delays and stops by 4% to 6%.

Figure 4.9 Live Oak Avenue Test Arterial

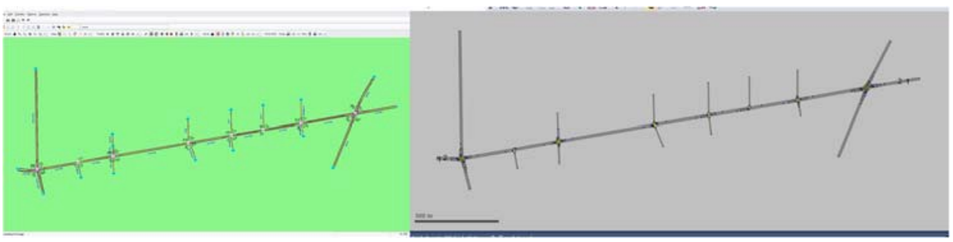


Figure 4.10 Live Oak Network coded in SYNCHRO and VISSIM models

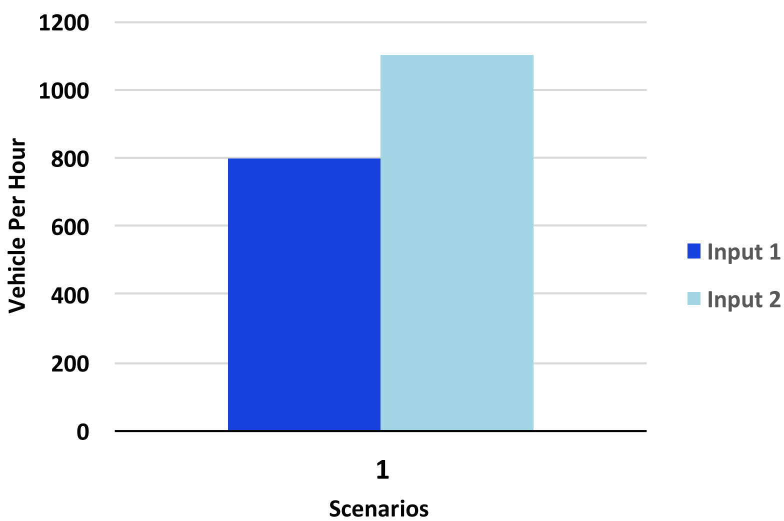


Figure 4.11 Live Oak Network Flow Profile

Table 4.3 SYNCHRO vs. Algorithm-Live Oak Avenue

TRAVEL DIRECTION	NUMBER OF STOPS			DELAY (sec/veh)		
	SYNCHRO	ALGORITHM	% DIFF	SYNCHRO	ALGORITHM	% DIFF
EB	4	3.8	5.0	145	139	4.1
WB	14.8	13.9	6.1	732	700	4.4

CHAPTER 5.

ALGORITHM IMPLEMENTATION

The simulation results from testing the proposed algorithm on three arterial networks with fixed traffic volumes indicate that the proposed offset optimization algorithm results in better performance than commonly used optimization tools. Next, the performance of the algorithms is evaluated using real-time traffic volumes.

This Chapter describes the implementation and testing of the proposed offset optimization algorithm to update the signal timing plans based on real-time information on traffic conditions. The algorithm was tested on a real-world arterial using a specially developed simulation testbed.

5.1 Software Implementation

The simulation of the offset adaptive algorithm with the VISSIM model requires that a Component Object Module (COM) program be created. COM programs can be written in any programming language/platform that supports COM programming (e.g., Python, Visual Basic, C++/C#, Java or MATLAB) to provide features or simulation abilities that are not available in the standard Version of the VISSIM model. Typical uses of VISSIM COM applications are rerouting traffic during major freeway incidents; dynamically changing driver/vehicle desired speed parameters as a response to speed reductions posted on changeable message signs, and implementing complex coordinated and demand responsive signal control strategies. Extensive knowledge and expertise in developing VISSIM simulation models and applications programming is needed to build COM application programs.

The VISSIM software has a COM interface as one of its built-in features. During a VISSIM model run, the COM interface provides the ability to communicate with other COM applications that are also running. These COM communications between VISSIM and Python or MATLAB can provide the means to pass data, model runtime status or model parameters between the two communicating applications. For example, MATLAB can invoke VISSIM, tell the VISSIM model to load a network, set model parameters, start a simulation model run, pause the simulation model run, retrieve simulation data and change VISSIM model parameters. In this project, a MATLAB-VISSIM COM was built to:

- Initiate the VISSIM simulation model run
- Pause the VISSIM simulation model:
    - Once or multiple times during the model run
    - At predetermined intervals or preselected times
    - Upon detection of specific conditions or events
- Obtain traffic volumes from VISSIM
- Optimize signal timing parameters given the current traffic volumes
- Continue the VISSIM simulation model until the next scheduled optimization

Figure 5.1 shows the flow of information between the COM program, VISSIM and the Offset Optimization Module. The COM program and the Offset Optimization Module were developed in MATLAB. At runtime, MATLAB controls the VISSIM model and calls the Offset Optimization Module. MATLAB initiates the VISSIM model run; it initiates and controls the VISSIM to MATLAB data transmissions (traffic volume data), the MATLAB to VISSIM data transmissions, and it calls the Offset Optimization Module (and controls parameter passing between MATLAB and the Offset Optimization Module).

Figure 5.2 shows the iterative COM program's sequence of events. The sequence of events being repeated with every signal cycle, or once every *n* cycles, where *n* is the number of signal cycles determined by the model user on how often the signal timing parameters (e.g., offsets and splits) should be updated.

Appendix A contains the MATLAB code for the MATLAB-VISSIM COM developed to support this coordinated and demand adaptive traffic signal optimization project.

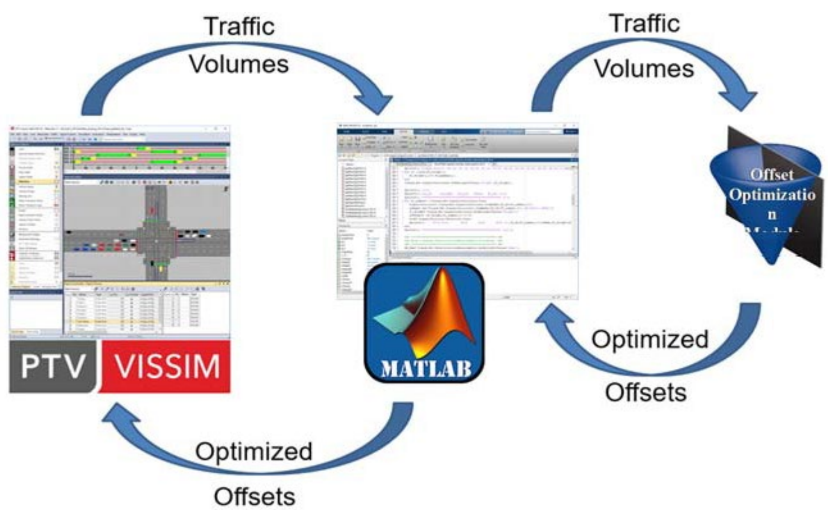


Figure 5.1 Information Flow during MATLAB-VISSIM "COM" Controlled Model Run

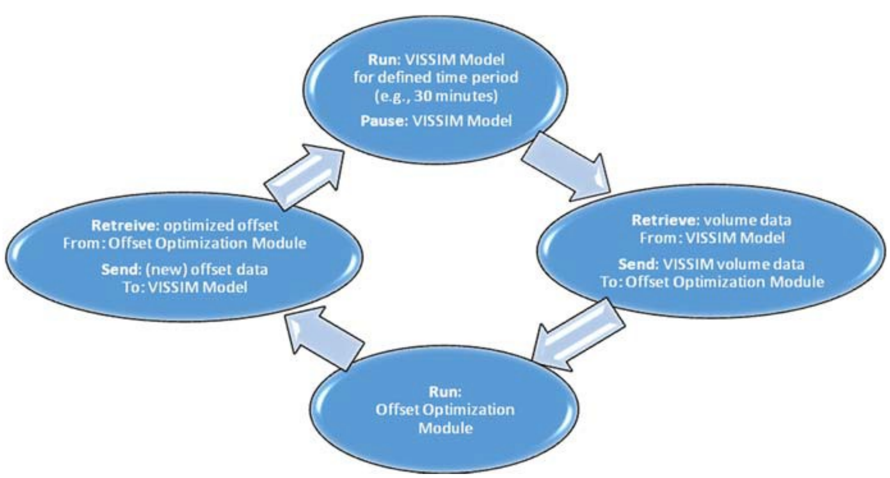


Figure 5.2 Control of VISSIM Model and Offset Optimization Module

5.2 Algorithm Testing

We tested the offset optimization on the San Pablo Test arterial (Figure 4.3) using the software platform shown in Figure 5.1. We simulated traffic conditions of five hours using the flowprofile shown in Figure 5.3. This profile was derived from a number of previous field studies.

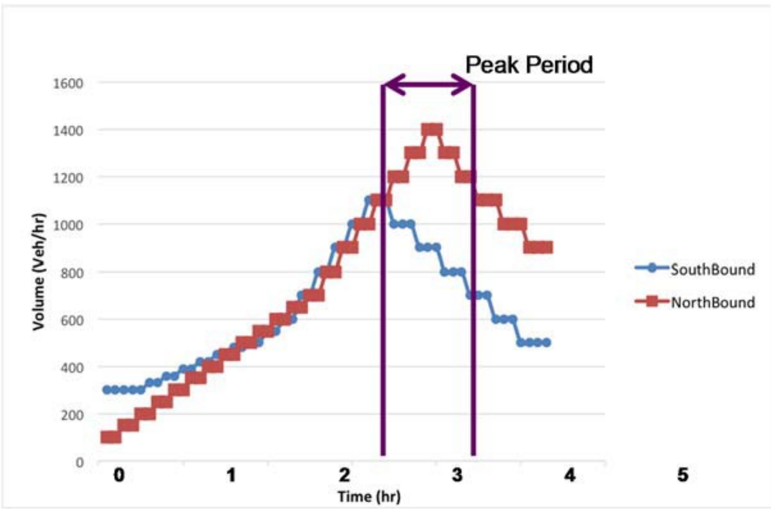


Figure 5.3 San Pablo Avenue Flow Profile for Real Time Offset Adjustments

The baseline conditions were offset optimized with the SYNCHRO model using the average traffic volume during the peak period and the signal timings were kept fixed throughout the five hour analysis period. We used the simulated volumes at selected time intervals as input to the optimization algorithm. The optimized offsets were implemented to the signal controller and the process was repeated in the next time interval.

We tested several time intervals (i.e., number of signal cycles) for offset update periods. Table 5.1 shows the delay and stops obtained per travel direction for each update interval. The best performance was obtained with 15 cycles (approximately 20 minutes). The main reason for worse performance are the transition effects. Signal timing changes necessitate that the signal controller go into transition to implement the new offset value for a time interval depending on the signal controller characteristics during which the signals are essentially uncoordinated. Thus the potential benefits of adjusting offsets to match traffic conditions are negated due to transition effects.

Comparison of the performance of offsets from SYNCHRO and the proposed algorithm show modest improvements of about 5% in the total delay and stops. These improvements are slightly better than the reported benefits of AVS-Light [21,22] adaptive control algorithm developed by FHWA.

Further analysis of the simulation results shows that the VISSIM simulated traffic volumes exhibit variability that leads to differences between the volumes for which the offsets were designed and the actual simulated volumes used to obtain performance measures.

Table 5.1 San Pablo Avenue: Stops and Delays .vs Offset Update Intervals

Travel Direction	# Signal Cycles	#Stops	Delay (sec)
North Bound	5	6	196
	10	5.2	159.2
	15	4.6	133
South Bound	5	5.2	142.8
	10	5.6	157.7
	15	4	106.9

CHAPTER 6

CONCLUSIONS

5.1 Summary of the Study Findings

In this study a control algorithm was developed to optimize the offsets on signal systems in real time based on volume data from loop detectors. The algorithm was evaluated through simulation in real-world arterial test sites. The study also developed an algorithm to determine detector errors, and explored the availability of continuous detector data to determine volumes and switching times for time of day (TOD) signal timing plans.

Offset optimization algorithm: the objective of the proposed optimization algorithm is to minimize the queue lengths at the intersection approaches. The algorithm performance was tested through simulation with the VISSIM microscopic model on three real-world arterials under a variety of traffic conditions. The algorithm-generated offsets had better performance than the offsets generated using the widely used SYNCHRO optimization tool. The level of improvements depends on the network geometry and traffic characteristics. On the average the improvements are higher at networks with complex geometries and traffic patterns.

The real-time update of offsets generated by the algorithm was evaluated on the San Pablo Avenue arterial using a custom made simulation testbed. The testbed was created using a COM interface between the VISSIM model ("real world") and the algorithm written in MATLAB (optimization software). The results indicate modest improvements in traffic performance over the SYNCHRO timings. The frequency of offset updates strongly affects the algorithm performance, due to the signal transition effects.

Detector diagnostics: a data filtering algorithm was developed to estimate the systematic error of detector data. The advantage of this algorithm over similar methods is that it works in real time and it can be a part of data collection process. Application of the algorithm at a real world intersection with multiple detectors resulted in error reduction of up to 25% on high volume approaches which has significant implications in determining appropriate signal settings.

TOD plan selection: A clustering algorithm was applied to determine both the traffic demands that are required input for developing TOD signal timing plans, and the associated plan switching times. The new timing plans based on the volumes obtained from the clustering algorithm outperformed the conventional TOD plans based on traffic volumes measured at predetermined time periods.

Future Research

Offset Optimization Algorithm: the real-time implementation of the algorithm needs to be extensively tested on other arterial networks and operating conditions. Emphasis should be based on dealing with transition effects during the offset updates, as well as development of procedures to determine if the offset updates are warranted based on "true" changes to traffic volumes and not to stochastic fluctuations. This may involve prediction of traffic performance under the existing and proposed signal settings.

Splits and Cycle Length Optimization: There is a need to develop and algorithm to adjust the green times and cycle length in addition to offsets. Splits (and the cycle length) are more sensitive to traffic growth especially for intersection approaches close to saturation (volume to capacity ratio close to 1). The split adjustment algorithm approach depends on the signal operation and detector data availability that provide measures to determine the need for adjustments. Examples include actuated phases that continually max out, or phases with unused green as determined by controller and detector data.

Detection errors: There is a need to incorporate the algorithm on detector diagnostics in the signal settings optimization software platform. The algorithm should be able to detect and correct detector errors and interpolate missing values due to missing or broken detectors.

REFERENCES

1. Cronin B., et al, "Integrated Corridor management," Public Roads, Vol 74 (3), 2010.
2. Stevanovic, A., "Adaptive Traffic Control Systems: Domestic and Foreign State of Practice," NCHRP Synthesis 403, Washington DC, 2010.
3. Luyanda, F, D. Gettman, L. Head, S. Shelby, D. Bullock, and P. Mirchandani, "ACS-Lite Algorithmic Architecture," Transportation Research Record # 1856, pp. 175-184,2003.
4. Muralidharan, A., C. Flores, and P. Varaiya," High-Resolution Sensing of Urban Traffic," 17th IEEE ITSC Conference, Qingdao, China, October 2014.
5. Day, C.M., D.M. Bullock, et al. "Performance measures for traffic signal systems: An outcome-oriented approach' Purdue University, West Lafayette, IN, 2014.
6. Taylor, M., "Automated Traffic Signal Performance Measures," ITS California Annual Meeting, Indian Wells, California, September 2016.
7. Dowling, R., and A. Skabardonis, "Methodology for Assessing Signal Timing Plans Under Varying Demands and Capacities," paper 11-0359, 90th TRB Annual Meeting, Washington DC, January 2011.
8. TrafficWare, "SYNCHRO Traffic Simulation and Optimization Software, 2012.
9. Los Angeles Department of Transportation, "Adaptive Traffic Control Software," Los Angeles, December 2005.
10. May, A., B. Coffman, R. Cayford, G. Merritt, "Automatic Diagnostics of Loop Detectors and the Data Collection System in the Berkeley Highway Laboratory," PATH Research Report, Institute of Transportation Studies, UC Berkeley, April 2004.
11. Jacobson, L. N., N. L. Nihan, and J. D. Bender, "Detecting Erroneous Loop Detector Data in a Freeway Traffic Management System," Transportation Research Record #1287, pp. 151–165, 1990.
12. Smith, B. L, "Forecasting Freeway Traffic Flow for Intelligent Transportation Systems Applications," Ph.D. dissertation. Department of Civil Engineering, University of Virginia, Charlottesville, 2005.
13. Peeta, S., I. Anastasopoulos, "Automatic Real-Time Detection and Correction of Erroneous Detector Data with Fourier Transforms for Online Traffic Control Architectures," Transportation Research Record #1811, 2002.
14. Chen, C., J. Kwon, J. Rice, A. Skabardonis, P. Varaiya, "Detecting Errors and Imputing Missing Data from Single-Loop Surveillance Systems," Transportation Research Record #1855, 2003.
15. Little J.D., M. D. Kelson, and N. H. Gartner, "MAXBAND: A versatile program for setting signals on arteries and triangular networks," Transportation Research Record #795, pp. 40-46, December 1981.
16. Chang, E.C.P., et al., "Analysis of Reduced Delay Optimization and other Enhancements to PASSER II-80, PASSER II-84—Final Report," Research Report 375-1F, Texas Transportation Institute, April 1984.
17. Robertson, D.I., "TRANSYT: A traffic network study tool," Report LR 253, 1967.
18. Wallace, C. E., et al. "TRANSYT-7F User's Manual," McTrans Center, December 1991.
19. Skabardonis, A., "Determination of Signal Settings in Signal Systems with Traffic Actuated Controllers," Transportation Research Record #1554, pp. 18-26, 1996.
20. Coogan, S., E. Kim, G. Gomes, M. Arcak, and P. Varaiya, "Offset optimization in signalized traffic networks via semidefinite relaxation," Transportation Research Part B, vol. 100, pp. 82–92, 2017.
21. Gettman, D., S.G. SAhelby, L. Head, "Data Driven Algorithms for Real-Time Adaptive Turning of Offsets in Coordinated Traffic Signal Systems," Transportation Research Record # 2035, pp. 1-9, 2007.
22. Gettman, D., "ACS Light FHWA Adaptive Control For Closed Loop Systems," Prepared for FHWA, Systems Overview, November 2006.3030

Appendix A

MATLAB COM Code

```
%%% =============================== %%%
%%% MATLAB-Script for Vissim 6+    %%%
%%% Last Update: June 15, 2017     %%%
%%% =============================== %%%
%%% SigPhase(:,1), Amber-->Red, Start of Red Phase: VISSIM Red==1 %%%
%%% SigPhase(:,2), Green-->Amber, Start of Amber Phase: VISSIM Amber==4 %%%
%%% SigPhase(:,3), Red-->Green, Start of Green Phase: VISSIM Green==3 %%%

clear all; clc; close all;



%%% ===================================================== %%%
%%% Vissim Network Signal Controller Parameters           %%%
%%% Select Initial Offsets: "Optimize" or "Baseline"       %%%
%%% ===================================================== %%%
NumSigs= 10;     %%% Number of Signals (intersections) in Corridor
CycleLen= 80;    %%% Corridor Signal Cycle Length (seconds)
AmberTime= 4;    %%% Default Amber time (usually about 4 seconds)

Scen= "Baseline"; %%% "Baseline" or "Optimize";
if strcmp(Scen,"Baseline")
  NewOffset= [00,21,24,75,37,27,60,53,21,63]; %%% Static Offsets for Baseline Runs
else
  NewOffset= [00,60,59,19,41,42,42,18,40,74]; %%% Initial Offsets for Optimization Runs
end %%% if strcmp(Scen,"Baseline")
```

```
%%% ========================================================= %%%
%%% Matlab should only pass one-dimensional arrays to COM %%%
%%% ================================================== ====== %%%
feature('COM_SafeArraySingleDim',1);


%%% ========================================================= %%%
%%% Set "Path_of_network" "and FileName" for current run %%%
%%% ========================================================= %%%
Path_of_network= 'C:\Projects\UCB-AdaptiveSignalControl\SanPabloVISSIM_06_BegCy_SP';
InpxFileName= fullfile(Path_of_network,'SanPablo_Existing_2012_Fixed_updated_ZA_1.inpx');
LayxFileName= fullfile(Path_of_network,'SanPablo_Existing_2012_Fixed_updated_ZA_1.layx');

%%% ======================================================== %%%
%%% Connecting the COM Server => Open a new Vissim Window %%%
%%% ======================================================== %%%
Vissim= actxserver('Vissim.Vissim');


%%% =========================================== %%%
%%% Load a Vissim Network; Load a Layout File %%%
%%% =========================================== %%%
flag_read_additionally= false;
Vissim.LoadNet(InpxFileName,flag_read_additionally);
Vissim.LoadLayout(LayxFileName);


%%% ======================================================== %%%
%%% Read & Display DataCollectionMeasurements attributes  %%%
%%% ======================================================== %%%
```

```
DM_Nam= Vissim.Net.DataCollectionMeasurements.GetMultiAttValues('Name');

DM_No= Vissim.Net.DataCollectionMeasurements.GetMultiAttValues('Name');

DM_DCC= Vissim.Net.DataCollectionMeasurements.GetMultiAttValues('DataCollectionPoints');

fprintf(1,'DCM_No     DCM_Name DCM_DetectorNo \n');

fprintf(1,'========================================== \n');

for DM_number= 1:size(DM_Nam,1)

  fprintf(1,' %02d %16s %16s
\n',DM_No{DM_number,1},DM_Nam{DM_number,2},DM_DCC{DM_number,2});

end

fprintf(1,'========================================== \n\n');



%%% ====================================================== %%%

%%% Get and Display Signal Controllers Initial Settings %%%

%%% ====================================================== %%%

SC_Name= Vissim.Net.SignalControllers.GetMultiAttValues('Name');

fprintf(1,'\n');

fprintf(1,'Count Number        Name NoSGs ProgNo \n');

fprintf(1,'========================================== \n');

for scLCV= 1:Vissim.Net.SignalControllers.Count

   sc(scLCV)= Vissim.Net.SignalControllers.ItemByKey(scLCV);%%% (vector of "SignalController")

   fprintf(1,' %02d       %02d %12s     %02d     %02d  \n',
          scLCV,get(sc(scLCV),'AttValue','No'),SC_Name{scLCV,2},sc(scLCV).SGs.Count,
get(sc(scLCV),'AttValue','ProgNo'));

end

fprintf(1,'========================================== \n\n');

%%% ================================== %%%

%%% Initialize cycle-Timing-Matrices %%%

%%% ================================== %%%

OldOffset= NewOffset;

cTimMat1= zeros(NumSigs,CycleLen,8); %%% (Number-Signals,Cycle-Sec,Number-Phases)
```

```
cTimMatS= zeros(NumSigs,CycleLen,8);   %%% (Number-Signals,Cycle-Sec,Number-Phases)

cTimMat2= zeros(NumSigs,CycleLen,8);   %%% (Number-Signals,Cycle-Sec,Number-Phases)

for isLCV= 1:NumSigs                %%% Loop through Intersections & Get Signal-Timing Matrix for each intersection

  [cTimMat1(isLCV,:,:),cTimMatS(isLCV,:,:),cTimMat2(isLCV,:,:)]=
getSigTimMats(isLCV,CycleLen,AmberTime,OldOffset(isLCV),NewOffset(isLCV));

end


fprintf(1,'\n');

fprintf(1,'Signal Group Error Checking \n [These should match.] \n');

fprintf(1,'========================================== \n');

for scLCV= 1:Vissim.Net.SignalControllers.Count

  cTimSGs= find(not(isnan(reshape(cTimMat1(scLCV,1,:),[1,size(cTimMat1,3)]))));

  sTimSGs= sc(scLCV).SGs.GetMultiAttValues('No');

  sTimSGs= cell2mat(sTimSGs(:,2))';

  fprintf(1,'SigNo %02d:',scLCV)

  fprintf(1,' %d',sTimSGs);

  fprintf(1,':  ');

  fprintf(1,' %d',cTimSGs);

  fprintf(1,'\n');

end

fprintf(1,'========================================== \n\n');

%%% ========================================================== %%%

%%% Simulation times to stop & re-optimize Signal Control %%%

%%% ========================================================== %%%

Random_Seed= 42;               %%% Simulation Random-seed

SimRes= 10;                    %%% Simulation Resolution (steps per second)

End_of_Simulation= 5200;       %%% Simulation Period (end of simulation, seconds)

Sim_Stop_Interval= CycleLen*5;

First_Sim_Stop= 400+Sim_Stop_Interval;
```

```
set(Vissim.Simulation,'AttValue','RandSeed',Random_Seed);

set(Vissim.Simulation,'AttValue','SimPeriod',End_of_Simulation);

set(Vissim.Simulation,'AttValue','UseMaxSimSpeed',true); %%% Set maximum speed:

set(Vissim.Simulation,'AttValue','SimRes',SimRes);


if strcmp(Scen,"Baseline")

  Sim_Stops= End_of_Simulation+20; %%% No stops in simulation time period

else

  Sim_Stops= First_Sim_Stop:Sim_Stop_Interval:End_of_Simulation;

end %%% if strcmp(Scen,"Baseline")

TotVolMat= zeros(size(Sim_Stops,2),size(DM_Nam,1));    %%% Matrix of Current-Step Volumes

OptOfsMat= zeros(size(Sim_Stops,2),size(SC_Name,1)); %%% Matrix of Optimal Offsets


%%% ================================== %%%

%%% Step through VISSIM Simulation %%%

%%% ================================== %%%

ERR_Count_Offset_algo= 0;                                    %%% Offset_algo ERROR Count

tic;


fprintf(1,'Vissim Simulation run started.\n');

fprintf(1,'======================================== \n\n');

for simLCV= 1:1:get(Vissim.Simulation,'AttValue','SimPeriod')-1 %%% Controls Simulation Stops
(seconds)

  set(Vissim.Simulation,'AttValue','SimBreakAt',simLCV);

  Vissim.Simulation.RunContinuous;                            %%% Start the simulation until SimBreakAt


  %%% ======================================== %%%

  %%% Set States for all Signal Controllers  %%%

  %%% ======================================== %%%

  cycLCV= rem(simLCV,CycleLen)+1;
```

```
for scLCV= 1:Vissim.Net.SignalControllers.Count
  scStateNos= sc(scLCV).SGs.GetMultiAttValues('No');
  newScStNo= reshape(cTimMat1(scLCV,cycLCV,not(isnan(cTimMat1(scLCV,cycLCV,:)))),
        [1,sc(scLCV).SGs.Count]);
  for sgLCV= 1:sc(scLCV).SGs.Count
    sc(scLCV).SGs.ItemByKey(scStateNos{sgLCV,2}).set('AttValue','State',newScStNo(sgLCV));
  end %%% for sgLCV= 1:sc(scLCV).SGs.Count
end %%% for scLCV= 1:Vissim.Net.SignalControllers.Count


if any(simLCV==Sim_Stops)
  stopNo= find(simLCV==Sim_Stops);
  fprintf(1,'\n');
  fprintf(1,'Sim Stop Time (sec): %04d \n', simLCV);
  LC_VehCTA=
Vissim.Net.DataCollectionMeasurements.GetMultiAttValues('Vehs(Current,Total,All)');
    TotVolMat(stopNo,:)= [LC_VehCTA{:,2}]';
  if simLCV==Sim_Stops(1)
    CurStepVols= TotVolMat(stopNo,:);
  else
    CurStepVols= mean(diff(TotVolMat(max(1,stopNo-1):stopNo,:),1),1);
  end %%% if LimLCV==1


  OldOffset= NewOffset;
  try
    [NewOffset]= Offset_algo(CurStepVols,Sim_Stop_Interval); %%% Calling the offset algorithm
  catch
    fprintf(1,'\n\n *** Offset_algo ERROR. *** \n\n');
    ERR_Count_Offset_algo= ERR_Count_Offset_algo+1;
  end %%% try
  NewOffset= round(NewOffset);
  OptOfsMat(stopNo,:)= NewOffset;
```

```
%%% =============================== %%%
    %%% Update Signal-Timing Matrices  %%%
    %%% =============================== %%%
    for isLCV= 1:NumSigs
      [cTimMat1(isLCV,:,:),cTimMatS(isLCV,:,:),cTimMat2(isLCV,:,:)]=
          getSigTimMats(isLCV,CycleLen,AmberTime,OldOffset(isLCV),NewOffset(isLCV));
    end
    fprintf(1,'Optimize Offsets, Sim_Time: %d, VISSIM Run-time: 6d %7.1f \n',simLCV,round(toc));
    fprintf(1,'Iter: %02d, OptOfs Vector= (%02d, %02d, %02d, %02d, %02d, %02d, %02d, %02d, %02d,
          %02d)\n',stopNo,OptOfsMat(stopNo,:));
    fprintf(1,'\n');
  end %%% if any(simLCV==Sim_Stops)
   %%% ================================================================ %%%
  %%% Switch to Updated ("Transition" or "Optimized") Signal Control  %%%
  %%% ================================================================ %%%
  if any(simLCV==Sim_Stops)
    cTimMat1= cTimMatS; %%% Switch to Transition-Cycle Offsets
  elseif any(simLCV==(Sim_Stops+CycleLen))
    cTimMat1= cTimMat2; %%% Switch to Updated (Optimized) Offsets
  end %%% if any(simLCV==Sim_Stops)
end %%% for simLCV= 1:get(Vissim.Simulation,'AttValue','SimPeriod')
Vissim.Simulation.RunContinuous; %%% Run last second of simulation
%%% ===================================== %%%
%%% Release Vissim Object (Close Vissim)  %%%
%%% ===================================== %%%
fprintf(1,'\n');
fprintf(1, 'Offset_algo Error Count: %d \n',ERR_Count_Offset_algo);
fprintf(1,'\n');
%%% Vissim.release;
```