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

Developing Operating Rules and Simulating Performance for One-dedicated-lane Bus Rapid Transit/Light Rail System

H.-S Jacob Tsao, Yasser Dessouky, Kevin Ingham, Rocky Ongkowidjojo, Jason R. Tsao

California PATH Research Report
UCB-ITS-PRR-2010-1

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.

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. This report does not constitute a standard, specification, or regulation.

Final Report for 65A0268

March 2010

ISSN 1055-1425

CALIFORNIA PARTNERS FOR ADVANCED TRANSIT AND HIGHWAYS

DEVELOPING OPERATING RULES AND SIMULATING PERFORMANCE

FOR

ONE-DEDICATED-LANE BUS RAPID TRANSIT/LIGHT RAIL SYSTEMS

H.-S. Jacob Tsao, Yasser Dessouky and Kevin Ingham
Department of Industrial and Systems Engineering
San Jose State University
San Jose, California, USA

Rocky Ongkowidjojo
Department of Computer Engineering
San Jose State University
San Jose, California, USA

Jason R. Tsao
Department of Electrical Engineering
University of California, Los Angeles
Los Angeles, California, USA

February 20, 2010

ACKNOWLEDGEMNT

The authors would like to thank Caltrans and California PATH for funding this research under the category of New and Innovative Research Idea. They are grateful for the guidance and support generously provided by Mr. Weibin Zhang of PATH and Ms. Sonja Sun and Mr. Bradley Mizuno of Caltrans.

DEVELOPING OPERATING RULES AND SIMULATING PERFORMANCE

FOR

ONE-DEDICATED-LANE BUS RAPID TRANSIT/LIGHT RAIL SYSTEMS

ABSTRACT

Caltrans has recently funded a project, through the Mineta Transportation Institute (MTI) at San Jose State University, about geometric and performance feasibility of using virtually only one dedicated but dynamically reversible lane to provide two-way Bus Rapid Transit (BRT) or light-rail services at the same or similar service levels as those achievable with two dedicated lanes, along the median of a busy commute corridor with at-grade crossings and regular provision of left-turn lanes. This project extended that deterministic performance study in four complementary directions. We (i) developed two episodes of simulation/animation to facilitate understanding key ideas of the concept, (ii) addressed the "backward-compatibility" issue, (iii) developed speed control rules to absorb the impact of the stochastic demand (and the different distances between pairs of neighboring stops) on the performance of a closed system, and (iv) developed speed-control rules and transit-signal-priority rules for an open system to ensure a green signal for all transit vehicles at any intersection. We believe that the next step in this line of research is to work on, simultaneously, deployment issues with city and transportation planners, acceptance issues with transit users, automobile drivers, other stakeholders (including the general public), design issues for actual deployment sites (based on inputs provided by the planners, transit users, automobile drivers and other stakeholders), and development of site-specific and high-fidelity study tools.

Key Words: *Bus Rapid Transit, Light Rail, One Dedicated Lane, Busway, Geometric Design, Transit Signal Priority*

Executive Summary

Caltrans has recently funded a project, through the Mineta Transportation Institute (MTI) at San Jose State University, about operational feasibility of using virtually only one dedicated but dynamically reversible lane to provide two-way Bus Rapid Transit (BRT) or light-rail services at the same or similar service levels as those achievable with two dedicated lanes, along the median of a busy commute corridor with at-grade crossings and regular provision of left-turn lanes. The scope of that project was limited to identification and resolution of major implementation issues, and assessment of the practicality and performance of the proposed system. A key issue is the geometric feasibility of using virtually only one lane to enable crossing of buses traveling in opposite directions. Another key issue is the scheduling feasibility of providing constant-headway service for both directions. That research has been completed recently. Crossing is accommodated on the otherwise unused or underused median space resulting from provision of the left-turn lanes. A number of geometric configurations have been developed and have been shown to be feasible for accommodating both crossing and passenger operations. A deterministic performance study of a closed system, in which the surrounding traffic is ignored, demonstrated good performance. Major findings of that research have been summarized in (Tsao et al., 2007, 2009a, 2009b, 2009c, 2009d).

This project extended that deterministic performance study in four complementary directions. For this performance research, we focused on a simple but the worst case, in which every bus stop is also equipped with a crossing space and there are no other "stand-alone" crossing spaces along the corridor. Due to the unconventional nature of the concept of one-dedicated-lane BRT or light-rail systems, we developed, in the first direction, two episodes of simulation/animation to facilitate understanding key ideas of the concept. The ultimate goal of the one-dedicated-lane BRT or light-rail system is for it to be expanded to a two-dedicated-lane system, when sufficient demand has been built up hopefully through the transit-oriented development facilitated by the one-dedicated-lane system, but without any need for excessive amount of modification to the infrastructure already put in place. In the consumer electronics industry, this requirement is referred to as "backward compatibility." This requirement is particularly important for a light-rail system because if a significant portion of the rails already laid and used for the one-dedicated-lane system must be moved when upgraded to a two-dedicated-lane system, the combined costs of the one-dedicated-lane and its expansion may not be justifiable. As a result, transportation planners need to be convinced of the feasibility of the end-state two-dedicated-lane system and of the backward compatibility. We addressed this backward-compatibility issue as the second direction.

In the third direction, we focused on speed control rules to absorb the impact of the stochastic demand (and the different distances between pairs of neighboring stops) on the performance of a closed system, i.e., with the surrounding traffic ignored or its effect negated. Our results demonstrate a promising potential of the concept of a one-dedicated-lane BRT or light-rail system for efficient operation, as an end-state or as an intermediate state of a two-dedicated-lane, space-efficient system. In the fourth direction,

we focused on speed-control rules and transit-signal-priority rules for an open system to ensure a green signal for all transit vehicles at any intersection. The dwell time of a transit vehicle at a stop is assumed to be constant, regardless of the demand, although the length between two consecutive stops may vary. We simulated the movements of BRT buses via the speed-control rules. We also developed an algorithm that produces transit-priority signaling cycle for every intersection along the corridor. Based on the simulated movements and the transit-priority signaling cycle, the transit vehicles can travel the corridor without impedance caused by the surrounding traffic. The two animation episodes developed in the first direction help visualize not only several key features of the one-dedicated-lane system but also insights that significantly simplify the derivation of the transit priority signaling cycle. These new concepts and simulation, optimization and animation tools developed by this seed-grant project can be combined to study geometric feasibility and traffic performance of a one-dedicated-lane BRT for possible deployment sites in the future.

As specified in the proposal of this project, the focus of this performance research was on a simple but the worst case, in which every bus stop is also equipped with a crossing space and there are no other "stand-alone" crossing spaces along the corridor. As pointed out earlier, crossing of buses traveling in two opposite directions can take place at any section where the unused or underused median space is sufficiently long to accommodate such crossing. The more such stand-alone crossing spaces (away from a bus stop), the closer the performance of the one-dedicated-lane BRT or light-rail system to its two-dedicated-lane counterpart. With the availability of such stand-alone crossing spaces, the impact of a delay to a bus can be more localized, and the delayed bus can catch up with the schedule with real-time transit signal priority. Although the development of this type of real-time transit signal priority for one-dedicated-lane BRT or light-rail systems is beyond the scope of this research project, design of this type of real-time transit signal priority systems, given the current state of the art of intelligent transportation systems in general or the state of art of transit signal priority in specific, is more of a development effort, rather than a research effort. We believe that the next step in this line of research is to work on, simultaneously, deployment issues with city and transportation planners, acceptance issues with transit users, automobile drivers, other stakeholders (including the general public), design issues for actual deployment sites (based on inputs provided by the planners, transit users, automobile drivers and other stakeholders), and development of site-specific and high-fidelity study tools.

TABLE OF CONTENTS

Title page i
Acknowledgment iii
Abstract v
Executive Summary vii
Table of Contents ix
List of Tables and Figures x1
1. INTRODUCTION 1
   1.1. Background 1
   1.2. Motivation for the Concept of One-dedicated-lane BRT or Light-rail Systems. 2
   1.3. Potential Space-Saving Benefit: Examples 2
   1.4. Key Issues and State of the Art 3
   1.5. Scope of the Current Project 3
   1.6. Organization of this Report 5
2. TWO SIMULATION-ANIMATION EPISODES 5
   2.1. Brief Overview of the Concept of One-dedicated-lane BRT 5
   2.2. Two Simulation/Animation Episodes 8
3. TWO-DEDICATED-LANE BRT SYSTEM WITH BACKWARD COMPATIBILITY 10
   3.1. Conventional Geometric Design for a Two-dedicated-lane BRT and A More Efficient Dedication of Two Lanes 10
   3.2. Backward Compatibility and New One-dedicated-lane and Two-dedicated-lane BRT Configurations 12
4. A SIMULATION TOOL FOR STUDYING IMPACT OF STOCHASTIC DEMAND AND VARIABILITY OF STATION SEPARATION 13
   4.1. Introduction 14
   4.2. System Flow Diagram 15
   4.3. Coding Description 16
   4.4. Verification 25
   4.5. Analysis 28
   4.6. Conclusions 32
5. OPERATIONAL RULES OF TRANSIT SIGNAL PRIORITY AND SPEED CONTROL FOR EFFICIENT OPERATION OF AN OPEN SYSTEM 33
   5.1. The Elapsed Time between Two Consecutive Bus Passages through an Intersection 36
   5.2. Algorithm for Optimizing Transit Signal Priority Cycles 39
   5.3. Computer Implementation and Analysis 41
6. CONCLUSION 43
APPENDIX A: Promodel Code of Simulation Tool for Study of Impacts of Stochastic Demand and Variability in Station Separation 45
APPENDIX B: Java Program Implementing the Transit Signal Priority Rules 65
REFERENCES 72

LIST OF TABLES AND FIGURES

List of Tables

Table 4.1. Train speed as a function of headway 29
Table 4.2. Number of passengers serviced between the two scenarios during the normal simulation time..... 30
Table 5.1. $M^{s}_{\min\max}$ for nine *s* values, associated with Δ = 1 minute and 5 minutes..... 42
Table 5.2. s^{opt} and $M^{opt}_{\min\max}$ for nine Δ values from one minute to nine minutes 43

List of Figures

Figure 2.1.a: Unused or underused median space on a typical arterial section equipped with one left-turn lane on each of the two ends 6
Figure 2.1.b: Median space unused or underused for traffic purposes: The section of the W. El Camino Real in Mountain View, California, USA between S. Shoreline Blvd. and Castro Street, shown as the street oriented diagonally . 6
Figure 2.2. One Lane Dedicated to Alternating Two-way Traffic..... 7
Figure 2.3. A Space Dedicated to Crossing Operation 7
Figure 2.4. An Integrated Bus Stop and Crossing Space for a One-dedicated-lane BRT on a 6-Lane Right-of-way: Occupying Two Lanes But Taking Away Only One Through Lane and Leaving Two General-Purpose Through Lanes per Direction, with Left-Turning Sacrificed..... 8
Figure 2.5. A screen-shot of simulation-animation episode 1. Note the different times at which two buses traveling at opposite different directions depart from a bus stop 9
Figure 2.6. A screen-shot of simulation-animation episode 2: Simulated bus crossing. Note that alighting and boarding takes place after the two buses have crossed and stopped at the two respective passenger platforms..... 10
Figure 3.1: A Straight but Inefficient Geometric Design for a "Bus-Stop" Section of a Two-Dedicated-Lane BRT on a 8-Lane Right-of-Way: Occupying Four Lanes and Taking Away Three Lanes to Accommodate Two Through Lanes and One Left-turn Lane for Each Direction 11
Figure 3.2: Slanting of Dedicated Lanes of a Two-dedicated-lane BRT System and the Saving of One Lane on a 7-Lane Right-of-Way, for a Bus-stop Section: Sacrificing Left-turn lanes for Passenger Platforms; Occupying Only Three Lanes and Taking away Only Two lanes to Accommodate Two Through Lanes and One Left-turn Lane for Each Direction..... 11
Figure 3.3: A Two-dedicated-lane BRT System Taking Away Exactly Two Lanes at a Bus Stop, with the Bus Stop Accommodated Completely on the Unused Median Space, without Sacrifice of Left-turn Lanes: Crosswalks behind Stopped Buses 12
Figure 3.4. A One-dedicated-lane BRT with which the Two-dedicated-lane BRT Illustrated in Figure 3.3 is Backward Compatible..... 13
Figure 4.0. The reaction required by the train speeds in the single-tracked system as headway is decreased 14

Figure 4.1: A geometric sketch of a section of a conventional dual-track urban light-rail system constructed on street median 15
Figure 4.2: Overview of the Five-stop Train System..... 16
Figure 4.3: Screenshot of the system overview 16
Figure 4.4: Screen-shot with "LinkXtoY" variable 25
Figure 4.5: Screen-shot with "LinkXtoY" variable 26
Figure 4.6: Screen-shot of the start of a 12-minute simulation run 27
Figure 4.7: Screen-shot of the end of the 12-minute simulation..... 27
Figure 4.8: Results report..... 28
Figure 4.9. Break-down of the time spent in system doing various activities between the two extreme data points of a 6 minute headway and a 10 minute headway . 31
Figure 4.10: Specification of passenger counts for the five stops in column Frequency 32
Figure 4.11: Performance report for a more realistic scenario with different levels of destination popularity 32
Figure 5.1: A Simple Space-Time Diagram for Study of Operational Performance..... 35
Figure 5.2. Bus locations at t=0 (or t=h) of a repeating cycle of length h 35
Figure 5.3. Bus locations at t=h/2 of a repeating cycle of length h 35
Figure 5.4. A macro-cycle divided into n micro-cycle of equal lengths..... 40
Figure 5.5. A macro-cycle with a special micro-cycle and with the rest of cycle time divided into n micro-cycle of equal lengths 40

DEVELOPING OPERATING RULES AND SIMULATING PERFORMANCE

FOR

ONE-DEDICATED-LANE BUS RAPID TRANSIT/LIGHT RAIL SYSTEMS

1. INTRODUCTION

1.1 Background

Public transportation is perhaps one of the few sustainable transportation solutions for urban or suburban areas in the future, particularly given the steady increase in energy demand by the world's developed nations, depleting fossil-fuel reserves, absence of clearly promising alternative fuels for transportation, rising fuel cost, mounting environmental concerns, and the energy and environmental pressures imposed by the fast development of China, India, Brazil, Russia and other emerging economies. Light-rail or subway system is the classical and conventional mass transit system used in most developed countries while the Bus Rapid Transit (BRT) is a new mass transit system that has been adopted by both developed countries, such as the U.S., and emerging economies, such as China and Brazil (Diaz et al., 2004, Jarzab et al., 2002; Levinson et al., 2002; Lin and Wu, 2007).

To some developed countries such as the U.S., a major problem with public transportation in most urban or suburban areas is the low population density, which is at least partially influenced by the prevailing zoning regulations. The vast majority of the US urban or suburban areas have been designed with automobiles as the primary means of people mobility. In many such urban-sprawl areas, the current demand for bus transportation or light-rail is so low that dedicating two full lanes in roadway median (except at the at-grade intersections) and the space needed for bus stops for exclusive use by buses or light-rail trains has led to or would lead to underutilization of the right-of-way, usually amid heavy automobile traffic during the peak commute hours. Such underutilization has led to or would lead to resentments of the driving public against public transportation. Transit-oriented development has been promoted as a means to increase demand for transit and reduce automobile use. Without a good transit system, transit-oriented development may be difficult; without such development, it may be difficult to justify the construction of a good transit system. This can be viewed as a "chicken-and-egg" problem.

Moreover, in many urban or suburban commute corridors, right-of-way sufficient for a two-dedicated-lane BRT or light-rail system simply does not exist. For example, portions of the Eugene-Springfield BRT, named "EmX Green Line", in Oregon, U.S are implemented with only one dedicated lane (Carey, 2006; Breakthrough Technologies Institute, 2007). To developing countries such as China, the construction and

development of BRT or light-rail system requires significant amount of investment, which sometimes is a big hurdle to the development of an efficient mass transit system.

1.2 Motivation for the Concept of One-dedicated-lane BRT or Light-rail Systems

These "chicken-and-egg", right-of-way and cost problems motivated our concept of one-dedicated-lane BRT or light-rail system, which effectively requires only one dedicated but dynamically reversible lane in the median of an arterial serving a busy commute corridor and significantly reduces the requirement of land and funding. We believe that it provides a promising alternative for the government and public to build an efficient and effective public transportation system in either developed countries such as the U.S. or developing countries such as China. For busy commute corridors that have sufficient right-of-way but do not have sufficient demand to warrant dedication of two mixed-use lanes to public transportation, the proposed system could be very useful as an intermediate step toward a two-dedicated-lane system because of its potential for facilitating transit-oriented development.

Caltrans has recently funded a project, through the Mineta Transportation Institute (MTI) at San Jose State University, about operational feasibility of using virtually only one dedicated but dynamically reversible lane to provide two-way Bus Rapid Transit or light-rail services at the same or similar service levels as those achievable with two dedicated lanes, along the median of a busy commute corridor with at-grade crossings and regular provision of left-turn lanes. A co-PI of this project is the PI of that project, which is entitled "*Bus Rapid Transit/Light Rail Implemented on One Dedicated Lane: Operational Feasibility, Practicality and Systems Analysis"* (Tsao et al., 2009a). For ease of reference in what follows, we will refer to that project as the MTI project.

1.3 Potential Space-Saving Benefit: Examples

Specific examples for the potential space-saving benefit can be found in the San Francisco Bay Area. The right-of-way of an overwhelming majority of the corridor being considered for the East Bay BRT Project, which seeks to deploy a two-dedicated-lane BRT system between Berkeley and San Leandro along the Telegraph Avenue and the International Boulevard, is equivalent to seven traffic lanes, with a parking lane considered as equivalent to a traffic lane and one dedicated left-turn (for both directions) also considered as one traffic lane. This amounts to two lanes for through traffic for each of the two directions. However, the geometrical configuration of a bus stop of the two-dedicated-lane BRT system as sketched in the current plan requires a right-of-way width equivalent to four traffic lanes. More precisely, two of the four are dedicated for the bus traffic and the other two are allocated to accommodate passenger activities, one for each direction; left-turning for each direction is accommodated within the construct of the four-lane configuration. This amounts to almost a 60% reduction of capacity for the non-BRT through traffic. If geometric symmetry is assumed, only one lane of through traffic will be allowed for each direction, and there will be space for parking only on one of the two curbsides. Corridor sections designated to accommodate such a bus stop may be recurrent bottlenecks for through traffic, not to mention non-recurrent traffic bottlenecks

during an accident or incident. The one-dedicated-lane BRT system to be studied in this proposal requires only a right-of-way width equivalent to two traffic lanes, instead of four, and allows four lanes of non-BRT through traffic, two in each direction. Its potential benefit is drastic – enabling a BRT system dedicated for two-way bus traffic while not reducing capacity for non-BRT through traffic.

The potential of the one-dedicated-lane BRT could also be significant for the El Camino Real corridor spanning San Francisco and San Jose if a dedicated BRT system is to be implemented along the corridor. The right-of-way of an overwhelming majority of the southern portion of the corridor is equivalent to nine traffic lanes, with the right-of-way dedicated to parking included. The one-dedicated-lane BRT system to be studied in this proposal allows six lanes of non-BRT through traffic, three in each direction. Its potential benefit is as drastic – enabling a BRT system dedicated for two-way bus traffic while not reducing capacity for non-BRT through traffic. When demand for BRT has been built up, the one-dedicated-lane BRT system can be easily expanded to a two-dedicated-lane system. This proposal seeks to optimize the operating rules and simulate the system performance, in order to verify the feasibility of the one-dedicated-lane BRT system.

1.4 Key Issues and State of the Art

The scope of the MTI project was limited to identification and resolution of major implementation issues, and assessment of the practicality and performance of the proposed system. A key issue is the geometric feasibility of using virtually only one lane to enable crossing of buses traveling in opposite directions. Another key issue is the scheduling feasibility of providing constant-headway service for both directions. That research has been completed recently. Crossing is accommodated on the otherwise unused or underused median space resulting from provision of the left-turn lanes. A number of geometric configurations have been developed and have been shown to be feasible for accommodating both crossing and passenger operations. A deterministic performance study demonstrated good performance. However, that performance study (of the MTI project) focused on a closed system, with the surrounding traffic ignored. It also ignores the stochastic nature of demand patterns. Major findings of that research have been summarized in (Tsao et al. 2007, 2009a, 2009b, 2009c, 2009d).

1.5 Scope of the Current Project

This project extended that the deterministic performance study of the MTI project in four complementary directions. Due to the unconventional nature of the concept of one-dedicated-lane BRT or light-rail systems, we developed, in the first direction, two episodes of simulation/animation to facilitate understanding key ideas of the concept. The ultimate goal of the one-dedicated-lane BRT or light-rail system is for it to be expanded to a two-dedicated-lane system, when sufficient demand has been built up, but without any need for excessive amount of modification to the infrastructure already put in place. In the consumer electronics industry, this requirement is referred to as backward compatibility. This is particularly important for a light-rail system because if a

significant portion of the rails already laid and used for the one-dedicated-lane system must be moved when upgraded to a two-dedicated-lane system, the combined costs of the one-dedicated-lane and its expansion may not be justifiable. As a result, transportation planners need to be convinced of the feasibility of the end-state two-dedicated-lane system and of the backward compatibility. We addressed this issue as the second direction.

In the third direction, we focused on speed control rules to absorb the impact of the stochastic demand (and the different distances between pairs of neighboring stops) on the performance of a closed system, i.e., with the surrounding traffic ignored or its effect negated. A simulation model has been developed, and a simulation tool based on the commercial simulation software ProModel has been developed. With the simulation tool, the performance of speed control as an operating rule was studied. A relationship between a headway and the required average transit-vehicle speed has been developed, given the values of some important operational parameters. This simulation tool can be used to study more realistic scenarios for speed control or to study other operating rules beyond speed control.

In the fourth direction, we focused on speed-control rules and transit-signal-priority rules for an open system to ensure a green signal for all transit vehicles at any intersection. The dwell time of a transit vehicle at a stop is constant, regardless of the demand, although the length between two consecutive stops may vary. More precisely, we study an open system where the transit traffic traveling on the dedicated system must contend with the traffic signals at the at-grade intersections of the dedicated lanes with cross streets. Given a required headway, the locations of the stops (which are assumed to be, as the worst possible case, the only locations where transit traffic can cross on the dedicated system), a pre-set known amount of dwell time of a transit vehicle at a stop, we simulated the movements of BRT or light-rail trains via the speed-control rules and determined where each transit vehicle should be at any moment in time. We also developed an algorithm that produces transit-priority signaling cycle for every intersection along the corridor, with any possible arrival times of the two opposite streams of buses or light-rail trains treated, the geometric configuration of the corridor roadway (particularly the distance between any pair of consecutive intersections), the required proportions of green, amber, red, etc. signals at each intersection, and bus acceleration/deceleration rates as key input parameters. Based on the simulated movements and the transit-priority signaling cycle, the transit vehicles can travel the corridor without impedance caused by the surrounding traffic. The two animation episodes developed in the first direction help visualize not only several key features of the one-dedicated-lane system but also insights that significantly simplify the derivation of the transit priority signaling cycle. These new concepts and simulation, optimization and animation tools developed by this seed-grant project can be combined to study geometric feasibility and traffic performance of a one-dedicated-lane BRT for possible deployment sites in the future.

These directions can be combined for more realistic simulation studies in the future. Our results demonstrate a promising potential of the concept of a one-dedicated-lane BRT or

light-rail system for efficient operation, as an end-state or as an intermediate state of a two-dedicated-lane, space-efficient system.

Our research findings are equally applicable to BRT systems or light-rail systems. For ease of discussion, we address only the former explicitly in Section 2 through 4. We address only the latter explicitly in Section 5 and use the equivalent concepts and terms of single-tracking commonly used in the rail industry.

1.6 Organization of this Report

The findings associated with the four directions are summarized in four sections. In Section 2, we introduce the two simulation-animation episodes, after a brief description of the concept of one-dedicated-lane BRT system. Section 3 addresses the critical issues of most efficiently dedicating space to a two-dedicated-lane BRT system and backward compatibility, i.e., the absence of any need to significantly modify the infrastructure already put in place for a one-dedicated-lane system as an intermediate solution. We propose a new geometric configuration for a two-dedicated-lane system that uses the median space even more efficiently than those proposed in the MTI project (Tsao et al., 2009a and 2009c). From this new perspective of backward compatibility, we propose also in Section 3 a new geometric configuration for a one-dedicated-lane system that can be expanded to efficiently to the new and more efficient two-dedicated-lane system. In section 4, we discuss the use of speed control rules to absorb the impact of stochastic demand, describe a simulation tool developed to study the stochastic performance, and report the stochastic performance estimated with the simulation tool. The ProModel programs developed for the simulation tool are given in Appendix A. In Section 5, we develop operational rules of transit signal priority and speed control to ensure efficient operation of an open system. The computer programs developed to implement the speed-control and transit-signal-priority rules are included in Appendix B. Section 6 provides concluding remarks and points out worthy subjects for future research.

2. TWO SIMULATION-ANIMATION EPISODES

In this section, we first briefly describe the concept of one-dedicated-lane BRT system and then introduce two simulation-animation episodes. Several dimensions of many design options and many geometric configurations have been proposed in (Tsao et al., 2009a, 2009b) . In this report, we use a minimum number of configurations to convey the key concepts for brevity.

2.1 Brief Overview of the Concept of One-dedicated-lane BRT

Although the right-of-way of an arterial serving a busy corridor may be wider at interactions with major cross streets, the total width of the right-of-way dedicated to the rest of the roadway of such a corridor changes only occasionally. In particular, the width of a section between two adjacent intersections equipped with one left-turn lane at each of the two opposite ends of the section typically remains constant. When compared to the length of such a section, a typical left-turn lane is rather short. Therefore, a significant

amount of median space tends to exist along the roadway between two such adjacent intersections, and such median space is not useful for facilitating the through traffic on a conventional roadway consisting of general-purpose (mixed-traffic) lanes. As a result, such median space is typically occupied with plants or is used for left-turning convenience into store parking lots. We, therefore, refer to such median space as "unused median space" or "underused median space." For discussion convenience, we use the former in the rest of this report. Figures 2.1.a and 2.1.b illustrate such unused or underused median space.

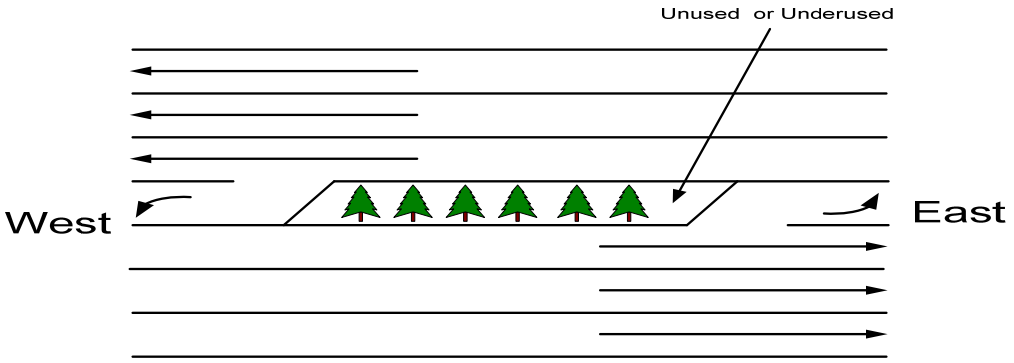

Figure 2.1.a: Unused or underused median space on a typical arterial section equipped with one left-turn lane on each of the two ends

Figure 2.1.b. Median space unused or underused for traffic purposes: The section of the W. El Camino Real in Mountain View, California, USA between S. Shoreline Blvd. and Castro Street, shown as the street oriented diagonally

Figure 2.2 illustrates how, by capitalizing on the unused median space, an additional, dedicated lane can fit into the median of an arterial. Note that the slanting as shown in Figure 2.2 is very important and actually saves one-lane worth of right-of-way. If the dedicated lane is straight with respect to the direction of the roadway and the two left-turn lanes are to be provided, then more space will be required beyond the one additional lane dedicated to the BRT. Also note that such slanting can also save one-lane worth of right-of-way for a corresponding two-dedicated-lane BRT system. In particular, such slanting can save one lane for the current design of East Bay BRT (AC Transit, 2007 & 2008), which calls for dedication of two straight lanes.

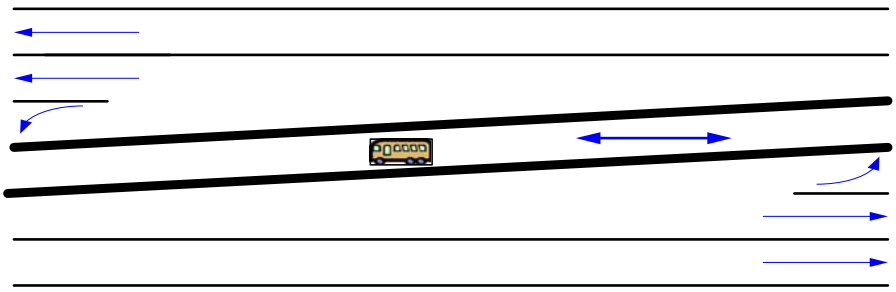

Figure 2.2. One Lane Dedicated to Alternating Two-way Traffic

We note that, as a result of the frequent availability of left-turn lanes at intersections and the typical constant section width, much unused median space exists throughout the arterial of a typical busy commute corridor. We can capitalize even more on the existence of such unused median spaces and use them for bus crossing. Figure 2.3 illustrates this use

.

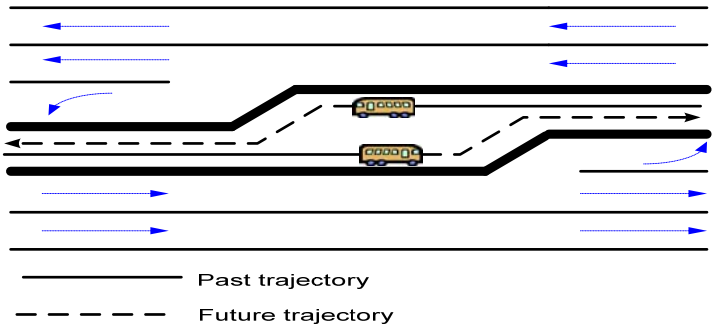

Figure 2.3. A Space Dedicated to Crossing Operation

The crossing space can be expanded to also accommodate a bus stop, with the possibility of sacrificing one or both of the two left-turn lanes, as illustrated in Figure 2.4.

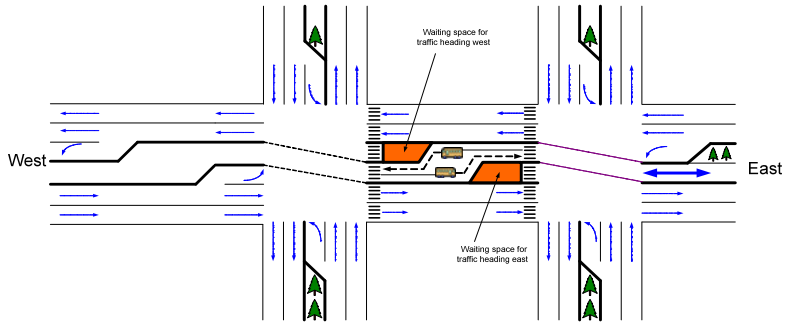

Figure 2.4. An Integrated Bus Stop and Crossing Space for a One-dedicated-lane BRT on a 6-Lane Right-of-way: Occupying Two Lanes But Taking Away Only One Through Lane and Leaving Two General-Purpose Through Lanes per Direction, with Left-Turning Sacrificed

2.2 Two Simulation/Animation Episodes

To help visualize the operations of the proposed one-dedicated-lane BRT and to test the responses of the system to varying section lengths and passenger boarding and alighting counts, we used Macromedia Flash Professional 8, originally developed by Macromedia and now part of Adobe (renamed Flash CS3 Professional), to generate two episodes of computer-generated animation (Tsao, JR. and Tsao,H.-S.J., 2008a and 2008b). To view them, the reader can enter the URL directly into any internet browser. The reader can also download the files and then open the files with Internet Explorer. Alternatively, the reader can download Adobe Flash Player (Adobe, 2008) and use it to play the episodes. The two episodes can be accessed through the internet. All the configurations sketched in this report are based on "right-hand-side driving." The idea is the same for systems in which vehicles are driven on the left-hand side, due to complete symmetry in the corresponding infrastructure geometry. The two episodes simulate bus movements for a roadway system where vehicles are driven on the left-hand side of the street and were developed for a conference held in India, where vehicles are driven on the left-hand side. We also use these episodes to illustrate the international applicability of the concept of one-dedicated-lane BRT.

The first episode simulates the crossing of two opposing streams of buses along a corridor arterial. In this episode, the section lengths, i.e., the distances between two adjacent bus stops vary. In addition, the numbers of passengers boarding and alighting the buses also vary. As a result, the times buses spend at bus stop vary. Nevertheless, speed is controlled so that the travel time, including the time required for passenger boarding and alighting, of any bus on any section is constant. This is to ensure a constant headway and timely crossing of buses traveling in the two opposite directions. (In these episodes, the constant headway is set to 10 minutes, and hence the constant travel time for each section, including time allotted for passenger activities, is set to 5 minutes.) All buses are assumed to have an identical acceleration rate and an identical deceleration rate, but they may have a different cruising speed. The cruising speed of a bus, after acceleration at the common constant rate but for a varying required duration, varies so as to accommodate the difference in section length and the difference in passenger boarding and alighting counts. Because of the purpose to simulate the "stream crossing" and

because of the limited display area on the computer monitor, the geometric configuration and passenger activities are omitted. Note the different times, shown as Time near the bottom-right corner of the display (in minutes), at which two buses traveling at opposite different directions depart from a bus stop. Such differences result from the corresponding differences in boarding and alighting counts. These are illustrated by the screen-shot captured in Figure 2.5. Also note the different speeds, which result from the different section lengths as well as different boarding and alighting counts. The algebraic equations simulating bus movements are detailed in Section 4.

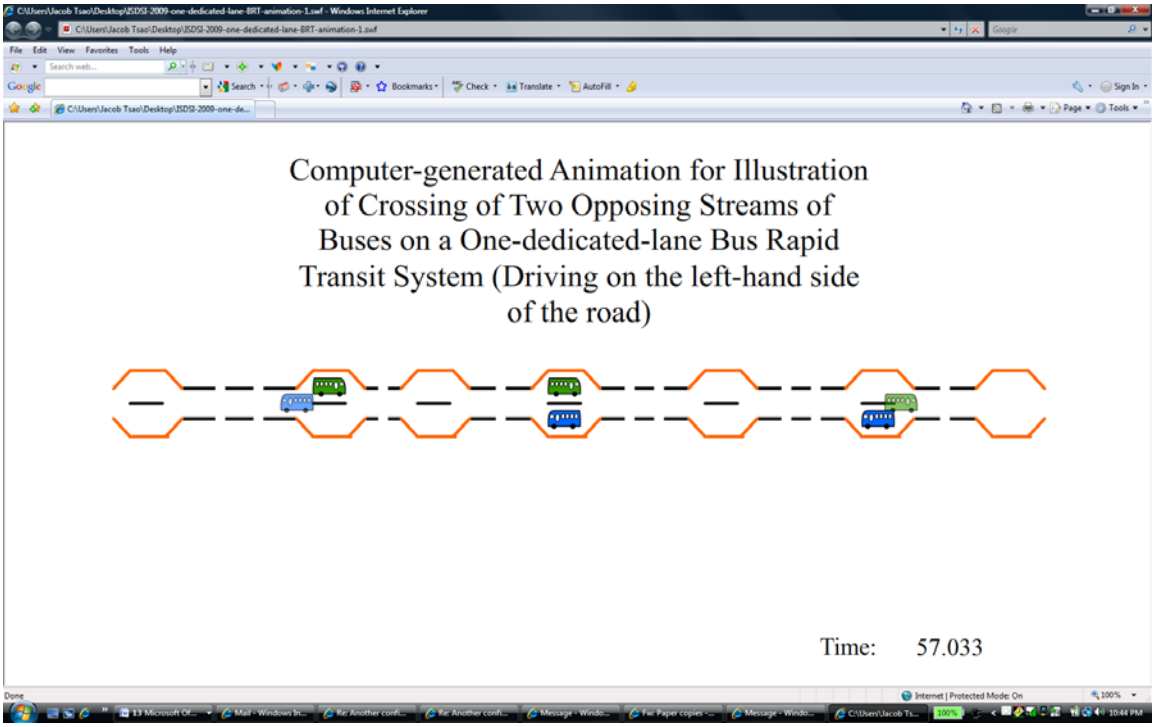

Figure 2.5. A screen-shot of simulation-animation episode 1. Note the different times at which two buses traveling at opposite different directions depart from a bus stop.

The other episode simulates the crossing of two buses at an integrated bus stop and crossing space. The geometric configuration used in this episode is the one shown in Figure 2.4. Note that alighting and boarding takes place after the two buses have crossed and stopped at the two respective passenger platforms, for safety rand other reasons. The simulated crossing is illustrated by the screen-shot captured in Figure 2.6.

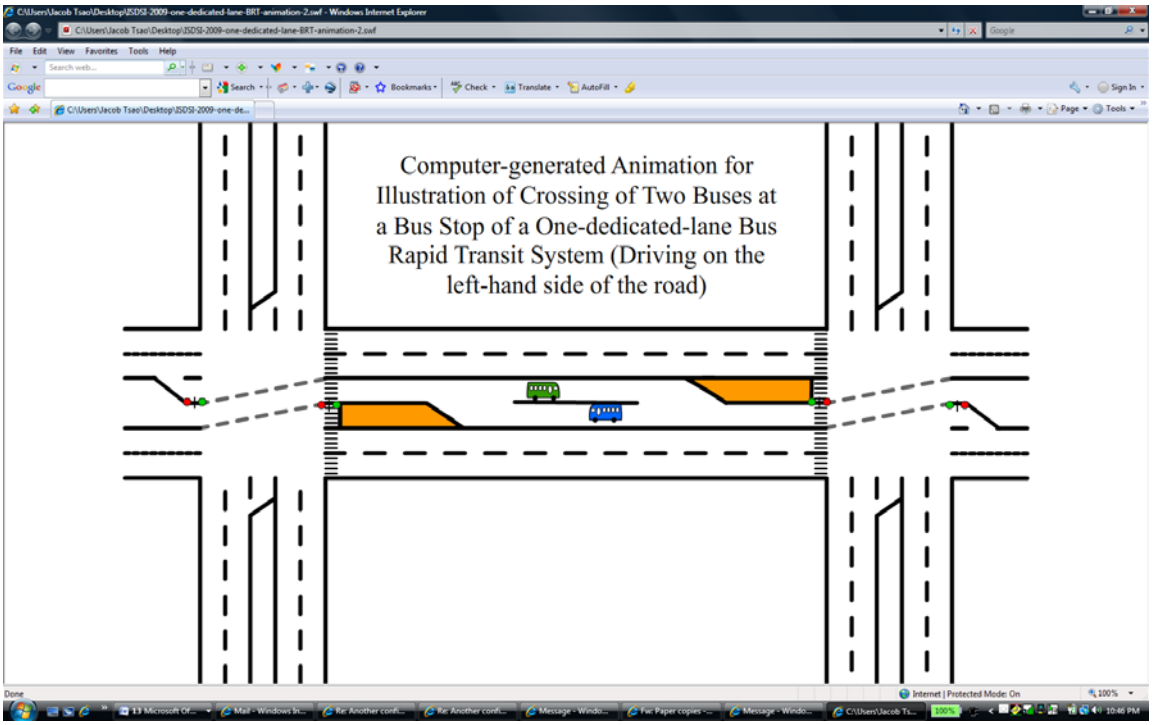

Figure 2.6. A screen-shot of simulation-animation episode 2: Simulated bus crossing. Note that alighting and boarding takes place after the two buses have crossed and stopped at the two respective passenger platforms

3. TWO-DEDICATED-LANE BRT SYSTEM WITH BACKWARD COMPATIBILITY

In this section, we first discuss the common, prevailing geometric design for a two-dedicated-lane BRT (or light-rail) system and point out the fact that such a design occupies four lanes of the right-of-way for any section equipped with a bus stop, i.e., for any bus-stop section, and takes away three through-lanes. We then propose the requirement of backward compatibility and a new configuration for a one-dedicated-lane BRT system that satisfies the backward-compatibility requirement.

3.1 Conventional Geometric Design for a Two-dedicated-lane BRT and A More Efficient Dedication of Two Lanes

Multiple designs for a bus stop of such a system exist. Figure 3.1 illustrates such a design (AC Transit, 2007 & 2008).

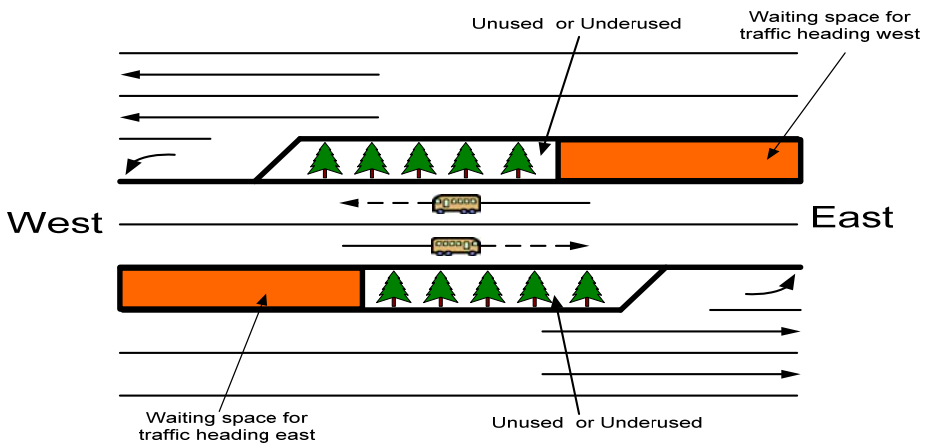

Figure 3.1: A Straight but Inefficient Geometric Design for a "Bus-Stop" Section of a Two-Dedicated-Lane BRT on a 8-Lane Right-of-Way: Occupying Four Lanes and Taking Away Three Lanes to Accommodate Two Through Lanes and One Left-turn Lane for Each Direction

Note that if the left-turn lanes can be sacrificed, the system illustrated in Figure 3.1 can be improved so that one lane can be saved. Such a design is illustrated in Figure 3.2. Note that no left-turn lanes are provided on this section, and the two dedicated lanes are slanted with respect to the orientation of the roadway. For a section without a bus stop, the common, conventional configuration can be easily obtained simply by replacing the two passenger platforms shown in Figure 3.1 (above) with trees. However, note that this configuration still occupies four lanes and takes away three lanes from general use. The configuration shown in Figure 3.2 can also be easily modified to become a geometric configuration for a section without a bus stop by replacing the two passenger platforms with two left-turn lanes. Note that this configuration occupies three lanes but takes away only two lanes, thus saving one full lane with respect to the common, conventional configuration.

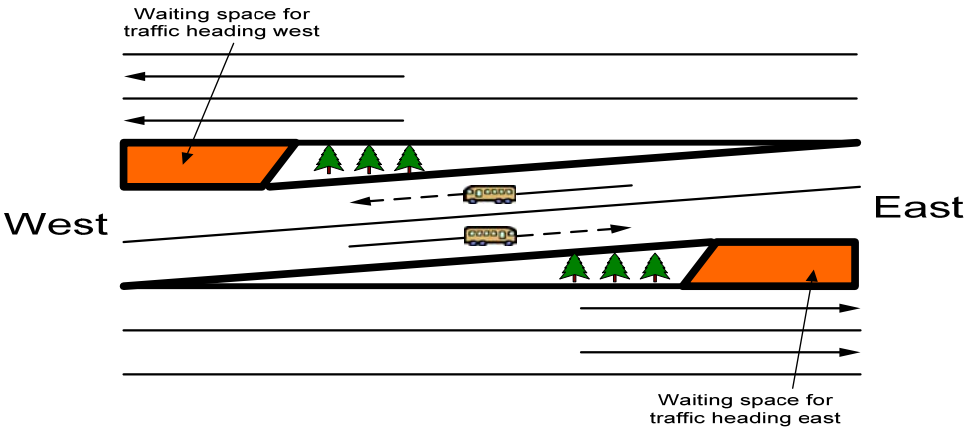

Figure 3.2: Slanting of Dedicated Lanes of a Two-dedicated-lane BRT System and the Saving of One Lane on a 7-Lane Right-of-Way, for a Bus-stop Section: Sacrificing Left-turn lanes for Passenger Platforms; Occupying Only Three Lanes and Taking away Only

Two lanes to Accommodate Two Through Lanes and One Left-turn Lane for Each Direction

3.2 Backward Compatibility and New One-dedicated-lane and Two-dedicated-lane BRT Configurations

It should be clear that the configuration of Figure 3.2 is backward compatible in the sense that its implementation via expansion from the configuration of Figure 2.4 requires moving only one passenger platform (and of course adding one dedicated lane) and such a move is actually unavoidable. Configurations illustrated in Figures 2.4 and 3.2 were proposed in the MTI project. But, the concept of backward compatibility was not defined until now.

We further propose a new configuration that is even more efficient in dedication of median space. The configuration illustrated in Figure 3.3 accommodates the passenger activities in the middle of the section and fully utilizes the unused or underused median space. Note that the platform can be accessed via mid-block crosswalks, but additional traffic signals may be required for safety. Also, to enhance safety, the pedestrian crosswalks can be located behind the positions where buses stop and passengers alight or board the buses. Note that this configuration occupies three lanes and takes away two general-purpose lanes. However, no left-turn lanes are sacrificed. No other configurations can be more efficient in utilization of the median space.

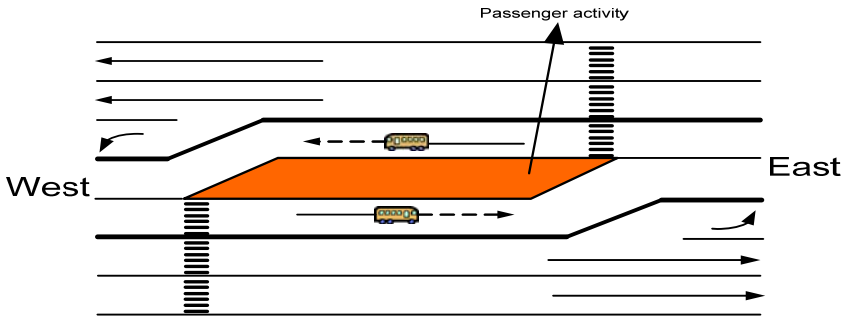

Figure 3.3: A Two-dedicated-lane BRT System Taking Away Exactly Two Lanes at a Bus Stop, with the Bus Stop Accommodated Completely on the Unused Median Space, without Sacrifice of Left-turn Lanes: Crosswalks behind Stopped Buses

We also propose a new one-dedicated-lane configuration for the critical requirement of backward compatibility for this most median-space efficient configuration. It is illustrated in Figure 3.4.

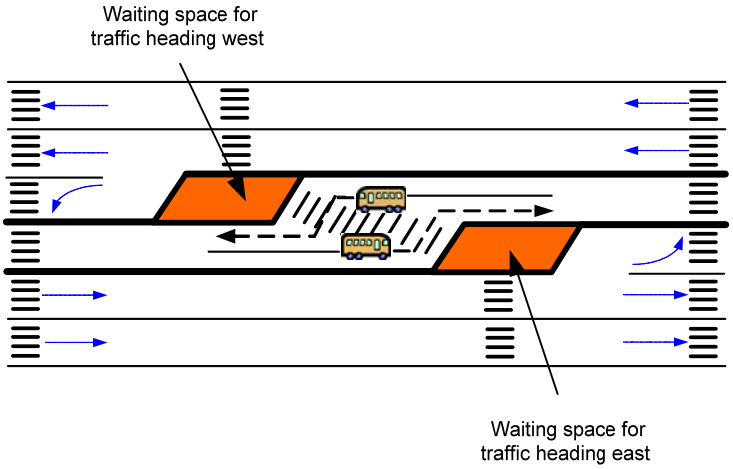

Figure 3.4. A One-dedicated-lane BRT with which the Two-dedicated-lane BRT Illustrated in Figure 3.3 is Backward Compatible

Note that, to expand this system to the two-dedicated-lane configuration sketched in figure 3.3, only one of the two passenger platforms need to be moved. Which one to be moved depends on whether the additional lane is added to the north or south side of the right-of-way of Figure 3.4. These two new configurations and the new concept of backward compatibility perhaps constitute the most efficient deployment path in terms of utilization of median space and make the concept of one-dedicated-lane BRT, as an intermediate step toward a two-dedicated-lane system, significantly more credible and desirable.

4. A SIMULATION TOOL FOR STUDYING IMPACT OF STOCHASTIC DEMAND AND VARIABILITY OF STATION SEPARATION

In this section, we focus on speed control rules to absorb the impact of the stochastic demand (and the different distances between pairs of neighboring stops) on the performance of a closed system, i.e., with the surrounding traffic ignored or its effect negated. We develop a simulation model and a simulation tool based on the commercial simulation software ProModel. We also use the simulation tool to study the performance of speed control as an operating rule, including the relationship between a headway and the required average transit-vehicle speed. This simulation tool can be used to study more realistic scenarios for speed control or to study other operating rules beyond speed control.

Rising vehicular traffic on urban roadways can create severe delays which cost governments and people both time and money. Given that narrow downtown urban corridors have little room for widening roadways other alternative methods must be analyzed to help ease congestion.

A five-stop single-track train system was created in ProModel simulating a light-rail line where trains can only cross at the passenger stops. The single-track system could be used by municipalities to take away less right-of-way currently used by general-purpose vehicular traffic.

This single-track train model utilizes the concept of headway to allow the trains to increase or decrease velocity to compensate for delays encountered during normal operation. The model was analyzed by comparison to a standard common dual-track train system.

It was found that trains in the single-track model must increase their speed nonlinearly between two stations to compensate for increased stress in the system. The increased stress was simulated by linearly reducing headway in iterations in a linear fashion, as summarized in Figure 4.0 below. This increased stress can be analogous to a requirement for increased customer service by having a high frequency of train arrivals.

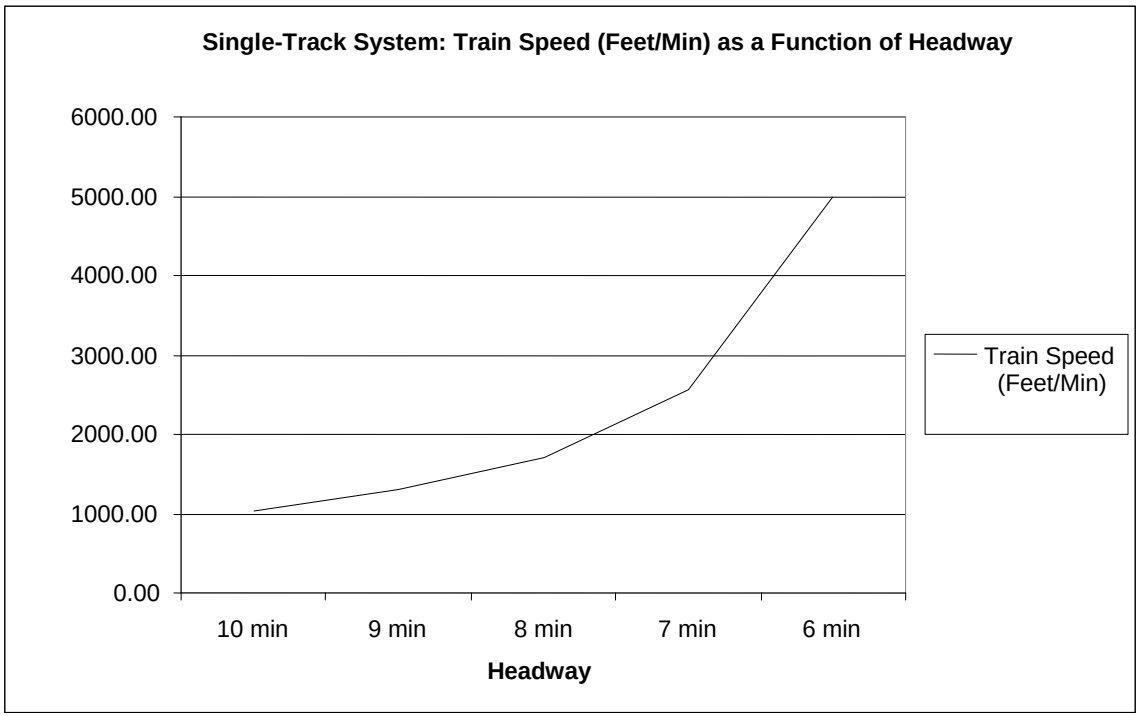

Figure 4.0. The reaction required by the train speeds (feet per minute) in the single-tracked system as headway is decreased.

The model described in this section can easily be modified to increase the number of stations and trains. It is important to note that the model should only be used as a base to build a more realistic model depending on actual real-world scenarios.

4.1 Introduction

Urban light-rail systems, as sketched in Figure 4.1, conventionally utilize a dual track system in which trains travel in one of two directions either inbound or outbound from a depot. The inbound and outbound trains have the ability to freely pass one another. Vehicular traffic travels in either direction on the outside of the light-rail tracks. The

penalty for this type of system is poor utilization of space along the travel corridor (see Figure 4.1).

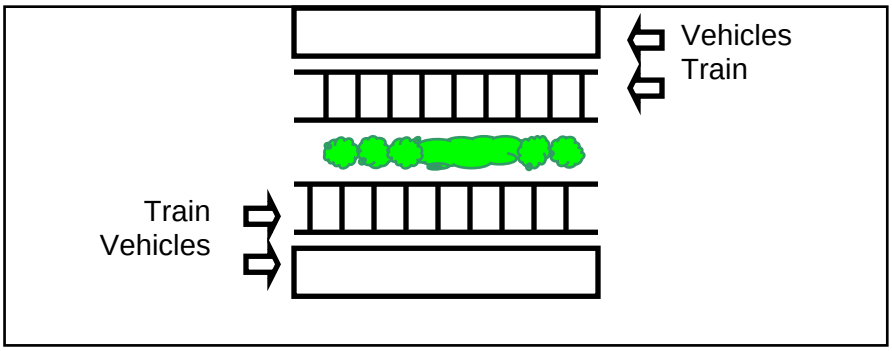

Figure 4.1: A geometric sketch of a section of a conventional dual-track urban light-rail system constructed on street median

Elimination of one of the dual tracks can drastically reduce the space requirement. However, trains on the single-tracked light rail system would only be able to pass each other at specified nodes. These nodes can correspond to their standard passenger stops.

An important aspect of this transportation system is the concept of headway. The basic idea is that the speed of the trains is a function of the amount of time they spend at a given stop. If an incoming train has been delayed at one stop then it must speed up to ensure that the passengers at the next stop do not wait longer than a specified amount of time.

Conversely, if the train is too far ahead of schedule it must slow down so that the spacing of the trains is kept appropriately. Without maintaining headway the single-tracked model will quickly become unstable. The concept of headway was incorporated into the single-tracked model to ensure that there is an even distribution of timing at the stops.

This section describes a simulation model of a single-tracked light rail system with inbound and outbound trains. The simulation was built using the ProModel software and can be used to compare the performance and characteristics of the dual-tracked and single-tracked systems.

There are many more adjustments that can be made to the model depending on a designer's ideas or actual system layout. This model is intended to be a baseline from which to add more layers of realism on a case-by-case basis in determining if the elimination of a dual track in favor of a single track is feasible.

4.2 System Flow Diagram

The single-tracked system modeled has five passing nodes connected by four links. The nodes represent passenger stops and the only places where the trains can pass. Each node represents a train stop. Two trains begin in opposite directions with one moving "East" and the other moving "West."

Passengers enter the system and proceed to one of five "Benches." Each bench is associated with a nearby train stop. From the bench the passengers move to the associated "Train Stop" when a train arrives. The passengers board the first train that arrives at the Train Stop and travel on that train until the train reaches their required stop. Once at the required stop the passenger will exit the train and move to the bench. From there the passenger will exit the system. An overview of the five-stop train system is shown in Figure 4.2:

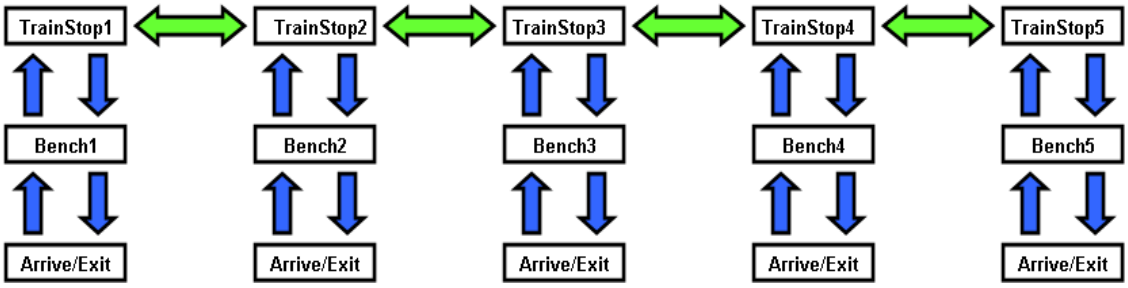

Figure 4.2: Overview of the Five-stop Train System

When each train reaches the end of the line at either Train Stop 1 or Train Stop 5 it will change directions and being to transit the other direction. The trains adjust their average speed upon leaving each station to compensate for the amount of time it took to load and unload passengers.

A screen-shot of the system overview is provided next in Figure 4.3. The ProModel software graphics are largely for information and troubleshooting purposes only. However the graphics were utilized to help verify particular aspects of the model.

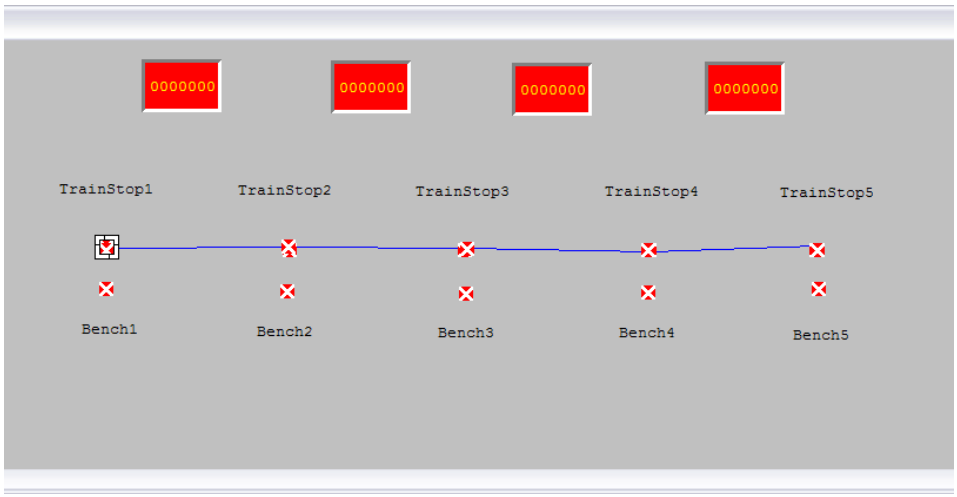

Figure 4.3: Screenshot of the system overview

4.3 Coding Description

Each of the following sections describes a module defined within the ProModel software and gives a detailed rationale behind the model design.

LOCATIONS

The backbone of the single-track train and passenger system in the ProModel software is the use of location objects. The location objects serve the function of allowing a space for the entities to arrive, exit, route, and interact. This model uses two different categories of locations: "Benches" and "Train Stops."

The first category is five locations named "BenchX" with infinite capacity where passenger entities can arrive, exit, and route from.

Next there are five locations named "TrainStopX" with infinite capacities. The purpose of the train stop locations is to provide a place for the passenger entities and train entities to merge using the "Load" command and unmerge using the "Unload" command. It is a place where the passengers are essentially boarding and debarking from their trains. The trains can then travel between train stop locations.

ENTITIES

There are two types of entities used in the simulation: "People" and "Train." The people entities are the passengers in the system and travel from the entrance of the system, then to a preselected destination, and then to the exit. The train entities travel only between train stops along the predefined path network at a variable speed. The speed the train must travel between stops is defined by the entity attribute "aTrainSpeed." This attribute is specified in the "Speed" column of this screen. See the ATTRIBUTES, ARRIVALS, and PROCESSING LOGIC sections for more information.

PATH NETWORK

The network named "Net1" is used to represent train travel between stops. It is defined as a non-passing network, which means that one entity cannot pass another from behind, but it does not prohibit two entities from colliding head-on. The distance between the nodes (train stops) is set at one mile (5280 feet) defined with the variable "PathDistance" (see VARIABLES section below).

ARRIVALS

There are seven rows in the arrival module corresponding to the 5 bench locations and 2 trains. A third, fourth, or even fifth train can be added to the model arriving at different locations if desired.

For example, people entities arrive at a bench (BenchX) one at a time with a probabilistic frequency as defined by the exponential probability distribution. Each "People" entity is assigned a final destination in the logic by use of a user defined probability distribution named "StopNo" and an attribute "aStop." As an example the following logic applies for "People" entities arriving at "Bench1:"

```
aStop = StopNo()
while aStop = 1 do
{
    aStop = StopNo()
}
```

This statement calls an integer value from the user defined distribution. The while/do statement ensures that no passenger will have the same destination as its own arrival. The distribution named StopNo() is defined as follows:

Percentage	Value
20	1
40	2
60	3
80	4
100	5

If it is desired to model some stations as being more popular than others then the "Percentage" column in the user distribution can be adjusted to reflect that scenario.

The two (or three or more) train entities only arrive once at the start of the simulation. The user can change the initial location of each bus to any of the 5 bus stops. Changing the initial location or using only one bus can be useful in understanding different logistical scenarios.

The train entities have the "aDirection" and "aTrainSpeed" attributes defined in the logic of the arrival stage. The default trains speed is zero and the direction can take either a 1 or 0 value depending on the desired direction of travel (see Attributes section below).

ATTRIBUTES

The three entity attributes defined in the model are named aStop, aDirection, and aTrainSpeed. Attributes aStop and aDirection take integer values that are carried with an entity throughout the simulation no matter where the entity travels and can be analyzed and changed in the model logic. aTrainSpeed takes a real number as its value is determined by an output of the speed function defined in the processing logic.

The attribute "aStop" is attached to the people entities and holds a value of 1 though 5 which corresponds to the entities final destination as defined in the Arrivals module. The attribute "aDirection" is attached to the train entities and used by the model logic to determine which direction each train is travelling. The model has been arbitrarily set so that if aDirection = 0 then the train is travelling towards TrainStop5 and if aDirection = 1 then the train is travelling in the opposite direction.

The value of the attribute aTrainSpeed is determined by the value of several variables. The basic equation used in the processing logic is:

aTrainSpeed = PathDistance / (HalfHeadway - X)

The variables PathDistance and HalfHeadway are all user defined in the Variables screen. The third variable X is a compensation for the amount of time spent at the train stop loading and unloading passengers (see PROCESSING LOGIC for more information). For example, if

HalfHeadway = 10 min
X = 5 min
PathDistance = 5280 feet

Then the speed with which the train must travel to its next stop is:

aTrainSpeed = 5280 ft / (10 min – 5 min) = 1056 ft / min

This is exactly 0.2 miles per min or 12 miles per hour.

GLOBAL VARIABLES

There are several different functions of the global variables utilized in the model. First, the group of variables named "LinkXtoY" is used to virtually set the single-tracked rail as occupied or unoccupied where X is the node the train has left and Y is the node to which it is travelling.

If the variable "LinkXtoY" is equal to 1 then it has been booked by a train entity for use. If it is equal to 0 then it is free to be booked by a train entity for use. These variables are a back-up or fail-safe for the headway concept. If the headway is removed from the model then these variables would ensure that no train can pass except for at one of the train stations.

Next, the variables "WaitXCtr" are used to count the number of times that a train waits for a track to be unoccupied. If the headway concept is utilized properly in the model then these variables should never be utilized.

The variables "vBenchX" are used in the processing logic to determine if a particular train stop "X" has people waiting for the train. They take a value of 0 if the train stop is empty and a value of 1 if there are people entities waiting.

The variable "PathDistance" is called by the Path Network "Net1" to determine the length of the track between nodes. This variable is user defined and its default value is 5280 feet.

The user defined variable “vMinStopTime” is called by the processing logic to determine the amount of time in minutes a train stops at a station if no passengers want to exit or none are waiting to board. This minimum time assumes that the train will still stop at a station for some time period even if no passengers are waiting or want to exit. See the Processing Logic section below for more details.

The variable “HalfHeadway” is defined as a deadline in minutes for each train to load, unload, and travel to the next stop. It is used by the processing logic to ensure that trains adjust their speed to account for any delays.

The variables “PeopleExitX” (where X is one through five) count the number of people exiting at a particular station. This variable helps to the analyst to understand the distribution of popularity of the exits.

There are eight non-integer variables named “vTrainSpeedXtoY” where X and Y correspond to links along the train pathway. These variables are the velocities with which the trains ultimately travel with. The logic that sets the variable values is defined in the Processing Logic section.

The five variables named “DesiredExitX” are used to turn the people entity attribute “aStop” into a global counting variable that the trains can use. The train entities can’t evaluate the attribute of a loaded entity in an IF/THEN statement as required in the Processing Logic. To overcome this problem the DesiredExitX variable acts as a counter to keep track of the locations that the people entities which to debark while they are on the trains.

USER DISTRIBUTIONS

One statistical distributions was defined in the User Distribution section of the model to determine each entities final location. The default distribution that has been defined in the model is given by the following values:

Percentage	Value
20	1
40	2
60	3
80	4
100	5

As it is written each train station should have an even distribution of destinations for the people entities. These values can be modified to reflect a higher preference for some destinations over others.

PROCESSING LOGIC

The processing logic module is where all instructions and definitions are executed. This module is the most important and essentially is where the model is built. The following

description will follow a “person” entity from entrance to exit and detail the various processing steps.

The person entity begins at the bench location. The location serves a dual purpose for entry and exit and therefore the person attribute “aStop” is analyzed here to determine if an entity is inbound to the system or outbound.

```
1.  if aStop > 0
2. then
3. {
4. vBench1 = 1
5. route 1
6. }
7. else
8. route 2
```

If the entity has just arrived then the attribute “aStop” will equal 1 and the entity will be routed to location “TrainStop1” if there is a load request from a train. This means that the person entity will stay at the bench until a train arrives. If the entity is exiting the attribute “aStop” will equal 0 and the entity will leave. Setting the global variable “vBench1” equal to 1 at this steps flags the system that there is a person waiting for a train at stop 1. The variable is set to 0 when the person entity is loaded onto the train. This variable type will cycle back and forth between 1 and 0 many times over the course of a simulation.

Use of the move rule “if Load request” is essential to the realism of the model. The person entity will wait and the move logic will not execute until a Load command is executed by the Train. Once at “TrainStop” the people entity is loaded onto the train and travels with the train until the desired stop.

At the point the people entities load onto the train the following move logic is executed:

```
1. vBench1 = 0
2. if aStop = 2 then
3. inc DesiredExit2
4. else
5. if aStop = 3 then
6. inc DesiredExit3
7. else
8. if aStop = 4 then
9. inc DesiredExit4
10. else
11. if aStop = 5 then
12. inc DesiredExit5
```

This logic will set the global variable vBench1 to a value of 0 indicating there are no longer any passengers waiting at the stop. Next, if the desired stop "aStop" is equal to a particular stop then the global variable "DesiredExitX" is incremented by 1 to indicate to the model that there is a request to exit at that stop. That information is later used in the processing logic to evaluate conditional statements.

The model has been arbitrarily set so that if aDirection = 0 then the train is travelling towards TrainStop5 and if aDirection = 1 then the train is travelling in the opposite direction. At TrainStop1 the train is at the end of the line and therefore can only travel in one direction. The following definition:

1. aDirection = 0

occurs when the train arrives at TrainStop1. On the other end of the line the following definition:

1. aDirection = 1

occurs when the train arrives at TrainStop5. This ensures that each train turns around and travels the other direction when it gets to the end of the line.

The other stand alone statement in the logic is "LinkXtoY = 0." This frees the rail link that was just traversed to allow another train to travel on it.
The processing logic for the end-of-the-line stations (1 and 5) are slightly different then the logic for the middle or intermediate stations (2, 3, and 4). This is because the end stations do not need separate statements for trains travelling to the East or West. The future direction of travel can only be one direction which makes the end-of-the-line stations logic simpler.

Four basic IF/THEN conditions will be evaluated at each station. These four conditions govern all the possibilities that can exist as a train arrives at a station:

1. Passengers are waiting and also want to exit
2. No passengers are waiting but some want to exit
3. Passengers are waiting but none want to exit
4. No passengers are waiting and none want to exit

Within each of the four IF/THEN statements are rules dictating what should happen. For the first condition that passengers are waiting and also want to exit:

```
if DesiredExit1 > 0 and vBench1 = 1
then
        {
        real Y = U(4,1)
        load INF IN Y
        unload INF IFF aStop = 1
```

```
        aTrainSpeed = PathDistance / (HalfHeadway - Y)
        vTrainSpeed1to2 = aTrainspeed
        route 1
        }
else
```

This statement will be evaluated if passengers are waiting for a train and also want to exit at that particular stop. A local variable "Y" takes a number from the uniform distribution between 3 and 5 minutes and then loads all waiting passengers in that time period. Next, all passengers desiring to exit that particular station are unloaded. Finally, the train speed to the next station is set with the attribute "aTrainSpeed" to compensate for the wait time. That attribute is turned into a global variable and the train is routed to exit the station.

If no passengers are waiting but some are wishing to exit, the following logic is executed:

```
if DesiredExit1 > 0 and vBench1 = 0
then
        {
        real Z = U(4,1)
        unload INF IFF aStop = 1
        wait Z
        aTrainSpeed = PathDistance / (HalfHeadway - Z)
        vTrainSpeed1to2 = aTrainspeed
        route 1
        }
else
```

Again in this case the global variable counter DesiredExitX is greater than 0 if a passenger is wishing to exit but unlike the first case vBench will equal 0 if no one is waiting. The local variable Z takes a number from the uniform distribution between 3 and 5 minutes and then unloads all passengers in that time period. As in the first case the attribute and variable for train speed are set using the local variable Z.

In the next case passengers are waiting but none are exiting:

```
if DesiredExit1 = 0 and vBench1 = 1
then
        {
        real A = U(4,1)
        load INF IN A
        aTrainSpeed = PathDistance / (HalfHeadway - A)
        vTrainSpeed1to2 = aTrainspeed
        route 1
        }
else
```

The global counter variable DesiredExitX is equal to 0 if no passengers wish to exit the train yet vBenchX is equal to 1 because at least one People entity is waiting. In this case the local variable A is set to a number between 3 and 5 and the waiting passenger(s) is loaded in that time. As in the other cases the train speed is set to compensate for the load time.

In the final case neither people are waiting nor wish to exit:

```
if DesiredExit1 = 0 and vBench1 = 0
then
        {
        wait vMinStopTime
        aTrainSpeed = PathDistance / (HalfHeadway -
vMinStopTime)
        vTrainSpeed1to2 = aTrainspeed
        route 1
        }
```

Both the global counter variable DesiredExitX and the global integer variable vBenchX are equal to 0 indicating neither people are waiting nor wish to exit. Therefore the train will wait for a minimum time as defined in the global variable vMinStopTime. The train speed is then set to compensate for the time spent at the station.

When a train arrives at an intermediate station (2, 3, or 4) each of the four scenarios described above must be nested into one of two scenarios. The first is if the train is heading one way and the other is the train is heading in the opposite direction. The IF/THEN statement to determine each scenario is:

```
if aDirection = 0 then
…
else

if aDirection > 1 then
…
```

Finally, the route command directs the ProModel logic to read the next steps for each possible direction of a train. The move logic contains expressions for dealing with the case if another train has “booked” the same track. For example,

```
1. if Link2to3 = 1 then
2. begin
3. inc Wait2Ctr
4. wait until Link2to3 = 0
5. Link2to3 = 1
6. move on net1
7. end
```

8. else
9. begin
10. Link2to3 = 1
11. move on net1
12. end

These commands in the output move logic are critical to ensuring that two trains do not pass each other while on the same track. Simple wait commands ensure that the global variable LinkXtoY of the desired destination has the appropriate status of 0 prior to moving the train entity. Once the desired track is free then it is booked by use of the command LinkXtoY = 1.

Once People entities exit the train the attribute aStop is set to equal 0 and the DesiredExitX counter variable is decreased by one. Finally, because the value of aStop is zero the People are routed from the bench to the exit where they are counted by PeopleExitX.

4.4 Verification

Verification of a simulation model can take on many forms. For this particular scenario it is of most interest to ensure that trains do not pass each other on a link between nodes. A visual method was devised to perform this verification.

In the screen-shots captured in Figures 4.4 and 4.5, the current value of each variable LinkXtoY is displayed in the red window. A zero represents the link has no train and can be booked by another. A one represents the link has been booked by a train and only one train may reside on the link.

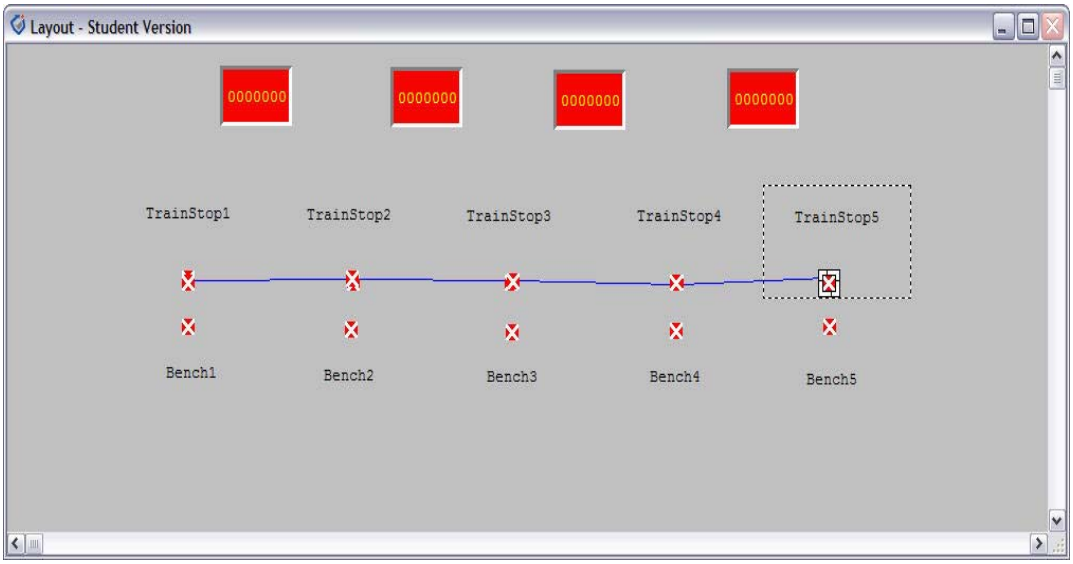

Figure 4.4: Screen-shot with "LinkXtoY" variable

The simulation is run at slow speed with two trains travelling in opposite directions. It is possible to visually verify that the variable LinkXtoY for each appropriate link is booked when a train travels on it.

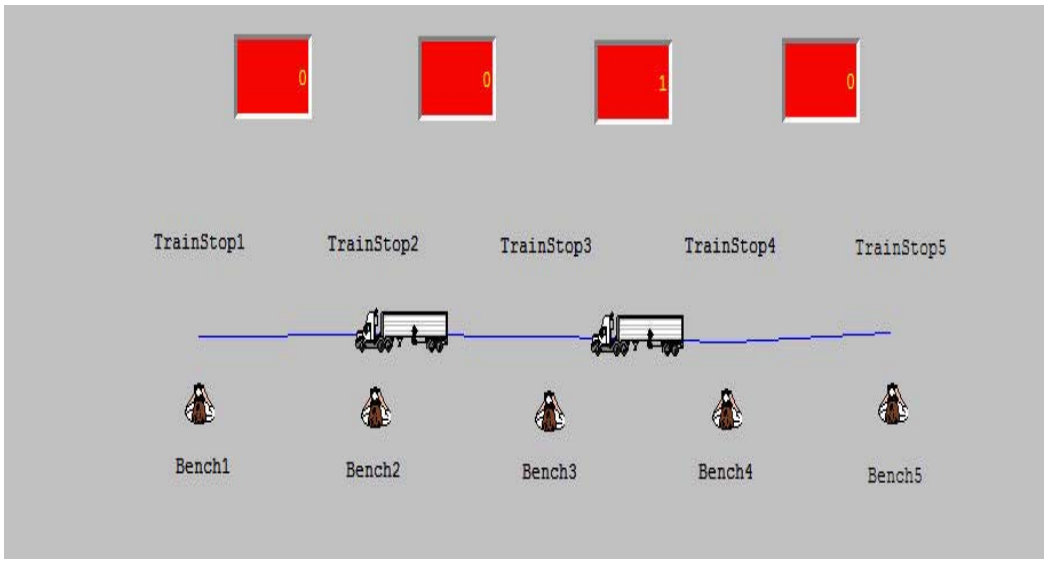

Figure 4.5: Screen-shot with "LinkXtoY" variable

Additional verification was performed by calculating the theoretical travel time for a train traveling in one direction and compared to the model results. Verification of proper distances travelled and overall setup can be achieved by following one train as it travels from station one to station five. The distance between each train station is equal to one mile (5280 feet) and the trains travels at a constant speed of 20 miles per hour. The theoretical time it should take to travel from the first train station to the fifth is given by without including the length of time stopped at each station is given by:

$$4 \text{ (legs)} \times 1 \text{ (mile)} / (20 \text{ miles per hour}) = 0.2 \text{ hour} = 12 \text{ min}$$

By setting the simulation run time at 12 minutes, it is possible to follow one train as it travels from station one to station five. Figure 4.6 is a screen-shot captured at the beginning of the 12-minute simulation run. (ProModel does not come with icons that represent a train or a bus, and, therefore, a train is represented by a truck in this and other figures involving train movement. In addition, ProModel supports few graphics features. The vehicle depicted in Figure 4.6 is moving from the left-hand side of the screen toward the right-hand side, with the blue line indicating future trajectory.)

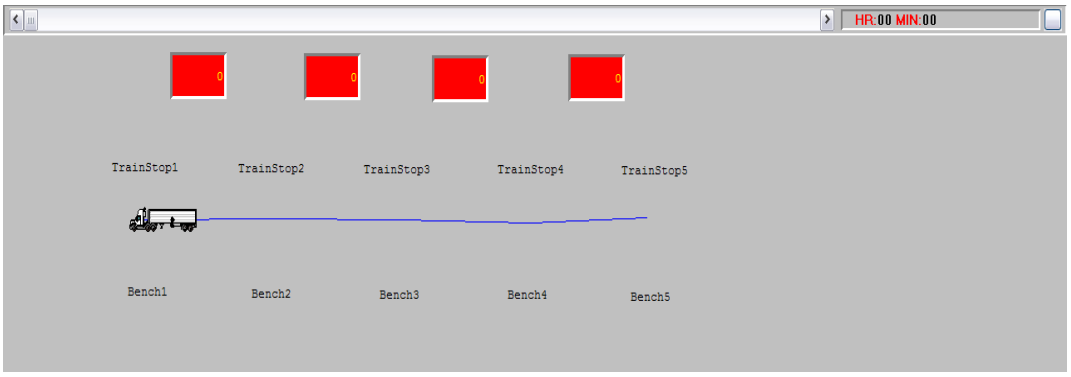

Figure 4.6: Screen-shot of the start of a 12-minute simulation run

The simulation ends after a12-minute run, and the train has travelled to station 5 as shown in Figure 4.7,with the blue line indicating the past trajectory.

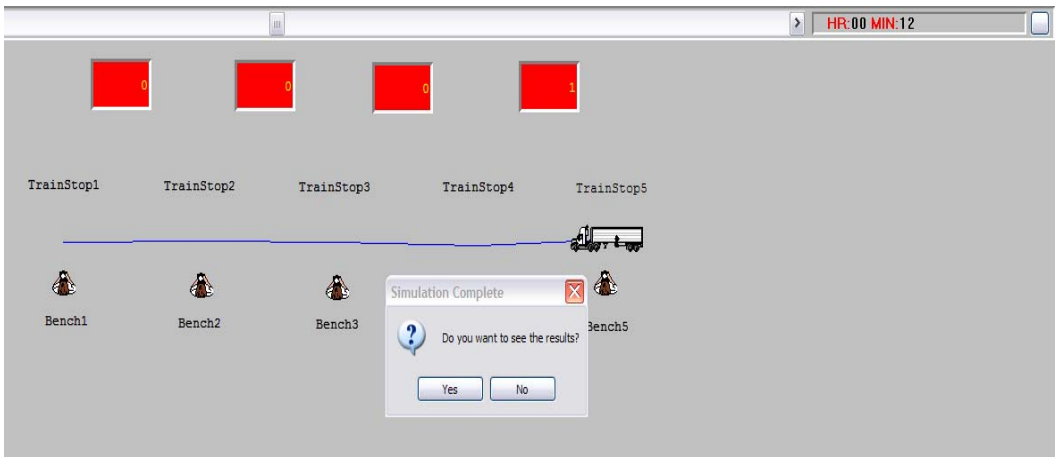

Figure 4.7: Screen-shot of the end of the 12-minute simulation

Furthermore, the use of the variable PeopleExitX provides a means of data collection of the number of people exiting at each station. If the user defined function "StopNo" works properly then there should be an approximately equal number of People entities exiting from each station.

Running the simulation for a 12 hours of model time helps to simulate the number of people who use the system for a normal day. 8 hours of warm-up time are added to the front end of the simulation run-time. The results are summarized in screen-shot captured in Figure 4.8.

General Report (Normal Run - Rep. 1)

General | Locations | Location States Multi | Node Entries | Entity Activity | Entity States | Variables

Bus Thesis_100708_Verification.MOD (Normal Run - Rep. 1)

Name	Total Changes	Avg Time Per Change (MIN)	Minimum Value	Maximum Value	Current Value	Avg Value
Link1to2	233.00	3.07	0.00	1.00	0.00	0.44
Link2to3	235.00	3.06	0.00	1.00	1.00	0.44
Link3to4	235.00	3.06	0.00	1.00	1.00	0.44
Link4to5	233.00	3.07	0.00	1.00	0.00	0.44
PathDistance	0.00	0.00	5280.00	5280.00	5280.00	5280.00
PeopleExit1	699.00	1.02	451.00	1150.00	1150.00	796.81
PeopleExit2	708.00	1.01	487.00	1195.00	1195.00	855.02
PeopleExit3	721.00	1.00	468.00	1189.00	1189.00	815.24
PeopleExit4	763.00	0.94	471.00	1234.00	1234.00	853.98
PeopleExit5	655.00	1.09	454.00	1109.00	1109.00	791.89
Wait1Ctr	0.00	0.00	0.00	0.00	0.00	0.00
Wait2Ctr	0.00	0.00	0.00	0.00	0.00	0.00
Wait3Ctr	14.00	50.46	5.00	19.00	19.00	10.82
Wait4Ctr	0.00	0.00	0.00	0.00	0.00	0.00
Wait5Ctr	13.00	55.29	2.00	15.00	15.00	7.09

Figure 4.8: Results report

The number of people who exit at each station is approximately equal and therefore the "StopNo" function works as expected.

4.5 Analysis

There are many interesting scenarios that can be analyzed with the model. The first part of the analysis will focus on the differences that occur between a single-tracked scenario and a double-tracked one in which the trains can pass each other on the links. This is an important focus if a government agency is looking to implement the single-tracked scheme and is already utilizing an existing double-tracked scheme. Some questions to consider are:

1. How does simulating increased stress on the single-track system impact train speed?

2. What train speeds achieved by the single-track scenario are unrealistic or dangerous for the system?

3. Are there differences in the number of people can be serviced by the dual-track scheme versus the single tracked one?

4. What percent of the total time in system do the People entities spend waiting, travelling, or moving?

5. What happens if some stations are more popular than others instead of a uniform popularity?

To begin to answer these questions the model was modified to simulate a double-tracked scheme. This will provide a "control" example for other experiments with single-track scenarios to be compared to.

Doing this requires deleting the requirement for a constant headway to be maintained, deleting the requirement for each LinkXtoY variable to be checked, and setting a constant speed. With this modified logic a baseline can be established by which various the single-tracked scenarios can be compared.

Trains in the double-track scenario are given a constant speed of 1000 feet per minute to travel between stations 1 mile apart. Although this figure seems slow at only 11.36 miles per hour, it serves as a good figure to compare to the single-track scenario.

The two fictional scenarios are otherwise equal. Both will have trains begin the simulation at opposite ends of the line, utilize the same run time, and utilize the same processing logic. Arrival rates for the People entities at each station were set as a function of the exponential distribution at E(1).

The baseline double-track scenario was run with 8 hours of warm-up time and 12 hours of normal run time with 10 replications. As a comparison several variants of the single-track scenario was run with 8 hours of warm-up time and 12 hours of normal run time with 10 replications. The variable "HalfHeadway" decreased by a minute each time to simulate increased stress on the system. The results for the train speed as a function of headway are shown in Table 4.1.

Table 4.1. Train speed as a function of headway

		Single Track: Maximum Train Speed (fps)				
Segment	**Double-Track**	**Headway: 10 min**	**Headway: 9 min**	**Headway: 8 min**	**Headway: 7 min**	**Headway: 6 min**
1 to 2	1000	1042.88	1291.95	1708.18	2572.27	4947.52
2 to 3	1000	1029.60	1297.90	1703.74	2561.92	4959.99
3 to 4	1000	1044.92	1301.27	1714.16	2532.54	5016.43
4 to 5	1000	1032.00	1301.56	1693.62	2555.42	4974.50
5 to 4	1000	1031.08	1285.88	1720.40	2564.13	4864.61
4 to 3	1000	1033.38	1296.27	1724.87	2574.83	5158.66
3 to 2	1000	1036.47	1294.66	1715.82	2570.41	4893.03
2 to 1	1000	1034.22	1283.42	1703.22	2581.91	5092.56
Average		1035.57	1294.11	1710.50	2564.18	4988.41

A graphical plot of train speed as a function of the headway in a single-track system is shown in Figure 4.0, which is included below again for ease of reading.

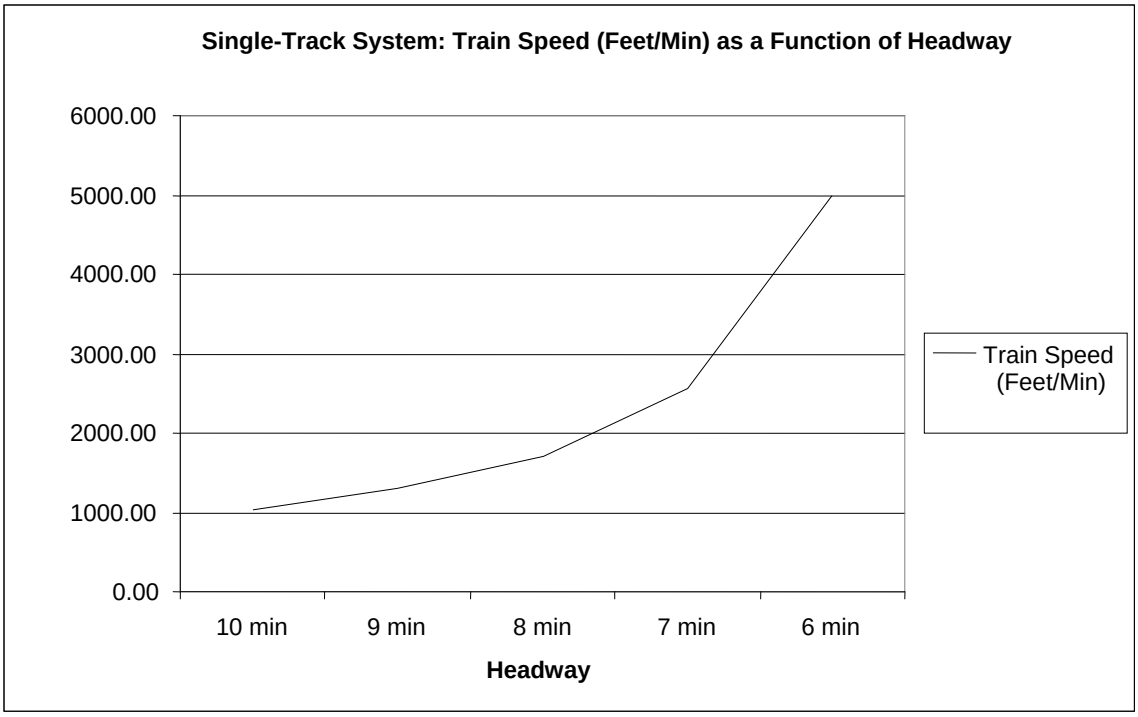

Figure 4.0. The reaction required by the train speeds (feet per minute) in the single-tracked system as headway is decreased.

An interesting result is that as headway is decreased simulating stress on the system, the train speed must increase exponentially to compensate. This is because of the function:

aTrainSpeed = PathDistance / (HalfHeadway - X)

is used to calculate train speed. Given the experimental control has train speed set at a constant 1000 fpm, for this particular system configuration it would probably not be advisable to decrease headway by more than 2 min which corresponds to a doubling in train speed. The results for the number of passengers serviced between the two scenarios during the normal simulation time are shown in Table 4.2.

Table 4.2. Number of passengers serviced between the two scenarios during the normal simulation time

		Single Track: Headway (min)		
People Exit at Station	**Double-Track**	**10**	**8**	**6**
1	734.00	689.00	763.00	733.00
2	748.00	719.00	734.00	745.00
3	696.00	723.00	747.00	737.00
4	700.00	727.00	726.00	738.00
5	708.00	719.00	736.00	710.00
Average	717.20	715.40	741.20	732.60

From this result it is clear that during a 12-hour interval there is no difference in the number of people that are able to be serviced by the single-track system versus the dual track system. The reason there is no difference is because the trains are programmed to adjust speed to maintain headway.

To determine what percent of the total time in system do the People entities spend waiting, travelling, or moving we can look at the "Entity States" screen in the output report. Figure 4.9 shows the break-down of the time spent in system doing various activities between the two extreme data points of a 6-minute headway and a 10-minute headway.

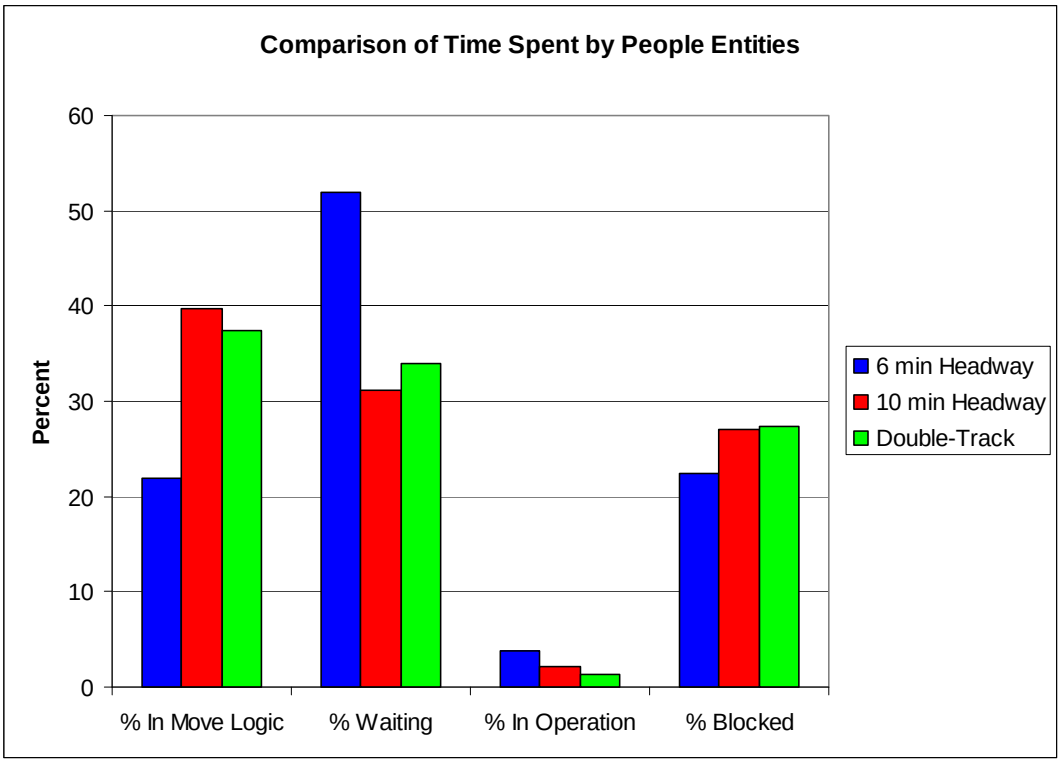

Figure 4.9. Break-down of the time spent in system doing various activities between the two extreme data points of a 6-minute headway and a 10-minute headway

The biggest difference in time spent by the people entities is in the 6-minute headway scenario. People entities tend to spent much less time moving as the trains are moving at a faster rate. The 10-minute headway scenario is very similar to the double-track control scenario.

Additionally parameters used to specify stations can be altered so that they are more popular than others as destinations. For example, the frequency of passenger arrivals to a station can be set to different levels, as shown in column "Frequency" of Figure 4.10.

Arrivals [5]

Entity...	Location...	Qty Each...	First Time...	Occurrences	Frequency	Logic...	Disable
People	Bench1	1		INF	E(10)	aStop = StopNo()	No
People	Bench2	1		INF	E(5)	aStop = StopNo()	No
People	Bench3	1		INF	E(1)	aStop = StopNo()	No
People	Bench4	1		INF	E(5)	aStop = StopNo()	No
People	Bench5	1		INF	E(10)	aStop = StopNo()	No
Train	TrainStop2	1	0	1			No
Train	TrainStop5	1	0	1			No

Figure 4.10: Specification of passenger counts for the five stops in column Frequency

Here the frequency of arrivals is still determined by the exponential statistical distribution but with Station 3 seeing arrivals at a rate of E(1), Stations 2 and 4 at a rate of E(5), and Stations 1 and 5 at a rate of E(10). This is a more realistic scenario as it represents the fact that some stations are located in more populous areas than others. A result of this scenario is that the trains cross at some specific stations and not others. In this example, the trains cross at Stations 2, 4 and 5 according to the positive values of the variable "WaitXCtr",of Figure 4.11 for X=2, 4 and 5.

General Report (Normal Run - Avg. Reps)

General | Locations | Location States Multi | Node Entries | Entity Activity | Entity States | Variables

Train Thesis_102208_Single-Track.MOD (Normal Run - Avg. Reps)

Name	Total Changes	Avg Time Per Change (MIN)	Minimum Value	Maximum Value	Current Value	Avg Value
Link1to2	19.80	33.93	0.00	1.00	0.00	0.04
Link2to3	19.90	33.86	0.00	1.00	0.10	0.04
Link3to4	19.80	35.37	0.00	1.00	0.00	0.04
Link4to5	19.60	34.60	0.00	1.00	0.10	0.04
PathDistance	0.00	0.00	5280.00	5280.00	5280.00	5280.00
PeopleExit1	236.30	2.71	92.20	328.50	328.50	212.25
PeopleExit2	238.90	2.81	106.20	345.10	345.10	229.95
PeopleExit3	232.80	2.89	101.70	334.50	334.50	222.06
PeopleExit4	241.60	2.91	117.90	359.50	359.50	235.62
PeopleExit5	225.60	2.77	87.20	312.80	312.80	204.04
Wait1Ctr	0.00	0.00	0.00	0.00	0.00	0.00
Wait2Ctr	0.10	19.13	0.00	0.10	0.10	0.07
Wait3Ctr	0.00	0.00	0.10	0.10	0.10	0.10
Wait4Ctr	0.10	30.72	0.00	0.10	0.10	0.06
Wait5Ctr	0.10	11.35	0.00	0.10	0.10	0.08

Figure 4.11: Performance report for a more realistic scenario with different levels of destination popularity

There are many more adjustments that can be made to the model depending on a designer's ideas or actual system layout. This model is intended to be a baseline from which to add more layers of realism on a case-by-case basis.

4.6 Conclusions

A single-track train system model was built using the ProModel simulation software to analyze the possibility of removal of a double-track train system.

The model analyzed the use of 5 train stations in a linear fashion. System description, verification, and some interesting results were detailed in the report. Most interesting is that if headway is decreased linearly then the train speed must increase nonlinearly to compensate. Decreasing headway is used to simulate increased stress on the system.

It is important to note that the model should only be used as a base to build a more realistic model depending on actual real-world scenarios.

5. OPERATIONAL RULES OF TRANSIT SIGNAL PRIORITY AND SPEED CONTROL FOR EFFICIENT OPERATION OF AN OPEN SYSTEM

To illustrate the crux of the operation of the proposed system, consider the simple and idealized time-space diagram of Figure 5.1, with the amount of time a bus spends on passenger activities not explicitly shown but absorbed in the bus travel time. This diagram shows how buses of a constant headway *h* traverse through the route between End 1 and End 2, in the same or the opposite directions. Crossing of buses in opposite directions takes place at the "crossing points", and a crossing point may be a stand-alone crossing space dedicated to only such crossing (without any passenger activities) or a combined facility with crossing and passenger activities integrated into one space. There may be more crossing spaces than what are shown as crossing points in the diagram. The additional crossing spaces can be used to help a bus return to schedule after a delay. Any segment of the dedicated lane that spans two adjacent crossing spaces can accommodate traffic in only one direction at any time, with the two opposite directions alternating through time.

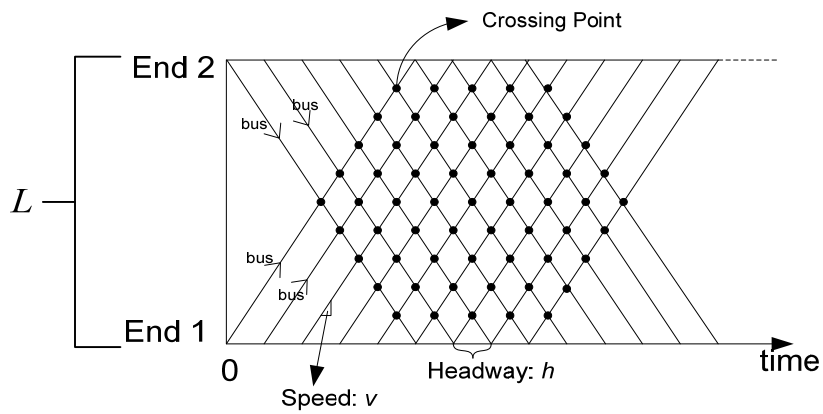

Figure 5.1: A Simple Space-Time Diagram for Study of Operational Performance

The most critical new issues for the MTI project include (a) how to accommodate crossing of buses traveling in opposite directions, which requires one additional lane beyond the one dedicated lane, without any negative impact on the traffic using the regular (i.e., "mixed-flow") lanes, (b) how to accommodate passenger access, boarding and alighting with a minimum amount of space, and (c) whether such accommodation can be provided at a sufficient number of locations in regular distance intervals so as to provide the same or similar level of service that is achievable by the conventional counterparts. The basic idea behind the proposed concept is actually very simple. Bus crossing is accommodated on the otherwise unused or underused median space resulting

from provision of the left-turn lanes. Although not necessary, some left-turn lanes can be sacrificed for bus stops. Crossing spaces can be strategically placed to achieve desired bus headway and travel time. As in the case of a two-dedicated-lane BRT system, bus speed profiles can be adjusted and transit signal priority employed to maximize adherence to the schedule.

A key performance issue is timely crossing of buses traveling in opposite directions. The proposed system requires that a minimum set of crossing spaces be strategically placed so that the travel time T between any adjacent pair of such spaces is the same as that between any other such pair. Consequently, the minimum headway h is equal to twice the constant travel time T, i.e., $h = 2T$. Moreover, any delay of any bus in reaching a crossing space will cause a delay not only in the schedule for buses traveling in its direction but also a delay to the schedule of the buses traveling in the opposite direction.

A key issue is to ensure on-time crossing. In this project, we focus on a simple but the worst case, in which every bus stop is also equipped with a crossing space and there are no other "stand-alone" crossing spaces along the corridor. We will allow the distance between two adjacent bus stops to vary, in order to accommodate real-world variability in the distance. The longer the distance, with everything else assumed to be identical, the faster the required speed.

In this section, we focus on speed-control rules and transit-signal-priority rules for an open system to ensure a green signal for all transit vehicles at any intersection. The dwell time of a transit vehicle at a stop is constant, regardless of the demand, although the length between two consecutive stops may vary. More precisely, we study an open system where the transit traffic traveling on the dedicated system must contend with the traffic signals at the at-grade intersections of the dedicated lanes with cross streets. This study is modular, with two independent modules. The two modules can be combined to develop a transit priority signaling cycle for any intersection along the corridor so that all buses can travel through the corridor without encountering any red light.

We first describe our strategies and approach. Our first strategy is to focus on an arbitrary section between two neighboring bus stops/crossing spaces and to develop transit priority signaling algorithms to ensure a green light for all BRT buses that arbitrary section. Achieving this implies a green light for all BRT buses throughout the corridor. As a second strategy, we consider the problem of transit signal priority for any given intersection along this arbitrary section as an independent problem from its counterparts for any other intersection on the same section. Since all the transit signal priority rules of all the intersections on this section will be developed to ensure unimpeded travel of the buses, these independently developed rules are actually quite inter-dependent.

Now, consider any arbitrary section and any arbitrary intersection of the section. We make a major observation about the operations of the one-dedicated-lane system and motivate our third strategy. By the recurrent nature of the traffic traveling on a section from one period of length h, i.e., the headway, to another, we need to focus on one period

of time of length h. Let us focus on the arbitrary section depicted in Figure 5.2. Treat time 0 as the time when two buses cross at the Stop 1, one being east bound and the other west bound. Note that no buses are present at Stop 2 at time 0. In fact, an important observation is that buses on the dedicated-lane system cross at alternate crossing spaces (i.e., bus stops in this worst case being treated). After $T=h/2$, the east-bound bus will cross with another west-bound bus at Stop 2, and no buses will cross at Stop 1. This is illustrated in Figure 5.3. The crossing taking place on this section will take place in turn between Stop1 and Stop 2. Within time interval $[0,h]$, only two buses will pass each intersection on this section. The first one is the east-bound bus of Figure 5.2. The second is a west bound bus that has not reached Stop 2 at time 0 and hence is not shown on Figure 5.2, but is the west-bound bus shown in Figure 5.3 for $T=h/2$. At $T=h$, buses cross at Stop 1 again, and the same movements are repeated for the next interval of length h, i.e., $[h, 2h]$. These observations can be easily visualized with the simulation-animation episode 1 discussed in Section 2 (Tsao, J.R. and Tsao, H.-S.J., 2008a).

To facilitate conceptual understanding and to simply calculations for the passing times of the two buses with interval $[0,h]$, let us imagine, as the third strategy, the problem as one in which one single bus travels from Stop 1 to Stop 2 and then returns from Stop 2 to Stop 1, after spending a short, fixed amount of dwell time at Stop 2. These imaged movements can be easily visualized with the simulation-animation episode 1 discussed in Section 2 (Tsao, J.R. and Tsao, H.-S.J., 2008a).

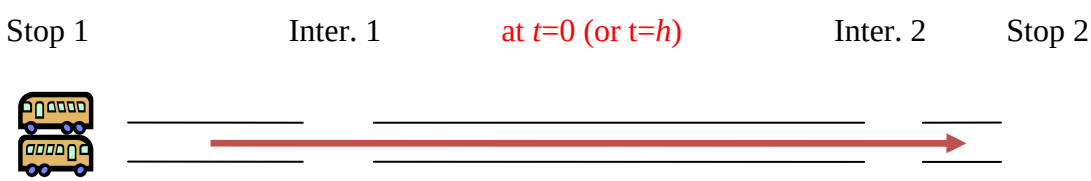

Figure 5.2. Bus locations at $t=0$ (or $t=h$) of a repeating cycle of length h

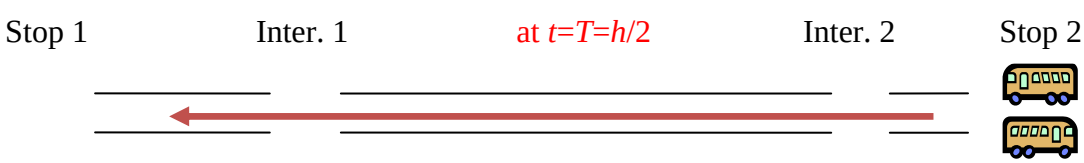

Figure 5.3. Bus locations at $t=h/2$ of a repeating cycle of length h

Consider any arbitrary intersection along any arbitrary section. For the purpose of optimizing transit priority signaling cycle, what one really needs is that difference in their passing times through the intersection. The difference actually has two values: (1) the elapsed time after an east-bound bus passes the intersection and before the next (west-bound) bus passes the intersection and (2) the elapsed time after a west-bound bus passes the intersection and before the next (east-bound) bus passes the intersection. These two values are completely dependent, and they add up to h. We can use either one; however,

we focus on the former, without loss of generality. We expand on the calculation of this elapsed time in Section 5.1.

Given the difference in the two bus passing times through an intersection, we then determine a transit-priority signaling cycle for the intersection so that the two buses pass through the intersection with a green light. In anticipation of stochastic elements influencing the movements of the buses, a major goal is to have a transit priority signaling cycle such that the buses pass through the intersection in the middle of a green phase or close to this middle as much as possible. While it is easy to develop an arbitrary transit priority signaling "macro-cycle" repeating itself from one period of length *h*, e.g., 10 minutes, to another such that the two buses do pass through the intersection right in the middle of a green phase, there are other concerns beyond transit priority, e.g., safety of pedestrians crossing the street on a crosswalk, degree of repetitiveness of the conventional cycle (i.e., a "micro-cycle") of length (e.g., 2 minutes) that the current motorists and pedestrians are used to, traffic demand, etc. In Section 5.2, we propose an algorithm to optimize the transit priority signaling for all the "micro-cycles" constituting a macro-cycle of length *h*.

5.1 The Elapsed Time between Two Consecutive Bus Passages through an Intersection

Consider any arbitrary section and any arbitrary intersection of the section, as illustrated in Figures 5.2 and 5.3. Also, imagine, equivalently, a bus that travels from Stop 1 to Stop 2 and returns to Stop 1, after a dwell time of *D* at Stop 2. The elapsed time after the bus passes the intersection the first time and before it passes the same intersection the second time on its way back from Stop 2 toward Stop 1 can be obtained easily if the inverse relation $t = f^{-1}(x)$ of the position function $x = f(t)$ of the bus location within the section [0, *L*] as a function of time in the interval [0,*h*]. The function $f(t)$ can be easily derived. Since $x = f(t)$ is not a one-to-one function, $t = f^{-1}(x)$ is not an inverse function but only an inverse relation. In fact, it is a "one-to-two" function; for each $x \in [0, L]$ there are two *t*'s that are mapped into this *x*. More precisely, if an intersection is located at *x* within [0,*L*], then the two *t*'s such that *f*(*t*)=*x* are the two times at which the bus passes through the intersection, in absence of any impedance to its travel.

We first define new notation and state assumptions. Note that to derive $f(t)$ for $t \in [0, h]$, i.e., the entire interval of [0,*h*], we only need to derive $f(t)$ for $t \in [0, T \equiv h/2]$, i.e., the first half of the interval [0,*h*], due to the symmetry. The notation defined below pertains to only $t \in [0, T \equiv h/2]$, i.e., the first half of [0,*h*].

Input Parameters:

- L : the length of the section
- D : the amount of dwell time at a stop (after crossing for safety, i.e., beyond time 0).

- R_a: the constant acceleration rate with which a bus accelerates from a stopped position to a constant cruising speed.
- R_d: the constant deceleration rate with which a bus decelerates from the constant cruising speed to a complete stop.

The only decision variable:

- T_a: the amount of time the constant acceleration lasts (beyond the dwell time *D*)

Variables derivable as a function of the only decision variable T^a:

- V_c: the constant cruising speed
- T_c: the amount of time the constant-speed cruising lasts (beyond time $D+T_a$)
- T_d: the amount of time the constant deceleration lasts (beyond time $D+T_a+T_c$ such that $D+T_a+T_c+T_d=T\equiv h/2$)

We assume that the bus accelerates at a constant rate R_a and decelerates at a constant rate R_d. We have tacitly assumed a macro-cycle of length *h* for transit priority signaling in the sense that the signaling cycle at any arbitrary intersection repeats itself after a period of *h*. While this is not mathematically necessary, we make this assumption to ensure the stability of signaling patterns.

Given these parameter values and assumptions, there is exactly one value of T_a that will satisfy the constraint $D+T_a+T_c+T_d=T\equiv h/2$.

The following equations describe the relationships between the derivable variables and the only decision variable T_a:

$$V^c = R_a T_a$$
$$V^c = R_d T_d$$

These imply

$$T_d = \frac{R_a}{R_d} T_a$$

By the requirement $D+T_a+T_c+T_d=T\equiv h/2$, we have

$$T_c = h/2 - D - T_a - \frac{R_a}{R_d} T_a .$$

Finally, the total distance traveled in these amounts of time must be equal to *L*. This leads to

$$\frac{1}{2}R_a T_a^2 + V_c T_c + \frac{1}{2}R_d T_d^2 = L .$$

Therefore, the solution for the unique T_a is the one that satisfies

$$\frac{1}{2}R_a T_a^2 + R_a T_a\left(\frac{h}{2} - D - T_a - \frac{R_a}{R_d}T_a\right) + \frac{1}{2}R_d\left(\frac{R_a}{R_d}T_a\right)^2 = L .$$

We denote this unique T_a as T_a^*. The optimal amount of cruising time is denoted as

$$T_c^* = \frac{h}{2} - D - T_a^* - \frac{R_a}{R_d}T_a^*$$

We are now ready to specify $f(t)$ for $t \in [0, T \equiv h/2]$ as follows.

$$f(t) = \begin{cases} 0 \quad if\ t \in [0, D] \\ \frac{1}{2}R_a(t-D)^2 \quad if\ t \in [D, D+T_a^*] \\ \frac{1}{2}R_a(T_a^*)^2 + R_a T_a^*\left(t - D - T_a^*\right) \quad if\ t \in [D+T_a^*, D+T_a^*+T_c^*] \\ \frac{1}{2}R_a(T_a^*)^2 + R_a T_a^* T_c^* + \frac{1}{2}R_d(t - D - T_a^* - T_c^*)^2 \quad if\ t \in [D+T_a^*+T_c^*, h/2] \end{cases},$$

As for $f(t)$ for $t \in [h/2, h]$, it is simply, due to symmetry:

$$f(t) = L - f\left(t - \frac{h}{2}\right), \quad if\ t \in [h/2, h] .$$

As mentioned earlier, $f(t)$ is not one-to-one in $[0, h]$ and therefore $t = f^{-1}(x)$ is not a function. For mathematical convenience, we can break $f(t)$ into two one-to-one components $f_1(t)$ for $t \in [0, h/2]$ and $f_2(t)$ $t \in [h/2, h]$. With these one-to-one functions, the two times within $[0, h]$ the bus passes through an intersection located at $x \in [0, L]$ are simply $t = f_1^{-1}(x)$ and $t = f_2^{-1}(x)$. As mentioned earlier, the key quantity for designing transit priority signaling cycle is the difference between the two times

$$\Delta \equiv f_2^{-1}(x) - f_1^{-1}(x)$$

or its (equivalent) complement with respect to the length h of the macro-cycle, i.e., $h - [f_2^{-1}(x) - f_1^{-1}(x)]$.

5.2 Algorithm for Optimizing Transit Signal Priority Cycles

The goal of transit priority signaling is to time the signaling cycle for every intersection along the corridor in such a way that when a bus passes through an intersection, the passing time is in the middle of a green phase. However, the need to accommodate the passing of two buses, one in each of the two opposite directions, at any intersection is the source of complexity.

Perhaps the intersection for which the task of designing the cycle is simplest and most straightforward is the intersections bordering a bus stop in the sense that there are only two choices. The two passing times at such an intersection should be placed in either a common green phase or in two consecutive green phases. The decision should depend on the traffic demand, including the level of passenger activities, the levels of through- and crossing traffic at the intersection, etc. In the rest of this section, we focus on the more complex cases, i.e., intersections not bordering a bus stop, and develop a method for designing the transit priority signaling cycles.

We consider an arbitrary intersection. Again, we assume a macro-cycle of length h and focus on any given period of length h. Although the two passing times for the section are dependent and the difference in the two passing times Δ cannot be any given value between 0 and h, the method proposed below accepts any value $\Delta \in [0, h]$ as an input parameter.

We consider two possible configurations of a macro-cycle in terms of how the micro-cycles constitute the macro-cycle; both configurations are intended to maintain as much regularity for the micro-cycles as possible, for the purposes of safety and smooth traffic flow. In the first configuration, a macro-cycle of length h is divided into a number of micro-cycles of equal length. Given the common practices, there are few choices of this number. For example, for a headway or macro-cycle of length h = 10 minutes, there may be four micro-cycles of length 2.5 minutes or five micro-cycles of 2 minutes in most cases. This first configuration is illustrated in Figure 5.4. In the second configuration, we allow one special micro-cycle only, and the other micro-cycles are required to be of equal length. More precisely, the length s of the special micro-cycle is a decision variable. The second configuration is illustrated in Figure 5.5. Note that the former configuration is a special case of the latter.

For each configuration, we first place the two passing times along an interval of length h. The only constraint is that they must be Δ apart, and therefore, there are infinite possible ways to place them. Before addressing the details about how to place the two passing times, we first discuss how to choose the best placements for each of the two configuration and how to choose between the two configurations. Note again that the length of the special micro-cycle is a decision variable. For each of the two configurations, we choose the time placements such that the maximum of the distances of the two placements from the two corresponding centers of the two green phases is the minimum among all possible placements. For the second configuration, we choose with respect to not only the distances from the green-phase centers but also the length s of the

special micro-cycle. We can then simply choose the configuration which leads to a smaller minimum or consider other factors in choosing between the two configurations. Note that if neither minima is acceptable, speed control can be used to increase the acceptability of the two minima.

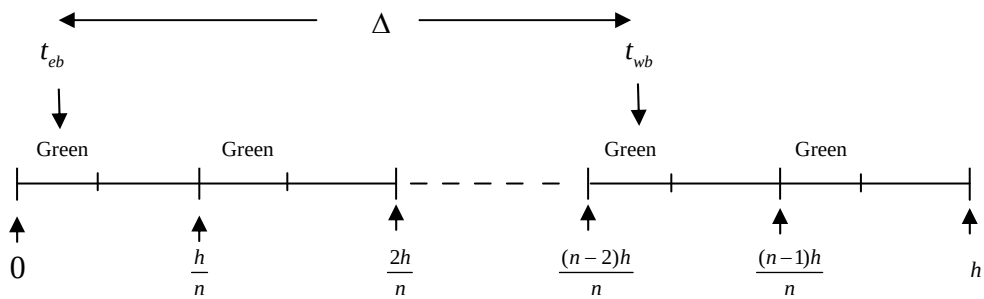

Figure 5.4. A macro-cycle divided into *n* micro-cycle of equal lengths

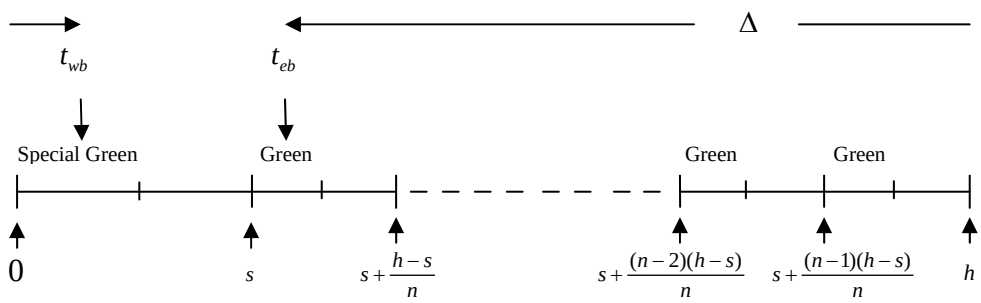

Figure 5.5. A macro-cycle with a special micro-cycle and with the rest of cycle time divided into *n* micro-cycle of equal lengths

We now address how to place the two passing times for any arbitrary intersection of any section and for either of the two configurations. We first define some new notation.

- t_{eb}: the time at which the east-bound bus passes the intersection within the focus period (i.e., macro-cycle) of [0,*h*].
- t_{wb}: the time at which the west-bound bus passes the intersection within the focus macro-cycle of [0,*h*].

We always place t_{eb} first, and it can be placed at any point within the macro-cycle of [0,*h*]. Given the placement of t_{eb}, t_{wb} should be placed at a time that is Δ amount of time beyond t_{eb}, i.e., $t_{wb} = t_{eb} + \Delta$, if and only if $t_{eb} + \Delta \le h$. If $t_{eb} + \Delta > h$, then t_{wb} should be placed at the point that is $t_{eb} + \Delta - h$ beyond time 0.

We specify the placement and optimization procedure for the second configuration in the following algorithm. The procedure for the first one is a special case.

Step 0: Set parameter values: The total number n of regular micro-cycles; the proportion g of the green phase for this intersection; the upper bound S for the length of a special cycle; all possible values s_k, k=1, 2, …, K, as the possible lengths of the special cycle; the amount Δ of time elapsed after the passing of the east-bound bus until the passing of the west-bound bus. Set k = 0, $s^{opt} = \infty$, $M^{opt}_{\min\max} = \infty$ and $t^{opt}_{eb} = \infty$.

Step 1. Set k = k+1. If k = K+1. Stop. Optimal s^{opt} and t^{opt}_{eb} have been found. Otherwise, set s = s_k and go to Step 2.

Step 2. Set the regular micro-cycle length to $\frac{h-s}{n}$ and set the boundary points of the micro-cycles to $\left[s + \frac{(i-1)(h-s)}{n}, s + \frac{i(h-s)}{n}\right]$, i=1,2,…,n. Set the center point of the green phase of the special micro-cycle to $\frac{gs}{2}$ and the n center points of the n regular green phases (of the n corresponding regular micro-cycles) to $s + \frac{(i-1)(h-s)}{n} + \frac{g(h-s)}{2n}$. Go to Step 3.

Step 3. Set $M^{s}_{\min\max} = \infty$. Set $t^{s}_{eb} = \infty$. For every t_{eb} from 0 to h (or a discrete refinement), perform the following:

- Determine the micro-cycle in which t_{eb} falls.
- Determine the micro-cycle in which t_{wb} falls. If $t_{eb} + \Delta \le h$, then $t_{wb} = t_{eb} + \Delta$ and determine the micro-cycle accordingly. If $t_{eb} + \Delta > h$, then $t_{wb} = t_{eb} + \Delta - h$ and determine the micro-cycle accordingly.
- Calculate the distance between t_{eb} and the center of the green phase of its micro-cycle; calculate the distance between t_{wb} and the center of the green phase of its micro-cycle. Set $M_{\max dis}$ to the maximum of the two distances.
- If $M_{\max dis} < M^{s}_{\min\max}$, then set $M^{s}_{\min\max} = M_{\max dis}$ and set $t^{s}_{eb} = t_{eb}$.

Step 4. If $M^{s}_{\min\max} < M^{opt}_{\min\max}$, then set $M^{opt}_{\min\max} = M^{s}_{\min\max}$, $s^{opt} = s_k$ and $t^{opt}_{eb} = t^{s}_{eb}$. Go to Step 1.

5.3 Computer Implementation and Analysis

The JAVA programs implementing the computational algorithm developed in 5.2 can be found in Appendix B. We conducted two numerical studies. In the first one, we use the algorithm to demonstrate the relationship between a particular difference Δ in the two

passing times and the length *s* of the special micro-cycle. In the second, we use the algorithm to demonstrate the relationship between the difference Δ in the two passing times and the optimal length *s* of the special micro-cycle, as well as the relationship between the difference Δ in the two passing times and the minimum of the maximum deviation of the two passing times from the corresponding centers of the green phases.

In both studies, we set the headway *h* to 10 minutes and fix the total number of micro-cycles to 5, with one special micro-cycle and four identical, regular micro-cycles. We also set the green-phase percentage to 60%; this signifies that the BRT system is implemented on an arterial whose through-traffic volume is significantly higher than the volume of crossing traffic. In the first study, we consider three Δ values and calculate, for each of the three Δ values and for a set of *s* values, the minimum of the maximum deviations $M^{s}_{\min\max}$ of the two time placements from the corresponding green-phase centers. The three Δ values are 1 minute, 2 minutes and 5 minutes; the set of *s* values are 1, 1.25, 1.5, 1.75, 2, 2.25, 2.5, 2.75 and 3 minutes. The results are summarized in Table 5.1. Note that when Δ = 2 minutes, the best length of special micro-cycle *s* is two minutes. This makes sense because this length is identical to the lengths of the four regular micro-cycles, all of which are 2 minutes. The two passage times can be placed at the exact middles of (any) two consecutive micro-cycles. Although this 2-minute length *s* of special micro-cycle is the best for Δ = 2 minutes or any other even minutes, it is the worst, among all the 9 *s* values, for odd Δ values, with the minimum maximum deviation $M^{s}_{\min\max}$ being 30 seconds.

Table 5.1. $M^{s}_{\min\max}$ for nine *s* values, associated with Δ = 1, 2 and 5 minutes (in seconds)

Δ value/s value (m)	1	1.25	1.5	1.75	2	2.25	2.5	2.75	3
1 min.	11	16	21	25	30	28	26	24	23
2 min.	8	6	4	2	0	2	4	6	8
5 min.	15	19	23	26	30	25	19	13	8

In the second study, we calculate, for each of nine Δ values, Δ =1, 2, …, 9 minutes, the optimal length s^{opt} of the special micro-cycle and the optimal min-max $M^{opt}_{\min\max}$ of the two time placements from the corresponding green-phase centers. The length s of the special micro-cycle is restricted between 1 and 3 minutes; we consider only discrete values of *s*, with one-second increments. The results are summarized in Table 5.2. While the minimum maximum deviation is unique for each Δ value, multiple optimal lengths of the special micro-cycle may exist, in which case we report only the shortest one. For example, the eleven optimal s^{opt} for Δ = 2 are 110, 111, 114, 115, 117, 118, 119, 120, 121, 122, and 123 seconds. It should not be surprising that 120 seconds is an optimal s^{opt} for Δ = 2, 4, 6 and 8 minutes and that the corresponding minimum maximum deviation is 0. Also note that the worst case among these nine Δ values is 11 seconds. With respect to the 72-second green phase of a 2-minute regular micro-cycle, this 11-second maximum deviation of the two optimal passage times with respect to their green-phase centers implies that the two passage times are within the middle 1/3 of the green

phase. There exist other methods to further reduce this maximum deviation and can be explored if necessary.

Table 5.2. s^{opt} and $M^{opt}_{\min\max}$ for nine Δ values from one minute to nine minutes (in seconds)

Performance/ Δ value (m)	1	2	3	4	5	6	7	8	9
s^{opt}	60	110	60	116	178	116	60	110	60
$M^{opt}_{\min\max}$	11	0	8	0	8	0	8	0	11

6. CONCLUSION

This project extended that deterministic performance study of the MTI project in four complementary directions. Due to the unconventional nature of the concept of one-dedicated-lane BRT or light-rail systems, we developed, in the first direction, two episodes of simulation/animation to facilitate understanding key ideas of the concept. We addressed the backward compatibility issue as the second direction and developed a new geometric configuration for a two-dedicated-lane BRT and a new geometric configuration for a one-dedicated-lane BRT such that the former is backward compatible.

In the third direction, we focused on speed control rules to absorb the impact of the stochastic demand (and the different distances between pairs of neighboring stops) on the performance of a closed system, i.e., with the surrounding traffic ignored or its effect negated. In the fourth direction, we focused on speed-control rules and transit-signal-priority rules for an open system to ensure a green signal for all transit vehicles at any intersection. Our results demonstrated a promising potential of the concept of a one-dedicated-lane BRT or light-rail system for efficient operation, as an end-state or as an intermediate state of a two-dedicated-lane, space-efficient system.

As specified in the proposal of this project, the focus of this performance research was on a simple but the worst case, in which every bus stop is also equipped with a crossing space and there are no other “stand-alone” crossing spaces along the corridor. As pointed out earlier, crossing of buses traveling in two opposite directions can take place at any section where the unused or underused median space is sufficiently long to accommodate such crossing. The more such stand-alone crossing spaces (away from a bus stop), the closer the performance of the one-dedicated-lane BRT or light-rail system to its two-dedicated-lane counterpart. With the availability of such stand-alone crossing spaces, the impact of a delay to a bus, e.g., a delay caused by an excessive number of boarding/alighting passengers or by the extra time required for elderly or disabled persons to board/alight the vehicle, can be more localized, and the delayed bus can catch up with the schedule with real-time transit signal priority. Although the development of this type of real-time transit signal priority for one-dedicated-lane BRT or light-rail systems is beyond the scope of this research project, design of this type of real-time transit signal priority systems, given the current state of the art of intelligent transportation systems in general or transit signal priority in specific, is more of a

development or adaptation effort, rather than a research effort. Details about transit signal priority can be found in Baker et al. (2002). Wang et al. (2008) provided a summary of the state of the art of such real-time transit signal priority and a comprehensive reference list. We believe that the next step in this line of research is to work on, simultaneously, deployment issues with city and transportation planners, acceptance issues with transit users, automobile drivers, other stakeholders (including the general public), design issues for actual deployment sites (based on inputs provided by the planners, transit users, automobile drivers and other stakeholders), and development of site-specific and high-fidelity study tools.

APPENDIX A: PROMODEL CODE OF SIMULATION TOOL FOR STUDY OF IMPACTS OF STOCAHSTIC DEMAND AND VARIABILITY IN STATION SEPARATION

```
Time Units:          Minutes
Distance Units:       Feet


************************************************************************
*                          Locations                                   *
************************************************************************

Name       Cap Units Stats       Rules      Cost
---------- --- ----- ----------- ---------- ------------
TrainStop1 INF 1     Time Series Oldest, ,
TrainStop2 INF 1     Time Series Oldest, ,
TrainStop3 INF 1     Time Series Oldest, ,
TrainStop4 INF 1     Time Series Oldest, ,
TrainStop5 INF 1     Time Series Oldest, ,
Bench1     INF 1     Time Series Oldest, ,
Bench2     INF 1     Time Series Oldest, ,
Bench3     INF 1     Time Series Oldest, ,
Bench4     INF 1     Time Series Oldest, ,
Bench5     INF 1     Time Series Oldest, ,


************************************************************************
*                           Entities                                   *
************************************************************************

Name       Speed (fpm)  Stats       Cost
---------- ------------ ----------- ------------
People     150          Time Series
Train      aTrainSpeed  Time Series


************************************************************************
*                         Path Networks                                *
************************************************************************

Name     Type        T/S            From     To       BI   Dist/Time    Speed Factor
-------- ----------- ---------------- -------- -------- ---- ------------ ------------
Net1     Non-Passing Speed & Distance N1       N2       Bi   PathDistance 1
                                      N2       N3       Bi   PathDistance 1
                                      N3       N4       Bi   PathDistance 1
                                      N4       N5       Bi   PathDistance 1
```

```
*********************************************************************
*                          Interfaces                               *
*********************************************************************

 Net       Node      Location
 ---------- ---------- ----------
 Net1       N3        TrainStop3
            N4        TrainStop4
            N1        TrainStop1
            N2        TrainStop2
            N5        TrainStop5
            N1        Bench1
            N2        Bench2
            N3        Bench3
            N4        Bench4
            N5        Bench5


**********************************************************************
*                           Mapping                                  *
**********************************************************************

 Net       From      To        Dest
 ---------- ---------- ---------- ------------
 Net1       N3        N2
            N3        N4


*********************************************************************
*                          Processing                               *
*********************************************************************

                         Process                  Routing

 Entity   Location   Operation          Blk  Output   Destination Rule     Move Logic
 -------- ---------- ------------------   ---- -------- ----------- -------  ------------
 People   Bench1     //Determine if passenger is outbound
                 if aStop > 0
                 then
                 {
                 vBench1 = 1
                 route 1
                 }
```

```
            else route 2
                        1   People   TrainStop1  LOAD 1   vBench1 = 0
                                          if aStop = 2 then
                                          inc DesiredExit2
                                          else
                                          if aStop = 3 then
                                          inc DesiredExit3
                                          else
                                          if aStop = 4 then
                                          inc DesiredExit4
                                          else
                                          if aStop = 5 then
                                          inc DesiredExit5

                        2   People   EXIT       FIRST 1  inc PeopleExit1
                                          move for 5 sec

People   Bench2     //Determine if passenger is outbound
            if aStop > 0
            then
            {
            vBench2 = 1
            route 1
            }

            else route 2        1   People   TrainStop2  LOAD 1   vBench2 = 0
                                          if aStop = 1 then
                                          inc DesiredExit1
                                          else
                                          if aStop = 3 then
                                          inc DesiredExit3
                                          else
                                          if aStop = 4 then
                                          inc DesiredExit4
                                          else
                                          if aStop = 5 then
                                          inc DesiredExit5
                        2   People   EXIT       FIRST 1  inc PeopleExit2
                                          move for 5 sec

People   Bench3     //Determine if passenger is outbound
            if aStop > 0
            then
            {
            vBench3 = 1
            route 1
```

```
            }

            else route 2        1    People   TrainStop3  LOAD 1   vBench3 = 0
                                                    if aStop = 1 then
                                                    inc DesiredExit1
                                                    else
                                                    if aStop = 2 then
                                                    inc DesiredExit2
                                                    else
                                                    if aStop = 4 then
                                                    inc DesiredExit4
                                                    else
                                                    if aStop = 5 then
                                                    inc DesiredExit5
                        2    People   EXIT       FIRST 1  inc PeopleExit3
                                                    move for 5 sec
People   Bench4     //Determine if passenger is outbound
            if aStop > 0
            then
            {
            vBench4 = 1
            route 1
            }

            else route 2        1    People   TrainStop4  LOAD 1   vBench4 = 0
                                                    if aStop = 1 then
                                                    inc DesiredExit1
                                                    else
                                                    if aStop = 2 then
                                                    inc DesiredExit2
                                                    else
                                                    if aStop = 3 then
                                                    inc DesiredExit3
                                                    else
                                                    if aStop = 5 then
                                                    inc DesiredExit5
                        2    People   EXIT       FIRST 1  inc PeopleExit4
                                                    move for 5 sec

People   Bench5     //Determine if passenger is outbound
            if aStop > 0
            then
            {
            vBench5 = 1
            route 1
            }
```

```
            else route 2
                        1   People   TrainStop5  LOAD 1   vBench5 = 0
                                          if aStop = 1 then
                                          inc DesiredExit1
                                          else
                                          if aStop = 2 then
                                          inc DesiredExit2
                                          else
                                          if aStop = 3 then
                                          inc DesiredExit3
                                          else
                                          if aStop = 4 then
                                          inc DesiredExit4

                        2   People   EXIT       FIRST 1  inc PeopleExit5
                                          move for 5 sec
Train    TrainStop1 //change train direction, 0 is going toward stop 5
            aDirection = 0

            //free previous link
            Link1to2 = 0

            //if passengers are waiting and want to exit:
            if DesiredExit1 > 0 and vBench1 = 1
            then
              {
              real Y = U(4,1)
              load INF IN Y
              timetest1 = Y
              unload INF IFF aStop = 1
              aTrainSpeed = PathDistance / (HalfHeadway - Y)
              vTrainSpeed1to2 = aTrainspeed
              route 1
              }
            else
            //if no passengers are waiting and some are exiting:
            if DesiredExit1 > 0 and vBench1 = 0
            then
              {
              real Z = U(4,1)
              unload INF IFF aStop = 1
              wait Z
              timetest2 = Z
              aTrainSpeed = PathDistance / (HalfHeadway - Z)
              vTrainSpeed1to2 = aTrainspeed
```

```
    route 1
    }
  else
  //if passengers are waiting but none are exiting:
  if DesiredExit1 = 0 and vBench1 = 1
  then
    {
    real A = U(4,1)
    load INF IN A
    timetest3 = A
    aTrainSpeed = PathDistance / (HalfHeadway - A)
    vTrainSpeed1to2 = aTrainspeed
    route 1
    }
  //if no passengers are waiting and none want to exit:
  else
  if DesiredExit1 = 0 and vBench1 = 0
  then
    {
    wait vMinStopTime
    aTrainSpeed = PathDistance / (HalfHeadway - vMinStopTime)
    vTrainSpeed1to2 = aTrainspeed
    route 1
    }

                    1    Train    TrainStop2  FIRST 1  //check if next link is
occupied and if so wait until it is free to reserve it
                                    if Link1to2 = 1 then
                                    begin
                                    inc Wait1Ctr
                                    wait until Link1to2 = 0
                                    Link1to2 = 1
                                    move on net1
                                    end

                                    //if next link is not occupied then reserve
it and move
                                    else
                                    begin
                                    Link1to2 = 1
                                    move on net1
                                    end



 Train    TrainStop2 if aDirection = 0
```

```
then
{
//free previous link
Link1to2 = 0

  //if passengers are waiting and want to exit:
  if DesiredExit2 > 0 and vBench2 = 1
  then
          {
          real Y = U(4,1)
          load INF IN Y
          unload INF IFF aStop = 2
          timetest4 = Y
          aTrainSpeed = PathDistance / (HalfHeadway - Y)
          vTrainSpeed2to3 = aTrainspeed
          route 1
          }
  else
  //if no passengers are waiting and some are exiting:
  if DesiredExit2 > 0 and vBench2 = 0
  then
          {
          real Z = U(4,1)
          unload INF IFF aStop = 2
          wait Z
          timetest5 = Z
          aTrainSpeed = PathDistance / (HalfHeadway - Z)
          vTrainSpeed2to3 = aTrainspeed
          route 1
          }
  else
  //if passengers are waiting but none are exiting:
  if DesiredExit2 = 0 and vBench2 = 1
  then
          {
          real A = U(4,1)
          load INF IN A
          timetest6 = A
          aTrainSpeed = PathDistance / (HalfHeadway - A)
          vTrainSpeed2to3 = aTrainspeed
          route 1
          }
  else
  //if no passengers are waiting and none want to exit:
  if DesiredExit2 = 0 and vBench2 = 0
  then
```

```
            {
            wait vMinStopTime
            aTrainSpeed = PathDistance / (HalfHeadway - vMinStopTime)
            vTrainSpeed2to3 = aTrainspeed
            route 1
            }
}
else
if aDirection > 0
then
{
//free previous link
Link2to3 = 0

  //if passengers are waiting and want to exit:
  if DesiredExit2 > 0 and vBench2 = 1
  then
            {
            real C = U(4,1)
            load INF IN C
            timetest7 = C
            unload INF IFF aStop = 2
            aTrainSpeed = PathDistance / (HalfHeadway - C)
            vTrainSpeed2to1 = aTrainspeed
            route 2
            }
  else
  //if no passengers are waiting and some are exiting:
  if DesiredExit2 > 0 and vBench2 = 0
  then
            {
            real D = U(4,1)
            unload INF IFF aStop = 2
            wait D
            timetest8 = D
            aTrainSpeed = PathDistance / (HalfHeadway - D)
            vTrainSpeed2to1 = aTrainspeed
            route 2
            }
  else
  //if passengers are waiting but none are exiting:
  if DesiredExit2 = 0 and vBench2 = 1
  then
            {
            real F = U(4,1)
            load INF IN F
```

```
                timetest9 = F
                aTrainSpeed = PathDistance / (HalfHeadway - F)
                vTrainSpeed2to1 = aTrainspeed
                route 2
                }
        else
        //if no passengers are waiting and none want to exit:
        if DesiredExit2 = 0 and vBench2 = 0
        then
                {
                wait vMinStopTime
                aTrainSpeed = PathDistance / (HalfHeadway - vMinStopTime)
                vTrainSpeed2to1 = aTrainspeed
                route 2
                }
    }           1   Train   TrainStop3  FIRST 1  //check if next link is
occupied and if so wait until it is free to reserve it
                                        if Link2to3 = 1 then
                                        begin
                                        inc Wait2Ctr
                                        wait until Link2to3 = 0
                                        Link2to3 = 1
                                        move on net1
                                        end

                                        //if next link is not occupied then reserve
it and move
                                        else
                                        begin
                                        Link2to3 = 1
                                        move on net1
                                        end
                    2   Train   TrainStop1  FIRST 1  //check if next link is
occupied and if so wait until it is free to reserve it
                                        if Link1to2 = 1 then
                                        begin
                                        inc Wait2Ctr
                                        wait until Link1to2 = 0
                                        Link1to2 = 1
                                        move on net1
                                        end

                                        else
                                        begin
                                        Link1to2 = 1
                                        move on net1
```

```
                                        end


Train    TrainStop3 if aDirection = 0
             then
             {
             //free previous link
             Link2to3 = 0
               //if passengers are waiting and want to exit:
               if DesiredExit3 > 0 and vBench3 = 1
               then
                       {
                       real Y = U(4,1)
                       load INF IN Y
                       unload INF IFF aStop = 3
                       timetest10 = Y
                       aTrainSpeed = PathDistance / (HalfHeadway - Y)
                       vTrainSpeed3to4 = aTrainspeed
                       route 1
                       }
               else
               //if no passengers are waiting and some are exiting:
               if DesiredExit3 > 0 and vBench3 = 0
               then
                       {
                       real Z = U(4,1)
                       unload INF IFF aStop = 3
                       wait Z
                       aTrainSpeed = PathDistance / (HalfHeadway - Z)
                       vTrainSpeed3to4 = aTrainspeed
                       route 1
                       }
               else
               //if passengers are waiting but none are exiting:
               if DesiredExit3 = 0 and vBench3 = 1
               then
                       {
                       real A = U(4,1)
                       load INF IN A
                       aTrainSpeed = PathDistance / (HalfHeadway - A)
                       vTrainSpeed3to4 = aTrainspeed
                       route 1
                       }
               else
               //if no passengers are waiting and none want to exit:
```

```
  if DesiredExit3 = 0 and vBench3 = 0
  then
          {
          wait vMinStopTime
          aTrainSpeed = PathDistance / (HalfHeadway - vMinStopTime)
          vTrainSpeed3to4 = aTrainspeed
          route 1
          }
}
else
if aDirection > 0
then
{
//free previous link
Link3to4 = 0
  //if passengers are waiting and want to exit:
  if DesiredExit3 > 0 and vBench3 = 1
  then
          {
          real C = U(4,1)
          load INF IN C
          unload INF IFF aStop = 3
          timetest11 = C
          aTrainSpeed = PathDistance / (HalfHeadway - C)
          vTrainSpeed3to2 = aTrainspeed
          route 2
          }
  else
  //if no passengers are waiting and some are exiting:
  if DesiredExit3 > 0 and vBench3 = 0
  then
          {
          real D = U(4,1)
          unload INF IFF aStop = 3
          wait D
          aTrainSpeed = PathDistance / (HalfHeadway - D)
          vTrainSpeed3to2 = aTrainspeed
          route 2
          }
  else
  //if passengers are waiting but none are exiting:
  if DesiredExit3 = 0 and vBench3 = 1
  then
          {
          real F = U(4,1)
          load INF IN F
```

```
                aTrainSpeed = PathDistance / (HalfHeadway - F)
                vTrainSpeed3to2 = aTrainspeed
                route 2
                }
        else
        //if no passengers are waiting and none want to exit:
        if DesiredExit3 = 0 and vBench3 = 0
        then
                {
                wait vMinStopTime
                aTrainSpeed = PathDistance / (HalfHeadway - vMinStopTime)
                vTrainSpeed3to2 = aTrainspeed
                route 2
                }
    }           1   Train   TrainStop4  FIRST 1  //check if next link is
occupied and if so wait until it is free to reserve it
                        if Link3to4 = 1 then
                        begin
                        inc Wait3Ctr
                        wait until Link3to4 = 0
                        Link3to4 = 1
                        move on net1
                        end

                        //if next link is not occupied then reserve
it and move
                        else
                        begin
                        Link3to4 = 1
                        move on net1
                        end

                    2   Train   TrainStop2  FIRST 1  //check if next link is
occupied and if so wait until it is free to reserve it
                        if Link2to3 = 1 then
                        begin
                        inc Wait5Ctr
                        wait until Link2to3 = 0
                        Link2to3 = 1
                        move on net1
                        end

                        else
                        begin
                        Link2to3 = 1
                        move on net1
```

```
                                        end


Train   TrainStop4 if aDirection = 0
            then
            {
            //free previous link
            Link3to4 = 0
              //if passengers are waiting and want to exit:
              if DesiredExit4 > 0 and vBench4 = 1
              then
                      {
                      real Y = U(4,1)
                      load INF IN Y
                      unload INF IFF aStop = 4
                      timetest12 = Y
                      aTrainSpeed = PathDistance / (HalfHeadway - Y)
                      vTrainSpeed4to5 = aTrainspeed
                      route 1
                      }
              else
              //if no passengers are waiting and some are exiting:
              if DesiredExit4 > 0 and vBench4 = 0
              then
                      {
                      real Z = U(4,1)
                      unload INF IFF aStop = 4
                      wait Z
                      aTrainSpeed = PathDistance / (HalfHeadway - Z)
                      vTrainSpeed4to5 = aTrainspeed
                      route 1
                      }
              else
              //if passengers are waiting but none are exiting:
              if DesiredExit4 = 0 and vBench4 = 1
              then
                      {
                      real A = U(4,1)
                      load INF IN A
                      aTrainSpeed = PathDistance / (HalfHeadway - A)
                      vTrainSpeed4to5 = aTrainspeed
                      route 1
                      }
              else
              //if no passengers are waiting and none want to exit:
              if DesiredExit4 = 0 and vBench4 = 0
```

```
  then
        {
        wait vMinStopTime
        aTrainSpeed = PathDistance / (HalfHeadway - vMinStopTime)
        vTrainSpeed4to5 = aTrainspeed
        route 1
        }
}
else
if aDirection > 0
then
{
//free previous link
Link4to5 = 0
  //if passengers are waiting and want to exit:
  if DesiredExit4 > 0 and vBench4 = 1
  then
        {
        real C = U(4,1)
        load INF IN C
        unload INF IFF aStop = 4
        timetest13 = C
        aTrainSpeed = PathDistance / (HalfHeadway - C)
        vTrainSpeed4to3 = aTrainspeed
        route 2
        }
  else
  //if no passengers are waiting and some are exiting:
  if DesiredExit4 > 0 and vBench4 = 0
  then
        {
        real D = U(4,1)
        unload INF IFF aStop = 4
        wait D
        aTrainSpeed = PathDistance / (HalfHeadway - D)
        vTrainSpeed4to3 = aTrainspeed
        route 2
        }
  else
  //if passengers are waiting but none are exiting:
  if DesiredExit4 = 0 and vBench4 = 1
  then
        {
        real F = U(4,1)
        load INF IN F
        aTrainSpeed = PathDistance / (HalfHeadway - F)
```

```
                vTrainSpeed4to3 = aTrainspeed
                route 2
                }
        else
        //if no passengers are waiting and none want to exit:
        if DesiredExit4 = 0 and vBench4 = 0
        then
                {
                wait vMinStopTime
                aTrainSpeed = PathDistance / (HalfHeadway - vMinStopTime)
                vTrainSpeed4to3 = aTrainspeed
                route 2
                }
    }           1   Train   TrainStop5  FIRST 1  //check if next link is
occupied and if so wait until it is free to reserve it
                                        if Link4to5 = 1 then
                                        begin
                                        inc Wait4Ctr
                                        wait until Link4to5 = 0
                                        Link4to5 = 1
                                        move on net1
                                        end

                                        //if next link is not occupied then reserve
it and move
                                        else
                                        begin
                                        Link4to5 = 1
                                        move on net1
                                        end

                    2   Train   TrainStop3  FIRST 1  //check if next link is
occupied and if so wait until it is free to reserve it
                                        if Link3to4 = 1 then
                                        begin
                                        inc Wait4Ctr
                                        wait until Link3to4 = 0
                                        Link3to4 = 1
                                        move on net1
                                        end

                                        //if next link is not occupied then reserve
it and move
                                        else
                                        begin
                                        Link3to4 = 1
```

```
                                        move on net1
                                        end

Train   TrainStop5 //change train direction, 1 is going toward stop 1
        aDirection = 1

        //free previous link
        Link4to5 = 0

        //if passengers are waiting and want to exit:
        if DesiredExit5 > 0 and vBench5 = 1
        then
          {
          real Y = U(4,1)
          load INF IN Y
          unload INF IFF aStop = 5
          timetest14 = Y
          aTrainSpeed = PathDistance / (HalfHeadway - Y)
          vTrainSpeed5to4 = aTrainspeed
          route 1
          }
        else
        //if no passengers are waiting and some are exiting:
        if DesiredExit5 > 0 and vBench5 = 0
        then
          {
          real Z = U(4,1)
          unload INF IFF aStop = 5
          wait Z
          aTrainSpeed = PathDistance / (HalfHeadway - Z)
          vTrainSpeed5to4 = aTrainspeed
          route 1
          }
        else
        //if passengers are waiting but none are exiting:
        if DesiredExit5 = 0 and vBench5 = 1
        then
          {
          load INF
          real A = U(4,1)
          load INF IN A
          aTrainSpeed = PathDistance / (HalfHeadway - A)
          vTrainSpeed5to4 = aTrainspeed
          route 1
          }
        //if no passengers are waiting and none want to exit:
```

```
else
if DesiredExit5 = 0 and vBench5 = 0
then
  {
  wait vMinStopTime
  aTrainSpeed = PathDistance / (HalfHeadway - vMinStopTime)
  vTrainSpeed5to4 = aTrainspeed
  route 1
  }

                    1    Train    TrainStop4  FIRST 1  //check if next link is
occupied and if so wait until it is free to reserve it
                                                  if Link4to5 = 1 then
                                                  begin
                                                  inc Wait5Ctr
                                                  wait until Link4to5 = 0
                                                  Link4to5 = 1
                                                  move on net1
                                                  end

                                                  //if next link is not occupied then reserve
it and move
                                                  else
                                                  begin
                                                  Link4to5 = 1
                                                  move on net1
                                                  end

 People   TrainStop1 dec DesiredExit1
              aStop = 0          1    People   Bench1      FIRST 1
 People   TrainStop2 dec DesiredExit2
              aStop = 0          1    People   Bench2      FIRST 1
 People   TrainStop3 dec DesiredExit3
              aStop = 0          1    People   Bench3      FIRST 1
 People   TrainStop4 dec DesiredExit4
              aStop = 0          1    People   Bench4      FIRST 1
 People   TrainStop5 dec DesiredExit5
              aStop = 0          1    People   Bench5      FIRST 1


*********************************************************************
*                        Arrivals                                   *
*********************************************************************

  Entity   Location   Qty Each   First Time Occurrences Frequency  Logic
  -------- ---------- ---------- ---------- ---------- ---------- ------------
```

```
Train    TrainStop1 1        0        1                aTrainSpeed = 0
                                     aDirection = 0
Train    TrainStop5 1        0        1                aTrainSpeed = 0
                                     aDirection = 1

Train    TrainStop3 1        0        0                aTrainSpeed = 0
                                     aDirection = 1
People   Bench1     1                INF      e(1)     aStop = StopNo()
                                     while aStop = 1 do
                                     {
                                        aStop = StopNo()
                                     }

People   Bench2     1                INF      e(1)     aStop = StopNo()
                                     while aStop = 2 do
                                     {
                                        aStop = StopNo()
                                     }

People   Bench3     1                INF      e(1)     aStop = StopNo()
                                     while aStop = 3 do
                                     {
                                        aStop = StopNo()
                                     }

People   Bench4     1                INF      e(1)     aStop = StopNo()
                                     while aStop = 4 do
                                     {
                                        aStop = StopNo()
                                     }

People   Bench5     1                INF      e(1)     aStop = StopNo()
                                     while aStop = 5 do
                                     {
                                        aStop = StopNo()
                                     }


*********************************************************************
*                          Attributes                               *
*********************************************************************

ID         Type        Classification
---------- ----------- --------------
aStop      Integer     Entity
```

```
aDirection  Integer      Entity
aTrainSpeed Real         Entity


*********************************************************************
*                        Variables (global)                         *
*********************************************************************

ID              Type        Initial value Stats
--------------- ------------ ------------- -----------
Link1to2        Integer     0             Time Series
Link2to3        Integer     0             Time Series
Link3to4        Integer     0             Time Series
Link4to5        Integer     0             Time Series
vBench1         Integer     0             Time Series
vBench2         Integer     0             Time Series
vBench3         Integer     0             Time Series
vBench4         Integer     0             Time Series
vBench5         Integer     0             Time Series
PathDistance    Integer     5280          Time Series
vMinStopTime    Real        0.5           Time Series
Wait1Ctr        Integer     0             Time Series
Wait2Ctr        Integer     0             Time Series
Wait3Ctr        Integer     0             Time Series
Wait4Ctr        Integer     0             Time Series
Wait5Ctr        Integer     0             Time Series
PeopleExit1     Integer     0             Time Series
PeopleExit2     Integer     0             Time Series
PeopleExit3     Integer     0             Time Series
PeopleExit4     Integer     0             Time Series
PeopleExit5     Integer     0             Time Series
HalfHeadway     Real        10            Time Series
vTrainSpeed1to2 Real        0             Time Series
vTrainSpeed2to3 Real        0             Time Series
vTrainSpeed3to4 Real        0             Time Series
vTrainSpeed4to5 Real        0             Time Series
vTrainSpeed5to4 Real        0             Time Series
vTrainSpeed4to3 Real        0             Time Series
vTrainSpeed3to2 Real        0             Time Series
vTrainSpeed2to1 Real        0             Time Series
DesiredExit1    Integer     0             Time Series
DesiredExit2    Integer     0             Time Series
DesiredExit3    Integer     0             Time Series
DesiredExit4    Integer     0             Time Series
DesiredExit5    Integer     0             Time Series
```

```
*********************************************************************
*                              Arrays                               *
*********************************************************************

 ID        Dimensions   Type        Import File Export File Disable       Persist
 ---------- ------------ ------------ ----------- ----------- -------------- -------------------------------
 Arr1      1            Integer                          None          No


*********************************************************************
*                              Macros                               *
*********************************************************************

 ID             Text
 --------------- ------------
 Mac1


*********************************************************************
*                         User Distributions                        *
*********************************************************************

 ID        Type        Cumulative   Percentage   Value
 ---------- ------------ ------------ ------------ ------------
 StopNo    Discrete     Yes          20           1
                        40           2
                        60           3
                        80           4
                        100          5
```

APPENDIX B: JAVA COMPUTER PROGRAMS IMPLEMENTING THE TRANSIT SIGNAL PRIORITY RULES

```
//THIS CLASS STORES INFORMATION ABOUT OPTIMAL
//EAST BOUND TIME - WEST BOUND TIME – MAX DEVIATION FROM THE
//TWO GREEN-PHASE CENTERS, WITHIN A MACRO-CYCLE AND GIVEN A
//DELTA, WHICH IS HARD-CODED
//time.java
public class Time {
	private int teb;
	private int twb;
	private int dmax;
	public Time(int teb, int twb, int dmax) {
		super();
		this.teb = teb;
		this.twb = twb;
		this.dmax = dmax;
	}
	public int getTeb() {
		return teb;
	}
	public int getTwb() {
		return twb;
	}
	public int getDmax() {
		return dmax;
	}
	public void setTeb(int teb) {
		this.teb = teb;
	}
	public void setTwb(int twb) {
		this.twb = twb;
	}
	public void setDmax(int dmax) {
		this.dmax = dmax;
	}
}
```

```
//THIS CLASS STORES INFORMATION ABOUT TRAFFIC SIGNAL PHASES.
//IT CONSISTS OF GREEN LIGHT-RED LIGHT START AND STOP TIMES AND
//CENTER OF GREEN PHASE AS WELL.
//midcycle.java
public class midCycle {
	private int startTime, stopTime, greenStart, greenStop, greenMid,
redStart, redStop;

	public midCycle(){
		startTime = 0;	stopTime = 0;
		greenStart = 0;	greenStart = 0;
		greenMid = 0;	redStart = 0;
		redStop = 0;
	}

	public void setTimes(int startTime, int stopTime, int
greenPercentage){
		this.startTime = startTime;
		this.stopTime = stopTime;
		this.greenStart=startTime;
		this.greenStop=greenStart+greenPercentage*(stopTime-
startTime)/100;
		this.redStart=this.greenStop;
		this.redStop= stopTime;
		this.greenMid = greenStart + (greenStop-greenStart)/2;
	}

	public int getStartTime(){
		return (startTime);
	}

	public int getStopTime() {
		return stopTime;
	}

	public int getGreenStart() {
		return greenStart;
	}

	public int getGreenStop() {
		return greenStop;
	}

	public int getGreenMid() {
		return greenMid;
	}

	public int getRedStart() {
		return redStart;
	}

	public int getRedStop() {
		return redStop;
	}

}
```

```
//BRT.java
import java.io.FileNotFoundException;
import java.io.FileOutputStream;
import java.io.IOException;
import java.util.ArrayList;

import org.apache.poi.ss.usermodel.Cell;
import org.apache.poi.ss.usermodel.Row;
import org.apache.poi.ss.usermodel.Sheet;
import org.apache.poi.ss.usermodel.Workbook;
import org.apache.poi.xssf.usermodel.XSSFWorkbook;

//THIS CLASS CALCULATES THE OPTIMAL PLACEMENT TIMES BASED ON GIVEN
INPUT
//AND DENOTES THE MAX DEVIATION AS MAX(D1,D2)
//THE PAIR OF TWO PLACEMENT TIMES THAT HAS THE MINIMUM MAX DEVIATION
//WILL BE CONSIDERED AS THE BEST PAIR.
//THE RESULT WILL BE STORED IN EXCEL FILE "WORKBOOK.XLSX"

public class BRT {
      private ArrayList<ArrayList<midCycle>> microCycle;
      private ArrayList<Time> allTime;

      int startTime, stopTime, retTime;
      int eastBound, westBound, deviation;

      FileOutputStream out = new FileOutputStream("workbook.xlsx");
      Workbook wb = new XSSFWorkbook();
      Sheet sh;
      Row r = null;
      Cell c = null;

      public BRT() throws FileNotFoundException{
            microCycle = new ArrayList<ArrayList<midCycle>>();
            allTime = new ArrayList<Time>();
      }

      public void setMicroCycle(ArrayList<ArrayList<midCycle>>
microCycle) {
            this.microCycle = microCycle;
      }

      public int getSizeMicroCycle() {
            return microCycle.size();
      }

      public void insertNode(int spGreenPerc, int greenPercentage, int
h, int n){
            ArrayList<midCycle> temp = new ArrayList<midCycle>();
            midCycle firstNode = new midCycle();
            firstNode.setTimes(0, spGreenPerc, greenPercentage);
            temp.add(firstNode);

            for (int i = 1; i <= n; i++)
            {
                  startTime = temp.get(temp.size()-1).getStopTime();
                  stopTime = spGreenPerc + i*(h-spGreenPerc)/n;
```

```
                midCycle node = new midCycle();
                node.setTimes(startTime, stopTime, greenPercentage);
                temp.add(node);
            }
            microCycle.add(temp);
        }

        public void setTimePair (int h, int delta, int num)
        {
            allTime.clear();
            for (int i = 0; i < h; i++)
            {
                int tEB = i;
                int tWB = tEB + delta;
                int deviation1 = 0;
                int deviation2 = 0;

                outerLoop:
                for (int j = 0; j < microCycle.get(num).size(); j++)
                {
                    int start =
microCycle.get(num).get(j).getStartTime();
                    int end =
microCycle.get(num).get(j).getStopTime();
                    int greenStop =
microCycle.get(num).get(j).getGreenStop();
                    if (tEB>= start && tEB <= end)
                    {
                        if(tEB <= greenStop)
                        {
                            deviation1 =
Math.abs(microCycle.get(num).get(j).getGreenMid() - tEB);
                            for (int k = 0; k <
microCycle.get(num).size(); k++)
                            {
                                int greenStopTwo =
microCycle.get(num).get(k).getGreenStop();
                                if(tWB > h){
                                    tWB = tWB -h;
                                    k--;
                                    continue;
                                }
                                    if(tWB <= greenStopTwo)
                                    {
                                        deviation2 =
Math.abs(microCycle.get(num).get(k).getGreenMid()-tWB);
                                        int dev =
Math.max(deviation1, deviation2);
                                        Time t = new Time
(tEB,tWB,dev);
                                        allTime.add(t);
                                        break outerLoop;
                                    }

                            }
                        }
                    }
```

```
                }
            }
        }

        public int[] getMinDeviation () //choosing the pair which has min
deviation value
        {
            Time min = null;
            int minDev = 1000000000;
            for (int i = 0; i < allTime.size(); i++)
            {
                int m = allTime.get(i).getDmax();
                if (m < minDev)
                {
                    min = allTime.get(i);
                    minDev = m;
                }
            }
            return new int [] {min.getTeb(), min.getTwb(),
min.getDmax()};
        }

        /*public void print ()
        {
            System.out.println("RESULT :");
            for (int i = 0 ; i < microCycle.size(); i++)
            {
                System.out.println("Start : "+
microCycle.get(i).getStartTime()+" Green Time Ends:
"+microCycle.get(i).getGreenStop()+" Stop:
"+microCycle.get(i).getStopTime());
            }
            //System.out.printf("Best East-West Bound Time : " +
getMinDeviation());
        }*/

        //write to excel file
        public void writeToExcel(int rowNum) throws IOException,
NumberFormatException{
            int cellNum;
            r = sh.createRow(rowNum);
                for (cellNum = 0; cellNum < 5; cellNum++){
                    c = r.createCell(cellNum);
                    /*if (rowNum == 0)
                  {
                    if(cellNum == 0)
                        c.setCellValue("s");
                    else if(cellNum == 1)
                        c.setCellValue("East Bound Time");
                    else if(cellNum == 2)
                        c.setCellValue("West Bound Time");
                    else if(cellNum == 3){
                        c.setCellValue("Deviation");
                    }
                  }
                  else
                  {*/
```

```
                        if(cellNum == 0)
                              c.setCellValue(rowNum+60);
                        else if(cellNum == 1)
                              c.setCellValue(getMinDeviation()[0]);
                        else if(cellNum == 2)
                              c.setCellValue(getMinDeviation()[1]);
                        else if(cellNum == 3)
                              c.setCellValue(getMinDeviation()[2]);
                  //}
            }
      }

      public void createNewExcelSheet(int n){
            sh = wb.createSheet((String.format("Delta=%d",n)));
      }
}
```

```
//Main.java
import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;

public class Main {

	/**
	 * @param args
	 * @throws IOException
	 * @throws NumberFormatException
	 */
	public static void main(String[] args) throws
NumberFormatException, IOException {
		// TODO Auto-generated method stub
		BufferedReader br = new BufferedReader(new
InputStreamReader(System.in));
		int totalTime, numOfuCycle, greenPerc, spGreenPerc;

	System.out.print("1. Enter Total Time (in mins)		:
"); totalTime=Integer.parseInt(br.readLine())*60;
	System.out.print("2. Enter Number of Micro Cycle		:
"); numOfuCycle=Integer.parseInt(br.readLine());
	System.out.print("3. Enter Green Light (in %)		:
"); greenPerc=Integer.parseInt(br.readLine());
	//System.out.print("4. Enter Special Green Light Time (in mins):
"); spGreenPerc=Integer.parseInt(br.readLine())*60;
	//System.out.print("4. Enter Time Interval (in mins)		:
"); interval = Integer.parseInt(br.readLine())*60;

	BRT myBRT = new BRT();
	for(int interval = 1; interval < 10; interval++){	//THIS
FOR LOOP WILL DO ITERATION FOR DELTA = 0 - 9
		myBRT.createNewExcelSheet(interval);
		for(spGreenPerc=60; spGreenPerc<=180; spGreenPerc++){

	myBRT.insertNode(spGreenPerc,greenPerc,totalTime,numOfuCycle);
			myBRT.setTimePair(totalTime, interval*60,
spGreenPerc-60);
			myBRT.writeToExcel(spGreenPerc-60);
		}
	}

	//myBRT.print();

	myBRT.wb.write(myBRT.out);
	myBRT.out.close();
	}
}
```

REFERENCES

AC Transit (2007), *AC Transit East Bay Bus Rapid Transit Project in Alameda County – Environmental Impact Statement/Environmental Impact Report*, Alameda-Contra Costa Transit District (May 2007). http://www2.actransit.org/news/articledetail.wu?articleid=42622c20&r=n#online (accessed on March 6, 2009).

AC Transit (2008), *AC Transit East Bay Bus Rapid Transit Project in Alameda County - Public Hearing Presentation*, Alameda-Contra Costa Transit District (June 2007). http://www.actransit.org/pdf/BRT_Boards_07.pdf?PHPSESSID=37c4b93aa1f0883b6dfd1bae43e35776 (accessed on March 6, 2009).

Baker, R.J., Collura, J., Dale, J.J., Head, L., Hemily, B., Ivanovic, M., Jarzab, J.T., McCormick, D., Obenberger, J., Smith, L., and Stoppenhagen, G.R. 'An overview of transit signal priority'. Intelligent Transportation Society of America, Washington, D.C., 2002.

Breakthrough Technologies Institute, "Eugene, Oregon EmX (as of September, 2007)," accessed at http://www.gobrt.org/Eugene.html on March 6, 2009.

Carey, G. "Bus Rapid Transit – The Eugene – Springfield, OR, USA, Experience," *ITE Journal*, (2006): 20 – 23.

Diaz, R.B., Chang, M., Darido, G., Kim, E., Schneck, D., Hardy, M., Bunch, J., Baltes, M., Hinebaugh, D., Wnuk, L., Silver, F., and Zimmerman, S. *Characteristics of Bus Rapid Transit for Decision Making*. Office of Research, Demonstration and Innovation, Federal Transit Administration, U.S. Department of Transportation, August 2004.

Jarzab, J.T., Lightbody, J., and Maeda, E. "Characteristics of Bus Rapid Transit Projects: An Overview," *Journal of Public Transportation* Vol. 5, No. 2 (2002) 31- 46.

Levinson, H.S., Zimmerman, S., Clinger, J. and Rutherford, C.S. "Bus Rapid Transportation: An Overview," *Journal of Public Transportation* Vol. 5, No. 2 (2002) 1–30.

Lin, Z. and, Wu, J., " Summary of the Application Effect of Bus Rapid Transit at Beijing South-Centre Corridor of China," *Journal of Transportation Systems Engineering and Information Technology, Volume 7, Issue 4, Pages 137-142, August 2007*.

Tsao, J.R. and Tsao, H.-S.J. (2008a), Computer-generated Animation for Illustration of Crossing of Two Opposing Streams of Buses on a One-dedicated-lane Bus Rapid Transit System, accessed on Sept. 30, 2009 at http://www.engr.sjsu.edu/jtsao/papers/ISDSI-2009-one-dedicated-lane-BRT-animation-1.swf .

Tsao, J.R. and Tsao, H.-S.J. (2008b), Computer-generated Animation for Illustration of Crossing of Two Buses at a Bus Stop of a One-dedicated-lane Bus Rapid Transit System, accessed on September 30, 2009 at http://www.engr.sjsu.edu/jtsao/papers/ISDSI-2009-one-dedicated-lane-BRT-animation-2.swf .

Tsao, H.-S. J., Wei, W., and Pratama, A., (2007), "Operational Feasibility of One-dedicated-lane Bus Rapid Transit/Light-rail Systems," presented at the *7th International Conference of Chinese Transportation Professionals*, May 21-22, 2007, Shanghai, China; published in the Proceedings.

Tsao, H.-S. J., Wei, W., and Pratama, A. (2009a), *Feasibility of One-Dedicated-Lane Bus Rapid Transit/Light-Rail Systems and Their Expansion to Two-Dedicated-Lane Systems: A Focus on Geometric Configuration and Performance Planning*, MTI Report No. 08-01, Mineta Transportation Institute, San Jose State University, San Jose, California, U.S.A.; accessed at http://www.transweb.sjsu.edu/MTIportal/research/publications/documents/BRT.pdf pdf on Sept. 7, 2009.

Tsao, H.-S. J., Wei, W., and Pratama, A (2009b), "Operational Feasibility of One-dedicated-lane Bus Rapid Transit/Light-rail Systems," *Transportation Planning and Technology*, Vol. 32, No. 3, pp. 239-260, 2009.

Tsao, H.-S.J., Wei, W., Pratama, A., and Tsao, J.R (2009c), "Launching Bus Rapid Transit with Only One Dedicated Lane for Two-Way Bus Traffic on Congested Corridors," presented at the *2nd Annual Conference of Indian Subcontinent Decision Science Institute* (ISDSI 2009) held on Jan. 3-5, 2009 in Mumbai, India and published in the Proceedings, pp. 113-125.

Tsao, H.-S.J., Wei, W., and Pratama A. (2009d), "Efficient Space Dedication to Bus Rapid Transit or Light Rail Systems," submitted to *Journal of Transportation Engineering* for possible publication.

Wang, Y, Hallenbeck, M., Zheng, J., Zhang, G., Corey, J., and Ma, X., *Comprehensive Evaluation of Transit Signal Priority System Impacts Using Field Observed Traffic Data*, Final Technical Report, Agreement T2695, Task 60, Washington State Transportation Center (TRAC), University of Washington, Box 354802, 1107 NE 45th Street , Seattle, Washington 98105-4631, June 2008.