

**ADAPTIVE TRANSIT SIGNAL PRIORITY BASED ON
REINFORCEMENT LEARNING, CONNECTED VEHICLES, AND
SOFTWARE IN THE LOOP SIMULATION**

A Dissertation
Presented to
The Academic Faculty

by

Dickness Kakitahi Kwesiga

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy in the
School of Civil and Environmental Engineering

Georgia Institute of Technology
May 2025

COPYRIGHT © 2025 BY DICKNESS KAKITAH KYESIGA

**ADAPTIVE TRANSIT SIGNAL PRIORITY BASED ON
REINFORCEMENT LEARNING, CONNECTED VEHICLES, AND
SOFTWARE IN THE LOOP SIMULATION**

Approved by:

Dr. Angshuman Guin, Co-Advisor
School of Civil and Environmental
Engineering
Georgia Institute of Technology

Dr. Michael Hunter, Co-Advisor
School of Civil and Environmental
Engineering
Georgia Institute of Technology

Dr. Randall Guensler
School of Civil and Environmental
Engineering
Georgia Institute of Technology

Dr. Samuel Coogan
School of Electrical and Computer
Engineering
Georgia Institute of Technology

Dr. Michael Rodgers
Oak Ridge National Laboratory

Date Approved: April 23, 2025

ACKNOWLEDGEMENTS

I kept postponing writing this section of my dissertation because, first, I felt it would be impossible to fully acknowledge everyone who has contributed to this work, and second, because words simply fall short of expressing the depth of gratitude I owe them. What follows is simply my earnest attempt.

First, I would like to express my gratitude to my advisors, Dr. Michael Hunter and Dr. Guin Angshuman, from whom I have had the pleasure of learning over the past four years. I am deeply thankful for their mentorship, the supportive environment they cultivated, and the opportunities they provided that allowed me to grow into an independent researcher. Their support extended far beyond the lab, encompassing both my personal well-being and professional development.

I would like to extend my sincere thanks to Dr. Randall Guensler and Dr. Michael Rodgers not only for serving on my dissertation committee but also for their valuable support and guidance throughout my time at Georgia Tech. Their insights encouraged me to broaden my perspective and explore my research questions beyond the boundaries of traffic operations. I am also deeply grateful to Dr. Samuel Coogan for his guidance, particularly on the reinforcement learning aspects of my dissertation. I truly appreciated his willingness to sit down with me on multiple occasions when I encountered challenges. Instead of a dissertation committee, I ended up with a team of dedicated advisors.

I want to thank my senior colleagues – Dr. Suyash Vishnoi, Dr. Saroj Abhilasha, Dr. Somdut Roy, Dr. Nishu Choudhary, Dr. Racheal Panik, Dr. Fan Huiying from whom I

learned a great deal and who offered guidance at various stages of my work. I also want to sincerely thank my past and present lab mates, who became close friends along the way including Adair Garrett, Gopikrishnan Suresh, Shiva Golzari, Matteo Saracco, Juwon Drake, Garyoung Lee, Seung Choi, Christian Douglas, Abraham Pizano, and many others. Your friendship and support have meant so much to me. Finally, I want to express my heartfelt appreciation to my longtime friends back home in Kampala, with whom I've maintained strong and cherished bonds throughout these six years away

Fully aware that I owe everything to my parents, I must highlight that their greatest gift to me was their love for education. From the very beginning of my secondary school, my dad was already guiding me towards the path of becoming a professor. I want to thank my siblings, Annet, Hope, Abia, Rodgers, Macklean, Darlen, Edison, and Simon with whom I deeply share the values of hard work, discipline, ambition, resilience, and mutual love, all instilled in us by our parents. I wouldn't be where I am today without the example set by my siblings, and I couldn't have completed my graduate school journey without their steadfast support and encouragement. Lastly, I want to express my deep gratitude to my girlfriend, Sheila Ampa, whose unwavering love and support have been a constant source of motivation, even from across the "pond". Sheila, thank you and I love you.

TABLE OF CONTENTS

Acknowledgements.....	iii
List of tables.....	x
acronyms.....	xvi
Chapter 1 Introduction.....	1
1.1 Background.....	1
1.2 Problem definition.....	4
1.3 Research objectives.....	8
1.4 Research scope.....	9
1.5 Dissertation organization.....	10
Chapter 2 Literature review.....	12
2.1 Overview.....	12
2.2 TSP systems.....	12
2.3 TSP Performance Measures.....	15
2.4 Arrival time prediction.....	18
2.5 Impact of dwell-time on TSP performance.....	20
2.5.1 Dwell-time data.....	20
2.5.2 Integrating dwell-time in TSP design.....	22
2.6 Impact of background signal timing on TSP performance.....	24
2.7 Integrating CV data in TSP algorithms.....	25
2.8 Reinforcement learning based signal control and TSP.....	26
2.8.1 Single agent reinforcement learning for traffic signal control.....	27
2.8.2 Multi agent reinforcement learning for traffic signal control.....	32
2.8.3 Single agent RL-based TSP.....	37
2.8.4 Multi-intersection/coordinated transit signal priority.....	38
2.9 Literature review summary.....	41
Chapter 3 Exploration of TSP fundamental principles.....	44
3.1 Introduction.....	44
3.1.1 Background.....	44
3.1.2 Objectives and Significance.....	45

3.2	Methodology.....	46
3.2.1	RBC- TSP Algorithm.....	46
3.2.2	Test network.....	49
3.2.3	Simulation Execution.....	52
3.2.4	Experiment design	53
3.3	Results	56
3.3.1	Measures of Effectiveness	56
3.3.2	Comparing impacts of GE and EG	56
3.3.3	TSP performance under different demand levels.....	59
3.3.4	Impact of cycle Length on TSP Performance	64
3.4	TSP Exploration Summary	69
Chapter 4	Impact of dwell-time on the performance of transit signal priority	71
4.1	Background.....	71
4.2	Problem definition and study objectives	72
4.3	Analysis of APC Data.....	73
4.3.1	Data Description	73
4.3.2	Data Pre-processing	74
4.3.3	Data Analysis Results	75
4.4	Impact of Dwell-time on TSP performance	82
4.4.1	Overview.....	82
4.4.2	Test Network.....	83
4.4.3	Experiment design	85
4.4.4	TSP performance analysis at far-side bus stop	87
4.4.5	Comparing TSP performance for far-side bus stops and near-side bus stops.	
	93	
4.5	Dwell-time Summary	96
Chapter 5	single intersection TSP based on reinforcement learning.....	99
5.1	Background.....	99
5.2	Methodology.....	101
5.2.1	Overview.....	101
5.2.2	Reinforcement learning overview.....	102

5.2.3	RL agent formulation and Architecture	108
5.2.4	Simulation Execution.....	120
5.3	Results and Discussion	121
5.3.1	Performance of DDQN – SC	121
5.3.2	Performance of DDQN-TSP.....	124
5.4	Single Intersection RL-based TSP Summary	127
Chapter 6 Adaptive transit signal priority based on multi-agent reinforcement learning		
128		
6.1	Background.....	128
6.2	Methodology.....	129
6.2.1	Multi-agent reinforcement learning overview	129
6.2.2	General traffic control Agents formulation.....	135
6.2.3	TSP control Agents.....	141
6.2.4	Case study	144
6.3	Results	148
6.3.1	Performance of VDN based MARL for general traffic control.....	148
6.3.2	Independent TSP agents.....	151
6.3.3	Coordinated TSP agents.....	154
6.4	MARL-based TSP Summary.....	157
Chapter 7 Corridor level adaptive reinforcement learning based signal Control		
159		
7.1	Background.....	159
7.2	Methodology.....	161
7.2.1	Proximal Policy Optimization (PPO) Overview.....	161
7.2.2	Comparing PPO and DQN convergence properties.....	164
7.2.3	Definition of state, action and reward functions.....	166
7.2.4	Case study	172
7.3	Results	184
7.3.1	Model training.....	184
7.3.2	MA-PPO vs field implemented coordinated ASC	186
7.3.3	Volume sensitivity and MA-PPO robustness	189
7.4	Corridor level Adaptive signal Control Summary.....	191

Chapter 8	Corridor level adaptive TSP based on reinforcement learning, connected vehicles and software in the loop simulation.....	194
8.1	Background.....	194
8.2	Methodology.....	195
8.2.1	Definition of state, action and reward functions.....	195
8.2.2	Test corridor Model	200
8.2.3	RL and simulation architecture.....	202
8.2.4	TSP Implementation architectures & test scenarios	202
8.3	Results	214
8.3.1	Model training.....	214
8.3.2	Bus travel time	217
8.3.3	General traffic delay	219
8.4	Corridor level Adaptive TSP Summary.....	224
Chapter 9	Conclusions, recommendations and contributions	226
9.1	Conclusions	226
9.1.1	Exploration of TSP fundamental principles.....	226
9.1.2	Impact of dwell-time on the performance of transit signal priority.....	228
9.1.3	Single Intersection TSP based on reinforcement learning.....	229
9.1.4	Adaptive transit signal priority based on multi-agent reinforcement learning 230	
9.1.5	Corridor level adaptive reinforcement learning based signal control.....	231
9.1.6	Corridor level adaptive TSP based on reinforcement learning, connected vehicles and software in the loop simulation	232
9.2	Recommendations	234
9.2.1	Defining thresholds for impacts on general traffic	234
9.2.2	Defining traffic state with readily available data.....	235
9.2.3	Explore parallelized training architectures	235
9.2.4	More structured training architectures for increasing complexity.....	236
9.2.5	Adaptive transition logic for RL_ASC_TSP	236
9.3	Contributions	237
9.3.1	Exploration of TSP fundamental principles.....	237

9.3.2	Analysis of trends in dwell-time data and the impact of dwell-time on TSP performance.....	238
9.3.3	Single intersection TSP based on reinforcement learning	239
9.3.4	Adaptive transit signal priority based on multi-agent reinforcement learning	240
9.3.5	Corridor level adaptive reinforcement learning based signal control.....	241
9.3.6	Corridor level adaptive TSP based on reinforcement learning, connected vehicles, and software in the loop simulation	242
	References.....	244

LIST OF TABLES

Table 1: Summary of MOEs in literature	16
Table 2: Summary of single agent reinforcement learning studies.....	29
Table 3: Summary of MARL for traffic control studies	36
Table 4: Signal Timing Parameters.....	50
Table 5: Splits and maximum GE for varying cycle lengths	56
Table 6: TSP strategies and effectiveness under different levels of demand	60
Table 7: TSP Effectiveness strategies for different dwell-time distributions.....	89
Table 8: TSP strategies and effectiveness at far and near-side bus stops	94
Table 9: DDQN-SC selected hyperparameters	116

Table of figures

Figure 1: Basic elements of a decentralized TSP system	13
Figure 2: TSP VISSIM® RBC Algorithm (author’s interpretation)	48
Figure 3: Intersection model in VISSIM®	49
Figure 4: Demand (vph) for different v/c ratios.....	51
Figure 5: Arrival profile of buses vs time in cycle	52
Figure 6: Impact on bus travel time at v/c ratio of 0.95 for GE and EG.....	57
Figure 7: Cross street Delay at v/c ratio of 0.95 for (a) SBT and (b) SBLT.....	58
Figure 8: SBT delay extent with v/c ratio of 0.95 after up to 10 second (a) GE and (b) EG	59
Figure 9: Impact on bus travel time at three levels of v/c ratio, for (a) GE and EG and (b) GE only	61
Figure 10: Bus travel time for 0.85, 0.95, and 1.0 v/c scenarios, under combined GE and GE, GE only, and no TSP	62
Figure 11: Cross Street Delay from GE and EG for (a) SBLT and (b) SBT	63
Figure 12: SBT Delay Extent after up to 10 second truncation for, (a) v/c = 1.0 (b) v/c= 0.95, and (c) v/c = 0.85	64
Figure 13: Bus travel time vs cycle length for (a) Only TSP affected buses and (b) All buses	66
Figure 14: Cross street delay vs cycle for (a) SBLT and (b) SBT	67
Figure 15: SBT Delay extent after truncation of up to 10 seconds for (a) cycle length of 110s, (b) cycle length of 130s and (c) cycle length of 150 seconds.....	68
Figure 16: MARTA Bus Route 39.....	74
Figure 17: Route 39 SB Dwell-time CDF for (a) all stops all day, (b) all stops AM peak, (c) selected stops all day and (d) selected stops AM peak.....	76
Figure 18: Route 39 SB Dwell-time variation for selected stops, with (a) zero dwell-time included, and (b) zero dwell-time excluded.	78
Figure 19: Route 39 dwell-time for selected SB stops (a, b) and NB (c, d) stops by time of day	79

Figure 20: Route 39 Dwell-time CDFs for (a) Near-side bus stops and (b) Far-side bus stops.....	80
Figure 21: Travel time in 200 ft bus far zone around (a) Near-side stops and (b) Far-side bus stops.	81
Figure 22: Fitted dwell-time distributions for stop (a) 905261 and (b) 901979.	82
Figure 23: Network Model in PTV-VISSIM® for (a) far-side bus stop and (b) near-side bus stops.....	84
Figure 24: Dwell-time Distributions from field data	86
Figure 25: Bus arrival profile at check-in.....	88
Figure 26: Bus travel time for different dwell-time distributions for (a) GE & EG and (b) GE only	90
Figure 27: SBT Delay for different dwell-time distributions	92
Figure 28: Bus Travel Time at different offsets for (a) DT02 and (b) DT03.	93
Figure 29: Travel time for TSP impacted buses at far and near-side stops under (a) GE only strategy and (b) GE and EG strategy.....	95
Figure 30:Cross street SBLT Delay, far-side vs Near-side bus stop	96
Figure 31: Elements of RL.....	103
Figure 32: Traffic simulation and RL modules.....	108
Figure 33; Event Based Script Embedded in VISSIM®	110
Figure 34: Single Ring Phase Sequence	114
Figure 35: Action selection and Implementation.....	114
Figure 36: RL agent architecture during the training of DDQN-TSP.	117
Figure 37: DDQN-SC learning curve.	122
Figure 38: Comparing DDQN-SC and A-SC intersection delay for (a) $v/c = 0.6$ and (b) $v/c = 0.95$	123
Figure 39: (a) DDQN-TSP learning curve and (b) Average bus delay during training..	125
Figure 40: (a) Bus travel time with and without TSP and (b) General traffic delay with and without TSP.....	126

Figure 41: Comparison of DDQN-TSP and A-SC TSP based on (a) bus travel time and (b) General traffic delay.....	127
Figure 42: multi agent reinforcement learning framework.....	130
Figure 43. VDN individual architecture. Adapted from Sunehag et al. (2017).....	134
Figure 44: Q-mix architecture. Adapted from (Rashid et al. 2018).....	135
Figure 45: Splits and actions at each intersection.....	138
Figure 46: Case Study network model in VISSIM®	145
Figure 47: Time step selection.....	146
Figure 48: VDN Based MARL learning curve.....	149
Figure 49: Comparing VDN MARL with ASC at $v/c=0.95$	150
Figure 50: Average bus delay at (a) Intersection A and (b) Intersection B during training for independent TSP agents	152
Figure 51: Bus travel time with and without TSP for Independent TSP agents.....	153
Figure 52: Side Street delay with and without TSP.....	154
Figure 53: Average bus delay at (a) Intersection A and (b) Intersection B during training for coordinated TSP agents	155
Figure 54: Bus travel time with and without TSP for coordinated TSP agents.....	156
Figure 55: Side Street delay with and without TSP.....	157
Figure 56: Architecture of Centralized critic for agent i , adapted from (Albrecht et al. 2024)	164
Figure 57: Comparing DDQN and PPO learning curves.....	166
Figure 58: Partially observable environment.....	167
Figure 59: NEMA two ring barrier diagram.....	169
Figure 60: Action Implementation in two ring barrier configuration.....	170
Figure 61: Test network extents in google maps	173
Figure 62: PM peak volumes	174
Figure 63: Network Model in VISSIM®.....	176

Figure 64: Field Vs calibrated model headways for (a) North Ave @ Spring St and (b) North Ave @ I-85 off-ramp	177
Figure 65: VISSIM®-MaxTime® SILs architecture.....	178
Figure 66: Specifying MaxTime® as an external controller in VISSIM® with the required files.....	179
Figure 67: Defining MaxTime® (a) TCP port and (b) signal groups.....	180
Figure 68: MaxTime®-VISSIM® connection settings in MaxTime®.....	180
Figure 69: Interaction of agents with PTV-Vissim® Simulation environment.....	181
Figure 70: Agent-Vissim interaction during simulation run.....	182
Figure 71: Algorithm architecture during training.....	183
Figure 72: Algorithm architecture during testing	184
Figure 73: Learning Curves for different intersection agents.....	186
Figure 74: Main line movement travel time for MaxTime® and MA-PPO for field measured traffic volumes	188
Figure 75: Cross street through movement Delays for MaxTime® and MA-PPO for field measured traffic volumes	189
Figure 76: Main street Delay from MA-PPO vs coordinated ASC at three volume levels	190
Figure 77: Cross street Delay from MA-PPO vs coordinated ASC at three volume levels	191
Figure 78: Local and global bus state definition.....	196
Figure 79: VISSIM® model for the test corridor including bus routes and bus stops ...	201
Figure 80: Dwell-time distribution in VISSIM® for all bus stops	202
Figure 81: MaxTime®-VISSIM® SILs architecture with RL-TSP	205
Figure 82: MaxTime® and RL-agent communication module	207
Figure 83: MaxTime® NTCIP settings	208
Figure 84: MaxTime® MIB Screenshot.....	209
Figure 85: Special background phasing sequence in MaxTime®	211

Figure 86: Implementation of RL agent actions in MaxTime®	212
Figure 87: TSP settings in MaxTime®	214
Figure 88: Learning curves for the first stage of training	215
Figure 89: Bus delay during training	216
Figure 90: Bus travel time for different TSP systems	218
Figure 91: Cross street traffic delay with and without TSP	220
Figure 92: Delay for WB through movements	222
Figure 93: Delay for EB through movements	223

ACRONYMS

A2C	Advantage Actor -Critic
AI	Artificial Intelligence
APC	Automated Passenger Count
API	Application Programming Interface
ASC	Actuated Signal Control
AVL	Automated Vehicle Location
BSM	Basic Safety Message
CI	Check-In
CI-CO	Check-In Check-Out
CNN	Convolutional Neural Network
CO	Check-Out
COM	Component Object Module
CTCE	Centralized Training and Centralized Execution
CTDE	Centralized Training and Decentralized Execution
CTSP	Conventional Transit Signal Priority
CV	Connected Vehicle
DDPG	Deep Deterministic Policy gradient
DDQN	Double Deep Q-Network
DLL	Dynamic Link Library
DNN	Deep Neural Network
DQN	Deep Q-Network
DSRC	Dedicated Short Range Communication
DTDE	Decentralized Training and Decentralized Execution
EG	Early Green
ETA	Estimated Time of Arrival
EVP	Emergency Vehicle Preemption
FSP	Freight Signal Priority
GDOT	Georgia Department of Transportation
GE	Green Extension
GPS	Geographical Positioning System
GUI	Graphical User Interface
IAM	Invalid Action Masking
IDQN	Independent Deep Q-Network
IQL	Independent Q-learning
LOS	Level of Service
LSTM	Long-term Short-term Memory
MADDPG	Multi-Agent Deep Deterministic Policy Gradient

MA-PPO	Multi-Agent Proximal Policy Optimization
MARL	Multi-Agent Reinforcement Learning
MARTA	Metropolitan Atlanta Rapid Transit Authority
MDP	Markov Decision Process
MIB	Management Information Base
MILP	Mixed Integer Linear Program
ML	Machine Learning
MOE	Measures of Effectiveness
MSE	Mean Squared Error
MUTCD	Manual on Uniform Traffic Control Devices
NEMA	National Electronic Manufacturer's Association
NTCIP	National Transportation Communications for ITS Protocol
OBU	Onboard Unit
OD	Origin-Destination
OID	Object Identifier
P-DQN	Parameterized Deep Q-Network
PDU	Protocol Data Unit
PER	Prioritized Experience Replay
PF	Priority Factor
PG	Policy Gradient
PoMDP	Partially Observable Markov Decision Process
PPO	Proximal Policy Optimization
RBC	Ring Barrier Controller
RL	Reinforcement Learning
RNN	Recurrent Neural Network
RSU	Roadside Unit
SC	Signal Control
SILs	Software in the Loop Simulation
SNMP	Simple Network Management Protocol
SRM	Signal Request Message
TCP	Transmission Control Protocol
TCQSM	Transit capacity and quality of service manual
TMC	Turning Movement Count
TRPO	Trust Region Policy Optimization
TSP	Transit Signal Priority
UDP	User Datagram Protocol
v/c	Volume to Capacity
VDN	Value Decomposition Network

CHAPTER 1 INTRODUCTION

1.1 Background

According to National Transit Summaries and Trends (NTST) published by Federal Transit Administration (FTA), urban transit systems provided 6.9 billion passenger trips in the year 2023 with 3.26 billion of these being bus trips (Federal Transit Administration 2023). Transit provides a means of travel to choice riders and to segments of the population who are unable to drive due to physical (age, disability) and financial disadvantages. According to the transit capacity and quality of service manual (TCQSM), passengers evaluate the transit quality of service based on travel time, availability (coverage, frequency, etc.), service delivery (reliability, comfort, etc.) safety, and security. Transit agencies evaluate the quality of service based on economics (ridership and cost efficiency), and transit impacts (economic, mobility, and environmental). While motorists use metrics of congestion impacts such as volume to capacity ratios and average system speed and delay (Transportation Research Board and National Academies of Sciences Engineering and Medicine 2013).

To improve bus transit quality of service, transit agencies must evaluate priorities to allocate their limited resources to invest in (a) new road and transit infrastructure including construction of dedicated bus lanes, bus stations, and queue jumps, (b) increasing and modernizing fleet, and (c) new technologies including dispatch and monitoring, transit signal priority (TSP), and passenger information systems. Investments in new technologies like TSP may result in significant service improvements for less capital investment compared to large transit infrastructure projects. TSP is the subject of this dissertation.

Traffic signals are traffic control devices that assign the right-of-way in time to different vehicular and pedestrian movements at an intersection. Properly designed traffic signals allow orderly movement of traffic, increase the traffic handling capacity of intersections, and improve safety by reducing the number and severity of conflicts (United States Department of Transportation and Federal Highway Administration 2009). On arterials, urban streets, and other facilities such as interchanges that have paired intersections, signal coordination allows traffic flow progression of one or more directional movements through adjacent intersections. Traffic signals most commonly work by assigning conflicting movements right-of-way in a defined cyclic sequence. The timing of each movement is estimated from the set objectives, including minimizing delay, queue length, maximizing throughput, maintaining coordination, and maintaining a safe operation.

TSP alters the normal timing operation by extending the priority phase, shortening nonpriority phases, rotating phase sequence, or inserting a special phase to allow a transit vehicle to pass through the intersection (Koonce 2008). TSP is different from signal preemption, which the Manual on Uniform Control Devices (MUTCD) defines as the “the transfer of normal operation of traffic control signals to a special control mode of operation” (United States Department of Transportation and Federal Highway Administration 2009). Preemption is necessitated to give right of way to emergency vehicles such as ambulances and fire engines and to clear the intersection space ahead of train or boat crossing. Signal preemption has a higher priority than signal priority with the former requiring skipping of all conflicting phases and imposing no limits on the green extension for the preempted movement until the preempted vehicle has exited the intersection (Koonce 2008). Compared to preemption, signal priority has a more

constrained solution space and is a more complex problem to solve due to the need to limit impacts on the nonpriority traffic movements.

TSP has the potential to improve transit reliability, schedule adherence, and ultimately transit ridership. Several previous simulation studies and a few pilot tests have reported reduced transit vehicle delays and travel time, and reduced travel time variability when TSP is implemented (Al-Sahili and Taylor 1996; Anderson and Daganzo 2019; Balke et al. 2000; Dion et al. 2004; Hu et al. 2014; Lee and Wang 2022; Muthuswamy et al. 2007; Ngan 2004; Rakha and Zhang 2004; Sheffield et al. 2021). A survey carried out by National Academies of Sciences Engineering and Medicine (2020) showed that several transit agencies across US are implementing some level of TSP. As mentioned earlier, the increasing popularity of TSP stems from its potential improvements in the transit system with modest investments in technology and equipment compared to large transit infrastructure projects.

In conventional TSP systems with check-in check-out (CI-CO) detectors, TSP system parameters including estimated time of arrival (ETA) and extension and truncation limits are estimated offline and pre-set in the controller. In adaptive TSP systems, whether selecting the variable parameters such as ETA, or entirely taking control of phase order and lengths, strategy selection occurs online as the bus approaches the intersection. Adaptive TSP systems are more effective due to responsiveness to ETA variations because of variations in traffic demand and dwell-time. Additionally, adaptive TSP systems better balance tradeoffs between bus performance and side street level of service (LOS) by considering real time traffic conditions in the selection of TSP strategies.

1.2 Problem definition

Early TSP studies, mainly using simulation, focused on evaluating the performance of TSP under different conditions and the impact of various parameters (Al-Sahili and Taylor 1996; Anderson and Daganzo 2019; Balke et al. 2000; Dion et al. 2004; Hu et al. 2014; Lee and Wang 2022; Muthuswamy et al. 2007; Ngan 2004; Rakha and Zhang 2004; Sheffield et al. 2021). The reported benefits and disbenefits of TSP vary widely across different studies, in part due to (1) the different measures of effectiveness (MOEs) adopted, and (2) the wide range of conditions and parameters of the transit vehicles and transit routes. There is limited documented guidance on the impacts of the different TSP strategies to both transit vehicles and the general traffic (National Academies of Sciences Engineering and Medicine 2020). In coordinated arterial systems, signal timing parameters including cycle lengths, splits, and offsets are typically selected to provide optimal service to the general traffic without consideration of transit vehicle operations. TSP is almost always considered as an afterthought. Considering TSP as an afterthought potentially limits the flexibility of signal control to accommodate TSP. There is need for further research into signal timing approaches that incorporate TSP in the optimization objectives.

One of the key variables in TSP system design is bus stop dwell-time. A few studies have shown that dwell-time magnitudes and variability affect TSP performance at near-side bus stops (Dion et al. 2014; Ngan 2004; Rakha and Zhang 2004). Some limited research efforts to evaluated the impacts of dwell-time magnitude and variability on TSP performance(Dion et al. 2014). Additionally, TSP studies that incorporate dwell-time in modeling TSP have mainly used deterministic dwell-time (Bayrak and Guler 2020; Dion et al. 2014; Rakha and Zhang 2004) or assumed dwell-time distributions that may not be

reflective of actual field dwell-time distributions (Ekeila et al. 2009; Hu et al. 2014; Ma and Yang 2007).

More recent TSP studies have focused on developing optimization strategies to improve bus performance while limiting impacts to the general traffic. Using traffic microscopic simulation environments, several studies have tested adaptive TSP algorithms that integrate connected vehicle (CV) data (Beak et al. 2018; Cvijovic et al. 2022; Hu et al. 2015; Hu et al. 2014; Hu et al. 2016; Lee et al. 2017; Mohammadi et al. 2020; Teng et al. 2019; Wang et al. 2020; Wu and Guler 2019; Yang et al. 2019; Zeng et al. 2015). Adaptive TSP algorithms have mainly been based on mathematical programming (Hu et al. 2015; Kim and Rilett 2005; Li et al. 2011; Ma et al. 2013; Truong et al. 2019; Zeng et al. 2015; Zhai et al. 2023). However, these studies' complex and nonlinear objective functions require high computational resources to solve in real time. Model free reinforcement learning (RL) approaches may be a suitable alternative because well-trained, model-free RL algorithms have no need to project future traffic states, making them computationally efficient to implement in real time. Recent efforts are testing machine learning (ML) approaches, especially RL (Cheng et al. 2022; Hu et al. 2023; Long et al. 2022; Shen et al. 2023; Yang and Fan 2024; Zhong et al. 2023). These ongoing efforts build on recent and still evolving RL-based adaptive traffic control algorithms (Aslani et al. 2019; Bálint et al. 2022; Bouktif et al. 2023; Casas 2017; Lee et al. 2022; Li et al. 2020; Li et al. 2016; Li et al. 2021; Liu et al. 2022; Shabestary et al. 2020).

RL-based traffic control studies have mainly considered single agent formulations with only a few recent studies considering multiple agents (Chang et al. 2024; Chen et al. 2021; Fu et al. 2023; Liu and Li 2023; Wu et al. 2020). Signal coordination is at the heart of

arterial traffic control; rarely do arterial intersections operate in isolation. Moving from single agent to multi-agent reinforcement learning (MARL) introduces new challenges that require modification or formulation of new learning frameworks. These challenges include equilibrium selection, non-stationarity of the environment, scaling to many agents, multi-agent credit assignment, and partial observability (Albrecht et al. 2024; Zhang et al. 2021).

Additionally, previous RL-based traffic control studies almost universally assume very simplified signal plans, including limiting the number of allowable phases (Aslani et al. 2019; Bálint et al. 2022; Bokade et al. 2023; Bouktif et al. 2021; Bouktif et al. 2023; Li et al. 2020; Li et al. 2016; Li et al. 2021; Liu and Li 2023; Liu et al. 2023; Shabestary et al. 2020), assuming fixed timing sequences (Aslani et al. 2019; Bálint et al. 2022; Bokade et al. 2023; Bouktif et al. 2021; Bouktif et al. 2023; Casas 2017; Chang et al. 2024; Fu et al. 2023; Lee et al. 2022; Li et al. 2020; Li et al. 2016; Li et al. 2021; Liu and Li 2023; Liu et al. 2023; Shabestary et al. 2020; Wu and Guler 2019), fixing the duration or intervals of green for each selected phase or phase combinations (Bálint et al. 2022; Chang et al. 2024; Li et al. 2020; Liang et al. 2019), and utilizing single ring barrier configurations (Bálint et al. 2022; Chang et al. 2024; Fan and Yang 2024; Li et al. 2020; Liang et al. 2019; Wang et al. 2019). For most part, RL approaches remain untested for more complex field signal timing plans.

The few previous studies that integrate TSP into RL-based adaptive traffic control do so by modifying state and reward functions to include transit flow parameters. At the intersection level, one RL agent is trained to take second by second level decisions to adaptively implement general traffic control and TSP. However, as stated by Shabestary et al. (2020), the industry may prefer “controllers that manipulate signals less frequently”. It

may be desirable to only manipulate the signals at second-by-second level during priority service and maintain actuated signal control (ASC) in absence of the bus. It may also be challenging to train one agent to optimally control both general traffic and implement TSP as bus arrivals at intersections are sparse occurrences compared to general vehicles. Training the same RL agent to learn general traffic control and bus prioritization may take prohibitively longer periods, especially in higher fidelity traffic microscopic simulation environments. TSP algorithms in field controllers are event based, triggered only when bus priority is requested. There is need to explore event-based RL algorithms only triggered when priority is requested or when a bus approaches the intersection.

TSP algorithms currently in use are almost universally implemented independently, intersection-by-intersection. Even in centralized TSP systems, TSP parameters and strategies are selected independently for each intersection. There is peer-to-peer functionality in some modern controllers that could allow some capabilities to select and implement coordinated TSP strategies across multiple intersections (Qfree 2020). There is almost no literature on the performance of these peer-to-peer functionalities for TSP implementation. A few previous studies have formulated mathematical programs to select coordinated TSP strategies across multiple intersections (Hu et al. 2015; Ma et al. 2013; Truong et al. 2019; Zeng et al. 2015; Zhai et al. 2023). These studies show the potential benefits of selecting coordinated TSP strategies across multiple intersections, especially closely spaced intersections. As indicated, there are limited MARL for adaptive traffic signal control. Fewer still have made attempts to integrate TSP (Li et al. 2023; Long and Chung 2023b; Yang and Fan 2024).

1.3 Research objectives

The main objective of this dissertation is to develop adaptive TSP algorithms based on deep reinforcement learning, employing data from connected vehicles, and software in the loop simulation (SILs). The algorithms are developed for both single and multiple intersections on arterial streets. Before moving to these advanced algorithms, a preliminary study is carried out to explore TSP fundamental principles in a simulated environment. The objective of this preliminary study is to identify critical factors and conditions that affect TSP performance and to establish baselines and traditional measures of effectiveness (MOEs) that are later used to evaluate the performance of RL-based algorithms. The following are the specific research tasks undertaken:

1. For a single intersection with the conventional CI-CO TSP system, evaluate the (a) performance of different TSP strategies, (b) impact of traffic demand, and (c) impact of background signal timing.
2. Analyze field dwell-time data and then assess the impact of dwell-time magnitude and variability on TSP performance for both far-side and near-side bus stops.
3. Formulate, train, and test a single RL agent to implement TSP at an isolated intersection.
4. Formulate, train, and test multiple RL agents to implement coordinated TSP strategies at multiple coordinated intersections.
5. Formulate a multi-agent adaptive signal control algorithm and test the same on a real-world corridor with actual field conditions and constraints.

6. Integrate TSP into the corridor wide traffic control algorithm formulated under task 5., including testing the algorithm in SILs with the background signal timing running on field implemented actuated signal control (ASC).

1.4 Research scope

This dissertation develops adaptive TSP algorithms based on RL, CV, and SILs. Before embarking on these rather advanced algorithms, the dissertation performs a sensitivity study in simulated environment to explore the fundamental principles of TSP including the selection of TSP strategies and determining the critical conditions that affect TSP performing.

The RL study starts with a simpler case of a hypothetical single intersection implementation with single RL agent then progresses to a simple hypothetical network of a pair intersections and finally formulates and tests the algorithms for real-world corridor implementation. Realizing that RL-based adaptive traffic signal control is not yet fully developed, especially MARL, the first step is to develop multi-agent RL-based adaptive signal control, which may then be extended to include TSP.

This dissertation is heavily focused on developing operational-level signal control logics for TSP. Several planning and policy level aspects of TSP are highlighted and provided for in the algorithms are considered in extensive detail. For instance, (a) the policy question of defining the “acceptable” compromises on the level of service (LOS) for general traffic and (b) factors for conditional implementation of TSP are not addressed in detail. However, the algorithms have been developed with adjustable parameters that can be tweaked to achieve the required level of priority and conditional implementation.

Results of the sensitivity study for the CI-CO TSP system indicate that TSP performance is most favorable in lower v/c conditions, green extension (GE) is more effective than early green (EG) and use of a cycle length slightly longer than the traffic demand based optimal cycle could lead to lower impacts to conflicting non-transit vehicles during TSP events. Analysis of field dwell-time data showed that dwell-time varies from stop to stop and for each stop by time of day (TOD). For far-side bus stops, dwell-time significantly impacted the bus arrival profile at the check-in detector and thus the TSP strategy selected while for near-side bus stops, dwell-time variability introduced more uncertainty in estimated time of arrival (ETA) and significantly reduced TSP effectiveness especially for GE. On a real-world based simulated seven-intersection test corridor, the formulated RL-based adaptive signal control algorithm showed superior performance to the field optimal actuated signal control (ASC) as evaluated by bus and mainline travel times and delay for side street movements. The formulated RL-based TSP systems show superior performance to the CI-CO TSP system as evaluated by bus travel time. RL-based TSP systems show marginal changes on the mainline travel time and side street delay for non-transit vehicles after TSP implementation.

1.5 Dissertation organization

The rest of this document has an additional eight chapters. Chapter 2 presents a critical review of the most recent literature examining the state of art and practice of TSP including the recent advances in AI based traffic and TSP control. Chapter 3 presents the methods and results of an exploratory study for TSP fundamental principles in a simulated environment. Chapter 4 presents an analysis of field dwell-time data and experiments to assess the impact of dwell-time on TSP performance. Chapter 5 discusses the formulation,

training, and testing of a single DQN RL agent to implement TSP at an isolated hypothetical intersection. Chapter 6 extends the single agent formulation of chapter 5 to MARL based on value decomposition network (VDN) and tests the performance of the formulated VDN TSP algorithms on a simple hypothetical network consisting of a pair of coordinated intersections. Based on lessons from Chapter 5 and Chapter 6, Chapter 7 formulates MARL adaptive signal control algorithm and tests the algorithm on a simulated real-world corridor with actual/complete traffic movements, traffic volumes, and geometry. Chapter 8 integrates TSP into the MARL-based signal control algorithms developed in Chapter 7. The MARL algorithms in Chapter 7 and Chapter 8 are based on MA-PPO. Lastly Chapter 9 presents the dissertation conclusions, recommendations, and contributions.

CHAPTER 2 LITERATURE REVIEW

2.1 Overview

This chapter presents a critical review of the most recent and relevant literature to examine the state of art and practice of TSP. The areas include TSP system architectures, TSP performance evaluation and the critical performance factors, TSP optimization algorithms, and integration of CV data. The review also covers recent applications of RL in traffic control and the ongoing efforts to integrate TSP in RL-based traffic control.

2.2 TSP systems

TSP aims to provide transit vehicles a free flow path through the intersection, or at least reduce the waiting time on the red phase. In terms of activity, TSP systems fall under two broad categories of passive and active TSP. Passive TSP involves selecting signal plans that consider and /or favor the flow of transit vehicles without active detection or tracking of transit vehicles in the system. Passive TSP is known to work best for high frequency bus routes with good knowledge of bus speed, dwell-times, schedule adherence, and other transit operational characteristics (Smith et al. 2005). Active TSP detects or tracks transit vehicles along the route and provides preferential treatment at signalized intersections by extending the normal phase green, providing early green (EG), rotating the normal phase sequence, or inserting a special bus phase. In this document unless otherwise stated, the term TSP refers to active TSP.

Figure 1 shows some elements of a conventional TSP system. Check-in detectors provide the means identifying an approaching bus and sending a priority call to the controller.

Check-out detector placed immediately downstream the stop line allow termination of priority service after the bus passes the stop line. The combination of check-in and check-out detector forms what shall be referred to as check-in check-out (CI-CO) TSP system in the rest of this dissertation. Near-side bus stops occur on the upstream side of the subject intersection but after the check-in detector. Far-side bus stops occur after the subject intersection and after the check-out detector. TSP systems in use mainly employ two strategies of green extension (GE) and EG also called red truncation. GE extends the current priority phase green to enable the transit vehicle to pass while EG strategy involves shortening the preceding phases to reach the priority phase green as early as possible.

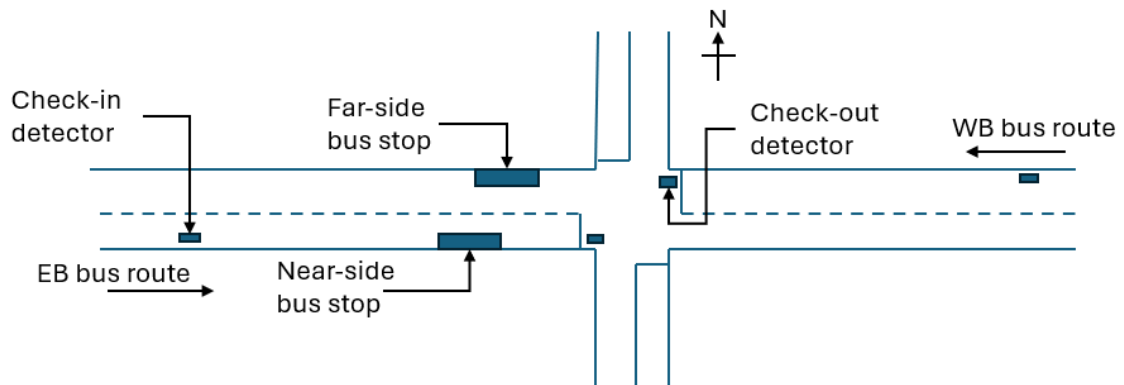


Figure 1: Basic elements of a decentralized TSP system

Active TSP can be unconditional where all buses approaching the intersection can request and be granted priority; or conditional where priority is only provided to buses meeting preset conditions. These conditions may include schedule and headway deviations above set thresholds, occupancy/load above a set threshold, conflicts with other buses, and previous granted requests (Ding et al. 2015; Hu et al. 2015; Hu et al. 2014; Lee and Wang 2022; Liao and Davis 2007; National Academies of Sciences Engineering and Medicine

2020; Zeng et al. 2015). In terms of architecture, TSP systems can be distributed or centralized. In distributed systems, requests are processed at the intersection level with only communication between the bus and the local controller. In centralized systems, TSP requests are processed at the transit or traffic management center (TMC), leveraging communications between the bus, transit management center or TMC, and the intersection controller. NTCIP 1211 provides a good reference for these architectures (NTCIP Signal Control Priority Working Group. et al. 2014). Increasingly TSP systems incorporate Automated Vehicle Location (AVL) to transmit bus information including location, speed, heading, schedule deviations, and other characteristics, better enabling the evaluation and processing of priority requests.

A survey carried out by National Academies of Sciences Engineering and Medicine (2020) sheds light on TSP systems implemented by various Transit agencies in US and Canada. Of 31 responding agencies, 13 operated some level of conditional TSP while 18 agencies implemented unconditional TSP. Twenty-four of the 31 agencies use distributed TSP architecture while seven agencies implemented centralized systems. Seventeen out of 30 responding agencies had some level of AVL implemented in their TSP systems. Conventional TSP (CTSP) systems have fixed location detectors or detection zones and actuation parameters preset in the controller, including estimated time of arrival (ETA), and GE and EG parameters. Signal timing optimization for TSP is performed offline and parameters are preset in the controller. More adaptive TSP systems have been researched and are increasingly being pilot tested. These adaptive TSP systems are equipped with AVL and the optimization of the signal timing including evaluation of the conditions and resolution of the conflicting requests is performed online in real or near real-time.

Integration of CV technologies in TSP systems is increasingly being studied and pilot tested. Studies integrating CV data into TSP systems are discussed in more detail later in the chapter.

2.3 TSP Performance Measures

Early TSP studies focused on evaluating the performance of TSP under different conditions and the impact of various parameters (Al-Sahili and Taylor 1996; Dion et al. 2004; Koonce et al. 2002; Muthuswamy et al. 2007; Ngan 2004; Rakha and Zhang 2004). These studies consisted of primarily sensitivity experiments using different simulation tools with hypothetical networks or actual networks as base models. Parameters found to affect TSP performance included bus headways, bus stop location, detector location, green extension limit, bus arrival within the cycle, bus stop dwell-time, and TSP strategy selected. Conditions affecting TSP performance included demand volumes for both main street and minor streets. These studies demonstrated that TSP has the potential to improve bus travel time or reduce bus delays, with marginal impacts on cross street traffic at low demand. Disbenefits to cross street traffic were shown to increase with increasing cross street volume.

Although the potential for positive TSP benefits is quite well established, the magnitude of the reported benefits and disbenefits varies widely across studies. In part, the variation derives from the selection of different measures of effectiveness (MOEs). TSP evaluation normally considers impacts on both transit and other vehicles in the network.

Table 1 presents a summary of some of the MOEs found in the reviewed literature. For buses most studies used travel time or delay, averaged by the number of buses. In addition

to delay, Ngan (2004) used green extension success rate which is the proportion of green extensions for which the intended buses successfully passed through the intersection. For general traffic, most studies use delay but there are measurement or definitional differences across studies. For example, total or average delay over an entire period or run versus a focus on localized effects by evaluating the delay over a couple of cycles immediately following TSP (Hu et al. 2014; Ngan 2004). To evaluate the tradeoffs between transit and other vehicles, more recent studies use person delay which is based on the occupancies of both transit and passenger cars (Hu et al. 2015; Lee and Wang 2022; Mishra et al. 2020; Zeng et al. 2015).

Table 1: Summary of MOEs in literature

Study and Methods	Transit MOEs and Results	General traffic MOEs and Results	Comments on Results
Rakha and Zhang (2004) Simulation with INTEGRATION software – fixed location detector	<ul style="list-style-type: none"> • Average bus delay • Average bus stops • Average bus fuel usage 	<ul style="list-style-type: none"> • Average car delay • Average car stops. • Average car fuel usage • Systemwide delay, stops and fuel consumption 	<ul style="list-style-type: none"> • up to 28.7% bus delay decrease • marginal impacts to general traffic in most of the studied scenarios
Ngan (2004) Simulation with Vissim – fixed location detector	<ul style="list-style-type: none"> • Average bus delay & travel time • GE success rate 	<ul style="list-style-type: none"> • Average cross street cycle delay • Number of recovery cycles required 	<ul style="list-style-type: none"> • Maximum benefits to buses at v/c = 0.8 • GE less effective with increased v/c • Higher bus delays for near-side bus stops

Table 1: Summary of MOEs in literature (continued)

Study and Methods	Transit MOEs and Results	General traffic MOEs and Results	Comments on Results
Sheffield et al. (2021) Analys of field operational TSP system	<ul style="list-style-type: none"> • On time performance • Mean schedule deviation. • Travel time • TSP requests vs TSP granted 	<ul style="list-style-type: none"> • Change in split failure. • Change in green time 	<ul style="list-style-type: none"> • Up to 2.5% improvement on time performance • Marginal impacts on general traffic even with unconditional TSP
Hu et al. (2014) Compares CTSP with adaptive TSP algorithm integrating CV	<ul style="list-style-type: none"> • Person delay in 3 cycles after TSP • Average bus delay 	<ul style="list-style-type: none"> • Person delay in 3 cycles 	<ul style="list-style-type: none"> • CV-TSP reduced bus delay up to 88% • Delay per person in system was reduced by up to 14%
Ekeila et al. (2009) Using VISSIM® simulation, compares CTSP and a modified TSP algorithm that uses AVL	<ul style="list-style-type: none"> • Average Bus travel time • delay across 	<ul style="list-style-type: none"> • Average Vehicle travel time and delay across simulation runs 	<ul style="list-style-type: none"> • Up to 33% reduction in bus travel time • No significant impacts on other vehicles
Muthuswamy et al. (2007) WATsim simultion	<ul style="list-style-type: none"> • Average bus TT 	<ul style="list-style-type: none"> • Average vehicle travel time 	<ul style="list-style-type: none"> • Up to 25% decrease in bus travel time
Wang et al. (2020) Field evaluation – compared TSP performance before and after signal retiming	<ul style="list-style-type: none"> • TSP requested vs TSP served. • bus reliability, • bus travel time • bus running time 	<ul style="list-style-type: none"> • NA 	<ul style="list-style-type: none"> • Up to 92% bus reliability
Zhou et al. (2006) VISSIM® simulation on a hypothetical intersection and a near-side bus stop	<ul style="list-style-type: none"> • Bus average delay 	<ul style="list-style-type: none"> • Total intersection delay 	<ul style="list-style-type: none"> • Bus delay reduced by up to 29% • Passenger car delay increased by up to 55%

Table 1: Summary of MOEs in literature (continued)

Study and Methods	Transit MOEs and Results	General traffic MOEs and Results	Comments on Results
Liao and Davis (2007) Ainsun simulation	<ul style="list-style-type: none"> • Bus travel time • Bus delay • Bus stop time 	<ul style="list-style-type: none"> • Speed, average TT, average delay, average stops 	<ul style="list-style-type: none"> • 4-15% reduction in bus travel time and 5-20% reduction in bus delay
Lee et al. (2017) Field testing TSP with CV algorithm on a Virginia Tech test bed	<ul style="list-style-type: none"> • Bus delay • Successful GE 	<ul style="list-style-type: none"> • NA 	<ul style="list-style-type: none"> • Reduced delay for the bus between 32- 75%

2.4 Arrival time prediction

One of the key challenges in TSP implementation is the determination of ETA of the bus approaching the intersection. In both fixed location check-in detector and AVL systems, an estimate of ETA is needed to evaluate the need for TSP service and the appropriate TSP strategy to employ. Uncertainty in bus travel time mainly comes from the level of congestion on the link and dwell-time at bus stops. Unfortunately, most signal controllers are programmed with a single ETA value, making it very difficult to be responsive to high travel time variability.

There are several studies for bus travel time prediction along entire routes; however these are intended for schedule improvements and passenger information, but these are not sufficiently granular for TSP applications (Farid et al. 2016; Jian et al. 2013; Kumar et al. 2019; Taparia and Brady 2021; Xu and Ying 2017). Several studies including Ngan (2004), Wu and Guler (2019) and Rakha and Zhang (2004) show reduced effectiveness of TSP at high levels of congestion which is partly attributable to increased uncertainty in travel time

and the insufficiency of a single, static ETA. There is a general adoption of historical average travel times in setting ETAs in field implementation of TSP, although limited published information or guidance exists on setting ETA. As discussed below, most of the research on ETA prediction and setting is undertaken for AVL and centralized TSP systems (National Academies of Sciences Engineering and Medicine 2020), despite the larger share of TSP systems remaining fixed location detector implementations.

Several studies that focus on developing more dynamic and adaptive TSP algorithms including variable ETA and ETA prediction modules (Balke et al. 2000; Ekeila et al. 2009; Kim and Rilett 2005; Li et al. 2011). In these studies, ETA is mainly predicted by using regression models (Ekeila et al. 2009; Hu et al. 2014; Kim and Rilett 2005; Li et al. 2011; Liu et al. 2007) Kalman filter models (Ekeila et al. 2009; Shalaby and Farhan 2004) and more recently machine learning methods (Farid et al. 2016; Jian et al. 2013). Other studies like Tan et al. (2008) have adopted probabilistic and Bayesian approaches. The models are built with historical and at times real time data with the main predicting variables including distance remaining, traffic volumes, time of day, and dwell-time for near-side bus stops. Other researchers have used analytical approaches including shockwave theory to estimate queuing ahead of the bus and travel time (Bhaskar et al. 2007; Liang et al. 2018; Liu et al. 2009; Tu et al. 2012). The analytical approaches mainly use detector and signal data as inputs. A model developed by Tu et al. (2012) uses image processing sensor to detect and process in real time the number of vehicles ahead of the bus and, together with signal status data, estimate ETA.

Koonce et al. (2002) and Zhou et al. (2006) focus not on ETA but on determining the optimal detector locations for TSP performance. Zhou et al. (2006) develops an analytical

methodology for determining optimal detector placements for queue jumpers while Koonce et al. (2002) establishes a rule based method evaluated on a field TSP system. In a sensitivity study by Wu and Guler (2019) it was found that different detector locations may be optimal depending on the level of congestion. AVL provides the capability to trigger TSP calls at any point as the bus approaches the intersection (thus, a call may be placed when the static ETA is correct). A few studies including Liu et al. (2007) have explored the potential of AVL to find optimal trigger points for TSP which may mean different request points for each individual bus and with different strategies for GE and EG. Liu et al. (2007) demonstrates the need to request EG early enough for the signal controller to have enough time to adjust the signal timings and to clear the queue ahead of the vehicle. However, the challenge remains of the upstream bus stop location when calls may need to be triggered after the bus stop to avoid the bigger issue of dwell-time uncertainty in ETA.

2.5 Impact of dwell-time on TSP performance

2.5.1 Dwell-time data

According to Transit capacity and quality of service manual (TCQSM), dwell-time is associated with boarding and alighting passenger volumes, fare payment method, vehicle type and size, in-vehicle circulation, and bus stop spacing (Transportation Research Board and National Academies of Sciences 2013). The manual provides an empirically developed model for estimation of dwell-time and its variability in the absence of field data. The manual also provides guidance on collection of field dwell-time data. Several studies have conducted manual surveys of field dwell-time onboard buses using hand held timers, GPS equipment, and video recorders (Ding et al. 2015; Fricker and Jon 2011; Grisé and El-

Geneidy 2017; Kim and Rilett 2005; Li et al. 2012). With more transit agencies equipping their fleets with automated passenger count (APC) and AVL equipment, an increasing number of studies are deducing dwell-time from APC and AVL data (Chen et al. 2004; Isukapati et al. 2017; Milkovits 2008; Moosavi et al. 2017; Rajbhandari et al. 2003; Ranjitkar et al. 2019).

APC systems automatically collect data including the number of passengers boarding and alighting, and time to open and close bus doors. Passenger loads can easily be deduced from APC data (Fan et al. 2023). One of the challenges of APC systems is the reliability of the sensors and the accuracy of the data collected itself. To assess the quality of APC data collected in Montreal, Quebec, Gris  and El-Geneidy (2017) compared APC data with manually collected dwell-time data for the same routes. Dwell-time estimated from APC was in most cases larger than the manually collected dwell-time which was attributed to excess time that elapsed before closing the doors after serving passengers. With the availability of large dwell-time datasets made possible by APC systems, several studies have attempted to build dwell-time prediction models using regression (Glick and Figliozzi 2019; Ranjitkar et al. 2019), probabilistic (AlHadidi and Rakha 2019; Dai et al. 2019; Isukapati et al. 2020; Rashidi and Ranjitkar 2015) and machine learning models (Ding et al. 2014; Ranjitkar et al. 2019; Rashidi et al. 2014; Xin and Chen 2016). Decision variables in these models include number passengers boarding and alighting, headway, schedule adherence, stop spacing, bus load, bus type, time of day, weather, bus crowdedness, etc.

Isukapati et al. (2017) analyzed trends in AVL dwell data collected over two years (2012 – 2014) from Allegheny County in Pennsylvania for two bus routes. The data showed dwell-time variance from stop to stop, by time of day, and with passenger activity. Stops

and periods with a larger proportion of non-zero dwell-times also showed a higher variability of dwell-times both for the same time intervals and for different times of the day. Li et al. (2012) analyzed dwell-time data from BRT stations in Changzhou, China. Dwell-time data between 3 and 180 seconds was found to fit a lognormal distribution. Additionally, dwell-time was highly correlated with the number of passengers boarding, number of passengers alighting, and passenger load.

2.5.2 Integrating dwell-time in TSP design

In coordinated arterial systems when buses stop midblock to serve passengers, they typically fall out of the progression bandwidth. Depending on the magnitude of dwell-time the buses may not be able to traverse the intersection during the current green. In a fully coordinated and under saturated system, the magnitude and variability of dwell-time may largely determine the bus arrival profiles at the intersection stop line. And as shown by Roy (2023) and Rakha and Zhang (2004) the performance of preemption and priority systems are affected by the priority vehicle arrival profiles within the signal cycle. Passive transit signal priority (TSP) systems are designed to favor the progression of transit buses through signalized intersections without actively detecting bus movements and altering signal timing. The successful design of passive TSP requires good knowledge of dwell-times at bus stops and bus speeds within the block. Active TSP requires accurate prediction of ETA between the priority request point and the intersection stop line. For near-side bus stops where the priority call may need to be placed before the bus passes the bus stop, dwell-time needs to be included in ETA prediction and depending on the variability of dwell-time the ETA uncertainty may be greatly increased.

It is established that TSP performance is better for systems with far side bus stops compared to near-side bus stops (Ngan 2004; Rakha and Zhang 2004). The better performance of TSP at far-side bus stops is attributed to increased uncertainty in ETA estimates resulting from the dwell-time variability at near-side bus stops. There have only been limited efforts to evaluate the impact of dwell-time magnitude and variability on TSP performance. Using VISSIM® simulation, Dion et al. (2014) performed experiments for cases of deterministic and stochastic dwell-time. In the stochastic case, a dwell-time distribution is generated by considering varying passenger arrivals at the bus stop. The results of the study show that dwell-time significantly affected TSP performance but counterintuitively, the variable dwell-time case showed better TSP performance. The better performance in the case of variable dwell-time was attributed to changes in arrival patterns at the stop line due to variable dwell-time. The key limitation of this study is that the dwell-time distributions adopted may not be representative of the field dwell-times.

In developing an adaptive TSP algorithm for near-side bus stops, Kim and Rilett (2005) developed a weighted least squares model that predicts the dwell-time confidence interval in real time based on bus headways. The TSP algorithm then selects a TSP strategy from GE, EG, and phase insertion targeting to span the dwell-time interval with the resultant priority window. Compared to the conventional TSP algorithm that used average dwell-time in ETA estimation, the improved algorithm gave better performance in terms of improved bus travel time and lower impacts on the general traffic.

In considering the impact of dwell-time in TSP implementation, several simulation studies consider dwell-time distributions that may not be reflective of field dwell-time distributions. Hu et al. (2014) considers a normal distribution with mean of 30 sec and 2

sec standard deviation, Rakha and Zhang (2004) models dwell-time deterministically with fixed values, Ma and Yang (2007) considers normal distribution with mean 30 sec and 10 sec standard deviation while Ekeila et al. (2009) considers normal distribution with mean 15 sec and 3 sec standard deviation. Other studies including Bayrak and Guler (2020) and Dion et al. (2014) adopt the linear dwell-time model presented in TCQSM in which dwell-time is modeled as a function of number of passengers boarding and alighting (Transportation Research Board and National Academies of Sciences 2013)

2.6 Impact of background signal timing on TSP performance

Some studies have revealed better performance of TSP under optimal background signal timing parameters, especially cycle length and phasing sequences (Muthuswamy et al. 2007; Rakha and Zhang 2004). Rakha and Zhang (2004) studied the impact of cycle length, splits, and phasing sequences on the performance of TSP and found that the vehicle optimal signal timings provide the best performance for TSP as less adjustment is needed to accommodate TSP. In one of the very few studies that report TSP field performance, Wang et al. (2020) analyzed TSP performance data for a 35 intersection CV corridor running TSP in Salt lake, Utah. The study compared TSP performance in terms of TSP requested, TSP served, bus reliability, travel time, and running time before and after signal retiming aimed at improving TSP performance. The study reports improvements on all measures after retiming.

Starting from a max-pressure based signal optimization model developed by Varaiya (2013), Xu et al. (2022) adds constraints to integrate transit vehicle operations in the base signal timings. Testing the model on a simulated Austin downtown network, the approach

showed better performance in terms of bus travel time compared to implementing TSP on base signal timings optimized solely for other vehicles and there were only minimal impacts to general traffic especially at lower traffic demands. Several studies have proposed modifications to the MAXBAND signal progression model formulated by Little et al. (1981) to incorporate transit bus parameters including speed and dwell-time in the optimization of the bandwidth (Dai et al. 2015; Han et al. 2022; Jeong and Kim 2014). Hu et al. (2015) formulates a TSP optimization model with CV data that allows communication between buses and adjacent signals to allow coordinated TSP and bus progression through adjacent signals.

Li and Jin (2017) developed an arterial coordination plan for five intersections with the objective of reducing total passenger delay on both transit vehicles and general-purpose vehicles. For transit buses, the algorithm considers delay at intersections and at bus stops thus incorporating bus dwell-time in developing the signal timing plans. The developed timings showed reduced passenger delay compared to the base timing plans. However, the algorithms are only tested with high bus frequencies (as high as 850 buses per hour) and it is not clear how the performance would be at low bus frequencies.

2.7 Integrating CV data in TSP algorithms

Several recent research studies and a few pilot tests have demonstrated the potential of CV technologies to improve TSP operations (Beak et al. 2018; Cvijovic et al. 2022; Hu et al. 2015; Hu et al. 2014; Hu et al. 2016; Lee et al. 2017; Mohammadi et al. 2020; Teng et al. 2019; Wang et al. 2020; Wu and Guler 2019; Yang et al. 2019; Zeng et al. 2015). Despite the significant progress made by these studies, there is still much effort required to reach

field ready to implement TSP solutions with CV technology. Some of the key challenges of CV technology adoption include the limited penetration, non-compatible field controllers, and other equipment, as well as the communication ranges and frequencies especially between Vehicle to Infrastructure (Beak et al. 2018; Li et al. 2008; Smith et al. 2005; Wu and Guler 2019).

A recent study by Cvijovic et al. (2022) demonstrates the state of art for incorporating CV data in TSP operations. The study formulates and tests a CV-TSP algorithm in a simulation environment in which Basic Safety messages (BSM) and Signal Request Messages (SRM) are exchanged between Onboard Units (OBUs) and Roadside Units (RSU) through Direct Short-Range Communication (DSRC). In a VISSIM® simulation, these processes are implemented using Component object Model (COM) as a virtual server. The modified TSP algorithm showed significantly improved performance compared to the conventional TSP systems. The study estimates ETA by assuming perfect knowledge of the queuing conditions ahead of the bus. An earlier study by Hu et al. (2014) implements almost the same processes but uses a shockwave-based approach developed in (Liu et al. 2009) to estimate ETA.

CV technology provides the means to implement conditional TSP in which TSP is only granted to qualifying buses depending on the lateness, occupancy, reservice time/lockout time, and conflicting priority requests. Performance optimization models can also take CV data in which vehicle locations and occupancy are used as in person-based delay optimization models (Lee and Wang 2022; Zeng et al. 2015).

2.8 Reinforcement learning based signal control and TSP

2.8.1 *Single agent reinforcement learning for traffic signal control*

Reinforcement learning (RL) is an approach of machine learning (ML) in which the learner, also called the agent, learns by continuous interaction with the environment. RL seeks to find the optimal mapping of states to actions, to achieve the highest numerical rewards. RL finds direct application in control systems including traffic signal control. A more detailed description of single agent RL is provided in Chapter 5.

2.8.1.1 Formulation of RL agents for traffic signal control

Initially discussed is RL for overall intersection control, TSP applications will be discussed in a later section. Previous studies have formulated RL-based traffic control algorithms and evaluated their performance in simulation environments. These studies have shown that RL based signal control can potentially be superior to conventional fixed-time and actuated signal control. The control decisions/actions mainly involve deciding the next phase in a fixed or variable phasing sequence (Bouktif et al. 2023; Li et al. 2020; Li et al. 2016; Liu et al. 2022) and deciding the green duration of the next phase in fixed phasing sequence (Aslani et al. 2019; Bálint et al. 2022; Casas 2017; Lee et al. 2022; Li et al. 2021; Shabestary et al. 2020). Table 2 presents a summary of single agent RL traffic control studies. Deep q-network (DQN) and its variations are by the far the most used but a few studies have used policy-based methods such as deep deterministic policy gradient (DDPG), actor critic, and proximal policy optimization (PPO).

Traffic states and rewards are defined/formulated using data that is, or can be, available from traffic sensors in the network and signal controllers, and more recently from CVs. CV data including vehicle positions and speeds allow detailed definition of state and reward

functions. Several recent studies formulate the state as a matrix of vehicle positions and or speeds on the approaches (Li et al. 2020; Liang et al. 2019; Liu et al. 2022; Wang et al. 2019). Matrix formulation forms an image such as cell representation that lends itself to better modeling with convolutional neural networks (CNN).

Most studies adopt discrete action spaces which mainly involve selection of the next phase in a set of the available phases. Some studies consider a fixed phasing sequence where others consider a variable phasing sequence in which the model can rearrange the phase order as well as skip some phases. Most of the reviewed studies assume very simplified signal plans including limiting the number of allowable phases (Aslani et al. 2019; Bálint et al. 2022; Bokade et al. 2023; Bouktif et al. 2021; Bouktif et al. 2023; Li et al. 2020; Li et al. 2016; Li et al. 2021; Liu and Li 2023; Liu et al. 2023; Shabestary et al. 2020), assuming fixed timing sequences (Aslani et al. 2019; Bálint et al. 2022; Bokade et al. 2023; Bouktif et al. 2021; Bouktif et al. 2023; Casas 2017; Chang et al. 2024; Fu et al. 2023; Lee et al. 2022; Li et al. 2020; Li et al. 2016; Li et al. 2021; Liu and Li 2023; Liu et al. 2023; Shabestary et al. 2020; Wu and Guler 2019), fixing the duration or intervals of green for each selected phase or phase combinations (Bálint et al. 2022; Chang et al. 2024; Li et al. 2020; Liang et al. 2019), and utilizing single ring barrier configurations (Bálint et al. 2022; Chang et al. 2024; Fan and Yang 2024; Li et al. 2020; Liang et al. 2019; Wang et al. 2019). For most part therefore, RL approaches remain untested for more complex field signal timing plans.

Table 2: Summary of single agent reinforcement learning studies

Study	RL algorithm and environment	State Definition	Actions	Reward
(Liu et al. 2023)	DQN for single intersection DDPG for network	Number of vehicles on the approach, traffic state	Next phase in fixed sequence	Function of queue length
(Bouktif et al. 2023)	DDQN with PER and SUMO simulation	Number of vehicles, queue length, waiting time, signal state	Next signal phase (variable phasing)	number of vehicles, negative queue length, negative waiting time
(Pang and Gao 2019)	Compares DDQN & DDPG - SUMO simulation	Cell image representation - vehicle positions	DDQN - Next signal phase (variable phasing) DDPG - duration of next phase (fixed sequence)	Average delay difference
(Liu et al. 2022)	DQN - CNN with SUMO simulation	Cell image representation Discrete vehicle positions and speed	Next signal phase (variable phasing)	Functions of queue length and waiting time
(Li et al. 2021)	Compares PPO, DQN, A2C SUMO simulation	Queue length and waiting time	Next signal phase (variable phasing)	Waiting time difference
(Lee et al. 2022)	DDPG Kinetic wave-based models	Space occupancy, delay, green duration	Phase duration in a fixed phase sequence	average delay weighted by traffic volume per lane and entropy of delay in the reward function.
(Bálint et al. 2022)	PG SUMO simulation	Number of vehicles in each lane	Next phase in a fixed sequence which is kept for 5 seconds	Function of traffic load variation across all lanes

Table 2: Summary of single agent reinforcement learning studies (continued)

Study	RL algorithm and environment	State Definition	Actions	Reward
(Li et al. 2020)	DQN-CNN SUMO Simulation	Cell image representation, vehicle locations and vehicle speeds	Next phase in a fixed sequence which is kept for 15 seconds	Cumulative waiting time
(Shabestary et al. 2020)	PPO-CNN SUMO simulation	Cell image representation Number of vehicles in each cell and average speed in the cell	Duration of each phase in a cycle	Average stopped vehicles in the cycle
(Liang et al. 2019)	DDQN- CNN with PER SUMO simulation	Cell image representation of vehicle locations and vehicle speeds	Change in the duration of the next phase	Cumulative waiting time between two cycles
(Wang et al. 2019)	Dueling DDQN - CNN	Detector actuations and occupancy, green duration	Next signal phase in a variable phase sequence	Function of number of vehicles and waiting time
(Aslani et al. 2019)	Actor critic	Number of vehicles on the approaches	Green duration selection from discrete values for the next phase in a fixed sequence	Function of waiting vehicles
(Casas 2017)	DDPG Aimsum simulation	Function of vehicle counts, average speed and detector occupancy	Phase duration of each phase in the next cycle	Function of vehicle counts average speed and detector occupancy
(Li et al. 2016)	DQN Paramics simulation	Queue length on all approaching lanes	Next phase in a fixed sequence	Difference in queue length
(Chanloha et al. 2015)	Q- learning with Cell transmission model	Number of vehicles on the approach	Next signal phase in a variable phase sequence	Delay during red light only
(El-Tantawy et al. 2014)	Q-learning Paramics simulation	Cumulative delay, queue lengths, arrival volumes	Next phase in a fixed sequence and next phase in a variable sequence	Minimizing delay, queue length, number of stops

Table 2: Summary of single agent reinforcement learning studies (continued)

Study	RL algorithm and environment	State Definition	Actions	Reward
(Bouktif et al. 2021)	Parameterized – DQN SUMO simulation	Queue length in each lane, current phase	Next phase in a variable sequence and the duration of selected phase	Queue length

2.8.1.2 Traffic simulation environments

The reviewed studies have used microscopic, mesoscopic, and macroscopic simulation engines in developing and testing RL models. Studies commonly use available off-the-shelf microscopic simulation models including SUMO (Bálint et al. 2022; Bouktif et al. 2023; Li et al. 2020; Li et al. 2021; Liu et al. 2022; Long et al. 2022; Pang and Gao 2019; Shabestary et al. 2020; Shen et al. 2023; Yang and Fan 2024; Zhong et al. 2023), PTV VISSIM® (Cheng et al. 2022), Aimsun (Casas 2017; Hu et al. 2023) and Paramics (Li et al. 2016). In addition to allowing a realistic replication of traffic flow in a network, to implement RL, microscopic simulation engines should allow interaction during simulation run time, including state observation and signal control adjustment. For example, SUMO and PTV VISSIM® provide the Traci and COM application programming interfaces (API), respectively (Lopez et al. 2018; PTV 2021). These interfaces allow access to most parameters during simulation. Additionally, to allow running of several hundred or thousands of simulations within a reasonable time frame, the selected simulation engine needs to provide a high level of run time efficiency. As seen in Table 2, SUMO is by far the most used microscopic simulation environment which most likely stems from the run time efficiency of the platform.

2.8.2 Multi agent reinforcement learning for traffic signal control

Multi-agent reinforcement learning (MARL) involves multiple agents interacting with a common environment and with each other. A more detailed description of MARL is provided later in Chapter 6. The following discussion focuses on overall intersection control with MARL rather than TSP, which will be discussed in a later section. A few recent studies have extended RL based traffic control to multiple intersections and networks. A common approach is to have each intersection controlled by a single agent. In the conventional arterial traffic signal control, coordination between adjacent signals is key to progressing the main line traffic platoons through multiple intersections. To achieve the same control with RL, individual intersections agents must work cooperatively towards maximizing networkwide rewards. Research efforts in multi-agent MARL for traffic control are mainly focused on how to achieve cooperation between individual intersection agents. Table 3 presents a summary of the most relevant studies applying MARL to traffic signal control. To achieve coordination some studies have used the paradigms of centralized training and decentralized execution (CTDE), mainly Q-mix. Other studies have focused on developing communication modules that allow intersections to exchange coded messages which may include the traffic state, and actions taken by the adjacent agents. There are also studies that use a combination of CTDE and explicit communication between adjacent agents. One of the key challenges of MARL is scalability. The “Network scale” column of Table 3 gives the size of the test network in the different studies.

Fu et al. (2023) tested multi-agent DPPG (MADDPG) and Qmix on two networks each with 4 signalized intersections. For MADPPG, independent DDPG agents control each intersection with decentralized training and decentralized execution (DTDE). The authors claim that the local optimums for each agent also formed the global optimum for the network. The algorithms were compared on the total rewards, speed of convergence, and stability. The study reported that multi-agent DDPG converged slower than Qmix but showed higher total rewards compared to Qmix.

Chen et al. (2021) enhanced the conventional Q-mix with an LSTM based communication module to improve coordination between the intersections. The LSTM module exchanges coded messages containing local historical actions and historical and current observations between adjacent intersections. The enhanced Q-mix is compared with the base Q-mix and other multi agent algorithms including independent q-learning (IQL), and independent advantage actor -critic (A2C) on a hypothetical 5*5 grid network and a real-world network of 27 intersections. For both networks, the performance of the conventional Q-mix and the enhanced Q-mix in terms of total rewards, convergence, and average vehicle speeds seem very comparable although the authors claim superior performance with the enhanced Q-mix. Comparable performance could partly be because the agent neural networks of Q-mix already have memory cells (RNN) to keep track of historical observations and states. IQL shows surprisingly good performance, especially with the hypothetical network despite some occasional instability issues. The reported performance of IQL reinforces the hypothesis that if sufficiently trained local agent optimums may also be the global optimums.

Chang et al. (2024) compares the performance of different MARL approaches for network of 9 intersections. The study explores what the authors term as “implicit” and “explicit” communication strategies between agents. Implicit communication is used to refer to the cooperation that occurs between agents in CTDE paradigms such as Q-mix and VDN. Explicit communication is used to refer to the direct exchange of observations and actions between agents using recurrent neural networks such as LSTM. The standard Q-mix with implicit communication is enhanced by adding a LSTM based explicit communication module. The enhanced Q-mix is compared with other advanced algorithms with only either implicit or explicit communication strategies. The base models for comparison also included centralized training and centralized execution (CTCE) DQN and IQL. Enhanced Q-mix showed improvement over only implicit and only explicit communication methods as evaluated by queue length, waiting time and travel time and without excessively increasing the training time. Intuitively, CTCE converged in less episodes, but the episodes took much longer total time to complete revealing the known problem of the “curse of dimensionality”.

Using Q-mix as the base, Bokade et al. (2023) focused on training each agent to learn a communication policy to know which other agents to communicate with and establish the most relevant information to send to each recipient. Recurrent neural network (RNN) based communication networks for each agent send messages to recipient networks on the forward pass and receive feedback from the recipient agents in form of gradients on the backward pass. The developed framework is tested on a 4x4 grid hypothetical network and a real-world network with seven intersections in SUMO. The communication enhanced

network performed significantly better than base Q-mix using queue length, wait time, and mean speed as performance metrics.

Chao et al. (2022) used a global agent to facilitate the coordination of the local agents. The global agent takes the global state and actions of all local agents and adjusts the actions of the local agents to achieve the best global reward. Both the local agents and global agent are based on multi-gent DDPG with the length of green time selected as the action. The study reports good results for a network of five intersections but is likely to have scalability challenges as the global agent must select the global actions to compare with the locally selected actions. Additionally, the study considers that each local agent has access to the local states at all intersections which may unnecessarily increase computational complexity.

To model multi agent cooperation, Wang and Wang (2023) used Friend-DQN based on friend or foe algorithm developed by Littman (2003). At time step $t+1$, an action taken by the agent at time t is compared with the actions of all other agents at time t . The objective of the training is to learn the selection of local actions that are global optimum. Wu et al. (2020) develops IQL for a four intersection network control focusing on the robustness of the trained algorithm and the generalization to different network complexities, congestion levels, demand variations and sensor data noise. Liu and Li (2023) uses IQL enhanced with a communication module that exchanges actions of neighboring agents to improve signal coordination. Actions in this study consist of phase length and are taken as an indication of phase traffic state at the neighboring agents.

As seen from Table 3, only a few recent studies have attempted to formulate MARL-based traffic control algorithms. These studies almost universally use value-based algorithms. However, recent literature shows that policy gradient based algorithms show better performance and convergence especially in partially observable environments (Morales 2020). In a study intriguingly entitled “the surprising effectiveness of PPO in cooperative multi-agent games”, the authors compared the performance of multi-agent proximal policy optimization (PPO) and other centralized training decentralized execution (CTDE) algorithms including Q-mix for four multi-agent environments. PPO showed comparable or superior performance and required minimum hyperparameter tuning

Table 3: Summary of MARL for traffic control studies

Study	RL algorithm	State	Action	Reward	Network scale
Fu et al. (2023)	MADDPG & QMix	Cell image representation	Next phase with a fixed length	Function of throughput, average speed, waiting time and queue length	4 intersections
Chen et al. (2021)	Q-mix with added communication module	Wait time, number of vehicles	Next phase in a variable sequence	Combination of queue length and wait time	5x5 grid and 27 intersections
Chang et al. (2024)	Qmix and Q-mix with communication	Number of vehicles, queue length, current phase	Next phase in a fixed sequence	Total delay for all vehicles	9 intersections
Bokade et al. (2023)	Q-mix with added communication module	Number of vehicles, average speed, queue length, current phase	Next phase in a variable phase sequence	Queue length	4X4 grid

Table 3: Summary of MARL for traffic control studies (continued)

Study	RL algorithm	State	Action	Reward	Network scale
Wu et al. (2020)	IQL	Current phase, phase duration, number of vehicles	Next phase in a fixed sequence	throughput	4 intersections
Liu and Li (2023)	IQL with communication	Cell image representation of vehicle position and speed	Reduce or increase phase length by fixed amount	Weighted combination of queue length and waiting time	3x4 grid

2.8.3 Single agent RL-based TSP

A few recent studies have extended the RL based signal control algorithms to include TSP. The common approach of incorporating TSP is to modify either one or both of the state and reward functions to include bus flow and its performance metrics (Cheng et al. 2022; Hu et al. 2023; Long et al. 2022; Shen et al. 2023; Yang and Fan 2024; Zhong et al. 2023). Except for Hu et al. (2023), the rest of the studies design second by second RL signal controllers incorporating bus parameters in the state and/or reward functions. The same algorithm is deployed to control general traffic and buses with bus entries populated with zeros when there are no buses on the approach. Weighting factors are utilized to define the level of desired priority for buses compared to other traffic. These factors are commonly based on vehicle occupancy and bus schedule deviations. The majority of these studies inflate the bus arrivals/frequencies to generate the necessary samples for the agent to learn TSP control. Long et al. (2022) considers 12 to 60 buses/hour, Shen et al. (2023) considers 32 buses/hour, and Zhong et al. (2023) considers 37 buses/hour. Such frequencies create scenarios where a bus arrives within the time horizon of the disruption in the traffic caused

by the TSP response to the previous bus. However, in the real world, bus arrivals are often less frequent occurrences. Agents trained with high bus frequencies may be biased to priority movements and may unnecessarily penalize nonpriority movements during implementation in lower bus arrival scenarios.

2.8.4 Multi-intersection/coordinated transit signal priority

In conventional TSP systems, and most especially decentralized TSP systems, TSP strategies are selected and implemented locally and independently at each intersection on the arterial. Priority time granted to the transit bus at the expense of general traffic may go to waste if the bus hits red at the next downstream intersection. Recognizing the problem of independent TSP strategies for each intersection, a few research efforts have been made to develop algorithms to implement coordinated TSP strategies across multiple intersections. Most of these studies are based on mathematical programming mainly involving mixed integer linear programs (MILP) (Hu et al. 2015; Ma et al. 2013; Truong et al. 2019; Zeng et al. 2015; Zhai et al. 2023). Recent studies by Li et al. (2023) and Long and Chung (2023b) apply multi-agent RL to implement coordinated TSP across multiple intersections.

2.8.4.1 Mathematical programming approaches

Zeng et al. (2015) introduced the person-based approach to arterial signal optimization which involves an objective function that minimizes the total person delay as opposed to total vehicle delay. The study formulates an online MILP model to estimate optimal signal timings at any two consecutive intersections on the arterial. Decision variables are the green splits at every intersection in each cycle with a fixed phasing sequence. By keeping

a fixed cycle length and fixed phasing sequence, optimizing green splits implicitly optimizes the offsets. The formulation integrates strategies, constraints, and penalties to maintain arterial progression and assumes knowledge of vehicle occupancies for both buses and general traffic. For a five-intersection network, the formulated algorithm showed better performance compared to fixed signal timings estimated with TRANSYT-7F. The developed formulation serves as an alternative approach to balancing tradeoffs between general vehicles and transit vehicles and resolving conflicts between conflicting bus requests. The person -based evaluation approach has been adopted by several other studies.

Hu et al. (2015) developed a MILP algorithm for coordinated TSP across multiple intersections. When the bus is detected, the algorithm estimates the optimal green reallocation signal timings at downstream intersections within a given threshold distance. The algorithm is updated on a rolling horizon basis and although signal timings are estimated for multiple intersections, only timings at the immediate intersection are implemented. The developed algorithm showed superior performance compared to the TSP algorithm implemented independently at each intersection especially when the intersection spacing is below 0.5 miles. The testing considered a dwell-time of average 30 seconds and 2 second standard deviation. A closely similar set up is taken by Truong et al. (2019).

Similar to Hu et al. (2015), Truong et al. (2019) estimates TSP strategies for multiple downstream intersections but only implements the selected strategies at the immediate intersection. The objective of the later study is to predict TSP strategies and associated priority times (GE and EG) that minimize expected bus delay within the considered arterial segment. A stochastic model for dwell-time is formulated to estimate the expected dwell-time at bus stops. The Truong et al. (2019) study reported extra 10% reduction in bus delay

for the coordinated TSP compared to CTSP. However, the study reported more impacts to the side street in terms of increased vehicle delay for the coordinated TSP compared to CTSP.

Ma et al. (2013) formulated a mathematical programming-based model to select coordinated TSP strategies for intersections falling within adjacent bus stops. The objective function of the model was to minimize the bus travel time between successive bus stops and to minimize ineffective priority time where a bus receives priority at one intersection but does not progress through all the intersections within the bus stops. The study reported significantly higher bus travel time savings and significantly reduced impacts to side street traffic for the coordinated TSP model compared to the conventional TSP implemented independently at each intersection. Zhai et al. (2023) formulates a MILP to select coordinated TSP strategies across multiple intersections with the underlying aim of minimizing bus bunching. In this study dwell-time is modeled by assuming a linear relationship with the number of passengers waiting at bus stops.

2.8.4.2 RL-based approaches

Long and Chung (2023a) extends the single agent RL algorithm developed in Long et al. (2022) to MARL to implement TSP cooperatively across multiple intersections. The study utilizes VDN, a CTDE approach discussed earlier to centrally learn a global action value function. The local states are as in Long et al. (2022). Two action formulations are considered (1) selecting the next phase in a fixed phasing sequence which basically involves deciding to keep or terminate the current phase and (2) selecting the next phase in a variable sequence of eight phases. The reward function is taken as the average person

delay on both buses and general-purpose vehicles. With priority requesting buses in multiple movements, RL agents are trained to select actions that minimize total person delay at the network level. In the performance evaluation, the fixed and variable phase sequence RL implementations are compared with a coordinated fixed timing plan designed in SIDRA. However, it is not clear from the paper how the fixed timing plan considers the priority to buses.

Li et al. (2023) formulates and tests a DQN based CTCE algorithm to implement TSP for self-driving buses across two intersections. The state is defined as a global state including the number of vehicles in each lane, average speeds in each lane, and vehicle arrival rates at both intersections. The joint action space involves selection of phase times at both intersections. The common reward is a combination of average delay and average queue length at both intersections. The study claims superior performance of the developed TSP algorithm compared to CTSP. However, the comparison is based on an overall intersection delay which could be masking out the impacts to the side street

2.9 Literature review summary

Early TSP studies mainly based on simulation focused on evaluating the performance of TSP under different conditions and parameters settings. These studies showed that under the right conditions, TSP has the potential to reduce transit vehicle delays, improve reliability and schedule adherence, and mitigate bus bunching. Although the potential for positive TSP benefits is quite well known, the magnitude of the reported benefits and disbenefits varies widely across studies. Some studies report significant improvement of bus quality of service with TSP while others report modest benefits. For the general traffic,

some studies report minimal TSP impacts while others report significant deterioration of level of service after TSP.

Some studies have shown improved TSP performance when the overall arterial signal timings including cycle lengths, offsets and splits are selected considering TSP objectives as opposed to accommodating TSP on top of signal timings optimized for only general traffic flow. With more and more transit agencies equipping their fleets with APC equipment, large datasets of APC data are available to deduce passenger loads, ridership and dwell-time. A few studies have shown the impact of dwell-time on TSP performance including the advantages of far-side stops compared to near-side bus stops. However, not many studies have leveraged these increasingly ubiquitous APC data to evaluate the impacts of field measured dwell-time on TSP performance and later alone establish strategies of integrating dwell-time in TSP system design.

Recent TSP research is focused on improving the performance of conventional TSP by formulating and testing more adaptive TSP algorithms. The adaptive TSP algorithms perform online optimization to select the optimal TSP strategies considering the traffic state and bus ETA. Increasingly, the adaptive TSP algorithms integrate CV data. CV data allow detailed definition of traffic state and estimation of ETA, provides the means to evaluate TSP requests including resolution of conflicting requests and implementation of conditional TSP. Almost universally, these algorithms utilize mathematical programming to find the optimal TSP strategy and parameters considering the traffic state, conflicting priority requests, bus ETA, signal state, previously served requests etc.

There is heightened interest in developing AI-based adaptive traffic control. RL is a natural choice for control problems. Several recent efforts have formulated and tested RL-based traffic control algorithms in simulation environments. These efforts show that RL-based control algorithms have the potential to outperform conventional fixed and actuated timing plans. These studies mainly rely on CV data to provide detailed definition of traffic state and reward functions. Most RL-based traffic control studies have been limited to single agent formulations, but a few recent efforts have attempted MARL traffic control. The few MARL studies almost universally utilize value-based methods despite recent literature showing that policy-based methods show better performance and convergence especially in partially observable environments. MARL formulations mainly focus on how to achieve cooperation between adjacent intersection agents to allow arterial traffic progression and optimal networkwide performance parameters. These studies have employed both DTDE and CTDE approaches. RL methods remain largely untested for real-world signal timing plans because of the simplifying assumptions on signal timing made almost universally across the reviewed studies.

A few ongoing efforts have attempted to extend RL-based traffic control to include TSP. In these studies, TSP is integrated into RL-based traffic control algorithms by modifying state and reward functions to include transit flow parameters and performance metrics. But like the general traffic control, these efforts have mainly been limited to isolated intersection applications. The ongoing efforts to develop RL-based TSP algorithms build on equally recent and still evolving RL-based adaptive traffic control algorithms. RL-based adaptive control for both general traffic and TSP pretty much remain open research areas.

CHAPTER 3 EXPLORATION OF TSP FUNDAMENTAL PRINCIPLES

3.1 Introduction

3.1.1 Background

As indicated in literature, the effectiveness of TSP is mixed, having been shown to reduce transit vehicle delays in some instances, while having minimal effect in others. Similarly, regarding the general traffic, some studies report minimal impacts while others report significant deterioration in the level of service experience by conflicting movements. Before moving to more advanced TSP systems and algorithms, this study first uses a one intersection model to explore the fundamental principles of TSP in a simulated environment. The conventional CI-CO system is considered. The focus of this chapter is to assess the performance of different TSP strategies, identify critical factors and conditions that affect TSP performance, and evaluate strategies for improved consideration of transit vehicles in signal timing development. The results of this chapter will guide the design of experiments in testing the more advanced algorithms described in the later chapters.

TSP aims to provide transit vehicles with a free flow path through an intersection or a series of intersections or at least reduce wait times. Under the right conditions, TSP has shown the potential to reduce transit vehicle delays, improve reliability and schedule adherence, and mitigate bus bunching (Ekeila et al. 2009; Hu et al. 2014; Lee and Wang 2022; Muthuswamy et al. 2007; Rakha and Zhang 2004). TSP system design involves balancing tradeoffs between providing green time to the priority movement, maintaining arterial

coordination, and minimizing the impact to conflicting vehicle's level of service (LOS). Active TSP detects or tracks transit vehicles along the route and provides preferential treatment at signalized intersections. TSP is typically implemented by either green extension (GE), i.e., extending the current priority phase green to enable the transit vehicle to pass, or early green (EG), which is also referred to as red truncation, in which the preceding phases are shortened to reach the priority phase green as early as possible.

Reported benefits and disbenefits of TSP vary widely across different studies, in part due to (1) the different measures of effectiveness (MOEs) adopted, (2) the wide range of conditions and parameters of the transit vehicles and transit routes, and (3) challenges in controlling for traffic demand, signal control, etc., during performance evaluation. Additionally, there is limited documented guidance on the impacts of the different TSP strategies to both transit vehicles and the general traffic (National Academies of Sciences Engineering and Medicine 2020). Lastly, is the realization of the need for further research into signal timing approaches that incorporate TSP in the optimization objectives. That is, in coordinated arterial systems, signal timing parameters including cycle lengths, splits, and offsets are typically selected to provide optimal service to the general traffic without consideration of transit vehicle operations. Considering TSP as an afterthought potentially limits the flexibility of signal control to accommodate TSP.

3.1.2 Objectives and Significance

Under this chapter, the objective is to isolate and measure the impacts of TSP events and evaluate TSP strategy performance. From this evaluation critical factors and conditions that affect TSP performance will be identified and strategies for improved consideration of

transit vehicles in signal control will be proposed. The following are the specific tasks undertaken:

1. Compare the impact of the TSP strategies, Green Extension (GE) and Early Green (EG)
2. Evaluate TSP performance at different congestion levels
3. Assess the impact of incorporating TSP into cycle length determination.

3.2 Methodology

Simulation experiments are performed on a hypothetical single intersection to evaluate the performance of TSP. PTV's VISSIM® is chosen as the simulation environment for its ability to model: (1) TSP through its Ring Barrier Controller (RBC) module, and (2) transit vehicle operation using the public transport (PT) line submodule. During each simulation run, performance data including travel time and delay (bus and general traffic traffic), as well as high-resolution controller data consisting of detector actuations, time of check-in and check-out, signal changes, and TSP events are collected and archived for post processing. Section 3.2 presents details of the tools and methods used.

3.2.1 RBC- TSP Algorithm

A study by Zlatkovic et al. (2010a) compared VISSIM®'s RBC with Econolite's ASC/3 also available in VISSIM® for TSP implementation. The study reported only minor differences in the TSP logic and performance results of the two controllers. As ASC/3 SILs runs on the same base code as Econolite's ASC/3 field controllers, the study results imply that VISSIM®'s RBC can realistically emulate field traffic controllers for TSP

implementation. Further, in a study by Roy (2023), VISSIM® RBC was also found to closely replicate the performance of Qfree's MaxTime® emulator which runs on the same software as Qfree's field controllers.

RBC's TSP algorithm functionality is described in the RBC manual (PTV 2020). As illustrated in Figure 2, (figure represents author's interpretation of VISSIM® RBC manual logic) the algorithm operates by granting GE or EG to the priority movement or maintaining the current phase timing based on the time within the cycle that the bus is predicted to arrive at the stop line. The stop line arrival time is estimated using the time of receipt of a call from the intersection's upstream check-in detector and ETA from the check-in detector to the stop line. Only a single ETA value may be entered into the RBC. Except for minor differences in the recovery process after TSP, the RBC process is fundamentally the same as several other TSP algorithms described in literature, including Qfree's MaxTime® and Econolite's ASC/3 SIL (Li et al. 2008; National Academies of Sciences Engineering and Medicine 2020; Ngan 2004; NTCIP Signal Control Priority Working Group. et al. 2014; Qfree 2020; Rakha and Zhang 2004; Smith et al. 2005; Zlatkovic et al. 2010a). With RBC, the maximum allowable EG is limited to the remainder of the preceding phase green after serving the phase minimum green or a preset minimum called priority minimum green, whichever is greater. The maximum allowable GE is limited to remaining cycle time after the set minimum green for all phases can be served in the cycle after TSP service. To return to coordination, the phases in the next cycle are proportionally reduced according to their split time, shortening the cycle by the length of the prior GE.

The user can control the algorithm responsiveness and balance the tradeoff between bus travel time and general traffic delay by (1) selecting priority minimum greens which predict maximum EG, (2) selecting maximum GE (within the constraint of the maximum allowable GE), (3) specifying the phases that can be skipped, if any, and, (4) setting the reservice parameter which specifies the minimum time until the next TSP request may be granted.

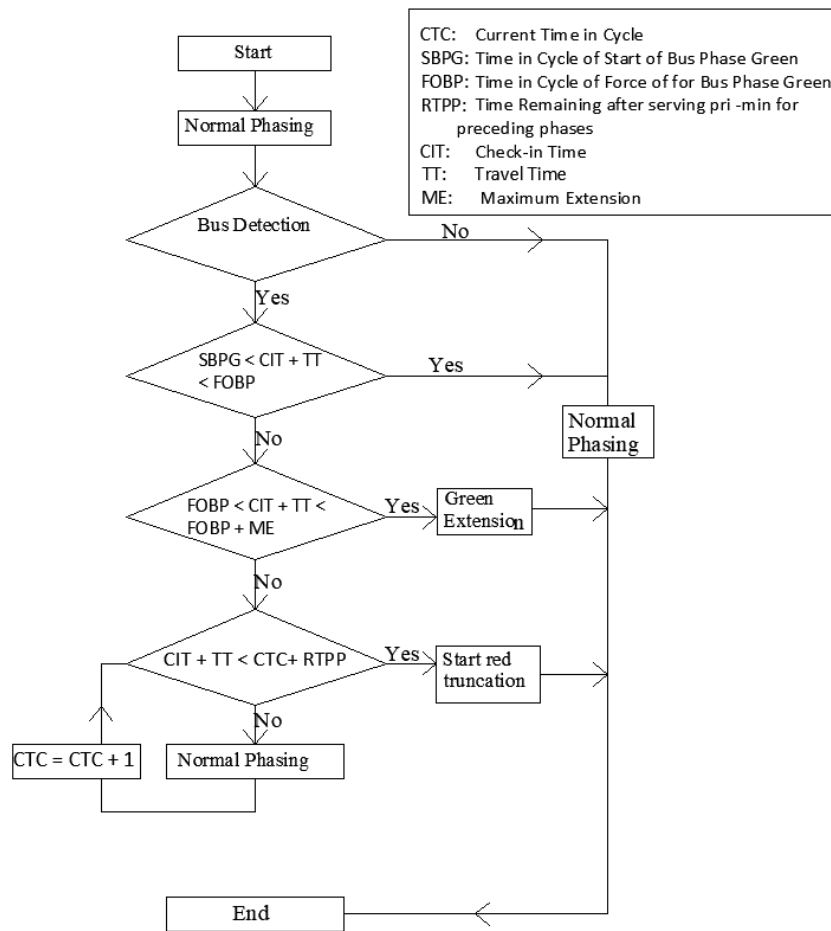


Figure 2: TSP VISSIM® RBC Algorithm (author’s interpretation)

3.2.2 Test network

3.2.2.1 Network geometry

Experiments in this Chapter 3 are performed on a single intersection from a network. The intersection runs in coordination, i.e., as part of a system, with a fixed cycle length and offset to a master cycle. Focus on a single intersection model allows testing of TSP outside the influence of platooned arrivals, allowing for a study of TSP responses not confounded by the arrival pattern. As shown in Figure 3, the model consists of a bus route on the main street (E-W) with the bus running in the coordinated movement. There is a fixed location check-in detector 1000ft upstream of the stop line, a checkout detector immediately after the stop line and a far-side bus stop immediately downstream of the intersection. There is also bus stop upstream of the check-in detector, representing the far-side bus stop of the upstream intersection. The network has a four-lane E-W major street and a two-lane minor street (N-S). All left turns on the major and minor streets have exclusive turn lanes and protected only signal phases. For simplicity, right turning movements are omitted for both streets.

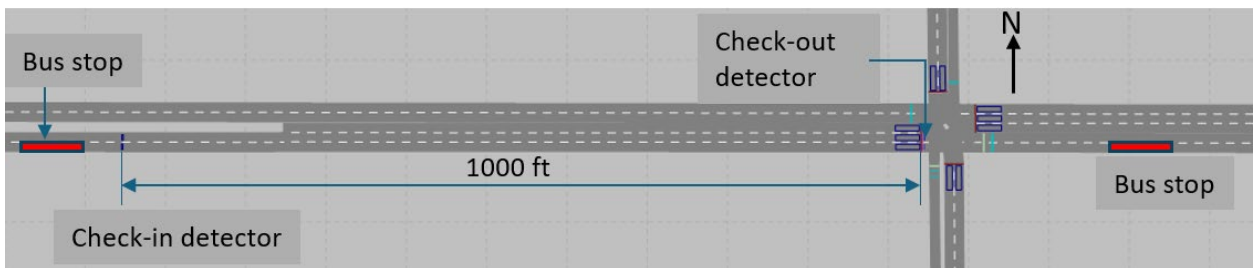


Figure 3: Intersection model in VISSIM®

3.2.2.2 Signal timings

The base model considers actuated-coordinated signal timing with a cycle length of 100 seconds and splits as shown in Table 4. Each phase has a yellow time of 4 seconds and an all-red time of 1 second. Floating force offs are set to allow unused green to go to the main street coordinated through movements. For the experiments, volumes are calculated to achieve the desired overall intersection volume to capacity (v/c) ratio. Both the major and minor streets have balanced volumes in all corresponding opposing directions, for instance Eastbound through (EBT) volume is equal to westbound through (WBT) and Eastbound left turn (EBL) volume is equal to westbound left turn (WBL) volume.

Table 4: Signal Timing Parameters

Signal Group (SG)	1	2	3	4	5	6	7	8
splits	15	45	20	20	15	45	20	20
Yellow	4	4	4	4	4	4	4	4
All-Red	1	1	1	1	1	1	1	1
Min Green	5	20	5	10	5	20	5	10

Preliminary experiments were run in VISSIM® to extract saturation headways so that matching values are used in calculating volumes for the different degrees of saturation. The resultant saturation headways are 2.0 seconds for the left turn movement corresponding to saturation flow rate of 1800 veh/h and 1.9 seconds for the through movement corresponding to saturation flow rate of 1894 veh/h.

3.2.2.3 Traffic Demand

Given the selected cycle length, splits and the estimated saturation flow rates, demands are set to achieve the desired volume to capacity (v/c) ratios for the given experiment scenarios.

Both the major and minor streets have balanced demands in all corresponding opposing directions. Experiments were conducted for v/c ratios of 0.85, 0.95, and 1.0. Keeping the base model splits, the volumes to achieve the v/c ratios are given in Figure 4. The figures inside the parentheses are the movement volumes in vph for v/c of 1.0, 0.95 and 0.85, in that order.

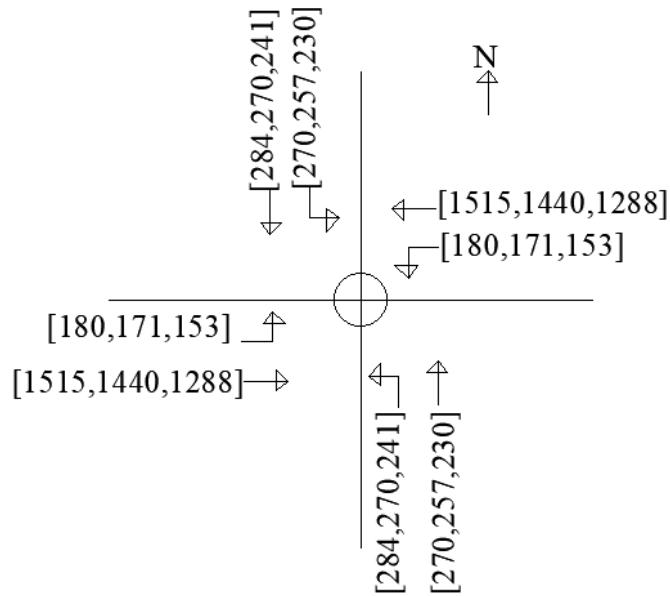


Figure 4: Demand (vph) for different v/c ratios

3.2.2.4 Bus inputs

Transit buses enter the system from the beginning of the main street on the west and drive eastward through the intersection. VISSIM® provides the capability to model transit vehicle operations including routes, bus stops, dwell-time, passenger loads, and passenger flows through the modules PT lines and PT line stops. One bus enters the system per hour to allow isolation of the impacts of each TSP event. To capture the impact of bus arrival time in the experiments, bus arrivals were approximately uniformly spread across the cycle

length. To achieve uniform arrival across the cycle length, each bus is introduced after 20 minutes into the hour (allowing for a warm-up period) and an additional random term between zero and the cycle length. Figure 5 shows the resulting number of buses arriving at the check-in detector at a given point in cycle. The data includes 300 buses, generated from 10 replicate trials of 30 hours each, with one bus entry each hour. All experiments in Chapter 3 consider unconditional TSP, in which all arriving buses request and can be granted priority without fulfilling any preset conditions.

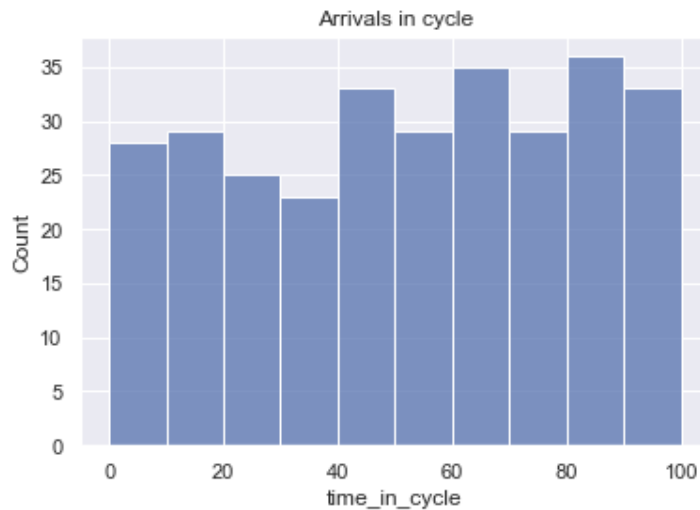


Figure 5: Arrival profile of buses vs time in cycle

3.2.3 Simulation Execution

For the VISSIM® simulations the Wiedemann 74 model is utilized with default parameters. Lane change behavior, acceleration, etc., also utilize default parameters. To capture sufficient bus samples, each simulation scenario is run for 30 hours, with one bus per hour as described above. To ensure independent data for each bus arrival, demands were entered in VISSIM® in intervals of 15 minutes with the last 15 minutes of each hour assigned a minimum flow of 200 vph and 50 vph for the main street through movement

and all other movements, respectively. Reducing the volumes in the last 15 minutes of each hour allows any accumulated queues or other impacts from the prior potential bus priority to clear the network, ensuring independence between bus arrival impacts. Conducting longer runs (30 hours) with assurance of independence between data collection intervals allows for the same statistical properties as multiple replicate trials (i.e., 30 replicate runs) while allowing for increased efficiency in model execution time and data post processing (Law 2015).

3.2.4 Experiment design

In this study, maximum GE was limited to 20% of the cycle length, which is below the maximum allowable GE and is consistent with what several transit agencies apply, as reported by National Academies of Sciences Engineering and Medicine (2020). Compare GE and EG TSP strategies.

The first set of runs explores the impacts and effectiveness of GE and EG. The first experiment allows both GE and EG options, followed by tests of TSP GE only, as GE only approach was undertaken by a subset of agencies in the literature (National Academies of Sciences Engineering and Medicine 2020; Smith et al. 2005). The RBC settings allow the user to select either of the two strategies of (1) GE and EG, or (2) GE only. For the base model with $v/c=0.95$, 10 replicate simulation runs (thus, 300 bus samples, i.e., 30 buses per replicate) were performed for each of the strategies and the baseline no TSP scenario.

Second-by-second controller data allowed for assessing the effectiveness of the TSP strategies, including, proportions of vehicles receiving GE and EG, proportion of successful or effective GE, length of GE, amount of truncation on each phase, and timing

changes in the recovery cycle or cycles. Unsuccessful or ineffective GE refers to when GE is granted and maximum GE is reached before the bus reaches the stop line, resulting in the bus stopping even though GE was granted. In post processing run results, controller data was fused with performance data to isolate and define the impacts of each TSP event. Time periods of up to 15 cycles (25 minutes) after check-in were investigated for differences between no TSP and TSP.

3.2.4.1 TSP performance under different demand levels

It was hypothesized that (1) at lower v/c ratios TSP may result in a higher rate of successful GEs and (2) at higher v/c ratios, GEs and EGs may have larger disbenefits to side street traffic as well as potentially be less effective in serving transit vehicles due to the increased uncertainty in travel time and higher likely of an unsuccessful GE. To evaluate the impact of different congestion levels, starting with a base model with an overall intersection v/c ratio of 0.95, volumes are adjusted while maintaining the same signal timings (cycle length and splits) to achieve v/c ratios of 0.85 and 1.0, as shown in Figure 4. At each v/c ratio, new optimal ETA values are set in the controller to account for varying travel time.

3.2.4.2 Impact of cycle length

Cycle length is normally selected to minimize general traffic delays and to maintain coordination in arterial systems. In common practice, cycle length selection does not specifically consider transit vehicle operations. This experiment is designed to assess the impact of cycle length selection and the tradeoff between general traffic delay and TSP efficiency. It was hypothesized that cycle lengths higher than the optimal traffic-based value (i.e., not considering transit) could improve TSP efficiency by providing more

flexibility in terms of the available maximum allowable GE and EG and reduce the potential negative TSP impacts (i.e., increased delays) to the side street. However, higher cycle lengths may increase overall intersection delay when under normal operation, and thus the trade-off between these effects must be considered.

With the v/c ratio of 0.95 volume set in Figure 4, an optimal cycle length of 110 seconds is found to minimize general traffic delay. The corresponding phase splits are shown in Table 5. For all phases, a total clearance time (Y+AR) of 5 seconds is provided. For the 110 second cycle, a maximum allowable GE set to 20% of the cycle length is equal to 22 seconds. For EG, minimum priority green, which is the green that must be served before truncation, is set as shown in Table 5. A truncation of up to 10 seconds is allowed on side street phases 4 & 8 and 3 & 7, while a truncation of 5 seconds is allowed on the main street left turns, i.e., phases 1 & 5. No truncation is allowed on the main line through movement but this is of little or no consequence as EG calls are mostly placed when phases 2 & 6 are red, and thus the priority movement is not available for truncation. Priority minimum greens shown in Table 5 are computed from the set truncation limits.

Cycle lengths of 130 and 150 seconds, as shown in Table 5, are selected to test TSP performance when the cycle is above the general traffic optimal cycle length. To have similar GE and EG amounts or ranges, the maximum allowable GE and EG limits are same as for the 110 second cycle length. Additionally, because of the need to locate the check-in detector after the upstream bus stop, no meaningful benefits were anticipated from increasing max GE. This was confirmed through preliminary runs, although results are not presented here for brevity. For EG, maintaining the same allowable truncation limits, the corresponding minimum priority green durations for the cycle lengths of 130 and 150

seconds are shown in the table. For each cycle length, 10 replicate simulation runs are made for the TSP and no TSP cases, resulting in a total of 60 simulation runs.

Table 5: Splits and maximum GE for varying cycle lengths

cycle length	Cycle Parameter	Phases								max GE
		1	2	3	4	5	6	7	8	
110	Splits	16	50	22	22	16	50	22	22	22
	Min priority green	6	45	7	7	6	45	7	7	
130	Splits	19	59	26	26	19	59	26	26	22
	Min priority green	9	54	11	11	9	54	11	11	
150	Splits	21	70	30	29	21	70	30	29	22
	Min priority green	11	65	15	15	11	65	15	15	

3.3 Results

3.3.1 Measures of Effectiveness

The selected MOEs included (1) TSP effectiveness and strategies in terms of proportions of granted GE and EG, and proportion of effective GE (i.e., the bus successfully traversed the intersection), (2) transit bus travel time, (3) cross street general traffic delay, and (4) extent/propagation of the delay after the TSP event. The first MOE is mainly presented with tables while the last three are presented with box plots to capture the variability in travel time for different buses and in the resulting delays to the other vehicles.

3.3.2 Comparing impacts of GE and EG

3.3.2.1 Bus travel time

Figure 6 shows the impact of GE and EG on bus travel time for a v/c ratio of 0.95. The plotted data is for a total of 300 buses (i.e., 10 replicates of each 30-hour run) for each of TSP strategies and no TSP scenarios. Comparisons between TSP and no TSP are made for

the same bus in the corresponding runs. The “n” values in the figure footnote represent the number of buses compared under each strategy. For instance, n = 105 means that there are 105 buses in GE only case being compared to the corresponding 105 buses in the no TSP scenario. The “n” value is included in all similar plots in the chapter and has similar interpretation unless stated otherwise. It is seen that GE results in higher levels of improvement over no TSP than EG. The better performance of GE is intuitively reasonable as GE results in a bus skipping the entire red duration whereas EG reduces the amount of red by the length of the EG truncation. The “No-Act” stands for buses that do not request for TSP.

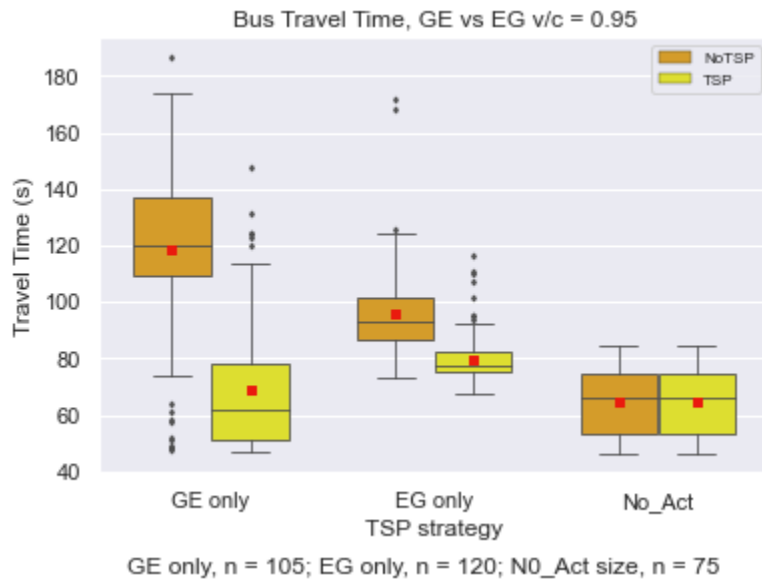


Figure 6: Impact on bus travel time at v/c ratio of 0.95 for GE and EG

3.3.2.2 Cross street delay

Figure 7 shows a comparison of cross street (a) Southbound through (SBT) and (b) Southbound left turn (SBLT) delays for GE and EG at the v/c ratio of 0.95, with a 100 second cycle, as shown in Table 5. The delay is calculated over 4 cycles after the TSP

event. For both movements, the magnitude of the change in delay (between TSP and no TSP) resulting from EG is significantly higher than that from GE. The higher delay for EG is to be expected as the impact of EG truncation is typically borne by the immediately preceding movement, while the impact of GE is distributed across all movements in the next cycle after TSP.

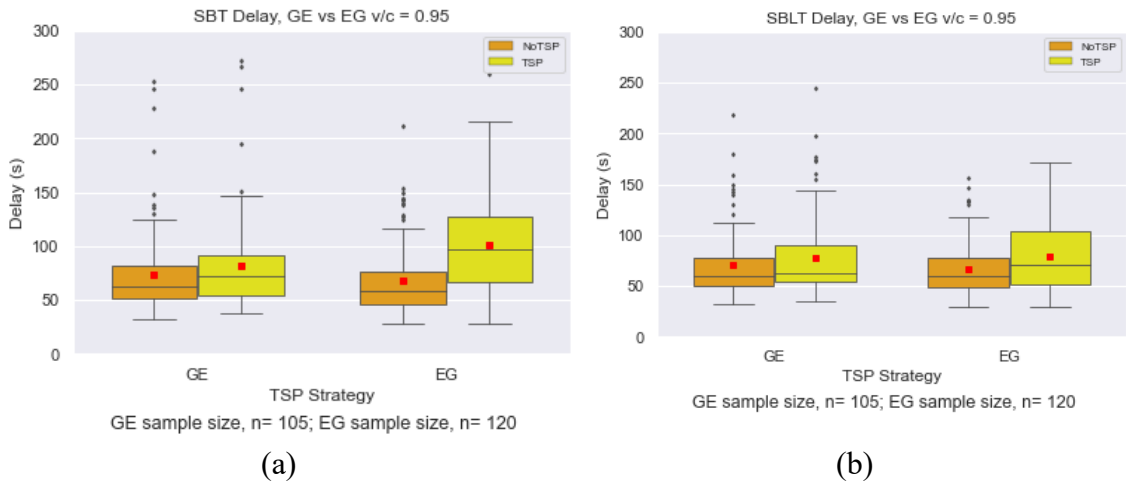


Figure 7: Cross street Delay at v/c ratio of 0.95 for (a) SBT and (b) SBLT

3.3.2.3 Cross street delay extent

Figure 8 shows the extent of delay after up to 10 seconds of (a) GE and (b) EG. The sample size, “n” shown below the figure, is the number of buses receiving EG of up to 10 seconds in the 10 simulation runs. For each of these buses the delay of the general traffic is observed for up to 14 cycles after TSP. The impact of EG extends at least eight cycles while the impact of the GE dissipates after the four cycles. As discussed above, the 10 second truncation is borne by the SBT movement alone while the 10 second GE is spread proportionally to all movements according to the splits.

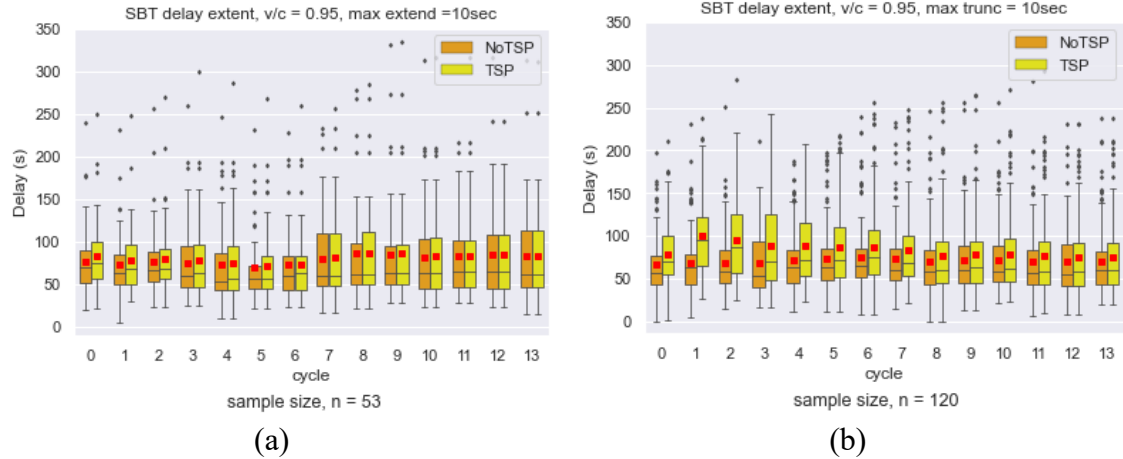


Figure 8: SBT delay extent with v/c ratio of 0.95 after up to 10 second (a) GE and (b) EG

It is seen in Figure 6 through Figure 8 that the GE has higher bus travel time benefits than EG, while EG results in more significant side street delays. While not shown for brevity, similar trends are seen for the v/c ratios of 1.0 and 0.85.

3.3.3 TSP performance under different demand levels

3.3.3.1 TSP Effectiveness

Table 6 shows the TSP strategies and TSP effectiveness for v/c ratios of 0.85, 0.95 and 1.0. In the first TSP strategy, both GE and EG are permitted while in the second only GE is implemented. “No_Action” is the number of buses for which no TSP was requested, “EG” indicates the number of buses that received EG while “GE” indicates the number of buses that received GE. “GE_Eff” is the number of buses that received green extension and successfully traversed the intersection on the provided GE, while GE_NoEff represents the number of buses that were granted GE but maximum GE was reached before the buses was able to traverse the intersection.

From the No_Action and GE columns, it is evident that more buses need TSP as the congestion increases. The need for more TSP service as the congestion increases is expected, as at lower congestion levels there is less queuing and higher speeds on the mainline, as well as additional green available for the coordinated mainline movement because of side street movements gapping out. GE effectiveness decreases with congestion as indicated by the number of failures (GE_NoEff). Increased GE failures are attributed to increased ETA uncertainty as congestion increases.

Table 6: TSP strategies and effectiveness under different levels of demand

TSP Strategy	v/c	EG	GE	GE Eff	GE_NoEff	No_Action
GE & EG	0.85	119	62	62	0	119
	0.95	120	105	100	5	75
	1	129	123	100	23	48
GE only	0.85	0	62	62	0	238
	0.95	0	109	104	5	191
	1	0	134	111	23	166

3.3.3.2 Bus travel time

Figure 9 shows the travel time for only buses receiving (a) GE and (b) EG. A higher bus travel time is seen as the congestion level increases from v/c ratio of 0.85 to v/c ratio of 1.0 in the baseline no TSP case. The variability in travel time also increases as congestion levels increase. As seen in (a), for v/c's of 0.85 and 0.95, GE reduces the travel time for this subset of buses to almost free flow travel time. However, for the v/c ratio of 1.0 several vehicles receiving GE still face queuing and therefore the average travel time is higher than the free flow travel time and there is high travel time variability for buses receiving GE. Moreover, the failed GE buses sit through the entire red period and as seen in Table 6, and the number of failures increases with v/c ratio. Part (b) of the figure reinforces an earlier

observation that GE provides higher benefits to the buses compared to EG, as GE allows buses to skip an entire red phase.

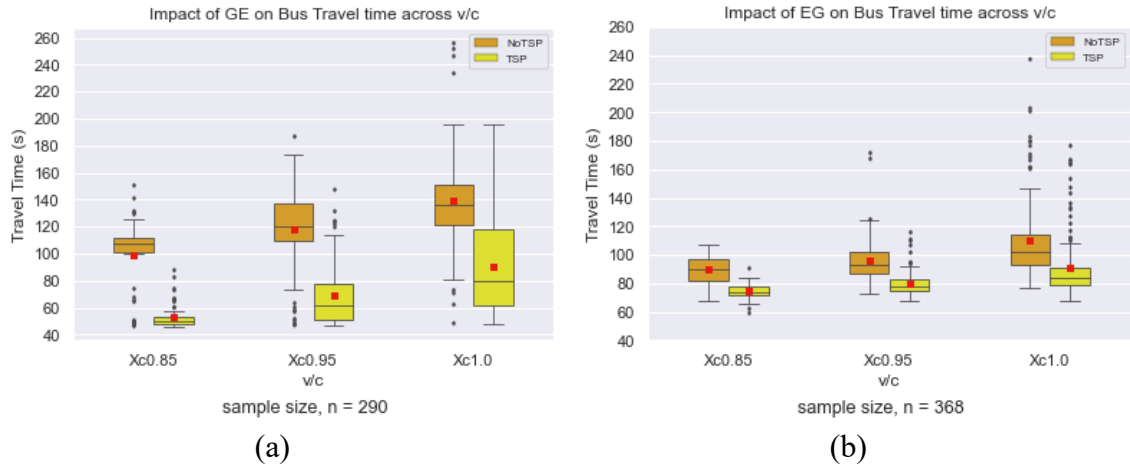


Figure 9: Impact on bus travel time at three levels of v/c ratio, for (a) GE and EG and (b) GE only

Figure 10 shows bus travel time for the 0.85, 0.95, and 1.0 v/c scenarios, under combined GE and EG, GE only, and no TSP. All buses are included whether they request or do not request TSP. All TSP strategies, for all v/c ratios, significantly outperform no TSP. In addition, for all v/c cases the combined GE and EG performs better than the GE only, with slightly lower average travel times and less travel time variability. The increase in travel time saved results from increased TSP requests and service, as shown in Table 6. However, the improved bus travel time benefit of the GE and EG strategy may be negated by increased delay to the side street, as discussed in delay results in the next section. For the GE only strategy, travel time reduction increases as v/c increases from 0.85 to 1.0.

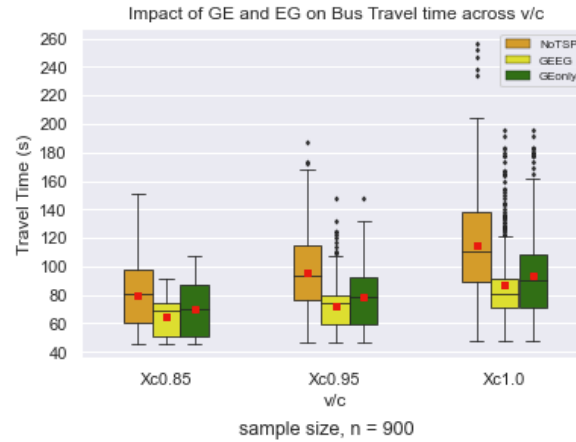


Figure 10: Bus travel time for 0.85, 0.95, and 1.0 v/c scenarios, under combined GE and GE, GE only, and no TSP

3.3.3.3 Cross street delay

Figure 10 presents the cross street SBT movement delay with (a) both GE and EG and (b) GE only implemented. To allow for comparison across v/c ratios delay is measured for four cycles, starting with the check-in cycle. As expected, the higher v/c ratios of 0.95 and 1.0 show higher delay increases compared to the v/c of 0.85. With a v/c ratio of 0.85 the additional delay is practically insignificant. For all v/c levels, the delay changes are lower for GE only strategy, which reinforces the earlier observation that EG tends to higher disbenefits to the side streets compared to GE.

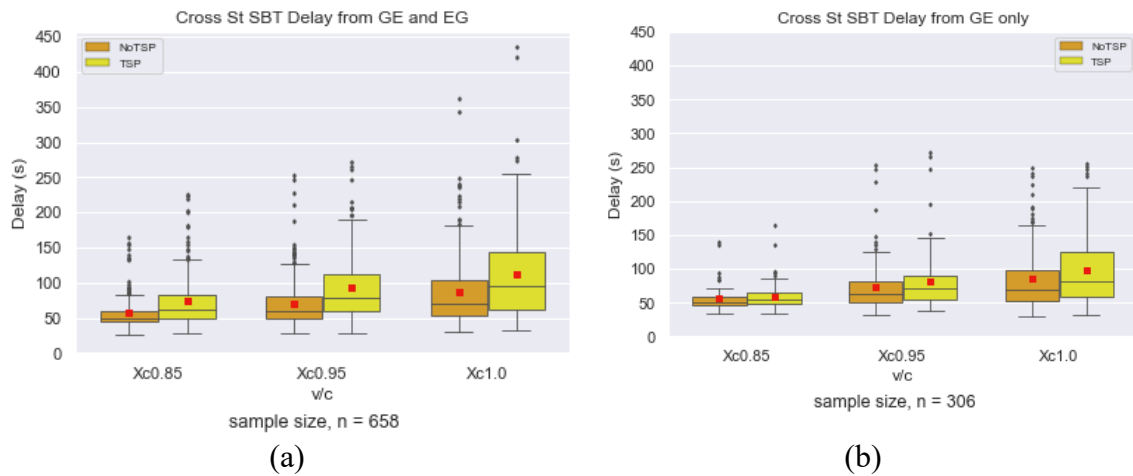


Figure 11: Cross Street Delay from GE and EG for (a) SBLT and (b) SBT

3.3.3.4 Cross street delay extent

The three v/c ratio demand levels are further compared in terms of the extent of delay caused by different levels of EG to the side street movements. For brevity, results are presented for only SBT movement and for only the maximum truncation of 10 seconds. Figure 12 shows the SBT delay extent caused by truncation of up to 10 seconds for (a) $v/c = 1.0$, (b) $v/c = 0.95$ and (c) $v/c = 0.85$.

For a v/c of 1.0, the delay increase is significant and persists through the observed 14 cycles after truncation, for the set maximum truncation of 10 seconds. For the v/c of 0.95, the delay increase persists to approximately the 9th cycle, recovering earlier than in the v/c ratio of 1.0. Additionally, the magnitude of the side street delay is lower when the v/c ratio is 0.95. Lower delay for v/c of 0.95 is as expected as the 0.95 v/c ratio allows for some slack in the timing to recovery from the TSP. For the v/c ratio of 0.85, the delay increase caused by 10 second truncation is only observed in the first six cycles, and the delay magnitudes

again decrease. For all v/c levels, while slightly muted, similar trends were seen when maximum truncation was set to 5 seconds (figures not shown).

The trend in these results demonstrates the critical importance in having slack in the signal timing plans (i.e., v/c sufficiently below 1.0), where the desire is to limit the impact to conflicting non-TSP movements.

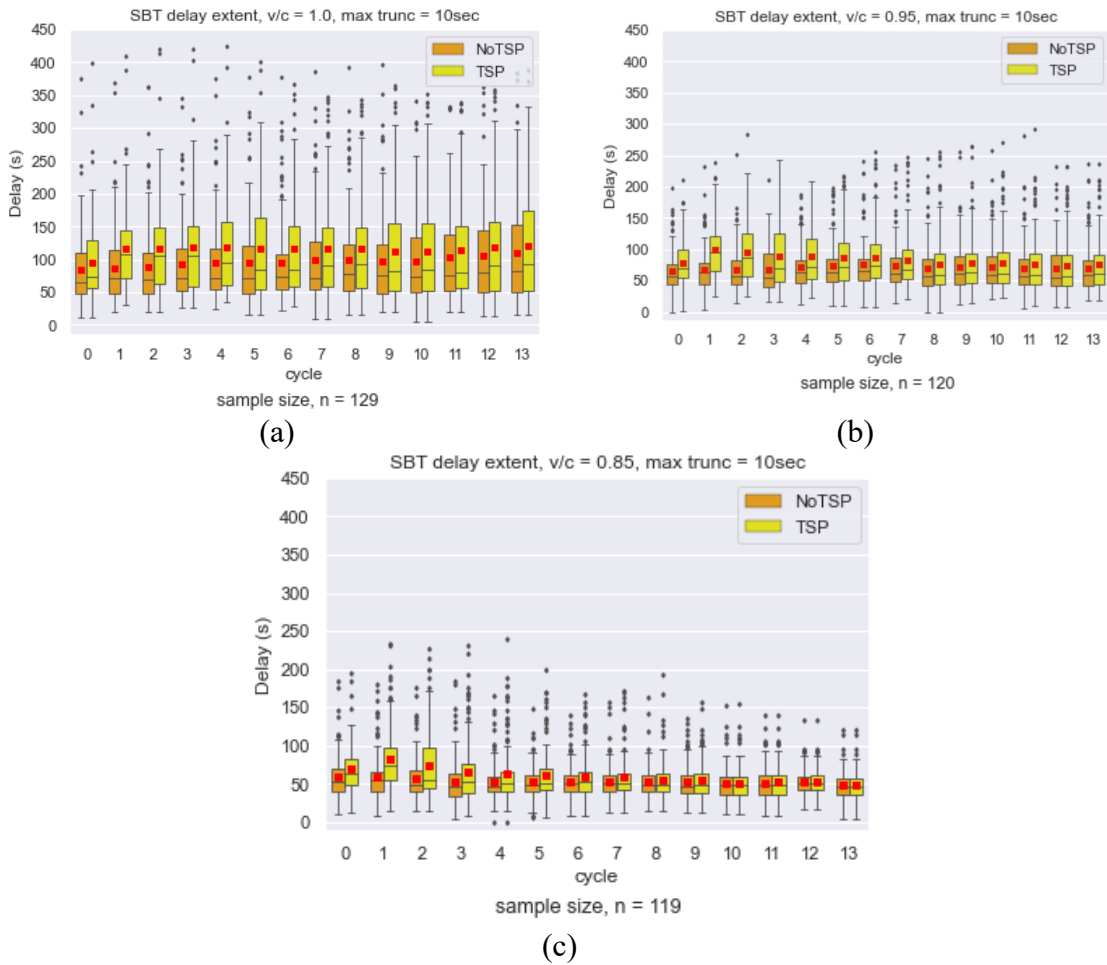


Figure 12: SBT Delay Extent after up to 10 second truncation for, (a) $v/c = 1.0$ (b) $v/c = 0.95$, and (c) $v/c = 0.85$

3.3.4 Impact of cycle Length on TSP Performance

3.3.4.1 TSP Effectiveness

Analysis of the TSP strategies and their effectiveness at different cycle lengths showed that (1) the number of buses not requiring TSP increases slightly as the cycle length increases, which is attributed to increased mainline green, and (2) due to increased red phase duration as the cycle length increases the number of buses requesting EG increases, while the number requesting GE decreases. As mentioned, the proportion of GE and EG partly determines the TSP effectiveness, with higher GE preferred.

3.3.4.2 Bus travel time

Figure 13 compares bus travel times with and without TSP for different cycle lengths. In Figure 13 (a) only TSP affected buses are included in the analysis while in Figure 13 (b) all buses are included. In both cases (a) and (b), travel time savings reduce as the cycle length increases. The maximum bus travel time benefit, when considering only buses that receive TSP, is provided by the optimal cycle length of 110 seconds. The higher benefit for the cycle length of 110 seconds is likely due to increasing percentages of EG and decreasing percentages of GE as the cycle length increases. When considering all buses, the 110 second cycle continues to provide lower bus travel times, although the differences in bus travel time between cycles lessens. Note that this part of the experiment limits maximum GE and EG for the two higher cycles to that of the optimal cycle, as constrained by the detector location due to presence of the upstream bus stop. Extending the detector further upstream would allow higher maximum GE and potentially more time savings for the higher cycle lengths.

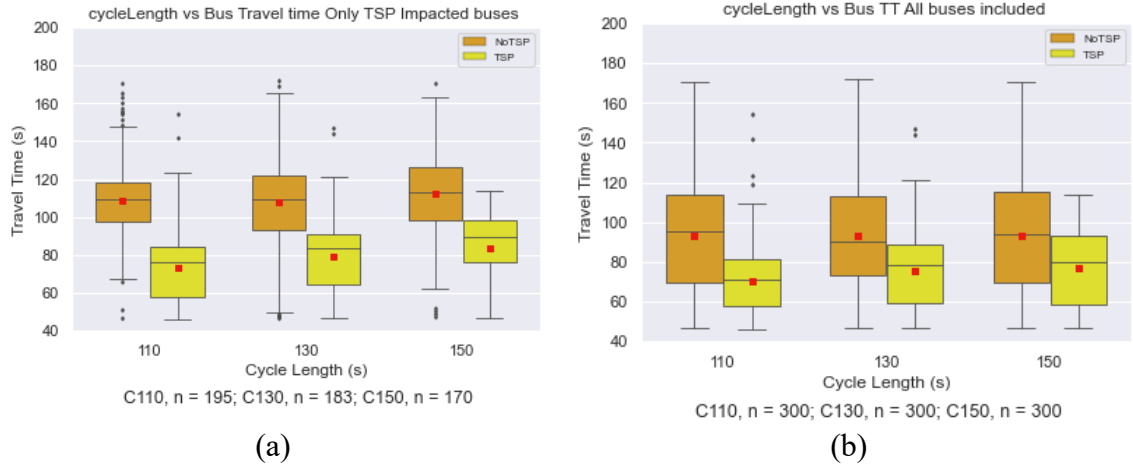
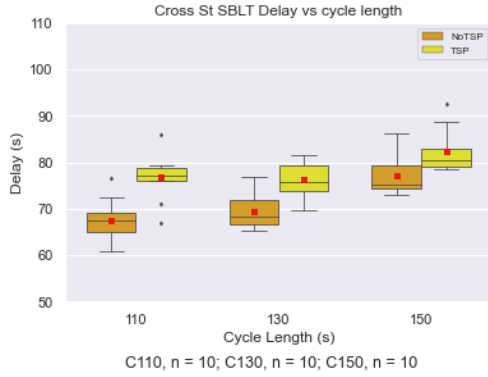


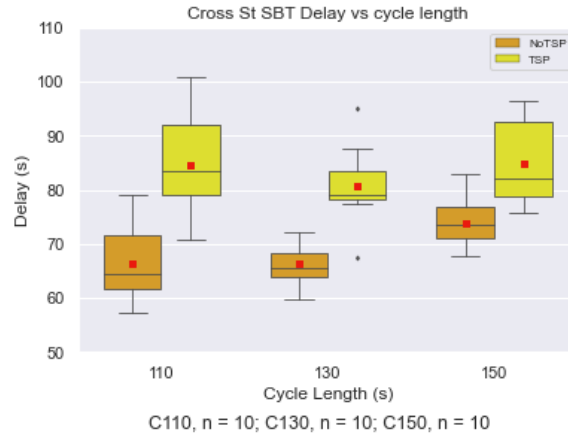
Figure 13: Bus travel time vs cycle length for (a) Only TSP affected buses and (b) All buses

3.3.4.3 Cross street delay

Figure 14 (a) and (b) shows the cross street SBLT and SBT delays respectively, for only those bus traversals where TSP was provided. Delay is averaged for 4 cycles after TSP. As expected, in the no TSP case delay increases as the cycle length increases from the optimal cycle length. The change in delay after TSP service is lowest at 130 seconds. The result reinforces the hypothesis that slack inherent in the higher cycle serves to absorb the increased delay after TSP events. However, as the cycle continues to increase, the general increase in delay due to increased cycle length outweighs the benefits of additional slack to absorb truncation. Thus, TSP is less disruptive to side street traffic at moderately higher cycle lengths than the optimal cycle length. This becomes a tradeoff between accepting higher base line delays (No TSP) and lower delay increases after a TSP event. The decision may further be guided by the frequency of TSP requests. For higher v/c ratios (i.e., 1.0) these trends become more severe, with side street delays during TSP at the lower cycle significantly higher than the higher cycle. This result may lead to a decision to not implement TSP under high v/c conditions.



(a)



(b)

Figure 14: Cross street delay vs cycle for (a) SBLT and (b) SBT

3.3.4.4 Cross street delay extent

Figure 15 presents the extent of the change in delay for SBT movement after truncation of up to 10 seconds for a (a) cycle length of 110 seconds, (b) cycle length of 130 seconds, and (c) cycle length of 150 seconds. For the cycle length of 110 seconds, delay increases are experienced up to eight cycles after truncation. For the cycle length of 130 seconds, delay increases persist for only the first three cycles, and the absolute value of the increase in delay per cycle is also less than in the 110 second cycle scenario. Therefore, moving from a cycle length of 110 seconds to 130 seconds, for the maximum truncation of 10 seconds, the time to dissipate the increase in delay after truncation reduces from up to eight cycles to three cycles. For the cycle length of 150 seconds, similar results to that of the 130 seconds cycle are seen, again delay increases for approximately three cycles after truncation. A maximum truncation of five seconds demonstrated similar, although slightly muted, results for all three cycle lengths (figures not shown).

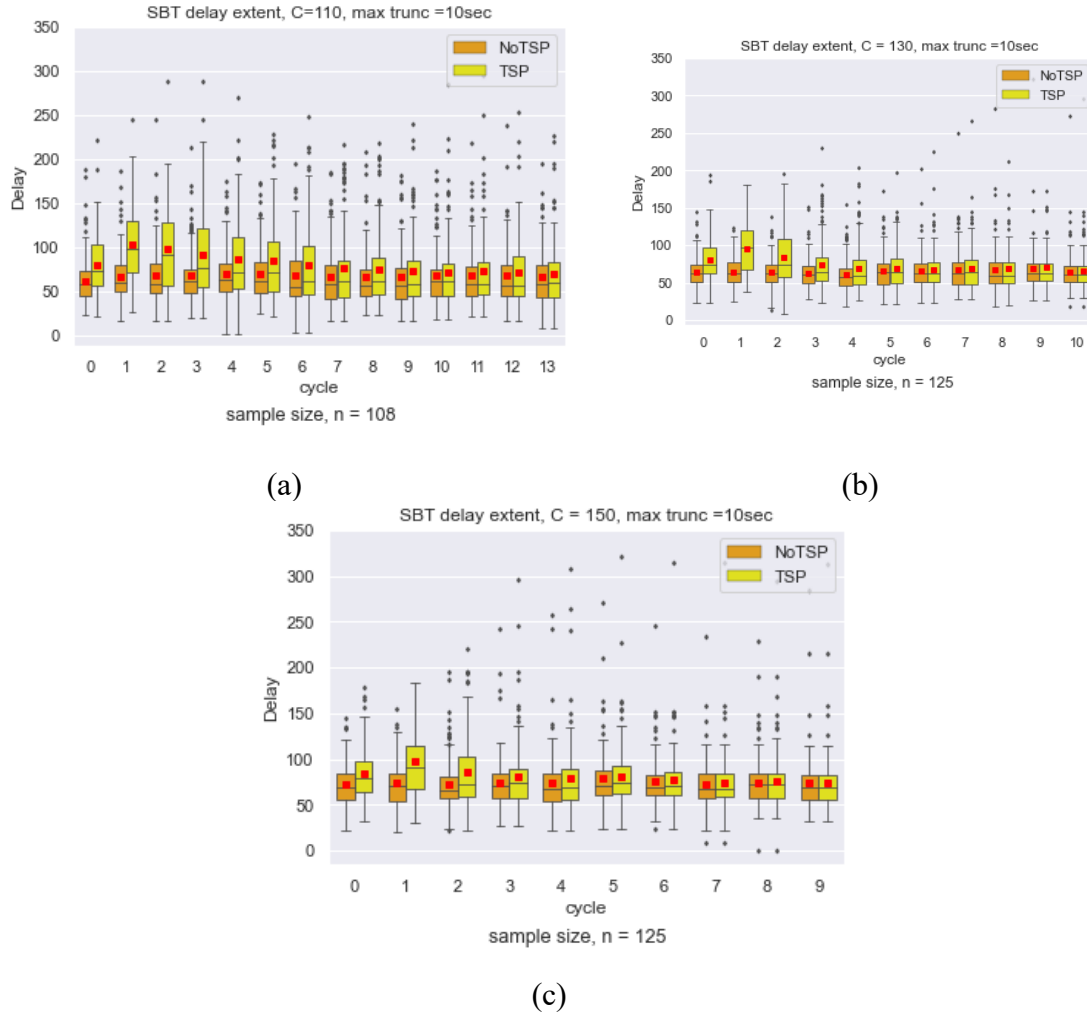


Figure 15: SBT Delay extent after truncation of up to 10 seconds for (a) cycle length of 110s, (b) cycle length of 130s and (c) cycle length of 150 seconds

In summary, increasing the cycle length from the optimum increases the baseline delay, i.e., the overall intersection delay when there is no TSP. However, the higher cycle lengths provide more slack to absorb delays from TSP events. For high v/c ratios the delay due to TSP may be unacceptable, with the impact lasting potentially many cycles. While a cycle length increase may result in an overall delay increase, it may also provide a means to allow for acceptable service during TSP.

3.4 TSP Exploration Summary

Chapter 3 uses a VISSIM® simulation environment to evaluate the performance of TSP strategies and establish the critical factors and conditions that affect TSP performance. Critical items assessed are TSP strategies, general traffic demand, and cycle length.

On comparing the two TSP strategies of GE and EG it was found that (a) GE tends to provide more significant bus travel time improvements than EG, (b) GE tends to have a lesser impact on conflicting movements than EG, and (c) the conflicting movement delay for the same number of seconds of GE versus EG tends to be higher and last for more cycles when EG is granted.

As congestion increases (modeled by increasing v/c ratios in this study) increases are seen in all the following: the percentage of buses requesting TSP, the likelihood of an ineffective GE, the proportion of buses receiving of EG, side street impacts, and the number of cycles for the sides side street delays to return to non-TSP levels. For a v/c ratio of 1.0 the side street may not return to non-TSP levels for an extended number of cycles past the TSP event, at least 14 cycles in this study. Thus, the likelihood of the impact of the previous bus still being present when the next bus arrives increases as v/c increases.

As cycle length increases (for the same demand) the proportion of buses requesting TSP (GE) decreases, while the effectiveness of GE requests improves. In addition, bus travel time averaged across all buses increases slightly and cross street delay due to TSP is lowest when the cycle is moderately higher than optimal (where optimal is calculated to optimize non-transit traffic demands). Cross street delay not in the presence of TSP increases as cycle length increases beyond optimal; however, the number of cycles required to dissipate

the effect to the side street reduces as well as the impact per cycle when a TSP call is served.

Overall, it is seen that TSP performance is most favorable in lower v/c conditions and GE provides the most benefit to individual buses. However, as congestion increases the effectiveness of TSP decreases. On a highly congested corridor, i.e., v/c ratios approaching or exceeding 1.0, it is possible that TSP may become infeasible as the conflicting non-TSP movements may have insufficient slack in available capacity to recover from the TSP-related green truncation.

Underlying TSP is balancing transit and other vehicle demands. The signal timing for optimal TSP performance, when considering both the transit vehicle and conflicting movements, may not be optimal when TSP is inactive. One implication of this finding is that the setting of a corridor's signal timing parameters should reflect the corridor's purpose. That is, if the corridor is designated to serve a significant transit function, then base timing parameters should be selected to improve bus travel time and to lessen TSP impacts on general traffic. Where transit is not a primary focus, then higher cycle lengths or TSP based offset may not be warranted.

CHAPTER 4 IMPACT OF DWELL-TIME ON THE PERFORMANCE OF TRANSIT SIGNAL PRIORITY

4.1 Background

In coordinated arterial systems when buses stop midblock to serve passengers, they typically fall out of the progression bandwidth. Depending on the magnitude of dwell-time the buses may not be able to traverse the intersection during the current green. In a fully coordinated and under saturated system, the magnitude and variability of dwell-time may largely determine the bus arrival profiles at the intersection stop line. And as shown by Roy (2023) and Rakha and Zhang (2004) the performance of preemption and priority systems are affected by the priority vehicle arrival profiles within the signal cycle. Passive transit signal priority (TSP) systems are designed to favor the progression of transit buses through signalized intersections without actively detecting bus movements and altering signal timing. The successful design of passive TSP requires good knowledge of dwell-times at bus stops and bus speeds within the block. Active TSP requires accurate prediction of ETA between the priority request point and the intersection stop line. For near-side bus stops where the priority call may need to be placed before the bus passes the bus stop, dwell-time needs to be included in ETA prediction and depending on the variability of dwell-time, ETA uncertainty may be greatly increased.

TCQSM defines dwell-time as the time spent at a bus stop serving passengers including time to open and close the doors (Transportation Research Board and National Academies of Sciences 2013). The definition excludes the deceleration and acceleration lost time while

getting into and out of the stop respectively. Dwell-time data can be collected manually onboard the bus with handheld devices recording the time for first door opening and last door closing. Manual data collection can also include passenger number counts. With the ubiquity of Automated Passenger Count (APC) systems which automatically and continuously record the passenger numbers and door activity, dwell-time data is mostly now deduced from APC data. The availability of large APC datasets including real time passenger load can potentially allow the development of better ETA prediction models, and more efficient and adaptive TSP systems.

4.2 Problem definition and study objectives

A few studies have shown that dwell-time magnitudes and variability affect TSP performance at near-side bus stops (Dion et al. 2014; Ngan 2004; Rakha and Zhang 2004). For internal intersections in coordinated systems, the arrival time of the transit bus at the intersection stop line may largely be determined by the dwell-time magnitude at the bus stop within the block. There are limited research efforts to evaluate the impacts of dwell-time magnitude and variability on TSP performance. Additionally, TSP studies that incorporate dwell-time in modeling TSP have mainly used deterministic dwell-time (Bayrak and Guler 2020; Dion et al. 2014; Rakha and Zhang 2004) or assumed dwell-time distributions that may not be reflective of actual field dwell-time distributions (Ekeila et al. 2009; Hu et al. 2014; Ma and Yang 2007). Dwell-time is known to be mainly affected by passenger activity, but the inherent trends and distributions of dwell-time data are not well established in literature. Several simulation studies have adopted normal distribution in modeling dwell-time but as shall be seen in the results section normal distribution does not fit field data.

Given these gaps in research, the objective of this chapter is twofold; (1) to analyze field dwell-time data and identify any underlying trends and distributions and (2) to assess the impact of dwell-time magnitude and variability on TSP performance. The impact of dwell-time on TSP performance is studied in a simulated environment and both far-side and near-side bus stops are considered.

4.3 Analysis of APC Data

4.3.1 Data Description

APC data for this study were provided by Metropolitan Atlanta Rapid Transit Authority (MARTA), the principal public transit operator for the Atlanta metropolitan region. APC data also included some AVL components, such as a record of the time the bus enters and leaves a 200 ft buffer zone around each stop. The sourced data were for the three busiest transit routes, route 39, route 05, and route 83, for one year from August 2021 to August 2022. Within this period, although bus ridership had not fully recovered to pre-pandemic levels, significant recovery had been observed as shown by a recent report by Federal Transit Administration (Federal Transit Administration 2024) For brevity, only results of route 39 are presented. Route 39 was the busiest of the three. Route 39 runs along Buford highway connecting Lindbergh and Doraville Transit stations serving 45 bus stops in each direction. A snapshot of the route is shown in Figure 16.

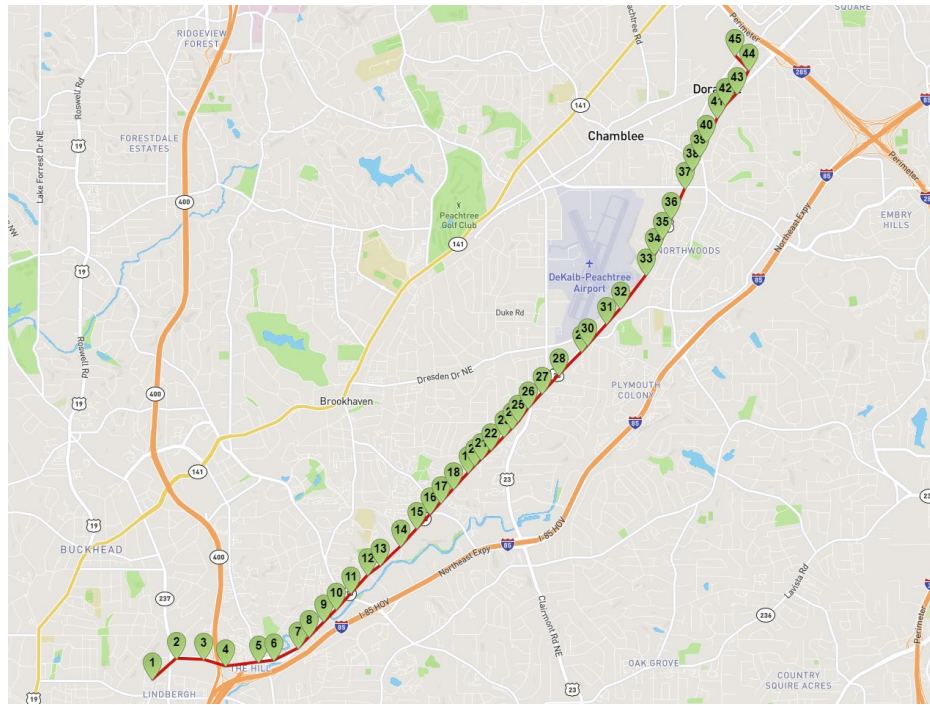


Figure 16: MARTA Bus Route 39

On route 39, MARTA operates 60' articulated buses with three doors and a total capacity of 81 passengers. Each of the three doors can be used for both boarding and alighting and are all equipped with APC sensors. APC sensors record the time for each door opening and closing and the number of passengers going through each door. Dwell-time was deduced as the difference between last door closing time and first door opening time.

4.3.2 Data Pre-processing

APC data is prone to issues including sensor errors and thus the first step was to perform data cleaning. Records with obvious data recording issues, which may be due to sensor faults or operator errors, were removed. These included missing vehicle ID, missing Trip ID, an exceedingly high number of boarding and alighting, and erroneous stop IDs. Records at garages (start and end stations) and schedule time points were separated out as

the bus is not just stopped to serve passengers but is also waiting for dispatch according to the schedule. Additionally, the garages had concrete shades which interrupted GPS transmission leading to missing entries. Records with wheelchair entries were separated out as the lift operations take more time and biases dwell-time. Records for weekends and holidays were also separated as the target was weekday commutes. Before pre-processing the raw data had 2,379,395 records which reduced to 1,036,101 after preprocessing. The most common issue was the records missing the vehicle ID, at 903680 records. Records missing vehicle ID also had no entries for arrival and departure time, passenger and door activity and, therefore, dwell-time could not be computed.

4.3.3 Data Analysis Results

4.3.3.1 Dwell-time by bus stop.

Figure 17 shows Route 39 southbound (SB) dwell-time cumulative distribution functions (CDFs) for (a) all stops all day, (b) all stops AM peak, (c) selected stops all day and (d) selected stops AM peak. Each curve represents a bus stop. From Figure 17 (a), it is seen that dwell-time varies from stop to stop. The vertical axis shows the proportion of the data that is below a given dwell-time value on the x-axis. The y-intercepts represent the proportions of the data with zero dwell-time which can be interpreted as the probability of skipping the stop. The probability of skipping a given stop varies from approximately 0.2 for the most frequently used stops to close to 1.0 for stops with almost no activity. It is seen that stops with a high proportion of non-zero dwell-times, also have a high dwell-time variance. Comparing Figure 17 (a) and Figure 17 (b) shows that both dwell-time magnitude and variability increase in the AM peak. For better visualization, Figure 17 (c) and Figure

17 (d) plot CDFs for ten stops with the highest dwell-time. The legend indicates the stop ID (905258 - Buford Hwy NE @ Briarwood Rd, 905463 - Buford Hwy NE @ Plaster Rd, 906132 - Buford Hwy NE@3658, 900568 - Buford Hwy NE @ Lenox Pointe, 905373 - Buford Hwy NE @3610, 901979 - Buford Hwy NE @ 3160, 905255 - Buford Hwy NE @ Afton Ln NE, 905261 - Buford Hwy NE @ 3580, 905252 - Buford Hwy NE @ Knoll Pl NE, 901538 - Buford Hwy NE @ N Cliff Valley Way). For almost all the ten stops the dwell-time magnitude and variance, and the proportion of non-zero dwell-time, increases for the morning peak compared to all-day data.

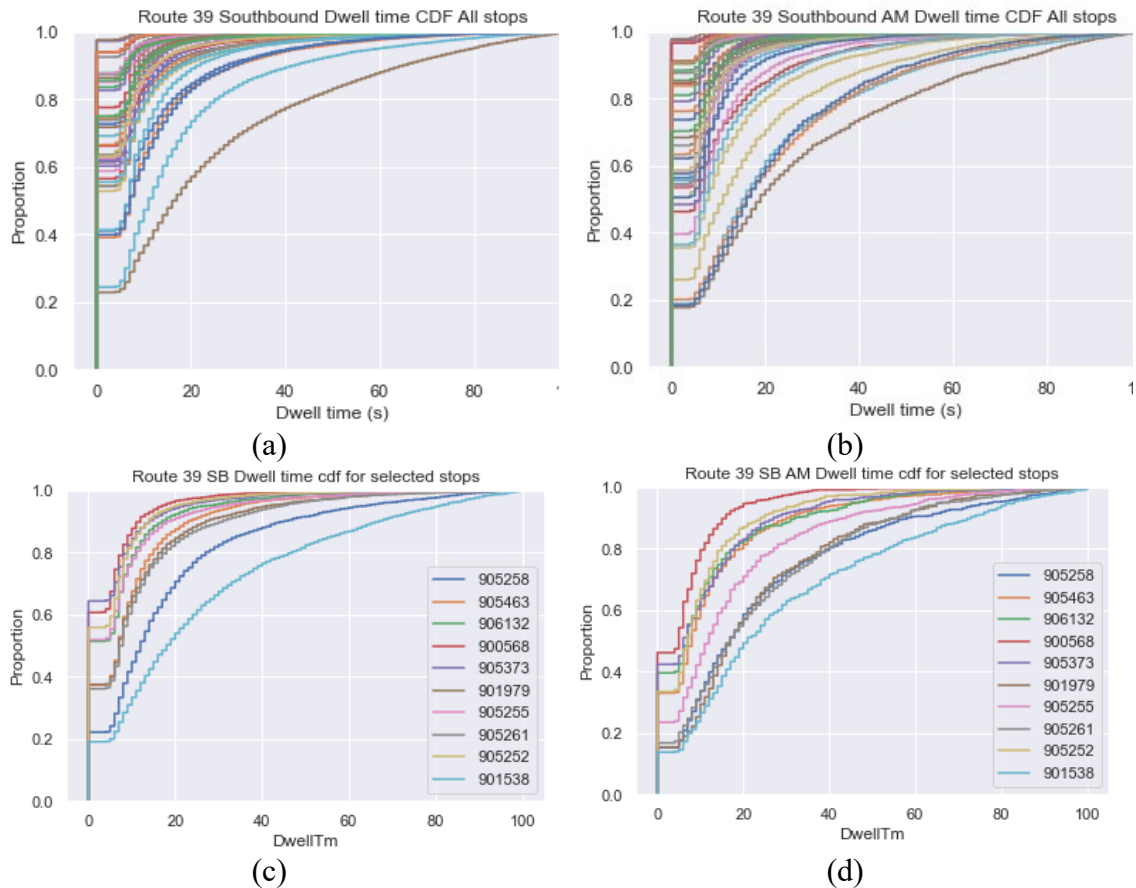


Figure 17: Route 39 SB Dwell-time CDF for (a) all stops all day, (b) all stops AM peak, (c) selected stops all day and (d) selected stops AM peak

Figure 18 shows Route 39 SB 6AM dwell-time variation across stops with (a) zero dwell-time included, and (b) zero dwell-time excluded. The two garage stops at the beginning and end of the route are excluded for reasons discussed earlier. The figure further illustrates the variability of dwell-time across different stops. Excluding zero dwell-time serves to show that when the bus stops, the stop durations vary significantly across the different stops. Out of the 43 stops, 15 stops have average and median dwell-times of close to zero meaning the stops are skipped most of the time. Of the remaining stops, average and median dwell-time are below 20 seconds for 21 stops. The remaining 7 stops have dwell-time averages falling above 20 seconds. Simulation experiments in the second part of the study explore the implications of these dwell-time variations in TSP system design and performance.

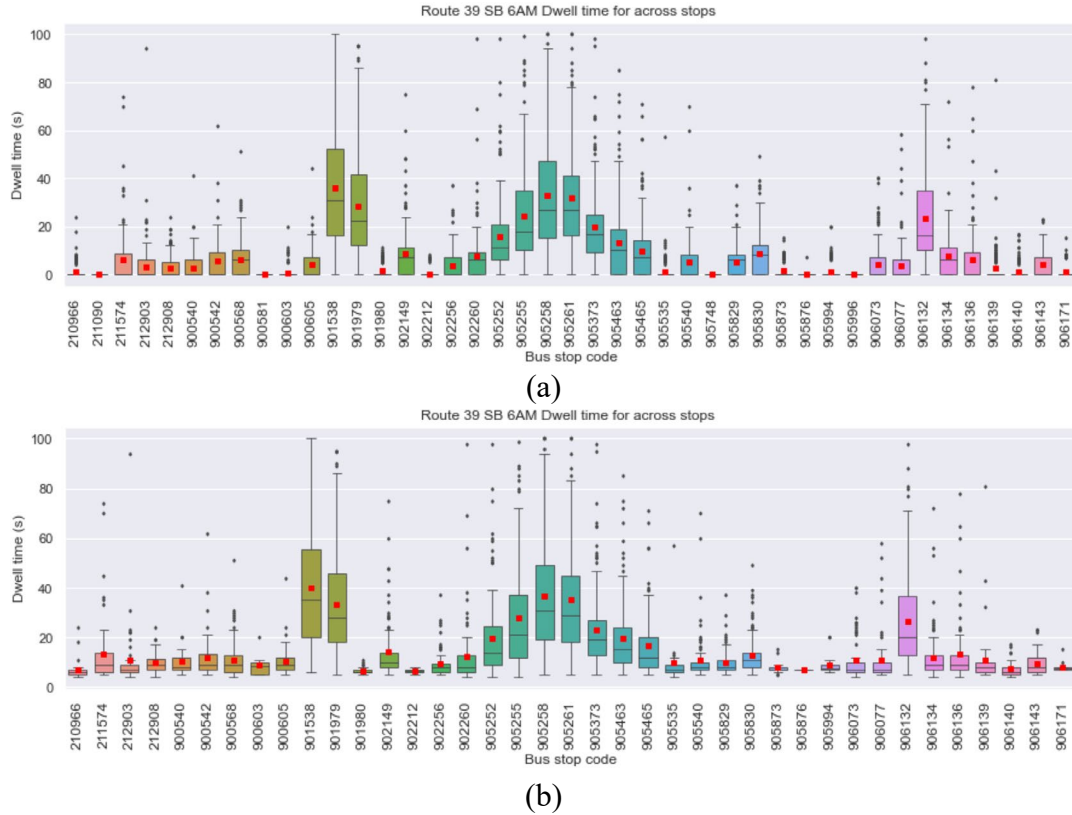


Figure 18: Route 39 SB Dwell-time variation for selected stops, with (a) zero dwell-time included, and (b) zero dwell-time excluded.

4.3.3.2 Dwell-time by time of day

Figure 19 shows dwell-time by time of day (TOD) for two stops with the highest dwell-time in the southbound (SB) direction (a, b) and the northbound (NB) direction (c, d) stops. The x-axis is TOD in hours while the y-axis is dwell-time in seconds. In the SB direction, for the stops shown and for most other stops, morning and afternoon peaks in dwell-time are observed. Average dwell-time is above 20 seconds for the morning peak and afternoon peak for most stops (only two stops shown). For NB, there is an afternoon peak starting from about 11AM for most stops. As shall be seen in the second part of the study, dwell-time at midblock bus stops in part determines bus arrival profiles at the downstream

intersection stop line. Significant variations of dwell-time by TOD may necessitate setting TSP parameters by TOD.

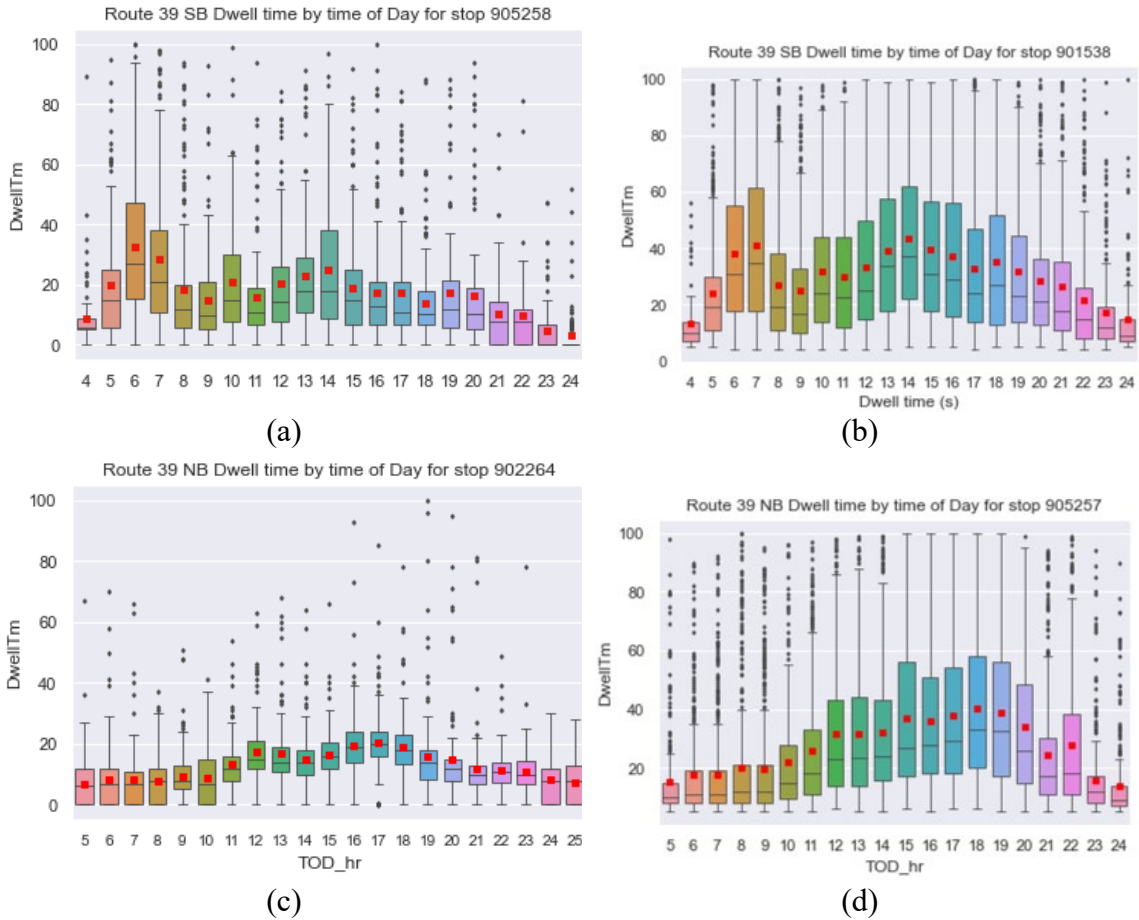


Figure 19: Route 39 dwell-time for selected SB stops (a, b) and NB (c, d) stops by time of day

4.3.3.3 Far-side vs Near-side bus stops

Figure 20 shows dwell-time CDFs for (a) 33 near-side bus stops and (b) 53 far-side bus stops. From visual inspection, it is seen that several near-side bus stops have higher dwell-times than most far-side bus stops. Whereas the difference could be from purely variations in passenger numbers, it is also known that bus drivers/operators tend to keep doors open longer at near-side stops as they wait for the green phase (Dueker et al. 2004; Grisé and El-

Geneidy 2017). To confirm the exact dwell-time at near-side stops, manual surveys onboard the bus could be undertaken to record the end of passenger activity. Additionally, APC data could be fused with automatic fare collection (AFC) data to identify points when the last of the waiting passengers enters the bus.

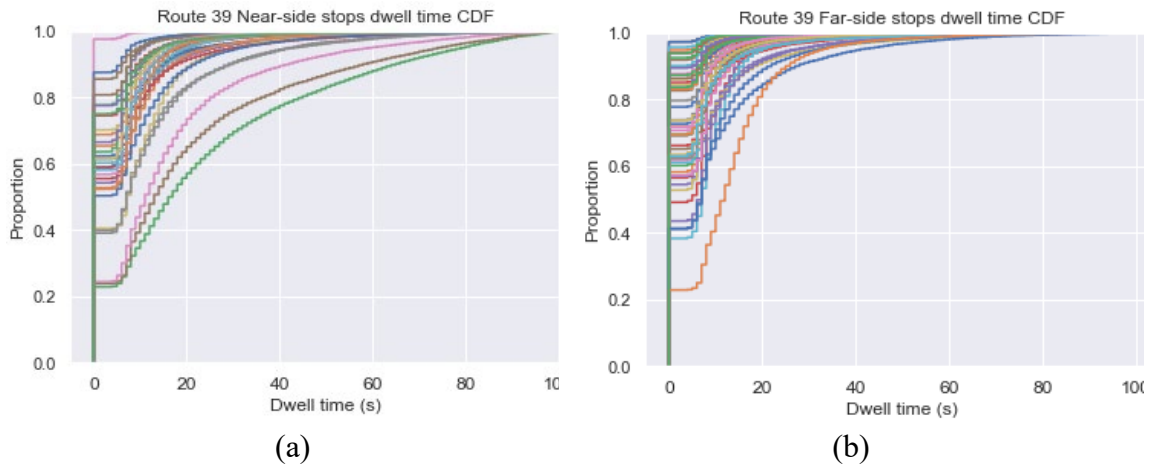


Figure 20: Route 39 Dwell-time CDFs for (a) Near-side bus stops and (b) Far-side bus stops

As indicated earlier, every time a bus stops, the AVL system records the arrival time and departure time into and out of a 200ft buffer zone around the bus stop. Figure 21 shows the time spent in the buffer zone (departure time minus arrival time) for (a) near-side bus stops and (b) far-side bus stops. It is seen that the magnitude and variance for travel time is much higher for near-side bus stops compared to far-side stops. Higher dwell-time at near-side bus stops may be expected as in addition to time for boarding and alighting the near-side stop dwell-time is likely to include the time spent in queue and/or waiting for a green phase, which is not experienced at the far-side bus stop. For near-side stops therefore, dwell-time as defined by TCQSM may not be accurately deduced from only AVL data.

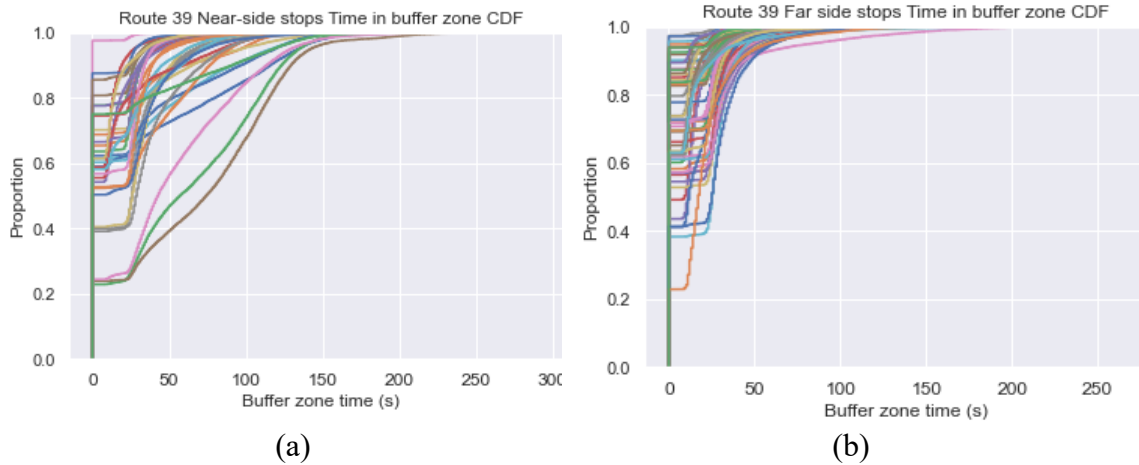


Figure 21: Travel time in 200 ft bus far zone around (a) Near-side stops and (b) Far-side bus stops.

4.3.3.4 Fitting dwell-time data

Exploratory histogram plots of dwell-time indicated that the data could be fitted to some parametric distributions. The “fitter” package in python was used to fit several candidate distributions to stop-by-stop data and the best fitting distributions were identified based on Kolmogorov – Smirnov (K-S) goodness of fit test. Figure 22 shows dwell-time distributions fitted to the AM peak data for two stops. The best fitting distributions for both stops, and for most other stops, were the inverse Gaussian, log normal, power lognormal, fisk (log-logistic), and Johnson’s SU distributions.

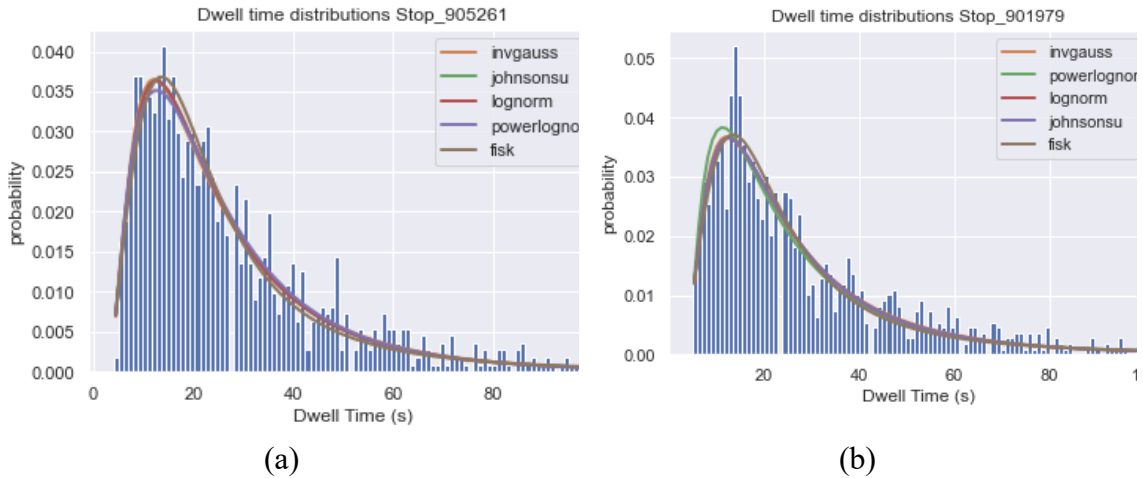


Figure 22: Fitted dwell-time distributions for stop (a) 905261 and (b) 901979.

Of the five distributions shown in Figure 22, inverse Gaussian was the best fitting for both stops. For stop 905261, the inverse Gaussian had parameters of $\mu = 0.608$, $\text{loc} = 0.377$, $\text{scale} = 43.89$, and a ks p-value of 0.293. For stop 901979, the inverse Gaussian had parameters of $\mu = 0.633$, $\text{loc} = 0.803$, $\text{scale} = 42.06$, and a ks p-value of 0.110. These high p-values mean that there is no evidence to reject the null hypothesis that the fitted distributions are the same as the observed distributions.

4.4 Impact of Dwell-time on TSP performance

4.4.1 Overview

Analysis of the dwell-time data in the preceding section showed that field dwell-time can vary widely for the same stop and from stop to stop. Section 4.4 presents the design and results of simulation experiments to assess how the dwell-time magnitude and variability affects TSP performance. Experiments are performed with PTV's VISSIM® as the simulation environment on a network consisting of a pair of coordinated signalized intersections.

4.4.2 Test Network

The objective was to carry out TSP experiments on a single intersection that is part of a coordinated arterial system. As shown in Figure 23, the network model consisted of the study intersection as well as an upstream intersection 1500ft to the west. The upstream intersection was included to allow for a platooned arrival pattern on the major road TSP intersection. The main street is four lanes (two lanes each direction) with all intersections having a left turn bay on the eastbound (EB) and westbound (WB) approaches. All north-south (N/S) minor streets are two lanes (one lane each direction) with left turn bays on all approaches. The bus route modeled is EB on the main street. Experiments are performed starting with a far-side bus stop as shown in Figure 23(a) and then a near-side bus stop as shown in Figure 23(b). The far-side and near-side bus stops are located at 1200ft and 100ft, respectively, upstream of the study intersection. The speed limit on both the main and cross street is assumed to be 40 mph and in PTV-VISSIM® the desired speed of all vehicles is set to 40 mph. PTV-VISSIM® default parameters for car following and lane changing models are maintained as this network is not modeled based on a field network.

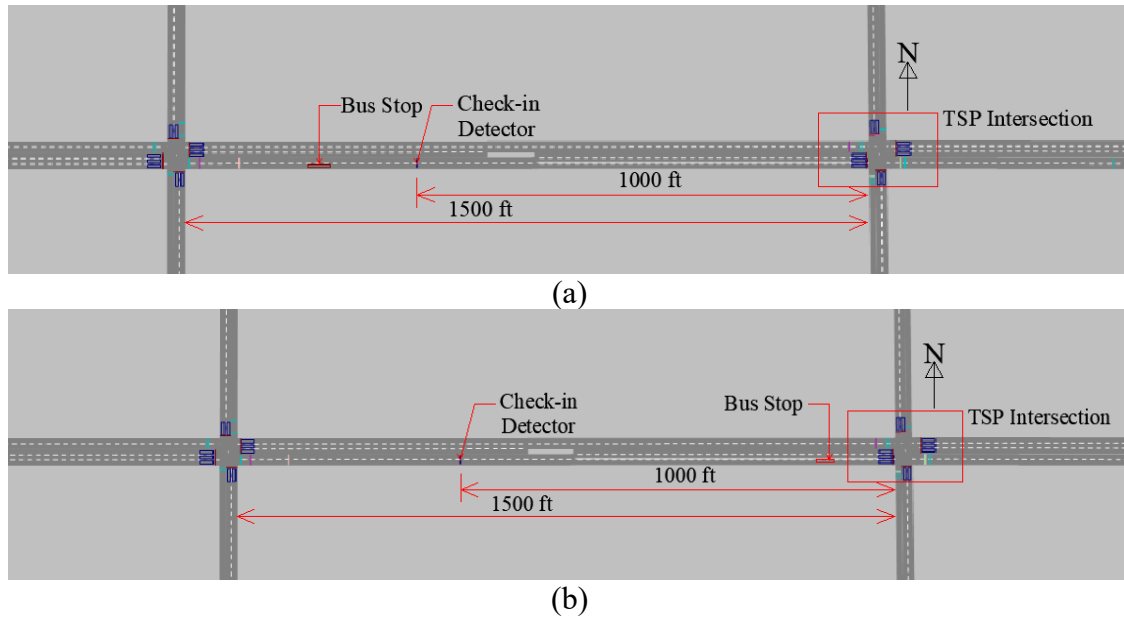


Figure 23: Network Model in PTV-VISSIM® for (a) far-side bus stop and (b) near-side bus stops

At both intersections, the main street has 1440 veh/h and 171 veh/h for through and left turns, respectively, while the cross street has 270 veh/h and 257 veh/h for through and left turns, respectively. For simplicity, no vehicles are assumed to turn right. Both intersections run on a coordination cycle length of 110 seconds with green splits of 16, 50, 22 and 22 seconds for phases (1,5), (2,6), (3,7) and (4,8), respectively. Yellow and all red durations, for all phases, are set to 4 and 1 second, respectively. Offsets are set to 22 and 0 seconds for the East and West intersections, respectively. TSP is only enabled on the East intersection with a check-in detector at 1000 ft upstream of the stop line and a check out detector immediately downstream of the stop line. The maximum Green Extension (GE) is set to 22 seconds, which is 20 % of the cycle length. Priority minimum greens are set to allow truncation of the side streets movements by up to 10 seconds, main street left turns by up to 5 seconds, and no truncation allowed for the main street through movements.

TSP operation is modeled using PTV-VISSIM®'s inbuilt ring barrier controller (RBC) which has been shown to closely emulate field traffic controllers (Ali et al. 2023; Roy 2023; Zlatkovic et al. 2010b). The TSP algorithm in the RBC works by providing GE or EG to the priority movement depending on the projected arrival time of the bus and the corresponding signal state. If the bus is projected to arrive at the stop line during the priority phase green, the algorithm does nothing and the bus passes through the intersection on normal green. If the projected arrival time of the bus is within the maximum allowable GE after the normal green, the priority phase is held in green until the bus goes through the intersection. If the bus is projected to arrive at the bus stop during red interval, the algorithm shortens the preceding conflicting phases to return early to the priority movement. The algorithm allows the user to select between a GE only strategy and a strategy that employs both GE and EG.

Transit bus operations are modeled with public transport (PT) line and PT line stop modules included in PTV-VISSIM®, with the capability to simulate bus routes, stops, dwell-time, passenger loads, and passenger movements. An eastbound bus route is modeled running through both intersections. To ensure that the impact of a bus has fully dissipated before introducing the next bus, only one bus enters the systems in each simulation hour with the bus input occurring after a 15-minute warm up period.

4.4.3 Experiment design

4.4.3.1 Dwell-time magnitude and variability at far-side bus stop

Under this task an experiment is designed to assess the impact of dwell-time magnitude and variability on TSP performance for a far-side bus stop. Three dwell-time distributions

shown in Figure 24 from field data are selected for the experiment. The distributions are for three stops on Route 39 SB in the morning peak. Dwell-time magnitude and variability, as well as the probability of stopping, increase moving from DT01 to DT02 to DT03. In PTV-VISSIM® the distributions are input directly as CDFs. For each dwell-time distribution, ten replicate runs are performed with TSP activated and ten replicate runs with TSP deactivated on the East intersection.

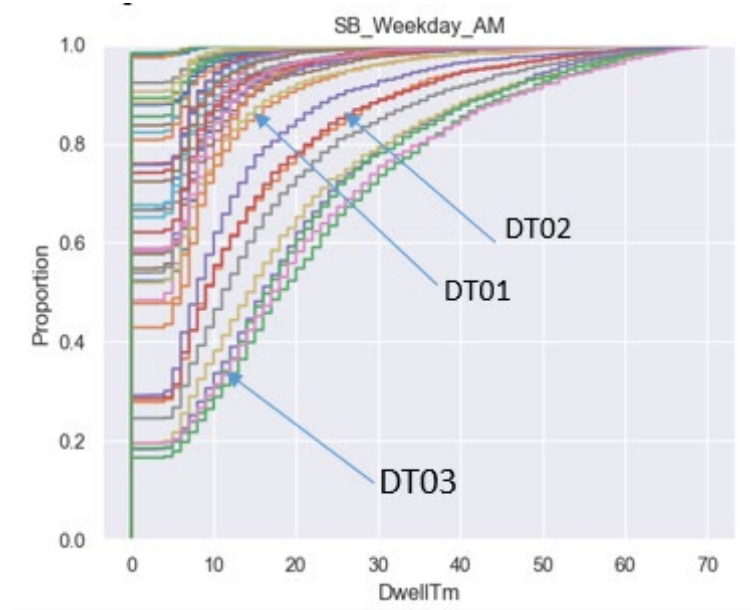


Figure 24: Dwell-time Distributions from field data

4.4.3.2 Impact of dwell-time variability at near-side bus stops

As discussed earlier, uncertainty in ETA is compounded by the variability of bus dwell-time at near-side bus stops, where a TSP request is sent to the controller before the bus passes the bus stop and without knowledge of whether the bus will stop or the duration of the stop. The challenges introduced by increased ETA uncertainty include: (1) providing GE but the bus dwells at the stop longer than expected and the provided extension goes to

waste, (2) providing EG when it is not required as the bus is delayed at the stop, and (3) not requesting TSP anticipating bus dwelling that does not happen. The first two are detrimental to both buses and side street traffic while the last affects only the bus.

With the near-side bus stop, ETA estimates include bus travel time at the prevailing speed, dwell-time at the bus stop, and bus deceleration and acceleration at bus stop. Preliminary runs are performed to estimate optimal ETA for each dwell-time distribution, which equates to estimating bus travel time from historical data. For the generated simulation data, distribution DT01 gave an average travel time of 45 seconds with a standard deviation of 12 seconds, distribution DT02 gave an average travel time of 49 seconds with a standard deviation of 13 seconds, while distribution DT03 gave an average travel time of 58 seconds with a standard deviation of 15 seconds. For each distribution, experiments were run with three ETA estimates including average (μ), average minus one standard deviation ($\mu-1\delta$), and average plus one standard deviation ($\mu+1\delta$). For all distributions, average ETA values gave the lowest bus travel times and thus the final experiments were run with average ETA values.

4.4.4 TSP performance analysis at far-side bus stop

4.4.4.1 TSP strategies and effectiveness

Figure 25 shows the number of buses arriving at the check-in detector at different points in the cycle with (a) plotting data for DT01, (b) for DT02, and (c) for DT03 (see Figure 24), i.e., increasing likelihood and length of stopping profiles. The red vertical line is drawn through the force-off point for the priority movement in which the bus is flowing. The force-off point is the point in the signal cycle where a given phase must be terminated

regardless of continued vehicle detection (Koonce 2008). Moving from DT01 through DT03, an increasing number of buses arrive at the detector after the force-off point (to the right of the red line) as they dwell longer at the bus stop. Thus, dwell-time magnitude at a far-side bus stop affects the arrival profile of the bus at the check-in detector.

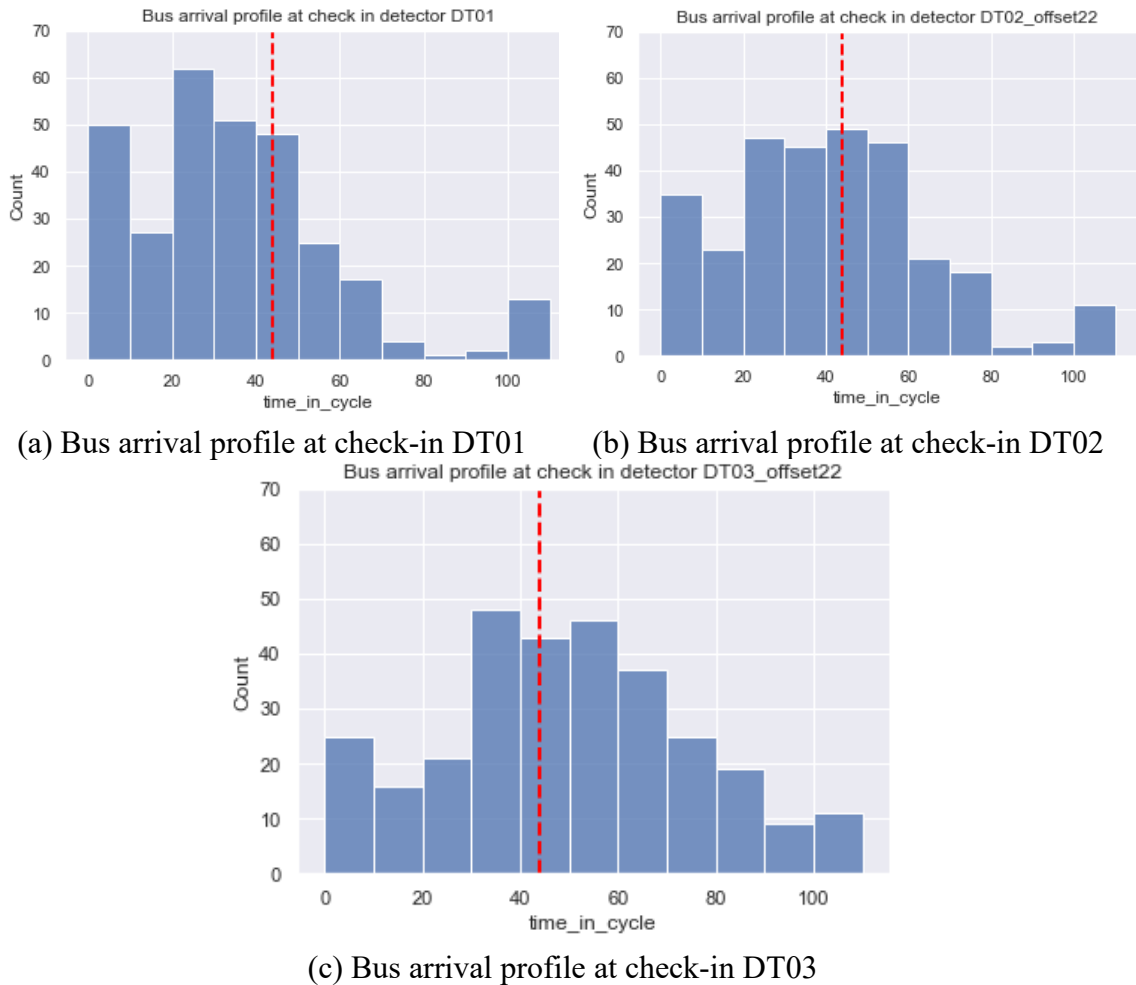


Figure 25: Bus arrival profile at check-in

Table 7 shows the number of buses receiving GE and EG for each dwell-time distribution and the effectiveness of the provided GE. The number of buses calling for EG increases progressively from DT01 to DT03 while the number requesting GE decreases. As shown in Chapter 3, GE provides more benefits to buses and is less detrimental to side street traffic

level of service (LOS) compared to EG. Therefore, it can be expected that TSP performance may be better for DT01 compared to DT02 and DT03. In the GE only strategy, buses reaching past the force-off point must wait for the programmed green. GE_Eff stands for GE effectiveness indicating the number of buses that are able to traverse the intersection on the provided GE. GE_NoEff indicates the number of buses that receive GE but are not able to traverse the intersection before the expiration of maximum GE. Comparing the number of effective and non-effective GEs for the three dwell-time distributions, it is seen that dwell-time magnitude and variability does not significantly affect GE effectiveness. The result is expected as the detection point occurs after the bus stop and thus dwell-time is not included in ETA estimates.

Table 7: TSP Effectiveness strategies for different dwell-time distributions

Strategy	DT	EG	GE	GE_Eff	GE_NoEff	No_Action
GE&EG	DT01	73	104	103	1	123
	DT02	110	92	91	1	98
	DT03	142	82	82	0	76
GE only	DT01	0	106	105	1	194
	DT02	0	92	91	1	208
	DT03	0	82	82	0	218

4.4.4.2 Bus travel time

Figure 26 shows the bus travel time for the three dwell-time distributions. Figure 26 (a) is for the GE and EG strategy while Figure 26 (b) is for the GE only strategy. Travel time is measured from the check-in detector to a point downstream the intersection. This range does not include the far-side bus stop; thus, the travel time does not include dwell-time and the associated deceleration and acceleration lost times. The plotted data includes all buses regardless of the presence of a TSP request. The red square in the plot represents the

average travel time. NoTSP represents a case of deactivated TSP. GEEG represents the TSP strategy with both GE and EG activated while GEonly represents the TSP strategy with only GE activated. The no TSP case shows very high variability in bus travel time especially for DT01 and DT02. The high travel time variability comes from the variability of dwell-time at the bus stop including zero dwell-time when the bus skips the stop. DT03 shows comparatively lower dwell-time as there is a lower probability of skipping the stop (zero dwell-time). Despite the high variability in the plots, it is seen that TSP is more beneficial for the lower dwell-time magnitudes and variability, such as that in DT01. The lower dwell-times allow more GE opportunities as more buses can arrive at the detector and request GE prior to the force-off point. For the GE and EG strategy, TSP reduces the travel time by 36.2%, 35.1%, and 32.9% for DT01, DT02 and DT03, respectively. Results of t-tests confirm that with TSP activated bus travel time reductions are statistically significant (p-value less than 0.05 for this effort) comparing DT01 to DT02 (p-value = 0.04) and comparing DT02 to DT03 (p-value = 0.004). Comparing DT01 and DT03, bus traveling time reductions are statistically significant with a p-value of almost 0.

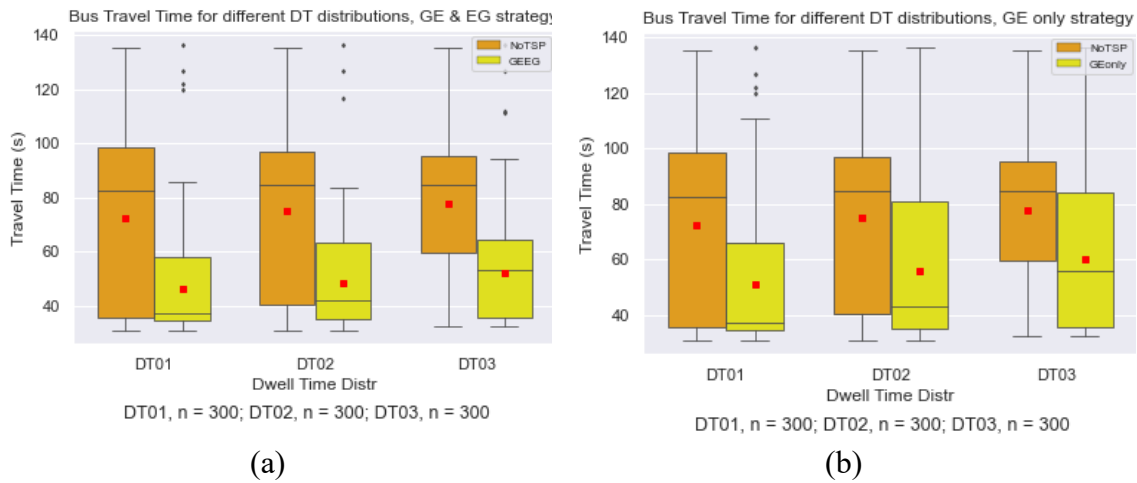


Figure 26: Bus travel time for different dwell-time distributions for (a) GE & EG and (b) GE only

Under the GE only strategy, TSP reduces the travel time by 29.2%, 25.1%, and 22.1% for DT01, DT02, and DT03, respectively. The decrease in TSP benefit from DT01 through DT03 is associated with the reduced number of GE opportunities as shown in Table 7. Results of t-tests confirm that the benefits of reduced bus travel time in GE only are significant from DT01 through DT03.

4.4.4.3 Cross street traffic delay

Impacts of TSP on general traffic are measured in terms of delay for the side street movements. Figure 27 shows the cross street southbound through (SBT) movement delay for different dwell-time distributions. For each bus receiving TSP, the average delay is estimated for vehicles on the cross street up to four cycles after check-in. It is seen that the delay increase caused by TSP increases as dwell-time increases (from DT01 to DT03). Results of t-test show that the delay difference between the TSP and no TSP cases is statistically significant (p-value = 0.0217) for DT02 compared to DT01, where DT02 has a greater difference. Comparing DT03 and DT01, the delay difference is again statistically significant (p-value of 0.0045) with DT03 having a greater difference. The delay differences are not statistically significant between DT03 and DT02 (p-value = 0.2189). The increased delay for the two higher distributions partly comes from a higher proportion of EG compared to GE for DT03 and DT02 relative to DT01.

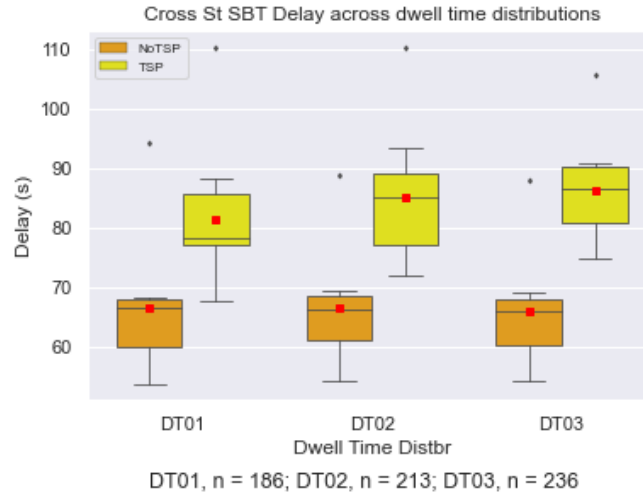


Figure 27: SBT Delay for different dwell-time distributions

4.4.4.4 Role of signal timing offsets

However, the above result is dependent on the coordination offset and indicates that for a given dwell-time distribution at the bus stop, signal timing offsets could be fine-tuned to possibly allow more bus progression without requesting priority, as well as provide more GE opportunities, and thus improve TSP performance. Figure 28 shows bus travel time resulting from adjusting offsets for (a) DT02 and (b) DT03. For both dwell-time distributions differences are seen between all offset levels for both with and without TSP, with difference being statistically different between the 22 and 40 second offsets. Without TSP, travel time improvement is attributed to more bus progression while with TSP, bus travel time improvement is attributed to increased GE/EG ratio as the offsets increase. Although, as the optimal offsets for buses may not be optimal for the non-transit vehicles this balance needs to be evaluated considering the traffic demand, bus headways, and bus and vehicle occupancies, among other factors.

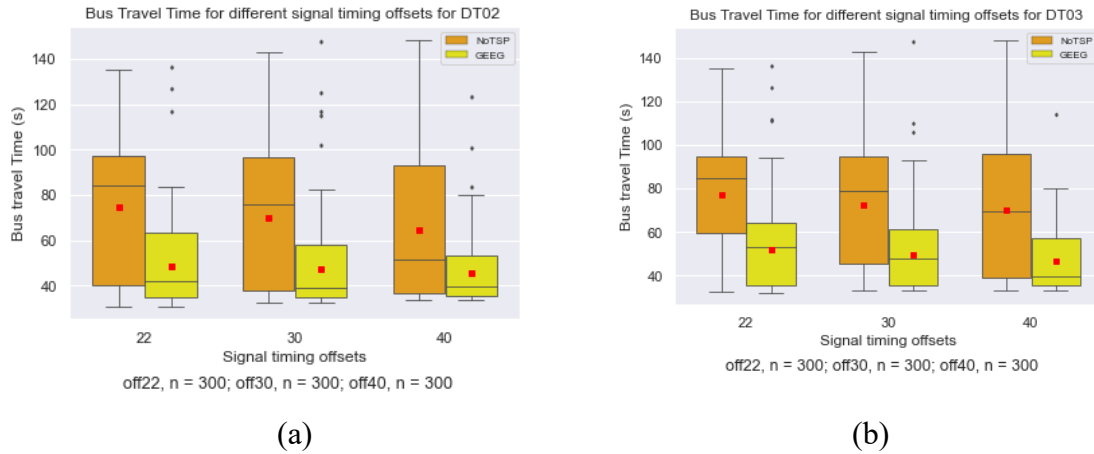


Figure 28: Bus Travel Time at different offsets for (a) DT02 and (b) DT03.

4.4.5 Comparing TSP performance for far-side bus stops and near-side bus stops.

4.4.5.1 TSP strategies and effectiveness

Table 8 summarizes the TSP strategies and effectiveness for far-side and near-side bus stops for the three dwell-time distributions. The most informative column is “GE_NoEff,” which represents the failed green extensions. The column “% GE_NoEff” represents the percentage of the failed GE compared to the total GE attempted. For the near-side bus stops, GE ineffectiveness increases as the magnitudes and variability of dwell-time increases from DT01 to DT03. Buses with a higher value of dwell-time than used in the estimate of ETA may be granted GE but fail to make it through the intersection. For both bus stop locations, the proportion of EG increases as dwell-time magnitude and variability increase. In addition to the change in arrival profile as discussed earlier, the increase in EG for near-side bus stops could also come from the selection of ETA that may limit the would-be successful GE.

Table 8: TSP strategies and effectiveness at far and near-side bus stops

Stop type	TSP Strategy	DT	EG	GE	GE Eff	GE NoEff	% GE NoEff	No Action
Far	GE & EG	DT01	69	117	113	4	3.4	114
		DT02	105	108	105	3	2.8	87
		DT03	145	91	87	4	4.4	64
	GE only	DT01	0	118	114	4	3.4	182
		DT02	0	108	105	3	2.8	192
		DT03	0	90	87	3	3.3	210
Near	GE & EG	DT01	51	145	105	40	27.6	104
		DT02	73	148	94	54	36.5	79
		DT03	133	121	72	49	40.5	46
	GE only	DT01	0	147	107	40	27.2	153
		DT02	0	150	95	55	36.7	150
		DT03	0	126	76	50	39.7	174

4.4.5.2 Bus travel time

Figure 29 presents a comparison of bus travel time for far-side and near-side TSP systems.

In both cases, travel time is measured for the same distance starting upstream of the far-side bus stop location. Thus, travel time includes dwell-time at the bus stop in both cases.

Figure 29 (a) plots buses receiving GE only while Figure 29 (b) plots buses receiving either GE or EG. For both plots, only buses receiving priority treatment (GE or EG) are included in the analysis. Comparing the TSP and No TSP for GE only of Figure 29 (a), it is seen that (1) TSP saves the bus more time for far-side bus stops compared to near-side bus stops, (2) the performance of TSP for the near-side is much more variable, and (3) for near-side bus stops, the performance of TSP in saving bus travel time deteriorates as the magnitude and variability of dwell-time increases. For the GE only strategy at a far-side bus stop, the average bus travel time reduces by 45.5%, 43.8%, and 45.1% for DT01, DT02, and DT03, respectively. For the GE only strategy at a near-side bus stop, average bus travel time reduces by 24.3%, 16.5%, and 12.3% for DT01, DT02, and DT03, respectively. The

deterioration in travel time savings at near-side bus stops comes from failed GE (see Table 8) that increases from DT01 to DT03. Considering the both GE and EG strategy shown in Figure 29 (b), the difference in TSP performance between near-side and far-side bus stops is more muted, especially for DT03. For far-side bus stop, the average bus travel time reduces by 35.1%, 30.5%, and 25.3% for DT01, DT02, and DT03, respectively. For near-side bus stop, the average bus travel time reduces by 25.2%, 20.6%, and 17.8% for DT01, DT02, and DT03, respectively. The more muted difference is because an increased number of EG (see Table 8) becomes more controlling for the higher dwell-time magnitudes and variability as discussed earlier. However, higher EG proportion is associated with more impacts on the side street as shown in the next section.

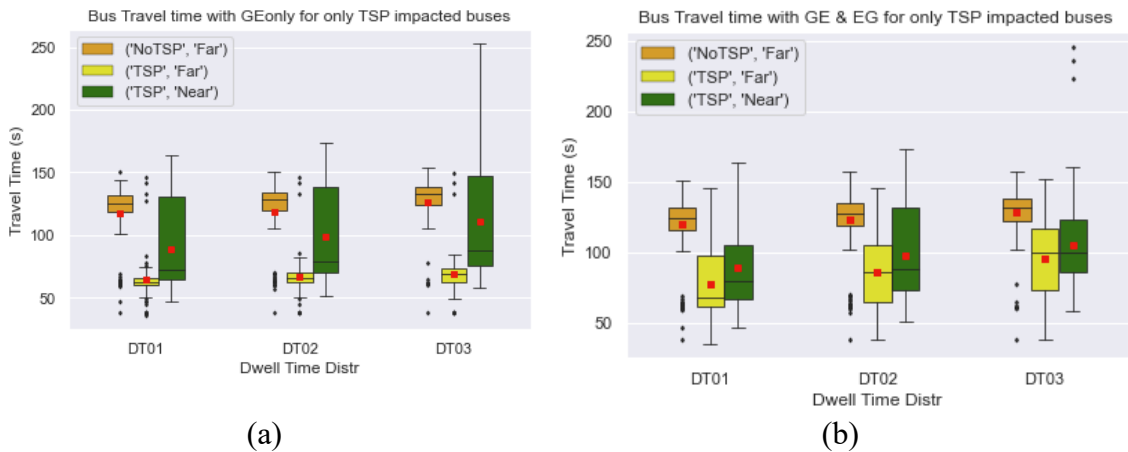


Figure 29: Travel time for TSP impacted buses at far and near-side stops under (a) GE only strategy and (b) GE and EG strategy.

4.4.5.3 Cross street traffic delay

Figure 30 shows a comparison of cross street Southbound left turn (SBLT) delay for near-side and far-side stops and the three dwell-time distributions. The plots show cross street passenger car delays in the four cycles after the bus check-in, whether the bus receives TSP

or not. It is noted that (1) delay caused by TSP is higher for near-side bus stops compared to far-side stops and (2) for near-side bus stops the delay caused by TSP increases with dwell-time magnitudes and variability. Considering the ineffectiveness of TSP at near-side bus stops in Table 8, green time is wasted at the expense of side street movements at a much higher rate than with far-side bus stops. Additionally, an increased proportion of EG compared to GE is associated with more impacts on side street traffic.

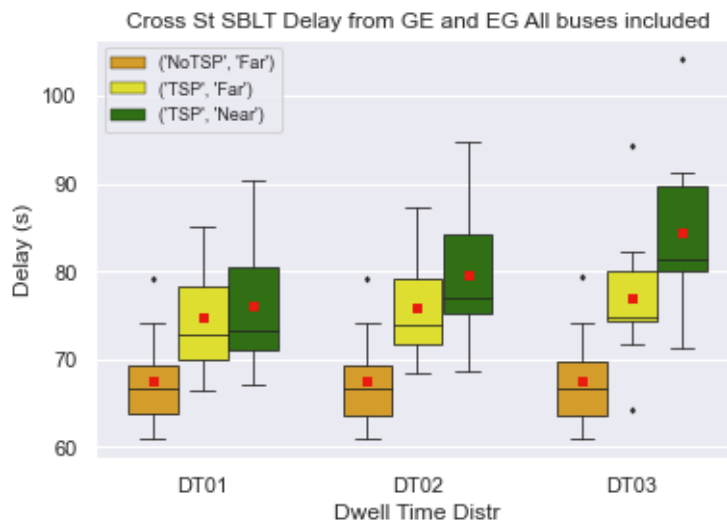


Figure 30: Cross street SBLT Delay, far-side vs Near-side bus stop

4.5 Dwell-time Summary

The first part of this study analyzed trends and distributions inherent in dwell-time data deduced from one-year of APC data for one of the busiest bus routes in Atlanta. Dwell-time varied from stop to stop and for each stop by time of day. Stops with high magnitude dwell-time also showed high dwell-time variance and a smaller probability of a bus skipping the stop. For several stops dwell-time distributions showed directional peaks during the peak hours of travel. For the southbound direction, most stops showed an AM

peak (600-900hours) and an afternoon peak (1300-1500 hours) while for northbound direction, most stops showed a more spread-out peak starting from 1100 to 1900 hours. Dwell-time data was closely fitted by inverse Gaussian, log normal, power lognormal, Fisk (log-logistic), and Johnson's SU distributions.

The second part of the study used a simulation environment to evaluate the impact of dwell-time magnitude and variability on TSP performance. Sensitivity experiments were performed with three field dwell-time distributions at both far-side and near-side bus stops. For the far-side bus stop, dwell-time significantly impacted the bus arrival profile at the check-in detector and thus the TSP strategy (GE, EG, No action) selected. Higher dwell-time magnitudes and variability led to a higher share of EG, which is not as effective as GE. Setting offsets considering the prevailing dwell-time distribution may help improve TSP performance. At near-side bus stops, dwell-time variability introduced more uncertainty in ETA estimation and significantly reduced TSP effectiveness, especially GE. TSP performance in terms of GE success, bus travel time and side street traffic delay was significantly better at far-side stops compared to near-side stops.

The current study considered a conventional TSP system which is still the most widely implemented system in the field. Some adaptive TSP systems integrate modules to estimate dwell-time in real-time and have capabilities to update TSP requests according to the realized dwell-time. Follow-up studies could consider the impact of dwell-time variability for adaptive TSP systems. There is still need to further research TSP strategies for near-side bus stops which includes investigating the effect of the distance between the bus stop and the TSP check-in point on the effectiveness of different strategies for near-side stops. Additionally, the current study uses APC data from August 2021 to August 2022. In this

period transit ridership had not reached full recovery from the impacts of Covid-19. Follow up studies could consider using more recent data for comparison purposes or for a longitudinal analysis.

CHAPTER 5 SINGLE INTERSECTION TSP BASED ON REINFORCEMENT LEARNING

5.1 Background

As discussed in Chapter 2, a key theme of TSP research is to develop optimization strategies to improve bus performance while limiting impacts to the general traffic. Using traffic simulation environments, several studies have tested adaptive TSP algorithms that integrate CV data (Beak et al. 2018; Cvijovic et al. 2022; Hu et al. 2015; Hu et al. 2014; Hu et al. 2016; Lee et al. 2017; Mohammadi et al. 2020; Teng et al. 2019; Wang et al. 2020; Wu and Guler 2019; Yang et al. 2019; Zeng et al. 2015). Data from CVs enables the formulation of more precise actuation strategies and better evaluation of traffic states.

Previous adaptive TSP algorithms have mainly been based on mathematical programming (Hu et al. 2015; Kim and Rilett 2005; Li et al. 2011; Ma et al. 2013; Truong et al. 2019; Zeng et al. 2015; Zhai et al. 2023). However, these studies' complex and nonlinear objective functions require high computational resources to solve in real time. Recent efforts are testing machine learning approaches, especially reinforcement learning (RL) (Cheng et al. 2022; Hu et al. 2023; Long et al. 2022; Shen et al. 2023; Yang and Fan 2024; Zhong et al. 2023). When trained, model free RL algorithms with no need to project future traffic states makes them more computationally efficient to implement in real time. On-going RL-based TSP efforts build on equally recent and still evolving RL-based adaptive traffic control algorithms (Aslani et al. 2019; Bálint et al. 2022; Bouktif et al. 2023; Casas

2017; Lee et al. 2022; Li et al. 2020; Li et al. 2016; Li et al. 2021; Liu et al. 2022; Shabestary et al. 2020).

As described in Chapter 2, TSP systems in use have an event-triggered algorithm that comes online when a priority request is sent to the controller. The controller switches from normal mode of operation to the priority mode. The few previous studies that integrate TSP into RL-based adaptive traffic control do so by modifying state and reward functions to include transit flow parameters. At intersection level therefore, one RL agent is trained to take second by second level decisions to adaptively implement general traffic control and TSP. However, as stated by Shabestary et al. (2020), the industry may prefer “controllers that manipulate signals less frequently”. It may be desirable to only manipulate the signals at second-by-second level during priority service and maintain actuated signal control (ASC) in absence of the bus. Additionally, it may be challenging to train one agent to optimally control both general traffic and implement TSP as bus arrivals at intersections are sparse occurrences compared to general vehicles. Training same RL agent to learn general traffic control and bus prioritization may take prohibitively longer periods, especially in higher fidelity traffic microscopic simulation environments.

The TSP algorithm developed in this chapter is event-based; that is, the algorithm is only triggered when a bus reaches within the defined connected vehicle (CV) communication range of the intersection. The intersection runs on a background adaptive signal controller which is also based on RL and CV. The background controller logic is first formulated and trained. Thus, two RL agents are formulated, trained, and tested. The first agent controls the intersection under general traffic conditions, i.e., in the absence of a bus while the second agent is event based, triggered to assume intersection signal control when a bus

enters the CV communication range of the intersection. Later in **Chapter 8**, the background ASC is performed by a MaxTime® emulator forming VISSIM®-MaxTime® SILs. Actions/signal timings from event triggered TSP agents are implemented in MaxTime® with National Transportation Communications for Intelligent Transportation Systems (NTCIP) commands.

In testing the TSP algorithm in this chapter and in the rest of the dissertation, all buses are allowed to trigger the TSP agent; however, logical conditions can easily be added to limit TSP service to only buses meeting certain criteria, for example buses behind schedule or above a set occupancy. As indicated, a key challenge of training RL agents in a high-fidelity microscopic simulation environment like VISSIM® is high runtime efficiency. Decisions on the architecture of the algorithms in this chapter and the rest of the dissertation are taken considering the need to improve the run time efficiency.

5.2 Methodology

5.2.1 Overview

This section describes the formulation of the RL agents including the adapted simulation environment, the agent structure, selected hyper parameters, training, and testing. In this study, two DDQN agents are formulated, the general traffic control agent (DDQN-SC) and the TSP agent (DDQN-TSP). DDQN-SC is first formulated and trained and then used as the background controller in the training of the TSP agent (DDQN-TSP). In the following sections the formulation of DDQN-SC agent is first presented before moving to DDQN-TSP.

5.2.2 Reinforcement learning overview

RL is an approach of ML in which the learner, also called the agent, learns by continuous interaction with the environment. RL seeks to find the optimal mapping of states to actions, to achieve the highest numerical rewards. By trying out different actions over time the agent learns to identify the actions that maximize rewards given the state of the environment. The first chapter of Sutton and Barto (2018) provides an insightful introduction of RL. Learning by constant interaction with the environment distinguishes RL from supervised learning and unsupervised learning, which are other common approaches of ML. In supervised learning the learner is shown different situations with the corresponding correct behavior/labels. The objective is for the system to learn how to generalize its responses to situations outside the training dataset. Unsupervised learning involves identification of hidden structures in unlabeled data.

Figure 31 shows the broad elements of RL. Albrecht et al. (2024) defines the environment as “a physical or virtual world whose state evolves over time and is influenced by the actions of the agents that exist within the environment”. An agent is the learning entity that takes state as the input and uses the policy to select actions to take. A policy is the algorithm used by the agent to map observed states to actions. In q-learning, this algorithm can be a table with states and corresponding actions or a simple function. In deep RL, the policy is a neural network. A reward is a numerical quantity used to evaluate the policy at every time step. At time step t , the agent observes the environment and obtains the state (s_t) which is then fed to the policy to select action (a_t). Action (a_t) is applied to the environment to obtain a new state (s_{t+1}) and reward (r_{t+1}) at the next time step ($t+1$).

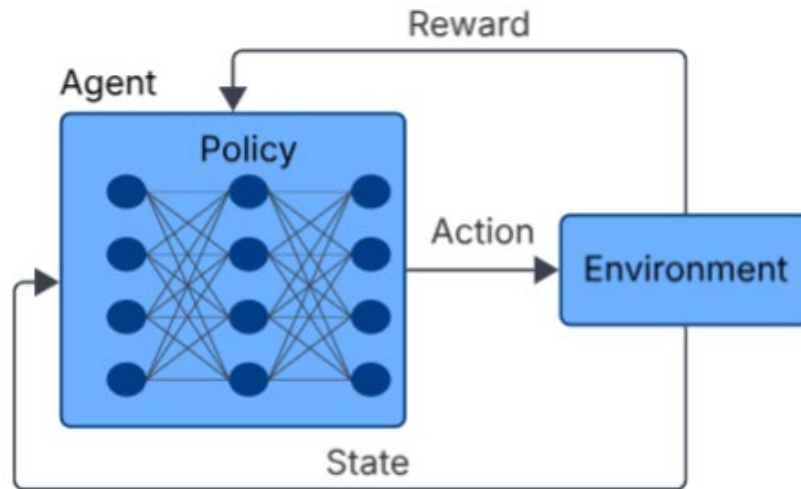


Figure 31: Elements of RL

RL learning problems are normally expressed mathematically as Markov decision processes (MDP) first formulated by Richard Bellman in 1950s. The next state of the system only depends on the current state and the action taken which also implies independence of past rewards. Rewards evaluate the policy in an immediate sense, but the algorithm needs to look past immediate maximum rewards or immediate good actions. Actions need to be evaluated considering future states and rewards. The need to consider future rewards gives rise to the concept of discounted rewards shown in Equation 1. Returns (R) represent the sum of all discounted rewards starting from state at time t to the end of episode after a time T . Future rewards are discounted to their equivalent values using a discounting factor γ in the range $[0,1]$ and represents the weight attached to future rewards compared to immediate rewards.

$$R_t = r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \dots + \gamma^T r_{t+T} = r_t + \gamma R_{t+1} \quad (1)$$

The need to consider discounted returns gives rise to the concept of value functions which is central to much of RL theory and most RL algorithms (Albrecht et al. 2024). Value

functions are used to define the expected returns of individual states and actions. State value function, $V^\pi(s)$ gives the expected returns starting from state (s) and following the policy (π) thereafter. Action value function $q^\pi(s,a)$ gives the expected returns starting from state (s), taking action (a), and following the policy there after. The objective of the agent is to maximize total reward in the long run as approximated by value functions.

Another element of RL is the model of the environment. The role of the model is to mimic the responses in the environment by, for example, predicting the next state and reward given the current state and action. Under this criterion, RL approaches can be split between model based and model free approaches. In model-based approaches, mathematical models are used to project forward to predict new states and rewards given the current state and action. The state of art RL algorithms are model free in a sense that there is no need to project forward but instead learn the best policies by trial and error.

Based on policy definition, RL approaches fall under value-based methods and policy-based methods. Value-based methods learn value functions and to select actions, the agent follows a policy derived from the learnt value function. In deep RL, value-based methods like DQN learn parameterized value-functions using neural networks. Policy-based based methods directly learn parametrized policy functions. Policy-based methods can easily learn stochastic policies which comes with several advantages including (1) better performance under partially observable environments, (2) better convergence properties as the action probabilities change smoothly as a function of learned parameters because policies are parameterized with continuous values (Morales 2020). Policy gradient (PG) algorithms are policy-based algorithms in which gradient-based optimization is used to optimize the policy parameters. Q-learning and DQN are the most popular value-based RL

methods with DQN being the first deep RL method (Mnih et al. 2015). REINFORCE, actor critic and its variations including advantage actor critic (A2C), PPO, trust region policy optimization (TRPO), deep deterministic policy gradient (DDPG) etc are some of the policy-based RL methods.

5.2.2.1 Deep Q-Network overview

Q-learning and DQN are the most popular value-based RL based algorithms. DQN formulated by Mnih et al. (2015) extends the tabular q-learning to estimate the value function with the neural network. On the forward pass, the neural network takes state, s as input and outputs action value estimates corresponding to each discrete action, a . Actions to take are derived from the action value estimates normally by using an ϵ -greedy algorithm. The objective of the training is to optimize the neural network parameters/weights to minimize the loss, $L(\theta)$ which is formulated as the mean squared error (MSE) between estimated and target action values as shown in Equation 2 where θ are the network parameters estimating the action value function Q , t is the current time step and y^t is the bootstrapped target values.

$$\mathcal{L}(\theta) = (y^t - Q(s^t, a^t; \theta))^2$$

$$y^t = \begin{cases} r^t & \text{if } s^{t+1} \text{ is terminal} \\ r^t + \gamma \max_{a'} Q(s^{t+1}, a'; \theta) & \text{otherwise} \end{cases} \quad (2)$$

The computed loss is backpropagated to compute gradients to optimize the parameters using gradient descent.

The ϵ -greedy algorithm selects the action corresponding to the highest q-value. DQN algorithm needs to strike a balance between exploration and exploitation. Exploration involves the agent trying out actions it has not tried before while exploitation involves the agent selecting the actions that have already shown better rewards. Exploration helps the agent to exhaust all actions and avoid getting stuck in local minimums while exploitation helps the agent leverage what is already learned. Exploration and exploitation tradeoff is implemented by setting an exploration probability close to one and gradually reducing it as the agent gains improved knowledge of the environment allowing it to exploit better rewards.

At every time step, observation and action histories called experiences are stored in the memory and later used to train the model. The stored experiences (e_t) at every time step consist of tuple of current state (s_t), action (a_t), next state (s_{t+1}), and reward (r_{t+1}), as shown in Equation 3. The stored experiences over time steps constitute what is termed as a memory buffer. After a given number of steps, the model is trained on batches sampled from the memory buffer.

$$e_t = (s_t, a_t, s_{t+1}, r_{t+1}) \quad (3)$$

The base DQN formulation suffers from instability issues resulting from a combination of (a) function approximation, (2) bootstrapped targets and (3) off policy learning. (Sutton and Barto 2018) famously dubbed this combination a “deadly triad”. The “deadly triad” is associated with extreme instability and divergence of value estimates. One of the proposed remedies is to use a separate/an additional neural network to estimate the bootstrapped target values with parameters θ' shown in Equation 4.

$$y^t = \begin{cases} r^t & \text{if } s^{t+1} \text{ is terminal} \\ r^t + \gamma \max_{a'} Q(s^{t+1}, a'; \theta') & \text{otherwise} \end{cases} \quad (4)$$

(Van Hasselt et al. 2016) extends the concept of two separate neural networks to form DDQN. In DDQN, the target value is computed by the main network and then the target network is used to evaluate the target value as shown in Equation 5 with all the terms defined before.

$$y^t = \begin{cases} r^t & \text{if } s^{t+1} \text{ is terminal} \\ r^t + \gamma Q(s^{t+1}, \operatorname{argmax}_{a'} Q(s^{t+1}, a'; \theta); \theta') & \text{otherwise} \end{cases} \quad (5)$$

Another stabilizing technique of DQN is breaking correlations in the training samples. ML methods assume that the training data samples are independent and identically distributed (i.i.d). This is not true for consecutive samples from the same episode in RL. To break this correlation, data is uniformly sampled in mini batches from memory and used for training. Sampling data in batches also allows performing multiple training iterations with the same data which improves data use efficiency (Albrecht et al. 2024). The loss function is computed over several batches each with size B.

Several other enhancements to DQN have been proposed aimed at improving stability, convergence and efficiency. These include DQN with prioritized experience replay (PER), and dueling DQN. As opposed to uniform sampling of experiences in the standard DQN, PER prioritizes samples from their “usefulness” in the learning process. Samples that are likely to aid faster learning are given higher priority and sampled more frequently. The prioritization of samples is mainly based on the temporary difference error associated with each experience (Schaul et al. 2015). Dueling DQN introduced by Wang et al. (2016)

performs separate estimation of state value function and state dependent action advantage. These different DQN variations can be combined into one algorithm as demonstrated by DeepMind researchers in Hessel et al. (2018).

5.2.3 RL agent formulation and Architecture

Figure 32 provides details of the interaction between the agent and VISSIM® simulation environment. The set up consists of a VISSIM® network model and three python scripts including the agent script, the traffic state script, and the agent-VISSIM® interface script.

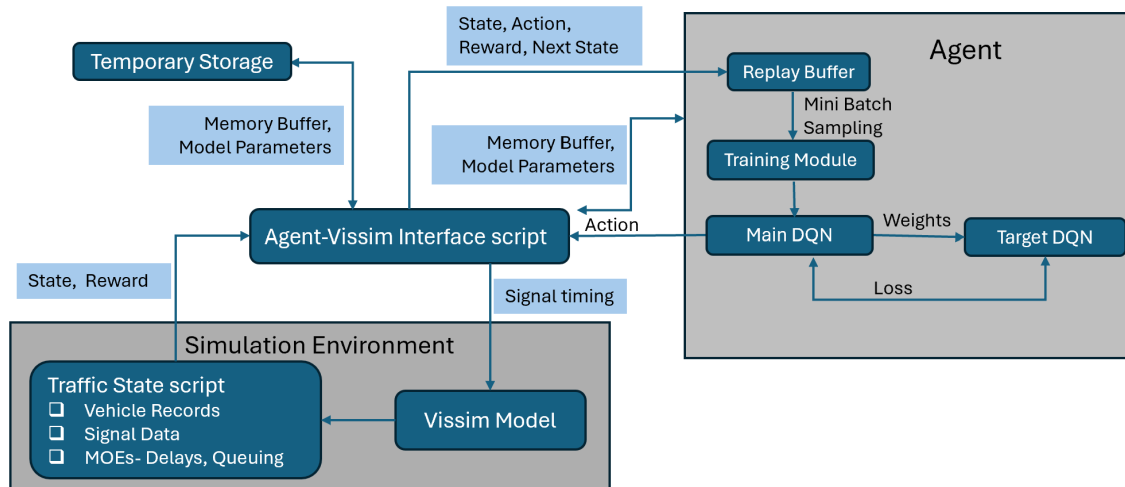


Figure 32: Traffic simulation and RL modules.

5.2.3.1 VISSIM® Simulation model

PTV’s VISSIM®, a widely used microscopic simulation platform is selected as the simulation environment. The network model is the same as in Figure 3 but with no check-in and checkout detectors and no upstream bus stop. The base network runs on an actuated signal control in free mode (no fixed cycle). All approaches have stop bar detectors for fully actuated signal control. The network has a four-lane E-W major street and a two-lane

minor street (N-S). All left turns on the major and minor streets have exclusive turn lanes and protected only signal phases. For simplicity, right turning movements are omitted from both streets. The model consists of a bus route on the main street (E-W), with a far-side bus stop immediately downstream of the intersection. For model training, traffic volumes are selected to achieve a volume to capacity ratio (v/c) of approximately 0.95 when running fixed signal timings. The traffic volumes are as described for $v/c = 0.95$ in Figure 4

At each time step, the agent provides new signal settings to the PTV VISSIM® simulation, and the new state and reward value are computed with data extracted from the simulation with the Traffic state python script. The extracted data includes vehicle records, signal data, and performance measures (i.e., vehicle delays and queuing). The traffic state script processes the extracted data and outputs the new state and reward.

PTV VISSIM® provides a COM interface, an application programming interface (API), that enables interaction with the simulation in run time. However, running the simulation through COM can significantly increase the run time, especially if there is continuous interaction to exchange data with the simulation. As shall be seen in the results section, hundreds of episodes were required for each agent to learn the optimal policy. Thus, run time efficiency becomes of great essence. This study adopts event-based scripts, a less widespread alternative to COM available in PTV VISSIM®. Event based scripting in VISSIM® involves embedding a script in the simulation and specifying functions to execute at given simulation time steps or when a stated event occurs (see “RunType” in Figure 33). Figure 33 shows an event-based script embedded in VISSIM® with four functions (FuncName) to execute at different times during the simulation. In this study using event-based scripts was significantly faster than using COM.

Count: 4	No	Name	RunType	Period	FromTime	ToTime	FuncName	Scope	ScriptFile
1	2		At time step start	1	0.00	MAX	Vis_Run	Simulation run	try_steps.py
2	3		Before simulation end	1	0.00	MAX	RL_training	Simulation run	try_steps.py
3	4		Before simulation end	1	0.00	MAX	save_output	Simulation run	try_steps.py
4	5		Before simulation st...	1	0.00	MAX	retrieve_out	Simulation run	try_steps.py

Figure 33; Event Based Script Embedded in VISSIM®

However, a drawback of event-based scripts is that VISSIM® data is not retained in memory at run completion. Thus, using event-based scripts instead of COM required developing code to bridge model data. For DQN this data includes 1) the training data (memory buffer) described earlier, 2) learned model parameters, that is, weights for both main and target network models, and 3) the exploration probability as it progressively decays. For PPO only learnt model parameters need to be saved. After each episode/simulation run the data is stored temporarily in a database and re-loaded at the start of the next simulation run. See the temporary storage module and its linkages in Figure 32.

5.2.3.2 Agent script

RL agent coded in a python script consists of the main DQN and target DQN, training module, and the memory buffer. Actions are selected using an epsilon greedy algorithm as discussed in section 5.2.2.1. During exploration, a random action is selected from a set of valid actions while during exploitation, the action is selected by making a forward pass through the main DQN using the current state as the input. At the end of each episode, the main DQN is trained by taking random samples from the replay memory. The training module includes the logic of sampling from the memory buffer, computing loss and

gradients, and backpropagation to update the weights and biases. The target DQN is updated after a given number of episodes by copying weights from the main DQN. The agent pseudo code is shown in algorithm 1.

Algorithm 1: DDQN -SC	
Input: learning rate lr, reward discount rate γ , memory buffer capacity, minibatch size, simulation time, T, number of episodes N, target update episodes N_{update}	
for episode = 1, N do	
	Re-load main DQN, target DQN, memory buffer, exploration probability ϵ
	Initialize decision point, time step Δt , current state S_t , next state S_{t+1}
	for t=1, T do
	Run one simulation step
	// select actions
	if t = decision point then
	Choose action $a^t = \begin{cases} \text{argmax } Q * (S^t), & \text{Uniform}(0,1) > \epsilon \\ \text{random } a^t \in A^t, & \text{otherwise} \end{cases}$
	Compute Δt
	end if
	// set new signal timings in VISSIM®
	if decision point $\leq t < \text{decision point} + \Delta t$ then
	Execute action a_t
	end if
	if t = decision point + Δt then
	Set decision point = decision point + Δt
	Obtain next state S^{t+1} , reward r^{t+1}
	store transition $(S^t, a^t, S^{t+1}, r^{t+1})$
	Set current state $S^t = \text{next state } S^{t+1}$
	end if
	end for
	// train the models
	If number of stored transitions > batch size then
	sample random minibatch of transitions $(s_i, a_i, s_{i+1}, r_{i+1})$
	compute target, $y^t = \begin{cases} r^t & \text{if } s^{t+1} \text{ is terminal} \\ r^t + \gamma Q(s^{t+1}, \text{argmax}_{a'} Q(s^{t+1}, a'; \theta); \theta') & \text{otherwise} \end{cases}$
	compute loss, $\mathcal{L}(\theta) = (y^t - Q(s^t, a^t; \theta))^2$
	compute gradients and perform gradient descent to update main network weights
	if number of episodes % $N_{update} = 0$ then
	Update target network

	end if
	end if
	end for

5.2.3.3 Agent – VISSIM® Interface script

The agent-VISSIM® interface python script is the central control script that facilitates the exchange of information between the agent, VISSIM® model, and temporary storage database. The script obtains the current state from the traffic state script, sends the state to the agent, and receives the selected actions. The script converts the actions received into signal timings which it then sends to PTV VISSIM®’s signal displays. At the end of the time step, the script sends the current state, action, reward, and next state to the memory buffer. As already mentioned, event-based scripts do not hold data across simulation runs. At the end of each episode/simulation run, the agent-VISSIM® interface script sends the memory buffer, learned model parameters, and current exploration probability from the agent to the temporary storage database. The script retrieves these data from the temporary database at the beginning of the next episode and loads them into agent.

5.2.3.4 State Definition for DDQN-SC

The study assumes availability of CV data which enables more detailed definition of the traffic state. DDQN-SC state is defined by two vectors: (1) a vector of size 10, populated with the number of vehicles in each of 10 approach lanes, and (2) vector of size 4, populated with the green duration for each signal phase. For the vehicle state vector (Veh_{state}), the study assumes knowledge of the number of vehicles within 800 ft of the stop line for all inbound movements. The 800ft is taken as the CV communication range for the intersection. An alternative to CV data could be video camera data at the intersection. The

main requirement is for the camera to be able to see the back of queue. In Equation 6, v_{mn} represents the number of vehicles in lane (n) of approach (m). The signal state (Sig_{state}) defined by green duration refers to how long the current phase has been green and can be derived from Signal Phasing and Timing (SPaT) data which is increasingly becoming more ubiquitous. As shown in Equation 7, the active green phase is populated with the green duration (22 seconds in this example) while the other phases are assigned 0.

$$Veh_{state} = \begin{bmatrix} v_{11} \\ v_{12} \\ \cdot \\ \cdot \\ v_{mn} \end{bmatrix} \quad (6)$$

$$Sig_{state} = \begin{bmatrix} 0 \\ 0 \\ 22 \\ 0 \end{bmatrix} \quad (7)$$

5.2.3.5 Action Definition for DDQN-SC

Action in this chapter is defined as the selection of the next phase to assign green from a discrete set of the available signal phases. For simplicity, a single ring barrier sequence is considered as shown in Figure 34. Adapting single ring configuration reduces the maximum number of phases from 8 in the two-ring barrier to 4 phases. This simplification shall be relaxed in **Chapter 8** when the formulated algorithms are tested on a real-world corridor with actual field implemented signal plans.

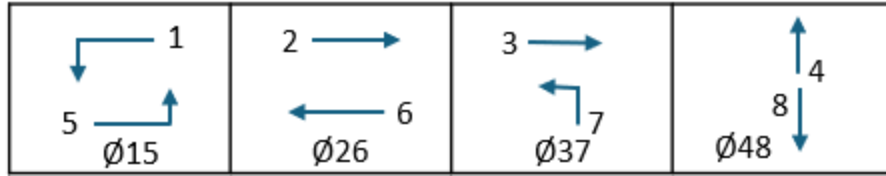
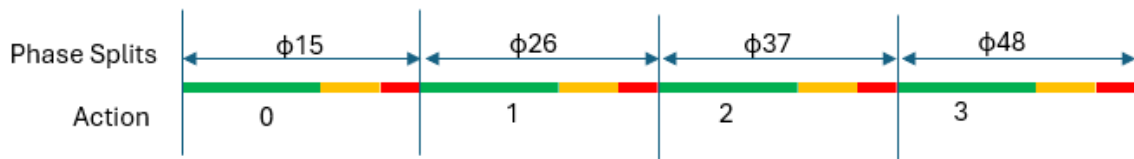
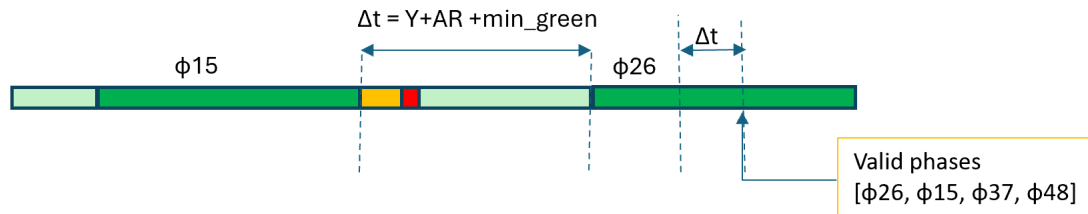


Figure 34: Single Ring Phase Sequence

Four phases are defined, North/South Left (ϕ_{37}), North/South Through (ϕ_{48}), East/West Left (ϕ_{15}), and East/West Through (ϕ_{26}). This represents the left turns and through movements both on the main and cross street resulting in a discrete set of 4 actions (0,1,2,3) as shown in Figure 35 (a).



(a) Available phases and actions



(b) Action implementation

Figure 35: Action selection and Implementation

At every decision point, the agent selects which of the four phases to assign green. If the next phase is different from the current phase, yellow and red clearance intervals are displayed before moving to the next selected phases. All yellow intervals are taken as four seconds while red clearance is taken as one second. At the start of a phase, minimum green will be served. From the onset of yellow to the end of minimum green, the agent does not take any action as shown in Figure 35 (b). If the next phase selected is the same as the

current phase, green time is extended by Δt , the base time step. The base time step is taken as 3 seconds. The algorithm time step therefore varies depending on the current and the next selected action. See Figure 35 (b).

5.2.3.6 Reward definition for DDQN-SC

Base reward is defined as negative average delay for all vehicles. Negative of the delay is used in the reward function as delay is a disutility metric and the algorithm needs to move in the direction of positive or less negative rewards. As shown in Equation 8, average delay is obtained by summing delay for each vehicle (d_i) and dividing by total number of vehicles, n . Two negative penalty terms are also imposed on the local reward. The first penalty N is imposed when queue length on any of the side street movements (ql_j) exceeds a set threshold (ql_{Th1}). The penalty is aimed at preventing excessive queuing on the side streets especially in the early training stages. The second penalty M is imposed when the phase changes from ϕ_t to ϕ_{t+1} before the end of maximum green when the queue length on the current phase ql_{ϕ_t} exceeds a set threshold ql_{Th2} . This penalty is aimed at reducing instability in earlier training stages. Training the agent with and without these penalties showed that the final model weights do not change significantly but with the penalties the models converge significantly faster.

$$reward\ r = -\frac{\sum_1^n d_i}{n} - N - M \quad (8)$$

$$N = \begin{cases} 0, & ql_j \leq ql_{Th1} \\ -200, & ql_j > ql_{Th1} \end{cases}$$

$$M = \begin{cases} 0, & \phi_{t+1} = \phi_t \\ -200, & \phi_{t+1} \neq \phi_t, ql_{\phi_t} > ql_{Th2} \end{cases}$$

5.2.3.7 Hyperparameter selection

Preliminary training was performed to select model hyperparameters. The final selected hyperparameters similar for both DDQN-SC are shown in Table 9. For brevity the process and results of hyperparameter selection is not included in this dissertation.

Table 9: DDQN-SC selected hyperparameters

Hyperparameter	Value/s
Learning rate, lr	0.001
Discount rate, gamma (γ)	0.99
Exploration probability decay rate	0.01
Memory buffer capacity	2000
Hidden layer neurons	64,128
Target network update frequency episodes	10

DDQN-TSP formulation and architecture

DDQN-TSP is formulated as an event triggered algorithm that comes online when a bus enters the DSRC zone for the intersection. Recall, DDQN-SC works as the adaptive traffic controller that runs the intersection in absence of the bus. Figure 36 shows the conceptual architecture and interaction of the two agents with VISSIM® during the training of DDQN-TSP. During training of DDQN-TSP, and in the implementation of the two trained algorithms, DDQN-SC controls the intersection until a bus requesting TSP enters the DSRC zone of the intersection. At this time, control then shifts to DDQN-TSP until the bus checks out. At the point of switching, the agent coming online takes the last state of the agent going offline. Over several episodes the agents learn to take actions that allow smooth transitions at these points. The pseudocode for the training of DDQN-TSP including the background DDQN-SC is shown in algorithm 2. The neural network for

DDQN-TSP has the same architecture as the neural network for DDQN-TSP and the same hyperparameters shown in Table 9 are utilized.

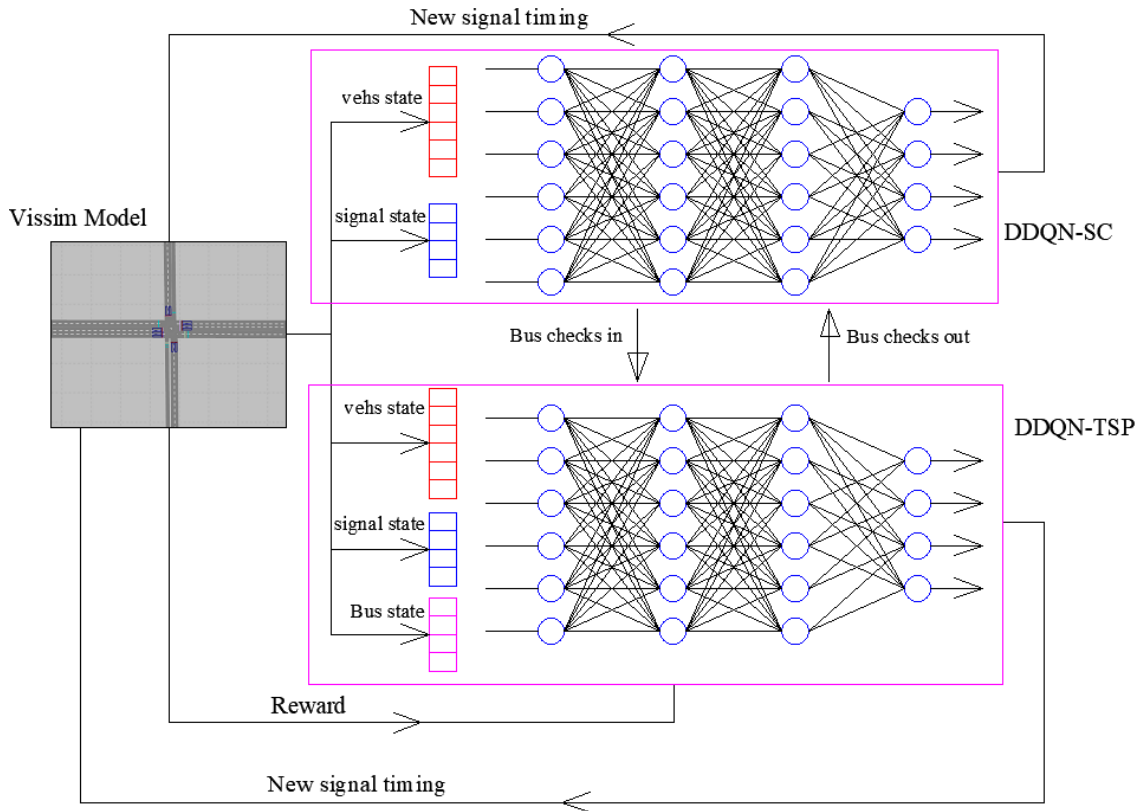


Figure 36: RL agent architecture during the training of DDQN-TSP.

5.2.3.8 State Definition for DDQN-TSP

The state definition for general traffic is expanded to include bus flow parameters as shown in Figure 36. In addition to the two vectors defined for DDQN-SC, two additional vectors are added for transit bus position and speed on the approach. It is assumed that all buses are connected, broadcasting their location and speed utilizing basic safety messages (BSM) as they approach the intersection. The intersection CV communication range range is taken as 800 ft and thus the bus only starts to communicate with the signal controller at 800 ft from the intersection. The 800 ft of the approach link is divided into 32 cells, each of 25

feet. Two vectors of length 32 with each entry representing a cell are created to represent bus position and speed as shown in Equation 9 and Equation 10, respectively. When the bus occupies a cell, the corresponding entry in the location matrix is populated with 1 and the with bus speed in the speed vector, otherwise both entries are populated with zero. Equations 7 and 8 show an example of the bus in the 4th cell from the beginning of the link traveling towards the stop line with a speed of 36 mph. The state definition with cell representation allows accommodation of different numbers of buses approaching the intersection.

$$Bus_{pos} = [0, 0, 0, 1, 0, 0 \dots \dots, 0] \quad (9)$$

$$Bus_{speed} = [0, 0, 0, 36, 0, 0 \dots \dots, 0] \quad (10)$$

5.2.3.9 Action definition for DDQN-TSP

Action for DDQN-TSP is also the selection of the next phase in the variable phasing sequence. Green extensions time step, (Δt) defined earlier remains set to 3 seconds. Constraints on minimum green of any phases running before the priority phase are implemented.

5.2.3.10 Reward definition for DDQN-TSP

The reward function shown in Equation 11 is formulated as a combination of bus delay (b_d), and a penalty term for queuing on the side street. The side street queuing penalty (N) is as described earlier, imposed when queue length on any of the side street movements exceeds a set threshold. The queue length threshold, ql_{Th1} to impose a penalty defines/establishes the desired level of priority for the bus and the acceptable deterioration

of level of service for the side street. Whereas there can be technical metrics to define this threshold including the lateness of the bus, the available queue storage on the side street, the bus passenger load etc., the selection of the threshold is ultimately a policy question of how much the side street motorists can be delayed to benefit the bus on the main line.

$$\text{reward } r = -b_d - N \quad (11)$$

$$N = \begin{cases} 0, & ql_j \leq ql_{Th1} \\ -200, & ql_j > ql_{Th1} \end{cases}$$

5.2.3.11 Bus input

During the training of DDQN-TSP, buses enter the simulation with an average headway of 15 minutes (900 seconds) with a random term between -120 and +120 seconds, added to generate random arrivals. The 15 minutes is deemed sufficient to allow dissipation of the impacts of the prior TSP before another bus enters the system. Bus operations including routes and stops are modeled with PT Line and PT Line Stop modules available in VISSIM®. Buses are introduced into the simulation with the “AddvehicleToPT” method available in COM.

Algorithm 2: DDQN -TSP	
Input: learning rate lr , reward discount rate γ , memory buffer capacity, minibatch size, simulation time, T , number of episodes N , target update episodes N_{update}	
for episode = 1, N do	
	Re-load main DQN, target DQN, memory buffer, exploration probability ϵ
	Initialize decision point, time step Δt , traffic current state S_t , traffic next state S_{t+1} , bus current state Sb_t , bus next state Sb_{t+1} , bus detection
	for $t=1, T$ do
	Run one simulation step
	Update bus detection
	// select actions
	if $t = \text{decision point}$ then
	Choose action $a_t = \text{argmax } Q(S_t)$
	if bus detected = 1 then

			Choose action $a_t = \begin{cases} \operatorname{argmax} Q_b * (Sb_t), & \text{Uniform}(0,1) > \varepsilon \\ \text{random } a^t \in A^t, & \text{otherwise} \end{cases}$
			end if
			Compute Δt
			end if
			// set new signal timings in VISSIM®
			if decision point $\leq t < \text{decision point} + \Delta t$ then
			Execute action a_t
			end if
			if $t = \text{decision point} + \Delta t$ then
			Set $\text{decision point} = \text{decision point} + \Delta t$
			Obtain traffic next state S_{t+1} , bus next state Sb_{t+1} , reward rb_{t+1}
			if bus detected = 1 then
			store transition $(Sb_t, a_t, Sb_{t+1}, rb_{t+1})$
			end if
			Set $S_t = S_{t+1}$, $Sb_t = Sb_{t+1}$
			end if
			end for
			// train the TSP models
			If number of stored transitions $>$ batch size then
			sample random minibatch of transitions $(S_i, a_i, S_{i+1}, r_{i+1})$
			compute target, y^t
			$= \begin{cases} r^t & \text{if } sb^{t+1} \text{ is terminal} \\ r^t + \gamma Q_b(sb^{t+1}, \operatorname{argmax}_a Q_b(sb^{t+1}, a'; \theta); \theta') & \text{otherwise} \end{cases}$
			compute loss, $\mathcal{L}(\theta) = (y^t - Q_b(sb^t, a^t; \theta))^2$
			compute gradients and perform gradient descent to update main network weights
			if number of episodes $\% N_{\text{update}}=0$ then
			Update target network
			end if
			end if
			end for

5.2.4 Simulation Execution

As the network is based on a hypothetical intersection, default model calibration was assumed, i.e., for car following the Wiedemann 74 model is utilized with default parameters. Lane change behavior, acceleration, etc., also utilize default parameters. Each episode lasts for 30 minutes during the training of DDQN-SC and 4 hours during the training of DDQN-TSP. More time is taken for DDQN-TSP to allow for sufficient bus

samples at 15-minute headways. For both agents, random seeds remain unchanged during training. During testing, different random seeds are used for each run. Training on different random seeds showed slightly more instability and did not provide any meaningful benefits during testing compared to training on one random seed. Only results for one random seed training are included in the learning curves in the results section. Simulations runs and agent training are performed on an x64-based PC equipped with 12th Gen Intel(R) Core i9-12900, 2400 MHZ, 16 Core(s), 24 Logical Processor(s), 128 GB of RAM, Intel (R) UHD Graphics 770 GPU and Windows 11 operating system.

5.3 Results and Discussion

This section presents and discusses the study results. First the DDQN-SC agent is compared with actuated signal control (A-SC) to validate the agent's ability to generate signal timings that perform as well as well-timed current control strategies and qualify its use as a background controller in training DDQN-TSP. The second part of the results show the performance of the TSP agent including the learning progress during training and impacts on bus travel time and general traffic delay during the testing phase.

5.3.1 Performance of DDQN – SC

Figure 37 shows the learning curve for the DDQN-SC with the episode number on the x-axis and average reward for each episode on the y-axis. The average reward for an episode is computed by averaging rewards gained at each step in the episode. The algorithm progressively learns the best policy by exploration and exploitation, eventually converging after approximately 400 episodes. For the specified computer, each episode of 30 minutes requires on average 36.5 seconds for loading the model and all associated files and

parameters, running the model for 30 simulation minutes, training the model at the end of episode, and saving the output. It is seen from the graph that stability improves (variability reduces) as the model converges. However, there is room for improving the stability, which may potentially be accomplished through other variations of DQN, including prioritized experience replay (PER), extended dueling, and others. This stability issue is revisited later in chapter 7 where the performance of DDQN is compared with the performance of PPO.

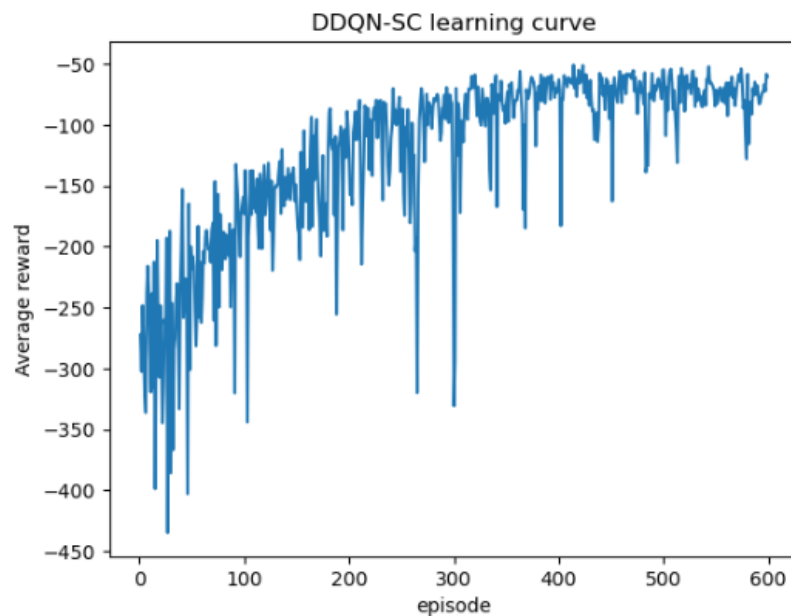


Figure 37: DDQN-SC learning curve.

Figure 38 shows a comparison of vehicle delay for four selected movements at the intersection for DDQN-SC and A-SC controls at (a) $v/c=0.6$ and (b) $v/c=0.95$. A-SC uses the inbuilt RBC controller in PTV-VISSIM® running in free mode, with the same minimum and maximum green times as set for DDQN-SC. Eastbound through (EB_TH) is the main street through movement, southbound through (SB_TH) is the side street through movement, southbound left turn (SB_LT) is the side street left turn movement and eastbound left turn (EB_LT) is the main street left turn movement. The results are from 10

replicate runs, each replicate being of 1 hour, including 15 minutes of warm up time and 45 minutes of data collection. The plotted data is the average vehicle delay from each replicate run and thus each box has 10 data points. The red square in the plot represents the average of the ten replicate run delays. For the side street movements (SB_TH and SB_LT), DDQN-SC results in lower delay compared to A-SC, with the difference more pronounced at $v/c = 0.6$. The main street through (EB_TH) delay is significantly lower for DDQN-SC at $v/c = 0.6$ and almost identical for both controls at $v/c = 0.95$. For main street LT (EB_LT), A-SC shows less delay at both levels of v/c , but again the difference is more evident at $v/c = 0.6$.

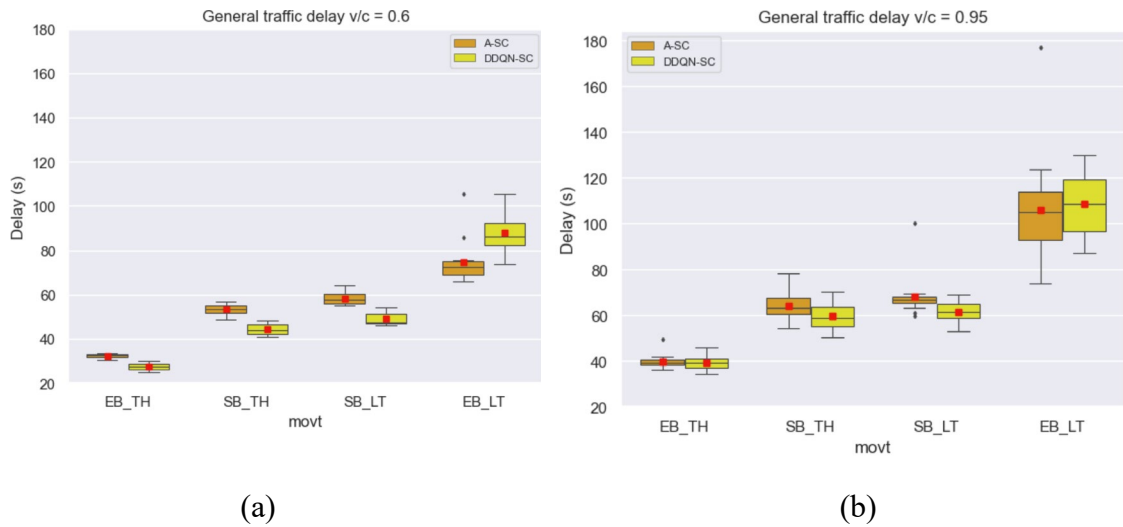


Figure 38: Comparing DDQN-SC and A-SC intersection delay for (a) $v/c = 0.6$ and (b) $v/c = 0.95$

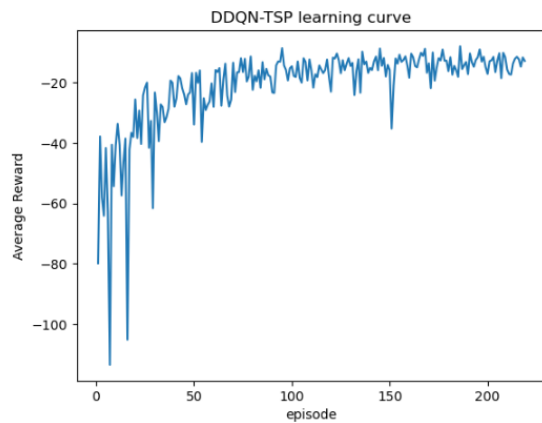
The observed relatively higher difference in performance of the two control types at $v/c = 0.6$ compared to $v/c = 0.95$ is intuitively reasonable. At $v/c = 0.95$, the intersection is close to capacity and most movements consistently max out, resulting in operation close to fixed time control for both DDQN-SC and A-SC. At $v/c = 0.6$, where there is more flexibility for the optimization, DDQN-SC shows more benefits for movements with the highest

volume. For example, on the main street, the EB_LT volume is 10% of the total approach volume and thus DDQN-SC favors EB_TH over EB_LT. Constraints in the reward function could be modified to alleviate this trade-off, if desired.

5.3.2 Performance of DDQN-TSP

Figure 39 (a) shows the learning curve for the DDQN-TSP with the episode number on the x-axis and average reward for each episode on the y-axis. As indicated in the methodology section, each episode is 4 hours long and includes 12 buses, with DDQN-TSP only running and collecting training data when the bus is on the approach. Figure 39 (b) shows the average bus delay during each episode. Each data point is an average of delay for the 12 buses in the episode. Bus delay progressively decreases with increasing episodes. From the figures it is seen that the algorithm converges after approximately 150 episodes.

Bus travel time with and without TSP at $v/c = 0.95$ is shown in Figure 40 (a). For without TSP, DDQN-TSP is not invoked, and DDQN-SC provides signal timing while the bus is present. Each simulation run contains 12 buses and lasts for 4 hours. Bus headway is 15 minutes, with a random term between -120 and +120 seconds, added to generate random bus arrivals. Overall bus travel time is reduced by approximately 21% with DDQN-TSP. Additional benefits could be realized by extending the agent control from a single intersection to multiple intersections, which is a subject of an ongoing study.



(a)



(b)

Figure 39: (a) DDQN-TSP learning curve and (b) Average bus delay during training.

Figure 40 (b) shows the general traffic delay for 3 selected movements with and without TSP, for 10 simulation runs. The vehicles included in the analysis traverse the intersection in the time interval 5 minutes (300 seconds) after bus check-in. The interval of 300 seconds is chosen following the results of Chapter 3 that showed that for v/c of 0.95, side street delay change persists up to about 300 seconds. It is seen that delay for the side street movements (SB_TH and SB_LT) marginally increases while the delay for main line through movement marginally reduces with TSP. This is intuitively reasonable as the main street through traffic benefits from increased green time given to the bus at the expense of side street traffic.

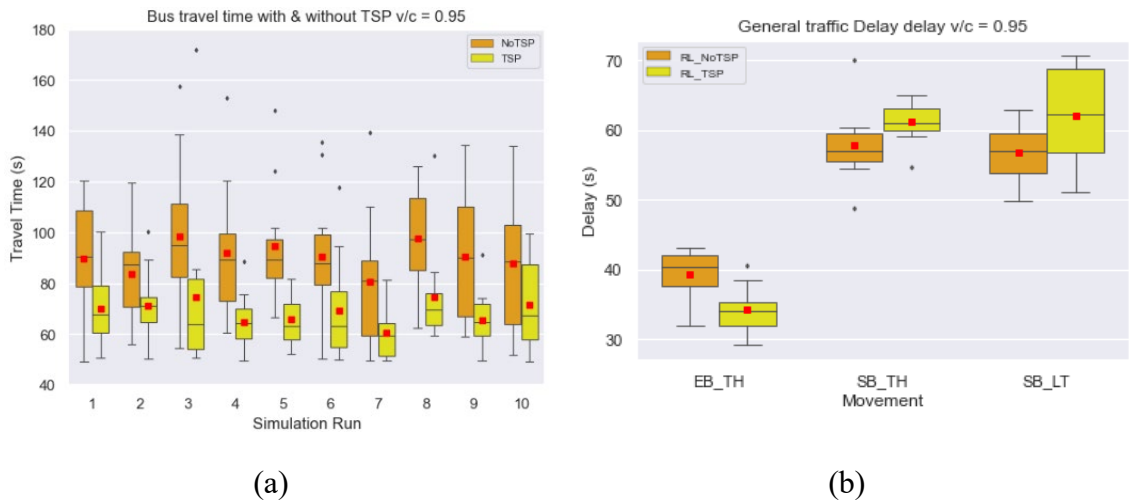


Figure 40: (a) Bus travel time with and without TSP and (b) General traffic delay with and without TSP

Lastly, a comparison is made between DDQN-TSP and A-SC TSP based on the bus travel time, Figure 41(a), and general traffic delay Figure 41(b). ASC_TSP and ASC_NoTSP, respectively, stand for A-SC control with and without TSP while RL_TSP and RL_NoTSP respectively stand for DDQN-TSP control with and without TSP. For TSP with A-SC, the inbuilt TSP algorithm in PTV VISSIM®'s RBC is used with green extension, red truncation, and skipping of the conflicted phases allowed. Maximum green extension is set to 20 seconds. It is seen that DDQN-TSP performs slightly better in reducing bus travel time. For the side street impact, the two algorithms have very comparable performance. DDQN-TSP leads to a slightly greater decrease in main through movement (EB_TH) with TSP, which is consistent with providing more priority to the bus. The delay difference for SB_TH is almost the same for both algorithms while for SB_LT, A-SC seems to have less impact. Under the stated conditions, it is seen that the two algorithms show a very comparable performance, as evaluated from bus travel time and general traffic delay.

Differences are likely to be seen if the algorithms are extended to multiple intersections which is a subject of an ongoing study.

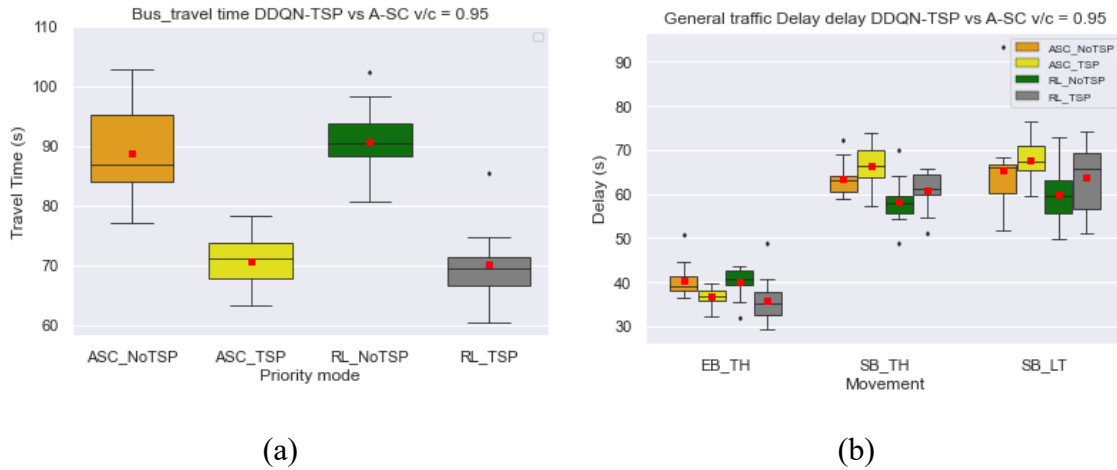


Figure 41: Comparison of DDQN-TSP and A-SC TSP based on (a) bus travel time and (b) General traffic delay.

5.4 Single Intersection RL-based TSP Summary

This study utilizes a microscopic simulation environment and CV data to develop and test an event-based RL agent that assumes intersection control from another RL based traffic signal controller when TSP transit buses enter the CV communication range of the intersection. The background general traffic controller is trained and tested, demonstrating comparable performance with an actuated controller for a single intersection. The trained RL based TSP agent is seen to reduce the bus travel time by about 21%, with marginal impacts to general traffic at a saturation rate of 0.95. The TSP agent also shows slightly better performance in improving bus travel time compared to actuated signal control with TSP. To improve run time efficiencies, PTV VISSIM®'s event-based scripting is used instead of the commonly used COM API. In the next chapter, the developed single agent formulation is extended to multiple agents to implement TSP at coordinated intersections.

CHAPTER 6 ADAPTIVE TRANSIT SIGNAL PRIORITY BASED ON MULTI-AGENT REINFORCEMENT LEARNING

6.1 Background

As discussed in Chapter 2, RL-based traffic control studies have mainly considered single agent formulations with only a few recent studies considering multiple agents (Chang et al. 2024; Chen et al. 2021; Fu et al. 2023; Li et al. 2023; Liu and Li 2023). In the same way, RL-based TSP has mainly been limited to single agent formulations with only a few recent efforts considering MARL-based TSP (Long et al. 2022). This chapter extends the single agent TSP modeling in the previous chapter to MARL-based TSP. This far, a single isolated intersection has been considered without the need to consider coordination, platooned arrivals, midblock bus stops and associated bus dwell-time, scalability issues, and the interaction of different agents in the same environment.

The first part of the study formulates MARL based adaptive signal control algorithm for an arterial road, not in the presence of a transit vehicle. Each arterial intersection is controlled by a single agent with all agents trained under centralized training and decentralized execution (CTDE) paradigm. The second part of the study formulates event triggered RL agents to implement TSP when a transit vehicle reaches within the defined CV communication range of the intersection. Two variations of MARL TSP agents are considered (1) independent TSP agents with decentralized training decentralized execution (DTDE) and (2) coordinated TSP agents under CTDE framework. In a case study, for a pair of coordinated intersections, both general traffic control agents and TSP agents are

trained and tested in a traffic microscopic simulation environment. Also discussed are architecture decisions made to improve run time efficiencies as these become more critical moving from single agent to multiple agents.

6.2 Methodology

6.2.1 *Multi-agent reinforcement learning overview*

MARL involves multiple agents interacting with a common environment and with each other. Figure 42 illustrates MARL framework with N agents interacting in the same environment. Each agent k obtains the local environment state s_k , uses policy π to select action a_k which is then implemented in the environment to get reward r_k . The most basic approach to formulating MARL problems is to reduce the problem to single agent learning through (a) central learning and (b) independent learning (Albrecht et al. 2024). Central learning trains a single central policy, π_c which takes global observations $(s_1, s_2, s_k, \dots, s_n)$ and selects a joint action set $(a_1, a_2, \dots, a_k, \dots, a_n)$ and sends each action a_k to each agent k for implementation. In independent learning, each agent each learns its individual policy π_k to select local actions a_k conditioning on local observations, s_k . In independent learning actions of other agents are only indirectly considered as part of environment dynamics. Both of these two approaches have their challenges. Central learning suffers from exponential growth of global action space with the increase in the number of agents. Independent learning suffers from non-stationarity of the environment due to actions of other agents in the same environment.

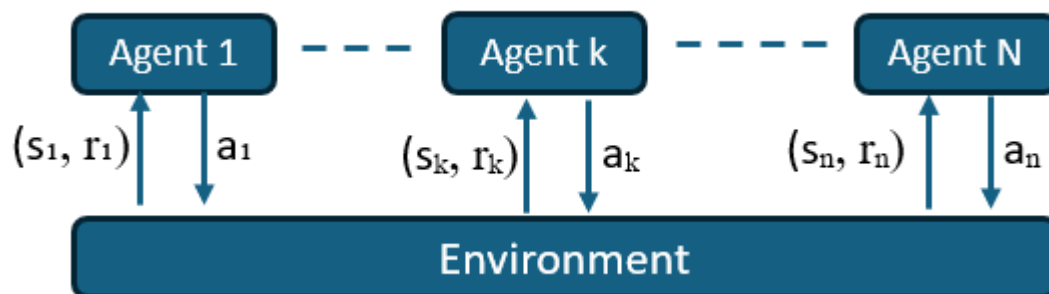


Figure 42: multi agent reinforcement learning framework

Moving from single agent to MARL introduces new challenges that require modification or formulation of new learning frameworks. These challenges include equilibrium selection, non-stationarity of the environment, scaling to many agents, multi-agent credit assignment, and partial observability (Albrecht et al. 2024; Zhang et al. 2021). The equilibrium selection problem results from the objectives of the different agents not necessarily being aligned and additionally there may be multiple equilibrium points that yield different returns for the agents. Non-stationarity of the environment results from actions of different agents with different objectives trying to optimize their policies. Non-stationarity of the environment invalidates the assumptions made under the single agent framework (Zhang et al. 2021). The scalability challenge results from the exponential growth of the action and state spaces with the number of agents. In single agent RL, the challenge of credit assignment only involves the question of which past actions contributed to the attained rewards but in MARL an additional question becomes which agent actions contributed to the received rewards (Albrecht et al. 2024). Partial observability results from agents not having access to the complete state of the environment which may lead to the selection of suboptimal actions.

6.2.1.1 Training and execution modes

Depending on the information available to the agents during training and during execution, MARL algorithms are categorized under decentralized training decentralized execution (DTDE), centralized training and centralized execution (CTCE) and centralized training decentralized execution (CTDE). In decentralized training, agents only have access to local observations and action histories while in centralized training agents have access to global observations and action histories. In decentralized execution, trained agents only condition on local observations to select their actions while in centralized execution agents condition on global observations to select actions.

In DTDE, each agent uses local observation and action histories to learn a local policy. The trained local policy is executed conditioning on local observations. DTDE is also called independent MARL. An example of value-based DTDE is independent deep q-network (IDQN). In IDQN each agent k trains its own parameterized action value function, $Q(\cdot; \theta_k)$ to select local actions a_k conditioning on local observations s_k and storing experiences in a separate memory buffer D_k . Independent REINFORCE is an example of policy-based DTDE. In independent REINFORCE, each agent k learns its own parameterized policy function to select local actions a_k conditioning on local observations s_k . The main challenge of DTDE is the non-stationarity of the environment due to the actions of other agents in the same environment. Despite the non-stationarity challenge, several studies applying RL to traffic signal control have reported good results with DTDE and it has been used as a reasonable baseline to evaluate the performance of more advanced algorithms (Chen et al. 2021; Fu et al. 2023; Rashid et al. 2018).

In CTCE, some type of information or mechanism is centrally shared both in training and execution. At the top of CTCE category is where the agents are reduced to a single agent using global observation and action histories to learn a global policy that is executed centrally. The key challenge of CTCE is scalability as the global action spaces grow exponentially with the number of agents. Another challenge of CTCE is the required communication structure to transfer local observations to the central agent and send back actions to be implemented by each agent. The required communication may be unattainable if agents are physically or virtually distributed entities (Albrecht et al. 2024).

Considering the challenges of purely DTDE and CTCE algorithms, a new paradigm of CTDE algorithms that is a hybrid of the two has come up and gained increased popularity. In CTDE, agents share some mechanism or information during training, but the learned policies are executed locally conditioning on local observations. A key advantage of CTDE system is that centralized functions can condition on full state of the environment and any other external data that does not need to be available during execution.

Popular value-based CTDE algorithms include value decomposition network (VDN) formulated by Sunehag et al. (2017) and Qmix formulated by Rashid et al. (2018). Both VDN and Qmix learn a central action value function with the two distinguished by how this function is decomposed. Popular policy-based CTDE algorithms based on actor-critic architecture have a centralized critic function that conditions on global states and any external data. These include centralized critic advantage actor-critic (A2C), centralized critic PPO, centralized critic deep deterministic policy gradient (DDPG) etc. Centralized critic PPO is explored in the next chapter. This chapter considers value-based CTDE algorithms.

6.2.1.2 Value decomposition network (VDN)

VDN trains a centralized action value function which is decomposed into individual agent action value functions that select local actions conditioning on local observations. As shown in Equation 12, the joint value function Q_{tot} is formulated as the sum of individual agent value functions Q_i which condition on local observations h^i .

$$Q_{tot} = \sum_{i=1}^d Q_i(h^i, a^i) \quad (12)$$

Figure 43. shows the algorithm implementation architecture adapted from the paper (Sunehag et al. 2017). On the forward pass the network for each agent takes local observations (Obs_1 and Obs_2) as the input and outputs the local value function (Q_1 and Q_2). The value functions are summed up to generate Q_{tot} which together with the common reward are used to compute the loss. The local action value functions (Q_i) are learned by backpropagating the gradients. The network incorporates an LSTM layer which has shown to improve performance in partially observable environments (Hausknecht and Stone 2015).

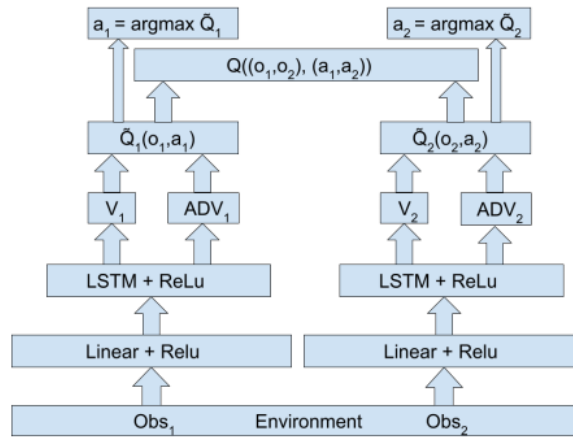


Figure 43. VDN individual architecture. Adapted from Sunehag et al. (2017)

6.2.1.3 Q-mix

Whereas in VDN the joint action value function is estimated as a simple sum of individual value action functions, in Q-mix the joint action value function is estimated by a neural network as a nonlinear combination of individual agent value functions. See Q-mix architecture in Figure 44 as adapted from the Rashid et al. (2018). The mixing network estimates the joint action value function. The weights of the mixing network are estimated by other hypernetworks (in red). A constraint is introduced to enforce monotonicity of the joint action value function in relation to the agent value functions. The constraint involves forcing the weights of the mixing network estimated by the hypernetworks to be positive which ensures that the local optimum actions are also global optimum actions. Training and implementation of Q-mix is the same as in VDN with the joint value function estimated on the forward pass and the agent value functions updated by back propagating the gradients.

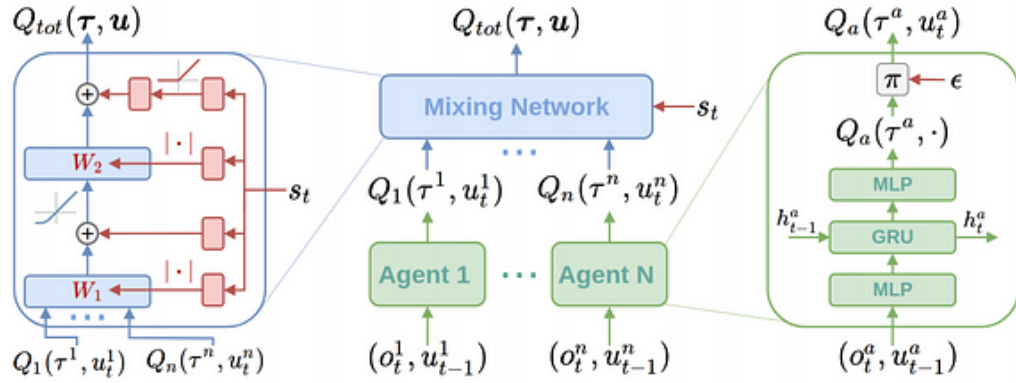


Figure 44: Q-mix architecture. Adapted from (Rashid et al. 2018)

This study adopts VDN architecture. VDN is chosen over Qmix for its simpler architecture and achieves good performance as shall be seen in the results section.

6.2.2 General traffic control Agents formulation

For the background controller (signal control when a transit vehicle is not present) an adaptive MARL algorithm based on VDN is formulated, trained, and tested. Each intersection on the arterial is controlled by a single agent. All agents have DDQN architecture in which there is a main network for predicting q-values and a target network for predicting the target q-values. Following the VDN architecture, the agents are trained centrally with a global action value function estimated as a sum of individual agent action value functions. Although the training is done centrally, in the execution each agent selects its own actions with the local action value function only conditioning on the local state. Algorithm 3 presents the pseudocode for MARL VDN for training N_A agents.

Algorithm 3: MARL VDN for traffic control	
Input: number of agents N_A , learning rate lr , reward discount rate γ , memory buffer capacity, minibatch size, simulation time, T , number of episodes N , target update episodes N_{update}	
for episode = 1, N do	
	Re-load main DQN, target DQN, memory buffer, exploration probability ϵ
	Initialize decision point, time step Δt , current state S_t , next state S_{t+1}
	for $t=1, T$ do
	Run one simulation step
	// select actions
	if $t = \text{decision point}$ then
	Obtain valid action set A_t^j for all agents $j \in \{1, \dots, N_A\}$
	Choose action $a_t^j = \begin{cases} \text{argmax } Q^j(S_t^j), & \text{Uniform}(0,1) > \epsilon \\ \text{random } a^j \in A_t^j, & \text{otherwise} \end{cases}$
	Compute Δt
	end if
	// set new signal timings in VISSIM®
	if $\text{decision point} \leq t < \text{decision point} + \Delta t$ then
	Execute action a_t^j
	end if
	if $t = \text{decision point} + \Delta t$ then
	Set $\text{decision point} = \text{decision point} + \Delta t$
	Obtain next state S_{t+1} , reward r_{t+1}
	store transition $(S_t, a_t, S_{t+1}, r_{t+1})$
	Set current state $S_t = \text{next state } S_{t+1}$
	end if
	end for
	// train the models
	If number of stored transitions $>$ batch size then
	sample random minibatch of transitions $(s_i, a_i, s_{i+1}, r_{i+1})$
	compute target, $y_i^j = \begin{cases} r_{i+1}, & \text{for terminal } s_{i+1} \\ r_{i+1} + \gamma * \max Q^j(S_{i+1}^j), & \text{for non terminal } s_{i+1} \end{cases}$
	Compute target, $y_i^{tot} = \sum_{j=1}^{N_A} y_i^j$; compute $Q_i^{tot} = \sum_{j=1}^{N_A} Q^j$
	compute loss, $L_i = (y_i^{tot} - Q_i^{tot})^2$
	compute gradients and perform gradient descent to update main network weights
	if number of episodes $\% N_{update} = 0$ then
	Update target network
	end if
	end if
	end for

6.2.2.1 State Definition

The problem is formulated as PoMDP in which each intersection agent has only access to local observations at that intersection. The study assumes the availability of CV data which enables more detailed definition of the traffic state. An alternative to CV data could be video camera data at the intersection. The main requirement is for the camera to be able to see the back of queue. At each intersection the state is defined by two components: (1) vehicle state (Veh_{state}) and signal state (Sig_{state}). As shown in Equation 13, Veh_{state} is a vector of the number of vehicles (v) in each lane (n) on each approach (m). With the assumption of CV data, the number of vehicles here is taken as the vehicles falling within the CV communication range (L ft) of the intersection or the within the block length whichever is the smallest. In the intersection signal state definition, the phase receiving green or yellow is populated with the duration of green or yellow that has elapsed. All other entries are set to zero. The study assumes that real time SPaT data is available for all intersections. Whereas the vehicle state is defined locally for each intersection, it is assumed that each intersection agent has access to the SPaT data for the immediate adjacent intersections. Thus, the signal state definition for each agent consists of the signal state at that intersection and at the immediately adjacent intersections. Equation 14 shows the Sig_{state} of an intersection with (n) adjacent intersections and a maximum of (p) entries at each included intersection. Each column represents one intersection.

$$Veh_{state} = \begin{bmatrix} v_{11} \\ v_{12} \\ \cdot \\ \cdot \\ v_{mn} \end{bmatrix} \quad (13)$$

$$Sig_{state} = \begin{bmatrix} 0 & \cdot & \theta_{1n} \\ \cdot & \cdot & \cdot \\ 22 & \cdot & \theta_{pn} \end{bmatrix} \quad (14)$$

6.2.2.2 Action definition

Action in this study is defined as the selection of the next phase from the available phases at that time step. This may be either keeping the current phase or selecting a different phase. For simplicity, a single ring barrier is considered as discussed in Chapter 5 and illustrated in **Figure 34** . The resultant phase splits are shown in Figure 45. The numbers in phase splits are according to NEMA movement codes. For instance, $\phi 15$ represents left turn movements 1 and 5.

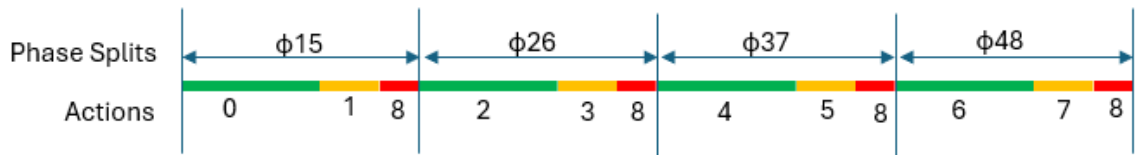


Figure 45: Splits and actions at each intersection

To enable centralized training with shared rewards, actions at all intersections are taken at the same time as training samples need to be from the same time step. The need to take actions at all agents at the same time introduces new complexity that did not exist for single agent in the previous chapter. In the previous chapter, actions were not taken during yellow interval, red clearance and minimum green durations. When the agent selected the next phase different from the current phase, yellow and red clearance and minimum green duration of the next phase were implemented before the agent selects another action. Because phase sequences and durations cannot be synchronized at multiple intersections, actions may need to be taken during yellow and red clearance intervals and minimum green duration.

Following previous studies like Long and Chung (2023b) and Fan and Yang (2024) yellow and red clearance intervals are included in the action set. Each of the four phases at each intersection is represented by two discrete entries for green and yellow indications. And finally, there is an entry for when all phases are red. Thus, a complete action set for each intersection includes 9 discrete actions (0,1,...,8) as shown in Figure 45. Including yellow and red clearance intervals increases the size of action space and potentially lengthens the training period. In the next chapter, a new formulation is proposed to address this challenge.

At the beginning of every time step, actions are selected at each intersection. Not all actions are available for selection at every time step. For instance, when minimum green is being serviced, only the minimum green action can be selected. At every time step, a set of valid actions is estimated, and invalid action masking (IAM) algorithm formalized by Huang and Ontañón (2020) is used to mask out the invalid actions in the action selection algorithm. Valid actions are selected considering the constraints of minimum green, maximum green, clearance phases and phasing sequence. In the implementation of IAM algorithm, predicted q-values of invalid actions are replaced with large negative numbers to prohibit the selection of the corresponding actions.

A key challenge of training RL agents in traffic microscopic simulation is run time efficiency, which worsens moving from single agent to multiple agents. Shorter time steps mean more frequent interruption of the simulation and thus higher run times. As shall be seen in the case study results, hundreds or even thousands of simulation runs were needed to reach convergence of each model. A base time step of 1 second is defined but can be optimized depending on signal states at the different intersections. The time step definition is better illustrated in the case study later in the chapter.

6.2.2.3 Reward definition

For centralized training, a reward function is selected to optimize the overall network traffic performance. The global reward, R is formulated as the average of local rewards at n intersections as shown in Equation 15.

$$\text{Reward, } R = \text{mean}(r_1, \dots, r_k, \dots, r_n) \forall i \in \{1, \dots, n\} \quad (15)$$

The local reward formulation is shown in Equation 16. The base local reward is taken as the average of the delay (d_i) for all vehicles (n) at the intersection. Two negative penalty terms are imposed on the local reward. The first penalty N is imposed when queue length on any of the side street movements (q_l) exceeds a set threshold (q_{lTh1}). The penalty is aimed at preventing excessive queuing on the side streets especially in the early training stages. The second penalty M is imposed when the phase changes from ϕ_t to ϕ_{t+1} before the end of maximum green when the queue length on the current phase $q_{l\phi_t}$ exceeds a set threshold q_{lTh2} . The M penalty is aimed at reducing instability in earlier training stages. Adding the two penalty terms significantly sped up the convergence. Because of the directionality of the peak traffic, it is quite often desired to have full bandwidth in the peak direction to progress the peak direction traffic through successive intersections. Additionally, in this study, it was desired to have full bandwidth coordination in the direction of the bus movement. Full bandwidth in the background coordination would later allow testing the TSP scenarios when the bus falls out the platoon to serve passengers at the midblock bus stop and the platoon continues through the downstream intersection. Directional coordination was achieved by including an extra term in the reward function. The base offset (θ) is estimated considering the block length and free flow speed. As a form

of initialization, agents for adjacent intersections receive a bonus positive reward (P) if the achieved offset (θ') was within a set $\Delta\theta$ seconds of the base offset (θ).

$$\text{reward } r_k = -\frac{\sum_1^n d_i}{n} - N - M + P \quad (16)$$

$$N = \begin{cases} 0, & ql_j \leq ql_{Th1} \\ -9999, & ql_j > ql_{Th1} \end{cases}$$

$$M = \begin{cases} 0, & \phi_{t+1} = \phi_t, ql_{\phi_t} \leq ql_{Th2} \\ -9999, & \phi_{t+1} \neq \phi_t, ql_{\phi_t} > ql_{Th2} \end{cases}$$

$$P = \begin{cases} 0, & \theta' - \Delta\theta \leq \theta \leq \theta' + \Delta\theta \\ 100, & \theta' - \Delta\theta > \theta \text{ or } \theta > \theta' + \Delta\theta \end{cases}$$

6.2.3 TSP control Agents

6.2.3.1 Overview

Using the trained RL agents as background traffic controllers, TSP agents, one for each intersection are formulated and trained. As already indicated two variations are considered, (1) independent TSP agents in DTDE framework and (2) coordinated TSP agents under CTDE framework. In the independent TSP variation, the objective is to train an agent at each intersection to implement TSP by selecting locally optimum TSP strategies at that intersection. With a midblock bus stops and with the intersections sufficiently apart, it can be hypothesized that independent TSP agents should suffice with no meaningful benefits expected from coordinating the TSP strategies. Each agent learns a separate action value function conditioning on its local state with the trained data collected when the TSP agent is active at the intersection. Only the intersection with the bus on the approach activates

the TSP agent while other intersections run on the background controllers. In the coordinated TSP variation, it is hypothesized that selecting coordinated TSP strategies at closely spaced intersections improves networkwide bus performance and limits networkwide impacts to the cross-street traffic. The coordinated TSP algorithm is based on VDN architecture as in the background MARL signal control algorithm. In a corridor wide implementation, which is a subject of Chapter 8, some intersections may need to run on independent TSP while others may work better on coordinated TSP depending on the spacing of intersections, bus stop locations etc. The coordinated TSP algorithm in this chapter considers two or more closely spaced intersections with TSP agents activated at each intersection when a bus first approaches the outmost intersection and all agents deactivated when the bus crosses the last intersection.

6.2.3.2 State definition

The local state definition for general traffic is expanded to include bus flow parameters. It is assumed that all buses are connected, broadcasting their location and speed utilizing basic safety messages (BSM) as they approach the intersection. The CV communication range of the intersection is defined as L ft from the stop line. The bus starts to communicate with the control agent at L ft from the stop line. The L ft of the approach is divided into (n) cells each of 25 ft. Two vectors of length (n) , with each entry representing a cell, are created to represent bus position and speed as shown in Equation 17 and Equation 18, respectively. When the bus occupies a cell, the corresponding entry in the location matrix is populated with 1 and the with bus speed in the speed vector, otherwise both entries are populated with zero. Equations 15 and 16 show an example of the bus in the 4th cell from the beginning of the link traveling towards the stop line with a speed of 36 mph.

$$Bus_{pos} = [0, 0, 0, 1, 0, 0 \dots \dots, 0] \quad (17)$$

$$Bus_{speed} = [0, 0, 0, 36, 0, 0 \dots \dots, 0] \quad (18)$$

6.2.3.3 Action definition

Action for TSP agents is also the selection of the next phase in the variable phasing sequence which consists of the same discrete set of 9 entries as in the general traffic controller discussion. IAM algorithm is applied to mask out invalid actions.

6.2.3.4 Reward

The objective of the TSP algorithm is to minimize bus delay and number of stops at the intersections while limiting adverse impacts to the side street. For independent agents, the local reward function shown in Equation 19 similar to the single agent reward formulation of the previous chapter.

$$reward\ r = -b_d - N \quad (19)$$

$$N = \begin{cases} 0, & ql_j \leq ql_{Th1} \\ -200, & ql_j > ql_{Th1} \end{cases}$$

For coordinated TSP, where the adjacent TSP agents are trained centrally, the local reward at each intersection is formulated as a function of general traffic delay, and bus delay as shown in Equation 20. A priority factor (PF) is applied to define the level of bus priority. PF becomes a hyperparameter which needs to be finetuned. A penalty, N is imposed to prevent excessive queuing on the side street. The global reward is taken as the average of the local rewards ($R = \text{mean}(r_1, \dots, r_n)$) where n is the number of agents where a coordinated TSP strategy is to be implemented.

$$\text{reward } r = -\frac{\sum_1^n d_i}{n} - PF * b_d - N \quad (20)$$

$$N = \begin{cases} 0, & ql_j \leq ql_{Th1} \\ -200, & ql_j > ql_{Th1} \end{cases}$$

6.2.4 Case study

The formulated agents are tested in a microscopic simulation environment for a pair of coordinated intersections.

6.2.4.1 Test network

As shown in Figure 46, the network model consists of two intersections A and B, 1600 ft apart. The 1600ft spacing is chosen to have the two intersections where coordination is likely beneficial while limiting the possibility of queue spillback. The CV communication range of each intersection is taken as 800 ft from the stop line. The main street is four lanes (two lanes each direction) with both intersections having a left turn bay on their EB and WB approaches. All N/S minor streets are two lanes (one lane each direction) with left turn bays on all approaches. The speed limit on both the main and cross street is assumed to be 40 mph and in VISSIM® the desired speed of all vehicles is set to 40 mph. VISSIM® default parameters for car following and lane changing models are maintained as this network is not modeled based on a field network. At both intersections, the main street has 1440 veh/h and 171 veh/h for through and left turns, respectively, while the cross street has 270 veh/h and 257 veh/h for through and left turns, respectively. For simplicity, no vehicles are assumed to turn right. The bus route modeled is Eastbound on the main street. There is a far-side bus stop on the route downstream of intersection A.

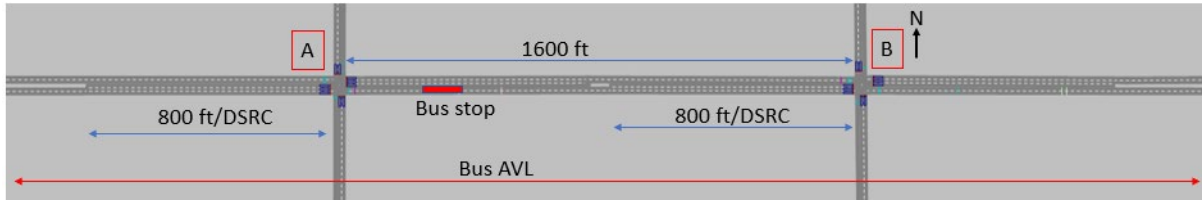


Figure 46: Case Study network model in VISSIM®

Bus movement is modeled using PT line and PT line stop modules available in VISSIM®. Dwell-time at the bus stop is modelled based on actual field Automated Passenger Counted (APC) data obtained from MARTA and analyzed in Chapter 4. In VISSIM®, dwell-time is entered as a continuous distribution function (CDF). VISSIM® assigns a dwell-time to each bus using the stop by randomly sampling from the entered dwell-time distribution. When the assigned dwell-time is zero, the stop is skipped. It is desired to have full bandwidth for the EB through movement.

6.2.4.2 General traffic control agents

(a) Definition of state, action, and reward

Following Equation 13, the local vehicle state is defined by a vector of 10 for a total of ten lanes on all approaches. Each entry consists of the number of vehicles in each lane within 800 ft CV communication range of the intersection. Following Equation 12, the signal state is defined by a matrix of two columns, each column representing the signal state at each of the two intersections. Each column has 9 rows as discussed in section 6.2.2. Action set at each intersection consists of a discrete set of 9 entries representing the green, yellow and all red displays for the different phases.

The algorithm time step (Δt) is taken as the minimum of the two time steps, each computed for each intersection. As defined in the previous chapter, the time step at each intersection is defined as a function of the current and next selected phases. If the next phase is the same as the current phase, a base time step of 1 second is considered to extend green. If the next phase is different from the current phase, time step becomes the sum of the current phase yellow and clearance intervals and minimum green of the next selected phase. In the current case of multiple agents where actions must be taken in clearance intervals and minimum green duration, Δt at each intersection is taken as the time remaining from the current time to the next decision/action point for that agent. For instance, if at the current decision point, intersection A is serving yellow interval, Δt for intersection A is computed as the sum of the remaining yellow phase, red clearance and the minimum green of the next selected phase. The overall Δt is then computed as the minimum of Δt at A and Δt at B. See Figure 47 for some example overall Δt . The alternative would be to take a fixed base Δt of se 0.1 second or 1 second but optimizing the time step significantly improved the run time efficiency especially in early stages of training where random actions are being selected and agents are serving mostly only minimum green to all phases.

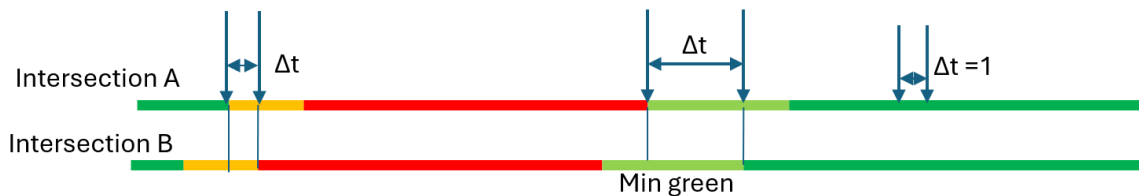


Figure 47: Time step selection

The local reward at each intersection is defined following equation 14.

6.2.4.3 Model training

After formulating the models, simulation runs are performed in VISSIM®, archiving training data at each time step that is later retrieved for training at the end of each episode/simulation run. Each episode lasts for 30 minutes with the same random seed through all runs. Event based scripts are preferred to drive the VISSIM® simulation as opposed to COM for their better runtime efficiency as explained in Chapter 5. Simulation runs and training are performed on an x64-based PC equipped with 12th Gen Intel(R) Core i9-12900, 2400 MHZ, 16 Core(s), 24 Logical Processor(s), 128 GB of RAM, Intel (R) UHD Graphics 770 GPU and Windows 11 operating system.

Preliminary training is undertaken to select the model hyperparameters. The final selected hyperparameters include: 3 layers of 128, 256 and 256 neurons with rectified linear unit activation function for both main and target networks, learning rate = 0.01, optimizer = Adam, discount rate (γ) = 0.99, exploration probability decay rate = 0.005, memory buffer capacity = 20000, and target network update frequency = 50 episodes.

6.2.4.4 TSP control agents

(a) Definition of state, action, and reward

The local traffic state is expanded to include bus state defined by bus position and bus speed as shown in Equation 17 and Equation 18 respectively. The CV communication range of each intersection is defined as 800 ft from the stop line. Dividing the 800 ft into cells of 25 ft gives a total of 32 cells. Thus, the bus position and speed vectors have length 32. Local action definition remains unchanged which involves the selection of the next signal display from 9 discrete options. For independent TSP agents, the reward for the bus

serving TSP is estimated with Equation 19. For coordinated TSP, local reward at each intersection is estimated by Equation 20.

(b) Model training

For independent agents, each episode/simulation run lasts for four hours with average bus headways of 15 minutes. For coordinated TSP agents, each episode lasts for 1 hour with average bus headways of 15 minutes. The architecture and the hyperparameters for each of the TSP agents are kept the same as for the traffic control agents which includes 3 layers of 128, 256 and 256 neurons with rectified linear unit activation function for both main and target networks, learning rate = 0.01, optimizer = Adam, discount rate (γ) = 0.99, exploration probability decay rate = 0.01, memory buffer capacity = 20000, and target network update frequency = 50 episodes.

6.3 Results

This section discusses the case study results. The first part compares the VDN based MARL for general traffic control with a coordinated actuated signal system under the same traffic conditions. The second part presents and discusses the performance of the event-based TSP control algorithms working on top of the VDN based MARL.

6.3.1 Performance of VDN based MARL for general traffic control

Figure 48 shows the VDN based MARL learning curve plotting episode number on horizontal axis and average reward on the vertical axis. Average reward is obtained by averaging rewards obtained at each time step within the episode. Each episode lasts for 30 minutes, with training data sampled after the run completion. As the agents are centrally

trained the learning curve is common to both agents. The reward is a global reward obtained by averaging the two local rewards, one from each agent. The agents progressively learn the policy, converging after around 900 episodes. From the plot it is seen that the learning process exhibits relatively good stability.

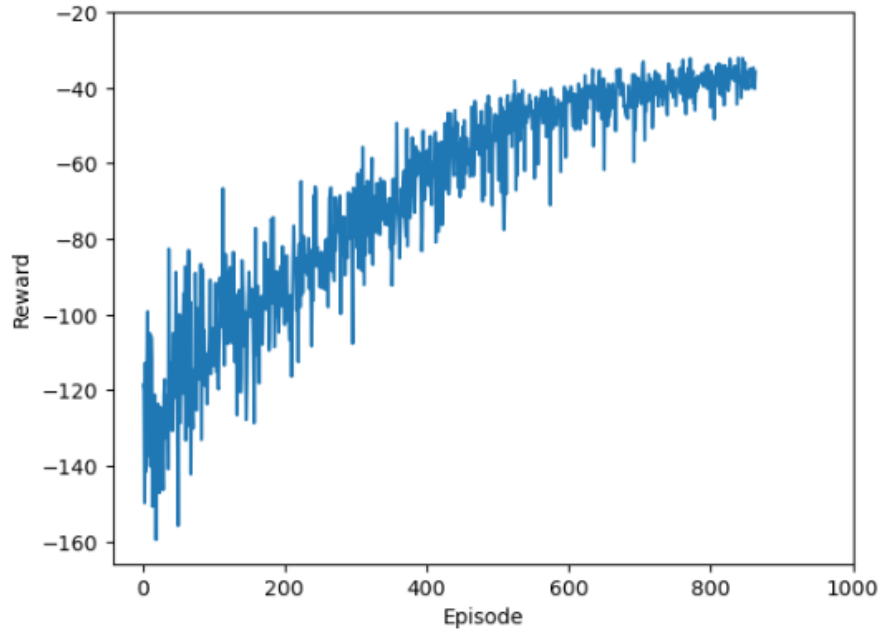


Figure 48: VDN Based MARL learning curve

Figure 49 shows a comparison of the performance of trained VDN based MARL and coordinated actuated signal control (ASC) at a volume to capacity ratio (v/c) of 0.95. Coordinated ASC timing parameters are estimated offline using commonly available commercial software. To allow for fair comparison with the VDN MARL solution similar constraints are imposed on the coordinated ASC solution, limiting to paired left turns and paired through movements on both main and cross streets which effectively reduces the timing to a four-phase single ring. Within Figure 49 six origin-destination (OD) movements are shown including corridor eastbound through (EB_TH), corridor westbound through (WB_TH), intersection A southbound through (A_SB_TH), intersection A northbound

through (A_NB_TH), intersection B southbound through (B_SB_TH), and intersection B northbound through (B_NB_TH). EB_TH is the coordinated movement. Each box plot includes data from 10 replicate runs with different random seeds. Each run lasts for 1 hour with the first 15 minutes as the warmup period with the below results over the last 45 minutes. The red square is the average of the data while the extents of the boxes indicate the upper and low quartiles.

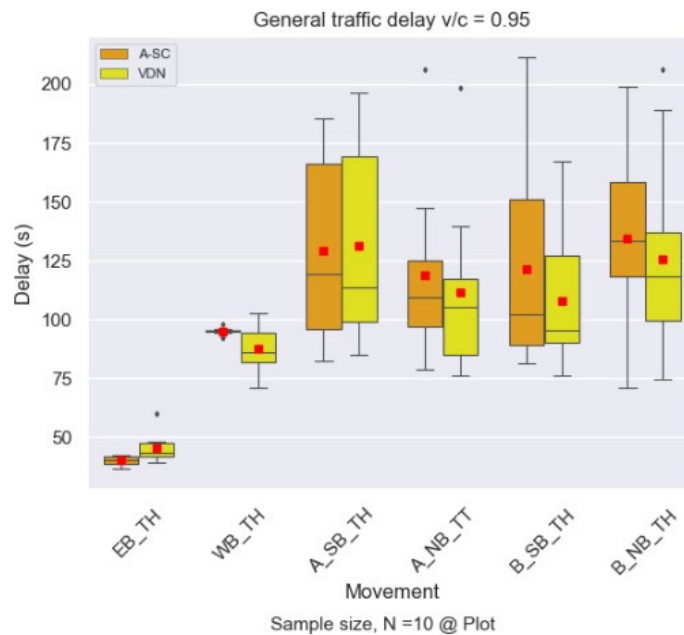


Figure 49: Comparing VDN MARL with ASC at v/c=0.95

ASC shows slightly better performance for the coordinated movement (EB_TH) but slightly worse performance for the opposite direction (WB_TH). This is an indication that the VDN solution is providing slightly more weight to the WB movement while the ASC solution more heavily favors the EB movement, although both solutions are achieving superior EB coordination. On the side street both control systems show high variability. VDN based MARL shows slightly lower means and medians in all but one instance, as well as lower range values. Overall VDN based MARL arguably shows slightly better

performance compared to ASC. As shown in the previous chapter, the superiority of the RL based control is likely to be more pronounced at lower v/c values where increased flexibility in the optimization with more frequent gap out opportunities.

It is seen that the implicit communication in VDN and the selected reward formulation achieved good coordination between the two signals without additional explicit communication modules as applied by Bokade et al. (2023); Chang et al. (2024); Chen et al. (2021). The training achieves a key objective of having an adaptive control algorithm working at least as well as ASC and achieving progression in the EB direction for the subsequent TSP experiments with the bus route in the EB direction.

6.3.2 *Independent TSP agents*

6.3.2.1 Learning rate and convergence

Figure 50 shows bus delay evolution during training for the two intersections. Each episode lasts for four hours with average bus headways of 15 minutes. The same random seed is utilized across the episodes. Each plotted point represents the average delay for all buses within each episode, which is typically 16 buses per episode. For both agents, bus delay progressively reduces as the agents learn the optimal policy and converges after approximately 200 episodes. However, sizable variability in bus delay persists even after convergence, which is indicative of the presence of instability in the training. The instability associated with DTDE is not unexpected as both agents interact with VISSIM® at the same time, leading to non-stationarity of the environment.

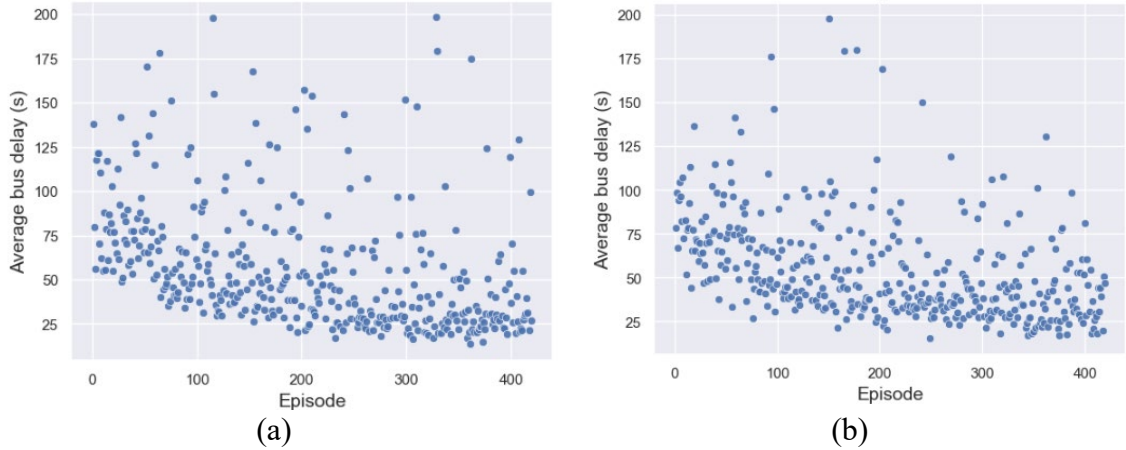


Figure 50: Average bus delay at (a) Intersection A and (b) Intersection B during training for independent TSP agents

6.3.2.2 Bus travel time

New simulation runs are performed to test the trained TSP agents. Each run lasts for 4 hours and includes 16 buses. Average bus headways are 15 minutes (900 seconds) with a random term added to the headway to generate random arrival profiles. Figure 51 shows bus travel time, with and without TSP, measured across three sections: (1) crossing intersection A, ending prior to the far-side bus stop, labeled Inter A_EB; (2) crossing intersection B, starting immediately after the bus stop. labeled Inter B_EB; and (3) crossing intersections A and B, labeled Inter A&B_EB. Inter A&B_EB includes dwell-time at the bus stop. The Figure 51 plots are based on the average travel time value from each of the 10 simulation runs. The no TSP case measures bus travel time with the two intersections running on VDN based MARL with no special bus treatment. The TSP case measures travel time with the TSP agents activated when the bus approaches the intersection. It is seen that TSP significantly reduces bus travel time across both intersections. The high variability of travel time in the no TSP case across intersection B (i.e., Inter_B_EB) is explained by the bus occasionally falling out of the platoon to serve bus passengers at the

stop. TSP significantly reduces this travel time variability. Averaging across runs, the travel time across both intersections reduces by 22% with TSP activated.

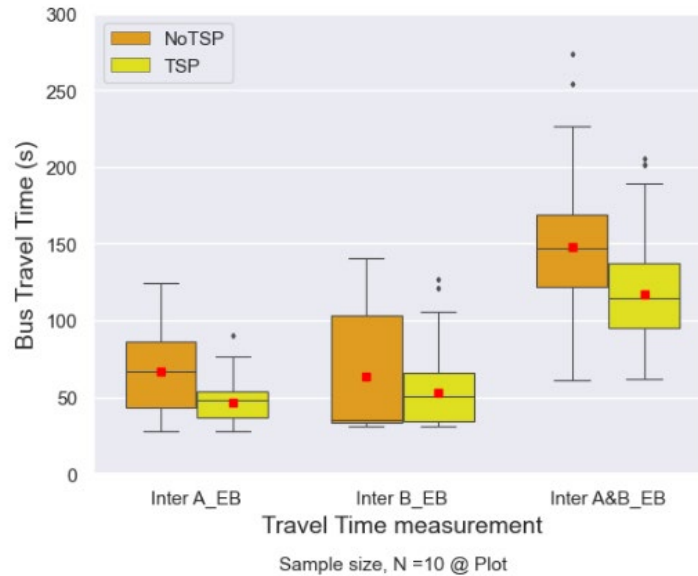


Figure 51: Bus travel time with and without TSP for Independent TSP agents

6.3.2.3 Side street Delay

Figure 52 shows general traffic delay for selected side street movements with and without TSP for averaged across the ten simulation runs. A_SB_TH and A_SB_LT stand for the southbound through and left-turn movements at intersection A, respectively, while B_SB_TH and B_SB_LT stand for the southbound through and left-turn movements at intersection B, respectively. The vehicles included in the analysis traverse the intersection in the time interval of five minutes (300 seconds) after bus check-in. The interval of 300 seconds is chosen following the results of Chapter 3 that showed that for v/c of 0.95, side street delay change persists up to approximately 300 seconds. For all movements except for B_SB_LT, general traffic delay only slightly increases when TSP is activated. The

adaptive algorithm can successfully adjust the side street movement phasing, compensating for the awarded bus priority. For B_SB_LT delay increases by about 18% after TSP. Additional refinement to the reward function could be considered to better balance the impacts across movements.

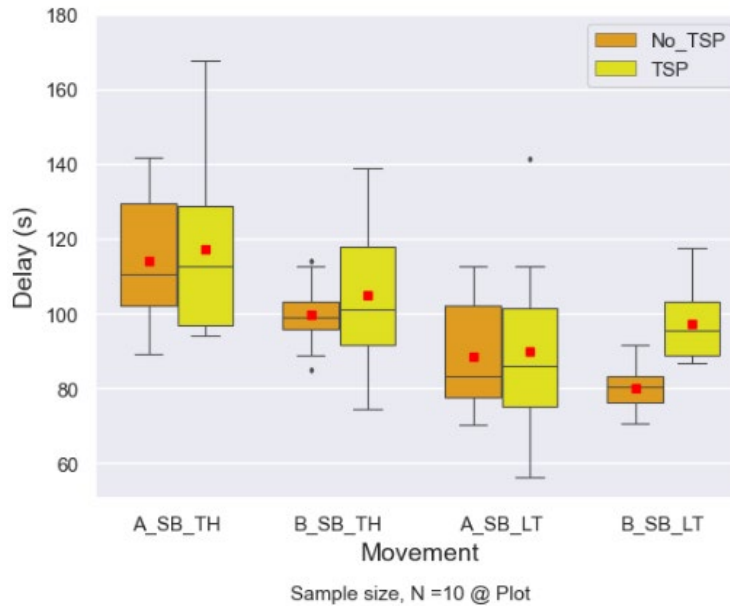


Figure 52: Side Street delay with and without TSP

6.3.3 Coordinated TSP agents

6.3.3.1 Learning rate and convergence

Figure 53 shows bus delay evolution during training for (a) intersection A and (b) intersection B for coordinated TSP agents. Different from the independent TSP experiment, in this case, each episode lasts for 1 hour with average bus headways of 15 minutes. The same random seed is utilized across episodes. Each plotted point is the average delay for all buses within the episode, which is typically 4 buses. For both agents, bus delay progressively reduces as the agents learn the optimal policy, converging after

approximately 800 episodes. The 800 episodes of one hour each implies relatively the same amount of simulation time for convergence as in the 200 episodes of four hour each for the independent agents of Figure 50. Compared to independent agents, the training exhibits good stability, which is facilitated by centralized training. A comparison of Figure 50 and Figure 53 illustrates the challenge of the non-stationarity associated with DTDE. Nevertheless, in both cases of independent training and centralized training, the agents converge to the same bus delay value of approximately 25 seconds.

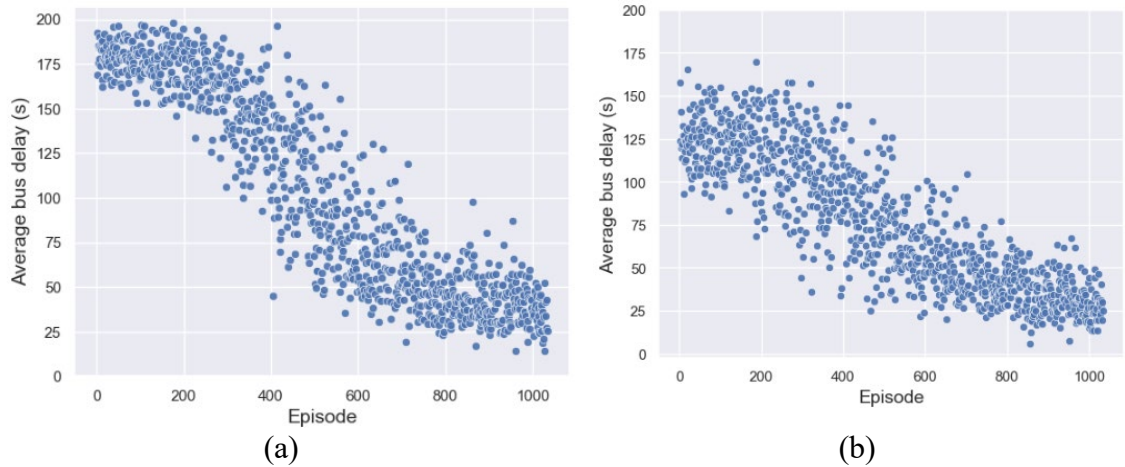


Figure 53: Average bus delay at (a) Intersection A and (b) Intersection B during training for coordinated TSP agents

6.3.3.2 Bus travel time

Figure 54 shows bus travel with and without TSP for coordinated TSP agents. The setup for test runs and the plotted movements are the same as described for the independent TSP agents in section 6.3.2 and Figure 51. Across the two intersections (i.e., Inter A&B_EB), coordinated TSP agents achieve a higher reduction in bus travel time compared to independent TSP agents. Bus travel time reduces by 27% compared to the 22% achieved with independent TSP agents. Contributing factors for this improvement may include: (1)

the utilization of coordinated TSP strategy reward functions specifically focused on multi-intersections operations, (2) intersection B having additional lead-time to adjust the signal timing when the bus is detected at A, and (3) improved stability in the training.

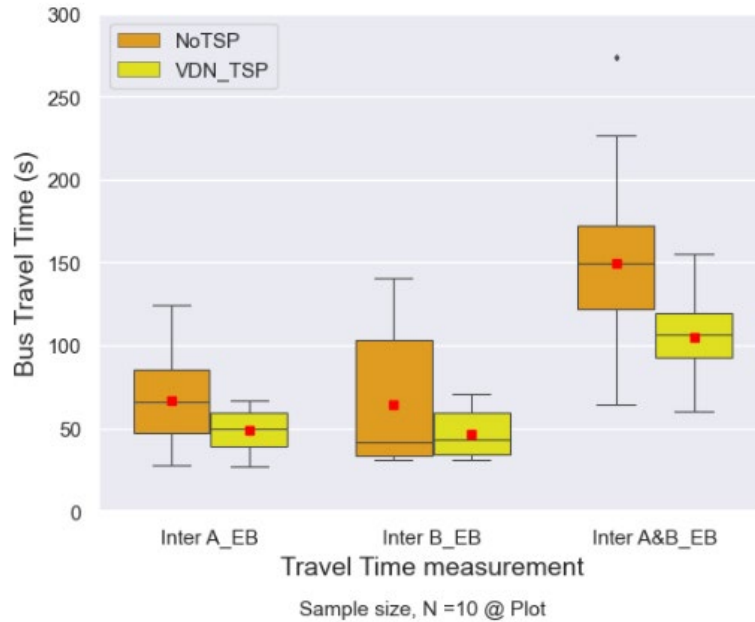


Figure 54: Bus travel time with and without TSP for coordinated TSP agents

6.3.3.3 Side street Delay

Figure 55 shows general traffic delay for the selected side street movements with and without TSP for coordinated TSP agents. For all movements except for A_SB_LT, side street delay increases slightly after TSP. The unexpected slight reduction in delay for A_SB_LT is likely due to the algorithm “overcompensating” for the time taken from the movement after TSP.

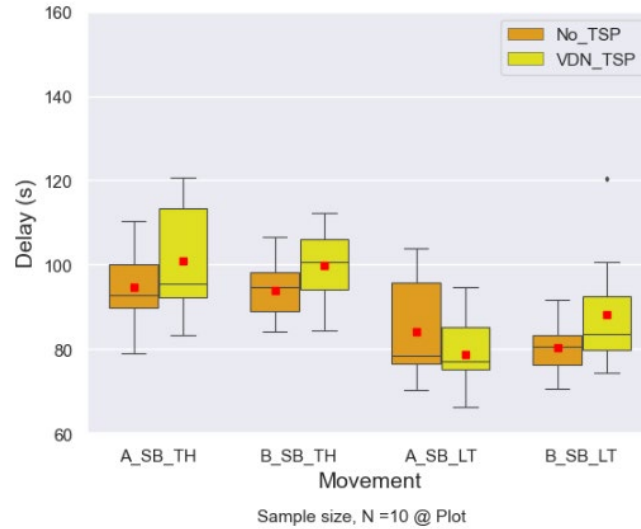


Figure 55: Side Street delay with and without TSP

6.4 MARL-based TSP Summary

This study integrates TSP into MARL for signal control. The first part of the study develops adaptive signal control based on MARL and CTDE paradigm for multiple coordinated intersections. The second part of the study formulates TSP control agents to implement TSP with a background adaptive RL-based general traffic control. Two variations of TSP control agents are formulated. In one variation, independent TSP agents are formulated under DTDE framework to implement TSP at each intersection. In the second variation, the TSP agents are centrally trained under CTDE framework and VDN architecture to select and implement coordinated TSP strategies across the adjacent intersections.

The formulated agents are trained and tested on a case study network of two coordinated intersections with a bus route in one direction and a bus stop between the two intersections. The trained general traffic agents show slightly better performance compared to coordinated actuated signal control based on overall intersection delay at v/c of 0.95. Both

independent and coordinated TSP agents converge to the same bus delay value, but independent agents show high instability throughout the training process. The two independent TSP agents reduce bus delay across the two intersections by 22% compared to the no TSP case while the coordinated TSP agents achieve 27% delay reduction. In both cases of TSP, there is slight increase in delay within five minutes of bus check-in for the majority of side street movements.

CHAPTER 7 CORRIDOR LEVEL ADAPTIVE REINFORCEMENT LEARNING BASED SIGNAL CONTROL

7.1 Background

As indicated in Chapter 2, most previous RL studies are limited to single agent formulations with only a few recent studies formulating MARL based traffic control. Signal coordination is at the heart of arterial traffic flow optimization. A key objective of MARL-based traffic control algorithms is to promote cooperation between individual intersection agents to progress traffic through adjacent intersections.

The few MARL studies almost universally adopt value- based MARL methods including q-mix based algorithms (Bokade et al. 2023; Chang et al. 2024; Chen et al. 2021; Fu et al. 2023), value network decomposition, VDN (Li et al. 2023) and independent q-learning algorithms (Bokade et al. 2023; Chang et al. 2024; Chen et al. 2021; Fu et al. 2023; Li et al. 2023; Wu et al. 2020). However, policy gradient algorithms are likely to achieve better performance in partially observable multi-agent environments compared to value-based methods (Albrecht et al. 2024; Morales 2020; Yu et al. 2022). In a study intriguingly entitled “the surprising effectiveness of PPO in cooperative multi-agent games”, the authors compared the performance of multi-agent proximal policy optimization (PPO) and other centralized training decentralized execution (CTDE) algorithms including Q-mix for four multi-agent environments. PPO showed comparable or superior performance and required minimum hyperparameter tuning compared to other algorithms (Yu et al. 2022).

Previous RL-based traffic control studies almost universally assume very simplified signal plans including limiting the number of allowable phases (Aslani et al. 2019; Bálint et al. 2022; Bokade et al. 2023; Bouktif et al. 2021; Bouktif et al. 2023; Li et al. 2020; Li et al. 2016; Li et al. 2021; Liu and Li 2023; Liu et al. 2023; Shabestary et al. 2020), assuming fixed timing sequences (Aslani et al. 2019; Bálint et al. 2022; Bokade et al. 2023; Bouktif et al. 2021; Bouktif et al. 2023; Casas 2017; Chang et al. 2024; Fu et al. 2023; Lee et al. 2022; Li et al. 2020; Li et al. 2016; Li et al. 2021; Liu and Li 2023; Liu et al. 2023; Shabestary et al. 2020; Wu and Guler 2019), fixing the duration or intervals of green for each selected phase or phase combinations (Bálint et al. 2022; Chang et al. 2024; Li et al. 2020; Liang et al. 2019), and utilizing single ring barrier configurations (Bálint et al. 2022; Chang et al. 2024; Fan and Yang 2024; Li et al. 2020; Liang et al. 2019; Wang et al. 2019). For most part therefore, RL approaches remain untested for more complex field signal timing plans.

This chapter formulates a multi-agent PPO (MA-PPO) adaptive signal control algorithm and tests the same on a real-world corridor with actual/complete traffic movements, traffic volumes, and network geometry including intersection spacings. The formulated MA-PPO has centralized critic architecture in which each agent is formulated with an actor network that selects independent actions conditioning on local observations and a centralized critic that estimates the agent's value function conditioning on global observations. All agents are formulated to allow selection and implementation of up to eight signal phases as commonly implemented in the field controllers. For the test corridor and field measured traffic volumes, the performance of the formulated MA-PPO is compared against currently field implemented actuated-coordinated signal timings modeled using VISSIM®-

MaxTime® software in the loop simulation (SILs). To test the robustness of the formulated algorithm, sensitivity experiments are performed with volumes adjusted (a) upwards by 5% and (b) downwards by 10% from the field measured volumes

7.2 Methodology

7.2.1 Proximal Policy Optimization (PPO) Overview

PPO belongs to a family of PG algorithms called actor-critic. Actor-critic algorithms have the actor network that learns a parameterized policy function to select actions and a critic network that learns a parameterized value function to evaluate the actions selected by the actor network. For discrete action spaces, on the forward pass, the actor network takes the environment state, s as input and outputs a probability distribution over the discrete action set. Actions to take are sampled from the output probability distribution. During training PG methods compute gradients of the loss function and optimize the policy parameters to follow the direction of higher expected returns. Returns are evaluated by state value and action value functions, $V(s)$ and $Q(s, a)$, respectively. The foundational actor-critic algorithm called advantage actor critic (A2C) computes estimates of advantage, $Adv(s, a)$ as shown in Equation 21 to guide policy gradients. Advantage intuitively quantifies how better it is to take a specific action over the average general action in a given state.

$$Adv(s, a) = Q(s, a) - V(s) = \begin{cases} r^t & \text{if } s^{t+1} \text{ is terminal} \\ r^t + \gamma V(s^{t+1}) - V(s^t) & \text{otherwise} \end{cases} \quad (21)$$

Gradients are computed from the gradient theorem of Equation 22 where J is the function measuring the quality of policy, $\Pr(\cdot|\pi)$ represents the state visitation probability under the policy π , s is the state and ϕ are the policy parameters (Albrecht et al. 2024).

$$\nabla_{\varphi} J(\varphi) = E_{s \sim \Pr(\cdot|\pi), a \sim \pi(\cdot|s; \varphi)} \left[Q^{\pi}(s, a) \frac{\nabla_{\varphi} \pi(a|s; \varphi)}{\pi(a|s; \varphi)} \right] \quad (22)$$

The actor loss $L(\varphi)$ is computed as in Equation 23 with the policy term and the advantage term which acts the weighting/magnitude.

$$\mathcal{L}(\varphi) = -Adv(s^t, a^t) \log \pi(a^t | s^t; \varphi) \quad (23)$$

On a forward pass, the critic takes the state as input and outputs a single value of the state value function. The critic optimization loss $L(\vartheta)$ shown in Equation 24 is computed as the squared error of the value estimate and the bootstrapped target estimate y^t where ϑ are the parameters of the critic network.

$$\mathcal{L}(\vartheta) = (y^t - V(s^t; \vartheta))^2$$

$$y^t = \begin{cases} r^t & \text{if } s^{t+1} \text{ is terminal} \\ r^t + \gamma V^{\pi}(s^{t+1}; \vartheta) & \text{otherwise} \end{cases} \quad (24)$$

A potential challenge of PG methods is the significant change in policy from any individual gradient update which can result in instability. Trust region policy optimization (TRPO) formulated by Schulman (2015) and PPO formulated by (Schulman et al. 2017) try to mitigate this risk. PPO builds on TRPO to formulate a computationally efficient surrogate objective function based on importance sampling weights, $\rho(s, a)$ which is defined as the ratio of probabilities of selecting an action (a) in state (s) for the new and old policies as shown in Equation 25.

$$\rho(s, a) = \frac{\pi_{new}(a|s; \varphi)}{\pi_{old}(a|s)} \quad (25)$$

The loss function shown in Equation 26 is computed by clipping $\rho(s,a)$ to restrict big changes to the policy for a single gradient update. The hyperparameter ϵ is selected to restrict how much the policy is allowed to change in a single update. Using $\rho(s,a)$, PPO is able to restrict the magnitude of policy updates and update the policy multiple times using the same data.

$$\mathcal{L}(\varphi) = - \min \left(\begin{array}{l} \rho(s^t, a^t) Adv(s^t, a^t) \\ clip(\rho(s^t, a^t), 1 - \epsilon, 1 + \epsilon) Adv(s^t, a^t) \end{array} \right) \quad (26)$$

7.2.1.1 Multi-agent PPO – Centralized Critic

PPO with centralized critic belongs to CTDE paradigm of MARL. Each agent’s actor network selects local actions conditioning on only local observations or local observation histories. The loss function of the actor remains as described earlier. The critic network is only utilized during training and can therefore utilize information that may not be accessible or available during execution. Centralized critics can condition on full state of the environment and any external data.

Figure 56 adapted from (Albrecht et al. 2024) shows the architecture of centralized critic for agent i conditioning on local observations of agent i (s_i) and on observations of all other agents in the environment (s_{n-i}) while still approximating the value of agent i ’s policy. In this centralized state value-based critic architecture, each agent i gets a local reward r_i and doesn’t take actions from other agents as inputs as done in centralized action-value based critics. Despite not taking action inputs, centralized state-value critics can still approximate advantage functions that provide preference over particular functions (Albrecht et al. 2024).

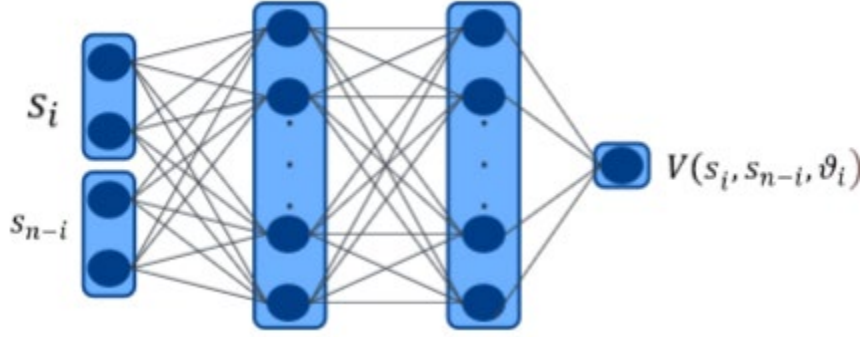


Figure 56: Architecture of Centralized critic for agent i, adapted from (Albrecht et al. 2024)

The loss function for agent i's critic is as in Equation 27 where ϑ_i are the network parameters, and r is the agent reward with other variables as defined before.

$$\mathcal{L}(\vartheta_i) = (y_i - V(s_i^t, s_{n-i}^t; \vartheta_i))^2$$

$$y_i = r_i^t + \gamma V(s_i^t, s_{n-i}^t; \vartheta_i) \quad (27)$$

7.2.2 Comparing PPO and DQN convergence properties

In the previous two chapters value-based DQN is utilized but as indicated, recent literature shows that PG methods may have better stability, convergence and better performance especially in partially observable environments (Albrecht et al. 2024; Morales 2020; Yu et al. 2022). Exploratory training is performed to compare the convergence properties of PPO and DQN. The single intersection set up of chapter 5 is considering including the network model, state and action definitions. Reward definition is modified replacing the base term of average delay summation with a summation of normalized vehicle delays as shown in Equation 28. In the equation d_v is the individual vehicle delay while d_{\max} is taken as the

highest expected vehicle delay for the intersection. All other terms are the same as described in chapter 5.

$$\text{reward } r = - \sum_{v \in V} \frac{d_v^t}{d_{max}} - N - M \quad (28)$$

$$N = \begin{cases} 0, & ql_j \leq ql_{Th1} \\ -200, & ql_j > ql_{Th1} \end{cases}$$

$$M = \begin{cases} 0, & \phi_{t+1} = \phi_t \\ -200, & \phi_{t+1} \neq \phi_t, ql_{\phi_t} > ql_{Th2} \end{cases}$$

Figure 57 shows a comparison of learning curves for DDQN and PPO. While both algorithms converge to almost the same reward value, PPO exhibits greater stability/less variability after convergence. Further hyperparameter finetuning and enhancements to DDQN including PER and dueling DQN could be pursued to improve the convergency of DDQN but again this shows the advantage of PPO that comparable or superior performance can be achieved with minimum tuning compared to DQN. The rest of the dissertation adopts PPO and it's multi-agent version, MA-PPO.

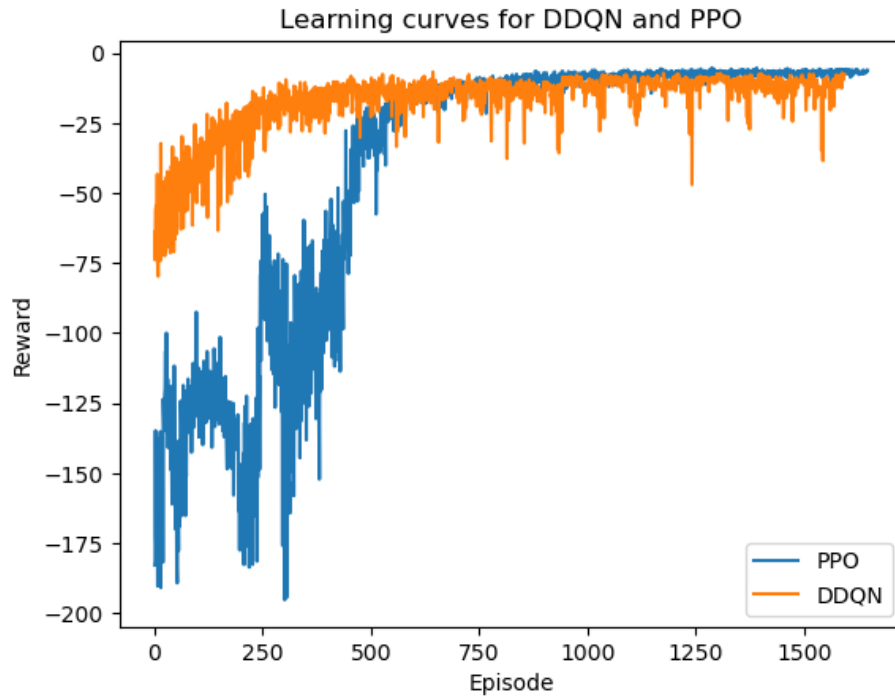


Figure 57: Comparing DDQN and PPO learning curves

7.2.3 Definition of state, action and reward functions

7.2.3.1 State definition

The problem is formulated as a DecPoMDP. At each intersection, the intersection actor network has access to only local observations. See Figure 58 showing the local observations/state (shaded) for agent i running intersection i that is part of a large arterial network.

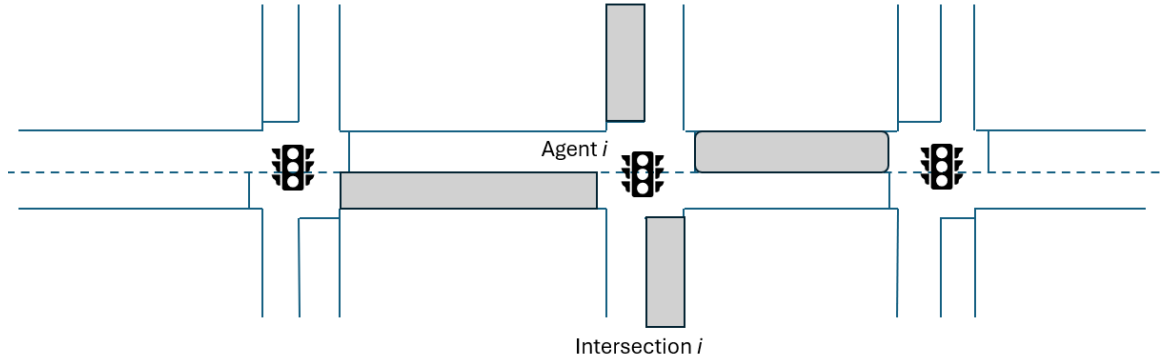


Figure 58: Partially observable environment

The local state has two component vectors, (1) vehicle state (Veh_{state}) and signal state (Sig_{state}). As shown in Equation 29, Veh_{state} is a vector of the number of vehicles (v) in each lane (n) on each approach (m). The study assumes the availability of connected vehicle (CV) data or video camera data at the intersection to deduce the traffic count at all the approaches of the intersection. The approach distance considered is the distance to the upstream intersection.

$$Veh_{state} = \begin{bmatrix} v_{11} \\ v_{12} \\ \cdot \\ \cdot \\ v_{mn} \end{bmatrix} \quad (29)$$

The Sig_{state} shown in Equation 30 has total entries corresponding to the total number of intersection signal phases (n_p). The entry for phase/s receiving green time are populated with the duration of green that has elapsed while all other entries are populated with zero. For all intersections of the corridor, the study assumes the availability of real time SPaT data which is increasingly ubiquitous.

$$Sig_{state} = \begin{bmatrix} 10 \\ 0 \\ 12 \\ \cdot \\ 0 \end{bmatrix} \quad (30)$$

The state for the centralized critic at intersection i (Sc_i) shown in Equation 31 is defined by (1) the local state of the agent which consists of Veh_{state} and Sig_{state} as defined above and (2) a concatenation of local states at all n other intersections. The ordering may matter. In this case the local state at the subject intersection i is placed in the first position.

$$Sc_i = [[Veh_{state_i} + Sig_{state_i}] + [Veh_{state_1} + Sig_{state_1} + \dots + [Veh_{state_n} + Sig_{state_n}]] \quad (31)$$

It is perhaps worth reiterating that the critic network is only used/needed during training and not during execution. During execution therefore, the trained algorithm shall not need access to the global state defined in Equation 31. This allows the flexibility on the information that can be included in the global state including external data.

7.2.3.2 Action definition and implementation

In this study, at each time t , an action is defined as whether the signal keeps the current phase or switches to any other phase in the set of valid phases. When the action changes from remaining in the same phase in time $t-1$ to selecting a different phase in t , the current phase yellow and red clearance intervals are displayed followed by the minimum green for the next selected phase. Every phase will have set minimum green that must be served, the phase green may be longer but not shorter than the minimum. Yellow and red clearance intervals allow safe termination of the current phase.

In Chapter 5 and Chapter 6, a single ring barrier was considered for simplicity. To more realistically replicate the field implemented signal plans, this study defines agent actions based on full dual ring barrier control, with the eight phase configurations as shown in Figure 59. In dual ring control one phase from each ring is active simultaneously, where Ring 1 consists of phases [1,2,3,4] and Ring 2 consists of phases [5,6,7,8]. Active phases must not conflict, thus the barriers are defined by the ends of phases [2,6] and [4,8]. When crossing a barrier both rings must change phases simultaneously. Accordingly, there are eight possible discrete actions [0,1,2,3,4,5,6,7] corresponding to eight paired phase combinations, [1,5], [1,6], [2,5], [2,6], [3,7], [3,8], [4,7], and [4,8]. Each action represents a compatible phase combination with one phase from each ring. For instance, action 0 calls for phase 1 in ring 1 and phase 5 in ring 2.

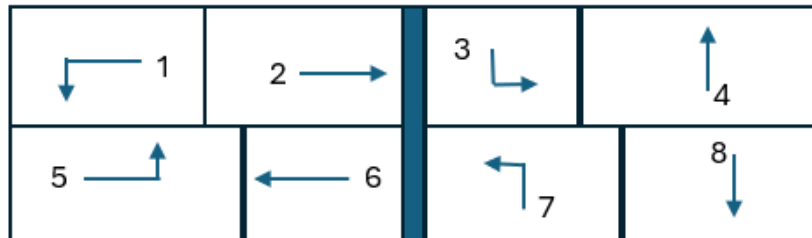


Figure 59: NEMA two ring barrier diagram

Chapter 5 and Chapter 6 describe action implementation for the dual ring barrier control in the current study is more complex as allowable actions may be limited at various points throughout the cycle or a phase. Example cases are illustrated in Figure 60. To implement dual ring control the terminology “active phase” to refer to the current phase and “committed phase” to refer to the next selected phase is adopted for each ring. The benefit of tracking active and committed phase is that the only situation these differ is when a

phase is yellow or red clear. Thus, yellow and red clear may be timed without specifically implementing these indications as separate actions. Rather than this approach most recent studies include yellow and red clearance as separate actions in the action set (Fan and Yang 2024; Long and Chung 2023b). However, this approach increases in the number of actions, potentially lengthening the training period.

Additionally invalid action masking (IAM) is used to force an agent to select the current active phase when required by signal control constraints. For example, when a phase is in its minimum green, yellow, or clearance intervals, the action must be to maintain the current active phase. IAM is also utilized to implement other constraints, such as implementing the ring barrier constraints. For DQN in Chapter 6, IAM was implemented by replacing q-values of invalid actions with large numbers. Similarly in PPO, logits corresponding to invalid actions are replaced with large negative numbers to preclude the selection of the corresponding action.

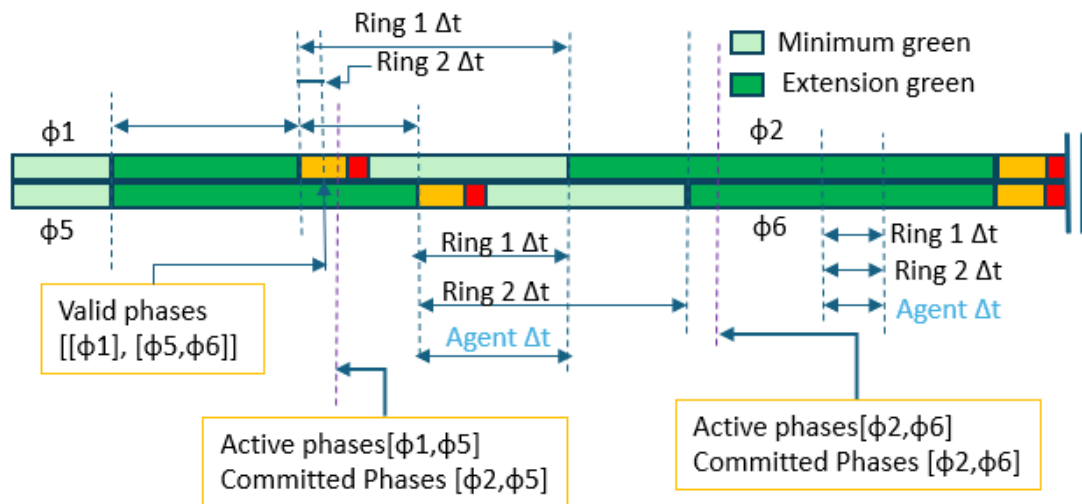


Figure 60: Action Implementation in two ring barrier configuration

Time step definition

In implementing RL actions in a simulation environment, there are two time steps being implemented, (1) the simulation time step and (2) the RL algorithm time step. The simulation time step defines the frequency of updating the simulation environment. In the simulation the time step is set to 0.1 second as this provides reasonable flow models and is the fidelity of field signal timing. The MARL algorithm time step defines the frequency of actions by the MARL agents. The size of the MARL algorithm time step (Δt) is not constant as defined next. The size of Δt determines the frequency of interaction with the simulation which largely determines the run time and the training duration as discussed in Chapter 5 and Chapter 6. It is thus critical to optimize Δt to improve run time efficiency.

To estimate the MARL time step Δt , the time step for each agent, Δt_{a_i} , must be estimated, where there are 1 to i agents in the MARL. To estimate Δt_{a_i} for an agent, it is necessary to first estimate the allowable time step for each agent ring $\Delta t_{a_i r_j}$, where there are j rings in agent i . Next, $\Delta t_{a_i r_j}$ for ring j depends on the state and duration of active phase and the committed phase in that ring. If the active phase is serving the clearance time, $\Delta t_{a_i r_j}$ for ring j is equal to the remaining clearance time plus the minimum green of the committed phase. If the active phase is running minimum green, $\Delta t_{a_i r_j}$ for ring j becomes the remaining minimum green duration. If the active green is running green extension, $\Delta t_{a_i r_j}$ is set to 1 second. The agent Δt_{a_i} is estimated as the minimum of the two ring time steps, i.e., $\min(\Delta t_{a_i r_1}, \Delta t_{a_i r_2})$. See example illustrations in Figure 60. Considering all agents in the environment, the overall Δt is selected as the minimum Δt_{a_i} for all agents, i.e., $\min(\Delta t_{a_1}, \Delta t_{a_2}, \dots, \Delta t_{a_i})$. The more agents there are in the environment the likely smaller the Δt and consequently the larger the run time. The minimum allowable Δt is 0.1 second which corresponds to the simulation resolution of 10 steps per second.

7.2.3.3 Reward definition

As shown in Equation 32, the local reward r_i at each intersection i is taken as the negative sum of the normalized delay for all vehicles V on all approaches of the intersection. In Equation 11, d_v^t is the individual vehicle delay at time t of each vehicle on the intersection approaches, while d_{max} is the maximum expected vehicle delay for that intersection. By using the ratio d_v^t/d_{max} , the delay is scaled between 0 and 1. The delay of each vehicle used in the computation of reward at the intersection excludes the delay encountered at previous intersections. Computing the reward with only delay encountered within the block avoids penalizing the side streets in favor of the main line traffic with accumulated delay from other intersections. As shown by Lee et al. (2022), the summation of normalized delay in the reward tends to outperform the direct summation of delay values.

$$r_i^t = - \sum_{v \in V} \frac{d_v^t}{d_{max}} \quad (32)$$

7.2.4 *Case study*

7.2.4.1 Test Network

The selected test network is a section of North Ave corridor in Midtown Atlanta. Figure 61 is a google map snapshot showing the test corridor and the surrounding area. The section consists of seven signalized intersections marked with red circles on the snapshot. Moving from west to east the included intersections are North Ave @ Tech pkwy NW, North Ave @ Techwood Dr NW, North Ave @ I-75/I-85 off-ramp, North Ave @ Spring St Nw, North

Ave @ W Peachtree St NW, North Ave @ Peachtree ST NE and North Ave @ Juniper St NE.

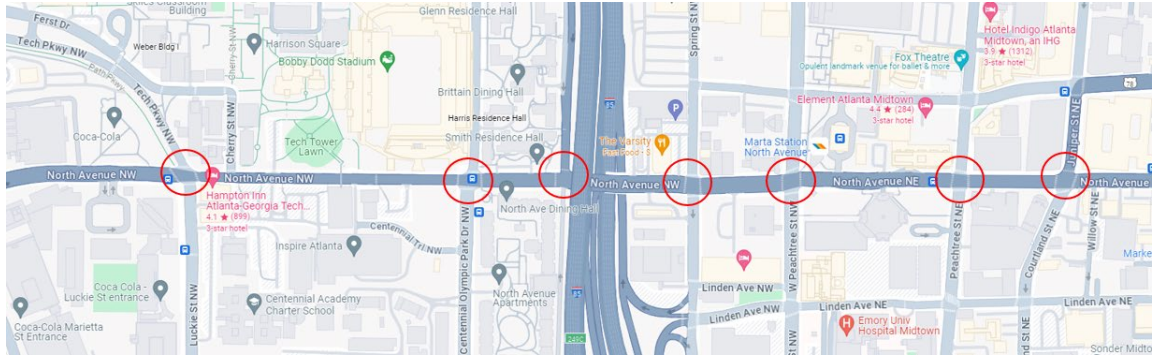


Figure 61: Test network extents in google maps

7.2.4.2 Data

The North Ave main street varies between a 4-lane and 6-lane facility, with turn bays at several intersections. Spring St. NW, W. Peachtree NW, and Juniper St. NE are one-way streets. All the seven intersections run on Qfree Advanced Traffic Controllers (ATC), utilizing the MAXTIME® traffic signal control software. For the signals, the Georgia Department of Transportation (GDOT), as part of an ongoing research project, provided detector layout maps and the signal timing databases. The signal timing databases include detector settings, controller settings, and signal timing plans. For this effort the PM peak was utilized. The existing corridor operates actuated-coordinated signal timing plans with a cycle length of 120 seconds in the PM peak. The coordinated movements are the North Ave. westbound and eastbound through movements, except for North Ave@ Spring St. NW where the Spring St. southbound movement is coordinated.

Figure 63 shows the PM peak (04:30-05:30 pm) turning movement counts at the seven intersections collected on April, 04th 2023. The data was collected semi-automatically with videos cameras. The volumes in the figure are in vehicles per hour occurring over the peak hour.

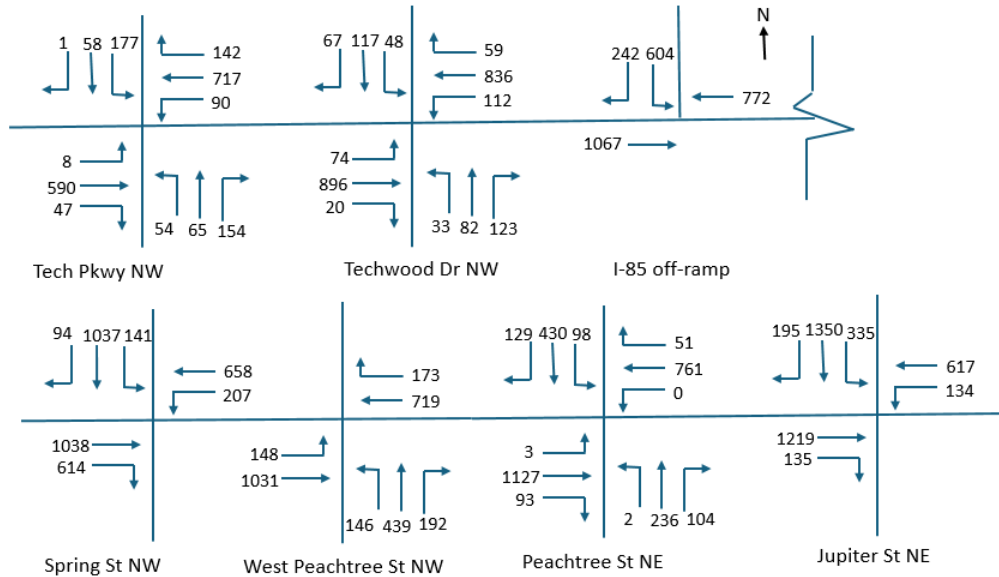


Figure 62: PM peak volumes

Additional trajectory data (i.e., vehicle location and speed) was drawn from drone videos as part of an ongoing GDOT research project. The drone data was collected on April 04th, 2023, in the PM peak from 3 pm to 7 pm. The data was collected with 6 drones to capture different angles and sections of the corridor. Field collected drone video data was processed using DataFromSky Software tools. The trajectory data was then utilized for simulation model calibration, discussed in section 7.2.4.4 below.

7.2.4.3 Model development

PTV-Vissim® is selected as the simulation environment. PTV-Vissim® is a microscopic, multi-modal traffic simulation software widely used in industry and research. The network geometry is drawn from google maps and confirmed through site visits. Model construction, quality control, and quality assurance follow the guidance laid out in Hunter et al. (2024). Figure 63 shows the network model in PTV-Vissim®. For the existing conditions simulation model, the MaxTime® emulator is utilized to control the intersections, forming the PTV-Vissim-MaxTime® SILs. The field controller MaxTime® databases with the field signal timing plans are loaded directly into the MaxTime® emulator which ensures that the existing conditions simulation runs the same signal timing plans and execution as the field. Connections between Vissim and MaxTime® are detailed in section 7.2.4.5.

The base model has fully actuated-coordinated signal timing plans with a cycle length of 120 seconds for the considered PM peak. The coordinated movements are the two main street through movements, westbound (WB) and (EB) except for North Ave @ Spring St NW where the cross street southbound (SB) movement is coordinated. For the considered PM peak, primary coordination is in the WB direction. Field signal timings are frequently fine-tuned using high resolution data from Automated Traffic Signal Performance Measures (ATSPMs). The signal timings are thus up to date and it may be safe to assume them close to optimal.

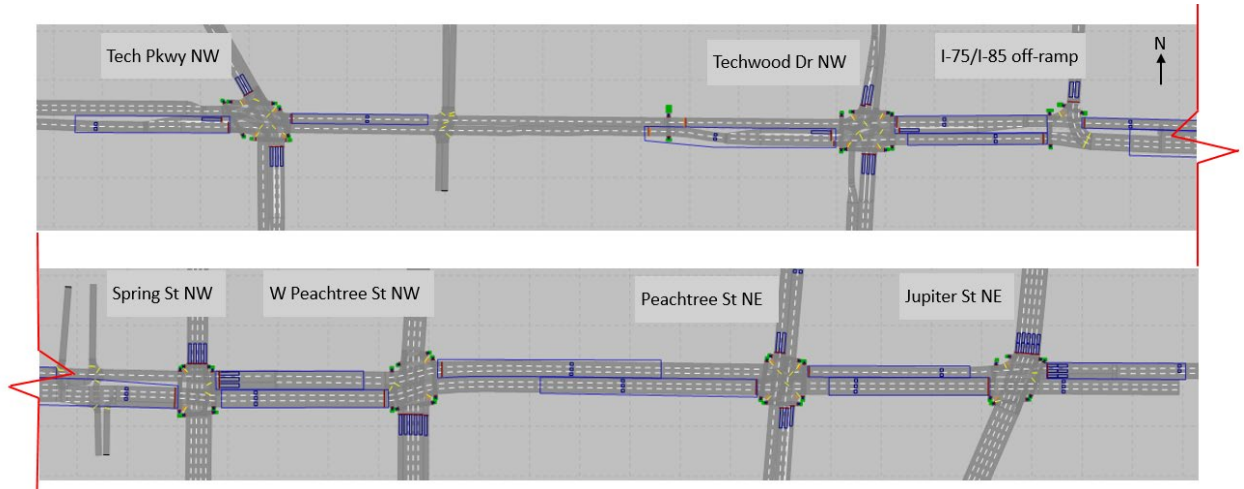


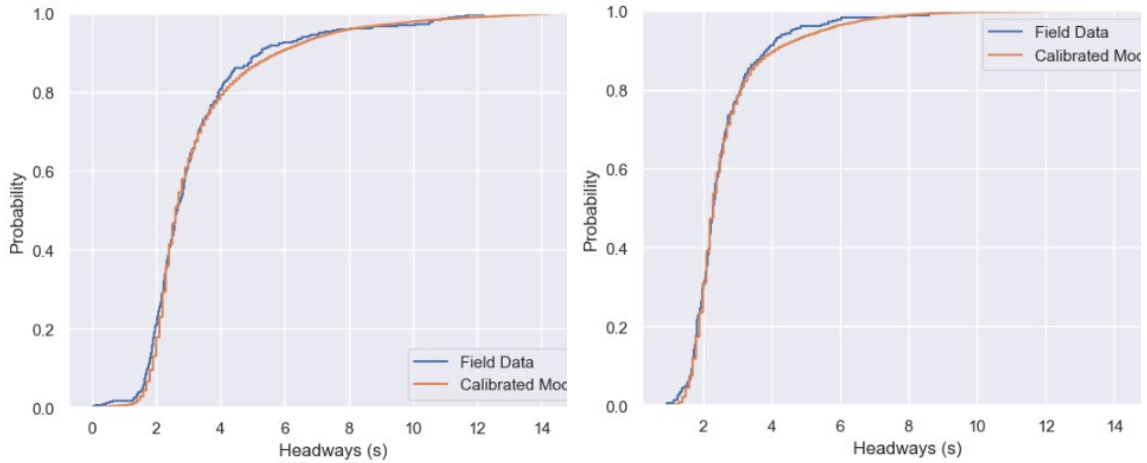
Figure 63: Network Model in VISSIM®

7.2.4.4 Model calibration

Model calibration involves adjusting the model parameters so that the model performance matches field conditions (Hunter et al. 2024). Model calibration was performed utilizing the vehicle trajectories drawn from the drone videos. The annotated trajectories were post processed to extract vehicle speeds, headways, accelerations, etc.

Hunter et al. (2024) was used as guidance for the PTV-Vissim® model calibration. For this effort two sets of parameters are calibrated, (1) speed distribution and (2) car following model parameters. The desired speed distribution parameters in PTV-Vissim® are adjusted such that that the desired speed distribution matches the field observed speed distribution. For car following, the parameters of Wiedemann 74 flow model, that is, a_x , b_{xAdd} , b_{xMult} , are adjusted such that the PTV-Vissim® simulated saturation headway distribution reasonably match field measured headway distribution.

Figure 64 shows a comparison of cumulative distribution function (CDF) of field-measured headways vs calibrated model headways at (a) North Ave @ Spring St and (b) North Ave @ I-75/I-85 off-ramp. The final calibrated model parameters are $a_x = 9$, $b_{xAdd} = 2.2$, and $b_{xMult} = 3.2$.



(a) North Ave @ Spring St

(b) North Ave @ I-75/I-85 Off-ramp

Figure 64: Field Vs calibrated model headways for (a) North Ave @ Spring St and (b) North Ave @ I-85 off-ramp

7.2.4.5 VISSIM®-MaxTime® SILs architecture

Figure 65 shows the base VISSIM®- MaxTime® SILs architecture. Each instance of MaxTime® is connected to VISSIM® with a unique transmission control protocol (TCP) port, VISSIM® controller number and a common VISSIM® IP as described below. MaxTime® takes vehicle detections from VISSIM® and uses the set signal timing plan to estimate the signal displays which are sent back to VISSIM® for display on signals. In essence, MaxTime® works as a virtual/external controller in VISSIM®. When the simulation is started in VISSIM® graphical user interface (GUI), VISSIM® waits for

MaxTime® to be started and establish the connection to the MaxTime® controller defined as an external controller in VISSIM®. To automate the starting of MaxTime® when VISSIM® simulation is started, an event-based script is embedded in VISSIM® that specifies the MaxTime® database and the connection settings of the MaxTime® instance to start. The script also facilitates simultaneous connection and running of multiple instances of MaxTime®, each for each intersection controller.

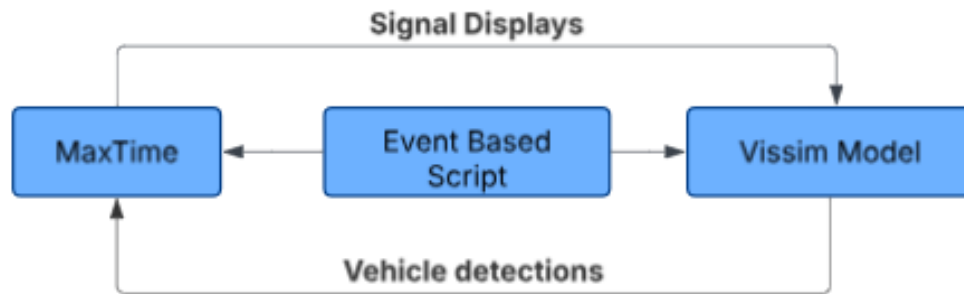


Figure 65: VISSIM®-MaxTime® SILs architecture

Connecting MaxTime® to VISSIM®

Qfree (2019b) provides a step-by-step guide on how to connect MaxTime® to VISSIM® and use MaxTime® as a virtual controller. In VISSIM® MaxTime® is specified as an external controller as shown in Figure 66. Three DLL files highlighted in Figure 66(a) are provided by Qfree/MaxTime® are specified in VISSIM®.

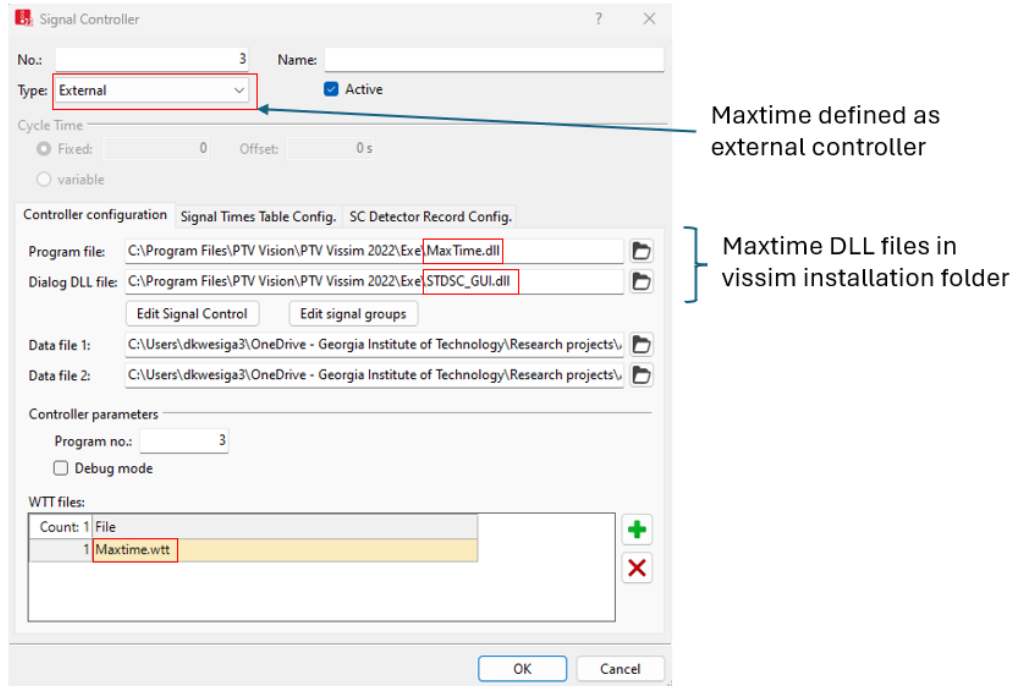


Figure 66: Specifying MaxTime® as an external controller in VISSIM® with the required files

A unique port number as highlighted in Figure 67(a) is defined for each intersection controller. And lastly the VISSIM® intersection signal groups to be controlled by MaxTime® are specified as shown in Figure 67 (b).

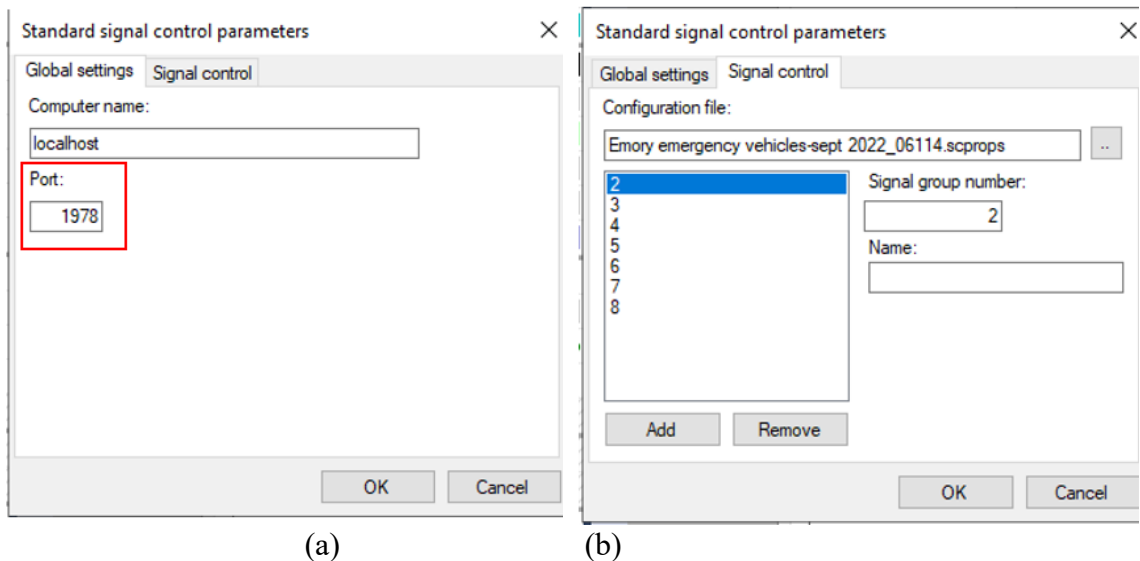


Figure 67: Defining MaxTime® (a) TCP port and (b) signal groups

Figure 68 shows Maxime’s VISSIM® connection settings. A VISSIM® IP of “127.0.0.1” is entered for all intersections. Other entries include the unique intersection TCP port number entered in VISSIM® and VISSIM® controller number entered as VISSIM® Id.

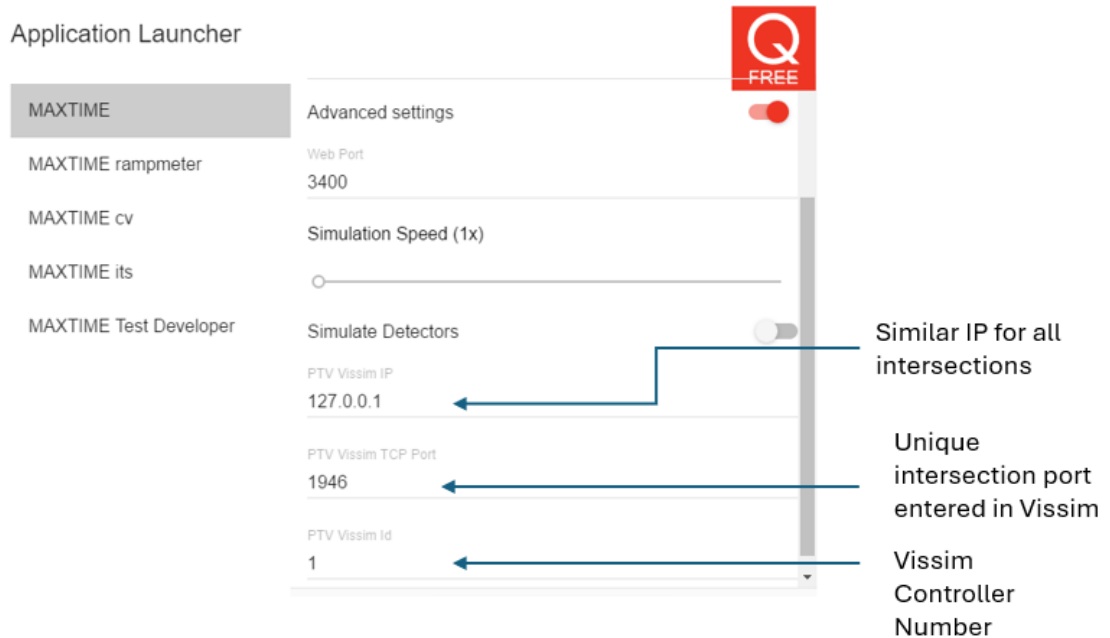


Figure 68: MaxTime®-VISSIM® connection settings in MaxTime®

7.2.4.6 RL and simulation architecture

Seven agents are formulated, one for each intersection. Figure 69 shows the interaction of agents with the PTV-Vissim® simulation environment. At each time step, each agent k pulls input from PTV-Vissim® including the local state S_k , and the global state S_{c_k} . Using the local state as input, each agent k selects action a_k , forming a set of global actions a_1 to a_7 .

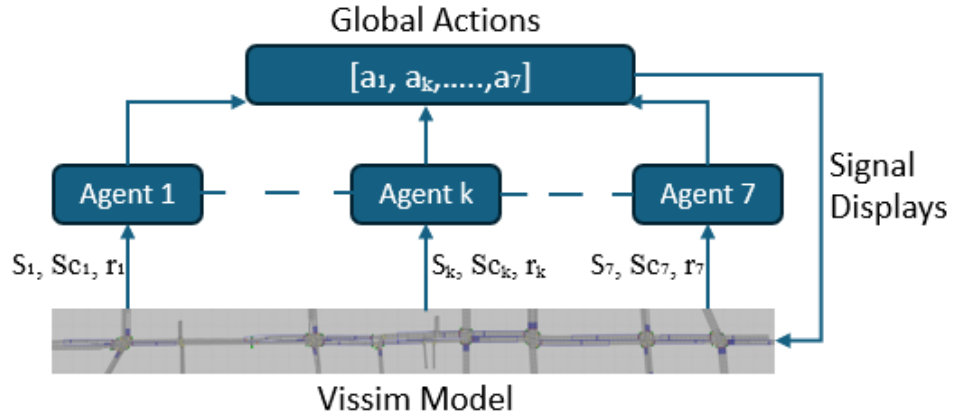


Figure 69: Interaction of agents with PTV-Vissim® Simulation environment

Figure 70 shows the detailed interaction between the agents (1-7) and PTV-Vissim® during the simulation run. The simulation is driven by event-based scripts as described in Chapter 5. Three python scripts are embedded in PTV-Vissim®, the “State & Reward Script”, the “Agent Script” and the “Model Runner Script”. The Model Runner Script (MRS) is the central script tracking the simulation time and algorithm time step, initiating actions of the former two scripts and providing signal timings to PTV-Vissim® for implementation.

At a timestep t , MRS passes the current local states (s_{1-7} to the agents’ actor networks (s_1 to agent 1 and so forth) to estimate the logits and actions. At the same time, MRS passes the global states Sck_{1-7} , one for each agent’s critic network to estimate the state values $V(Sck_{1-7})$ for that time step. With the selected actions, MRS estimates Δt as described earlier. The selected actions for each agent are converted into signal timings and sent to PTV-Vissim® for display for a duration of Δt . The minimum green and yellow and clearance times are set to the same values present in the field signal timing plans. After Δt , MRS requests the next local state, global state and reward for each agent from the State &

Reward Script. The State & Reward Script extracts vehicle records and signal states from PTV-Vissim® to formulate local state, global state, and reward functions as discussed in section 7.2.3 and Equations 19 through 32. A tuple of local state, global states, actions, rewards, logits and state values (s_{1-7} , $S_{c_{1-7}}$, a_{1-7} , logits_{1-7} , r_{1-7} , $V(S_{c_{1-7}})$) with 1 to 7 corresponding to for each agent are saved in the memory buffer for each agent for training at episode end.

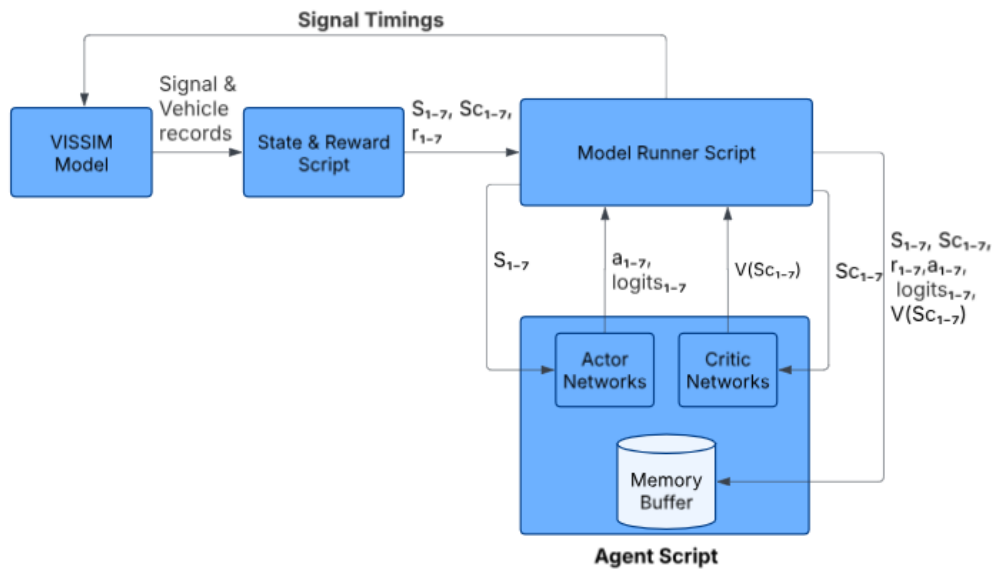


Figure 70:Agent-Vissim interaction during simulation run

7.2.4.7 Agent architecture, training and Hyperparameter selection

All the seven agents are formulated with the same architecture for the actor and critic networks. Preliminary training is performed to fine tune the hyperparameters. The finally selected actor network has two hidden layers with 64 and 128 neurons, a clip ratio of 0.2, and a learning rate of 0.0003. The finally selected critic network has three hidden layers with 128, 256 and 256 neurons, a learning rate of 0.001 and a discount factor of 0.99.

Figure 71 shows the training architecture for each agent k . Training is performed at the end of each episode using data archived only that episode (on-policy learning). For a full trajectory over the entire episode using archived rewards and state values, advantage function values are estimated following Equation 21. Using the archived actions, logits, actions and local states, the actor network estimates the importance sampling weights as per Equation 25. Using importance sampling weights and estimated advantage values the actor clipped loss is estimated per Equation 26. With the computed loss values, the parameters of the policy/actor network are updated using gradient ascent. Using the archived global state values and rewards, the critic loss is computed as per Equation 27. The parameters of the critic network are then update via gradient descent.

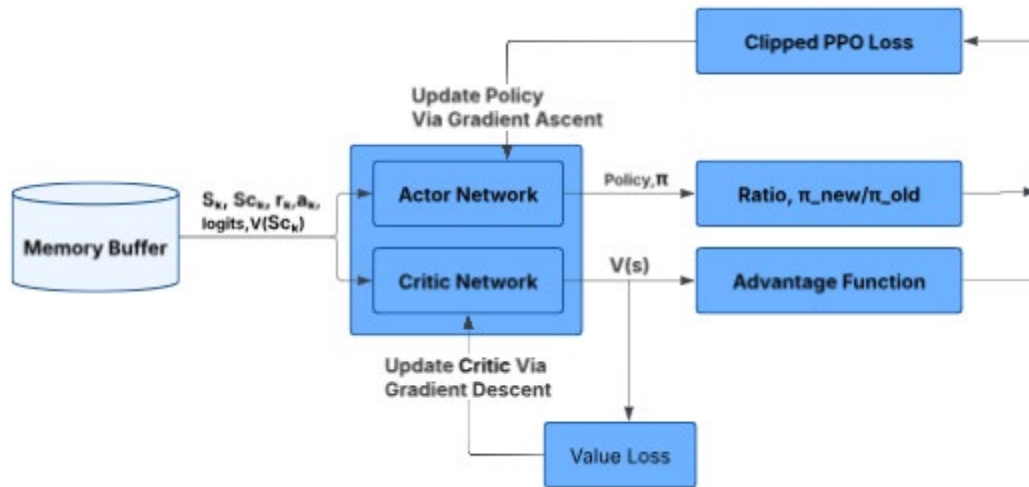


Figure 71: Algorithm architecture during training

7.2.4.8 Testing

Figure 72 shows the algorithm architecture during testing. As indicated, critic networks are only active during training but not during implementation/testing. During testing, agents

1-7, take local states (s_{1-7}) as input to predict the actions/signal timings that are implemented locally at each intersection. For each of the tests described in the section 7.3, ten replicate simulation runs are performed, each with a unique random seed. Each simulation run lasts for one hour with data collected only in the last 45 minutes, with the first 15 minutes of the hour taken as the warm-up period.

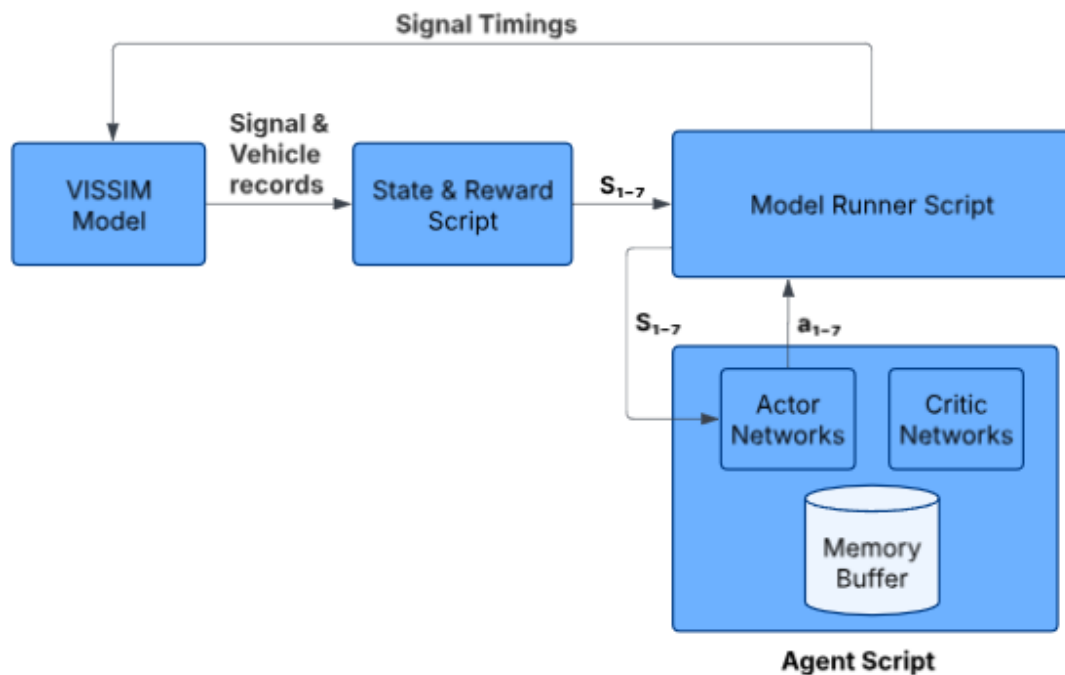


Figure 72: Algorithm architecture during testing

7.3 Results

7.3.1 Model training

Figure 73 shows the learning curves for the seven agents, one for each intersection. The x-axis plots the episode number while the y-axis plots the reward. Each data point is the average of rewards for all steps within the episode. Each episode lasts for 30 simulation

minutes. Simulation runs and model training are performed on an x64-based PC equipped with 12th Gen Intel(R) Core i9-12900, 2400 MHZ, 16 Core(s), 24 Logical Processor(s), 128 GB of RAM, Intel (R) UHD Graphics 770 GPU and Windows 11 operating system. It took approximately 60 hours to reach a fully trained model with the main limiting factor being VISSIM®'s runtime efficiency as fully discussed in Chapter 5.

The first four agents listed (North Ave. at Juniper St., Peachtree St. NE, W. Peachtree St., and Spring St. NW) converge within the first 300 episodes. North Ave at I-75/I-85 off-ramp also converges within the first 300 episodes except for continued variability which eventually dies down. These intersections each have a maximum of four phases and despite the higher or comparable traffic volumes at these intersections, convergence occurs significantly faster than at intersections with additional phases.

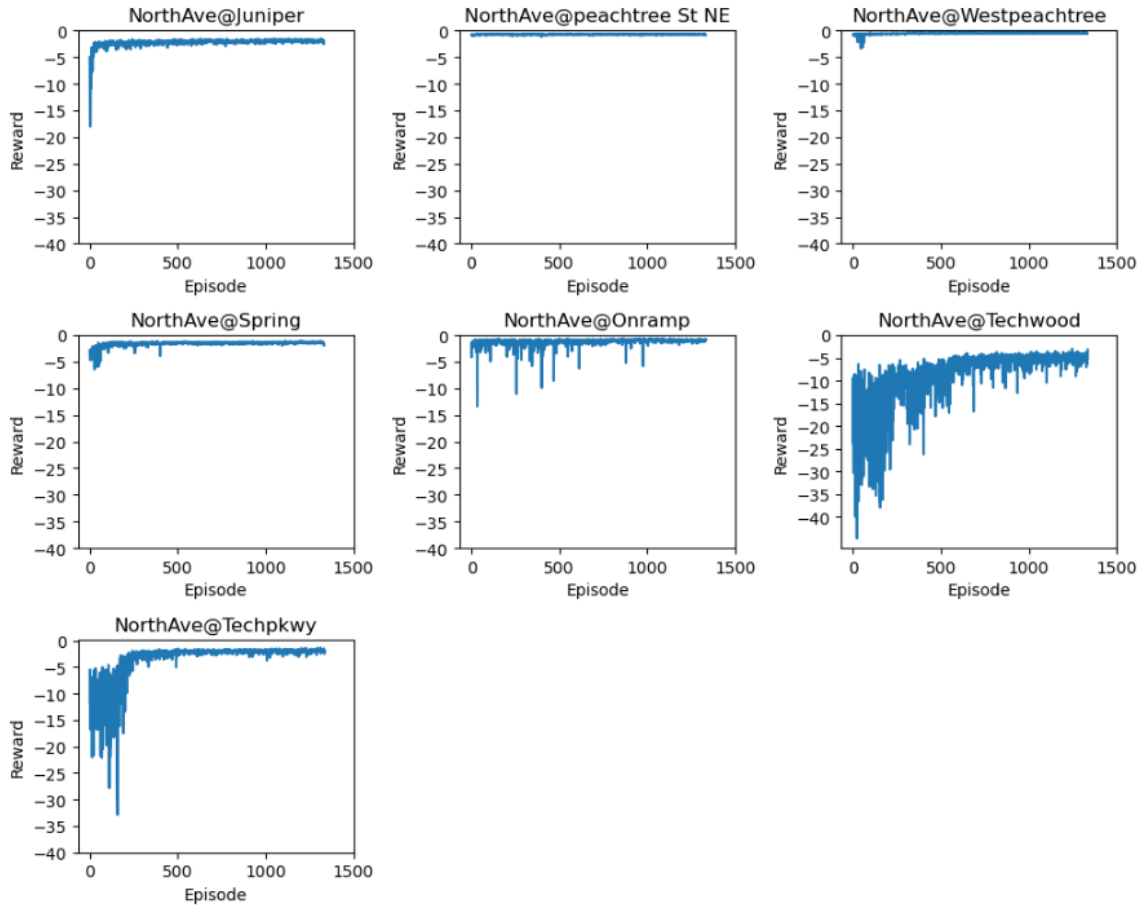


Figure 73: Learning Curves for different intersection agents

North Ave. at Tech Pkwy. NW (7 phases) converges after approximately 600 episodes while North Ave. at Techwood Dr. NW (8 phases) requires more than 1000 episodes to converge. The complexity of signal plans, including the number and sequence of phases appears to be the primary factor affecting the convergence which is intuitively reasonable as the agents have larger action spaces.

7.3.2 MA-PPO vs field implemented coordinated ASC

The performance of the trained MA-PPO is evaluated against the field implemented signal timing plans using delay and travel time as performance measures. Figure 74 shows a

comparison of main line movement travel time from the trained MA-PPO adaptive signal control and the field implemented coordinated ASC plans modeled using the MaxTime® emulator. The test volumes are the PM peak field volumes as shown in Figure 62. The two movements are defined on the main street, from end to end of the model, in the eastbound (EB) and westbound (WB) direction. EB travel time is measured from a point upstream of North Ave @ Techpkwy to a point downstream of North Ave @Juniper St NW while WB travel time is measured from a point upstream of North Ave @Juniper St NW to a point downstream of North Ave @ Techpkwy. Each box plot is formed from ten data points, the average travel time for all vehicles in each of the ten replicate simulation runs.

In the WB which is the primary coordination direction for ASC, both MA-PPO and ASC show very comparable performance, with MA-PPO showing slightly lower travel time. However, in the WB, MA-PPO performed significantly better than ASC. Compared to the field implemented plans, MA-PPO showed a travel time improvement of 24% in the WB direction. Coordinated ASC is typically optimized to favor progression in a primary coordination direction, with the secondary direction likely receiving poorer service. This approach may be sub-optimal, especially if there is a high proportion of traffic in the secondary coordination direction. This is the case on North Ave. corridor. The superiority of MA-PPO comes from the ability to optimize performance in both directions.

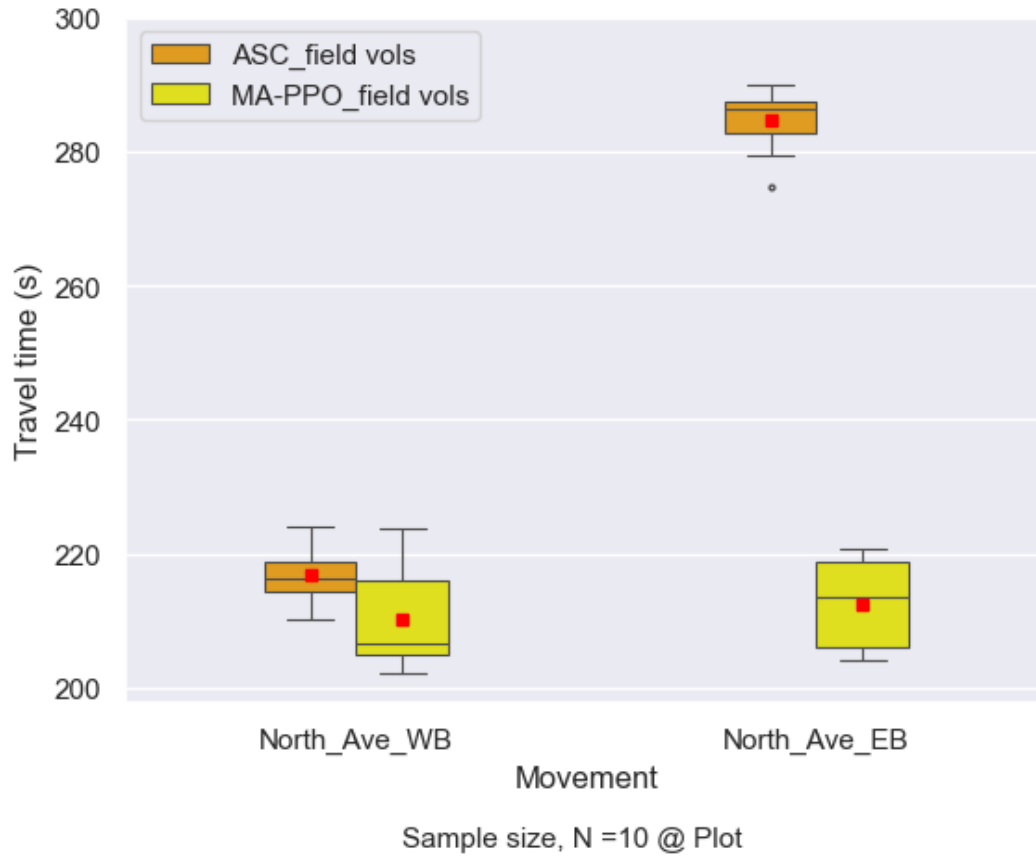


Figure 74: Main line movement travel time for MaxTime® and MA-PPO for field measured traffic volumes

Figure 75 shows the performance of the two control systems for the cross street through movements using delay as the performance measure. MA_PPO shows significantly less delay for all the measured cross street movements (only through movements shown here for brevity). This includes Spring St. southbound which is the coordinated movement in the field ASC signal timings. Thus, the selected reward system and the centralized training leads to better performance for both main and cross streets at all intersections compared to the field implemented ASC.

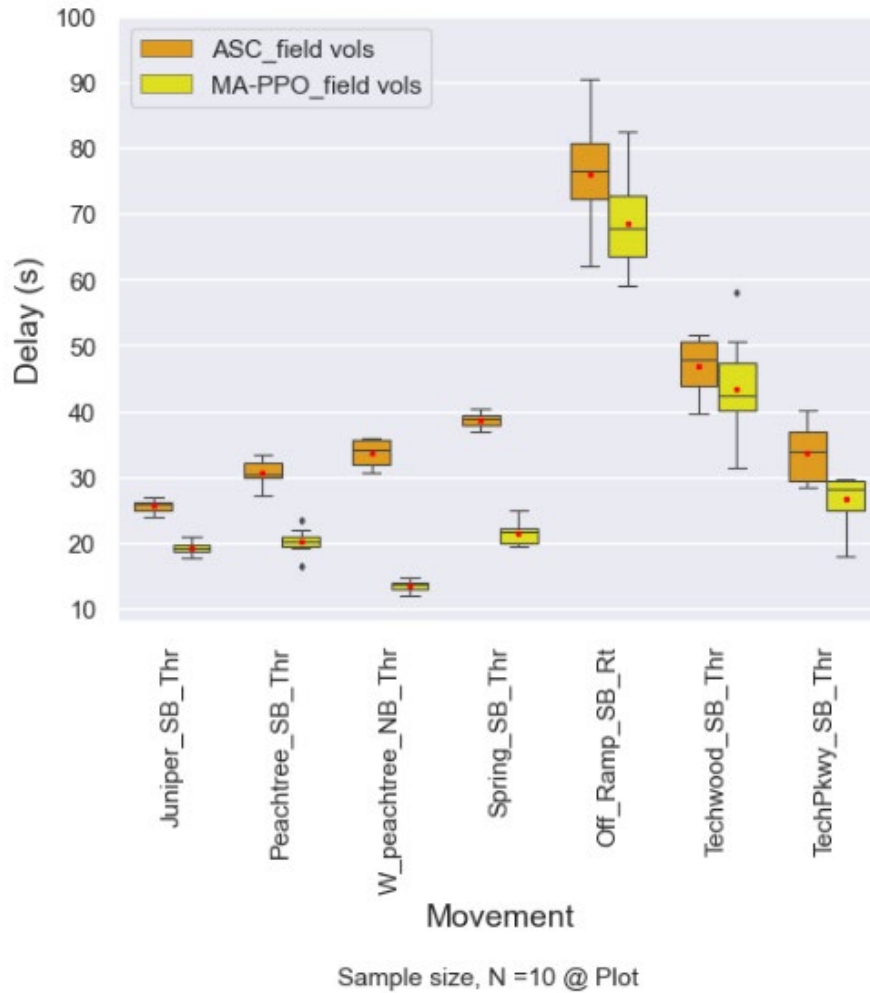


Figure 75: Cross street through movement Delays for MaxTime® and MA-PPO for field measured traffic volumes

7.3.3 Volume sensitivity and MA-PPO robustness

The above result is based on a single day field volume set and quite possibly may not reflect the typical day for which the ASC signal timings are optimized. To capture the impact of variability of traffic volumes and further test the robustness of the trained MA-PPO, sensitivity experiments are performed for two other volumes sets. Keeping the origin-destination (O-D) ratios fixed, the model entry volumes are adjusted (a) higher by 5% and

(b) lower by 10% from the field measured volumes of Figure 62. The limit of 5% higher is considered as some intersections are at or close to capacity for the field volumes.

Figure 76 shows the main street travel time from MA_PPO and ASC for the three volume levels. Travel time is measured from end to end of the model as described in 7.3.2 above. As expected, travel time increases as the volumes increase for both control systems. MA-PPO consistently outperforms ASC at all the three volume.

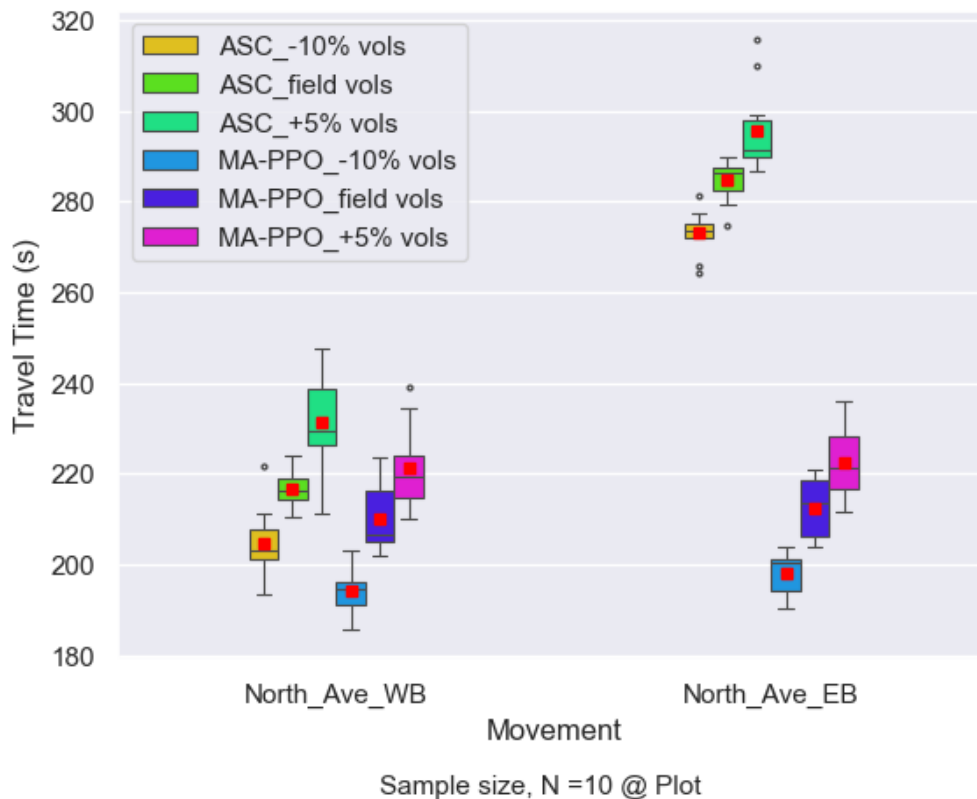


Figure 76: Main street Delay from MA-PPO vs coordinated ASC at three volume levels

Figure 77 shows a similar comparison for the cross streets, using delay as the performance measure. As with the North Ave. mainline, MA-PPO consistently outperforms ASC at the

three volume levels. The results of this sensitivity analysis demonstrate the robustness of the trained model to adapt to variations in traffic demand.

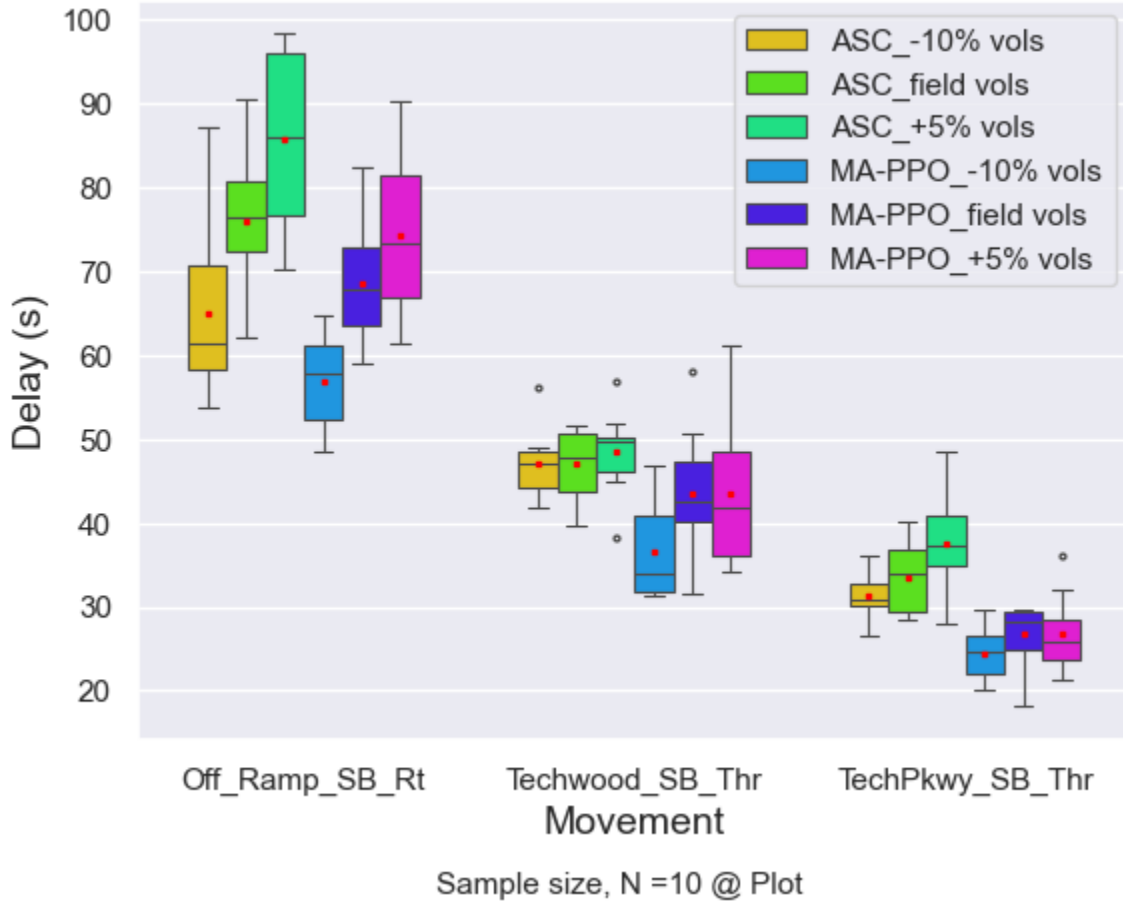


Figure 77: Cross street Delay from MA-PPO vs coordinated ASC at three volume levels

7.4 Corridor level Adaptive signal Control Summary

This study formulates a multi-agent proximal policy optimization algorithm to implement adaptive and coordinated traffic control along an arterial corridor. The formulated MA-PPO has centralized critic architecture in which each agent has an actor network that selects independent actions conditioning on local observations and a centralized critic that

estimates the agent's value function conditioning on global observations. All agents are formulated to allow selection and implementation of up to eight signal phases as commonly implemented in the field controllers. The formulated algorithm is tested on a simulated real-world corridor with seven intersections. The performance of the formulated MA-PPO adaptive control algorithm is compared with the field implemented ASC signal timing plans modeled using PTV-Vissim®-MaxTime® SILs.

The speed of convergence for each agent largely depended on the size of the action space which in turn depended on the number and sequence of signal phases. The intersections with 4 or fewer phases converged within 300 episodes each of 30 simulation minutes while the intersections with 7 and 8 phases required double to triple the number of episodes for convergence.

For the field measured traffic volumes, for all movements, the trained MA-PPO performed significantly better than the field implemented ASC signal timings. For the main street, MA-PPO optimizes performance in two directions compared to ASC which largely optimizes performance in the primary coordinated direction. Using travel time as the performance measure, the two control systems showed very comparable performance in the WB direction which is the PM peak direction and the primary coordination direction for ASC. In the EB direction, MA-PPO showed a 24% reduction in travel time compared to ASC. For cross streets, MA-PPO showed significantly reduced delay

To capture the variability of traffic volumes and further test the robustness of the trained MA-PPO, sensitivity experiments are performed for two other volumes sets, (a) 5% above and (b) 10% below the field measured volumes. For both main street movements and cross

streets movements, MA-PPO consistently outperformed ASC at the three volume levels. This demonstrated the robustness of the trained model to adapt to variations in traffic demand.

A key challenge of training of RL agents in a high-fidelity traffic microscopic simulation environment like VISSIM® is run time efficiency to reach hundreds to thousands of simulations runs required to fully train the agents. This study uses online serialized training for the formulated agents. Sixty hours were required to complete training of the agents. This limited the exploration and experimentation that could be achieved. Ongoing follow up studies are considering parallelized training architectures to reduce the required simulation run time.

CHAPTER 8 CORRIDOR LEVEL ADAPTIVE TSP BASED ON REINFORCEMENT LEARNING, CONNECTED VEHICLES AND SOFTWARE IN THE LOOP SIMULATION

8.1 Background

Chapter 7 formulates and tests adaptive signal control for a real-world corridor with actual field conditions. The current chapter extends the work in Chapter 7 by integrating TSP into the adaptive signal control algorithm developed for an entire arterial corridor. The objective is to enhance/extend TSP algorithms developed in the previous chapters to real-world corridor applications and test these under simulated field conditions and constraints and evaluate their scalability.

As with the RL-based traffic control algorithm developed in the previous chapter, the TSP algorithm formulated in this chapter has a centralized critic PPO architecture. Each intersection is controlled by a single agent selecting local actions conditioning on local observations. The agents are centrally trained to learn to select coordinated TSP strategies across multiple intersections, as discussed in Chapter 6. The TSP agents need to learn both how to control general traffic and prioritize the bus. The incorporation of the bus results in learning becoming much more challenging compared the Chapter 7 formulations.

In this chapter, two implementations of RL-based TSP are considered: (1) general traffic is controlled by the same RL agents even in absence of the bus (i.e., agents are always active), and (2) general traffic control is provided by the ASC, with the TSP agents only active when a bus is present. In both cases the same trained TSP agents are utilized. The

ASC is modeled using the MaxTime® emulator, forming the VISSIM®-MaxTime® SILs as described in the previous chapter. In evaluating the performance of developed TSP algorithms, CTSP with a CI-CO detector system implemented in VISSIM®-MaxTime® SILs is taken as the baseline. This system is configured with “optimal” TSP settings and parameters based on Chapter 3 and Chapter 4.

8.2 Methodology

This section presents the formulation, training, and testing of the corridor level TSP agents. In testing the algorithms, the same test corridor developed Chapter 7 is utilized with modifications for bus routes and bus stops.

8.2.1 Definition of state, action and reward functions

8.2.1.1 State definition

For each agent actor network, local general traffic state (veh_{state}) and local signal state (Sig_{state}) are maintained as defined in the Chapter 7. In addition to the general traffic, the local bus state is defined. It is assumed that each intersection agent perceives the bus when the bus is within two blocks of the intersection. Thus, the bus state (bus_{state}) is defined within two blocks of the intersection while veh_{state} is defined within one block. Buses travel end-to-end in both directions of the corridor. See Figure 78 for an example of the zone considered for the local bus state of intersection D. As seen in Chapter 4, the arrival profile of the bus at downstream intersections is largely determined by the dwell-time at upstream bus stops. On most bus routes, it is likely that there is at least one bus stop within two blocks. Extending the intersection bus perception distance beyond two blocks will likely

not accrue any meaningful benefit due to the variability in ETA introduced by the bus stops. Thus, for this effort, to improve efficiency in training and testing the bus state was limited to two blocks; however, future efforts will test this assumption.

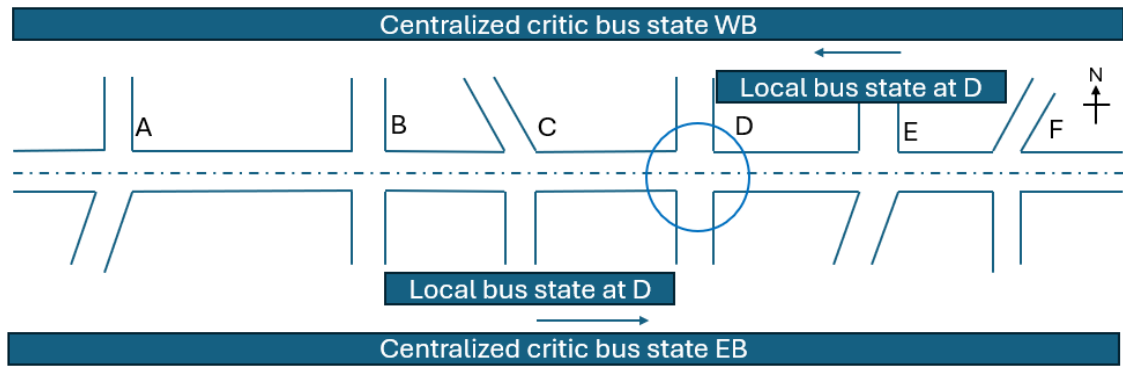


Figure 78: Local and global bus state definition

In Chapter 5 and Chapter 6, bus state was defined by dividing the communication range into cells of 25 ft with a cell occupied by a bus populated with a “1”, else populated with “0”. The bus position and bus speed vectors were each of size equal to the length of communication zone divided by 25. In this chapter, to reduce this state dimension and improve scalability, a maximum of four buses is considered in each direction at any point in time, i.e., it is expected that no more than four buses in each direction shall be within the corridor extents considered. Except on the most heavily travelled transit corridors, this should be a realistic assumption. Bus state is still defined with two parameters, the bus position and bus speed except that here the bus position and bus speed vectors are each limited to a size of four in each direction. The size of four comes from the defined maximum number of buses in each direction. Additionally, the bus position is now defined as the distance traveled from the entry point into the network to the current position scaled by the total length of the route on which the bus is traveling. Equation 33 and Equation 34

show and example local bus position vector and local bus speed vector, respectively, for the westbound two blocks approaching intersection i . Equation 35 and Equation 36 show an example local bus position vector and local bus speed vector, respectively, for the Eastbound two blocks approaching intersection i . The equations illustrate an example of one westbound bus present within the two blocks of intersection i , at position 0.5 (halfway through the corridor), with a speed of 36 mph. Concatenating the position and speed vectors in both directions as shown in Equation 37 gives rise to the overall local bus state of size sixteen.

$$Bus_{pos_WB_i} = [0.5, 0, 0, 0] \quad (33)$$

$$Bus_{speed_WB_i} = [36, 0, 0, 0] \quad (34)$$

$$Bus_{pos_EB_i} = [0, 0, 0, 0] \quad (35)$$

$$Bus_{speed_EB_i} = [0, 0, 0, 0] \quad (36)$$

$$Local\ Bus_{state_i} = Bus_{pos_WB_i} + Bus_{speed_WB_i} + Bus_{pos_EB_i} + Bus_{speed_EB_i} \quad (37)$$

As shown in Equation 38, the local bus vector is concatenated with traffic and signal states to form the overall local state at the intersection.

$$S_i = Veh_{state_i} + Sig_{state_i} + Local\ Bus_{state_i} \quad (38)$$

The centralized critic state for each agent defined in Chapter 7 is updated to include the global bus state which consists of the positions and speeds of all buses in both directions for the entire length of the corridor. See Figure 78 for an illustration of centralized critic state extents. Again, limiting the expected total number of buses to four in each direction

gives rise to an overall global bus state of size sixteen. See Equation 37 with the terms defined in Equation 33 through Equation 36.

$$Global\ Bus_{pos_WB} = [0.5, 0.8, 0, 0] \quad (33)$$

$$Global\ Bus_{speed_WB} = [36, 32, 0, 0] \quad (34)$$

$$Global\ Bus_{pos_EB} = [0.2, 0.6, 0.9, 0] \quad (35)$$

$$Global\ Bus_{speed_EB_i} = [10, 0, 21, 0] \quad (36)$$

$$Global\ Bus_{state} = [Global\ Bus_{pos_WB} + Global\ Bus_{speed_WB}] + [Global\ Bus_{pos_EB} + Global\ Bus_{speed_EB}] \quad (37)$$

As shown in Equation 38, the overall centralized critic for each agent is defined by concatenating the local states including local traffic and local bus states together with the global bus state. The vector is ordered such that the local state of the subject intersection, i , is listed first and the remaining local states follow in order according to the direction of travel.

$$Sc_i = \{ [Veh_{state_i} + Sig_{state_i} + Local\ Bus_{state_i}] + [Veh_{state_1} + Sig_{state_1} + Local\ Bus_{state_1}] + \dots + [Veh_{state_n} + Sig_{state_n} + Local\ Bus_{state_n}] + Global\ Bus_{state} \} \quad (38)$$

8.2.1.2 Action definition

The action definition remains unchanged from the previous chapter which involves the selection of the next phase from a set of available phases. See section 7.2.3.2.

8.2.1.3 Reward definition.

The TSP agent needs to learn to balance tradeoffs between providing priority to the bus and avoiding excessive deterioration of LOS for conflicting traffic movements. The agent needs to learn both how to control general traffic and prioritize the bus. Learning is much more challenging compared to Chapter 7 formulations.

Training is divided into two stages, with two different reward definitions. The first stage focuses on training the agents' general traffic control, as performed in chapter 7 but now with bus flow parameters in the state and reward functions. The local reward function for agent i , $r1_i$ for stage 1, is defined as shown in Equation 39. The first term is the summation of normalized vehicle delay as defined in the previous chapter, while the second term is the summation of normalized bus delay d_b for all buses present, with the buses ($b \in B$), weighted by a priority factor (PF). The PF defines the weight to be given to the bus in this first stage of training. Agents are trained to convergence with this reward before advancing to stage 2 of the training.

$$r1_i^t = - \sum_{v \in V} \frac{d_v^t}{d_{max}} - \sum_{b \in B} \frac{PF * d_b}{d_{max}} \quad (39)$$

Stage 2 of the training starts after the convergence of stage 1. The local reward function for agent i , $r2_i$ for stage 2, is defined as shown in Equation 40 with all terms already defined. In this reward function, the second term is the summation of delay for all buses present in the local state. Compared to stage 1 reward, the bus term in stage 2 reward heavily penalizes bus delay and incentivizes bus priority. Priority factor (PF2) is used to define the desired level of priority. The first term penalizes the excessive deterioration of LOS for general

traffic. In absence of the bus, the second term reduces to zero and only the first term becomes impactful.

$$r2_i^t = - \sum_{v \in V} \frac{d_v^t}{d_{max}} - \sum_{b \in B} PF2 * d_b \quad (40)$$

8.2.2 Test corridor Model

The same VISSIM® model developed and calibrated in Chapter 7 is used, with the addition of bus routes and bus stops. A screen shot of VISSIM® model with bus locations is shown in Figure 79. On North Ave., there are two MARTA bus routes, Route 50 and Route 51, both starting/terminating at the North Ave. bus station, which is at the intersection of North Ave. and W. Peachtree St. NW. The existing routes go through five intersections of the test corridor starting from North Ave@ W. Peachtree St. NW to North Ave@ Tech Pkwy. NW in both the EB and WB directions. In this study, the bus routes are adjusted/extended to also go through last two East most intersections, North Ave@ Peachtree St. NE and North Ave@ Juniper St. NE. Extending the bus routes enables testing TSP performance over the entire length of the TSP corridor.

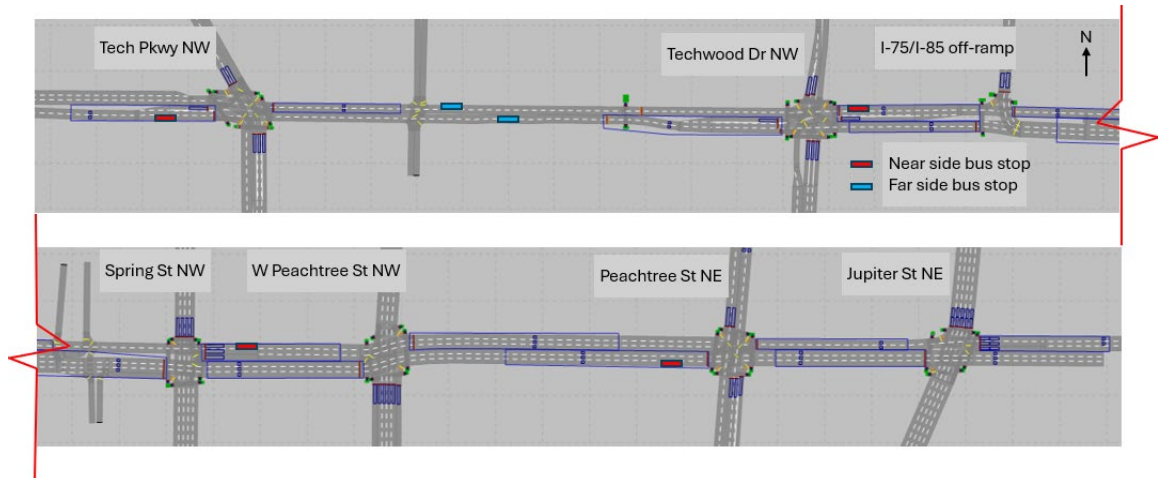


Figure 79: VISSIM® model for the test corridor including bus routes and bus stops

As described previously in 5.2.3.11, bus routes and bus stops are modeled using PT Line and PT stop modules in VISSIM®. In the field, some blocks have more than one bus stop in each direction. For simplicity, the number of bus stops per block is restricted to one in each direction. In the developed network model. In the final model, there are three bus stops on the WB route of which two are near-side and one is a far-side bus stop. There are also three bus stops on the EB route of which two are near-side and one is a far-side bus stop. The positions of bus stops by type are shown in Figure 79. Dwell-time is modeled based on the dwell-time data presented in Chapter 4. As previously discussed, dwell-time is entered in VISSIM® as a CDF from which VISSIM® samples dwell-time values to assign each bus at that stop, including zero dwell-time in which case the stop is skipped. Figure 80 shows the dwell-time CDF entered in VISSIM® for all bus stops.

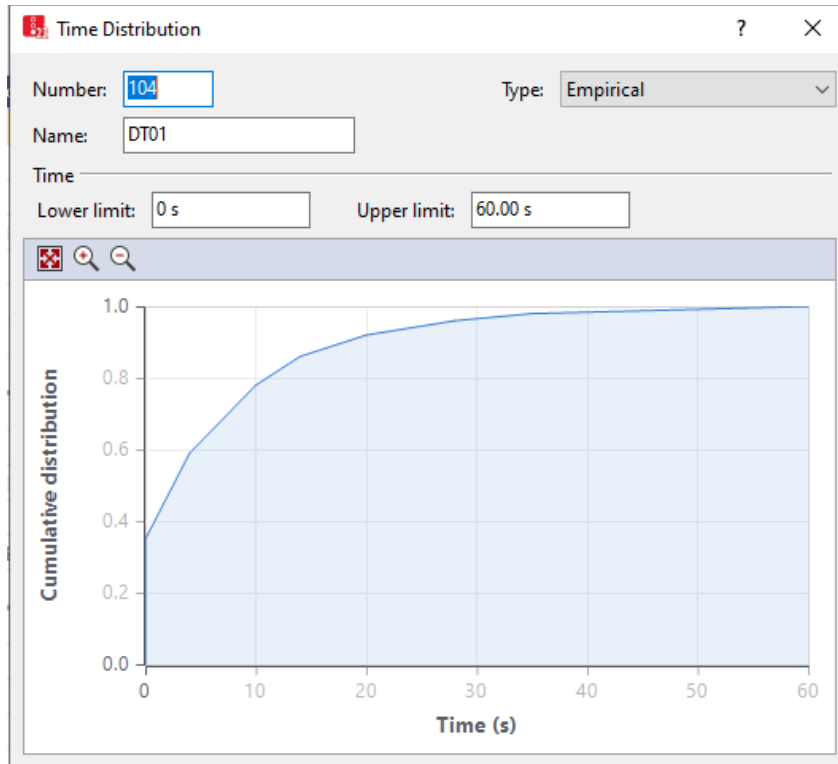


Figure 80: Dwell-time distribution in VISSIM® for all bus stops

8.2.3 RL and simulation architecture

RL and simulation architecture during training remains the same as shown in Figure 71. The same hyperparameters and network architectures are as described in section 7.2.4.7. Each episode lasts for 45 minutes with a total of 5 buses entering the network in each direction (EB and WB). Compared to the training in the previous chapter, here episode length is increased by 15minutes to allow more bus samples in the training data.

8.2.4 TSP Implementation architectures & test scenarios

The sections above discussed the formulation and training of corridor level TSP agents. As stated, with the trained agents, two implementations of RL-based TSP are considered; (1) RL-based adaptive TSP where the general traffic is controlled by the same RL agents even

in absence of the bus. This system is termed as the RL_RL TSP system, and (2) RL-based adaptive TSP with the traffic control when a bus is not presence being ASC using the MaxTime® emulator, forming the VISSIM®-MaxTime® SILs as described in the Chapter 7. This system is termed as the RL_ASC TSP system. In the RL_ASC TSP system, each TSP agent is activated to take over control from MaxTime® when the bus enters within two blocks of the intersection and the agent is deactivated when the bus clears the intersection. With RL agents trained under the CTDE paradigm as described in section 6.2.1 , the agents are triggered independent of each other. The trained agents seek globally optimal actions conditioning only on local agent observations.

8.2.4.1 RL-RL TSP System

In Chapter 5 and Chapter 6, separate TSP agents were trained for general traffic control and TSP implementation with TSP agents only triggered when the bus enters the CV communication range of the intersection. In this chapter, a different approach is taken for RL_RL TSP where the same set of agents is trained to implement both tasks.

During implementation, each trained agent selects and implements local actions based on its local state. Each agent perceives the bus when the bus enters within two blocks of the intersection and selects actions to prioritize the bus and control general traffic. The no TSP (NoTSP) version of this system is implemented treating the bus as a regular passenger vehicle in the state input (no bus detection). When the TSP system is deactivated the entries into the bus state remain “0” even where a bus is present. The system operates without knowledge of bus presence and therefore no special attention to the bus.

8.2.4.2 RL_ASC TSP system

This section describes the implementation of RL-based adaptive TSP with background control (i.e., traffic control when a bus is not present) running ASC. Figure 81 shows the implemented VISSIM®-MaxTime® SILs architecture with RL-based TSP. The system consists of interactions between the VISSIM® network model, the trained RL agent, and the MaxTime® emulator. In absence of the bus, MaxTime® takes vehicle detections (see red dotted line in Figure 81) from VISSIM® and selects signal indications to implement using the selected signal timing plan. The selected signal indications are sent to VISSIM® for display. The connections/communication between VISSIM® and MaxTime® are described in detail in section 7.2.4.5.

The Model Runner Script (MRS) keeps track of all bus trajectories in VISSIM®. This is like high resolution AVL data. When a bus is within 2 blocks of the intersection, the intersection RL agent takes control from MaxTime®. Given the decentralized execution, any agent can run its intersection independent of whether other agents are active. When the RL agent comes online, the RL agent takes the local state as input and outputs actions which are converted into signal indications to implement.

The composition of local state described earlier with Equation 38 consists of vehicle state, signal state, and bus state. Vehicle records, including bus flow parameters, are extracted from VISSIM® to formulate the vehicle and bus state. During training where MaxTime® is not in the loop, signal state data forming part of the state is extracted from VISSIM® as shown in Figure 70. During execution with MaxTime® in the loop, signal state is extracted from MaxTime®. This enables implementation of the committed phase concept introduced

in section 7.2.3.2 of the previous chapter. That is, at the end of phase green, the controller (MaxTime®) commits to a phase to serve next, after the clearance period of the current phase. This committed (i.e., next) phase cannot be changed until the end of its minimum green. Thus, if a RL agent takes over from MaxTime® during a clearance interval, the agent is constrained to implement the phase already committed to by MaxTime®, up to the end of the phase minimum green. The IAM algorithm therefore needs to have information about the committed phase that is not yet displayed in VISSIM®. For this reason, therefore, signal state data is extracted from MaxTime® not VISSIM®. See Figure 81. The details of extracting signal state from MaxTime® are discussed in the next section.

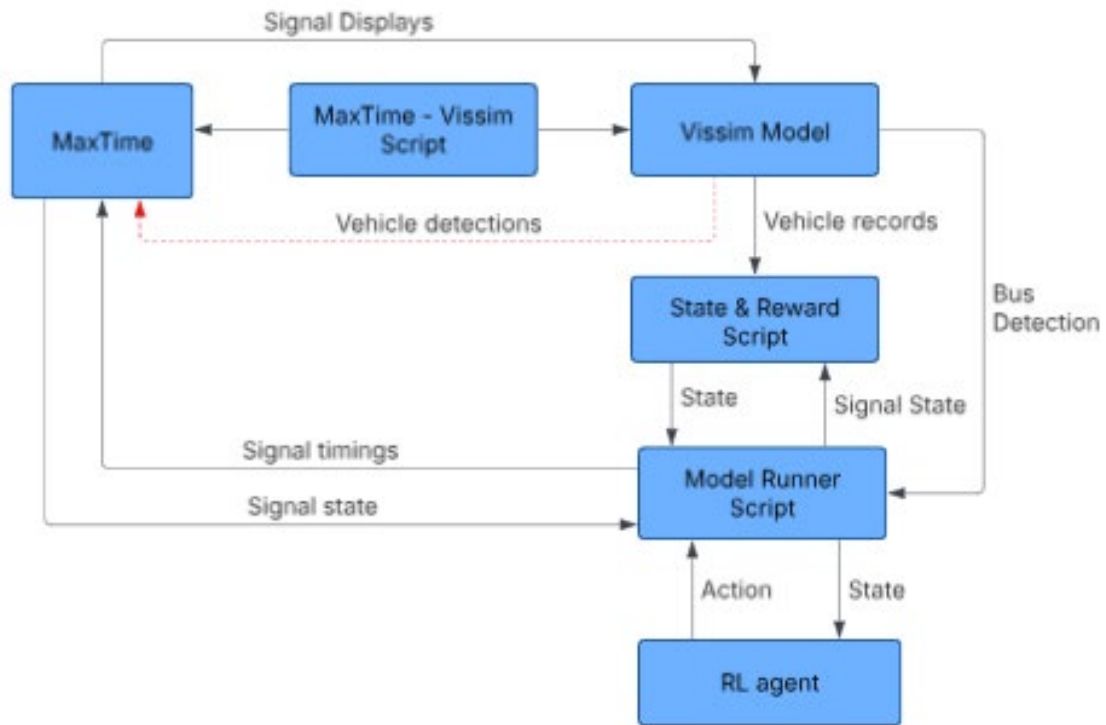


Figure 81: MaxTime®-VISSIM® SILs architecture with RL-TSP

The state and reward script processes data from both VISSIM® and MaxTime® to formulate the agent’s local state as described in section 8.2.1.1. The RL agent takes this

state as input and predicts the actions to implement. The MRS converts actions into signal indications and sends them to MaxTime® for implementation. The selected signal timings are implemented in MaxTime® using NTCIP commands of hold, forceoff, and omit. The following section discusses communication between MaxTime® and the RL agents and the implementation of RL actions.

(a) RL-agent -MaxTime® Communication

The MRS communicates (sending and receiving) with MaxTime® through NTCIP messages sent by means of Simple Network Management Protocol (SNMP). The communication module consists of the SNMP manager, the SNMP Agent, and the SNMP management information base (MIB) Browser as shown in Figure 82.

The SNMP manager is based on a python SNMP library/module called PySNMP. The SNMP Agent is a MaxTime® software unit/module that handles communication with the SNMP manager. The SNMP manager connects to each instance (one for each intersection) of the MaxTime® (SNMP agent) through a unique user datagram protocol (UDP) port. NTCIP messages are exchanged between the SNMP manager and the SNMP agent using two protocol data units (PDU): (1) GetRequest which retrieves variable status from the agent to the manager, and (2) SetRequest which writes to objects (changes values of variables) in the agent.

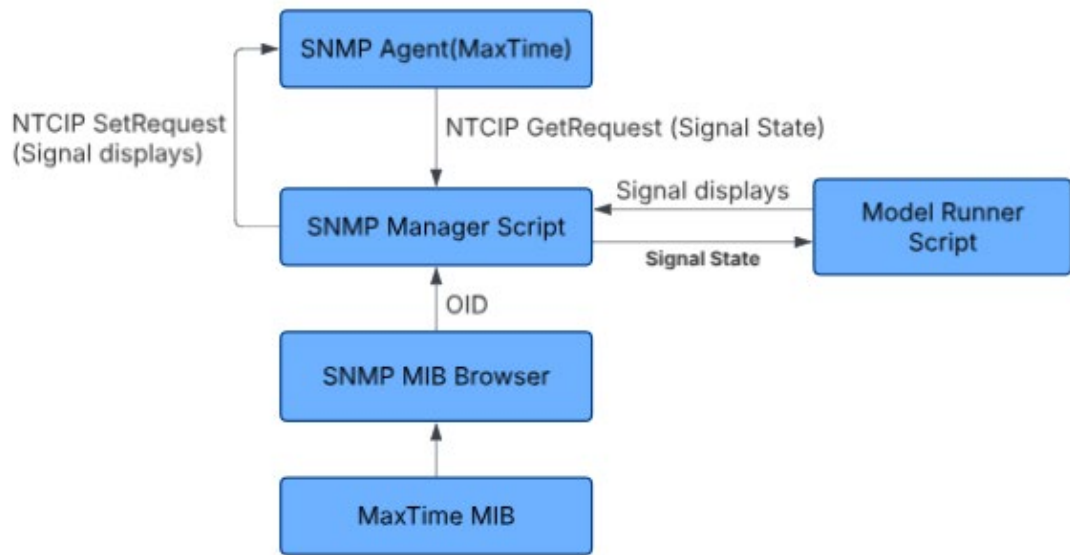


Figure 82: MaxTime® and RL-agent communication module

Figure 83 shows the NTCIP settings in MaxTime® GUI for a given intersection. The unique UDP port identifies the MaxTime® instance to the SNMP Manager. The community Name and Access Level specify the user access rights which can be read only or read and write.

NTCIP Settings

Administrator Community Name: Administrator

UDP Port: 163

TCP Port: 0

NTCIP Enable: Enabled

NTCIP Community Names

	Community Name	Access Level
1	dkwesiga3	Read-Write
2	public	Read-Write
3	public	Read-Write
4	public	Read-Write
5	public	Read-Write
6	public	Read-Write
7	public	Read-Write
8	public	Read-Write
9	public	Read-Write
10	public	Read-Write

Figure 83: MaxTime® NTCIP settings

Qfree provides open access to MaxTime®’s MIBs which contains a hierarchical database of all MaxTime® objects identified by unique objective identifiers (OID). Controller objects that are normally editable allow read-write access while status objects allow read only access. The “iReasoning MIB Browser” produced by “iReasoning Networks” is utilized to extract the relevant OIDs. Figure 84 shows a screenshot of MaxTime® ASC MIB opened in iReasoning MIB Browser. Figure 84 (a) shows the standard ASC MIB based on NTCIP 1202 while Figure 84 (b) shows MaxTime® specific ASC MIB. The vendor (Qfree) specific MIB has extra objects that may not be available in the standard MIB. Both MIBs were utilized.

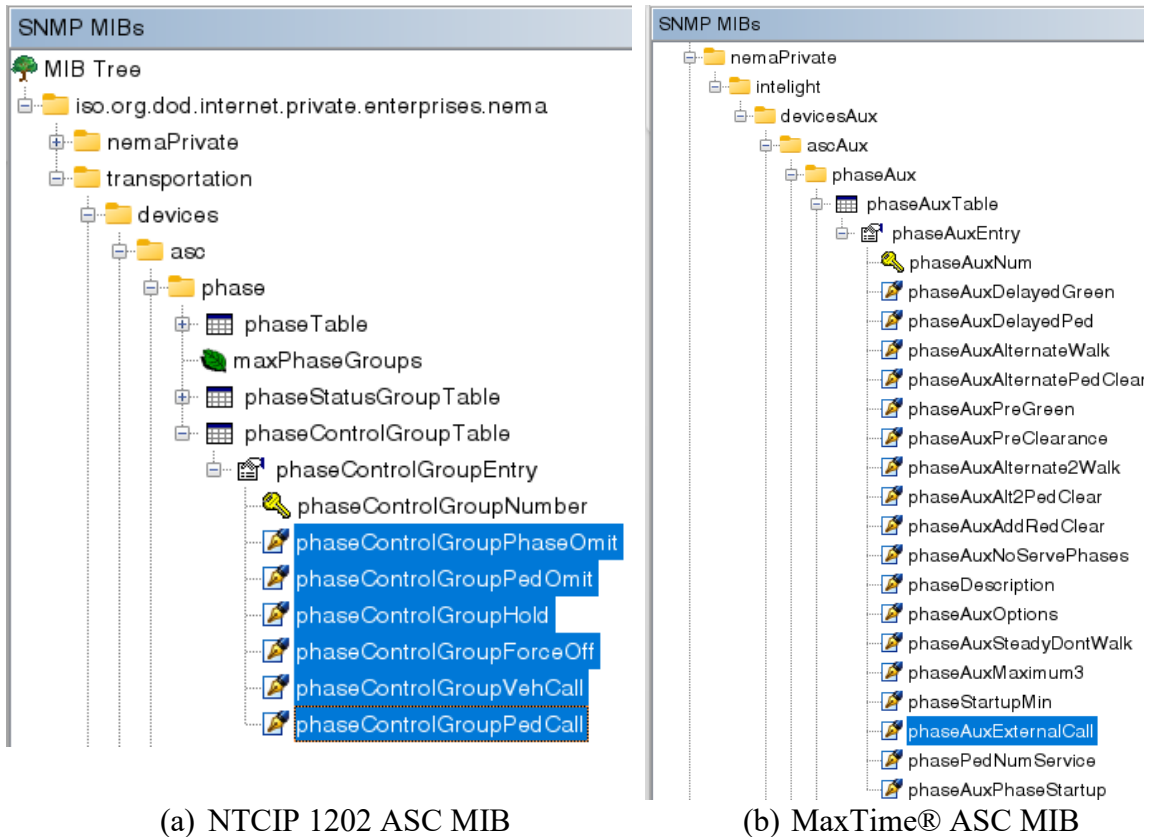


Figure 84: MaxTime® MIB Screenshot

(b) RL action implementation in MaxTime®

As discussed, the selected RL action may be to keep the current phase or switch to another phase in the phasing sequence. In this implementation MaxTime is always running and the NTCIP calls are used to force the controller to switch phases requested by the RL rather than the ASC response to the detector calls. Thus, with MaxTime® running background traffic control, selected RL actions are implemented with force off, hold, and omit NTCIP commands. See the highlighted objects in Figure 84 (a). Phase hold keeps the current phase running, Phase forceoff terminates the current phase, phase omit skips the specified phase(s). The task here is to implement the RL selected phase sequence and duration in MaxTime® which then passes the displays to VISSIM®. To have a fixed background

sequence in MaxTime®, when the RL agent comes online, all the enabled phases are set to minimum recall as shown in Figure 85. Setting minimum recall on all phases ensures a fixed phasing sequence, as entered in the ring barrier sequence. Using this method, when a phase is forced off or omitted, the next phase to be implemented by MaxTime® is known. Additionally, active coordination mode is set to manual free to run the controller in free mode (no coordination for the subject intersection/controller). Running the controller in free mode allows more flexibility, with no of fixed cycle and offsets constraints. This signal plan with minimum phase recalls and manual free mode is referred to as “special phasing plan” here after. After the bus exits the intersections, the minimum recalls are removed, and the original coordination pattern is re-activated. All this is implemented online through NTCIP commands.

The NTCIP commands of forceoff, hold, and omit must conform to the constraints of the ring-barrier diagram, otherwise MaxTime® will not implement the commands. For instance, a running phase cannot be forced off /terminated before its minimum green has been served. Starting from Chapter 7, the RL action selection and implementation is based on the dual-ring barrier with the constraints implemented by tracking active phase and committed phase in each ring and using the IAM masking to preclude the selection of invalid phases, see section 7.2.3.2. The constraints include minimum green, yellow and red clearance, double barrier crossing, etc. In the implementation of the signal control logic developed in Chapter 7 and in the implementation of RL_RL TSP in section 8.2.4.1 above, the selected actions conform to the constraints of the ring-barrier diagram, although the actions are implemented by “ContrByCOM” command without the conformity check by the controller. In the implementation of RL_ASC TSP, exact same actions are selected as

in RL_RL TSP but the selected actions are implemented through NTCIP commands, and a conformity check is done by MaxTime® before executing the commands.

Global Phase Recalls		Phase	1	2	3	4	5	6	7	8
<input checked="" type="checkbox"/>	Min		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
<input type="checkbox"/>	Max		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
<input type="checkbox"/>	Ped		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
<input type="checkbox"/>	Act. Walk Rest		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

Figure 85: Special background phasing sequence in MaxTime®

The flowchart of Figure 86 shows the logic for implementation of RL actions in MaxTime®. The process is initiated by MRS detecting bus entry and activating the RL agent. MRS tracks the simulation time and estimates the RL agent algorithm time step length as discussed in section 7.2.3.2. Decision time is the simulation time at which the agent needs to select and implement a new action. At the decision time the agent estimates the next phase of the signal. If the next phase is equal to the current phase, the NTCIP hold command is used to hold the current phase in green. If the next phase is different from the current phase, NTCIP force off command is sent to terminate the current phase. In the latter case an extra check is made to see if there are any intermediate phases between the current phase and the next phase. If there are intermediate phases, an NTCIP omit command is sent to omit the intermediate phases. MaxTime® will automatically display yellow and red clearances when a phase is forced off. The computation of Δt was discussed earlier in detail in section 7.2.3.2.

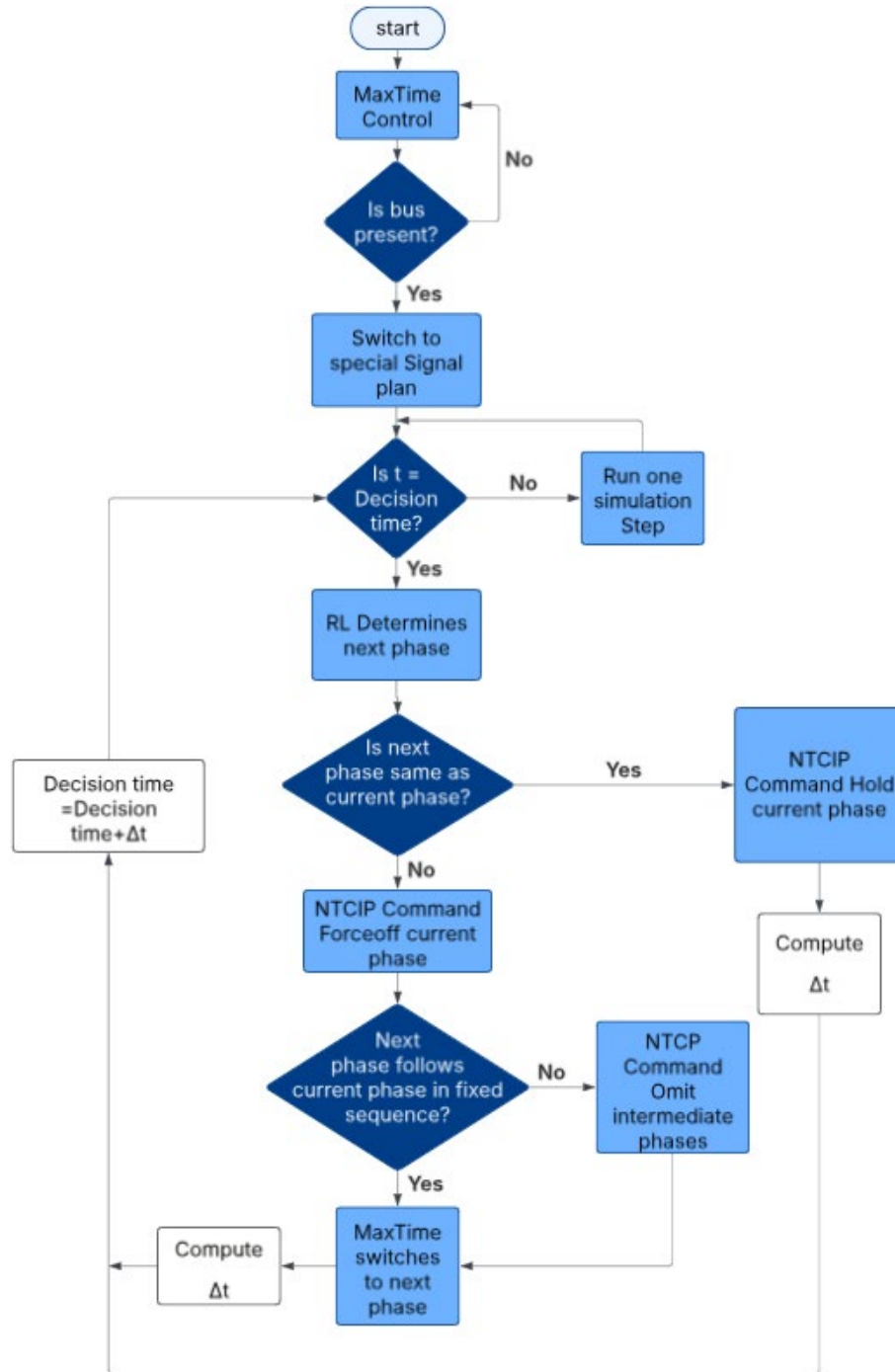


Figure 86: Implementation of RL agent actions in MaxTime®

The NoTSP version of this system is the same as CI-CO NoTSP system with MaxTime® inbuilt TSP algorithm disabled. There is no special attention paid to the approaching bus and thus the TSP agent is not triggered.

8.2.4.3 CI-CO TSP with inbuilt MaxTime® TSP algorithm.

CI-CO/CTSP implemented with the in-built MaxTime® TSP algorithm is used as the baseline to evaluate the two RL-based TSP systems. The “optimal” settings and parameters of the CI-CO, including ETA, GE and EG limits, and reservice/lockout times are selected based on the Chapter 3 and Chapter 4.

For blocks without bus stops or with near-side bus stops, check-in detectors are placed at the block start point. For blocks with far-side bus stops, check-in detectors are placed immediately after the bus stop. Preliminary runs are performed to select ETA values to set for each intersection as described in Chapter 3 and Chapter 4. The cycle length is allowed to vary by at most 20% during TSP service and in transition back to coordination. To have a “fairer” comparison with RL-based TSP, the algorithm is allowed to skip all non-priority phases whenever possible. Reservice time/lockout time (time before granting another TSP request) is set to one cycle (120 seconds). Priority minimums for all phases are set to the phase minimum green times.

Figure 87 (a) show priority detector settings while Figure 87 (a) shows the priority phase settings in MaxTime®. Qfree (2019a) provides step by step instructions on how to configure TSP in MaxTime®.

Prioritor and Preempt Detectors

Detector Plan 1 Show All Detectors Show All Parameters

Detector	Description	Low Call	High Call	Low Num	High Num	Lead/Trail
20		None	Prioritor	0	2	Lead Request
21		None	Prioritor	0	2	Lead Release
22		None	Prioritor	0	1	Lead Request
23		None	Prioritor	0	1	Lead Release

(a) Priority detector settings

Prioritor Phase Settings

Show All Prioritors

Prioritor	Enabled	Priority	Priority Phases	Skip Phases	Skip Ped	Delay Time	Arrival Time	Max Presence
1	Enabled	1	2	4,5		0	6	100
2	Enabled	1	6	4,5		0	6	100

(b) Priority phase settings

Figure 87: TSP settings in MaxTime®

The NoTSP case of CI_CO involves running the model with TSP disabled in the priority settings of MaxTime®. As indicated, this becomes exactly the same as the RL_ASC NoTSP case.

8.3 Results

8.3.1 Model training

As discussed already, training occurs in two stages with the first stage mainly focused on teaching the agents general traffic control and the second stage focused on teaching the agents to how to find the optimal TSP strategy. Figure 88 shows the learning curves for the seven agents, one for each intersection during the first stage of training. The x-axis plots the episode number while the y-axis plots the reward. Each data point is the average of rewards for all steps within the episode. Similar convergence behavior is seen as in the previous chapter with convergence largely depending on the number of phases and

complexity of the phasing sequence. North Ave.@ Tech Pkwy. and North Ave@ Techwood Dr. NW, with 7 and 8 phases, respectively, needed up to 600 episodes to converge. After convergence of the first stage of training, the second stage of training followed with the modified reward function as discussed in the methodology section of the chapter.

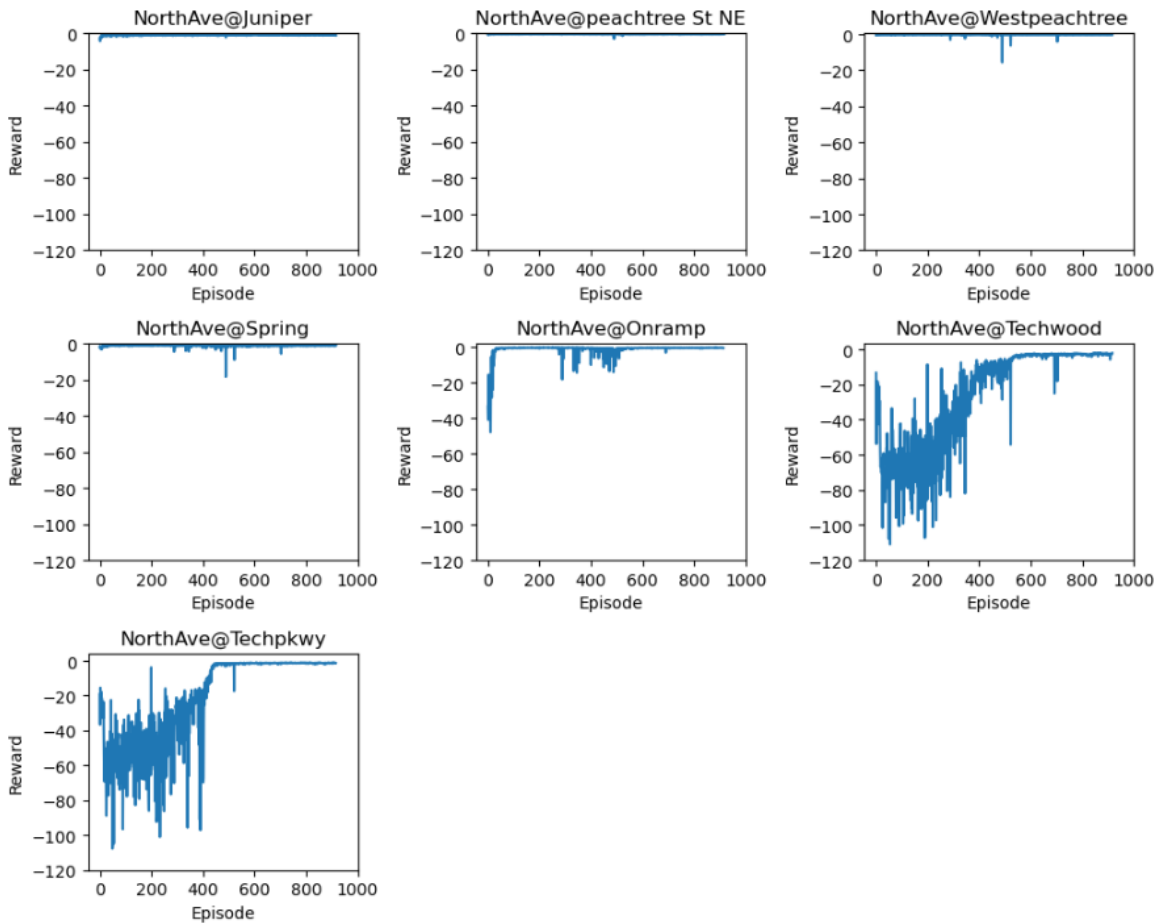


Figure 88: Learning curves for the first stage of training

Figure 89 shows the evolution of bus delay during training. The x-axis plots the episode number while the y-axis plots the bus delay. Each data point is the average delay of buses in the episode. Each episode lasts for 45 minutes with a total of five buses entering the

network in each direction (EB and WB). The delay is measured across the entire length of the corridor from upstream of North Ave@ Juniper St. NE to the downstream of North Ave@ Tech Pkwy. for the WB bus route and from upstream of North Ave@ Tech Pkwy. to the downstream of North Ave@ Juniper St. NE for the EB bus route. The figure shows the evolution of bus delay for the two stages of training. It is observed that for both directions, bus delay flattens after the convergence of the first stage of training which is at approximately 600 episodes. The second stage of training starts from 900 episodes. With the new reward function, bus delays progressively drop again converging after an additional 1000 episodes. The final average bus delays is approximately 100 seconds for EB and approximately 130 seconds for the WB direction.

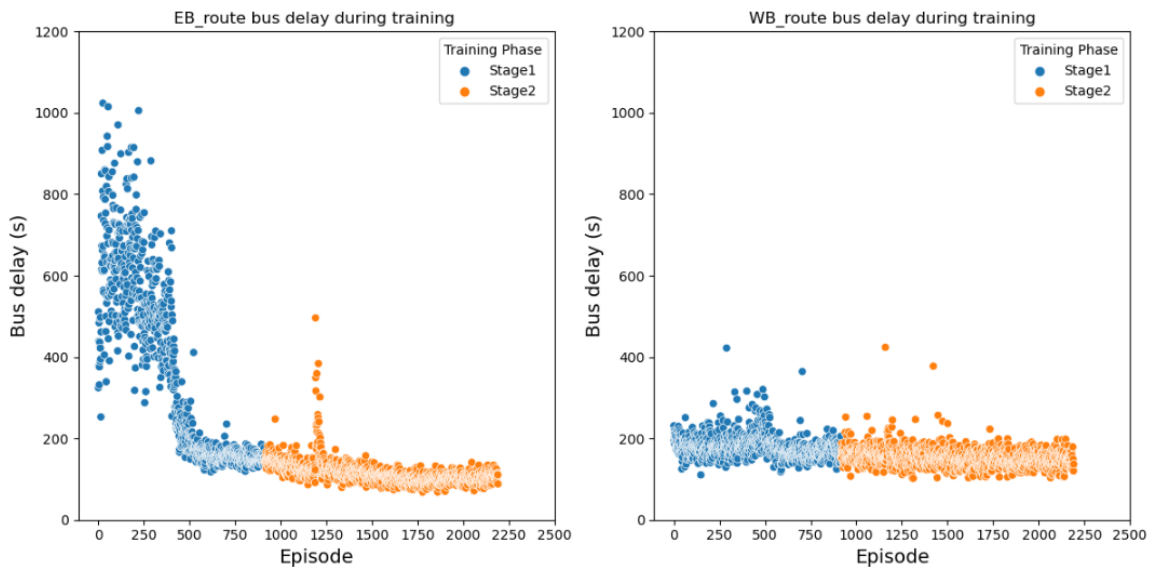


Figure 89: Bus delay during training

The above result of two stage training prompted a search for more formalized techniques of training agents/machines for tasks of progressively increasing complexity. One such approach is curriculum learning first formalized by Bengio et al. (2009a). Inspired by the

human learning process of starting with basic concepts during infancy and structured curricula in schools, curriculum learning trains agents/machines starting with simpler aspects of the task or easier subtasks and then gradually increases the complexity. The technique has shown potential for improved speed of convergence (Bengio et al. 2009b; Morales 2020; Soviany et al. 2022). Advanced traffic signal control strategies such as priority and preemption seem to be prime candidates for curriculum learning where agents could start by learning simplified signal plans, such as single ring control, then progress to dual ring control, and finally integrate special timing plans of priority and preemption. The idea of curriculum learning is reserved for future studies and shall be further mentioned in the recommendations section of the next chapter.

8.3.2 *Bus travel time*

Figure 90 shows the bus travel time for the three TSP systems, CI_CO, RL_ASC, and RL_RL TSP. As described previously, “NorthAve_WB” stands for North Ave. westbound, which is the primary coordination direction for ASC in the considered PM peak. “NorthAve_EB” stands for North Ave. eastbound. Bus travel time is measured from end-to-end of the model for both directions as previously described. Bus travel time includes dwell-time at the bus stops.

In the WB direction, CI_CO TSP does not provide any significant change in bus travel time compared its NoTSP case (CI_CO_NoTSP). It is likely that for the prevailing combination of traffic volumes, background signal timings (optimized for WB), bus stop locations, and dwell-time, that CI_CO TSP cannot provide further improvements to the bus travel time. RL_ASC_TSP shows a 12% decrease in travel time compared to its NoTSP

case (RL_ASC_NoTSP). RL_RL_TSP shows a 11.5% reduction in travel time compared to its NoTSP case (RL_RL_NoTSP). In the EB direction, CI_CO_TSP shows 18% decrease in travel time, RL_ASC_TSP shows a 31% decrease in travel time, and RL_RL_TSP shows a 25% decrease in travel time, all compared to their respective NoTSP cases.

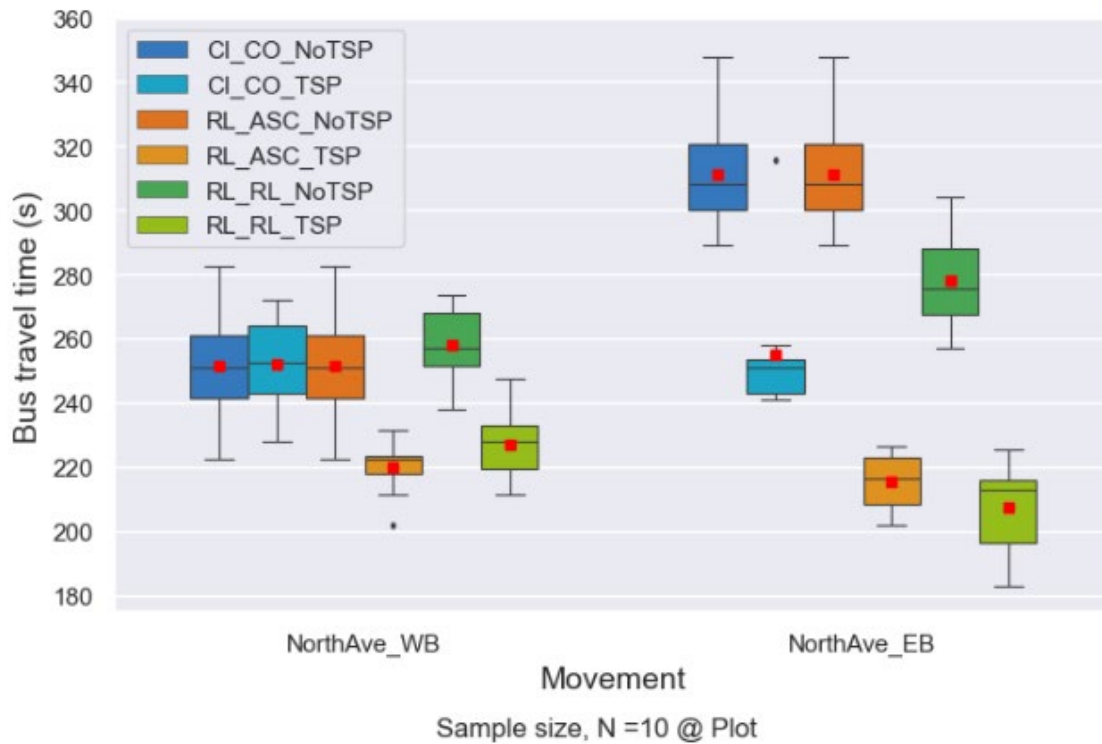


Figure 90: Bus travel time for different TSP systems

Both RL_ASC and RL_RL_TSP significantly outperform CI_CO. In WB direction, RL_ASC TSP system shows 12% lower bus travel with CI_CO TSP system. RL_RL TSP system shows 9% lower bus travel with CI_CO TSP system. In EB direction, RL_ASC_TSP system shows 16% lower bus travel with CI_CO TSP system. RL_RL TSP system shows 18% lower bus travel with CI_CO TSP system. The superior performance of RL-based TSP is attributed to the ability of these RL algorithms to (1) select coordinated

TSP strategies across multiple intersections and (2) adaptively select TSP strategies based on the current traffic state compared to CI-CO TSP that relies on fixed TSP parameters set in the controller and (3) ability to respond to variation of dwell-time by adjusting the TSP strategy depending on whether the bus stops or does not stop.

8.3.3 *General traffic delay*

Analysis is performed to assess the changes in delay experienced by general traffic with TSP implementation. Changes in delay are measured for five minutes after bus entry into the block. Five minutes is chosen following the results of Chapter 3. The same bus entry times are used for TSP and NoTSP cases. Delay impacts are measured for both cross street and main street traffic. It is hypothesised that side street traffic might experience more delay after TSP as more time is assigned to the priority movements on the main street. For CI-CO_TSP and RL_ASC_TSP, main street traffic may experience more delay due to interruptions of coordination and the required transitions back to coordination; however, it is also possible that the main street traffic may experience lower delay due to increased green time while serving the bus.

8.3.3.1 Cross street traffic

Figure 91 shows the cross-street traffic delay with and without TSP for the three TSP systems with bus in both EB and WB directions. For CI_CO, there are slight increases in delay for almost all movements, the highest being for I-75/I-85 off ramp SB right turn (Odd_Ramp_SB_Rt). The increase in delay is intuitively reasonable as CI-CO TSP systems takes time from side street to implement GE and EG. The changes in delay are though kept marginal likely because of limiting cycle variations to 20% (24 seconds for a

cycle length of 120 seconds). I-75/I-85 off ramp SB right turn sees proportionally higher delay increase due to a high traffic demand.

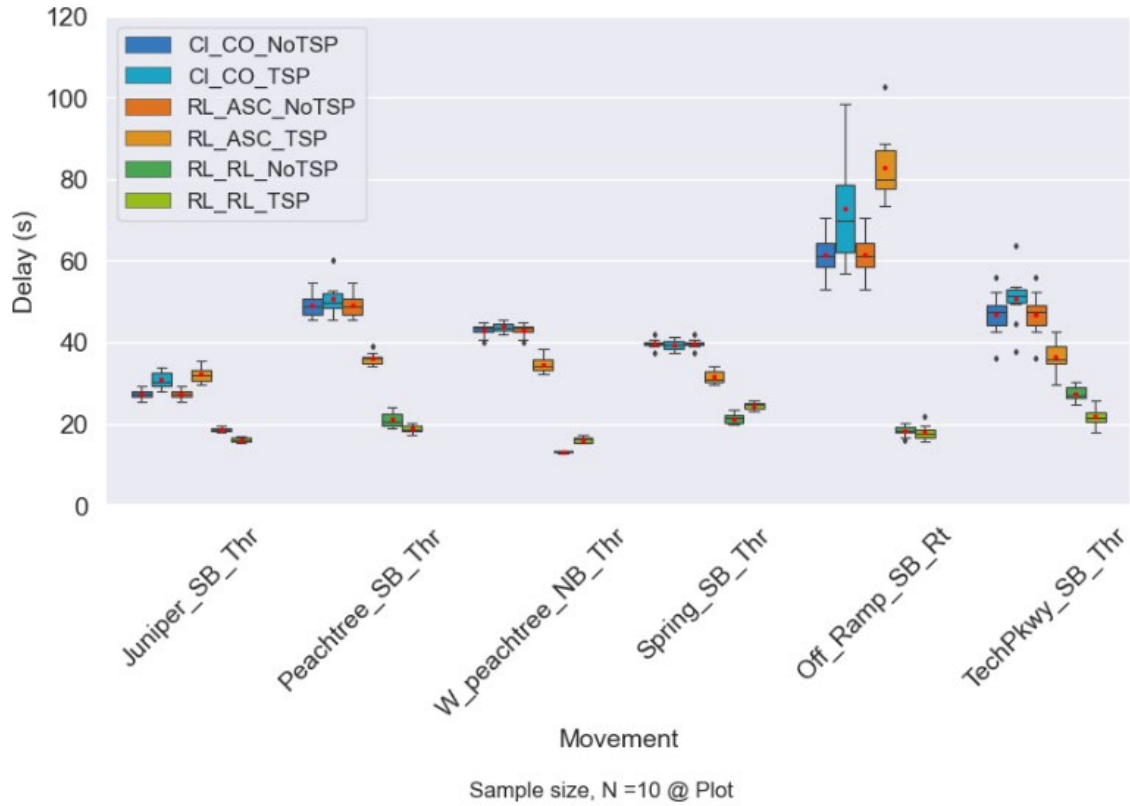


Figure 91: Cross street traffic delay with and without TSP

For RL_ASC system, a mixed impact is observed with some movements seeing a decrease in delay while others seeing an increase. As indicated in the previous chapter, RL provides better general traffic performance for both main street and side street traffic compared to ASC. It is therefore likely that the observed delay improvements in side street are due to better traffic control during the time when the bus is within the two blocks of the intersection and RL the intersection RL agent is activated. As with the CI_CO system, I-75/I-85 off ramp SB right turn (Off_Ramp_SB_Rt) experiences the highest delay increase

during TSP. The high delay for Odd_Ramp_SB_Rt is attributed to the heavy traffic volume on the offramp with limited room to accommodate TSP.

For RL_RL_TSP the NoTSP baseline is significantly lower than the baseline for the other two systems. Changes from the baseline after TSP are very marginal as RL_RL_TSP adaptively compensates for the extra time given to the main street.

8.3.3.2 Mainline through traffic -WB direction

Figure 92 shows delay for WB through traffic with and without TSP for the three TSP systems. For each intersection, delay is measured from the start of the block upstream of the intersection to a point immediately after crossing the intersection. For all TSP systems, delay without TSP is under 30 seconds for all movements. As per results of Chapter 3, this indicates that there is enough room to absorb the impacts of TSP and no dramatic changes in delay would be expected after TSP.

For CI_CO TSP, slight changes in delay are observed for all movements. Some movements see slight increases while others see slight decreases. Decreases in through traffic delay likely come from increased green time for the main line traffic during TSP. Increases in delay likely come from disruptions in coordination and the transitions back to coordination. Note that the changes to cycle lengths are limited to 20%, contributing to limiting any impacts. For RL_ASC_TSP, larger delay increases are observed for Peachtree St. NE (Peachtree_WB_Thr), West Peachtree St. (W_Peachtree_WB_Thr), Spring St. NW (Spring_WB_Thr), and Techwood Dr. NW (Techwood_WB_Thr). As mentioned, when the bus passes the intersection, control resorts to MaxTime® with a set PM peak coordination plan. MaxTime® therefore goes through transition to reach the set offsets and

cycle length. The observed delay increases are attributed to this transition period and are higher than observed and expected for the CI_CO system which has constraints on the cycle variations. Ongoing studies are considering enhancing the RL_ASC_TSP algorithm to enable shorter and smoother transitions.

For RL_RL_TSP system, there are minimal changes in delay for WB through movements. The system adaptively distributes the impacts from TSP not only at intersection but at network level.

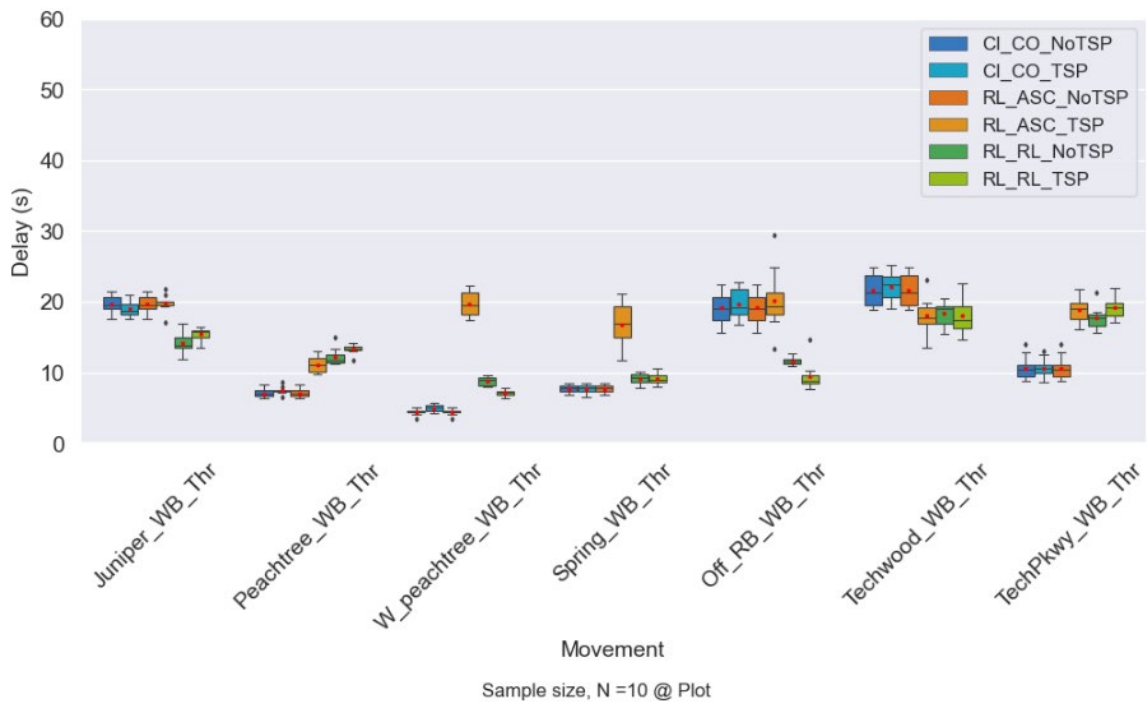


Figure 92: Delay for WB through movements

8.3.3.3 Mainline through traffic -EB

Figure 93 shows delay for EB through traffic with and without TSP for the three TSP systems. EB is the secondary coordination direction for ASC. For CI_CO TSP system,

there is a slight decrease in delay for all movements when TSP is implemented. The main street EB general traffic benefits from the extra time given to the main street movements. The benefit is not offset by interruptions in coordination likely because (a) EB is the secondary coordination direction, and (b) cycle variations are limited to 20% of cycle length. For RL_ASC TSP, all movements experience a delay decrease except for Juniper St. NE EB (Juniper_EB_Thr) and Tech Pkwy. EB (Techpkwy_EB_Thr). The RL_RL_TSP system shows decreases in delay for all movements except for Tech Pkwy. EB (Techpkwy_EB_Thr) that sees a slight increase in delay. For the three systems, the delay decreases intuitively come from extra time assigned to the main line during bus priority.

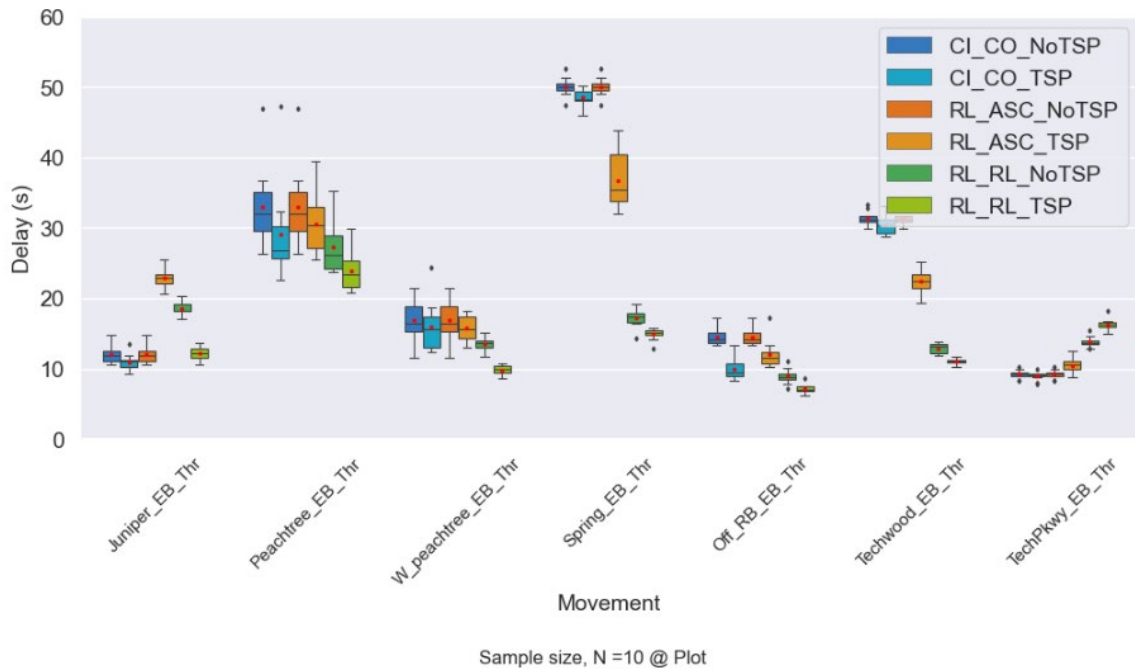


Figure 93: Delay for EB through movements

8.4 Corridor level Adaptive TSP Summary

This chapter extends the work of in Chapter 7 by integrating TSP into the adaptive signal control algorithm developed for an entire arterial corridor. RL agents are trained in two stages with two reward definitions. The first stage focuses on training the agents general traffic control, while second stage focuses on training the agents to select optimal TSP strategies. With trained corridor level TSP agents, two implementations of RL-based TSP are considered: (1) RL-based adaptive TSP with general traffic controlled by the same RL agents in both the presence and absence of the bus, and (2) RL-based adaptive TSP with traffic control when a bus is not present running ASC. The performance of the two RL TSP systems are compared with the performance of the CI-CO TSP implemented with the MaxTime® inbuilt TSP algorithm.

Bus delay progressively decays during the two stages of training, eventually converging after about 1900 episodes to an average of about 130 seconds in WB direction and 100 seconds for EB direction.

Both RL_ASC_TSP and RL_RL_TSP significantly outperform CI_CO. In the WB direction, RL_ASC_TSP shows a 12% lower bus travel time compared to CI_CO_TSP and RL_RL_TSP shows a 9% lower bus travel time compared to CI_CO TSP. In EB direction, RL_ASC_TSP shows a 16% lower bus travel time compared to CI_CO_TSP and RL_RL_TSP shows 18% lower bus travel with CI_CO_TSP. The superior performance of RL-based TSP is attributed to the ability of these RL algorithms to: (1) select coordinated TSP strategies across multiple intersections, and (2) adaptively select TSP strategies based

on the current traffic state compared to CI-CO_TSP, which relies on fixed TSP parameters set in the controller.

For cross street traffic, CI_CO_TSP shows slight increases in delay for almost all movements. For RL_ASC_TSP, a mixed impact is observed with some movements seeing a decrease in delay while others see an increase. For the RL_RL_TSP, the baseline NoTSP is significantly lower than the ASC NoTSP baseline and delay changes are minimal for this control system as RL adaptively compensates for the extra time taken from the side street.

For the main street through traffic, delay impacts are measured across each intersection. In the EB direction, all the three TSP systems show decreases in delay for almost all movements. The delay decrease is attributed to extra green time received by through traffic during TSP service. In WB direction, mixed impacts are observed. For CI_CO_TSP and RL_ASC_TSP, some movements experience slight increases in delay while others experience slight decreases. Delay increases for some movements are attributed to disruptions to coordination, particularly for the RL_ASC_TSP, where MaxTime® needs to transition from free mode to coordination after TSP. Delay decreases are attributed to extra time received by main street movements during TSP and possibly during transition. For RL_RL_TSP there are minimal changes in delay for EB through movements. The system adaptively distributes the impacts from TSP not only at intersection but at network level.

CHAPTER 9 CONCLUSIONS, RECOMMENDATIONS AND CONTRIBUTIONS

9.1 Conclusions

This dissertation develops adaptive Transit Signal Priority (TSP) algorithms based on reinforcement learning (RL), using connected vehicles (CV) data, and software in the loop simulation (SILs). Before embarking on these rather advanced algorithms, the dissertation performed a sensitivity study in simulated environment to explore the fundamental principles of TSP, including the selection of TSP strategies and determining the critical conditions that affect TSP performance.

The RL study starts with a simple hypothetical single intersection implementation with a single RL agent, then progresses to a hypothetical pair of intersections. Building on the lessons learned through these experiments, the last step is the formulation and testing of RL algorithms on a simulated real-world corridor. As RL-based adaptive traffic signal control is not yet fully developed in the literature, especially Multi-Agent Reinforced Learning (MARL), a MARL based adaptive signal control is first developed, then extended to include TSP. The key findings of each chapter are summarized in the following sections.

9.1.1 Exploration of TSP fundamental principles

This study, found in Chapter 3 uses a VISSIM® simulation environment to evaluate the performance of TSP strategies and establish the critical factors and conditions that affect TSP performance. Critical items assessed are TSP Green Extension (GE) and Early Green (EG) strategies, impacts of general traffic demand, and the importance of cycle length. In

comparing the two TSP strategies of GE and EG it was found that: (a) GE tended to provide more significant bus travel time improvements than EG, (b) GE tended to have a lower impact on conflicting movements than EG, and (c) the conflicting movement delay for the same number of seconds of GE versus EG tended to be higher and last for more cycles when EG is granted. This result may be used as a guide in the selection of TSP strategy (GE or EG) to implement.

As congestion increases (modeled by increasing v/c ratios), the following were also observed to increase: the percentage of buses requesting TSP, the likelihood of an ineffective GE, the proportion of buses receiving of EG, negative side street impacts, and the number of cycles required for the sides side street delays to return to non-TSP levels. For a v/c ratio of 1.0 the side street may not return to non-TSP levels for an extended number of cycles past the TSP event, at least 14 cycles in this study. Thus, the likelihood of the impact of the previous bus still being present when the next bus arrives increases as v/c increases.

As cycle length increases (for the same demand), the proportion of buses requesting TSP (GE) decreases, while the effectiveness of GE requests improves. Furthermore, the bus travel time averaged across all buses increases slightly. The cross-street delay due to TSP is lowest when the cycle is moderately higher than optimal (where optimal is calculated to optimize non-transit traffic demands). The number of cycles required to dissipate the effect to the side street and the impact per cycle when a TSP call is served also decrease. However, cross street delay not in the presence of TSP increases as cycle length increases beyond the non-TSP optimal.

Overall, TSP performance appears to be most favorable in lower v/c conditions and GE provides the most benefit to individual buses. However, as congestion increases the effectiveness of TSP decreases. On a highly congested corridor, i.e., v/c ratios approaching or exceeding 1.0, it is possible that TSP may become infeasible because the conflicting non-TSP movements may have insufficient slack in available capacity to recover from the TSP-related green truncation.

Underlying TSP is the balance between transit demand and other vehicle demands. The signal timing for optimal TSP performance, when considering both the transit vehicle and conflicting movements, may not be optimal when TSP is inactive. One implication of this finding is that the setting of a corridor's signal timing parameters should reflect the corridor purpose. That is, if the corridor is designated to serve a significant transit function, then base timing parameters should be selected to improve bus travel time and to lessen TSP impacts on general traffic. Where transit is not a primary focus then higher cycle lengths or TSP based offsets may not be warranted.

9.1.2 Impact of dwell-time on the performance of transit signal priority

As seen in section 4.3, the first part of this study analyzed trends and distributions inherent in dwell-time data deduced from one-year of Automated Passenger Count (APC) data for one of the busiest bus routes in Atlanta. Dwell-time varied from stop to stop and for each stop by time of day. Stops with high magnitude dwell-time also showed high dwell-time variance and a smaller probability of a bus skipping the stop. For several stops, dwell-time distributions showed directional peaks during the peak hours of travel. For the southbound direction, most stops showed an AM peak (600-900 hours) and an afternoon peak (1300-

1500 hours) while for northbound direction, most stops showed a more spread-out peak, from 1100 to 1900 hours. Dwell-time data were closely fitted by inverse Gaussian, log normal, power lognormal, Fisk (log-logistic), and Johnson's SU distributions.

The second part of the study (section 4.4) used a simulation environment to evaluate the impact of dwell-time magnitude and variability on TSP performance. Sensitivity experiments were performed with three field dwell-time distributions at both far-side and near-side bus stops. For the far-side bus stop, dwell-time significantly impacted the bus arrival profile at the check-in detector and thus the TSP strategy (GE, EG, No action) selected. Higher dwell-time magnitudes and variability led to a higher share of EG, which is not as effective as GE. Setting offsets considering the prevailing dwell-time distribution may help improve TSP performance. At near-side bus stops, dwell-time variability introduced more uncertainty in ETA estimation and significantly reduced TSP effectiveness, especially GE. TSP performance in terms of GE success, bus travel time, and side street traffic delay was significantly better at far-side stops compared to near-side stops.

9.1.3 Single Intersection TSP based on reinforcement learning

This study (Chapter 5) utilizes a microscopic simulation environment and CV data to develop and test an event-based RL agent. This agent assumes intersection control from another RL based traffic signal controller when TSP transit buses enter the CV communication range of the intersection. The background general traffic controller is trained and tested, demonstrating comparable performance with an actuated controller for a single intersection. The trained RL based TSP agent is seen to reduce the bus travel time

by approximately 21%, with marginal impacts to general traffic at a saturation rate of 0.95. The TSP agent also shows slightly better performance in improving bus travel time compared to actuated signal control with TSP. To improve run time efficiencies, PTV VISSIM®'s event-based scripting is used instead of the commonly used COM API. Using event-based scripting was many times fold faster than using COM and is what partly made training feasible as using COM was prohibitively slow.

9.1.4 Adaptive transit signal priority based on multi-agent reinforcement learning

This study (Chapter 6) integrated TSP into MARL for signal control. The first part of the study developed adaptive signal control based on the MARL and a (Centralize Training Distributed Execution) CTDE paradigm for a pair of coordinated intersections in a microscopic simulation environment. Two agents, one for each intersection, were centrally trained under a Value Decomposition Network (VDN) architecture. The objective was to minimize total network delay while achieving signal coordination in a prioritized direction. The trained agents show slightly better performance compared to coordinated actuated signal control, based on overall intersection delay at v/c of 0.95. In the second part of the study, event-based TSP was developed. In one variation, independent TSP agents were formulated and trained under a Decentralized Training Decentralized Execution (DTDE) framework to implement TSP at each intersection when a bus is present. In a second variation, the two TSP agents were centrally trained under CTDE framework and VDN architecture to select and implement coordinated TSP strategies across the two intersections when a bus is present. In both cases the agents converged to the same bus delay value; however, the independent agents showed high instability throughout the training process. The instability is likely to increase with decreasing spacing between

intersections (spacing was 1600ft for the case study network) and will likely affect convergence. During testing, the two independent agents reduce bus delay across the two intersections by 22% compared to the no TSP case while the coordinated TSP agents achieve 27% delay reduction. In both cases, there is a slight increase in delay within five minutes of bus check-in for most bus side-street movements.

9.1.5 Corridor level adaptive reinforcement learning based signal control

This study (Chapter 7) formulates a multi-agent proximal policy optimization algorithm to implement adaptive and coordinated traffic control along an arterial corridor. The formulated Multi Agent Proximal Policy Optimization (MA-PPO) has centralized critic architecture in which each agent has an actor network that selects independent actions conditioning on local observations and a centralized critic that estimates the agent's value function conditioning on global observations. All agents were formulated to allow selection and implementation of up to eight signal phases as commonly implemented in the field controllers. The formulated algorithm was tested on a simulated real-world corridor with seven intersections. The performance of the formulated MA-PPO adaptive control algorithm was compared with the field implemented ASC signal timing plans modeled using PTV-Vissim®-MaxTime® SILs.

The speed of convergence for each agent largely depended on the size of the action space, which in turn depends on the number and sequence of signal phases. The intersections with four or fewer phases converged within 300, 30-simulated-minute episodes, while each of the intersections with seven and eight phases required double to triple the number of episodes for convergence.

For the field measured traffic volumes, for all movements, the trained MA-PPO performed significantly better than the field implemented ASC signal timings. For the main street, MA-PPO optimizes performance in two directions compared to ASC, which largely optimizes performance in the primary coordinated direction. Using travel time as the performance measure, the two control systems showed very comparable performance in the WB direction, which is the PM peak direction and the primary coordination direction for ASC. In the EB direction, MA-PPO showed a 24% reduction in travel time compared to ASC. For cross streets, MA-PPO showed significantly reduced delay

To capture the variability of traffic volumes and further test the robustness of the trained MA-PPO, sensitivity experiments were performed for two additional volumes sets: (a) 5% above, and (b) 10% below the field measured volumes. For both main street movements and cross streets movements, MA-PPO consistently outperformed ASC at the three volume levels. This demonstrated the robustness of the trained model to adapt to variations in traffic demand.

9.1.6 Corridor level adaptive TSP based on reinforcement learning, connected vehicles and software in the loop simulation

This study (Chapter 8) extends the work done in Chapter 7 by integrating TSP into the adaptive signal control algorithm developed for an entire arterial corridor. RL agents are trained in two stages, each with a unique reward definition. The first stage focuses on training the agents general traffic control, while second stage focuses on training the agents to select optimal TSP strategies. With trained corridor level TSP agents, two implementations of RL-based TSP are considered: (1) RL-based adaptive TSP with general

traffic controlled by the same RL agents in absence of the bus, and (2) RL-based adaptive TSP with ASC when a bus is not present. The performance of the two RL TSP systems is compared with the performance of CI-CO TSP implemented with MaxTime® inbuilt TSP algorithm.

Bus delay progressively decreases during the two stages of training, eventually converging after about 1900 episodes to an average of about 130 seconds in WB direction and 100 seconds for EB direction.

Both RL_ASC_TSP and RL_RL_TSP significantly outperform CI_CO_TSP. In the WB direction, RL_ASC_TSP shows a 12% lower bus travel time compared to CI_CO_TSP. RL_RL_TSP shows 9% lower bus travel time compared to CI_CO_TSP. In EB direction, RL_ASC_TSP shows 16% lower bus travel time compared to CI_CO_TSP. RL_RL_TSP shows 18% lower bus travel time compared to CI_CO_TSP. The superior performance of RL-based TSP is attributed to the ability of these RL algorithms to: (1) select coordinated TSP strategies across multiple intersections, and (2) adaptively select TSP strategies based on the current traffic state compared to CI_CO_TSP, which relies on fixed TSP parameters set in the controller.

For cross street traffic, CI_CO_TSP shows slight increases in delay for almost all movements. For RL_ASC_TSP, a mixed impact is observed with some movements seeing a decrease in delay while others see an increase. Decreases in delay are attributed to improved control of traffic when the RL agent is activated, while delay increases are attributed to the impacts of transition process to get back into coordination. For the RL_RL_TSP system, the baseline NoTSP is already significantly lower than the ASC

NoTSP baseline. Delay changes are minimal for this control system as RL adaptively compensates for the extra time taken from the side street.

For the main street through traffic, delay impacts are measured across each intersection. In the EB direction, all the three TSP systems show decreases in delay for almost all movements. This is attributed to extra green time received by through traffic during TSP service. In WB direction, mixed impacts are observed. For CI_CO_TSP and RL_ASC_TSP, some movements experience slight increases in delay while others experience decreases in delay. Delay increases for some movements are attributed to disruptions to coordination and transitioning from free mode to coordination in RL_ASC_TSP. Delay decreases are attributed to extra time received by main street movements during TSP. For the RL_RL_TSP, there are minimal changes in delay for the EB through movements. The system adaptively distributes the impacts from TSP not only at intersection but at network level.

9.2 Recommendations

9.2.1 Defining thresholds for impacts on general traffic

This dissertation is focused on developing operational signal control logic for TSP. Several planning and policy level aspects of TSP are reserved for later consideration/integration and further study. TSP system design involves balancing tradeoffs between improving bus service and avoiding excessive deterioration of the general traffic level of service. The challenge in selecting the threshold is in determining the acceptable degradation of general traffic for improved bus service. (Although, it is highlighted that improved bus service so not always come at the cost of vehicle service). Whereas there can be technical metrics to

define this threshold, including the lateness of the bus, the available queue storage on the side street, the bus passenger load, etc., the selection of the threshold is ultimately a policy question. The dissertation incorporates adjustable parameters in the algorithms that can be tweaked to achieve the desired tradeoff. Detailed studies are required to formalize and provide guidance on selecting these thresholds. These studies could involve surveys of the general public, transit riders, and motorists.

9.2.2 Defining traffic state with readily available data

In the definition of traffic state, the dissertation assumed availability of CV data to establish the total count of vehicles in each lane, on each approach. However, the current penetration rate of CVs is not yet sufficient to establish the traffic state to this detail. In absence or as an alternative to CV data, video cameras at intersections, capable of seeing the back of queue could provide videos/images to define the traffic state. Although, such camera placements are still rare. It is therefore recommended that future studies explore defining the traffic state with readily available data including volumes from upstream and downstream loop detectors.

9.2.3 Explore parallelized training architectures

A key challenge of training of RL agents in a high-fidelity traffic microscopic simulation environment such as VISSIM® is the run time efficiency necessary to reach the hundreds to thousands of simulation runs required to fully train the agents. This dissertation uses online serialized training for all developed algorithms. Several days were required to complete training of each agent(s). This limited the exploration and experimentation that

could be achieved. Ongoing follow up studies are considering parallelized training architectures to reduce the required simulation run time.

9.2.4 More structured training architectures for increasing complexity

The results of two stage training prompted a search for more formalized techniques of training agents/machines for tasks of progressively increasing complexity. One such approach is curriculum learning whose formal formulation is attributed to Bengio et al. (2009a). Inspired by the human learning process of starting with basic concepts during infancy and structured curricula in schools, curriculum learning trains agents/machines starting with simpler aspects of the task or easier subtasks and then gradually increases the complexity. Advanced traffic signal control strategies such as priority and preemption are potentially prime candidates for curriculum learning where agents could start by learning simplified.

9.2.5 Adaptive transition logic for RL_ASC_TSP

In the implemented RL_ASC_TSP, when the bus exits the intersection, the RL agent is deactivated and ASC again takes control of the intersection without external manipulation. When the background signal plan is coordinated, the ASC has to go through the transition from free mode to the selected pattern. This transition can take several minutes and may cause delay increases. Follow up studies should consider strategies to shorten the required transit period and the impact on the general traffic. One approach could be to develop a traffic demand responsive transition logic implemented by the RL agent with control handed back to ASC when the required cycle length and offset have been, or are almost reached.

9.3 Contributions

This section presents the contributions of the dissertation under Chapter 3 through Chapter 8.

9.3.1 *Exploration of TSP fundamental principles*

A comprehensive review of literature presented in Chapter 2 shows that the benefits and impacts of TSP vary widely across studies. This is in part due to (1) the different measures of effectiveness (MOEs) adopted, (2) the wide range of conditions and parameters of the transit vehicles and transit routes, and (3) challenges in controlling for traffic demand, signal control, etc., during performance evaluation. A 2020 Transit Cooperative Research Program (TCRP) report of discussed in Chapter 2 assesses the current state of practice of transit signal priority (TSP) for 31 surveyed transit agencies in US and Canada. The study found that despite the widespread adoption of TSP, there was no universal understanding of the selection of TSP strategies and parameters, the quantification of TSP benefits and impacts, and identified the need to further research advanced signaling approaches integrating TSP, among other recommendations. Chapter 3 of this dissertation is an effort to contribute to bridging these gaps in research and literature. The chapter makes the following contributions.

1. The study demonstrates a selection of MOEs to evaluate TSP performance. High resolution controller data (consisting of detector and signal events) is fused with performance data to isolate and measure impacts of each TSP event.
2. Through a series of simulation experiments the study establishes the impacts of different TSP strategies, and TSP parameters. TSP performance envelopes are

established, demonstrating conditions of traffic demand where TSP is most beneficial and where it may not be feasible. The study makes the first steps towards much needed guidelines for selection of TSP strategies and parameters based on traffic demand and other existing field conditions.

3. With experiments on cycle length, the study demonstrates the benefits of including TSP in the initial arterial signal timing objectives and not considering TSP as an afterthought.

9.3.2 Analysis of trends in dwell-time data and the impact of dwell-time on TSP performance

There have been limited efforts to evaluate the impact of dwell-time magnitude and variability on TSP performance. Moreover, the existing efforts have largely adopted idealistic dwell-time distributions that are not representative of the field dwell-times. This is despite the increasing ubiquity of large dwell-time datasets from APC data. Chapter 4 of this dissertation analyses trends in dwell-time data and assesses the impact of actual field dwell-time data on TSP performance. The contributions of the chapter include the following.

1. From the analysis of dwell-time data derived from one-year APC data, the study demonstrates the variation of dwell-time from stop to stop, and by time of day for each stop. Parametric distributions that fit dwell-time data are identified. This analysis presents the basis for more realistic modeling of dwell-time in absence of field data.

2. A series of simulation experiments are performed based on field dwell-time data to assess the impact on TSP performance for both near-side and far-side bus stops. The experiments performed and the results obtained demonstrate the need for improved integration of dwell-time in the selection of TSP strategies and parameters. Recommendations are made for finetuning background signal timing parameters, such as offsets depending on the corridor objectives for transit and general traffic.

9.3.3 Single intersection TSP based on reinforcement learning

There is heightened interest to develop AI-based traffic control systems. Several recent studies have formulated and trained RL-based traffic controllers integrating CV data. There are ongoing efforts to extend RL-based traffic control algorithms to include TSP. RL-based traffic control and RL-based TSP control remain open research areas with several gaps to fill. This chapter makes the following contributions.

1. This study formulates event-based RL algorithms to implement TSP and background traffic control. By training two separate agents, one for general traffic and one for TSP, improvements are made in (1) training effectiveness, and (2) robustness and generalizability for both agents as bus arrivals are often much lower compared to general traffic. Pseudocodes are included in the dissertation and python codes made publicly available on GitHub.
2. A key challenge of training RL algorithms in high fidelity traffic microscopic simulation platforms is run time efficiency as hundreds to thousands of runs are needed to train the agents. For VISSIM®, a widely used microscopic simulation

platform, the study demonstrates architecture selection and optimization strategies to improve run time efficiency and allow scalability. Improving run time efficiency is an overarching theme of Chapter 5 through Chapter 8.

9.3.4 Adaptive transit signal priority based on multi-agent reinforcement learning

Most of the previous RL-based traffic control studies have considered single agent formulations with without the added challenges of signal coordination/agent cooperation, non-stationarity of the environment, and scalability issues. The chapter makes the following contributions.

1. This study adds to the very few studies that consider MARL for traffic signal control. It is shown that centralized training and the proper selection of the reward function can achieve signal coordination without the need for additional explicit communication modules proposed in the previous studies.
2. There is almost no literature on MARL-based TSP. This study formulates, trains, and tests different variations of MARL agents to implement TSP across multiple coordinated intersections. The state of art TSP algorithms are implemented separately for each intersection with only limited and still evolving use of peer-to-peer user defined logic in certain traffic controllers. A few studies have proposed coordinated TSP strategies based on mathematical programming. In this chapter, RL agents are formulated to implement TSP coordinated strategies at adjacent intersections. Coordinated TSP agents show better performance compared to independent agents with even greater benefits anticipated with closer intersection spacings tested in Chapter 8. As with the single agent formulation above,

pseudocodes are included in the dissertation and python codes made publicly available on GitHub.

9.3.5 Corridor level adaptive reinforcement learning based signal control

RL-based signal control is still evolving with more efforts required to reach field ready implementable formulations. As discussed in Chapter 2 and Chapter 7, previous RL-based traffic control studies assume highly simplified signal timing plans compared to what is normally implemented in the field. Additionally, the limited efforts that have attempted to formulate MARL-based traffic control have almost universally adopted value-based RL methods. However, recent literature shows that policy-based RL methods are likely to outperform value-based methods, especially in partially observable environments. The chapter makes the following contributions.

1. This study demonstrates the selection and implementation of RL actions based on a full dual-ring barrier controller. Compared to the single ring and other simplified configurations, the dual ring set up is (1) much more complex to formulate and implement, and (2) requires more training time for the agents to fully learn to control this complex system.
2. The study demonstrates the formulation, training, and testing of centralized critic MA-PPO for arterial adaptive signal control. Training architectures are included in the document and python codes made publicly available on GitHub.
3. The study presents comparisons of the formulated MA-PPO adaptive signal control system and the field implemented ASC signal timings modelled with SILs. The formulation and testing are performed for a simulated real-world corridor with

actual field volumes and geometry. For the field measured traffic volumes, for all movements, the trained MA-PPO performed significantly better than the field implemented ASC signal timings.

4. For the field implemented ASC signal timings, the study demonstrates architectures for VISSIM® - MaxTime® SILs which enables more realistic replication of field implemented signal timings.

9.3.6 Corridor level adaptive TSP based on reinforcement learning, connected vehicles, and software in the loop simulation

This study integrates TSP into the formulated corridor level adaptive TSP. With trained TSP agents, two implementation architectures are tested, (1) RL agents controlling both general traffic and implementing TSP, and (2) RL agents implementing TSP and the background traffic control running on MaxTime® ASC. The following are some of the contributions made by the study:

1. The study formulates and trains RL agents to select coordinated TSP strategies across multiple intersections. The agents are formulated and tested for real-world corridor implementations. The study demonstrates the use of segmented training with different reward functions for each training segment. This enables the same agents to be trained to control general traffic and implement bus priority.
2. The study tests the formulated TSP algorithms on a simulated real-world corridor of seven intersections with actual field conditions. This testing allows evaluation of TSP benefits and impacts at the corridor level. Additionally, the corridor level tests allow the assessment of the scalability of the developed algorithms.

3. The study demonstrates the implementation architecture of RL-based TSP with background signal controlling running on ASC. The architecture includes VISSIM® - MaxTime® SILs and event triggered RL-based TSP agents. NTCIP commands are used to implement RL selected actions in MaxTime®.

REFERENCES

- Al-Sahili, K. A., and W. C. Taylor. 1996. Evaluation of Bus Priority Signal Strategies in Ann Arbor, Michigan. *Transportation research record*, 1554(1):74-79. <https://doi.org/10.1177/0361198196155400110>
- Albrecht, S. V., F. Christianos, and L. Schäfer. 2024. *Multi-Agent Reinforcement Learning: Foundations and Modern Approaches*. MIT Press.
- AlHadidi, T., and H. A. Rakha. 2019. Modeling Bus Passenger Boarding/Alighting Times: A Stochastic Approach. *Transportation Research Interdisciplinary Perspectives*, 2:100027. <https://doi.org/https://doi.org/10.1016/j.trip.2019.100027>
- Ali, M. D. S., H. Haule, J. Kodi, P. Alluri, and T. Sando. 2023. Transferability of a Calibrated Microscopic Simulation Model Parameters for Operational Assessment of Transit Signal Priority. *Public Transport*, 15(3):791-812. <https://doi.org/10.1007/s12469-023-00329-4>
- Anderson, P., and C. Daganzo. 2019. Effect of Transit Signal Priority on Bus Service Reliability. *Transportation Research Procedia*, 38:2-19. <https://doi.org/10.1016/j.trpro.2019.05.002>
- Aslani, M., M. S. Mesgari, S. Seipel, and M. Wiering. 2019. Developing Adaptive Traffic Signal Control by Actor-Critic and Direct Exploration Methods. *Proceedings of the Institution of Civil Engineers: Transport*, 172(5):289-298. <https://doi.org/10.1680/jtran.17.00085>
- Bálint, K., T. Tamás, and B. Tamás. 2022. Deep Reinforcement Learning Based Approach for Traffic Signal Control. *Transportation Research Procedia*, 62:278-285. <https://doi.org/10.1016/j.trpro.2022.02.035>
- Balke, K. N., C. L. Dudek, and T. Urbanik. 2000. Development and Evaluation of Intelligent Bus Priority Concept. *Transportation research record*, 1727(1):12-19. <https://doi.org/10.3141/1727-02>
- Bayrak, M., and S. I. Guler. 2020. Determining Optimum Transit Signal Priority Implementation Locations on a Network. *Transportation research record*, 2674(10):387-400. <https://doi.org/10.1177/0361198120934792>

- Beak, B., M. Zamanipour, K. L. Head, and B. Leonard. 2018. Peer-to-Peer Priority Signal Control Strategy in a Connected Vehicle Environment. *Transportation research record*, 2672(18):15-26. <https://doi.org/10.1177/0361198118773567>
- Bengio, Y., J. Louradour, R. Collobert, and J. Weston. 2009a. Curriculum Learning *Proceedings of the 26th Annual International Conference on Machine Learning*, Montreal, Quebec, Canada. <https://doi.org/10.1145/1553374.1553380>
- Bengio, Y., J. Louradour, R. Collobert, and J. Weston. 2009b. Curriculum Learning. *Proceedings of the 26th annual international conference on machine learning*, 41-48.
- Bhaskar, A., E. Chung, O. de Mouzon, and A.-G. Dumont. 2007. Methodology for Travel Time Estimation on a Signalised Arterial.
- Bokade, R., X. Jin, and C. Amato. 2023. Multi-Agent Reinforcement Learning Based on Representational Communication for Large-Scale Traffic Signal Control. *IEEE Access*, 11:47646-47658. <https://doi.org/10.1109/ACCESS.2023.3275883>
- Bouktif, S., A. Cheniki, and A. Ouni. 2021. Traffic Signal Control Using Hybrid Action Space Deep Reinforcement Learning. *Sensors*, 21(7). <https://doi.org/10.3390/s21072302>
- Bouktif, S., A. Cheniki, A. Ouni, and H. El-Sayed. 2023. Deep Reinforcement Learning for Traffic Signal Control with Consistent State and Reward Design Approach. *Knowledge-Based Systems*, 267. <https://doi.org/10.1016/j.knosys.2023.110440>
- Casas, N. 2017. Deep Deterministic Policy Gradient for Urban Traffic Light Control. *arXiv preprint arXiv:1703.09035*. <http://arxiv.org/abs/1703.09035>
- Chang, A., Y. Ji, C. Wang, and Y. Bie. 2024. Cvdmarl: A Communication-Enhanced Value Decomposition Multi-Agent Reinforcement Learning Traffic Signal Control Method. *Sustainability (Switzerland)*, 16(5). <https://doi.org/10.3390/su16052160>
- Chanloha, P., J. Chinrungrueng, W. Usaha, C. Aswakul, P. Chanloha, C. Aswakul, W. Usaha, and J. Chinrungrueng. (2015). *Traffic Signal Control with Cell Transmission Model Using Reinforcement Learning for Total Delay Minimisation* (INTERNATIONAL JOURNAL OF COMPUTERS COMMUNICATIONS & CONTROL ISSN, Issue.

- Chao, C.-C., J.-W. Hsieh, and B.-S. Wang. 2022. Cooperative Reinforcement Learning on Traffic Signal Control. <http://arxiv.org/abs/2205.11291>
- Chen, M., X. Liu, J. Xia, and S. I. Chien. 2004. A Dynamic Bus-Arrival Time Prediction Model Based on Apc Data. *Computer-Aided Civil and Infrastructure Engineering*, 19(5):364-376. <https://doi.org/https://doi.org/10.1111/j.1467-8667.2004.00363.x>
- Chen, X., G. Xiong, Y. Lv, Y. Chen, B. Song, and F. Y. Wang. 2021. A Collaborative Communication-Qmix Approach for Large-Scale Networked Traffic Signal Control. IEEE Conference on Intelligent Transportation Systems, Proceedings, ITSC, 3450-3455. <https://doi.org/10.1109/ITSC48978.2021.9564683>
- Cheng, H. K., K. P. Kou, and K. I. Wong. 2022. Transit Signal Priority Control with Deep Reinforcement Learning 2022 10th International Conference on Traffic and Logistic Engineering (ICTLE),
- Cvijovic, Z., M. Zlatkovic, A. Stevanovic, and Y. Song. 2022. Conditional Transit Signal Priority for Connected Transit Vehicles. *Transportation research record*, 2676(2):490-503. <https://doi.org/10.1177/03611981211044459>
- Dai, G., H. Wang, and W. Wang. 2015. Signal Optimization and Coordination for Bus Progression Based on Maxband. *KSCE Journal of Civil Engineering*. <https://doi.org/10.1007/s12205-015-1516-4>
- Dai, Z., X. Ma, and X. Chen. 2019. Bus Travel Time Modelling Using Gps Probe and Smart Card Data: A Probabilistic Approach Considering Link Travel Time and Station Dwell Time. *Journal of Intelligent Transportation Systems*, 23(2):175-190. <https://doi.org/https://doi.org/10.1080/15472450.2018.1470932>
- Ding, J., M. Yang, W. Wang, C. Xu, and Y. Bao. 2015. Strategy for Multiobjective Transit Signal Priority with Prediction of Bus Dwell Time at Stops. *Transportation research record*, 2488(1):10-19. <https://doi.org/10.3141/2488-02>
- Ding, J., M. Y. Yang, Y. Cao, and S. Kong. 2014. Dwell Time Prediction of Bus Rapid Transit Using Arima-Svm Hybrid Model. *Applied Mechanics and Materials*, 587-589:1993-1997. <https://doi.org/10.4028/www.scientific.net/AMM.587-589.1993>
- Dion, F., G. Abu-Lebdeh, and M. Ghanim. 2014. The Impact of Dwell Time Variability on Transit Signal Priority Performance. *Canadian Journal of Civil Engineering*, 41. <https://doi.org/10.1139/cjce-2012-0306>

- Dion, F., H. Rakha, and Y. Zhang. 2004. Evaluation of Potential Transit Signal Priority Benefits Along a Fixed-Time Signalized Arterial. *Journal of Transportation Engineering*, 130(3):294-303. [https://doi.org/doi:10.1061/\(ASCE\)0733-947X\(2004\)130:3\(294\)](https://doi.org/doi:10.1061/(ASCE)0733-947X(2004)130:3(294))
- Dueker, K., T. Kimpel, and J. Strathman. 2004. Determinants of Bus Dwell Time. *Journal of Public Transportation*, 7(1):21-40. <https://doi.org/http://doi.org/10.5038/2375-0901.7.1.2>
- Ekeila, W., T. Sayed, and M. E. Esawey. 2009. Development of Dynamic Transit Signal Priority Strategy. *Transportation research record*, 2111(1):1-9. <https://doi.org/10.3141/2111-01>
- El-Tantawy, S., B. Abdulhai, and H. Abdelgawad. 2014. Design of Reinforcement Learning Parameters for Seamless Application of Adaptive Traffic Signal Control. *Journal of Intelligent Transportation Systems: Technology, Planning, and Operations*, 18(3):227-245. <https://doi.org/10.1080/15472450.2013.810991>
- Fan, H., H. Lu, Z. Dai, R. Passmore, A. Guin, K. Watkins, and R. Guensler. 2023. Combined Effect of Changes in Transit Service and Changes in Occupancy on Per-Passenger Energy Consumption. *Transportation research record*, 2677(2):1252-1265. <https://doi.org/10.1177/03611981221111160>
- Fan, W., and T. Yang. 2024. Transit Signal Priority Control with Connected Vehicle Technology: Deep Reinforcement Learning Approach [Tech Report]. <https://rosap.ntl.bts.gov/view/dot/77807>
- Farid, Y. Z., E. Christofa, and L. Paget-Seekins. 2016. Estimation of Short-Term Bus Travel Time by Using Low-Resolution Automated Vehicle Location Data. *Transportation research record*, 2539(1):113-118. <https://doi.org/10.3141/2539-13>
- Federal Transit Administration. (2023). *National Transit Summaries and Trends*.
- Federal Transit Administration. (2024). *Effects of the Covid-19 Pandemic on Transit Ridership and Accessibility*. <https://www.transit.dot.gov/research-innovation/effects-covid-19-pandemic-transit-ridership-and-accessibility-report-0268>
- Fricke, and D. Jon. 2011. Bus Dwell Time Analysis Using on-Board Video.

- Fu, X., S. Chen, Q. Liang, and Y. Li. 2023. Research on Multi-Agent Reinforcement Learning Traffic Control. 2023 IEEE International Conference on Control, Electronics and Computer Technology, ICCECT 2023, 231-239. <https://doi.org/10.1109/ICCECT57938.2023.10140678>
- Glick, T. B., and M. A. Figliozzi. 2019. Analysis and Application of Log-Linear and Quantile Regression Models to Predict Bus Dwell Times. *Transportation research record*, 2673(10):118-128. <https://doi.org/10.1177/0361198119848701>
- Grisé, E., and A. El-Geneidy. 2017. Identifying the Bias: Evaluating Effectiveness of Automatic Data Collection Methods in Estimating Details of Bus Dwell Time. *Transportation research record*, 2647(1):33-40. <https://doi.org/10.3141/2647-05>
- Han, Y., M. Kim, and Y. Kim. 2022. Progression Control Model to Enhance Performance of Transit Signal Priority. *IEEE Access*, 10:14397-14408. <https://doi.org/10.1109/ACCESS.2022.3146716>
- Hausknecht, M., and P. Stone. 2015. Deep Recurrent Q-Learning for Partially Observable Mdns. <http://arxiv.org/abs/1507.06527>
- Hessel, M., J. Modayil, H. Van Hasselt, T. Schaul, G. Ostrovski, W. Dabney, D. Horgan, B. Piot, M. Azar, and D. Silver. 2018. Rainbow: Combining Improvements in Deep Reinforcement Learning. Proceedings of the AAAI conference on artificial intelligence,
- Hu, J., B. Park, and Y.-J. Lee. 2015. Coordinated Transit Signal Priority Supporting Transit Progression under Connected Vehicle Technology. *Transportation Research Part C: Emerging Technologies*, 55. <https://doi.org/10.1016/j.trc.2014.12.005>
- Hu, J., B. Park, and A. E. Parkany. 2014. Transit Signal Priority with Connected Vehicle Technology. *Transportation research record*, 2418(1):20-29. <https://doi.org/10.3141/2418-03>
- Hu, J., B. B. Park, and Y.-J. Lee. 2016. Transit Signal Priority Accommodating Conflicting Requests under Connected Vehicles Technology. *Transportation Research Part C: Emerging Technologies*, 69:173-192. <https://doi.org/https://doi.org/10.1016/j.trc.2016.06.001>
- Hu, W. X., H. Ishihara, C. Chen, A. Shalaby, and B. Abdulhai. 2023. Deep Reinforcement Learning Two-Way Transit Signal Priority Algorithm for Optimizing Headway

- Adherence and Speed. *IEEE Transactions on Intelligent Transportation Systems*, 24(8):7920-7931. <https://doi.org/10.1109/TITS.2023.3266461>
- Huang, S., and S. Ontañón. 2020. A Closer Look at Invalid Action Masking in Policy Gradient Algorithms. *arXiv preprint arXiv:2006.14171*.
- Hunter, M., M. O. Rodgers, M. Saracco, and A. Pizano. 2024. Vissim™ Simulation Calibration Procedure [Tech Report]. <https://rosap.ntl.bts.gov/view/dot/74243>
- Isukapati, I., C. Igoe, E. Bronstein, V. Parimi, and S. Smith. 2020. Hierarchical Bayesian Framework for Bus Dwell Time Prediction. *IEEE Transactions on Intelligent Transportation Systems*, PP:1-10. <https://doi.org/10.1109/TITS.2020.2979390>
- Isukapati, I., H. Rudová, G. Barlow, and S. Smith. 2017. Analysis of Trends in Data on Transit Bus Dwell Times. *Transportation Research Record: Journal of the Transportation Research Board*, 2619:64-74. <https://doi.org/10.3141/2619-07>
- Jeong, Y., and Y. Kim. 2014. Tram Passive Signal Priority Strategy Based on the Maxband Model. *KSCE Journal of Civil Engineering*, 18(5):1518-1527. <https://doi.org/10.1007/s12205-014-0159-1>
- Jian, D., Z. Lu, and Z. Yan. 2013. Mixed Model for Prediction of Bus Arrival Times. 2013 IEEE Congress on Evolutionary Computation, 2918-2923. <https://doi.org/10.1109/CEC.2013.6557924>
- Kim, W., and L. R. Rilett. 2005. Improved Transit Signal Priority System for Networks with Near-side Bus Stops. *Transportation research record*, 1925(1):205-214. <https://doi.org/10.1177/0361198105192500121>
- Koonce, P. 2008. Traffic Signal Timing Manual [Tech Report]. <https://rosap.ntl.bts.gov/view/dot/800>
- Koonce, P., J. Ringert, and T. Urbanik. 2002. Detection Range Setting Methodology for Signal Priority. *Journal of Public Transportation*, 5. <https://doi.org/10.5038/2375-0901.5.2.6>
- Kumar, B. A., R. Jairam, S. S. Arkatkar, and L. Vanajakshi. 2019. Real Time Bus Travel Time Prediction Using K-Nn Classifier. *Transportation Letters*, 11(7):362-372. <https://doi.org/10.1080/19427867.2017.1366120>

- Law, A. M. 2015. *Simulation Modeling and Analysis*. 5th International ed. New York. McGraw-Hill New York.
- Lee, H., Y. Han, Y. Kim, and Y. H. Kim. 2022. Effects Analysis of Reward Functions on Reinforcement Learning for Traffic Signal Control. *PLoS ONE*, 17(11 November). <https://doi.org/10.1371/journal.pone.0277813>
- Lee, W.-H., and H.-C. Wang. 2022. A Person-Based Adaptive Traffic Signal Control Method with Cooperative Transit Signal Priority. *Journal of Advanced Transportation*, 2022:2205292. <https://doi.org/10.1155/2022/2205292>
- Lee, Y.-J., S. Dadvar, J. Hu, and B. B. Park. 2017. Transit Signal Priority Experiment in a Connected Vehicle Technology Environment. *Journal of Transportation Engineering, Part A: Systems*, 143(8):05017005. <https://doi.org/doi:10.1061/JTEPBS.0000062>
- Li, D., J. Wu, M. Xu, Z. Wang, and K. Hu. 2020. Adaptive Traffic Signal Control Model on Intersections Based on Deep Reinforcement Learning. *Journal of Advanced Transportation*, 2020. <https://doi.org/10.1155/2020/6505893>
- Li, F., Z. Duan, and D. Yang. 2012. Dwell Time Estimation Models for Bus Rapid Transit Stations. *Journal of Modern Transportation*, 20(3):168-177. <https://doi.org/10.1007/BF03325795>
- Li, H., S. Li, and X. Zhang. 2023. Coordination Optimization of Real-Time Signal Priority of Self-Driving Buses at Arterial Intersections Considering Private Vehicles. *Applied Sciences*, 13(19):10803. <https://www.mdpi.com/2076-3417/13/19/10803>
- Li, L., S. Member, Y. Lv, and F.-Y. Wang. (2016). *Traffic Signal Timing Via Deep Reinforcement Learning*. <http://ieeexplore.ieee.org>.
- Li, M., Y. Yin, W.-B. Zhang, K. Zhou, and H. Nakamura. 2011. Modeling and Implementation of Adaptive Transit Signal Priority on Actuated Control Systems. *Computer-Aided Civil and Infrastructure Engineering*, 26(4):270-284. <https://doi.org/https://doi.org/10.1111/j.1467-8667.2010.00677.x>
- Li, R., and P. J. Jin. 2017. Transit Signal Priority Optimization for Urban Traffic Network Considering Arterial Coordinated Signal Control. *Advances in Mechanical Engineering*, 9(8):1687814017700594. <https://doi.org/10.1177/1687814017700594>

- Li, Y., J. He, and Y. Gao. 2021. Intelligent Traffic Signal Control with Deep Reinforcement Learning at Single Intersection. *ACM International Conference Proceeding Series*, 399-406. <https://doi.org/10.1145/3467707.3467767>
- Li, Y., M. Li, Y. Li, S. Beard, A. Skabardonis, P. Koonce, K. Zhou, W.-B. Zhang, L. Hegen, K. Hu, and Z. S. Sun. 2008. Transit Signal Priority Research Tools [Tech Report]. <https://rosap.ntl.bts.gov/view/dot/16521>
- Liang, X., X. Du, G. Wang, and Z. Han. 2019. A Deep Reinforcement Learning Network for Traffic Light Cycle Control. *IEEE Transactions on Vehicular Technology*, 68(2):1243-1253. <https://doi.org/10.1109/TVT.2018.2890726>
- Liang, Y., Z. Wu, J. Li, F. Li, and Y. Wang. 2018. Shockwave-Based Queue Length Estimation Method for Presignals for Bus Priority. *Journal of Transportation Engineering, Part A: Systems*, 144(9):04018057. <https://doi.org/doi:10.1061/JTEPBS.0000175>
- Liao, C.-F., and G. A. Davis. 2007. Simulation Study of Bus Signal Priority Strategy: Taking Advantage of Global Positioning System, Automated Vehicle Location System, and Wireless Communications. *Transportation research record*, 2034(1):82-91. <https://doi.org/10.3141/2034-10>
- Little, J. D., M. D. Kelson, and N. H. Gartner. 1981. Maxband: A Versatile Program for Setting Signals on Arteries and Triangular Networks.
- Littman, M. 2003. Friend-or-Foe Q-Learning in General-Sum Games.
- Liu, D., and L. Li. 2023. A Traffic Light Control Method Based on Multi-Agent Deep Reinforcement Learning Algorithm. *Scientific Reports*, 13(1). <https://doi.org/10.1038/s41598-023-36606-2>
- Liu, H., W.-H. Lin, and C.-w. Tan. 2007. Operational Strategy for Advanced Vehicle Location System–Based Transit Signal Priority. *Journal of Transportation Engineering*, 133(9):513-522. [https://doi.org/doi:10.1061/\(ASCE\)0733-947X\(2007\)133:9\(513\)](https://doi.org/doi:10.1061/(ASCE)0733-947X(2007)133:9(513))
- Liu, H. X., X. Wu, W. Ma, and H. Hu. 2009. Real-Time Queue Length Estimation for Congested Signalized Intersections. *Transportation Research Part C: Emerging Technologies*, 17(4):412-427. <https://doi.org/https://doi.org/10.1016/j.trc.2009.02.003>

- Liu, S., G. Wu, and M. Barth. 2022. A Complete State Transition-Based Traffic Signal Control Using Deep Reinforcement Learning. 2022 IEEE Conference on Technologies for Sustainability, SusTech 2022, 100-107. <https://doi.org/10.1109/SusTech53338.2022.9794168>
- Liu, X.-Y., M. Zhu, S. Borst, and A. Walid. 2023. Deep Reinforcement Learning for Traffic Light Control in Intelligent Transportation Systems. <http://arxiv.org/abs/2302.03669>
- Long, M., and E. Chung. 2023a. Transit Signal Priority for Arterial Road with Deep Reinforcement Learning. 2023 8th International Conference on Models and Technologies for Intelligent Transportation Systems (MT-ITS), 1-5. <https://doi.org/10.1109/MT-ITS56129.2023.10241759>
- Long, M., and E. Chung. 2023b. Transit Signal Priority for Arterial Road with Deep Reinforcement Learning. 2023 8th International Conference on Models and Technologies for Intelligent Transportation Systems, MT-ITS 2023, <https://doi.org/10.1109/MT-ITS56129.2023.10241759>
- Long, M., X. Zou, Y. Zhou, and E. Chung. 2022. Deep Reinforcement Learning for Transit Signal Priority in a Connected Environment. *Transportation Research Part C: Emerging Technologies*, 142:103814. <https://doi.org/https://doi.org/10.1016/j.trc.2022.103814>
- Lopez, P. A., M. Behrisch, L. Bieker-Walz, J. Erdmann, Y.-P. Flötteröd, R. Hilbrich, L. Lücken, J. Rummel, P. Wagner, and E. Wießner. 2018. Microscopic Traffic Simulation Using Sumo *The 21st IEEE International Conference on Intelligent Transportation Systems*, <https://elib.dlr.de/124092/>
- Ma, W., W. Ni, L. Head, and J. Zhao. 2013. Effective Coordinated Optimization Model for Transit Priority Control under Arterial Progression. *Transportation research record*, 2366(1):71-83. <https://doi.org/10.3141/2356-09>
- Ma, W., and X. Yang. 2007. A Passive Transit Signal Priority Approach for Bus Rapid Transit System. 2007 IEEE Intelligent Transportation Systems Conference, 413-418. <https://doi.org/10.1109/ITSC.2007.4357625>
- Milkovits, M. N. 2008. Modeling the Factors Affecting Bus Stop Dwell Time:Use of Automatic Passenger Counting, Automatic Fare Counting, and Automatic Vehicle Location Data. *Transportation research record*, 2072(1):125-130. <https://doi.org/10.3141/2072-13>

- Mishra, S., L. Kattan, and S. C. Wirasinghe. 2020. Transit Signal Priority Along a Signalized Arterial: A Passenger-Based Approach. *ACM Trans. Spatial Algorithms Syst.*, 6(1):Article 5. <https://doi.org/10.1145/3355611>
- Mnih, V., K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, S. Petersen, C. Beattie, A. Sadik, I. Antonoglou, H. King, D. Kumaran, D. Wierstra, S. Legg, and D. Hassabis. 2015. Human-Level Control through Deep Reinforcement Learning. *Nature*, 518(7540):529-533. <https://doi.org/10.1038/nature14236>
- Mohammadi, R., C. Roncoli, and M. N. Mladenovic. 2020. Transit Signal Priority in a Connected Vehicle Environment: User Throughput and Schedule Delay Optimization Approach. 2020 Forum on Integrated and Sustainable Transportation Systems (FISTS), 252-257. <https://doi.org/10.1109/FISTS46898.2020.9264898>
- Moosavi, S., A. Ismail, and A. Balali. 2017. Evaluating Bus Dwell Time at Key Stops Using Automatic Data Collection Systems. *Ite Journal*, 87:45-49.
- Morales, M. 2020. *Grokking Deep Reinforcement Learning*. Manning Publications.
- Muthuswamy, S., W. R. McShane, and J. R. Daniel. 2007. Evaluation of Transit Signal Priority and Optimal Signal Timing Plans in Transit and Traffic Operations. *Transportation research record*, 2034(1):92-102. <https://doi.org/10.3141/2034-11>
- National Academies of Sciences Engineering and Medicine. (2020). *Transit Signal Priority: Current State of the Practice*. <https://nap.nationalacademies.org/catalog/25816/transit-signal-priority-current-state-of-the-practice>
- Ngan, V. 2004. Impacts of Various Traffic Parameters on Transit Signal Priority Effectiveness Impacts of Various Parameters on Transit Signal Priority Effectiveness [Text]. *Journal of Public Transportation*, 7(3).
- NTCIP Signal Control Priority Working Group., American Association of State Highway and Transportation Officials., Institute of Transportation Engineers., National Electrical Manufacturers Association., and Joint Committee on the NTCIP. 2014. *National Transportation Communications for Its Protocol : Objects Definitions for Signal Control and Prioritization*. Washington, D.C.
- Rosslyn, Va. AASHTO : ITE ;

NEMA.

Pang, H., and W. Gao. 2019. Deep Deterministic Policy Gradient for Traffic Signal Control of Single Intersection. 2019 Chinese Control And Decision Conference (CCDC), 5861-5866. <https://doi.org/10.1109/CCDC.2019.8832406>

PTV. 2020. *Ring Barrier Controller User Manual*. Karlsruhe Germany. PTV AG.

PTV, A. 2021. Ptv Vissim 2021 User Manual. *PTV AG: Karlsruhe, Germany*.

Qfree. (2019a). *How to Configure Transit Priority in Maxtime*.

Qfree. (2019b). *How to Use Maxtime 2.0 as a Virtual Controller with Vissim Simulation*.

Qfree. 2020. *Maxtime 2.X Reference Guide*.

Rajbhandari, R., S. I. Chien, and J. R. Daniel. 2003. Estimation of Bus Dwell Times with Automatic Passenger Counter Information. *Transportation research record*, 1841(1):120-127. <https://doi.org/10.3141/1841-13>

Rakha, H., and Y. Zhang. 2004. Sensitivity Analysis of Transit Signal Priority Impacts on Operation of a Signalized Intersection. *Journal of Transportation Engineering*, 130(6):796-804. [https://doi.org/doi:10.1061/\(ASCE\)0733-947X\(2004\)130:6\(796\)](https://doi.org/doi:10.1061/(ASCE)0733-947X(2004)130:6(796))

Ranjitkar, P., L.-S. Tey, E. Chakravorty, and K. L. Hurley. 2019. Bus Arrival Time Modeling Based on Auckland Data. *Transportation research record*, 2673(6):1-9. <https://doi.org/10.1177/0361198119840620>

Rashid, T., M. Samvelyan, C. S. de Witt, G. Farquhar, J. Foerster, and S. Whiteson. 2018. Qmix: Monotonic Value Function Factorisation for Deep Multi-Agent Reinforcement Learning. <http://arxiv.org/abs/1803.11485>

Rashidi, S., and P. Ranjitkar. 2015. Estimation of Bus Dwell Time Using Univariate Time Series Models. *Journal of Advanced Transportation*, 49(1):139-152. <https://doi.org/https://doi.org/10.1002/atr.1271>

- Rashidi, S., P. Ranjitkar, and Y. Hadas. 2014. Modeling Bus Dwell Time with Decision Tree-Based Methods. *Transportation research record*, 2418(1):74-83. <https://doi.org/10.3141/2418-09>
- Roy, S. (2023). *Emergency Vehicle Preemption Strategies Using Machine Learning to Optimize Traffic Operations* [PhD, Georgia Institute of Technology]. 2023-04-30. <https://hdl.handle.net/1853/72099>
- Schaul, T., J. Quan, I. Antonoglou, and D. Silver. 2015. Prioritized Experience Replay.
- Schulman, J. 2015. Trust Region Policy Optimization. *arXiv preprint arXiv:1502.05477*.
- Schulman, J., F. Wolski, P. Dhariwal, A. Radford, and O. Klimov. 2017. Proximal Policy Optimization Algorithms. *arXiv preprint arXiv:1707.06347*.
- Shabestary, S. M. A., B. Abdulhai, H. Ma, and Y. Huo. 2020. Cycle-Level Vs. Second-by-Second Adaptive Traffic Signal Control Using Deep Reinforcement Learning. 2020 IEEE 23rd International Conference on Intelligent Transportation Systems (ITSC), 1-8. <https://doi.org/10.1109/ITSC45102.2020.9294171>
- Shalaby, A., and A. Farhan. 2004. Prediction Model of Bus Arrival and Departure Times Using Avl and Apc Data. *Journal of Public Transportation*, 7. <https://doi.org/10.5038/2375-0901.7.1.3>
- Sheffield, M. H., G. G. Schultz, D. Bassett, and D. L. Eggett. 2021. Sensitivity Analysis of the Transit Signal Priority Requesting Threshold and the Impact on Bus Performance and General Traffic. *Transportation research record*, 2675(5):149-163. <https://doi.org/10.1177/0361198120985853>
- Shen, W., L. Zou, R. Deng, H. Wu, and J. Wu. 2023. A Bus Signal Priority Control Method Based on Deep Reinforcement Learning. *Applied Sciences*, 13(11):6772. <https://www.mdpi.com/2076-3417/13/11/6772>
- Smith, H. R., P. B. Hemily, M. Ivanovic, ITS America., and United States. Department of Transportation. (2005). *Transit Signal Priority (Tsp) : A Planning and Implementation Handbook*. <http://www.fta.dot.gov/documents/TSPHandbook10-20-05.pdf>

- Soviany, P., R. T. Ionescu, P. Rota, and N. Sebe. 2022. Curriculum Learning: A Survey. *International Journal of Computer Vision*, 130(6):1526-1565.
- Sunehag, P., G. Lever, A. Gruslys, W. M. Czarnecki, V. Zambaldi, M. Jaderberg, M. Lanctot, N. Sonnerat, J. Z. Leibo, K. Tuyls, and T. Graepel. 2017. Value-Decomposition Networks for Cooperative Multi-Agent Learning. <http://arxiv.org/abs/1706.05296>
- Sutton, R. S., and A. G. Barto. 2018. *Reinforcement Learning: An Introduction*. 2nd ed. Cambridge, Massachusetts. MIT Press.
- Tan, C.-W., S. Park, H. Liu, Q. Xu, and P. Lau. 2008. Prediction of Transit Vehicle Arrival Time for Signal Priority Control: Algorithm and Performance. *IEEE Transactions on Intelligent Transportation Systems*, 9:688-696.
- Taparia, A., and M. Brady. 2021. Bus Journey and Arrival Time Prediction Based on Archived Avl/Gps Data Using Machine Learning. 2021 7th International Conference on Models and Technologies for Intelligent Transportation Systems (MT-ITS), 1-6. <https://doi.org/10.1109/MT-ITS49943.2021.9529328>
- Teng, K., H. Liu, and L. Rai. 2019. Transit Priority Signal Control Scheme Considering the Coordinated Phase for Single-Ring Sequential Phasing under Connected Vehicle Environment. *IEEE Access*, 7:61057-61069. <https://doi.org/10.1109/ACCESS.2019.2915665>
- Transportation Research Board, and National Academies of Sciences, Engineering,, Medicine,. 2013. *Transit Capacity and Quality of Service Manual, Third Edition*. Washington, DC. The National Academies Press. <https://doi.org/doi:10.17226/24766>
- Transportation Research Board, and National Academies of Sciences Engineering and Medicine. 2013. *Transit Capacity and Quality of Service Manual, Third Edition*. Washington, DC. The National Academies Press. <https://doi.org/doi:10.17226/24766>
- Truong, L. T., G. Currie, M. Wallace, C. D. Gruyter, and K. An. 2019. Coordinated Transit Signal Priority Model Considering Stochastic Bus Arrival Time. *IEEE Transactions on Intelligent Transportation Systems*, 20(4):1269-1277. <https://doi.org/10.1109/TITS.2018.2844199>

- Tu, T., K. Sano, and N. Y. 2012. Bus Priority Strategy Comparisons at Signalized Intersections. *Journal of Japan Society of Civil Engineers, Ser. D3 (Infrastructure Planning and Management)*, 68:I_719-I_729. https://doi.org/10.2208/jscejipm.68.I_719
- United States Department of Transportation, and Federal Highway Administration. (2009). *Manual on Uniform Traffic Control Devices*. Retrieved August 2022 from https://mutcd.fhwa.dot.gov/pdfs/2009/pdf_index.htm
- Van Hasselt, H., A. Guez, and D. Silver. 2016. Deep Reinforcement Learning with Double Q-Learning. Proceedings of the AAAI conference on artificial intelligence,
- Varaiya, P. 2013. Max Pressure Control of a Network of Signalized Intersections. *Transportation Research Part C: Emerging Technologies*, 36:177-195. <https://doi.org/https://doi.org/10.1016/j.trc.2013.08.014>
- Wang, Q., X. Yang, B. D. Leonard, and J. Mackey. 2020. Field Evaluation of Connected Vehicle-Based Transit Signal Priority Control under Two Different Signal Plans. *Transportation research record*, 2674(7):172-180. <https://doi.org/10.1177/0361198120921161>
- Wang, S., and S. Wang. 2023. *A Novel Multi-Agent Deep Rl Approach for Traffic Signal Control*.
- Wang, S., X. Xie, K. Huang, J. Zeng, and Z. Cai. 2019. Deep Reinforcement Learning-Based Traffic Signal Control Using High-Resolution Event-Based Data. *Entropy*, 21(8). <https://doi.org/10.3390/e21080744>
- Wang, Z., T. Schaul, M. Hessel, H. Hasselt, M. Lanctot, and N. Freitas. 2016. Dueling Network Architectures for Deep Reinforcement Learning. International conference on machine learning, 1995-2003.
- Wu, C., Z. Ma, and I. Kim. 2020. Multi-Agent Reinforcement Learning for Traffic Signal Control: Algorithms and Robustness Analysis. 2020 IEEE 23rd International Conference on Intelligent Transportation Systems (ITSC), 1-7. <https://doi.org/10.1109/ITSC45102.2020.9294623>
- Wu, K., and S. I. Guler. 2019. Estimating the Impacts of Transit Signal Priority on Intersection Operations: A Moving Bottleneck Approach. *Transportation Research*

Part C: Emerging Technologies, 105:346-358.
<https://doi.org/https://doi.org/10.1016/j.trc.2019.06.003>

Xin, J., and S. Chen. 2016. Bus Dwell Time Prediction Based on Knn. *Procedia Engineering*, 137:283-288.
<https://doi.org/https://doi.org/10.1016/j.proeng.2016.01.260>

Xu, H., and J. Ying. 2017. Bus Arrival Time Prediction with Real-Time and Historic Data. *Cluster Computing*, 20(4):3099-3106. <https://doi.org/10.1007/s10586-017-1006-1>

Xu, T., S. Barman, M. W. Levin, R. Chen, and T. Li. 2022. Integrating Public Transit Signal Priority into Max-Pressure Signal Control: Methodology and Simulation Study on a Downtown Network. *Transportation Research Part C: Emerging Technologies*, 138:103614.
<https://doi.org/https://doi.org/10.1016/j.trc.2022.103614>

Yang, K., M. Menendez, and S. I. Guler. 2019. Implementing Transit Signal Priority in a Connected Vehicle Environment with and without Bus Stops. *Transportmetrica B: Transport Dynamics*, 7(1):423-445.
<https://doi.org/10.1080/21680566.2018.1434019>

Yang, T., and W. Fan. 2024. Transit Signal Priority under Connected Vehicle Environment: Deep Reinforcement Learning Approach. *Journal of Intelligent Transportation Systems*:1-13. <https://doi.org/10.1080/15472450.2024.2324385>

Yu, C., A. Velu, E. Vinitsky, J. Gao, Y. Wang, A. Bayen, and Y. Wu. 2022. The Surprising Effectiveness of Ppo in Cooperative Multi-Agent Games. *Advances in Neural Information Processing Systems*, 35:24611-24624.

Zeng, X., X. Sun, Y. Zhang, and L. Quadrifoglio. 2015. Person-Based Adaptive Priority Signal Control with Connected-Vehicle Information. *Transportation research record*, 2487(1):78-87. <https://doi.org/10.3141/2487-07>

Zhai, X., F. Guo, and R. Krishnan. 2023. An Online Optimal Bus Signal Priority Strategy to Equalise Headway in Real-Time. *Information*, 14(2):101.
<https://www.mdpi.com/2078-2489/14/2/101>

Zhang, K., Z. Yang, and T. Başar. 2021. Multi-Agent Reinforcement Learning: A Selective Overview of Theories and Algorithms. *Handbook of reinforcement learning and control*:321-384.

Zhong, N., K. Liu, and Y. Li. 2023. Deep Q-Learning Network Model for Optimizing Transit Bus Priority at Multiphase Traffic Signal Controlled Intersection. *Mathematical Problems in Engineering*, 2023:9137889. <https://doi.org/10.1155/2023/9137889>

Zhou, G., A. Gan, and X. Zhu. 2006. Determination of Optimal Detector Location for Transit Signal Priority with Queue Jumper Lanes. *Transportation research record*, 1978(1):123-129. <https://doi.org/10.1177/0361198106197800116>

Zlatkovic, M., P. Martin, and A. Stevanovic. 2010a. *Evaluation of Transit Signal Priority in Rbc and Asc/3 Software-in-the-Loop Simulation Environment*.

Zlatkovic, M., P. Martin, and A. Stevanovic. 2010b. Evaluation of Transit Signal Priority in Rbc and Asc/3 Software-in-the-Loop Simulation Environment *Transportation Research Board 89th Annual Meeting*, Washington DC, United States.