

UC Riverside

UC Riverside Electronic Theses and Dissertations

Title

A Distributed Approach to Dynamic Routing Algorithms Based on Vehicle to Vehicle Interaction

Permalink

<https://escholarship.org/uc/item/7003x00d>

Author

Boskovich, Scott Matthew

Publication Date

2014

Peer reviewed|Thesis/dissertation

UNIVERSITY OF CALIFORNIA
RIVERSIDE

A Distributed Approach to Dynamic Routing Algorithms
Based on Vehicle to Vehicle Interaction

A Dissertation submitted in partial satisfaction
of the requirements for the degree of

Doctor of Philosophy

in

Electrical Engineering

by

Scott Matthew Boskovich

March 2015

Dissertation Committee:

Dr. Matthew J. Barth, Chairperson
Dr. Christian Shelton
Dr. Qi Zhu

The Dissertation of Scott Matthew Boskovich is approved:

Committee Chairperson

University of California, Riverside

Acknowledgements

I want to thank Dr. Matthew Barth for the guidance and support over the many years while at UCR and participating with the research at the Center for Environmental Research and Technology. His experience in the subject matter has been incredibly helpful, and inspiring. I especially want to thank him for challenging me to think about new methods to solve the problems within ITS and expanding the concepts with new applications. His guidance on developing papers while learning what others are researching has been very helpful towards me completing my efforts.

I also want to thank my family, Sean, Nicole, and especially my wife Katherine, for the countless long days and nights in my pursuits of new ideas and research. Their support has been an inspiration during those long hours.

Finally, I want to thank Mr. Mike Todd, Dr. Kanok Boriboonsomsin, Dr. George Scora, Dr. Guoyuan Wu, Ms. Qiu Jin, Mr. David Kari and the staff at CE-CERT for their support and discussion about the simulations, dialog about new methods and techniques for research and analysis in this subject matter. These discussions helped frame the context of several ideas presented in this research.

ABSTRACT OF THE DISSERTATION

A Distributed Approach to Dynamic Routing Algorithms
Based on Vehicle to Vehicle Interaction

by

Scott Matthew Boskovich

Doctor of Philosophy, Graduate Program in Electrical Engineering
University of California, Riverside, March 2015
Dr. Matthew J. Barth, Chairperson

Intelligent Transportation Systems (ITS) allow for broad implementation approaches regarding the distribution of data and the processing of that data for use in a particular framework. In this dissertation, analysis and approaches in describing methods of vehicular interaction incorporating the architectures are explored. These architectures include centralized and decentralized designs, and can be applied to a number of ITS applications, such as the distributed routing problem.

Within ITS, concepts of vehicle communication with other vehicles and the traffic systems infrastructure are understood. However, the concepts of vehicle-to-vehicle (V2V), vehicle-to-infrastructure (V2I), and infrastructure-to-vehicle (I2V) that describe the relationships between how the data are captured and processed using these techniques

remains vague. This dissertation provides a distinction and relationship between vehicle and infrastructure architecture topologies. When considering the distributed architecture, many researchers have presented the relationship of the vehicles and a parallel to multi-agent system. This concept is explored in this dissertation and extended to recognizing that fundamental to V2V distributed approaches is the notion that the interaction between the agents is vital to ensuring that the agents are capable of interacting successfully. This dissertation also explores the method of implementation the communication between the vehicles from a perspective of programming the ITS algorithms.

As a key element of this dissertation, the developed distributed architectures are applied to the distributed vehicle routing problem. The distributed routing problem is the notion that vehicles share their current state while on the road with other vehicles within its vicinity. As the data are shared, vehicles are able to assess the shared information and make alterations to their current route to reduce traffic congestion while improving their travel time. This research work examines and compares the effect on transportation mobility when incidents occur when: 1) ITS concepts are not used at all; 2) when ITS concepts are used based upon a centralized architecture; and 3) with a decentralized architecture. Simulations of both hypothetical and actual road data base simulations are performed comparing the various architectural approaches.

TABLE OF CONTENTS

Table of Contents	i
List of Figures	ix
List of Tables	xii
Chapter 1 Introduction	1
1.1 Problem Statement	2
1.2 Centralized Traffic Management Center (TMC).....	5
1.3 Decentralized Techniques	10
1.3 Contributions	12
1.4 Organization	13
Chapter 2 Related Work and Background	15
2.1 Intelligent Transportation System Architectures.....	16
2.1.1 Centralized ITS Architecture Approaches	17
2.1.2 Distributed Architectures and Multi-Agent Systems	23
2.2 Routing techniques	23
2.3 Multi Agent Systems and Agent Behaviors in ITS	25
2.4 Functional Reactive Programming	29
Chapter 3 Centralized and Decentralized Classification	33
3.1 Overview	33
3.2 Centralized and Distributed ITS Topology Classifications	34
3.2.1 Cell 1 Topology (C,C) – Centralized Data Acquisition and Centralized Processing	36
3.2.2 Cell 2 Topology (CD,C)	38
3.2.3 Cell 3 Topology (D,C)	40
3.2.4 Cell 4 Topology (C,CD) Centralized Data Acquisition and Centralized / Decentralized Processing	41
3.2.5 Cell 5 Topology (CD,CD).....	43
3.2.6 Cell 6 Topology (D,CD) Decentralized Data Acquisition and Centralized / Decentralized Processing	44
3.2.7 Cell 7 Topology (C,D) Centralized Data Acquisition and Decentralized Processing.....	46

3.2.8 Cell 8 Topology (CD,D) Centralized / Decentralized Data Acquisition and Decentralized Processing	48
3.2.9 Cell 9 Topology (D,D) Decentralized Data Acquisition and Decentralized Processing	50
Chapter 4 Multi-Agent Approach in A Distributed Architecture	53
4.1.1 Type A – Arbiter Prototype.....	56
Type B – Arbiter Prototype	57
Type I - Formation Flow	57
Type II – Attraction / Detraction	58
Type III – Environment Flow.....	58
4.3 Vehicle Interaction	69
4.4 Vehicle Cooperability	70
4.5 Cooperable Interaction Definitions	75
4.5.1 Cooperable Interaction Definitions	76
4.5.2 Action Segments and Plans and Turtle Plots	79
4.6 Application of Functional Reactive Programming Paradigm	85
Chapter 5 Implementation – Application to Distributed Routing.....	87
5.1 Centralized Routing Algorithm	87
5.2 Decentralized Routing Algorithm	90
5.3 Implementing FRP for Decentralized Vehicle Routing	93
Chapter 6 Results	98
6.1 Simulation Environment - SUMO	99
6.2 Typical Road networks	106
6.3 Modeling Existing Road-Network CA SR-60.....	108
6.4 SIMULATIONS RESULTS of Road Networks.....	110
6.4.1 Non-ITS Based Results for a Hypothetical Road Network.....	110
6.4.2 Applying ITS for Dynamic Routing – Centralized for Hypothetical Road Network	114
6.4.3 Applying ITS for Dynamic Routing - Decentralized	118
6.4.4 Applying Mixed Centralized and Decentralized Based Results for a Hypothetical Road Network.	122
6.5 Simulations for Example Networks.....	126

6.5.1 CA SR-60 Non-ITS Based Results.....	126
6.5.2 CA SR-60 Centralized-ITS Based Results	130
6.5.3 CA SR-60 Decentralized-ITS Based Results	134
6.5.4 CA SR-60 Mixed Centralized and Decentralized ITS Based Results	137
6.5.5 CA SR-60 ITS Based Results	138
Chapter 7 Conclusions and Future Work.....	142
Chapter 8 Bibliography	146

LIST OF FIGURES

Figure 1-1. Typical Traffic Management Center Implementation	8
Figure 2-1. NTCIP Communications model between transportation elements, sensors and infrastructure	18
Figure 2-2. Relates the various elements within the NTCIP to one another via the network configurations	19
Figure 2-3. NTCIP Function versus protocol between vehicles, infrastructure, and field processing equipment	20
Figure 3-1. Cell 1 - (C,C) Fully centralized ITS topology for both data acquisition and processing	38
Figure 3-2. Cell 2 - (CD,C) Centralized and Decentralized Data Acquisition, fully centralized decision processing ITS Topology.	40
Figure 3-3. Cell 3 - (D,C) Decentralized data acquisition, fully centralized decision processing ITS Topology.	41
Figure 3-4. Cell 4 - (C,CD) Centralized data acquisition, and centralized and decentralized decision processing ITS Topology	42
Figure 3-5. Cell 5 - (CD,CD) Centralized and decentralized data acquisition, and processing ITS Topology	44
Figure 3-6. Cell 6 - (D,CD) Centralized data acquisition, and centralized and decentralized decision processing ITS Topology	46
Figure 3-7. Cell 7 - (C,D) Centralized data acquisition and decentralized decision processing ITS topology.	48
Figure 3-8. Cell 8 - (CD,D) Centralized and decentralized data acquisition, decentralized decision ITS Topology.	49
Figure 3-9. Cell 9 - (D,D) Decentralized data acquisition, with decentralized decision processing ITS topology.	50
Figure 4-1 Generic FSM Structure for state behavior decompositions	55
Figure 4-2 Example of Agents behaviors defined using Type A and Type B behavior prototypes.....	55
Figure 4-4 Sample road network where vehicles reroute around the incident shown in red.....	61
Figure 4-5 Evolution of state sequences for a small passenger car using behavior prototypes	65
Figure 4-6 Evolution of state sequences for a small passenger car using behavior prototypes.....	66
Figure 4-7a,b Half perimeter path 1 vehicle / Half perimeter path 2 vehicles.....	71
Figures 4-8a-c Intended Path / Intended Path - no cooperation / Updated Path with cooperation.....	72
Figure 4-9 – Basic Turtle Plot	81
Figure 4-10. Example of two turtle plots that will have their slots compared.	82
Figure 5-1. Distributed Vehicle Algorithm State Machine	94
Figure 5-2. Grouped Distributed Vehicle Algorithm State Machine.....	95

Figure 5-3. Behavior Distributed Vehicle Algorithm State Machine	96
Figure 6-1. Basic hypothetical road network using SUMO for simulation.....	100
Figure 6-2. Basic SUMO link with a vehicle approaching and loop detectors at an intersection.....	101
Figure 6-3. SUMO intersections	101
Figure 6-4. Primary Freeway link with loop sensors used in centralized solution	102
Figure 6-5. Freeway off-ramps to the arterial routes.	102
Figure 6-6. SUMO Hypothetical simulation environment used to model the simulation.....	104
Figure 6-7. CA SR-60 Corridor Ramona to Euclid used for simulation and analysis for routing.....	105
Figure 6-8 CA SR-60 Corridor with distances indicating the available routes	105
Figure 6-9 Eastern Orange County CA SR-91 Corridor	107
Figure 6-10 Central Orange County CA SR-22	108
Figure 6-11 Volume of traffic along Corridor CA SR-60 during the month of April	109
Figure 6-12. Histogram illustrating the times to travel from the origin prescribed destination using Greedy Routing Algorithm and no ITS methods were employed	112
Figure 6-13 a-f Simulation Capture of Hypothetical Road Network without using an ITS Architecture Figure 6-13a illustrates no congestion. For each successive screen capture, the congestion becomes worse indicate by the red colored links in the final screen capture.....	113
Figure 6-14. Histogram illustrating the times to travel from the origin prescribed destination using the centralized ITS Routing Algorithm.....	115
Figure 6-15 a-f Simulation Capture of Hypothetical Road Network using a Centralized ITS	117
Figure 6-16. ITS Histogram illustrating the times to travel from the origin prescribed destination using the decentralized ITS Routing Algorithm.....	119
Figure 6-17 a-f Simulation Capture of Hypothetical Road Network using a Decentralized ITS.	121
Figure 6-18. ITS Histogram illustrating the times to travel from the origin prescribed destination using the combined centralized and decentralized ITS Routing Algorithm.	123
Figure 6-19 Average Travel Times for rerouting with various loading. The horizontal axis groups the travel times by arterial loading.....	124
Figure 6-20 Average Travel Times without route information methods for rerouting with various loading.	127
Figure 6-21. a-c Simulation Capture of Hypothetical Road Network without using a ITS Architecture.	129
Figure 6-22 Average Travel Times with centralized route information methods for rerouting with various loading	131
Figure 6-23 a-c Simulation Capture of Hypothetical Road Network using a Centralized ITS Architecture. The location of the incident is indicated by the red circle.....	133
Figure 6-24 Average Travel Times with Decentralized route information methods for rerouting with various loading	134

Figure 6-25 a-c Simulation Capture of Hypothetical Road Network using a Centralized ITS Architecture. The location of the incident is indicated by the red circle.....	136
Figure 6-26 Average Travel Times with Centralized and Decentralized route information methods for rerouting with various loading	138
Figure 6-27 Comparing the average travel times through the CA SR-60 corridor	139

LIST OF TABLES

Table 3-1 ITS Classification Matrix differentiating location of data acquisition and processing	35
Table 3-2 Legend for ITS Classification Matrix Illustrations	36
Table 3-3 Summary of Matrix of ITS Comparing Topologies.	51
Table 4-1. Interaction Properties.....	78
Table 6-1. Average Re-route time vs Technique.	123
Table 6-2 Average Re-route time vs Technique.....	140
Table 6-3 Comparing the Hypothetical with the CA SR-60 Road Network..	141

Chapter 1 INTRODUCTION

Intelligent Transportation Systems are evolving at an ever increasingly rapid pace. Applying advancements in sensors, communication technology, and processing capabilities can now provide the driver and vehicle with increased information about the state of current road conditions. This allows for significant efficiencies, for example a driver no longer needs to arbitrarily select which route to take, and/or which route has the least amount of traffic congestion. With additional road condition information, the driver can assess which route will get him to his destination in either the shortest amount of distance and/or least amount of time. However, once a driver is in-route to his destination, the traffic conditions may abruptly change (e.g., a traffic accident may occur). The question becomes, what would be the *best* alternative to his current route when something occurs inducing significant and sudden congestion. If vehicles on a common section of road alter their routes to reroute around the sudden congestion, and each vehicle is using similar data and algorithms without knowledge of the other vehicles intents, then the resultant action is transference of congestion to the new route. The net effect results in no improvement in transit time around the incident.

1.1 PROBLEM STATEMENT

As traffic road network data becomes more readily available via traffic databases such as the Caltrans Performance Monitoring System (PeMS) through the use of embed road loop sensors [1], the desire to make use of these data to provide the driver with real time road network data to help alleviate congestion has also grown [2]. This congestion can be caused by many reasons. Furthermore, making an informed decision about which route to take after a sudden traffic incident is not simply a matter of knowing the alternative routes, or even the current loading for a given route. Instead a driver could make a *more informed decision* by gaining better situational awareness by knowing the decisions from the *other* surrounding affected drivers. Furthermore, if the all of the surrounding drivers knew of each other's route selection and were able to coordinate their routes, they could significantly lessen the congestion impacts on the alternative routes. This is commonly known as the distributed or dynamic routing problem [2].

This dynamic routing concept is explored from many perspectives in this dissertation including: 1) where the decisions can and should take place, namely by the driver or vehicle itself in a informed distributed manner versus using a classical centralized data capture and processing approach; and 2) knowing what the impacts of how to make the decisions, and exploring the decision process

techniques. This dissertation investigates whether a decentralized approach is beneficial, and explores both centralized and decentralized implementations in detail.

To explore these dimensions of the problem stated above, this dissertation examines solutions from multi-agent robotics, system operations research, distributed mobile-computing devices, and emerging programming paradigms, all of which help provide a solution. This dissertation ties together different concepts to illustrate a viable approach such that the driver has available to him the tools to capture situational awareness and provide better achievable route options. There are many aspects and motivations that contribute to pursue and attain these possibilities.

As mentioned previously, one of the motivations for integrating these technologies into a single solution is to provide context-sensitive information to the driver as he travels from his origin to his destination. During this travel, the driver is faced with ever increasing traffic densities often caused by a traffic incident with unknown frequencies of occurrence thereby causing increased delays in travel as well as higher fuel consumption and emissions, or even possibilities for further incidents [3]. If the driver knows of alternative routes that are conducive to his type of

vehicle, then he would be more inclined to pursue an alternate route versus maintaining his original route and waiting out the incident. The route options may include taking routes that contain arterial roads or alternative freeway routes. However, in the pursuit of taking an alternate route, the driver would want to ensure that other vehicles are not all making the same choice where the traffic congestion is not simply moved to his new route by the other driver's actions.

When choosing an alternate route, simply knowing the current traffic conditions of the roads at a given point in time is not sufficient to make an informed decision. Instead, in the event a sudden incident occurs, each driver typically performs the same analysis of the current conditions. Furthermore, in general, most drivers then seek an alternate route usually based upon the shortest time to arrive at similar destinations. With today's navigation technology, drivers will use similar routing tools and software thereby making similar routing selections based on using the same information and techniques. This often causes a transfer of traffic congestion from the original incident location to another location of a common reroute. To prevent this, vehicles must be aware of their own route and reroute in such a way this new route is not necessarily the same as the other vehicle's routes, based on cooperation. To achieve this, the decision where each vehicle shall reroute to must be performed such that the alternate routes spread out the congestion using the available road network capabilities.

Fundamentally this process of route selection decision and allocation can occur using either a centralized or a decentralized architecture. In centralized approach, traffic data (e.g. speed, density, and/or flow) are collected from one or more sources within the road network and routed through a data network to a central processing center. Currently, the data are typically captured using embedded loop sensors [1] that determine traffic densities and speeds for a given road segment. The data are then relayed through a hierarchy of data networks to a traffic management center. Once the raw data are captured, they are then aggregated to provide a composite picture of the state of the road network. The data are then pulled by the user to determine a set of routes based on their current location to their final destination.

1.2 CENTRALIZED TRAFFIC MANAGEMENT CENTER (TMC)

Using the traditional centralized approach carries with it a significant overhead and set of assumptions. This primarily is the notion that traffic information is up to date, and that there is sufficient coverage from the acquisition via embedded loop detectors or other sensors. Typically the sensors can be faulty (e.g., embedded loop sensors often become damaged) and network connectivity can be intermittent. Secondly, there is the assumption that there is sufficient coverage by the traffic sensors themselves.

Aside from the physical acquisition of the road network data, another challenge is to provide drivers with specific instructions to navigate around a congested region. Currently, although the data are available and provided to the driver, the driver is left to his own knowledge or set of decisions to best reroute around the incident. A TMC currently does not specifically notify each driver as to their personal alternative route around the incident. The driver is then left with finding his own best route which all other drivers are doing, thereby moving the congestion to another location. Ideally, the typical proposed centralized solution would provide specific routes to the drivers them from all making their own decisions. Currently to support this type of existing centralized solution once the data are collected the aggregation of the data becomes important.

Logistically within the centralized approach, support of these systems becomes complex and expensive to maintain. As hardware fails in the harsh conditions of being outside, newer hardware designs must be implemented. While this provides the opportunity to keep up to date with new capabilities, it also carries with it an element of supportability to older aging systems that must maintain legacy interfaces with protocol's and network physical interfaces.

Finally, the TMC must maintain a large computing center to process the incoming data and monitor overall traffic conditions. This type of center is akin to a military war room filled with computers / monitors and people to maintain this type of equipment. Currently TMCs are unable to directly address any one individual vehicle to provide guidance of routing to prevent incident moving. Instead, currently, each vehicle simply receives the information regarding the current road conditions from the TMC to make new route decisions [4]. Given that people want to generally get from where they are in the safest fastest manner possible, the algorithms employed are greedy minimum path algorithms based on solutions such as the Dijkstra Routing Algorithm [5]. In this case, the device directs them to the best available route without knowledge of what the other routes the other vehicles intend on taking.

When analyzing this centralized approach, data that are captured by given networks often are not aggregated with other networks. Furthermore, until very recently, data from a freeway system are not directly integrated with arterial roadway network data from cities or regions. Therefore, any arterial alternative routes are not considered when the state of the network is assessed. Instead regional information is not known by any one agency [6]. It should be noted that road network information that is multisource is nature and provides arterial road data integration is starting to change. Propriety solutions available from companies such as *Inrix*, [7]

and *Sigalert* [8] have begun to introduce data that provide information blended from different sources. However, since these implementations are propriety, their algorithms are not well known [7], [8].

The Typical TMC in use today to capture and process current road network conditions is shown below in Figure 1-1.

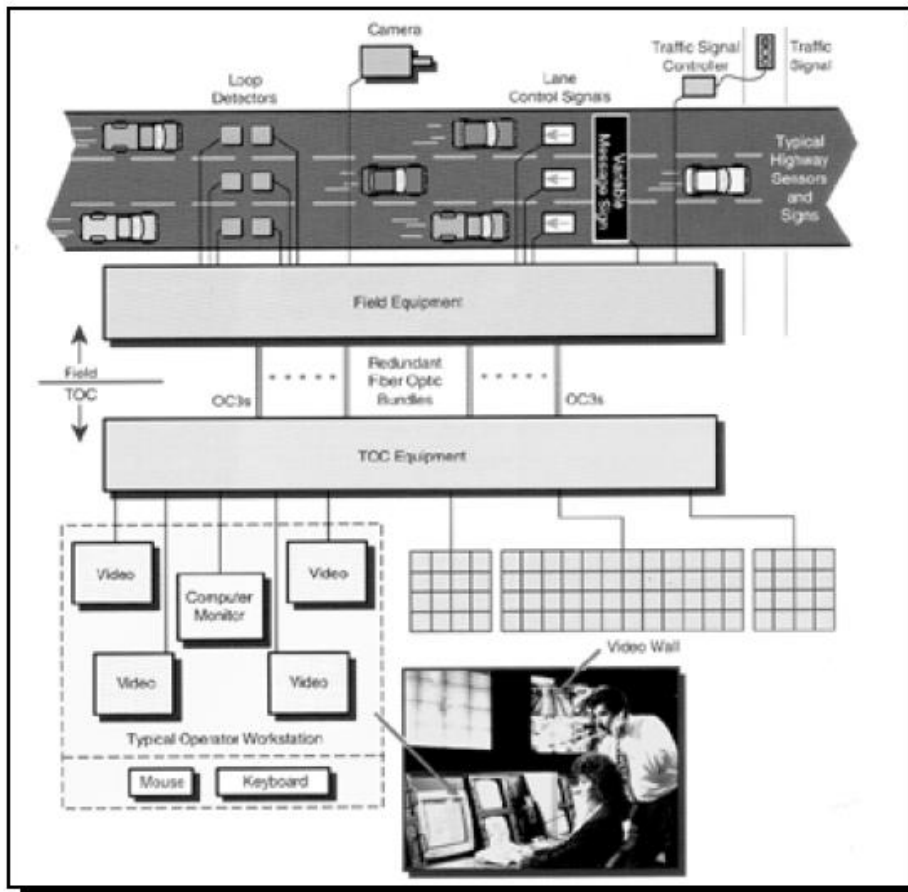


Figure 1-1. Typical Traffic Management Center Implementation [9]

A TMC consists of many elements to be able to support the breadth of road network status data capture and processing. As shown in Figure 1-1, the TMC receives data about the status of the road network typically using embedded loop detectors or other various sensors such as camera systems, and radar detectors [9]. The loop detectors detect the occurrence as vehicles pass over them and relay this information to field equipment processing units that captures the data from the loop detectors for a given link. This link data are aggregated in the field equipment which then relays it to the TMC. The road network also may contain intersections controlled using signal controllers which can also provide information to the TMC. Data are processed by the TMC and is then relayed to the drivers using electronic highway message signs providing the driver about possible road conditions, and advisement to alternative routes.

The TMC implementation supports and requires significant data propagation layers as indicated in [9]. This approach uses an open-system, systems engineering approach that implements well established protocols, and networking layers. This model of an integrated system and infrastructure solution is implemented in many suburban environments in the U.S. including many Southern California Caltrans Districts that include the metropolitan Los Angeles area, and surrounding regional vicinities.

In contrast to this contemporary scenario, in this dissertation, we present alternative solutions where the data of the road network is captured locally in real time and relayed to the surrounding vehicles that require the information. Two alternate approaches are provided whose merits are discussed and contrasted. In these solutions, an extended centralized method is presented as well as a fully decentralized method. In the decentralized approach, there is no central data aggregation. Instead the data are collected by the vehicles themselves processed and passed along to other surrounding vehicles. The decisions are computed locally within each vehicle as to where the best route to take goes.

1.3 DECENTRALIZED TECHNIQUES

Implementing a decentralized approach provides several advantages, namely in that the vehicles can serve as the traffic sensors themselves. Therefore, a system of sensors, data aggregators, or traffic management centers are all not required. Furthermore, the vehicles that are interested in the data for the given vicinity are able to access it [10]. Additionally, since there are no networks from varying municipalities that require interaction and/or interfacing, vehicles on the arterial roads are able to freely communicate with the vehicles on the freeway and vice versa to gain a better assessment of the current state of the road networks in the vicinity.

The decentralized approach borrows concepts from distributed multi-agent robotics. The methods used in this approach are based upon defining vehicle actions on behavioral algorithms. However, this approach is not without its own consequences as will be discussed further. Within the multi-agent approach, the decision processing can become a challenging problem when the vehicles are required to process data as it arrives in an asynchronous manner, yet must still process the data in near real time using synchronous mechanisms. When investigating this element of the solution, it became clear that what is actually being considered understands how the vehicles interact with one another and how they collaborate with one another in order to arrive at a solution for routing that is conducive for all vehicles.

Finally, recognizing that fundamental to distributed vehicle architectures is understanding how the vehicles interact with one another. Also, many times point solutions are able to be developed that include the interactions of the vehicles, however there lacks a framework that defines the specifics of the relationships between these interactions. The approaches discussed in this dissertation provide a means to evaluate the distributed approach versus the centralized architectures. These approaches provide the necessary framework to support the vehicle interactions constructs. Furthermore, the distributed routing solution is provided

as an example of vehicle interactions. These considerations discuss the challenges associated with using a distributed approach while further provided solutions to these challenges.

1.3 CONTRIBUTIONS

Within this dissertation there are several new contributions provided to the areas of ITS. This dissertation provides an analysis and approach that uses distributed processing methods that capture new formalism to aid in understanding the various relationships between centralized and decentralized ITS architectures. It also presents a distributed routing technique that takes advantage of the heterogeneity of the vehicle types on a road network versus treating all of the vehicles as the same. It recognizes that key to distributed vehicle routing is the underlying interaction relationships of the vehicles themselves. Within this perspective, a new framework that recognizing vehicle interaction is presented. Finally, it applies the programming paradigm of functional reactive programming within the context of ITS as an approach to address limiting implementations that would otherwise be associated with traditional programming approaches. Several key ideas including the technique of differentiating vehicle types and a using specific agent behaviors have been presented in two IEEE ITS conference papers, as well as in a submission

transactions paper relating new programming methods for use in these applications.

1.4 ORGANIZATION

The dissertation is organized as follows: Chapter 2 provides a discussion on the background. It presents in detail the traditional centralized TMC. Chapter 2 then discusses the background towards the distributed approach. Next the relationship between the distributed ITS, and background in multi-agent systems (MAS) applied towards ITS is presented. Finally in support of the multi agent systems a programming paradigm known as Functional Reactive Programming (FRP) is presented as an approach supporting the distributed and MAS ITS approaches. Chapter 3 presents a discussion on the formalization of the relationships of centralized and decentralized architectures, and presents a method of categorizing these relationships. Chapter 4 introduces a new framework to understand the interactions of the vehicles in a distributed vehicle network as well as introduces the notion of cooperability amongst the vehicles. In Chapter 5 presents an application of distributed architectures with an implementation of distributed routing. Chapter 6 presents the simulation environments and simulation of a hypothetical road network and actual road network. Within these simulations, various approaches of architectures were implemented to analyze the differences in

travel time when an incident occurs, and the results of the simulation. Finally, in Chapter 7, the conclusions are presented that unite the various aspects of the work as well as present insights into future work.

Chapter 2 RELATED WORK AND BACKGROUND

Intelligent Transportation Systems (ITS) include some very recent advances and concepts that deal with the communication of vehicles between one another while also using available resources to determine the best alternative routes under current levels on congestion. However implementing these algorithms also requires methods and techniques that are sufficient to provide the necessary structure. Recently there have been newer models of programming that support some of the requirements necessary for principles that will be provided as approaches detailed in this dissertation. This chapter summarizes the material for the state of the art of current ITS architectures, and routing approaches, while also providing the background to support the material discussed in subsequent chapters. Section 2.1 provides a discussion on the current ITS national architecture, while section 2.2 discusses the various current approaches to using various architectures and routing techniques with available resources. Section 2.3 provides an introduction to the programming paradigm of Function Reactive Programming while Section 2.4 provides a brief analysis of using functional reactive programming.

2.1 INTELLIGENT TRANSPORTATION SYSTEM ARCHITECTURES

Intelligent Transportation Systems rely on the underlying connectivity of vehicles and roads network infrastructure and their interconnectedness. Fundamental to any ITS approach is the acquisition of information and the distribution of that data to provide a higher degree of awareness to those using the transportation resources. The architecture employed to acquire and distribute this information becomes the subject of many areas of research within ITS. Furthermore, the structure of ITS, and the acquisition and dissemination of the information as can reside in a centralized architectural approach. However due to the increased capabilities of vehicle processing and mobile processing, the requirement to host the processing within in a large central center can be migrated to the individual vehicle [11].

2.1.1 CENTRALIZED ITS ARCHITECTURE APPROACHES

Currently the basis for the ITS solutions implemented in the US today is outlined in the national ITS architecture and compose of many agencies and layers [12]. Within this architecture exists a definition of the integration of how sensors, computing centers and users interact with the data and it's the collection process. This is outlined in the National Transportation Communications for ITS Protocol (NTCIP) family of standards. One of the tenets of the national ITS architecture is that of open standards to allow for systems to inter-operate with one another without requiring proprietary protocols or hardware. An example of this is the transport layer using TCP/IP. The frameworks of these elements are defined by the NTCIP. The NTCIP also defines a family of general purpose communications protocols and transportation specific data dictionaries to support the computer hardware for computer systems including those located along the roadway for capturing link data and the processing centers. The NTCIP also and provides the definition for interoperability within the data communications architecture for traffic management. This includes vehicular traffic as well as other modes of transportation.

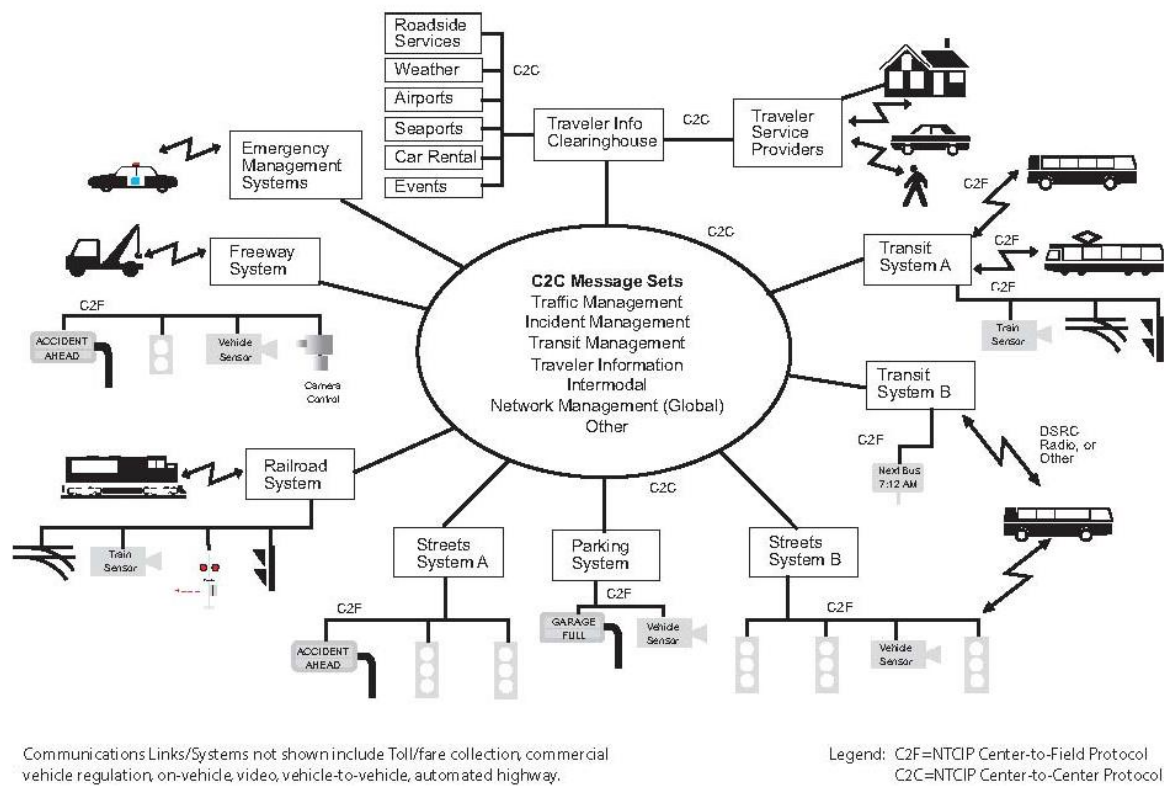


Figure 2-1. NTCIP Communications model between transportation elements, sensors and infrastructure [12]

As stated earlier, the NTCIP is a family of communications standards for transmitting data and messages between computer system used in ITS as shown in Figure 2.1. It describes the communication between TMCs referred to by the NTCIP as Center to Center (C2C) data, as well as Center to Field (C2F) which is referred to as TMC to equipment in the field. This includes loop sensors attached to embedded roadside processing, closed circuit television, or dynamic message signs providing drivers with road closures and various status messages.

Systems that can take advantage of NTCIP are:

- Dynamic Message Signs
- Traffic Signals
- Field Masters (closed Loops Systems)
- Data collection and monitoring devices such as traffic counters and classifiers
- Environmental Sensors
- Ramp Meters
- Vehicle Detectors
- CCTV (cameral control only)
- Video switches
- Highway Lighting Control

The layers of the network include five distinct layers and the functions shown in Figures 2-2 and 2-3 define how the entities within the NTCIP relate to one another.

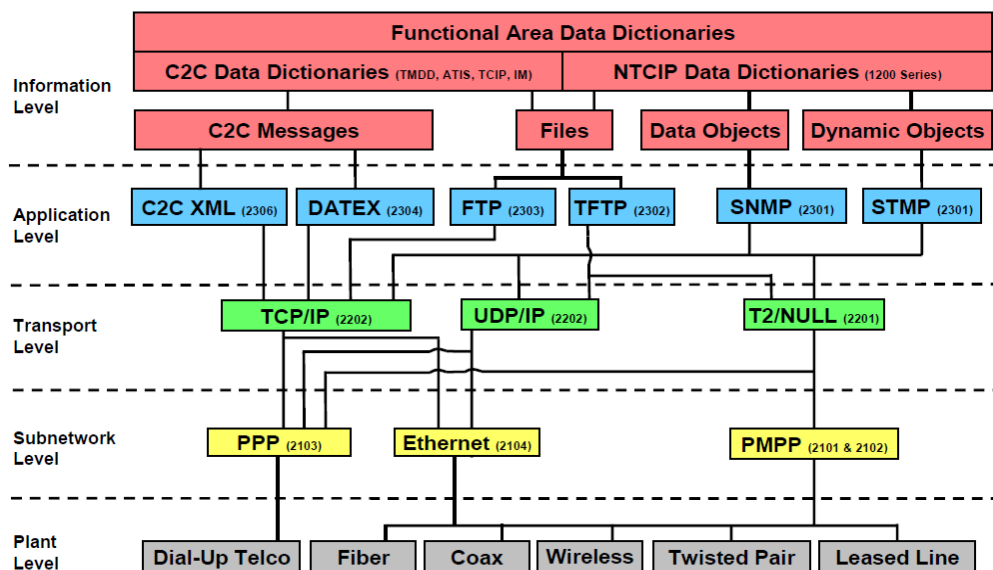


Figure 2-2. Relates the various elements within the NTCIP to one another via the network configurations [12].

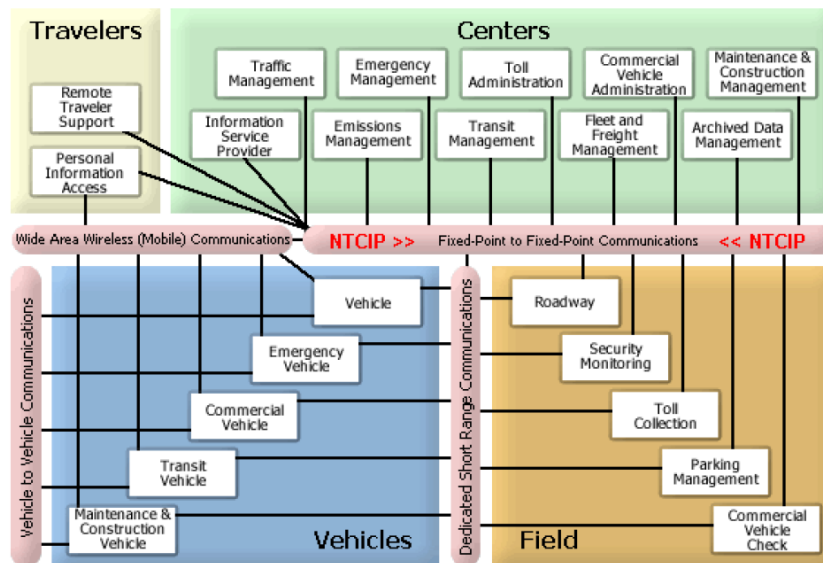


Figure 2-3. NTCIP Function versus protocol between vehicles, infrastructure, and field processing equipment [12]

Examining the ITS architecture material illustrates there is a significant infrastructure to capture road network conditions on the freeway system. This includes arrays of loop sensors, data collection hardware, hierarchies of data collection networks that feed into various centralized traffic management centers. Unfortunately this data captures real-time data regarding the freeway system, and combining this data with road conditions that are arterial to the freeways is problematic due to the different centers and agencies that must participate within the architecture. There is no certainty that each center provides support to other entities due to cost of implementation or maturity of the region. Additionally there becomes a significant cost to maintain the data collection hardware outlined in the [12].

Fortunately, today cars and travelers are able to capture their position and velocity information by using current handheld devices and integrated in-vehicle informatics that implement maps and GPS technologies, and also as in the case of in-car informatics, vehicle speed information is also available from the vehicle's primary Electronics Control Unit (ECU) [6][13]. Travelers are able to plot out their route using these capabilities and devices to find the optimal route based on their origin and their desired destination [10].

When considering what route best to take, one aspect to consider is whether these greedy routing problems should be solved using an extension of the current ITS centralized architecture or a de-centralized architecture. In a centralized architecture, data are gathered by the vehicle and/or road sensor network, processed centrally followed by determining a set of actions, and then commands are distributed to individual vehicles. In contrast, in a de-centralized system, the information is gathered locally, communication occurs between local vehicles, and then actions are assigned by the vehicles themselves. When considering using the centralized solution for the individual vehicles it becomes a rather daunting task to articulate specific vehicles to specific routes when hundreds to thousands simultaneously are involved and the computations are to be done in a near real time manner.

Efforts presented in [12] discuss a method of comparing centralized and decentralized methods with the aid of a database of traffic patterns. The vehicles are either connected to a centralized system that contains an active database of link performances, or they are connected to one another in a Vehicle to Vehicle approach illustrating that the distributed approach improved travel times. This work found that using a distributed approach performed better of the centralized approach.

2.1.2 DISTRIBUTED ARCHITECTURES AND MULTI-AGENT SYSTEMS

When considering the distributed architectures for ITS, in this architecture each vehicle is considered to be its own processing agent. As discussed in [10] a method to utilize vehicle processing as the agent is presented. The term agent is used to denote an intelligent actor that interacts with its environment by means of a sensor and actuator. Therefore a multi-agent systems is one where a system is comprised of several agents that work together in some capacity [10]. Recognizing that each vehicle is an agent, the vehicles interact through some means of communication between one another. Because of this model of communication, this approach is often referred to as Vehicle-to-Vehicle or V2V for short. Because of the ad-hoc structure of the communications between the vehicles, this notion is also considered as a Vehicular ad-hoc Network or simply VANET [14]

2.2 ROUTING TECHNIQUES

When a driver is presented with decision for a given route to take, most often, the decision process to determine the best route is the fastest available. Several strategies exist to determine what route to take. Overall, the decisions can be categorized into three general approaches. The work in [14] categorizes classic dynamic route planning as: Optimal, Heuristic and Hybrids. Using this method of

classification, sub-classifications in the Optimal approach include the Dijkstra and Incremental Graph[15] , Heuristic Algorithms that include the traditional A* Algorithm and the newer methods of that include Genetic Algorithms[16], as well as the Ant Colony Optimization [17] and the Tabu search methods[18]. There are others that exists such as the D* which are variants of those listed. However these routing techniques do not specify where the processing for the routes exists nor do they capture the varying conditions of congestions at any point in time.

Other methods, such as those discussed in [19] and [20] investigate methods that use a geometric approach based on Voronoi cells. In these approaches the approach in [19] Voronoi cells are used to allocate neighbor regionality to share changing routes. This approach provides discussion to the application of routing with awareness by each vehicle of the surrounding vehicles and provides a technique for decentralized coordination. The discussion in [20] presents a mathematical approach to understanding modeling and optimizations while illustrating successful methodologies using Voronoi geometries.

Applying the notion of distributed methods is becoming a current topic for routing vehicles within ITS. As discussed in [21], the concept of available route selection is based upon a *Social Welfare*. In this approach, the available routes are selected

based upon the assessment of *Fairness* where the fairness is calculated by bidding for a path. Each vehicle submits a bid. When all vehicles have submitted bids, the vehicle with the highest bid wins the path. Another approach discussed in [22] presents a method that if a vehicle's velocity is below a given threshold, then the vehicle will change its route to one that has fewer vehicles, but also increases communication coverage between the various vehicles within the network. In doing so the new route provides better connectivity coverage between the vehicles. Finally in [23], the technique of applying genetic algorithms for providing electric vehicles with routes that are optimized to provide charging stations as they vehicles travel throughout their intended routes. Routes are selected on the availability of the resource to charge the vehicle when the vehicle would otherwise arrive at the charging station and the charging station is available while also selecting a route that maximizes the available charge of the vehicle. Each one of these approaches relies upon vehicles communicating their state, and intended purpose to the vehicles surrounding it.

2.3 MULTI AGENT SYSTEMS AND AGENT BEHAVIORS IN ITS

In this section, multi-agent systems and their approach to implement a general solution for vehicles in a distributed architecture is presented. As described in [11], and [24], traffic networks fit naturally into the multi-agent control paradigm. Using the method, the vehicles in the distributed architecture are considered to agents

within the multi-agent system. In this approach, agents possess a set of rules that describe a behavior for an agent to execute under certain circumstances. A behavior is defined as being a mapping of sensory inputs to a pattern of actions that in turns performs a specific task [25].

The origins of using behaviors to model various agent actions began with the hierarchical paradigm that uses distinct states. These states are enumerated as the sequence of SENSE, PLAN, ACT [25]. In this case the agent would acquire information about its environment using data from sensors and create a model of its environment. Once the agents knew of its environment, it would devise an action plan to achieve this task. The task may be decomposed into many smaller sequential to compose the entire solution. Finally, the agent would act upon the described tasks.

However as discussed in [25] and [26], this original approach suffers from assuming that the agents has obtained a sufficient amount of information about the world. As discussed in [27], during the PLAN state, the agent is unable to maintain continuous knowledge of the environment as it changes. Finally as discussed in [28], the agent has difficulty representing all of the information about is world in a computationally

viable method. Attempts to resolve these issues in the complete SENSE-PLAN-ACT approach were marginally successful [27].

To address these limitations within the PLAN phase, [26] removes this requirement and constructs however as discussed in [27], the systems is a reactive approach typically employed by animal behavior. Instead, this type approach uses a SENSE-ACT model were an example of this would be the behavior exhibited in cockroaches [29]. Similarly, as discussed in [30] and [31], the pioneering work of Craig Reynolds illustrates the complex behavior of flocking birds using three simple reactive rules. Reynolds implemented the rules, in software to describe a behavior that mimics a flock of birds in in a virtual world. Each bird, known as a 'boid' uses attraction, cohesion, and repulsion to create the overall flocking behavior. This work later was extended from simulated bird flocks to autonomous wheeled vehicles by Mataric in [32]. Although the SENSE-ACT plan provided reactive behaviors, discussed in [27], the pure reactive approach is limited in applications that require a form of assessment. In this case the PLAN phase is reintroduced however, it now performs as a supervisor function to the SENSE and ACT phases.

By using agents from a multi-agent approach in ITS [33], an example of that vehicles act as agents within the road network would perform calculations for optimal routes

by sharing the information between one another. These agents within the road network create a VANET. However, these VANETs are clusters of agent's that are constantly moving and therefore must be able to react to a situation quickly in a matter of seconds or less [34]. Furthermore, the composition of the clustering of the vehicles is always in flux [33]. Because of this requirement the manner in which the data are shared becomes timely and approaches must be in place to ensure that the vehicle agents are able to accommodate these data as it is shared between vehicles.

Finally, in [35] a review of the application use of agent technology in transportation systems is discussed. This review provides a discussion that captures the agent computing paradigm in the domain of traffic and transportation systems. It provides an analysis regarding multi agent systems looking at dynamic routing and congestion management as well addressing some critical issues in the development of these types of systems. This paper presents several approaches that employ the use of an agent based approach for distributed ITS architectures.

The approach of using modeling vehicles as agents that exhibit behaviors is a current trend in the distributed architecture approach. However, these approaches do not differentiate vehicles and the heterogeneity that would compose a cluster of vehicles within a multi-agent system. Furthermore, although the interaction of the

agents is noted as fundamental, there is not a defined approach for understanding the interaction between the agents. Finally, as identified in [30a] the timeliness of the data and processing of the data is critical, though there isn't a prescribed method to accommodate the real time issues of the data.

2.4 FUNCTIONAL REACTIVE PROGRAMMING

When considering the vehicle agents in a Multi-Agent System composing a VANET, the agents must react to the changing conditions. Using behaviors exhibit a good approach; however the behaviors are reactive in nature. It is unknown to the agents when these changes will occur. Furthermore, as the data are shared among the vehicles, the time of receipt of the data to the vehicles is not known, that is the data being received is asynchronous in nature to the vehicle itself [36].

When an entity is required to update its action, various competing cost functions will dictate the new action plan based upon the vehicles intended use and action. For example, route re-plans can be cost functioned based on vehicle type, and its cost functions for energy efficiency, density, entity similarity. As discussed in [24], routing density and load leveling may be best performed by when specific types of vehicles are intended to specific route types. In the distributed architecture, agents interact and share information that they acquire about the road network. The

vehicles will need to perform this task using software and hardware. However, it is not known a-priori how many vehicles will be communicating with other vehicles.

The challenge is to be able to propagate the updated information amongst the vehicles in a timely manner and process the data within the vehicle. Traditionally, an event loop programming paradigm that observes the data being received by checking for the receipt of new available data. This is known as the observer pattern. The observer pattern is used in event handling and where an object maintains a list of dependents known as observers, and notifies them automatically of any state changes. The notification is performed by invoking one of the methods of the dependents on the list. Because of the event driven nature of the Observer Pattern for event handling, its implementation generally results in code that becomes unmanageable and inefficient. Furthermore as the system increases in complexity, the growth in managing the observer pattern grows geometrically [37], [38].

Functional Reactive Programming (FRP) is a programming paradigm that was introduced in the seminal paper by Conal Elliot and Paul Hudak [39] to address the observer pattern problems specifically in Graphical User Interface (GUI). The content was further extended by Nilsson in [40]. Originally developed using Haskell as a domain specific language, it has now been adapted either directly or indirectly via extensions to languages such as Java, C#, Python, Groovy, C++, and C#.

Since its original inception, FRP has been applied to many applications that require reactive solutions. One of the first applications of FRP beyond GUI was implemented to provide reactive behaviors for an autonomous robot vehicle [41], [42]. In this work, behaviors were implemented reactively where when data changed from a sensor, then a specific action by the vehicle was performed. Another recent example of applying FRP to complex systems was using the Microsoft reactive extensions for Java (RxJava) to support the development of video streaming for Netflix. In this application [43], the FRP extensions were able to manage the concurrency of the data streams. The FRP extension programming paradigm to host languages has provided a replacement an *iterable* type which must be observed with the observer pattern. This is further extended to support LISTS of *observable* data types which provide a reactive behavior to any one of the data sets associated within the list of observables. An observable data type provides the ability for a producer to signal the consumer that there is no more data available, while also providing information regarding error occurrence. Because of the combination with a functional model, the service layer is safe to provide concurrency as well as the following:

- Conditionally return immediately from a cache
- Block instead of using threads if resources are constrained
- Use multiple threads
- Use non-blocking I/O
- Migrate an underlying implementation from network based to in-memory cache

- Asynchronous Data streams
- Safe concurrent processing

[44], [45].

Due to the nature that FPR is able to handle the varying nature of asynchronous event driven in a high-level way to work with interactions in real time[46] it becomes a new approach of how to address the implementation programming the agents within a distributed ITS architecture. These are critical elements for agents to effectively communicate data with one another in a continuous basis.

Chapter 3 CENTRALIZED AND DECENTRALIZED CLASSIFICATION

3.1 OVERVIEW

Using techniques of Intelligent Transportation Systems requires capturing the data from about the road network. In a centralized architecture, these sensors may include road network loop sensors, roadway camera systems. For a decentralized architecture the vehicles themselves become the sensors. In ITS, when the vehicles communicated to acquire the data regarding the data, the data are said to be provided by the infrastructure. As a result this is referred to Vehicle-to-Infrastructure or simply V2I. This is in contrast to the distributed architecture where the vehicles communicate with each other in a vehicle-to-vehicle approach, or simply V2V.

While declaring an approach as V2V or V2I, suggests either centralized or decentralized approaches, the two approaches may be present. Unfortunately simply indicating a V2V or a V2I approach lacks specificity of how the overlap may occur. Presented here is a new way to provide clarification this clarification to the architecture topologies for data acquisition and processing of the data.

3.2 CENTRALIZED AND DISTRIBUTED ITS TOPOLOGY CLASSIFICATIONS

To provide the distinction of architecture topologies, the concept of a *Decision and Information Acquisition Matrix* is proposed. This matrix encompasses the attributes of data and processing from both the *Infrastructure and Vehicle perspective*. The matrix is composed using a dimension representing information acquisition, while the other dimension is composed of the location of processing. In the matrix, each cell is represented by a C, D or CD where the C represents Centralized method or Infrastructure-based, and the D represents a Distributed solution, and CD is a combination of both methods. The type of ITS structure is defined by the cell location in the matrix as:

(Information Acquisition, Processing Location)

Decomposing the matrix into each possible case provides the following matrix where each cell is considered to be a type of ITS architecture representation to define where the data are captured from, and where the data are processed.

Table 3-1 ITS Classification Matrix differentiating location of data acquisition and processing

Centralized

Information Acquisition

Distributed

Processing Location











↓

1 / (C,C)	2 / (CD,C)	3 / (D,C)
4/ (C,CD)	5/ (CD,CD)	6/ (D,CD)
7/ (C,D)	8/ (CD,D)	9/ (D,D)

Distributed

In Table 1, each cell in the ITS Matrix has unique properties and based on the requirements of the application, one or more cells can be implemented to provide a complete ITS solution. This provides a technique to classify how data are captured and where the processing is performed. The cells of the matrix are defined as follows. The figures describing each cell use the following nomenclature:

Table 3-2 Legend for ITS Classification Matrix Illustrations

Symbol	Function
	Sensor - The sensor may be a loop sensor, camera system, radar sensor. Provides data to the TMC
	Data Aggregator - Gathers the data from the sensors for a given link of road
	Traffic Management Center
	Ad-hoc network between vehicles
	Indicates data collected by the sensors to be processed at the TMC
	Indicates data sent from the data aggregator collected from the sensors
	Indicates data sent to from the TMC to the vehicles as actions to perform
	Unprocessed Data provided to the TMC to be redistributed to vehicles
	Decision processed actions performed by the vehicle
	Data provided to the vehicle to aid the vehicle in processing

3.2.1 CELL 1 TOPOLOGY (C,C) – CENTRALIZED DATA ACQUISITION AND CENTRALIZED PROCESSING

This is an ITS topology illustrated in Figure 3-1 which is a fully centralized. The sensors acquire the information and collect the data. The processing involves necessary actions to take by the vehicle such as proving the best appropriate routes for the vehicle to perform based upon the data collected via the sensors from the

road network. Data flow is considered to be unidirectional between nodes from the each element. Figure 3-1 illustrates an example of this approach and is considered to be a baseline for subsequent variations. Since the sensors do not transmit directly to the vehicles, but instead via the TMC, they are considered part of the centralized solution [13].

Advantages to this topology is that over the long term, strict enforcement of communication standards can be controlled since all of the data must pass through the infrastructure. However, this approach requires a high cost primarily due to the processing centers and the installation of the network. Data acquisition can be coarse depending on the spacing of the sensors. Additionally, in the case of freeways, the loop detectors are processed by one TMC while for arterial roads, they are managed by separate agencies that are not always affiliated with the TMC for the freeways. Another disadvantage of this approach is the time to implement. The TMC must be constructed, as well as the installation of the network, and sensor systems if not currently installed. Once the system is installed, making changes to the system becomes difficult due to the amount of change necessary to the TMC, and data networks.

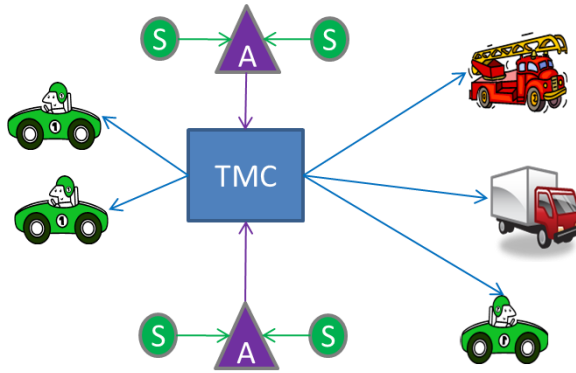


Figure 3-1. Cell 1 - (C,C) Fully centralized ITS topology for both data acquisition and processing

3.2.2 CELL 2 TOPOLOGY (CD,C) CENTRALIZED / DECENTRALIZED DATA ACQUISITION AND CENTRALIZED PROCESSING

The ITS framework topology illustrated in Figure 3-2 is one that is fully centralized for data processing at the TMC. In this approach data are acquired from both a centralized-based infrastructure sensors and also uses distributed methods of data acquisition. Data are acquired by both traditional methods such as loop detectors and sent to the TMC via the aggregators and also acquired via vehicles known as probe vehicles. Probe vehicles are those vehicles that provide their position, speed and other relevant information to aid in the assessment of the link [14]. The probe vehicles send their information of the infrastructure using wireless technology [15]. Data flow is unidirectional from the sensors and non-probe vehicles. The data are bidirectional between the probe vehicles that provide data to the infrastructure and the TMC. All command decisions are provided from a centralized location at the TMC.

Advantages to this approach are that data are acquired via probe vehicles where stationary sensors may not be located providing a higher degree of fidelity over that in Cell -1. Also, since the data must pass through the TMC, a strict adherence to standards can be enforced. However the disadvantages to this approach reside in the cost of the TMC as well as the network that captures the data via the stationary sensors. Also, implementation of this type of system due to the infrastructure construction requires time to implement. Another disadvantage to this approach is that once the system is designed it can become difficult to modify due to the large scale and impact the changes would present. However, when compared to Cell-1, because data are captured in the vehicles as well, the system is more robust to implement change by updating the vehicles without requiring updates to the physical infrastructure.

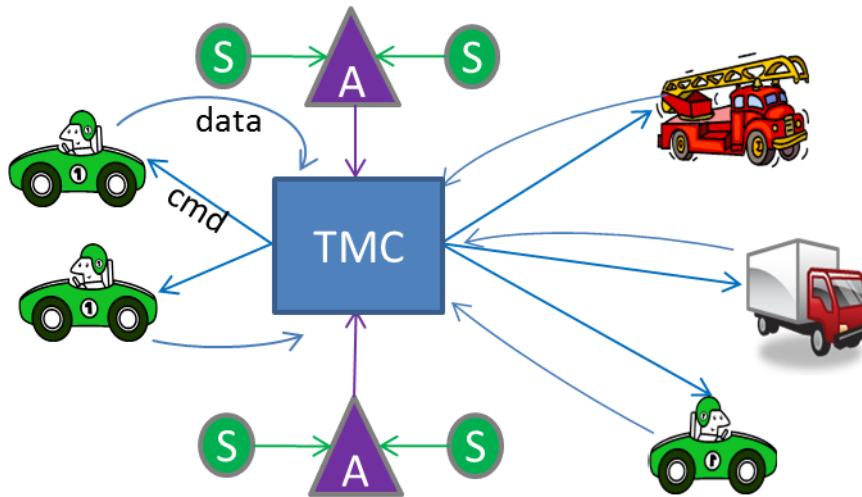


Figure 3-2. Cell 2 - (CD,C) Centralized and Decentralized Data Acquisition, fully centralized decision processing ITS Topology.

3.2.3 CELL 3 TOPOLOGY (D,C) DECENTRALIZED DATA ACQUISITION AND CENTRALIZED PROCESSING

This topology that is illustrated in Figure 3-3 is an ITS framework topology where there are no fixed infrastructure based sensors. Instead, data are only a product provided by the vehicles. This would typically involve capturing network information by use of probe vehicle methods and provided to the TMC. The data from the probe vehicles are then sent to the TMC.

The advantages to this approach are that there is that cost can be reduced since there are no longer sensor systems or a network infrastructure required to provide data to the TMC. Instead the cost is transferred to the drivers in the probe vehicles. Because the data are provided to the TMC, an adherence to standards is able to be enforced, however because the data are provided via the vehicles.

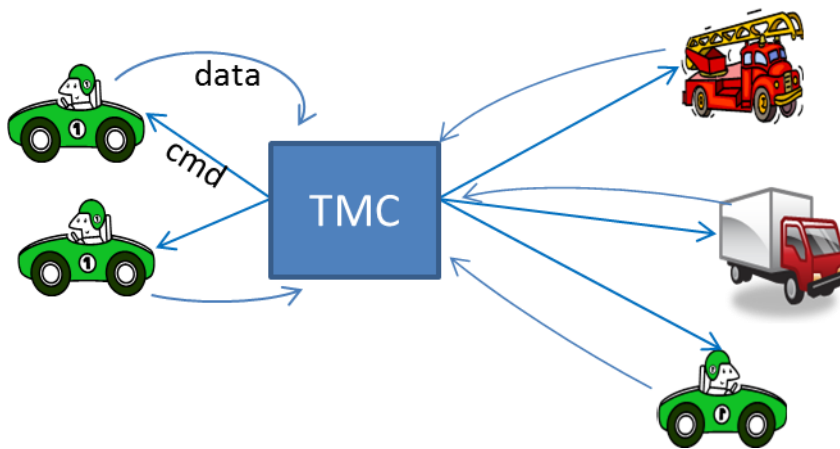


Figure 3-3. Cell 3 - (D,C) Decentralized data acquisition, fully centralized decision processing ITS Topology.

3.2.4 CELL 4 TOPOLOGY (C,CD) CENTRALIZED DATA ACQUISITION AND CENTRALIZED / DECENTRALIZED PROCESSING

Shown in Figure 3-4 is an ITS framework topology where data acquisition is performed through the infrastructure, however application data processing is performed by the processing center at a high level, and at a low level by the vehicles.

The command from the TMC may be thought of as a suggestion depending on the action to be taken by the driver. In this approach, data visibility is acquired from the sensors within the infrastructure, but depending on the fidelity of the sensors, the driver may have a better knowledge about the events specific to his current location. In this case the command issued by the TMC may include one or more possible solutions provided to the driver where the driver makes the final decision.

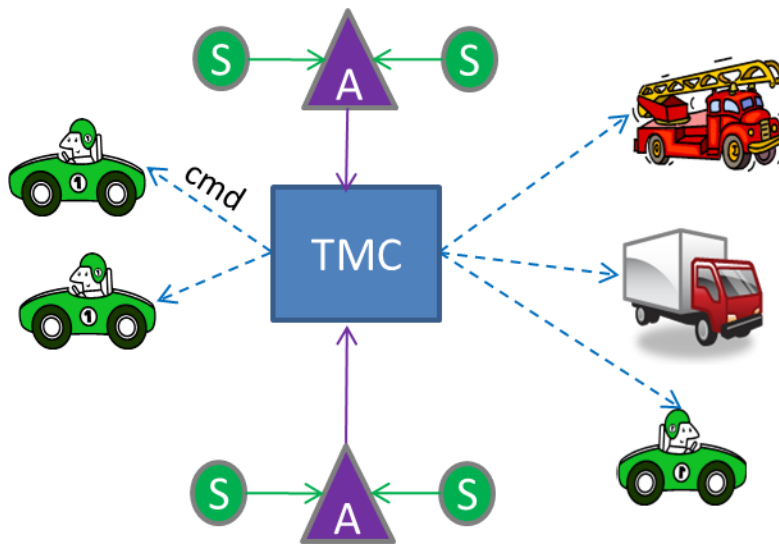


Figure 3-4. Cell 4 - (C,CD) Centralized data acquisition, and centralized and decentralized decision processing ITS Topology

3.2.5 CELL 5 TOPOLOGY (CD,CD) CENTRALIZED / DECENTRALIZED DATA ACQUISITION AND CENTRALIZED / DECENTRALIZED PROCESSING

This ITS framework topology illustrated in Figure 9 is where data are acquired by both the infrastructure and the vehicles on the network. Data are provided to the TMC for dissemination, as well as to other vehicles that are equipped to accept data and process the data. Data flows in a bidirectional manner between the vehicles that can capture and process their own data as also with the TMC.

The advantages to this approach are that not all of the vehicles need to be vehicles that provide sensor data, however probe vehicles are able to provide data to the TMC to increase data fidelity. Likewise, the processing can occur in the vehicle as equipped reducing computational and network demand on the TMC while those that use the TMC are able to do so. Because of the TMC this approach provides stability over the life of the system, however due to the inter-vehicle capabilities, the adherence to the TMC standard for vehicles to communicate with one another is not necessary. However, this does provide good robustness of the system as well. Vehicles that are able to interact directly with one another are able to easily adopt new features without requiring the TMC to have the capabilities implemented. This approach also provides high data fidelity due to capturing data both at stationary

network sensor systems, as well as the vehicles themselves. The disadvantages of this approach is it is the highest cost. It incurs all of the cost of the centralized, while also requiring cost for each of the vehicles.

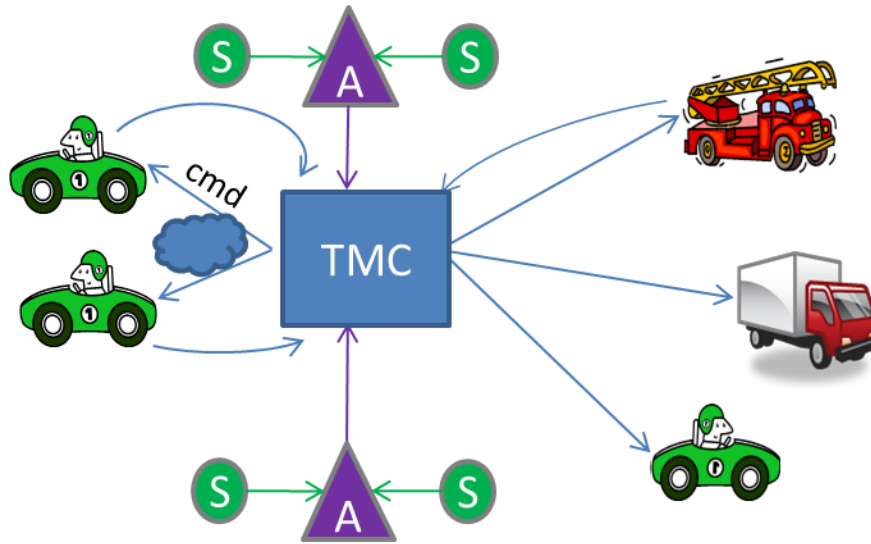


Figure 3-5. Cell 5 - (CD,CD) Centralized and decentralized data acquisition, and processing ITS Topology

3.2.6 CELL 6 TOPOLOGY (D,CD) DECENTRALIZED DATA ACQUISITION AND CENTRALIZED / DECENTRALIZED PROCESSING

Figure 3-6 illustrates the topology for decentralized data acquisition and both centralized and decentralized processing. This is an ITS framework that acquires

data from the vehicles while the vehicles receive possible actions from TMC. Data are still aggregated at a TMC from all of the vehicles connected to it. The TMC provides drivers insight about the overall state of the road network while the drivers will make local decisions. An example of this could be a private company communicating with a fleet of vehicles such as a taxis company [47]. A taxis is able share amongst its drivers various conditions of demand needed by users and the current route conditions. The data are collected from each taxis and sent to the a TMC. The taxis TMC is then able to dispatch taxis as needed to accomodate demand while achieving the fastest availble travel routes. In this approach data are bidirectional between the TMC and the vehicles.

The advantages to this appraoch is it can be quick to implement, and can be robust as well since the connectivity is a closed system. Additinally, having a TMC managed and controlled provides long term stability. Data fidelity is dependant upon where the vehicles are located. Depending upon the vehicle locations, there may be some areas of the road network may not have current data and the data that is known may be aged.

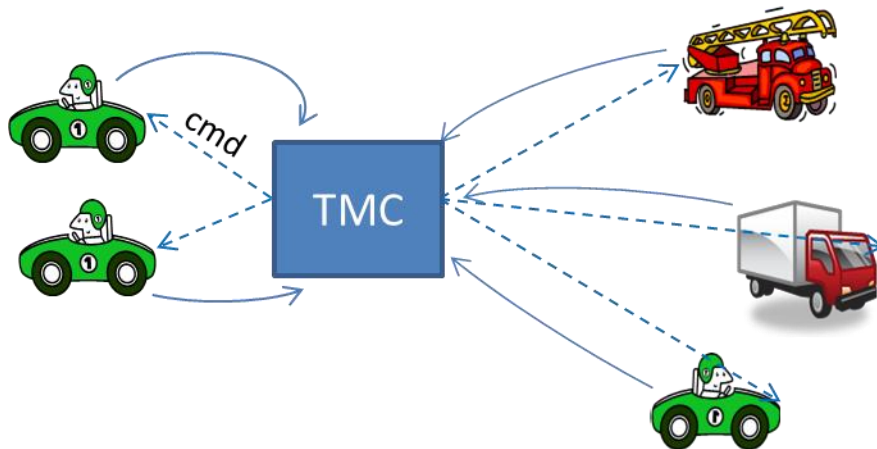


Figure 3-6. Cell 6 - (D,CD) Centralized data acquisition, and centralized and decentralized decision processing ITS Topology

3.2.7 CELL 7 TOPOLOGY (C,D) CENTRALIZED DATA ACQUISITION AND DECENTRALIZED PROCESSING

The topology described in Figure 3-7 is an ITS framework topology that uses infrastructure sensors and simply provides data to the vehicles where the processing is performed. Essentially this mimics the current ITS architectures where the TMC doesn't provide detailed data, but rather only aggregated link data from the roadway network to the vehicle. The vehicle's decision is made without knowledge of the decisions made by the neighboring vehicles. This approach is basically what exists today when drivers check current traffic conditions before leaving; however they do not use information about what the other drivers to influence their decision.

The advantages to this approach is a TMC that does not require significant processing, and does provide a long term stability by ensuring data provided to the vehicles is captured in a known format from the road network sensors. Because the decisions are made by the vehicle using the available data, the decisions processing can be upgraded without impact to the infrastructure. The disadvantages to this approach are that the TMC does perform the decision processing. Instead the vehicles themselves make decisions based upon the data received from the infrastructure sensors. This is indicated by the dashed lines looping back from the vehicle to itself. In this case the data would only have the fidelity of the road network sensors. The time to implement this system resides in the development of the sensor networks with the TMC. Cost of the TMC and sensor systems exist, however the cost of the processing is allocated to the vehicles.

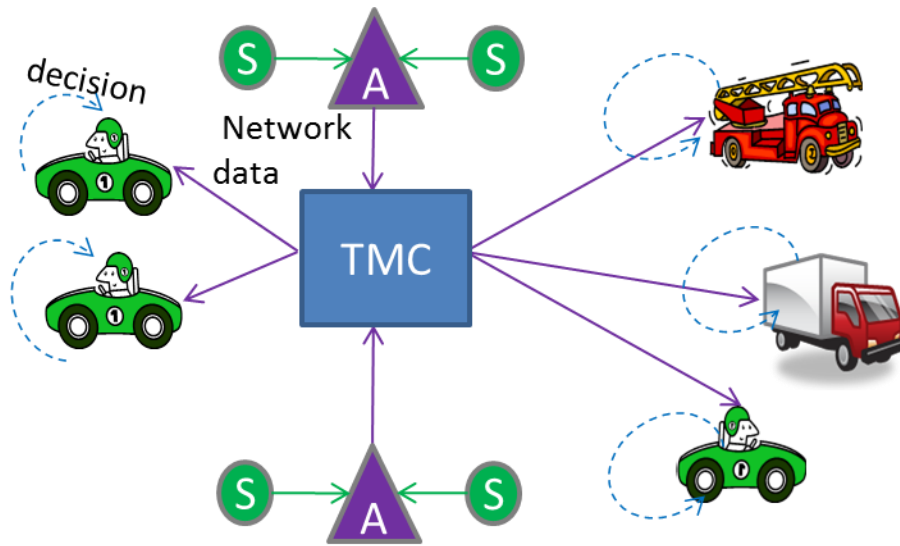


Figure 3-7. Cell 7 - (C,D) Centralized data acquisition and decentralized decision processing ITS topology.

3.2.8 CELL 8 TOPOLOGY (CD,D) CENTRALIZED / DECENTRALIZED DATA ACQUISITION AND DECENTRALIZED PROCESSING

The topology shown in Figure 3-8 is an ITS framework that uses the infrastructure to capture data from sensor systems as well as the vehicles. The data are gathered by the TMC and sent back to the vehicles where the vehicles perform the processing based upon the received data. This is indicated by the red arrows to differentiate that the data from the TMC is not processed only provided to the vehicles. This is on contrast to the previous TMC based approaches. The decisions made by each vehicle are made without knowing the intent of the other vehicles. Instead the

vehicles perform the necessary processing. The data between the vehicles and the TMC is bi-directional.

The advantages to this approach is in increased fidelity by providing data to the vehicles from both the infrastructure as well as by the vehicles connected to the TMC. Because of the connectivity with the TMC there is an adherence to required standards. The disadvantages to this approach is lack of ability to quickly adapt to changes in technologies. The cost for this approach is impacted by the TMC as well as the support for the sensor systems and associated network requirements.

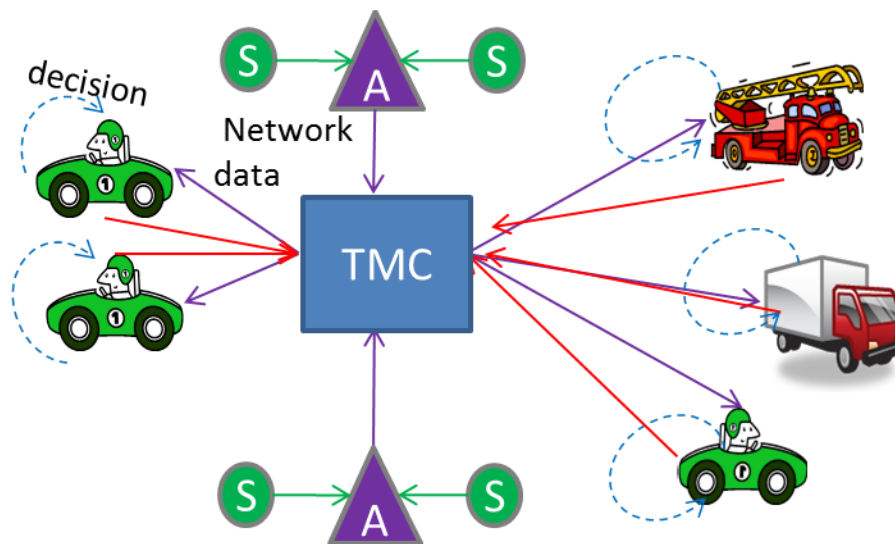


Figure 3-8. Cell 8 - (CD,D) Centralized and decentralized data acquisition, decentralized decision ITS Topology.

3.2.9 CELL 9 TOPOLOGY (D,D) DECENTRALIZED DATA ACQUISITION AND DECENTRALIZED PROCESSING

The ITS Framework topology illustrated in Figure 3-9 is fully distributed in the processing of data and data acquisition. Data are acquired by vehicles sharing information within some local range or to a cloud of vehicles in the network and processed by each vehicle locally. This approach requires strict adherence to standards such as those set forth in [6]. This is considered to be a pure multi-agent system implementation for an ITS framework [10], [11].

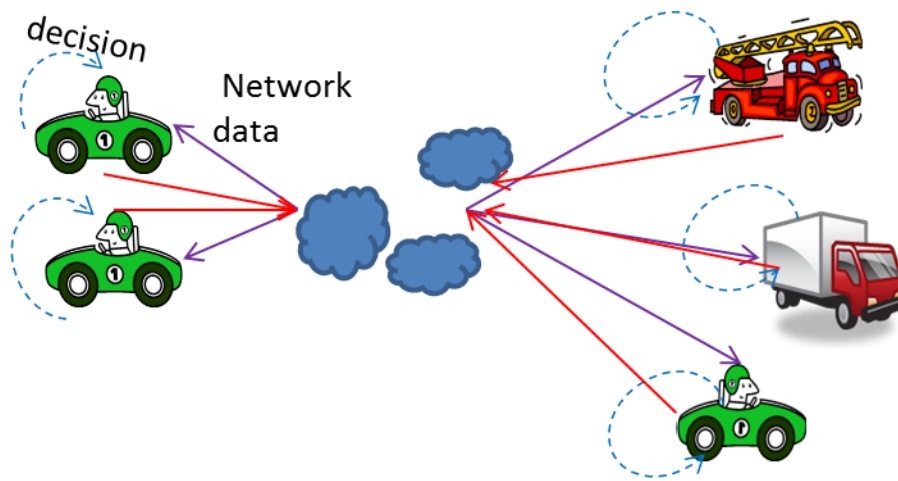


Figure 3-9. Cell 9 - (D,D) Decentralized data acquisition, with decentralized decision processing ITS topology.

Provided below in Table 3 is a summary of the advantages and disadvantages for each cell.

Table 3-3 Summary of Matrix of ITS Comparing Topologies. A '+' indicates higher degree of applicability to criteria, while a '-' indicates a lesser degree of applicability to criteria.

Cell	Data Capture	Processing	Cost	Robustness	Time to fully implement	Long-term Stability	Data Fidelity
1	Centralized	Centralized	\$\$\$	--	--	++	-
2	Centralized / Decentralize	Centralized	\$\$\$\$	--	--	++	++
3	Decentralized	Centralized	\$	+	-	+	+
4	Centralized	Centralized / Decentralized	\$\$\$\$	--	--	+	-
5	Centralized / Decentralized	Centralized / Decentralize	\$\$\$\$\$	++	--	+	++
6	Decentralized	Centralized / Decentralized	\$\$\$	++	-	+	+
7	Centralized	Decentralized	\$\$\$	0	--	+	--
8	Centralized / Decentralized	Decentralized	\$	--	--	+	++
9	Decentralized	Decentralized	\$	++	++	--	+

Presented in Table 3, is a summary of each cell and its relative costs to each implementation. The cost based upon the inclusion of a TMC, sensor systems, and cost to vehicles. The robustness represents the degree of difficulty to update and modify the system design. A '+' represents a higher degree of adaptability, where as a '-' is a lesser degree of adaptability. Long-Term is a parameter that represents the likelihood that the technology will be compatible over time. When connectivity to a TMC occurs, standards are set forth by a governing body [12]. This is due in part to the large investments made to develop the TMCs. The advantages to this is that there is a lower likelihood that a standards implemented will change quickly. For example, where there is no connectivity to the TMC that is only between the

vehicles, companies are able to quickly change, upgrade and modify protocols. A simple example of this is the networking layers of a wired Ethernet, versus the quicker evolving of 802.11x. Finally data fidelity is a parameter to capture data to provide to the overall system. This parameter relates the data captured via the road network sensors, and the vehicles which themselves are capturing data such as probe vehicles.

The matrix is a method to quantify the relationship between the various levels of centralized and decentralized data capture and vehicle processing. It provides an ability to assess existing systems and the application of ITS to new locations. Using the matrix, ITS designers are able to mature and evolve with the development of technology. It provides a method to describe what cell is being used and a method to capture interface standards, data dictionaries, processing and network throughput requirements from a system perspective.

Chapter 4 MULTI-AGENT APPROACH IN A DISTRIBUTED ARCHITECTURE

Using a Multi-Agent System approach to a distributed architect for example as discussed in [44] is viable; this approach requires a unique combination of actions that are a function to that specific problem being solved. In this sense, specific solutions for specific problems are well understood, however it becomes difficult provide a general solution for agents where the interactions between agents would prevent them from their task. Instead, presented here is an approach that describes a technique to prevent possible interactions that result is a preventing the agents from performing their tasks. Moreover, it defines a method to aid in ensuring that the agents are continue to perform their actions as desired under conditions not provisioned.

Within a multi-agent system, an agent is described by a finite state machine (FSM) for a particular behavior. The hierarchical state is decomposed into the following types of states. Within each state possibility, there exist two fundamental prototypical sub-states, whose composition of the FSM formulates the agent's behavior using two specific behavior primitive types. The two fundamental FSM prototype sub-states within each agent include a *Type B* defined as the behavior type, and a *Type A* defined as the Arbiter Type:

Type-B - Behavior Type. The *Type B* is defined to represents a set of sub states of an agent that exhibits a particular behavior. It is considered a general purpose action or set of actions that an agent performs described using a prescribed set of states and actions. An example of a general type B Behavior would be “Follow the Leader”, or “Follow Fastest Route”.

Conversely,

Type-A Arbiter Type. The *Type A* is a special type of behavior specifically to perform behavior selection between one or more possible behavior. This arbitration would be perform by means of a cost functions.

Additionally, in each state, there may be one or more Type B behaviors operating concurrently but arbitrated via the Arbiter prototype. A state diagram will have the structure as shown in Figure 4-1 where there is up to n behaviors $B_0 \dots B_n$ arbitrated via A

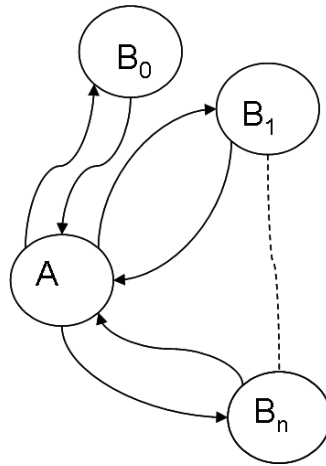


Figure 4-1 Generic FSM Structure for state behavior decompositions

As an example of an agent's behaviors with an arbiter is presented as shown in Figure 4-2.

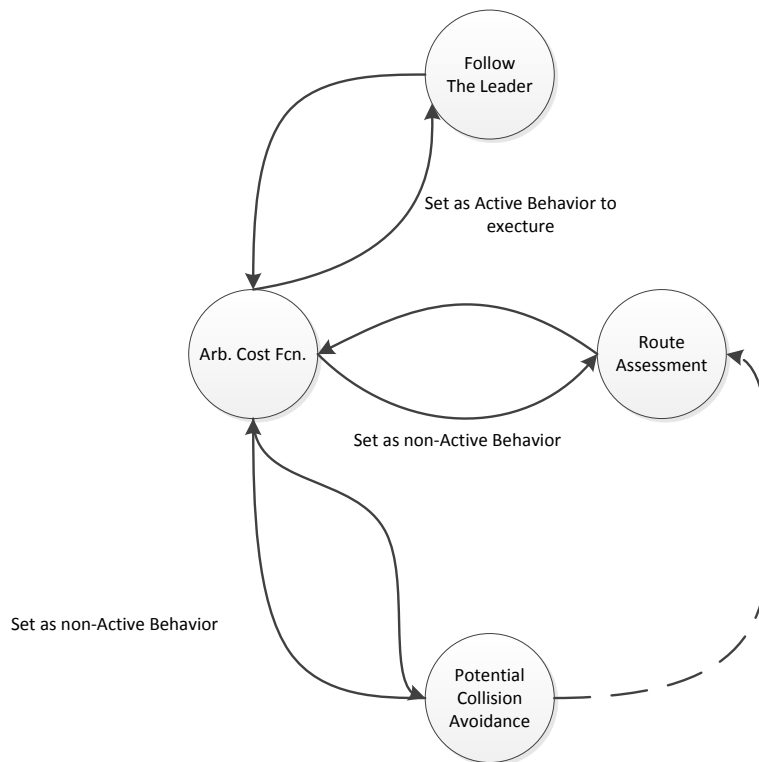


Figure 4-2 Example of Agents behaviors defined using Type A and Type B behavior prototypes

In this example, the agent is continuously looking for faster routes, while concurrently following in a 'Follow the leader' grouping of vehicles. It is also ensuring that there are not any vehicles within the surrounding vicinity have not inadvertently stopped which would cause an accident. It is noted that information from one behavior state can affect the behavior of another. This is shown by the dashed lines connecting the behaviors. The active behavior is set at the 'Follow the Leader' behavior. The arbiter is evaluating the current performance of the vehicles travel time as it travels on its current route. It is arbitrating using a cost function of travel time to travel along its route. However, when the vehicle speed is no longer sufficient the vehicle would then invoke the behavior of removing itself from the current grouping of vehicles to find an alternate route.

4.1.1 TYPE A – ARBITER PROTOTYPE

The Type-A sub-state prototype performs a decision making process based on constraints imposed by the local environment and state. For example, a road network segment can be characterized by some attributes such as current travel time. This is a constraint that will determine how to arbitrate between implementing various Type-B primitives. Another constraint that articulates which

type of behavior B to be transitioned to/from is the composition of the types of vehicles within a certain area defined as a *cluster*. That is, the composition of the cluster will influence the optimal outcome of the decisions. Additionally, internal state information about the vehicle itself can dictate the Type-A prototype's arbitration. This information may be constant data such as vehicle size, top speed/torque, or the arbitration may include dynamic data such as current velocity versus a given location. Each of these data elements is used to determine what action to take as it arbitrates its various behavior primitives. The design of the Arbiter Prototype is based upon a similar set of lower level characteristics known as behavior elements. The behavior elements have specific compositions that adhere to a structural format.

TYPE B – ARBITER PROTOTYPE

TYPE I - FORMATION FLOW

This behavior primitive includes behaviors that implement location of the individual vehicle relative to a leader type within the cluster and where one vehicle is relative to another in a 1-D, 2D, or 3-D sense. These behavior types include:

- ***Follow the Leader:*** *The second and subsequent vehicles maintain a given distance from the vehicle in front of it adjusting distance for the given velocity. The leader implements a different behavior*

- **Lead the Follower:** *The vehicle $n-1$ of n vehicles stays in front of vehicle n at a given distance and heading, for all vehicles 1 to $n-1$.*
- **Lateral Follow [Left, Right, Center, etc...]:** *Vehicles stay lateral to one another without any vehicle moving faster or slower than its neighbor. The Leader can be any given vehicle such as the far most left, far most right, or any vehicle in between*

TYPE II – ATTRACTION / DETRACTION

Essentially this also provides the behavior primitive to perform *collision avoidance* via external sensing through onboard sensors and/or via vehicle-to-vehicle communication. There are also higher levels of affinities that support this type. For example a small car should avoid being trapped between two large semi-tractor trailers in the event of a potential collision. This behavior type captures avoiding high density, low speed links. It can also include a cohesion element that allows vehicles of similar types to route using similar routes.

TYPE III – ENVIRONMENT FLOW

This behavior primitive type provides basic maneuvering functions and traveling techniques. They assume no knowledge of local neighbors other than for acquiring flow information. These behavior types include:

- **Directed Way-point:** Vehicles are directed to specific locations
- **Lane Following:** Vehicles follow the lane maintaining an equidistant spacing that defines the lane space
- **Line Following:** A vehicle follows a line at a commanded velocity
- **Edge Following:** A vehicle follows an edge such as a curb or wall, etc at a commanded velocity
- **Open-Space Bisection:** A vehicle follows bisects an open space it is traveling in
- **A*:** The vehicle implements the A* routing algorithm to travel from its origin to its destination
- **Dijkstra:** The vehicle implements the Dijkstra routing algorithm to travel from its origin to its destination

The motivation in describing the behavior algorithms to follow specific structure provides a means of validation. Each behavior primitive is able to be developed then implemented independently, and then assembled together for final simulation and validation. This architectural framework provides the algorithm designer with a set of tools to have the expected outcomes.

By recognizing that vehicles differ in intent and type, we can characterize the MAS scenario as a heterogeneous multi-agent system, which allows for additional information to make decisions that are better suited to the type of vehicle being driven. Furthermore, on any given roadway, the vehicle type composition can vary greatly. For example, vehicle lengths can vary from the passenger car to a two-axle single unit truck. Taking this into account, in the event of freeway incidents, we segment the traffic flow onto the nearby city streets such that the congestion is reduced versus each vehicle taking the individual optimal route. Using VANET technology, each vehicle has the ability to convey information to its neighbors to influence the arbitration between behavior primitives.

An example road network consisting of a freeway and city roads is shown in Figure 4-4.

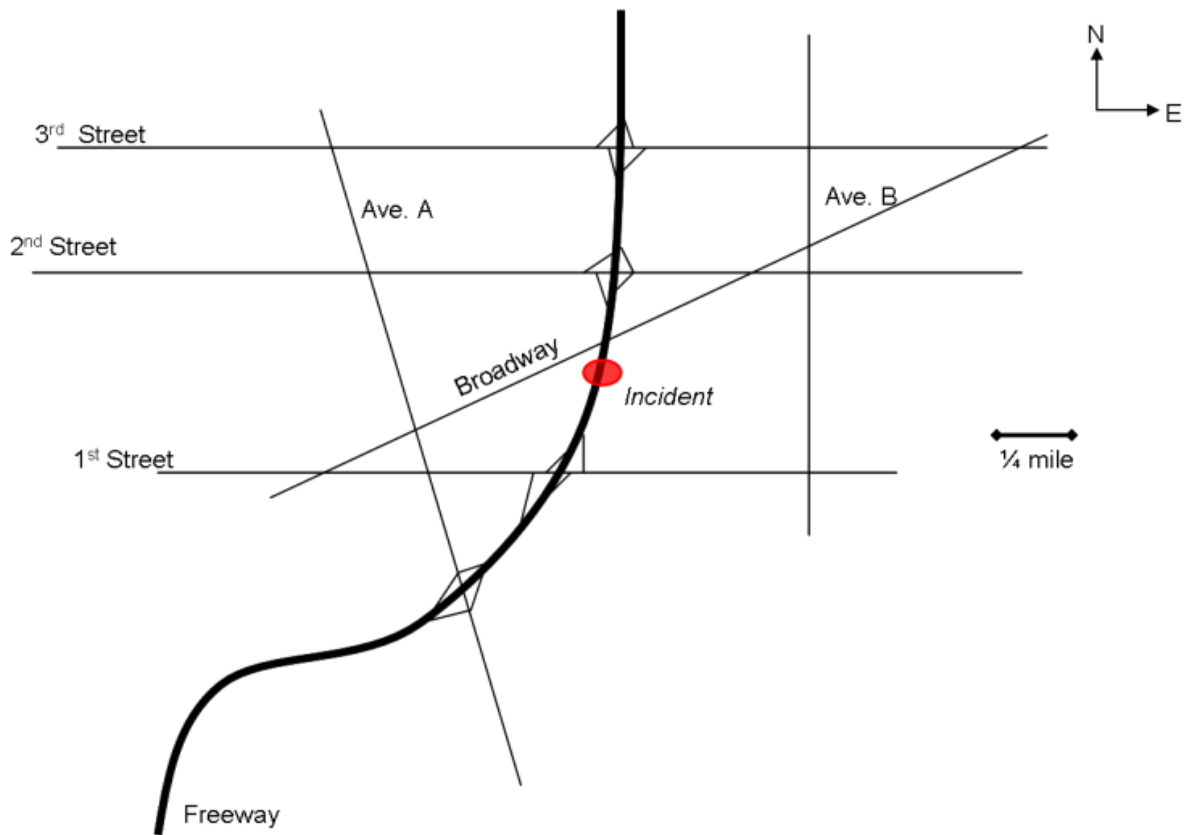


Figure 4-3 Sample road network where vehicles reroute around the incident shown in red.

In this example network, the number streets, 1st, 2nd, and 3rd are East-West bound, and the letter avenues A, and B are North and South bound streets. The number streets and avenue A have on-ramp and off-ramp freeway access. Cutting diagonal across the area is Broadway; however, it does not have ramp access to the freeway. Indicated on the network is a traffic incident between the 1st and 2nd street ramps which closes the freeway in the northbound direction. Although this is a fictitious

road network, it captures typical road grids that drivers would otherwise decide which would be the best alternate route to take. It assumes that each of the city roads is two lanes per side with left turn lanes. Each intersection is signaled with a protected left turn phase. Although there may be smaller city roads, they are considered residential roadways and not viable alternate routes. As typical in many regions, right turns are allowed on red in these cases.

By using a simple A* minimum path algorithm based on distance while traveling northbound, the shortest route would instruct the driver to exit on 1st street West bound, by turning left. The driver would continue to avenue A and turn right, and then quickly turn right onto Broadway heading north-east. Upon reaching 2nd street, the driver would turn left traveling west and then re-enter the freeway.

Looking at this approach, we quickly see the left turn lane on the off-ramp to exit the freeway will become severely congested as cars attempt to take the freeway off ramp, causing the ramp queue to spill over on to the freeway. Depending on the flow on Avenue A, once on Avenue A, the right turn will not queue as fast as the original left turn. Upon reaching 2nd street, the left turn lane will quickly queue up eventually congesting Broadway.

Alternately, a potential routing algorithm may instead favor the overall north-east freeway travel, and instruct the driver to exit on 1st heading east until the driver reaches Avenue B. The driver would then turn left and head north to 2nd Street and then turn left, following it to the on-ramp. Although initially, the right turn will not queue as fast as the left turn onto Avenue B, the congestion will be transferred to the left turn queue to turn onto Avenue B, and then again from Avenue B onto 2nd street.

By using a VANET approach, and the distributed behaviors, with each vehicle implementing its appropriate behavior, the vehicles can communicate their current speed, location and type of vehicle with the vehicles within a radius defined by the communications technology, such point to point radio, or mobile phone towers [48]. This allows the vehicles around a given vehicle to know how long each vehicle is, and, thereby self-direct themselves to be queued to turn left or right. The vehicles know their current position based on an onboard telematic device and a digital map.

As the vehicles approach the incident, internal state information about the vehicles blocked are broadcast to other vehicles (using the VANET-based communications), instructing them to exit the freeway. Additionally, the internal velocity may dictate that necessary action is required such as determining if an exit ramp is located within distance and time to exit the freeway. At this point the vehicles within the

cluster have identified the types and sizes such that a local list of vehicles is enumerated. In this example, large vehicles are self-directed toward the left onto 1st street while small cars are self-directed to the right. Depending on the number of small and large vehicles, small vehicles may also turn left to reduce the load on the right turn lane queue.

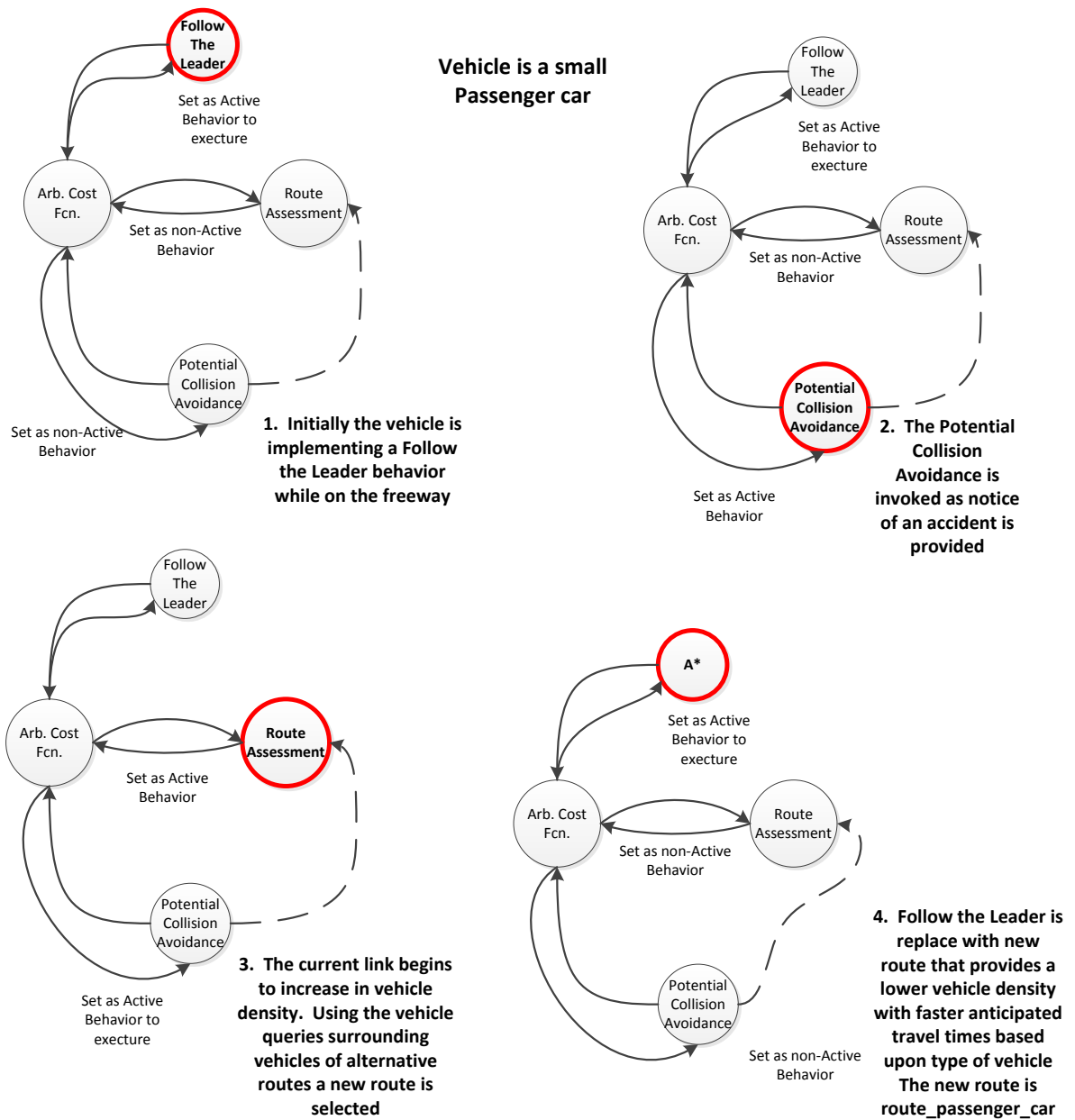


Figure 4-4 Evolution of state sequences for a small passenger car using behavior prototypes

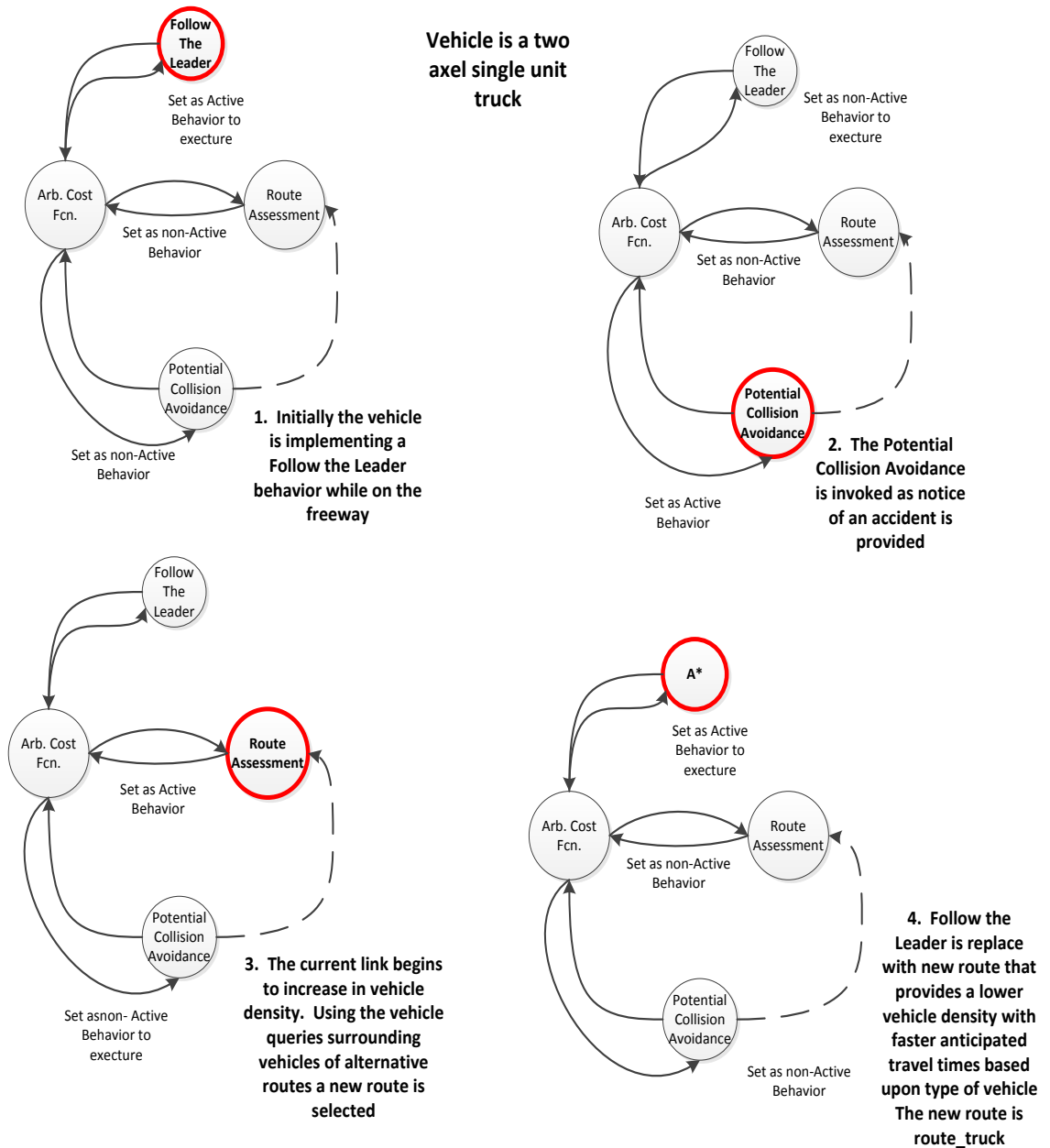


Figure 4-5 Evolution of state sequences for a small passenger car using behavior prototypes

After the initial separation of the cluster is performed, additional separation of the cluster of the larger vehicles that turn left onto 1st is performed. The smaller vehicles versus trucks are directed to turn right on avenue A taking the passenger car route, while the delivery trucks are directed to continue to the crossing of Broadway and 1st street, which would be a truck route alternate. As Broadway may become congested, the original left turned vehicles are further self-directed to reroute to turn left again onto avenue A, and continue north until 2nd street or 3rd street depending on how the 2nd street onramp flows change. This is similar to dynamic traffic assignment where a cost feedback is incorporated for centralized control [9]. However in this approach the control is decentralized throughout the multi-agent network.

The process is again repeated as current conditions of the routes as the vehicles continue their travel. For the smaller cars that were self-directed to the right and turn left onto Avenue B, they can use road attribute data about the intersections and V2V information to assess flow conditions and alternate in group sizes according to the road attribute to either turn left onto 2nd street or continue north until 3rd street. During the time of traveling in the smaller clusters, the individual vehicles will exhibit a follow-the-leader behavior, while the leader is following a directed point & lane following behaviors running concurrently with Type II. Likewise, as the two

clusters are initially following avenue B north. The vehicles in this cluster use V2V communication to travel with spacing that otherwise would not be possible with a follow the leader behavior. Additional freeway onramps can be utilized to spread out the impact of the incident on the freeway until the incident is removed. Finally, as the cluster may report the off-ramp at 1st street is becoming too congested based on internal speed and vehicle composition, larger vehicles are now self-directed to exit at Avenue A and travel north to third street to make a right turn onto 3rd and reenter the freeway.

This detailed example assumes that the freeway incident completely closes all lanes. This is not always the case. In the case of only one lane being open, the same framework can be utilized to self-direct vehicles to either remain on the freeway such as leaving large tractor trailers versus exiting the freeway. Additionally, various elements of the surrounding neighborhood can be also be characterized. For example, it may be unsafe to route vehicles through specific streets that drivers unaware of the area may not know the safety of the neighborhood.

4.3 VEHICLE INTERACTION

In the context of a distributed architecture, fundamentally the underlying scope of action is dictated by the interaction of the individual agents. For example, vehicles in a distributed architecture share data with one another, and potentially select their routes due to the location of the other vehicles. As data are exchanged with each vehicle or as described in multi-agent systems, each agent, this exchange is an interaction. The exchange of data dictates the subsequent action. Because the interaction of the vehicles becomes the basis for subsequent actions by each vehicle, the vehicle interaction provides the fundamental relationship within a distributed multi agent system.

As discussed in [48] is a popular method to describe vehicle interactions by means of vehicle behaviors, while elaborations discussed in [18] extend these concepts. Applying principles of multi-agent systems to develop vehicle decisions becomes one of understanding the interactions that occur with each of the vehicles within their areas of interaction. This presents the challenge of understanding how these vehicles can best interact with one another.

4.4 VEHICLE COOPERABILITY

When two or more automated vehicles exist within the same time and space partition, the programming implemented within each vehicle must ensure that these vehicles are capable of interacting with each other. Furthermore, it is anticipated that the interaction involves up to hundreds of vehicles simultaneously interacting with one another. The question becomes how can the system design of the vehicles interacting with one another can and will cooperate with one another as more and more automated vehicles are confined to a shared space while performing tasks such as intersection management or traffic load leveling.

Knowing that each vehicle is a complex processing platform, we choose to consider the processing of each vehicle as a whole. Additionally, knowing that in computational systems, the concept of computability is a notion that given sufficient resources and sufficient time, an algorithm is said to be computable if the task requested to be performed completes [49]. Looking at intelligent vehicles that exhibit computable algorithms within themselves and that the vehicle is able to complete its computable task is also computable.

A very simple example of this would be for a vehicle that is able to traverse the shortest known path that is on a circular path. Furthermore, consider to be computable when the vehicle completes its traversal. This example can then be further extended to place the vehicle on a path as shown in Figure 4-7a. The vehicle then completes the task of circumscribing half of the path as shown. Additionally, the path is the width of the vehicle. From an individual vehicle perspective, the task can be considered computable.

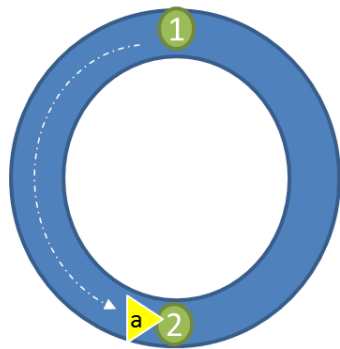
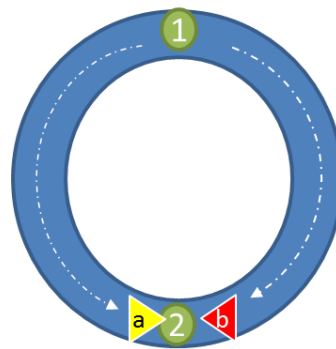


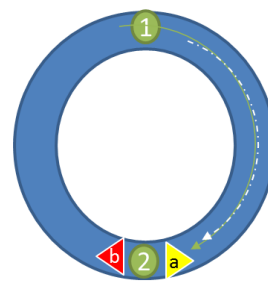
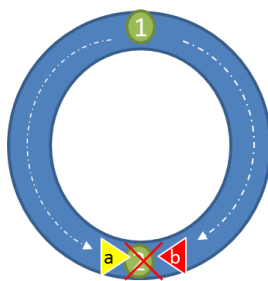
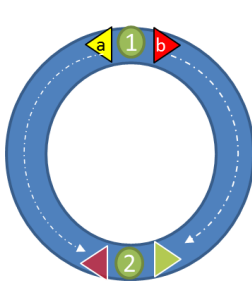
Figure 4-6a,b Half perimeter path 1 vehicle



Half perimeter path 2 vehicles

From the previous example, next consider the example in Figure 4-7b. This illustrates when two vehicles, **vehicle a**, and **vehicle b** both perform the same task as traverse from points 1 to just before point 2. Both vehicles are considered to be the same type of vehicle.

In this example, the vehicles are able to complete their tasks and traverse from point 1 to point 2, and would appear to cooperate with each other. However, suppose that the destinations for each vehicle were slightly adjusted such that vehicles have a target destination to complete traversal to just past point 2 as shown in Figure 4-8a. If the vehicles do not have any type of avoidance, and that they are identical vehicles, there would be a collision at point 2.



Figures 4-7a-c Intended Path

Intended Path - no cooperation

Updated Path with cooperation

If the vehicles exhibit collision avoidance they would reverse back to point 1, and repeat the process oscillating back and forth. In this case, the vehicles would not be cooperating with each other. However, suppose the vehicles can communicate with one another, and share the intended path solution, and the algorithms were able to recognize the intended path such that one of the vehicles was able to modify its original path. If then for example, **vehicle a** could rotate 180 degrees follow **vehicle b** until at its destination and re-rotate back 180 degrees thereby providing the intended solution. The vehicles are said to be *Cooperable* when two vehicles (or entities) are able to complete their tasks given the interaction between them does not impede their ability to complete their computable functions.

When examining this concept, several point solutions can be created to accomplish this task when the lack of *cooperability* exists. The challenge to this is during the development of the algorithms, can the algorithms be written in such a manner that enables them to be considered **Cooperable Algorithms** when two or more entities interact with one another? Specifically we provide a means that when two or more vehicles must interact and cooperate in a common space that they are able to do so. Compounding this problem is that in any given space, the number of vehicles is unknown, as well as their time entry into the region of space under consideration and their time of departure.

Thus far, vehicle behaviors have been developed to solve specific problems. To combat the challenge to the particular solutions, a more general approach that provides a higher degree of cooperability is provided for vehicle agents that share a common space where the following exists: First we define an entity as a vehicle to be a computation unit where each vehicle is modeled in the overall space as a single threaded process that interacts with each of the other entities. Therefore within a given space R there exists a distributed system of N processing units. Each of the entities in the overall systems processes algorithms concurrent to one another asynchronously. Additionally each entity processes data synchronous within its own computational system and communication between entities is available and considered necessary.

However, if each vehicle is able to calculate a potential interaction such that when there is a possibility of an interaction, then one of the vehicles projected path is updated to alleviate collision then the vehicles will be cooperable. Unfortunately, using imperative programming methods, such as coding in C, or python, to check for possible interactions, as the number of vehicles within the shared space R increases linearly, the complexity grows at an exponential rate [37], and [43]. Therefore the likelihood that a vehicle can in a near real time manner be able to determine an interactions is possible become computationally challenging [50].

4.5 COOPERABLE INTERACTION DEFINITIONS

When considering the behavior interaction between vehicles in a distributed sense, it is necessary to understand the intrinsic nature of these interactions. Currently, there is no formal definition of defining the relationship of entities through. Instead they interactions are considered strictly by application. Within a distributed approach to ITS there exists several possible interaction applications such as routing, platooning, and vehicles that enter uncontrolled intersections, or providing vehicles the ability to know about conditions further ahead in the road network that could potentially cause accidents. This is described by considering the vehicles to be cooperable with one another. Presented here is an initial framework that initiates this work.

4.5.1 COOPERABLE INTERACTION DEFINITIONS

To understand what is meant to know if an interaction, or possible interaction between vehicles can or will occur during travel it is required that an agent be aware of potential interactions with over agents, and that the agents are capable of cooperating. By doing so, the notion of *Cooperable Interaction* to consist of the following definitions:

Definition 1:

$E_1 \text{ Co } E_2$ *Entity 1 and Entity 2 are Completely Cooperable*

If entities do not interact then the outcome of the interaction operator results between the two entities is 1.

Furthermore, if all Entities are cooperable with each other, then the group of entities is considered to cooperable defined as:

$$\prod_{n=0}^m I_n = 1$$

Where $I \in \{0,1\}$

Definition 2: $E_1 \text{ Cx } E_2$ *Entity 1 and Entity 2 are not Completely Cooperable*

If entities interact then the outcome of the interaction operator results between the two entities is 0..

$$\prod_{n=1}^m \mathbf{I}_n = 0$$

Where $\mathbf{I} \in \{0,1\}$

Definition 3: $E_1 \mathbf{C} E_2$ Entity 1 and Entity 2 can be Cooperable

$$\sum_{n=1}^m \frac{\mathbf{I}_n}{n} > C_{threshold}$$

Given m Where $C_{threshold}$ is set by time – space constraints

Furthermore, we define an **Entity 1** as a computational agent with a given **action plan** 1 shown as:

$$E^{Plan}_{Entity}$$

Where an **action plan** is defined as a set of *actions* that define a behavior from points p_1 to p_2 during times t_1 to t_2 . It consists of a piece-wise set of connected actions. Furthermore, an entity may evolve a given plan n number of interactions until a cooperable interaction is found.

The interactions of entity 1 and entity 2 is described as

$$\mathbf{E}^1_1 \mathbf{I} \mathbf{E}^1_2$$

If E1 interacts with E2 during or on a given segment then either E1 or E2 must change its course of actions (COA) this is expressed by the following computational statement.

$$\text{If } \mathbf{E}^1_1 \mathbf{I} \mathbf{E}^1_2 = \mathbf{Cx} = \mathbf{e} > \mathbf{E}^1_1 \rightarrow \mathbf{E}^2_1 \rightarrow \mathbf{E}^1_1 \mathbf{I} \mathbf{E}^1_2 = \mathbf{Co}$$

Where : $=\mathbf{e} > \dots \rightarrow \dots \Leftrightarrow$ is where the action plan evolves from the current to the next

Properties of Interaction identified thus far can be described in Table 2 as:

Table 4-1. Interaction Properties

Property	Name	Notes
$\mathbf{E}^1_1 \mathbf{I} \mathbf{E}^1_1 = 1$	Identity	An Entity interacts with itself
$\mathbf{E} \forall \mathbf{A}_1 \mathbf{I} \mathbf{E} \forall \mathbf{A}_2 = \mathbf{E} \forall \mathbf{A}_2 \mathbf{I} \mathbf{E} \forall \mathbf{A}_1$	Commutation	Entity 1 for all action plans can interact with Entity 2 for all actions plans the same as vice versa
$(\mathbf{E}^1_1 \mathbf{I} \mathbf{E}^1_2)' = 1$	Inverse	There exists no interaction between \mathbf{E}^1_1 and \mathbf{E}^1_2
$[\mathbf{E}^1_1 \mathbf{I} \mathbf{E}^1_2] \mathbf{I} \mathbf{E}^1_3 \neq \mathbf{E}^1_1 \mathbf{I} [\mathbf{E}^1_2 \mathbf{I} \mathbf{E}^1_3]$	Associativity	Associativity does not exist

4.5.2 ACTION SEGMENTS AND PLANS AND TURTLE PLOTS

We also recognize that Interactions can occur in both time and space where no two entities can exist at the same time at a given location. By taking this into account, an interaction occurs when the location then time has coincidence within some epsilon area or epsilon time for a given segment within the action plan known as an ***action-segment***.

Furthermore it is recognized that there can exist more than one interaction between two entities for a given segment. In this case there are ***m*** interactions between entity 1 and entity 2

$$E^1_1 \mathbf{I}_m E^1_2 = 0$$

When two entities share a common space and/or time, there can exist n interactions between two entities. Therefore, an ***Action Plan*** provides a list of n actions that include segments and durations. From this an action plan is defined as follows:

Action Plan A1 is defined as follows where

$$A1 = [(Segment_{x0}, Segment_{y0}, t_0), \\ : \quad : \quad : \\ [(Segment_{xn}, Segment_{yn}, t_n)]]$$

Formally, we define an action plan to be a list of segments that are geometrically a linked list of estimated position and time per the given action plan. For each segment between the two points, there is a set of discrete slots known as a **slot-list** where interactions can occur. An action plan is the current list of segments used to describe a route from start to end. Within an entity, there may contain one (or more previous action plans) current action plan. The set of all action plans compose the **Course of Action**. The **Course of Action** may only have one action plan, but it could have updated more than one with the most recent action plan being the current plan in use. To describe segments within an action plan, we introduce the concept of a **Turtle Plot** that pictorially describes the slot list for a given segment. We will use *Turtle Plots* to describe the potential interaction for the slots.

To describe the relationship of when and where spatially an entity exists we introduce the concept of a Turtle plot. A “Turtle Plot” shown in Figure 4-9 is plot that illustrates time and /or space interaction possibilities. They contain a list of geometric location and time slots in discrete steps. Turtle plots are plots where

vehicles when sharing information about their location share their slot list to identify possible interactions. When a slot interaction occurs the slot list is updated based on the code design hence updating the action list for the particular entity.

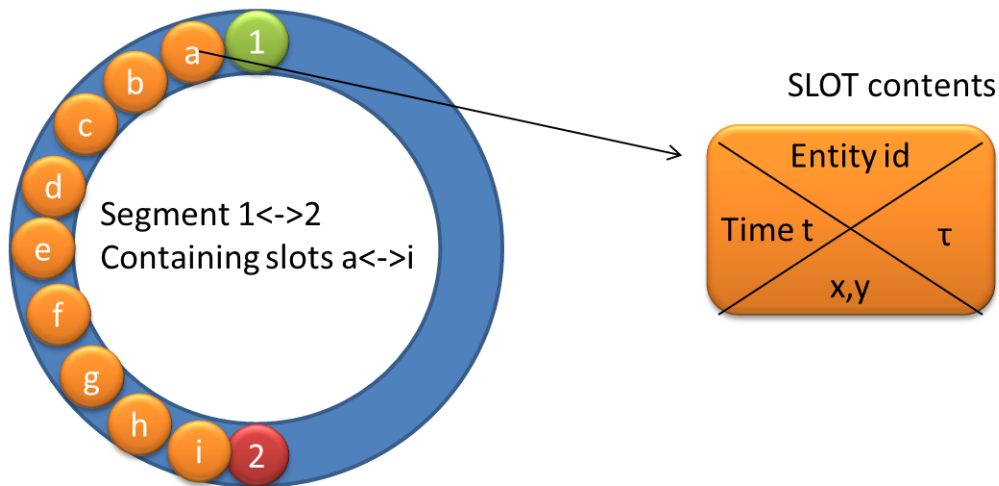


Figure 4-8 – Basic Turtle Plot

The slot list is a tuple comprising of :

{Entity ID, “universal time”, estimated position (x,y), τ } within an action list segment

Within a slot list when overlap occurs there is a more likely an interactions creating a failure, cooperability thereby producing **Cx**. Slot Lists are communicated with all vehicles with the radius R to determine if interactions occur. The position is in a global reference frame common to all entities, i.e. latitude and longitude for the position. The time is time at which the entity will be at point 1, and is from a

common global source. The parameter τ is the time offset from that time t at point 1.

Figure 4-10 illustrates how two Turtle Plots are used to check for possible interactions on a circular vehicle path.

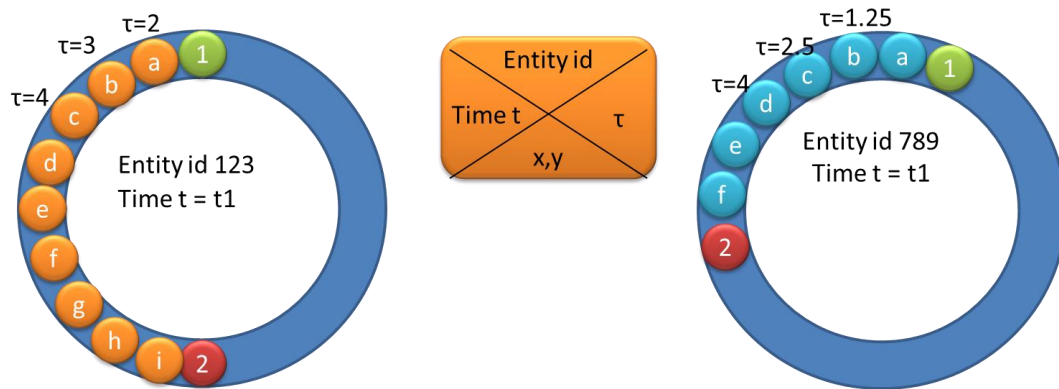


Figure 4-9. Example of two turtle plots that will have their slots compared.

In Figure 4-10, it is shown how that depending on the τ for each (x,y) position there is an overlap and one entity must evolve its action for this segment. Spatial overlaps occur between entities 123, and 879 as shown:

$$123 \cap 789 = 123\{1, a, b, c, d, e, f\} \& 789\{a, b, c, d, e, f, 2\}$$

When including estimated time slots additional overlaps occur between the two entities in the d-slot,3 receptively.

Next, the Slot Lists are shared with other entities within some range to assess by the arbiter processing located within each entity. The Arbitrators use these interaction algebra rules to update course of actions to avoid interaction hence achieving cooperabilty.

When a system is projected at time t such that no interactions occur between entity 1 and entity 2, then the system is Cooperable. That is

$E_1 \text{ Co } E_2$ *Entity 1 and Entity 2 are complete Cooperable*

$$\prod_{n=0}^m \mathbf{I}_n = 1$$

When each entity with the set of all entities computes the possibilities of interaction, each entity communicates its first interactions via the entity closest within range r . If an interaction is predicted to occur the action list segment must be updated.

Arbitration rules

1st: The entity with the lowest inertia updates first based on entity meta data

2nd: The entity that has greater route alternative options

3rd : The entity whose cost function yields similar results without an evolution to the action within some threshold

4th : The entity who hasn't adjusted yet. *Rational: If an entity has already adjusted it was because of impact to another entity already for that segment.*

5th : In the event of all other criteria results in ties, compare the entity ID. The lower number evolves its action for the segment (actual cost functions)

4.6 APPLICATION OF FUNCTIONAL REACTIVE PROGRAMMING PARADIGM

When an entity is required to update its action, various competing cost functions will dictate the new action plan based upon the vehicles intended use and route. For example, route re-plans can be cost functioned based on entity type, and its cost functions for energy efficiency, density, entity similarity. Implementation of such a scheme should not be considered trivial. Once each entity shares its slot lists and action plans with the other vehicles in radius r , neighbors, it is highly likely that an interaction is estimated to occur. When then happens the appropriate entity updates its action plan and re-shares it various slots. The challenge is to be able to propagate the updated information. The event loop programming paradigm of call backs will quickly reduce the computational capacity of the processor as the geometric growth in complexity occurs [51], nor is it predictable as to when this will occur.

To combat this challenge to meet the constraints of near real time, and asynchronous events, the implementation is performed using a functional reactive programming (FRP) technique first discussed in [52]. Finally, the slot length parameter can be variable that allows spacing to increase indicating that possible route options should be chosen by lessening the spacing to increase or decrease possible route option. The entity interaction calculation is performed to ensure a

vehicle using methods of functional reactive programming to address the call back observer stream challenge when the decision process must cope with only data streams that are changing as opposed to constantly performing the call-back operations.

FPR is able to handle the varying nature of asynchronous event driven in a high-level way to work with interactions. [53]. It provides control flow structures for *time*. Additionally, it uses the observable pattern but iterates over the observables within the pattern to detect a change from the observable without requiring the individual call backs. Therefore, as slot lists propagate through the radius r of an entity, the behavior algorithm reacts to changes in the streams from the other entities while being able to maintain the near real time performance necessary.

By understanding the interaction between entities, a system of distributed entities can cooperate within a shared time and space region. Each entity can use its own arbitrating cost functions and rule sets to carry out its intended task. By sharing the information where interactions can occur, the entity is capable of not interfering with other entities within these spaces and times. Therefore, basic behaviors such as follow the leader are Cooperable, however arbitration behaviors must exhibit valid entity interactions to remain Cooperable.

Chapter 5 Implementation – Application to Distributed Routing

To validate the theory of approach discussed in prior sections, implementation was performed using an event step traffic simulator SUMO. Furthermore, approaches were coded using custom python modules that interact with the simulation environment. During the simulation, performance metrics were gathered from the simulation in code for additional analysis to compare the metrics.

5.1 CENTRALIZED ROUTING ALGORITHM

The development of the analysis focused on implementing the Cell 1 architecture and the Cell 9 architecture. These were chosen since they are considered to be the extreme conditions of the topologies. When considering the other topologies, those are considered combinations of cell1 and cell9 either in the data gathering and/or the data processing. Furthermore, as a baseline, the non-ITS approach was implemented as well. The provided a basis of comparison which is general use for current non-ITS solutions.

The Cell 1 topology discussed earlier captures data and processes the data in a centralized is shown in listing 5.1 and is described as follows.

At each time t , the loop sensors are sampled to determine the velocity of the vehicles passing over them. The data are captured within each link of road network and the average velocity of each link from the loop sensor is calculated. If the velocity for the last link is less than the expected link velocity for the given allowable density, then a possible route alternative is provided to those vehicles. However, since per the discussion of discriminating types of vehicles, vehicle type is used to select routes more conducive for the vehicle type, the route is chosen based upon the next available route for that type of vehicle. An available route is one that is capable of supporting the diverted traffic and thus above its given threshold. This algorithm would be required to reside within the infrastructure. While the approach is simple, the challenge becomes being able to service that large number of request and processing for reach vehicle within in a real-time aspect.

The vehicle would be connected to the TMC in a virtualized point to point connection via the complex network structure outlined in the ITS national architecture. Furthermore, more than one unrelated incidents within an TMC's control would be possible, though the rerouting of the vehicles associated with each

incident would be unrelated. However, the TMC processing would be required to process the vehicles in each incident thereby increasing the processing loading demands.

Algorithm 5.1: Centralized ITS Routing Algorithm

```
foreach LoopSensor in Network
{
    AcquireCurrentVelocity()
}
foreach link
    FindLinkAverageVelocity()
    If Link Average Velocity < VelocityThreshold
        foreach vehicle
            GetVehicleType()
            Get Possible
            RouteAlternativeListVehicleTypeAvailable
            Route=RouteAlternativeVehicleType_NextAvailable
        }
    }
```

Listing 1 – Centralized ITS Algorithm

5.2 DECENTRALIZED ROUTING ALGORITHM

Conversely, the algorithm described in the topology of Cell 9 is one that implements the fully distributed data acquisition and the decision. This is shown in listing 5.2 and is described as follows.

For discussion of the algorithm consider two different vehicles

- 1) **Vehicles_Foward:** Vehicles ahead of a particular current vehicle pertaining to a discussion.
- 2) **Vehicle_Particular:** A particular vehicle that has neighbors ahead of it on a given link.

Each vehicle maintains a list of vehicles that are within a range r . For these vehicles that are within that range, the current position, velocity, route and type are shared amongst each neighbor within that range. As each vehicle receives incoming streams of data from its neighbor vehicles, each vehicle manages its behavior based on a set of behavior rules. The neighbor vehicles ahead report back their current velocity the vehicles behind. If the **Vehicle_Particular** determines that the **Vehicles_Foward** still maintain a sufficient velocity for that type of for that specific **Vehicle_Particular**, then the **Vehicle_Particular** maintains its current route.

However, if the **Vehicles_Foward** velocity is not sufficient, then the **Vehicle_Particular** looks for the neighbors within range r that of the **Vehicle_Particular** type on different routes that satisfy the velocity threshold. If

there are no vehicles on different alternative routes with routes available that meet the velocity threshold, the **Vehicle_Particular** selects the best next available route conducive for that type of vehicle.

Algorithm 5.2: Decentralized ITS Routing Algorithm

```
// Each vehicle preforms the algorithm concurrently in its own time-base
AcquireVehcilesCurrent [Px, Py, Theta, Vv]
Find Neighbor Vehicles within r of [Px,Py]
Foreach NeighborVehicle
{
    getNeighbor Vehicle (position, route, velocity, type)
    if (SameRouteNeighborAheadVelocity >VelocityThresholdVehicleType)
        then
            Route=CurrentRoute
        else
            using closestVehicleSameType
            Foreach SameVehicleTypeNeighborDifferntRouteAhead
            {
                if SameVehicleTypeNeighborDifferntRouteAhead > VelocityThreshold
                then Route=
closestVehicleSameType[currentSameVehicleTypeNeighborDifferntRouteAhead]
                else if not EndOfNeighborList use next_closestVehicleSameTy
                    else
                        route=next_availableRouteForVehicleType
            }
        }
}
```

Listing 2 - Distributed ITS Algorithm

The challenge with this type of algorithm is that it depends on incoming data streams to implement these behaviors. Furthermore, the streams occur in an asynchronous manner. As each stream is received the code implementing the algorithm must ensure that it doesn't block on the incoming streams while waiting for the data. The incoming streams are considered to be following the observer pattern which would give rise to difficulty creating a structure of code that is not susceptible to ad-hoc implementation and difficult to test.

However, when further analyzing this approach it is recognized that the solution is a specific solution. While it works well for this type of situation the question about knowing alternative applications of success remains a question. For example, each approach that requires different distributed decisions would be generated from a standalone perspective. Moreover, because each agent is in its own processing space and in the real world asynchronous to one another testing the approach and implementation will become challenging due to the interaction of the various parts within in the overall system. Although an agent may work by itself, there is no guarantee it will work with other agents, as demonstrated in section 3 when discussing the circumference traversal problem. The ad-hoc nature of the approach still makes this a difficult solution to accept to ensure that vehicles once autonomous on the highway will be able to interact with one another in a cooperative manner. Therefore the next step in this process is to reexamine this

approach where a second look at how to construct the system using a separate method for the programming.

FRP manages data n asynchronous data streams in an efficient manner [54] and [55]. Because of the capability of this programming approach, the use of FRP is investigated as an approach to work with the various asynchronous streams of data from agents within connected to one another[57]. This provides a means to share the data set that is observable to the neighboring vehicles.

5.3 IMPLEMENTING FRP FOR DECENTRALIZED VEHICLE ROUTING

A solution for performing a decentralized routing is presented in Figure 5-1. This figure describes the algorithm 5.2 which uses a traditional approach of the observer pattern. Within this design, there are several applications where each vehicle that is connected to the vehicle performing this algorithm is observed for its changed status.

If this algorithm is decomposed, it is recognized that there are fundamental behaviors that can be captured. These fundamental behaviors are then illustrated by common coloring of states as shown here in Figure 5-2.

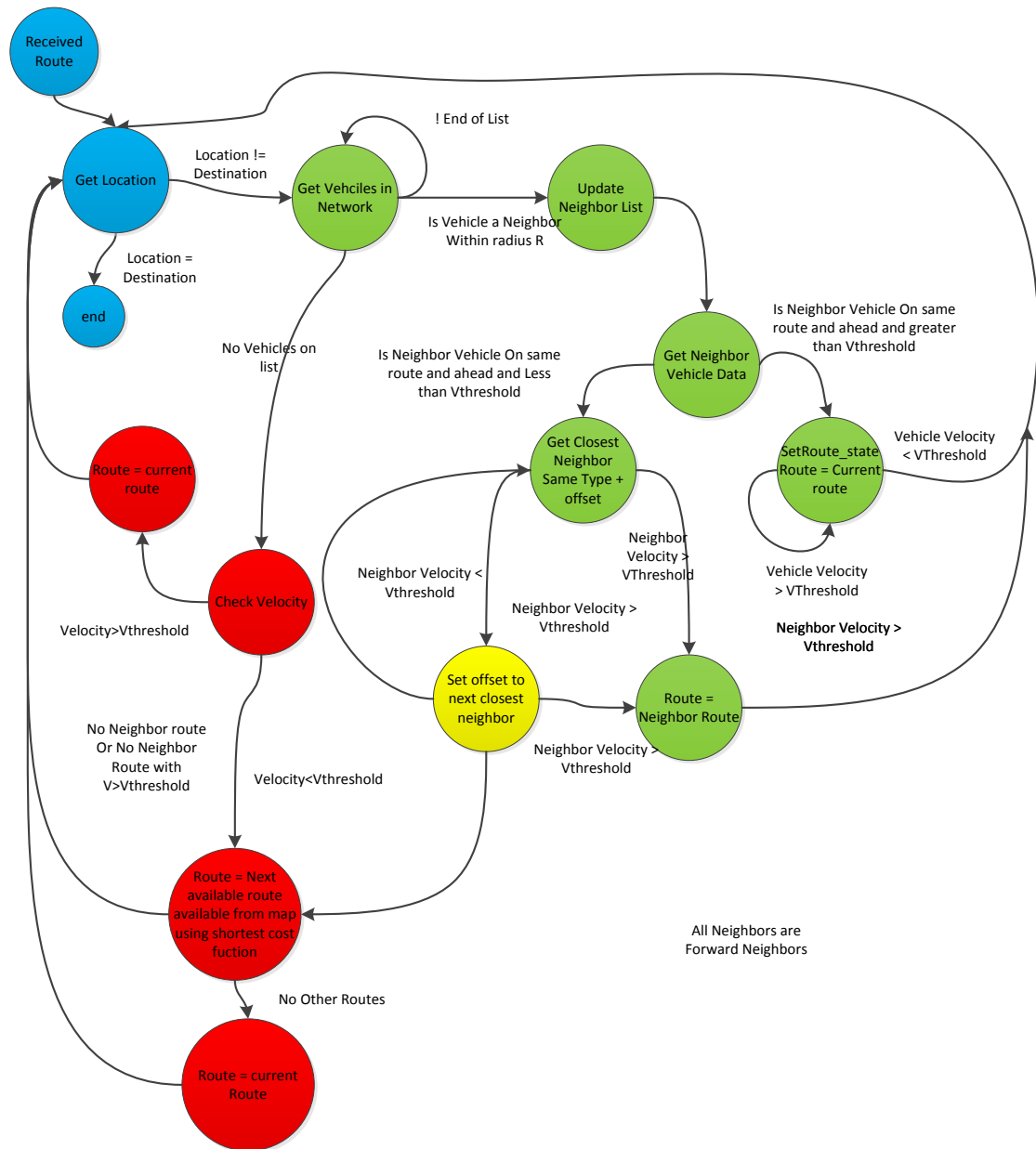


Figure 5-2 - Grouped Distributed Vehicle Algorithm State Machine

Looking at the state machine by color the behaviors are further described as the following in Figure 5-3:

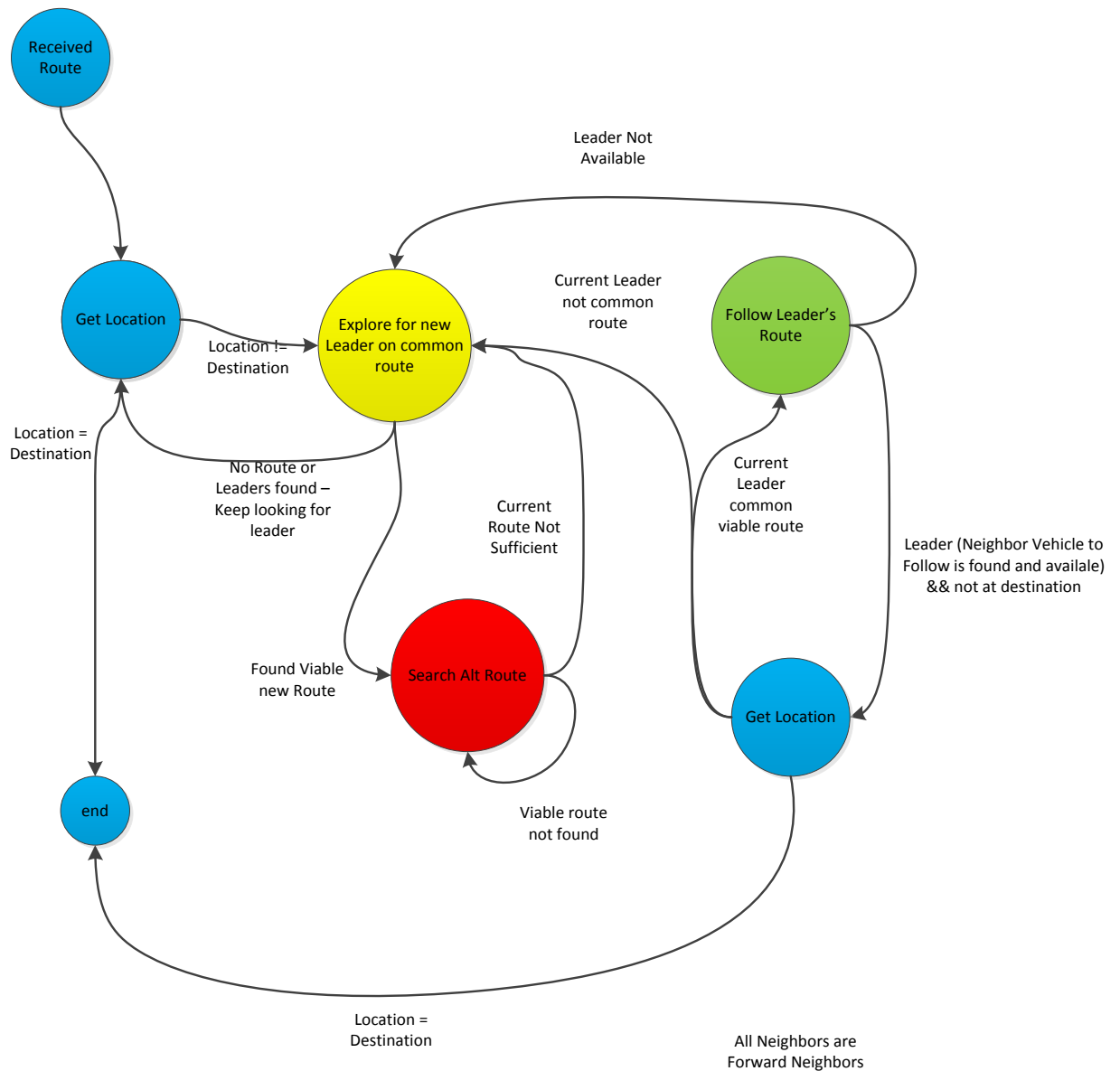


Figure 5-3 – Behavior Distributed Vehicle Algorithm State Machine

In Figure 5-3 each color state namely Red, Yellow and Green are considered to be fundamental behaviors. Furthermore, these very behaviors are implementable easily using FRP methodologies especially as discussed in [45], [53], and [55]. The original algorithm was that was coded using imperative methods, was then recoded using an FRP framework and achieved similar results in system level throughput, however each function became an encapsulated function within the overall behavior. Taking this approach provides an approach to the distributed algorithm behavior that improves the overall composability that is otherwise difficult to construct.

Chapter 6 RESULTS

To analyze the differences between and cell 9 topologies and algorithms in listings 5.1 and 5.2, the algorithms were coded using Python, initially using an imperative method, and later using an FRP approach making use of the Python extensions for FRP known as *Trellis*[58]. To perform the analysis of the differences from the distributed routing algorithms vs the centralized routing algorithm, the system was tested without any form on ITS involved. To perform the simulation, Simulation of Urban Mobility (SUMO, see [59]) was utilized. Using SUMO, a set of simulations was constructed which were controlled by the python code and interfaced to the SUMO simulator via the Traci Interface [59]. SUMO is an open source, microscopic, multi-modal traffic simulation software package. It allows a modeler to simulate how a given traffic demand which consists of single vehicles moves through a given road network. The simulation allows users to address a large set of traffic management topics. It is purely microscopic: each vehicle is modelled explicitly, has an own route, and moves individually through the network. Though SUMO is open source and C modules can be compiled into the main body of the program to alter its capabilities, one of its strongest attributes is the ability to interface to the simulation engine per event step via an interface known as the Traffic Control Interface.

The TraCI connectivity is achieved via a network connection either local on the machine or via a machine on the same network as the SUMO simulator. TraCI communicates directly to the simulation engine and allows most parameters for each vehicle to be read and/or manipulated through the API commands. It also provides access to the road network parameters, signals and the infrastructure. This provides a means to acquire the velocity of the vehicles as well as position loop sensors for data acquisition, at each event step. Therefore, any parameter updates are done without affecting the time base in the simulation. There are TraCI modules written in Python, Java, and MATLAB, and C# to date. By using TraCI, the various algorithms for the vehicles, supporting a standard non-ITS and the ITS approaches are able to be evaluated by implementing the behaviors about the road network and vehicles into modules. Using TraCI provided this feature for SUMO, not really supported in the other simulation tools, allowing for specific behaviors for each vehicle to be implemented. [56]

6.1 SIMULATION ENVIRONMENT - SUMO

To perform the analysis, a representative road network was constructed. Shown in Figure 6-1 is the screen shot from SUMO illustrating the scenario discussed in the previous section. Figure 6-2 illustrates a screen shot from SUMO of a vehicle approaching a loop sensor on a given link, while Figure 6-3 illustrates a signaled intersection being simulated. SUMO provides signal phase and timing and also

handles the appropriate vehicle lane manipulates according to their car following and generalized logic. However, these behaviors are able to be manipulated using external code via TraCI using python and other languages.

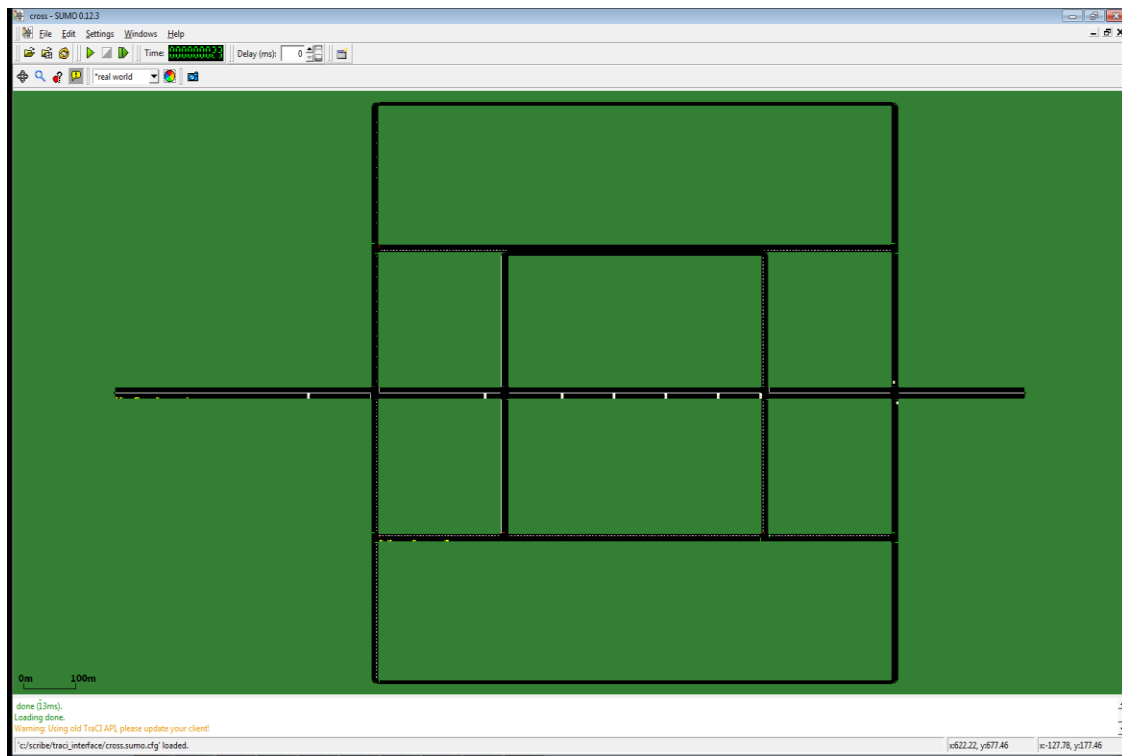


Figure 6-1. Basic hypothetical road network using SUMO for simulation.

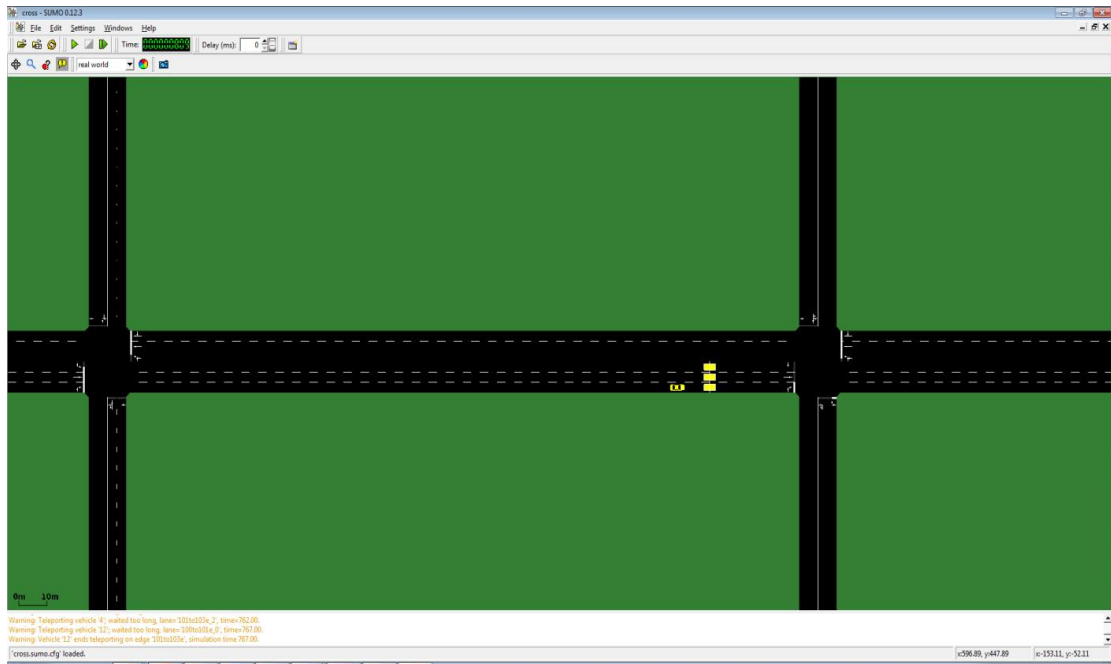


Figure 6-2. Basic SUMO link with a vehicle approaching and loop detectors at an intersection.

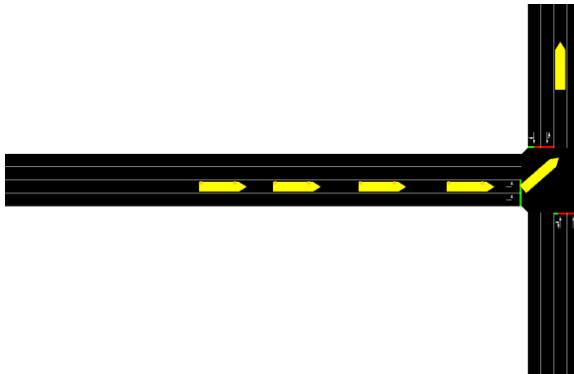


Figure 6-3. SUMO manages and provides the signal phase and timing for controlled intersections.

For analysis, the road-network shown in Figure 6-2 illustrates a roadway typical set of links. The center roadway outlined in red is a freeway without any controlled intersections but contains loop sensors along the links which shown in Figure 6-4. Shown in Figure 6-5 are the ramp links outlined in orange.

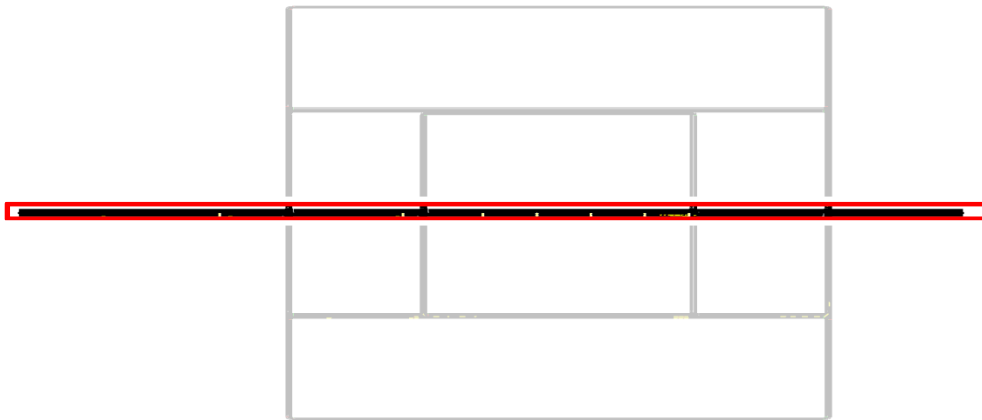


Figure 6-4. Primary Freeway link with loop sensors used in centralized solution that vehicles use as initial route.

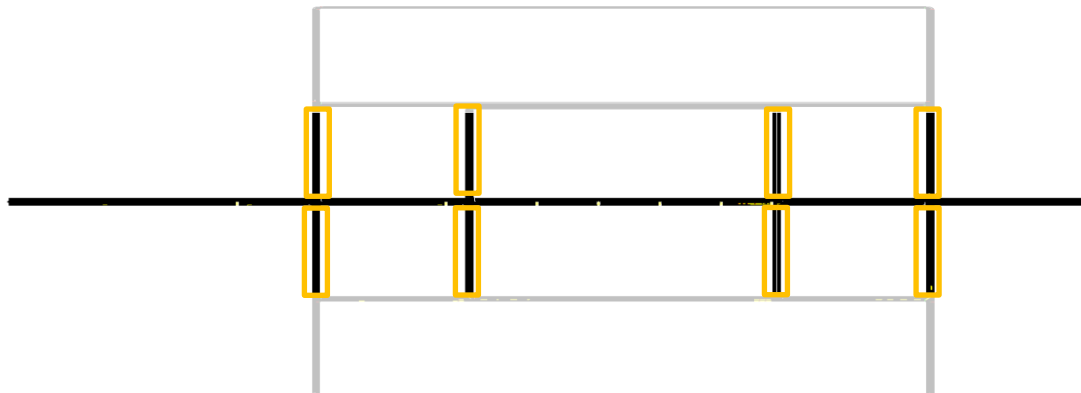


Figure 6-5. Freeway off-ramps to the arterial routes.

To analyze and compare the algorithms for routing based upon centralized vs decentralized approaches, two simulations that exhibit an incident were created. The first simulation is a hypothetical freeway with arterial roads to the north and south of it, while the second simulation was based on a real road network using vehicle volumes and rates attained using PeMS [1]. Figure 6-6 shows where the incident occurs on a freeway whose location is denoted by the circle. The solid yellow arrow indicates the freeway and the flow for the freeway with loop detectors placed every 100 meters. The dashed lines indicate the rerouting alternatives the vehicle would take if they wanted the shortest route around the incident. In these simulation scenarios, vehicular traffic on the freeway consists of 20% delivery trucks, and 80% passenger cars. The volume of vehicles per hour was modeled was set to 3600 vehicles per hour (VPH).

In addition to the hypothetical model and network, an actual segment of freeway was used to analyze the various approaches. This is shown in Figure 6-7. This freeway corridor is located approximately 20 miles east of Los Angeles and has arterial roads Ramona, Central, and Euclid providing North/South routes as well as the arterial roads Francis, Philadelphia, Walnut, and Riverside that provide East/West routes. Other North/South routes do not have ramps to the freeway. Figure 6-8 provides the distance between the arterials and the freeway in units of meters.

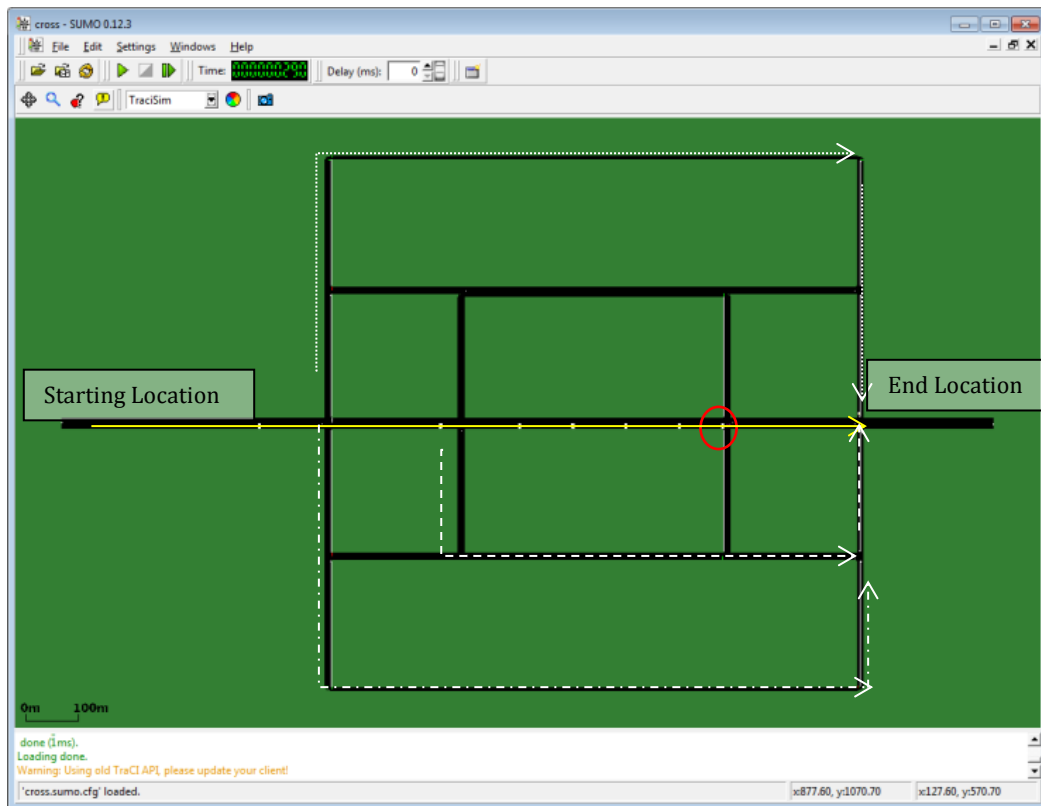


Figure 6-6. SUMO Hypothetical simulation environment used to model the simulation in conjunction with Traci and Python

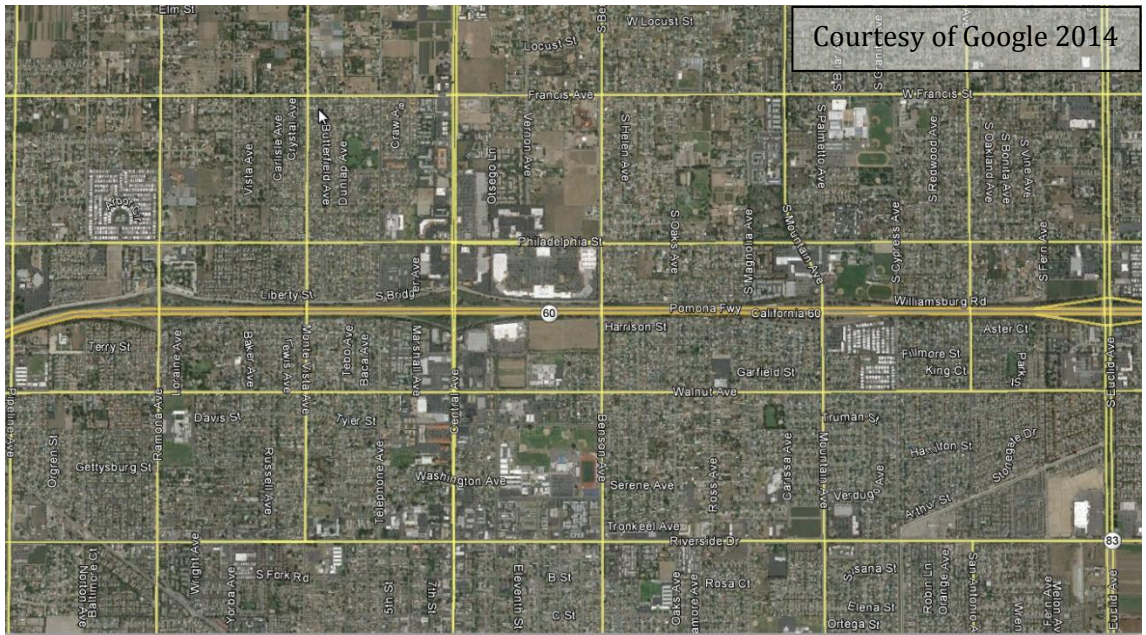


Figure 6-7. CA SR-60 Corridor Ramona to Euclid used for simulation and analysis for routing algorithms

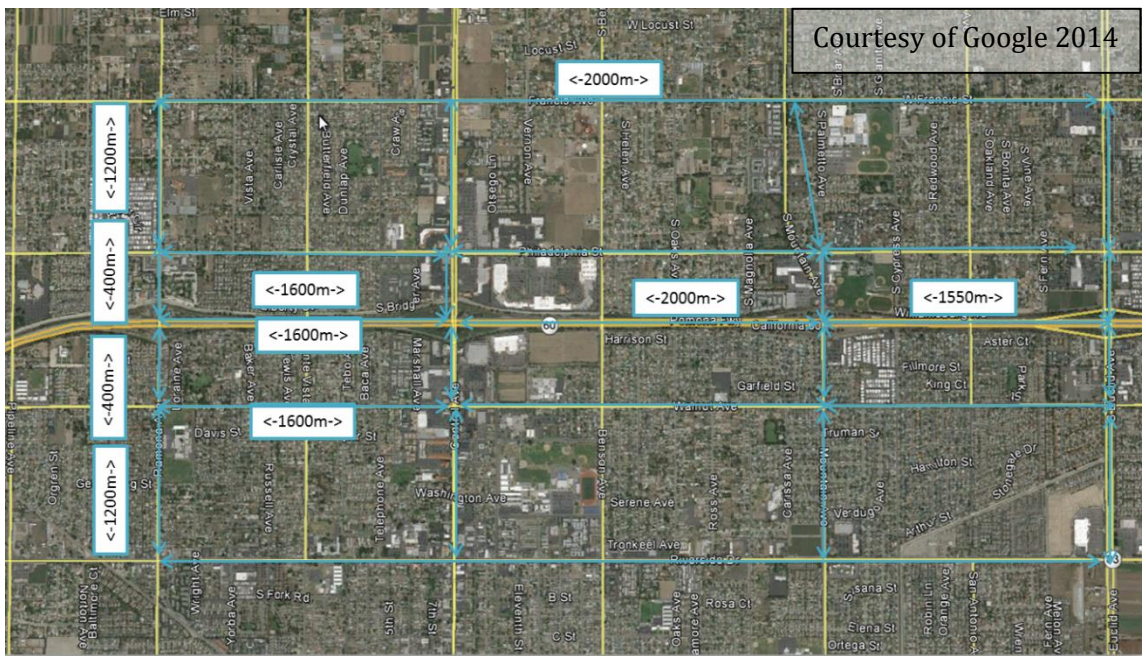


Figure 6-8 CA SR-60 Corridor with distances indicating the available routes for the road network.

For each simulation scenario performed, the impact of the alternate route(s) on various arterial traffic flows was observed, as well as the vehicle travel time from the freeway was measured. The different conditions for arterial flow rates include the maximum flow for arterials, $\frac{1}{2}$ of the maximum the flow for the arterials, noted as medium arterial loading, $\frac{1}{4}$ of the flow noted as light arterial loading, and finally no arterial traffic. The maximum arterial loading was set at 1800 VHP, 900 VHP, and 450 VHP respectively.

6.2 TYPICAL ROAD NETWORKS

The scenario shown in Figures 6-9 and 6-10 are very typical road networks from the real world, and hence the rational for selecting this as the targeted structure for the hypothetical network. For example, a very common road network that exhibits very typical incidents is the California (CA) SR-91 State Freeway through eastern Orange County in California. This route uses the freeway as primary access between the western Riverside and South Western San Bernardino Counties and often exhibits traffic incidents. The freeway segment is between the CA SR-55, and the Eastern Transportation Corridor Freeway CA SR-241. In addition, there are arterial road networks that flank the north and south sides with different alternatives. These arterial alternatives themselves can exhibit local traffic loading without freeway incident impacts. To the North of the freeway La Palma Rd, and further North of this

is Orangethorpe. To the south of the CA SR-91 segment is E. Santa Ana Canyon Road, and further south is Nohl Ranch Road / E. Canyon Rim Rd.

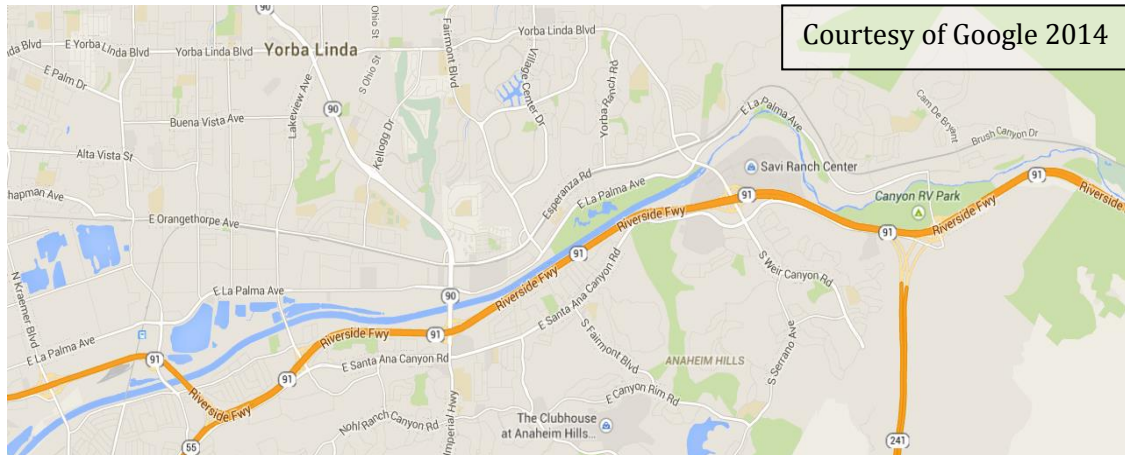


Figure 6-9 Eastern Orange County CA SR-91 Corridor

Another similar corridor along the CA SR-22 which is further west located between the Interstate 405 and Interstate 5. The north flanking arterial routes are Garden Grove Blvd. and Chapman Ave. while flanking the south is Westminster Blvd. and Hazard Ave.

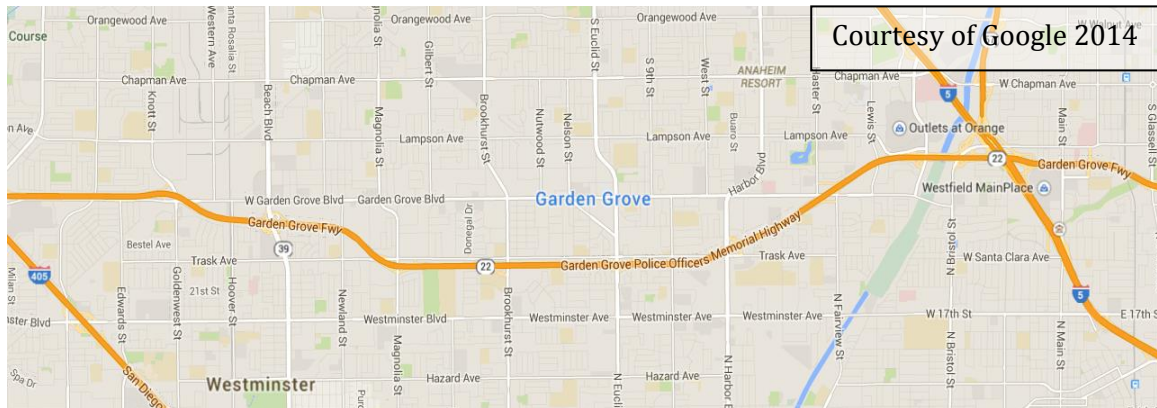


Figure 6-10 Central Orange County CA SR-22

6.3 MODELING EXISTING ROAD-NETWORK CA SR-60

To validate the hypothetical network presented in section 6.2 data was from obtained from the PeMS [1] database and was used to model the route in the Los Angeles California freeway system shown in Figure 6-7. As presented earlier, this region of the freeway includes the arterial roads from Ramona to Euclid. To the north of the freeway is Philadelphia, and further north is Francis. To the south is Walnut, and further south is Riverside. The model is supplied with vehicle traffic based upon data captured during the month of April of 2009. The data captures the volume of vehicles per hour for each day of the month every hour. Shown below in Figure 6-11 is the volume in vehicles per hour for the month of April 2009.

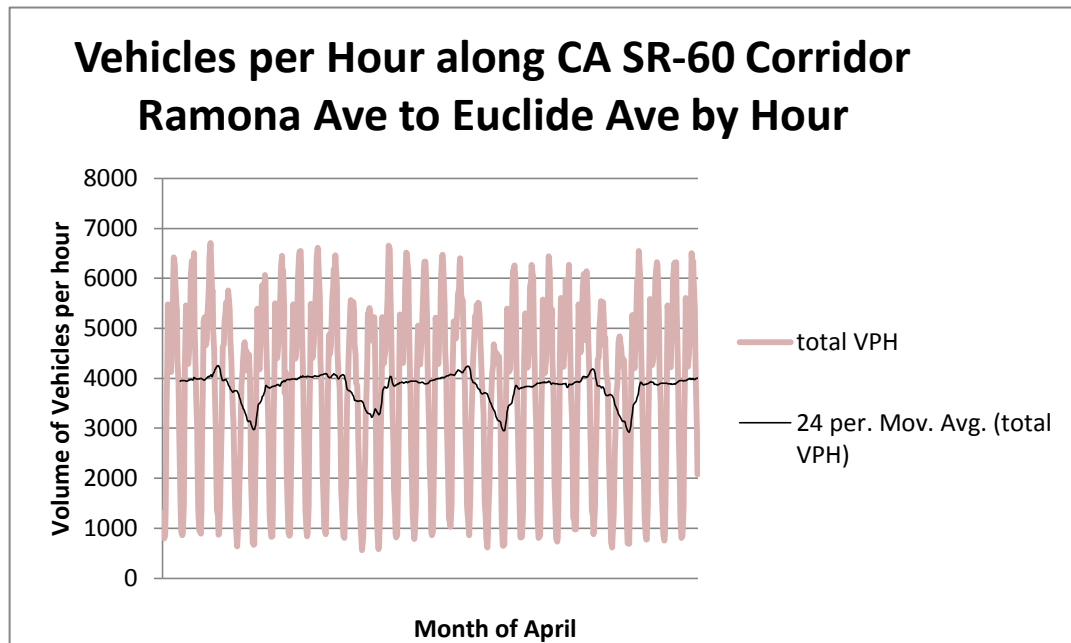


Figure 6-11 Volume of traffic along Corridor CA SR-60 during the month of April used to characterize the simulation model

To perform the simulation, a 24 hour moving average was computed to model the routes along the CA SR-60. Arterial traffic was modeled using four different loading values. This results in a typical vehicle per hour (VPH) or approximately ~4000 total across all lanes. Using this data, the simulation was performed at this loading the free route at 4000 VPH.

6.4 SIMULATIONS RESULTS OF ROAD NETWORKS

Simulations for the hypothetical and CA SR-60 corridor were performed capturing the time of travel for the vehicles to reroute around the incident. For each simulation, SUMO was configured to provide a link coloring to indicate the volume of vehicles for a particular link. The screen output simulation was then captured with various screen shots provided to illustrate the amount of volume for key areas. The routes are colored using SUMO's link density settings. When the link is 50% vehicles and 50% non-vehicle for the space of the link, the link is colored red. The coloring is a linear gradient between green to red with yellow set at the midpoint. The link is green at free flow, however as the volume increase and vehicles begin to queue, the color changes from green to yellow and finally to red. This provides a visual comparison to compare the effectiveness of the distributed routing results.

6.4.1 NON-ITS BASED RESULTS FOR A HYPOTHETICAL ROAD NETWORK

As a baseline, the condition where no ITS methods exist was simulated. This is the condition where no knowledge of the state of the network is known and only the best route that has the shortest deviation around the incident was initially examined. As discussed earlier, this is in the case of greedy routing, where drivers reroute themselves without any traffic information, based on their local knowledge. During this simulation, when the cars queued up to where the existing off-ramp,

they exited and reroute to the arterial roads. Furthermore, each arterial intersection is a controlled intersection with a traffic light. The overall throughput of the simulation was measured as is shown in Figure 6-12 by capturing the time the vehicle at the starting location until it travels to the end location measured in seconds as shown in Figure 6-6. It should be noted that the time is the simulation time indicated in the upper right corner the SUMO window. When each vehicle enters the map, the simulation time is captured. When that vehicle passes the end location, its vehicle's time is captured again to arrive at the simulated travel time. The travel time of each vehicle was captured and a histogram of travel time bins created shown in Figure 6-12.

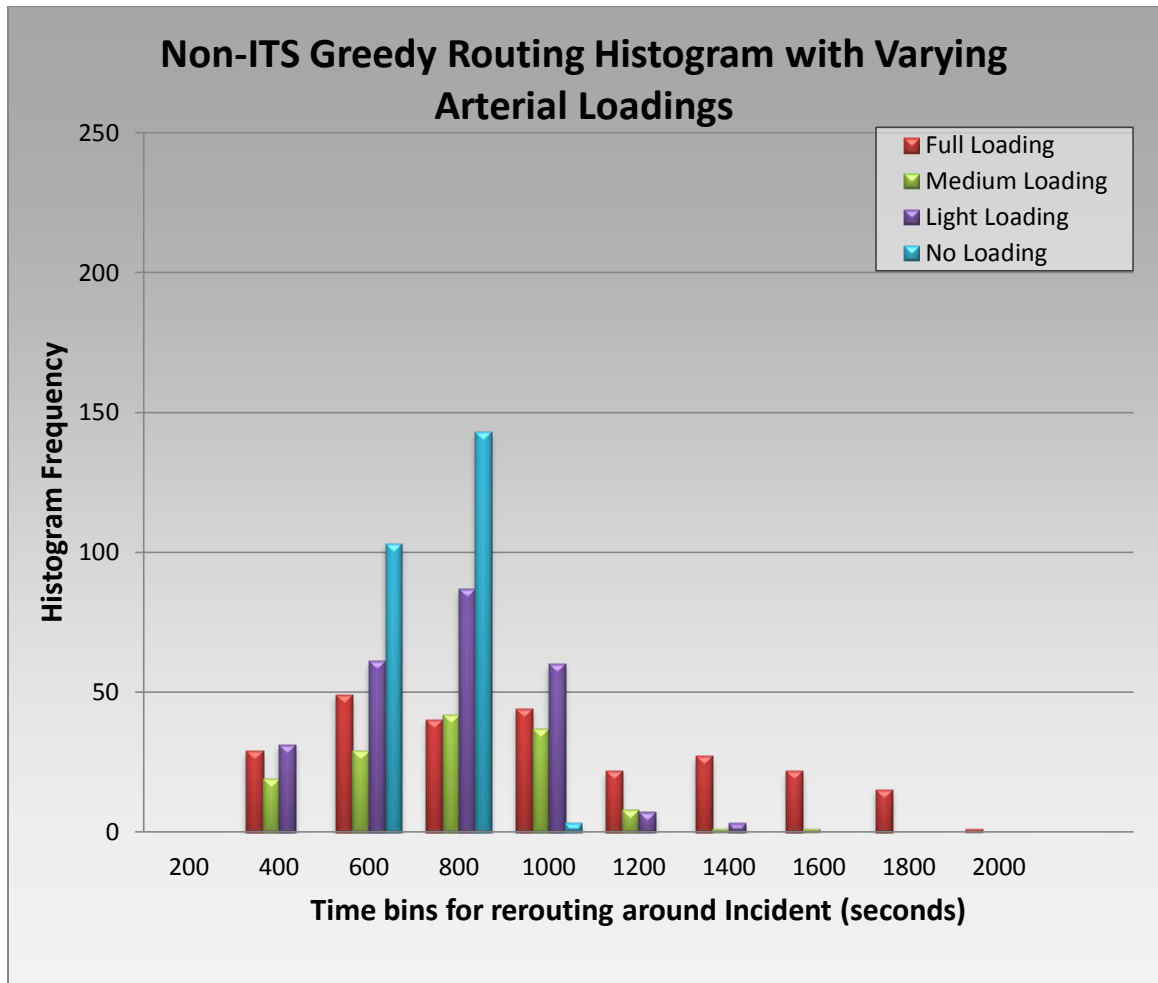


Figure 6-12. Histogram illustrating the times to travel from the origin prescribed destination using Greedy Routing Algorithm and no ITS methods were employed

In the non-ITS routing the histogram illustrates that duration of travel that the routing around the incident incurred. When there was no loading from the arterial roads, the bulk of the routing times occurred in of 600 to 800 seconds. However as loading increased, the number of increased to include a frequency of 20 vehicles that incurred up to a 1800 second delay due to the incident.

Shown in Figures 6-13a-f are the screen shots taken of the simulation in progress using the hypothetical road network that does not provide any ITS routing aid to the vehicles. It illustrates over time how the network's queues lengths and volumes increase in a given link.

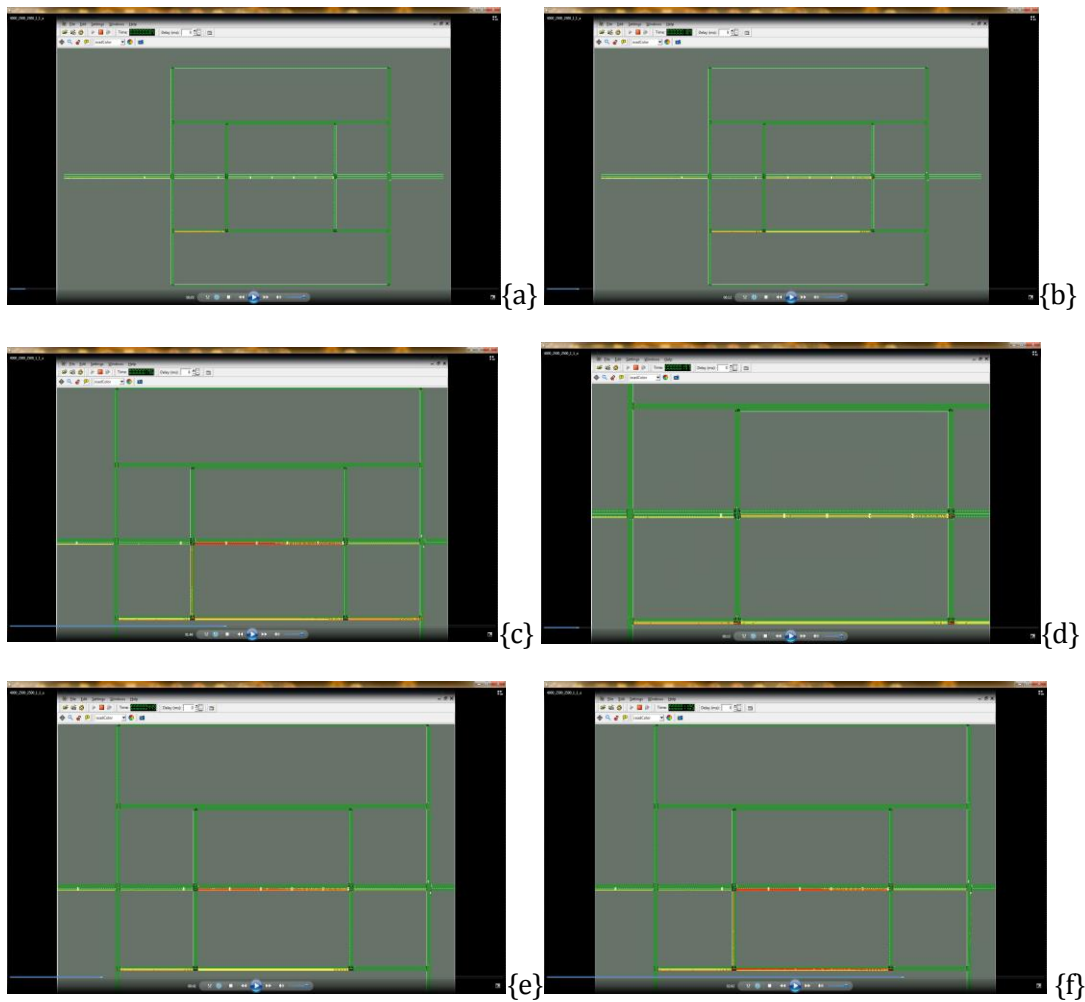


Figure 6-13 a-f Simulation Capture of Hypothetical Road Network without using an ITS Architecture
Figure 6-13a illustrates no congestion. For each successive screen capture, the congestion becomes worse indicate by the red colored links in the final screen capture.

6.4.2 APPLYING ITS FOR DYNAMIC ROUTING – CENTRALIZED FOR HYPOTHETICAL ROAD NETWORK

When applying the cell 1 topology which is the fully centralized approach and discussed using the Algorithm in listing 5.1, the following histogram results as shown in Figure 6-14. Following the baseline greedy simulation, this simulation was set up using the framework topology discussed in cell-1. The loop sensors were used to provide information to a centralized decision algorithm which directed the vehicles to the alternate routes once the loops sensors recognized the speed of two linear adjacent detectors that indicated zero speed. This then directed the vehicles to the lower route indicated by the dashed line as indicated in Figure 6-6.

To lessen the impact on the lower route, delivery trucks were directed to a truck route, the uppermost route as its primary bypass, and are indicated by the dotted line. If the loop sensor just before the off-ramp detected zero velocity, then the bottom most route is selected for the cars before the loop sensor indicated by the dot-dashed line. If the impact to exit to the upper route and alternate lower routes still caused a sensed back-up at the left most loop sensor, then the route above the freeway, but below the truck route is used. Looking at the link, it is shown to have a reduced capacity by its linkages, and therefore that route is considered to have limited flow capabilities.

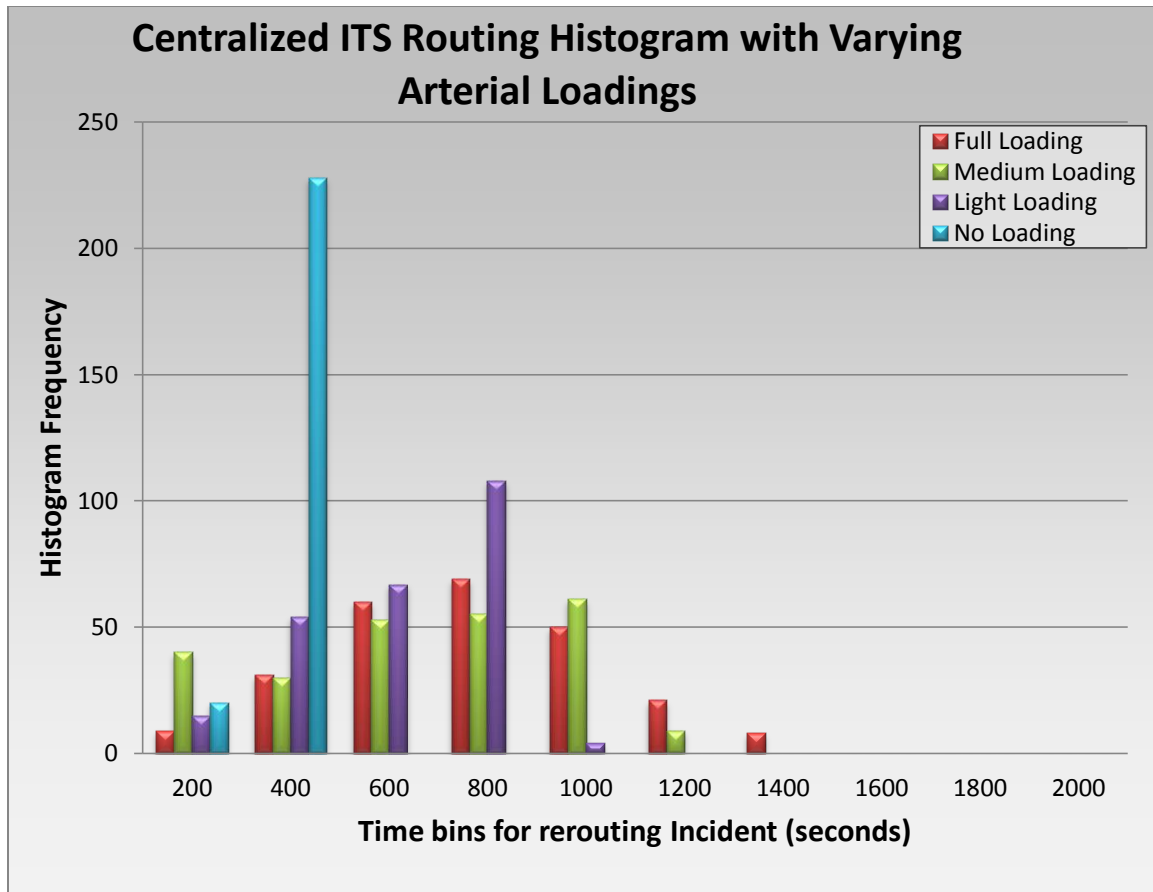


Figure 6-14. Histogram illustrating the times to travel from the origin prescribed destination using the centralized ITS Routing Algorithm

Using a centralized approach illustrates that overall travel time decreased as vehicles routed around the incident and were directed by a centralized process that captured data using loop sensors. When comparing this to the non-ITS scenario, it is evident that travel time around the incident improved shifting the time bins to the left. Table 6-1 presents a summary of average travel times. The longest time under full arterial loading was 1,267 seconds. Furthermore, the average for each of the 4 conditions was 215 seconds, 256 seconds, 339 seconds, and 313 seconds, for non-

arterial loading, light arterial loading, medium arterial loading, and full arterial loading respectively.

Shown in Figures 6-15a-f are the set of screen shots are from a screencast from the simulation using the hypothetical road network. This second screen cast is one that illustrates when there is a centralized ITS solution where the TMC notifies each vehicle its best route based upon the type of vehicle it is. It shows over time how the network is continues manages to improve the throughput of the network, however due to the decreased sensor fidelity and knowledge of arterial roads not know the TMC, the queues do grow as shown by the color change between screen captures.

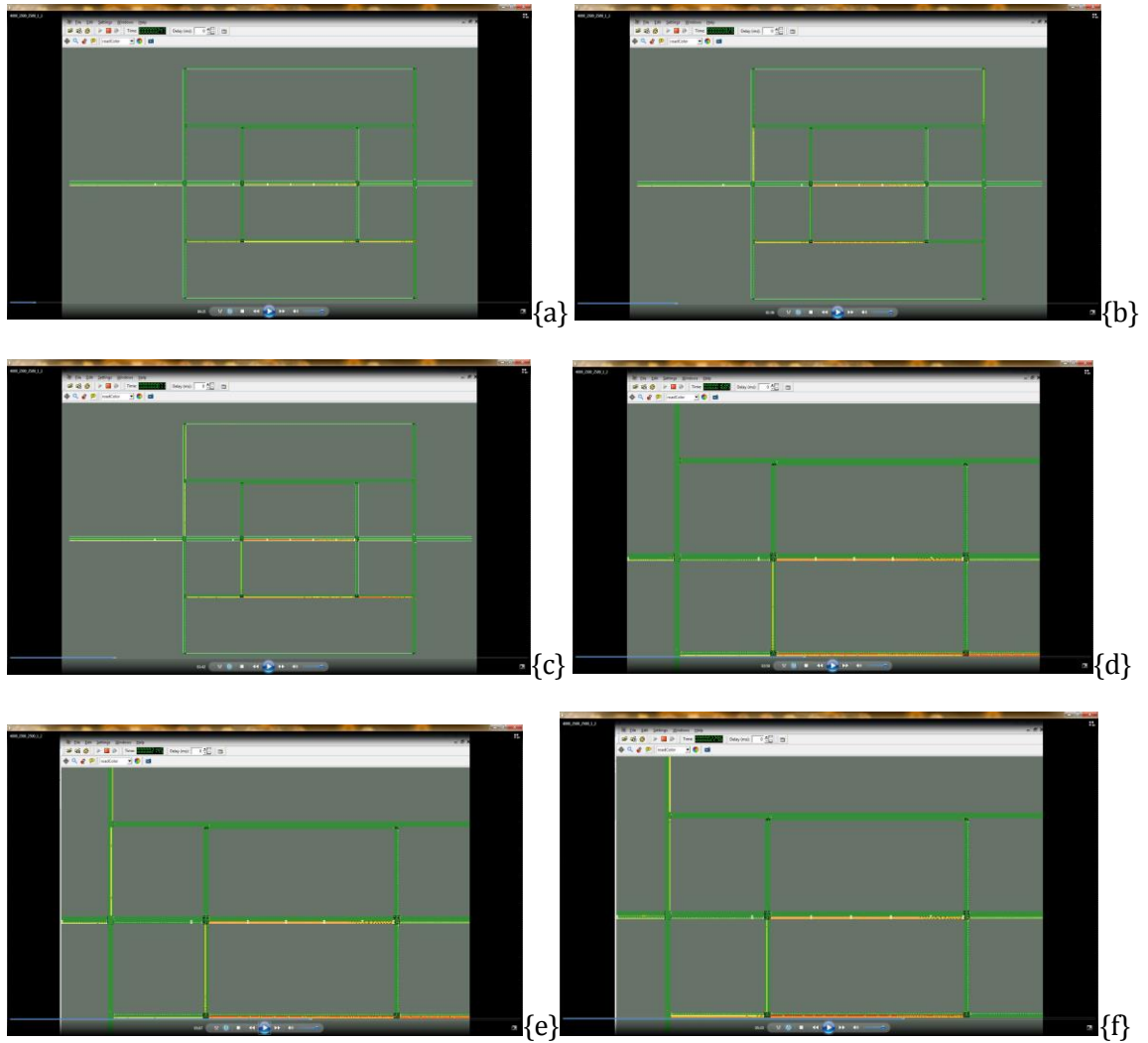


Figure 6-15 a-f Simulation Capture of Hypothetical Road Network using a Centralized ITS Architecture. Figure 6-15a illustrates no congestion. For each successive screen capture, the congestion becomes worse indicate by the red colored links in the final screen capture. Note that this screen captures a decreased amount of red link coloring vs Figure 6-13.

6.4.3 APPLYING ITS FOR DYNAMIC ROUTING - DECENTRALIZED

Comparing the non-ITS approach to the ITS approach illustrates the benefits of applying ITS methods to rerouting. When the Cell-9 Topology described in listing 5.2 is illustrated, the following data are generated and shown in Figure 6-16. The distributed ITS framework incorporates a distributed environment approach. In this simulation, each vehicle is able to communicate with any vehicle that is within range of some direct short range communication (DSRC) type radio. For this simulation, the range was set to be 500 meters such that any vehicle that is within 500 meters is considered to be a neighbor. During the simulation, each vehicle keeps track of its neighbors which can change depending on the route selection process. The vehicles communicated the current location, link segment and velocity to its neighbors. When a vehicle had ten neighbors that are at zero velocity, the vehicle would reroute based upon its current location, the available routes, and what type of vehicle it was. For this case as defined by the other ITS method Cell-1, the 1st alternative car route is the dashed route, followed by the dot-dash route. The 1st and only delivery truck route is the dotted route. Arterial route vehicles were not equipped with radios.

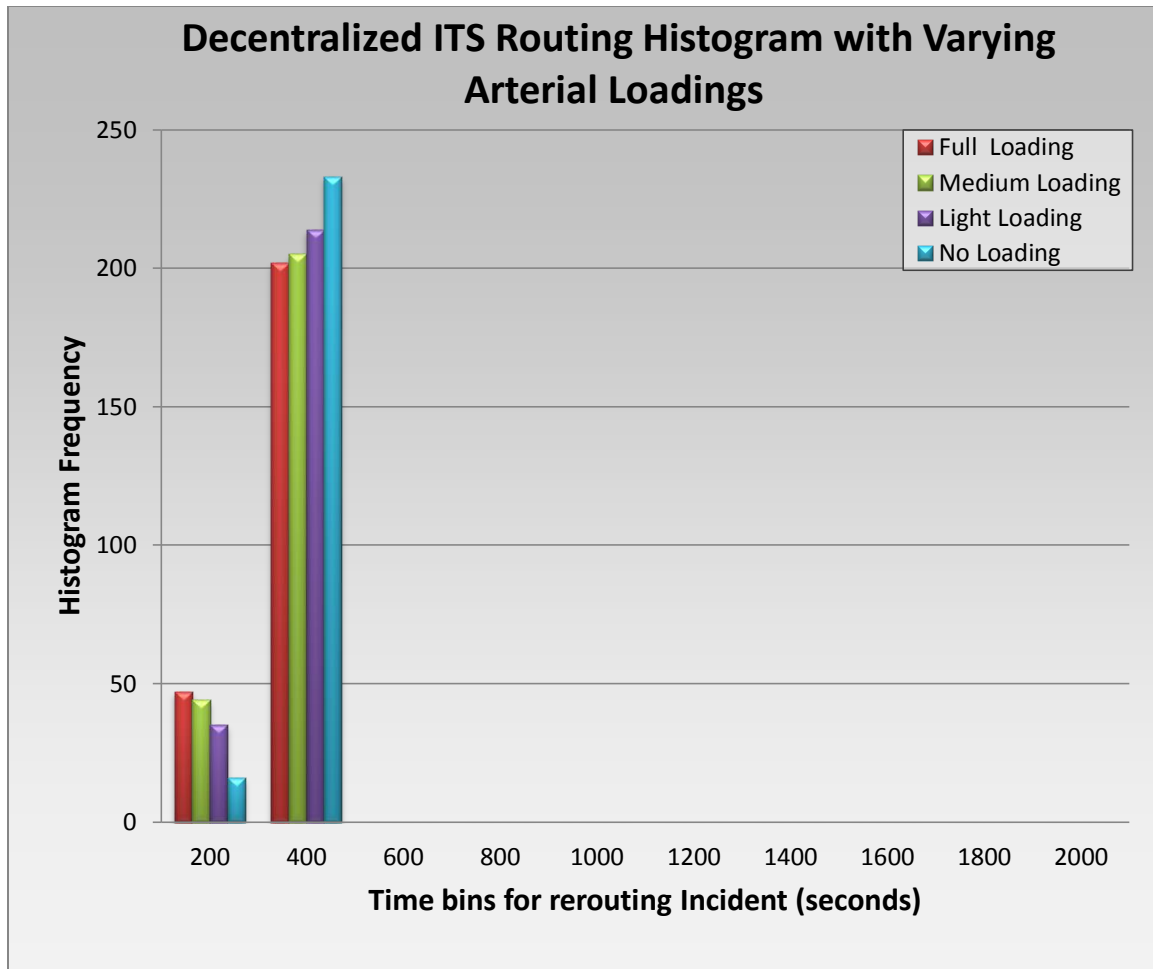


Figure 6-16. ITS Histogram illustrating the times to travel from the origin prescribed destination using the decentralized ITS Routing Algorithm.

Looking at the data from the fully decentralized approach, it is seen that the overall time to reroute is further improved over the centralized method. This is best illustrated looking at the average travel times for each approach as illustrated in Table 6-1, and Figure 6-19. In table 6-1 the distributed method regardless of arterial loading has the highest of occurrence of fastest travel times versus the other methods. One thing to note is that if there is not enough density to support

sufficient communication, as in the no-loading simulation, the vehicles exhibited longer rerouting times versus those simulations where there was sufficient density to support inter-vehicle communication.

The final set of screen shots are from a screencast from the simulation using the hypothetical road network. This last screen cast is one that illustrates when there is a distributed architecture ITS solution. Each vehicle is able to communicate with its neighbors on the freeway as well as the arterial roads. The simulation was done here using the FRP coding style. In these screen shots, the network is easily able to manage the queues as indicated over time of the simulation the color only becomes a medium yellow.

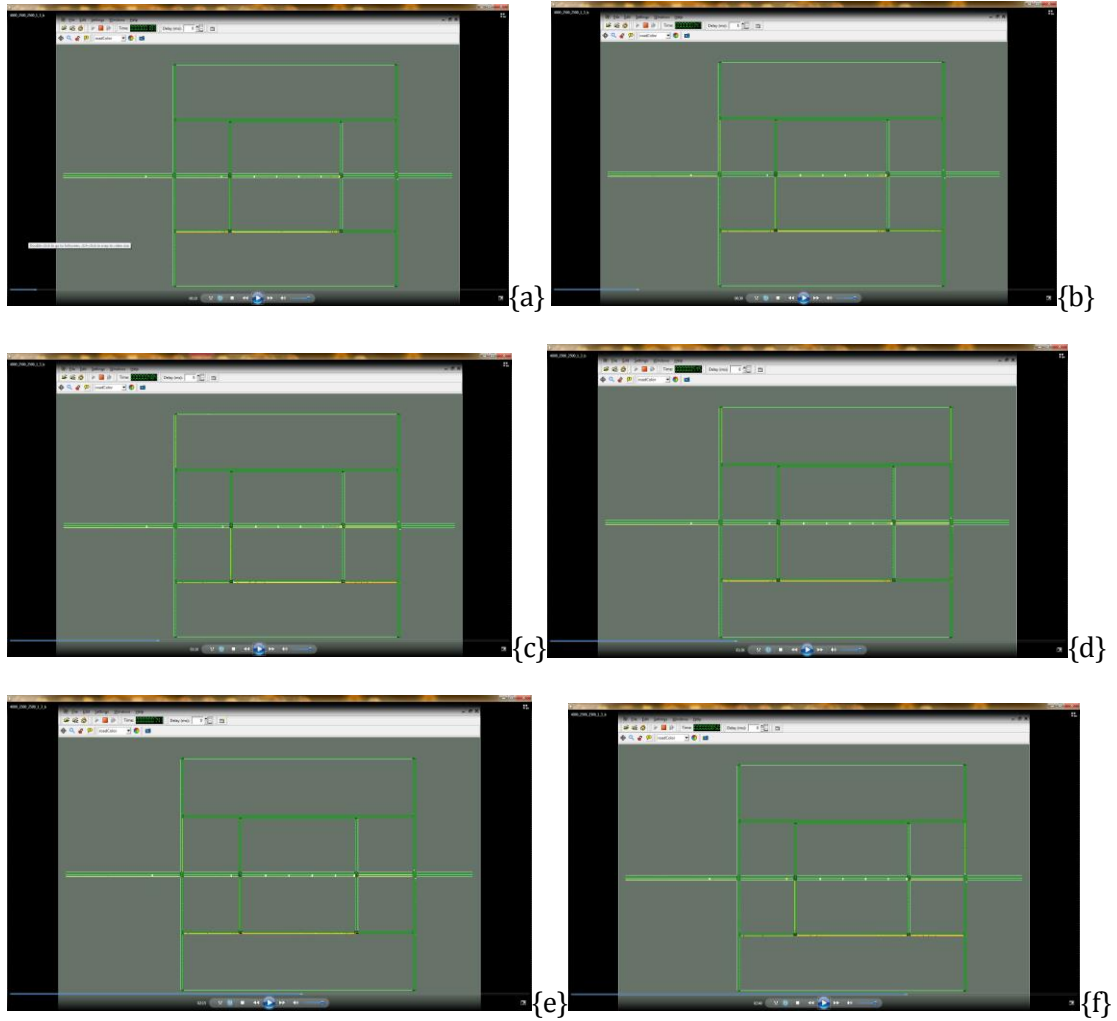


Figure 6-17 a-f Simulation Capture of Hypothetical Road Network using a Decentralized ITS Architecture. Figure 6-17a illustrates no congestion. For each successive screen capture, the congestion becomes only moderately increased as indicated by links as only medium yellow.

6.4.4 APPLYING MIXED CENTRALIZED AND DECENTRALIZED BASED RESULTS FOR A HYPOTHETICAL ROAD NETWORK.

This approach uses both architecture approaches of the two previous sections. It illustrates that while it performs better than both the non-ITS and centralized architecture. When 50% of the vehicles are able to communicate with each other, the overall travel time is reduce as those vehicles are able to communicate with one another about what routes take. The travel time histogram is shown in Figure 6-18.

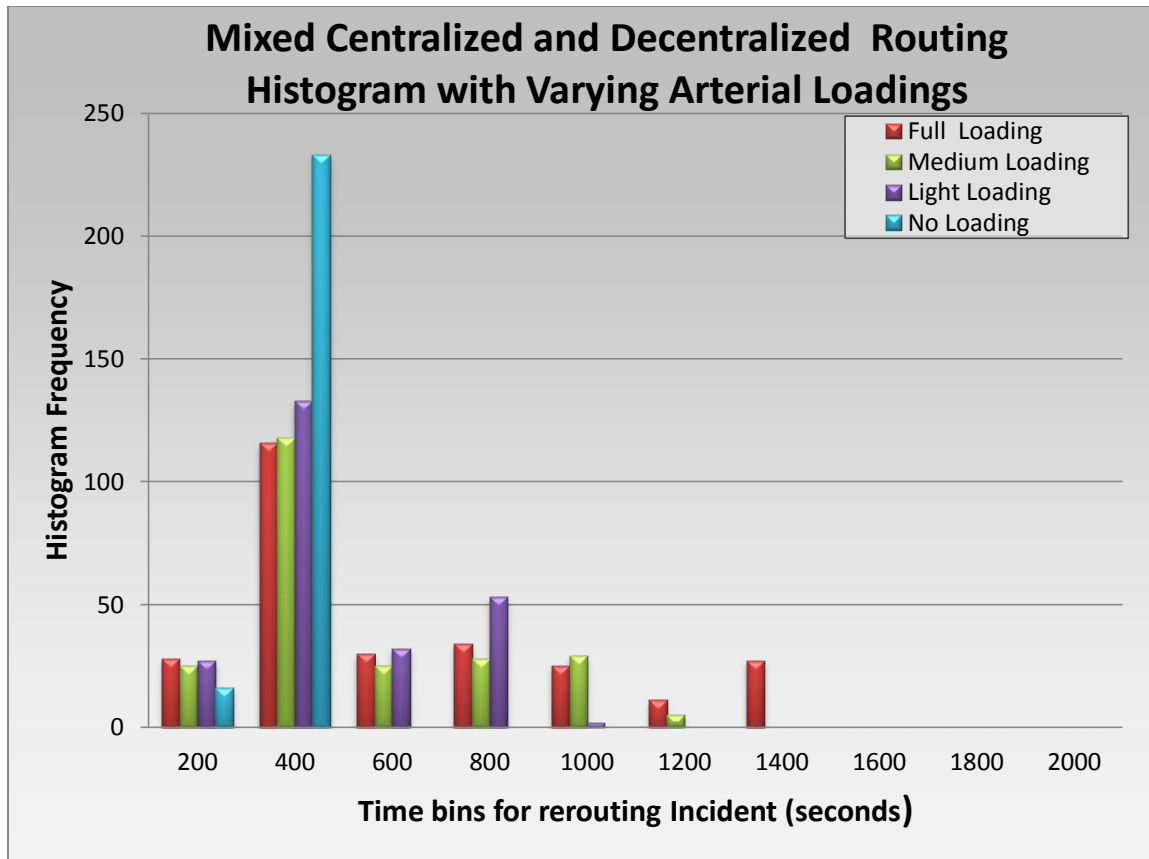


Figure 6-18. ITS Histogram illustrating the times to travel from the origin prescribed destination using the combined centralized and decentralized ITS Routing Algorithm.

Table 6-1 shows a summary comparison of the route times for a given architectures and is plotted in the average time of the ITS techniques to the non-ITS technique is shown below in Table 6-1 and Figure 6-19.

Table 6-1. Average Re-route time vs Technique. FA, MA, LA, NA indicates a fully loaded arterial, medium loaded arterial, lightly loaded arterial, and non-loaded arterial respectively for the hypothetical road network.

Hypothetical Network Average Route Times				
Technique	FA	MA	LA	NA
Distributed ITS	217.45	218.36	220.66	223.66
Centralized ITS	655.77	665.29	528.87	223.68
Centralized Decentralized ITS	439.52	411.79	373.45	223.64
Greedy - No ITS	691.89	883.91	654.72	430.78

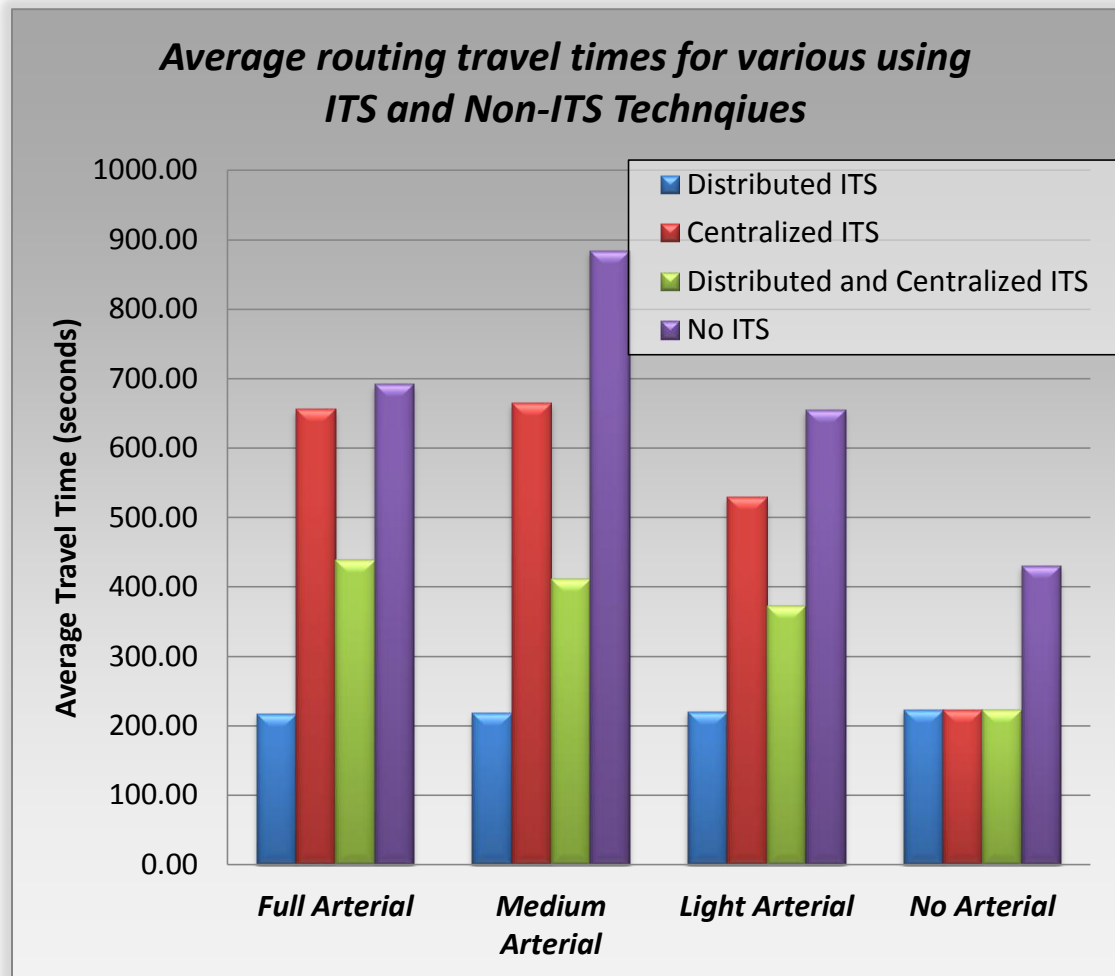


Figure 6-19 Average Travel Times for rerouting with various loading. The horizontal axis groups the travel times by arterial loading. The Vertical axis is the average travel time for type of architecture.

In the hypothetical road network the distributed architecture represented in cell-9 of the matrix performed the best. It provided a higher fidelity of sensing and was able to enable each vehicle type the opportunity to take a route more conducive when the congestion began to occur. The combined decentralized / centralized

architecture, cell-5 performed next and is representative of an ITS systems where the infrastructure provides data to those vehicles that do not participate in sharing data with other vehicles. This is then followed by the centralized ITS architecture. This architecture provided decreased travel time through the network in the event of an incident, though did not have the fidelity of the sensors to reroute as well. Finally, the non-directed ITS approach, represented in cell-7 performed the worst. This is where the vehicles become aware of congestion and take the next route based an A* routing algorithm without knowledge of the vehicles surrounding them.

6.5 SIMULATIONS FOR EXAMPLE NETWORKS

The second simulation performed was using data to synthesize a simulation based upon the vehicle volume and rate of an actual route. This is the route along CA SR-60 between Ramona, and Euclid.

6.5.1 CA SR-60 NON-ITS BASED RESULTS

As a baseline, the condition where no ITS methods exist was simulated. This is the condition where no knowledge of the state of the network is known and only the best route that has the shortest deviation around the incident was initially examined. As discussed earlier, this is used in greedy routing. During this simulation, when the cars queued up to where the existing off-ramp, they exited and reroute to the arterial roads. Furthermore, each arterial intersection is a controlled intersection with a traffic light. The overall throughput of the simulation was measured as is shown in Figure 6-20.

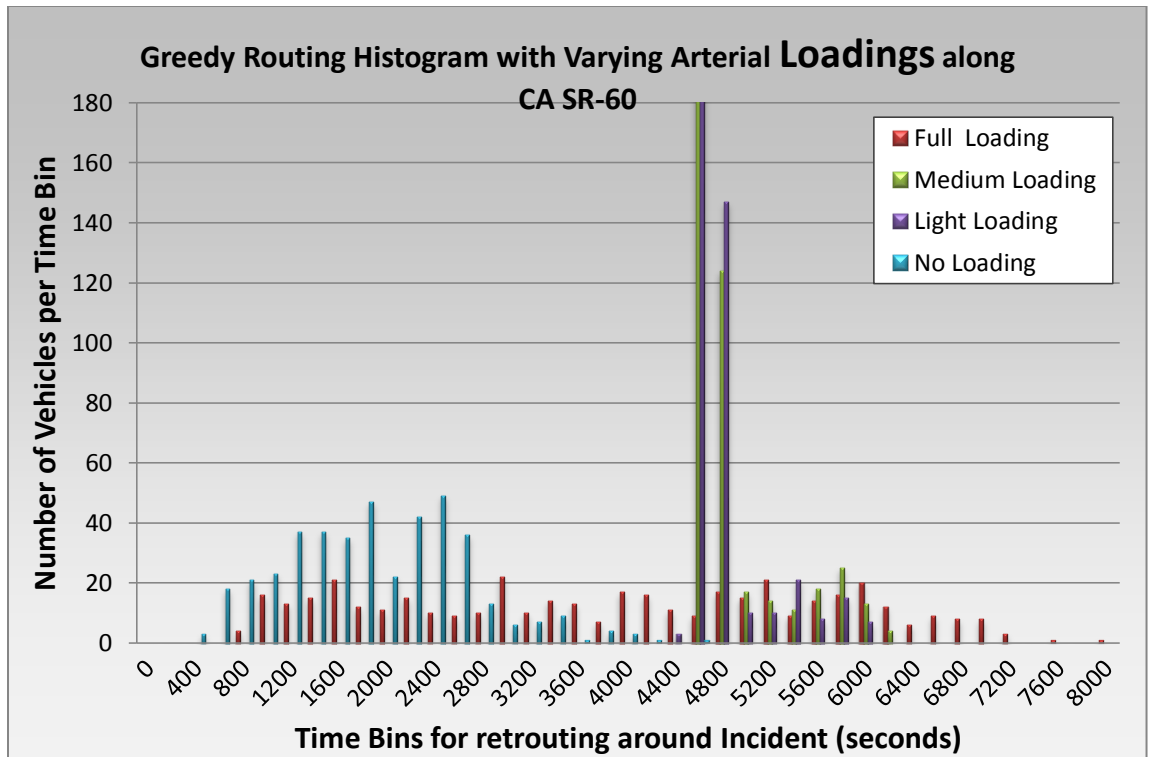
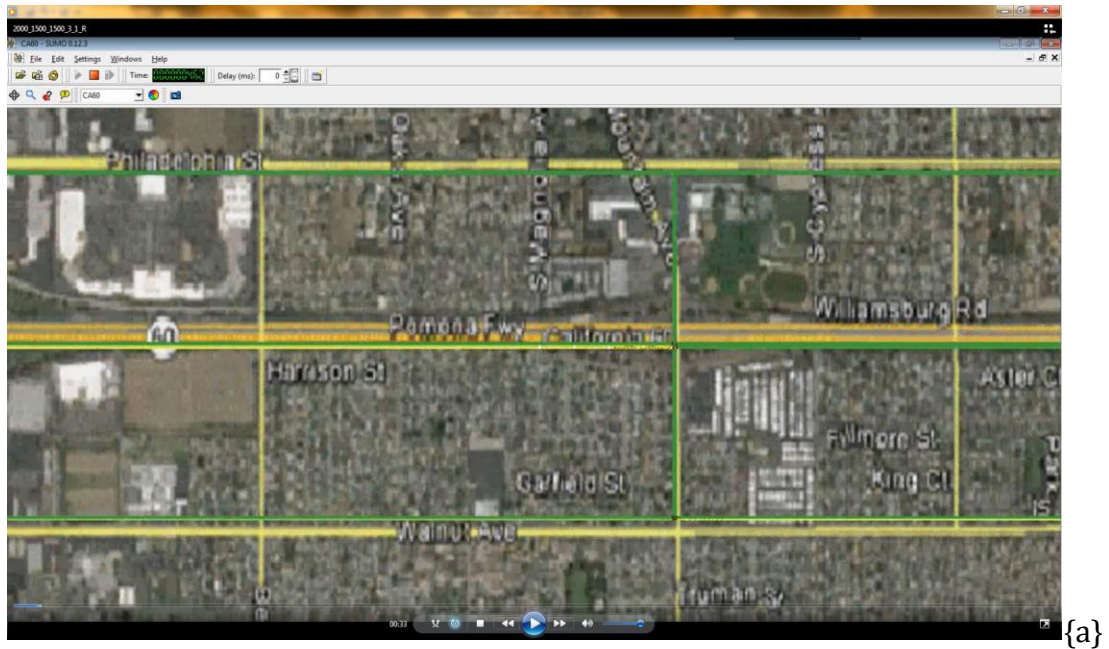


Figure 6-20 Average Travel Times without route information methods for rerouting with various loading.

Using this approach, the simulation illustrates in the histograms how there was full loading of the arterial the time to route around the incident includes the longest times, while when there is no loading, the time to route around the contained the faster times. When the arterial is loaded ad a light to medium level, it exhibits a strong clustering near the midpoint of the fully loaded simulation, however it is able to accommodate the loads, and therefore these simulations do not have travel times to the right of the graph.

Shown in Figures 6-21 a-c are the screen shots of the simulation at the beginning and after the congestions occurs.



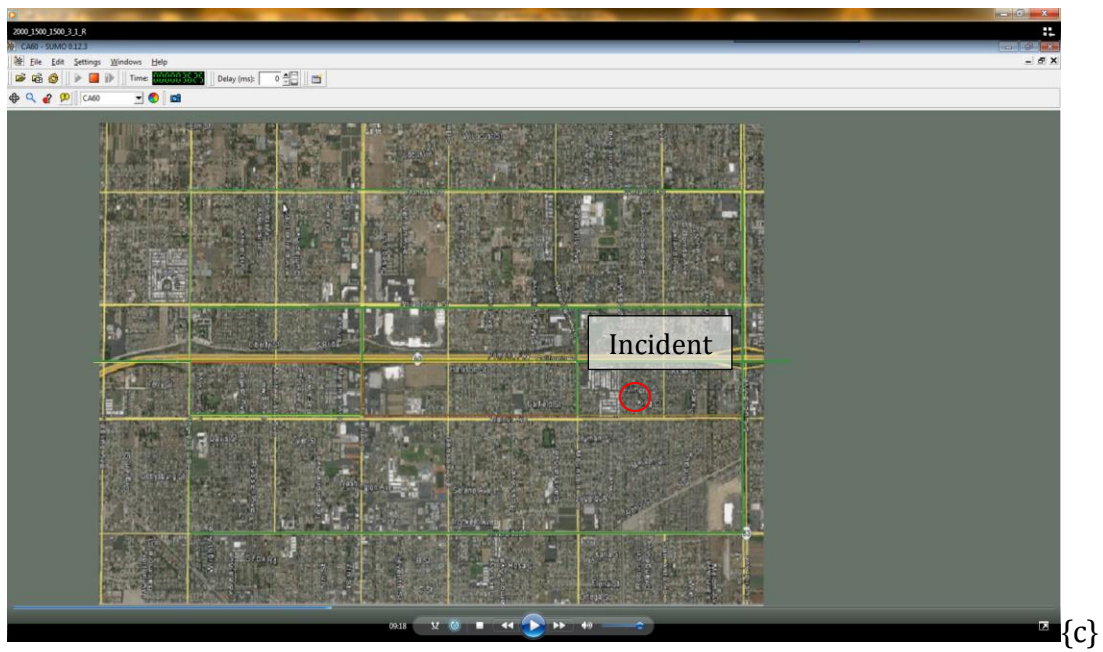
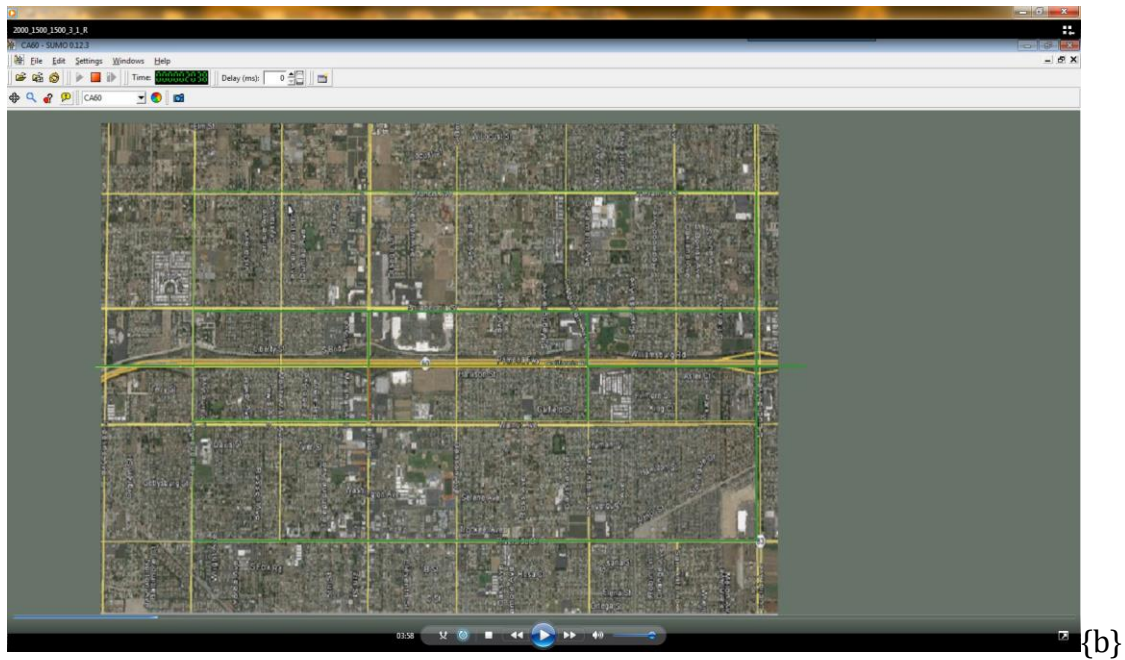


Figure 6-21. a-c Simulation Capture of Hypothetical Road Network without using a ITS Architecture. The incident is indicated by the red circle.

Similar to the hypothetical network, the CA SR-60 network over time continues to increase in volume of vehicles for a given link as the links change color from green to red on the A* routes, while the other links in the network remain green.

6.5.2 CA SR-60 CENTRALIZED-ITS BASED RESULTS

When applying the cell 1 topology which is the fully centralized approach and discussed using the Algorithm in listing 5.1 the following histogram results as shown in Figure 6-22. This exhibits similar results to that of the hypothetical road network. Following the baseline greedy simulation, this simulation was set up using the framework topology discussed in cell-1. The loop sensors were used to provide information to a centralized decision algorithm which directed the vehicles to the alternate routes once the loops sensors recognized the speed of 2 linear adjacent detectors that indicated zero speed. This then directed the vehicles to the lower route along Walnut Ave. shown in Figures 6-7 and 6-8.

To lessen the impact on the lower route, delivery trucks were directed to truck route, the uppermost route to Francis as a primary bypass. If the loop sensor just before the off-ramp detected zero velocity, then the bottom most route is selected for the cars before the loop sensor. If the impact to exit to the upper route and alternate lower routes still caused a sensed back-up at the left most loop sensor,

then the route above the freeway, the truck route Riverside Ave. is used. Looking at the link, it is shown to have a reduced capacity by its linkages, and therefore that route is considered to have limited flow capabilities.

Shown in Figure 6-22 is the histogram for the centralized architecture. As expected from the hypothetical simulation, the when using knowledge of the road condition the centralized architecture is able to reduce the overall travel time even though there is an increase in the arterial loading. However due to lack of knowledge of the arterial loading, the heavier loading on the arterial causes an increase travel times.

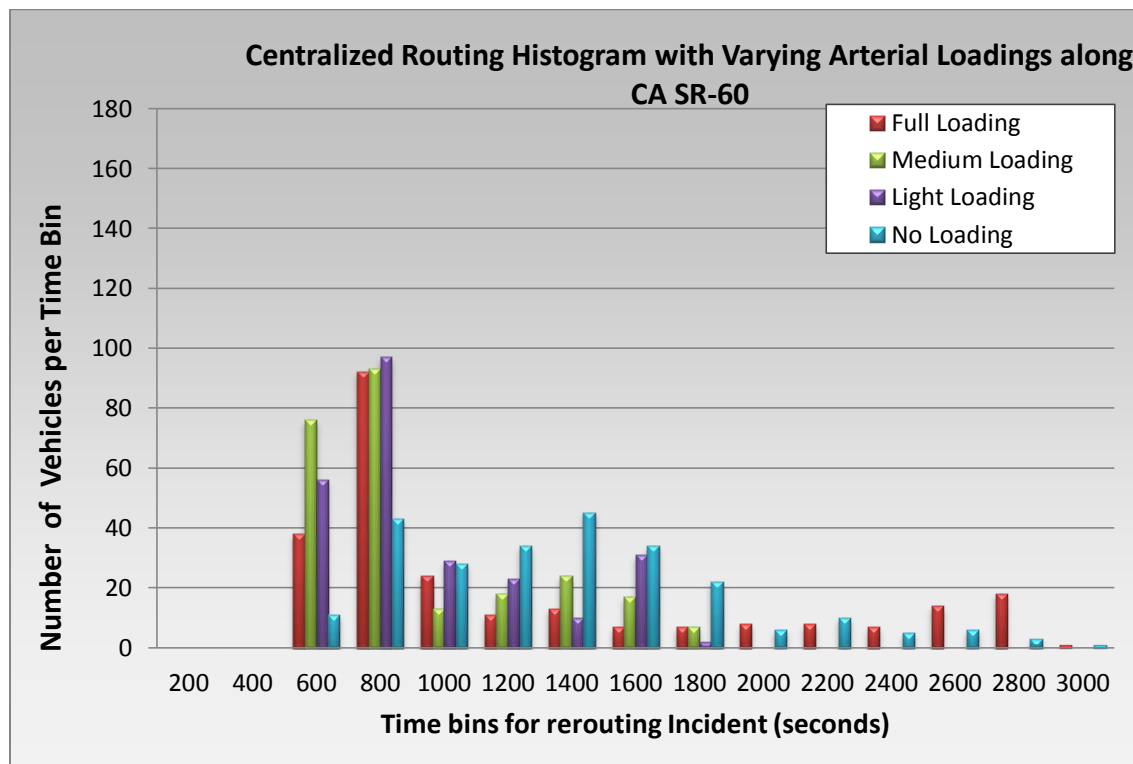
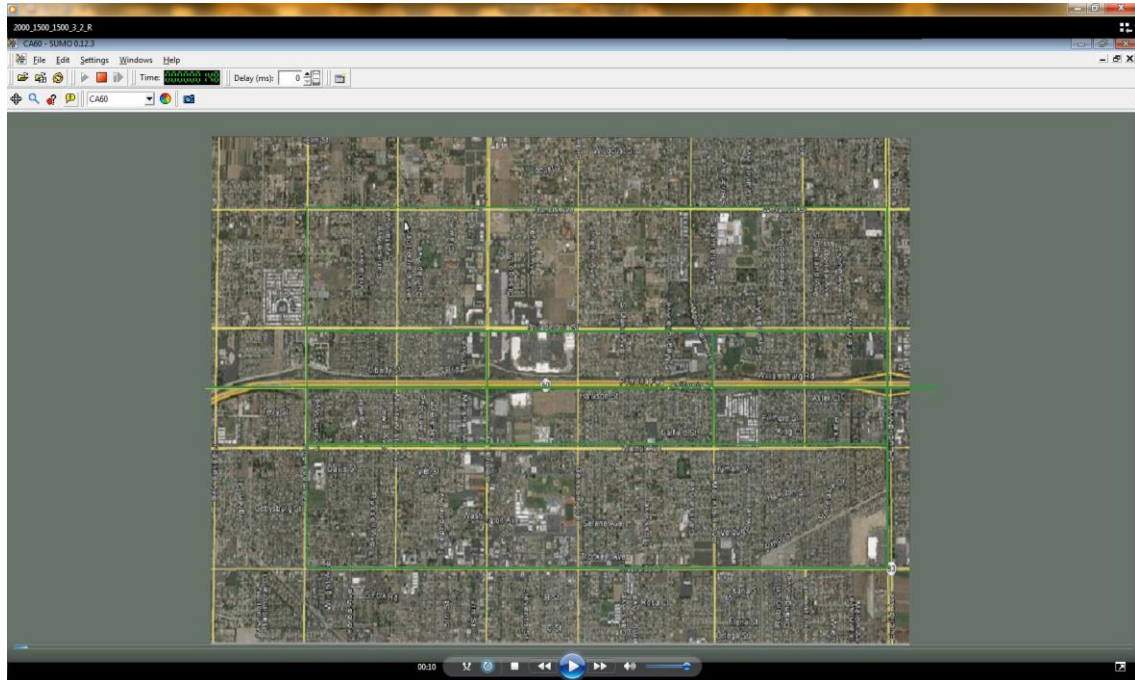
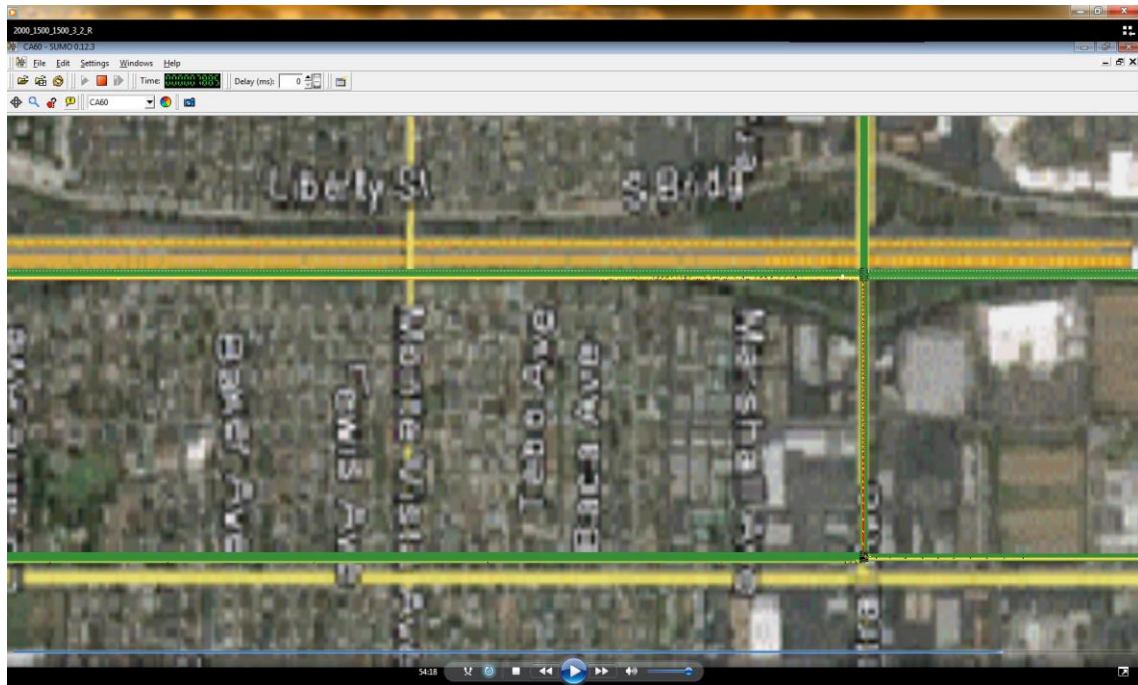


Figure 6-22 Average Travel Times with centralized route information methods for rerouting with various loading

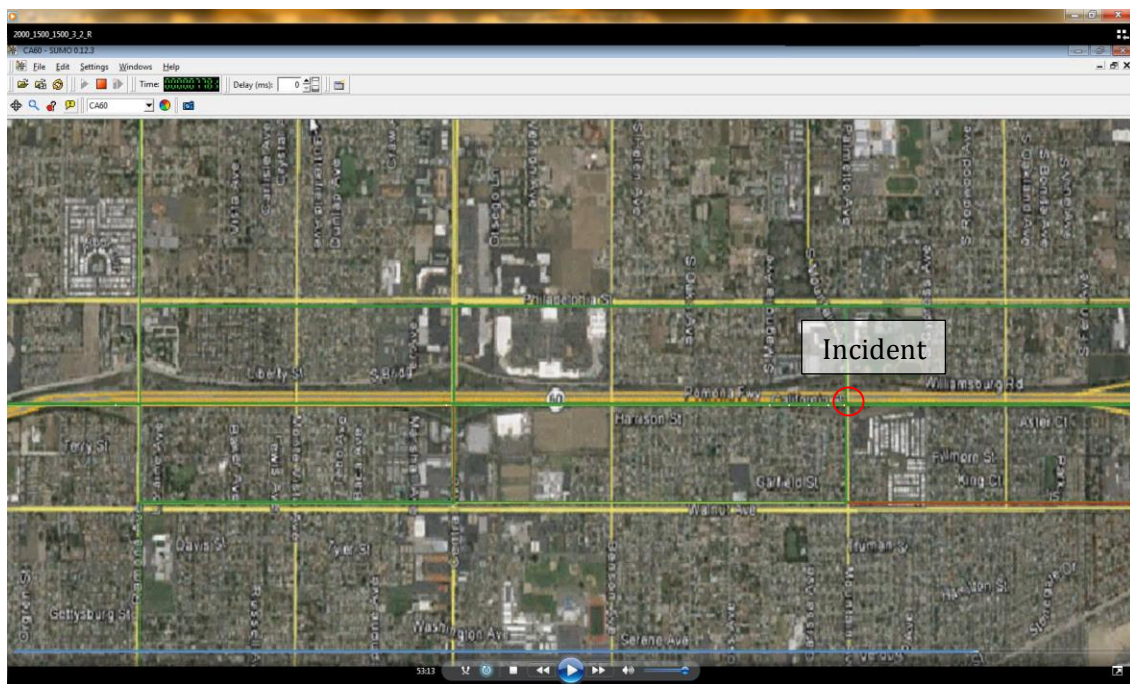
The simulation outputs for this are shown in Figures 6-23a-c. In this simulation run, the links remain green longer and only become until finally red.



{a}



{b}



{c}

Figure 6-23 a-c Simulation Capture of Hypothetical Road Network using a Centralized ITS Architecture. The location of the incident is indicated by the red circle.

6.5.3 CA SR-60 DECENTRALIZED-ITS BASED RESULTS

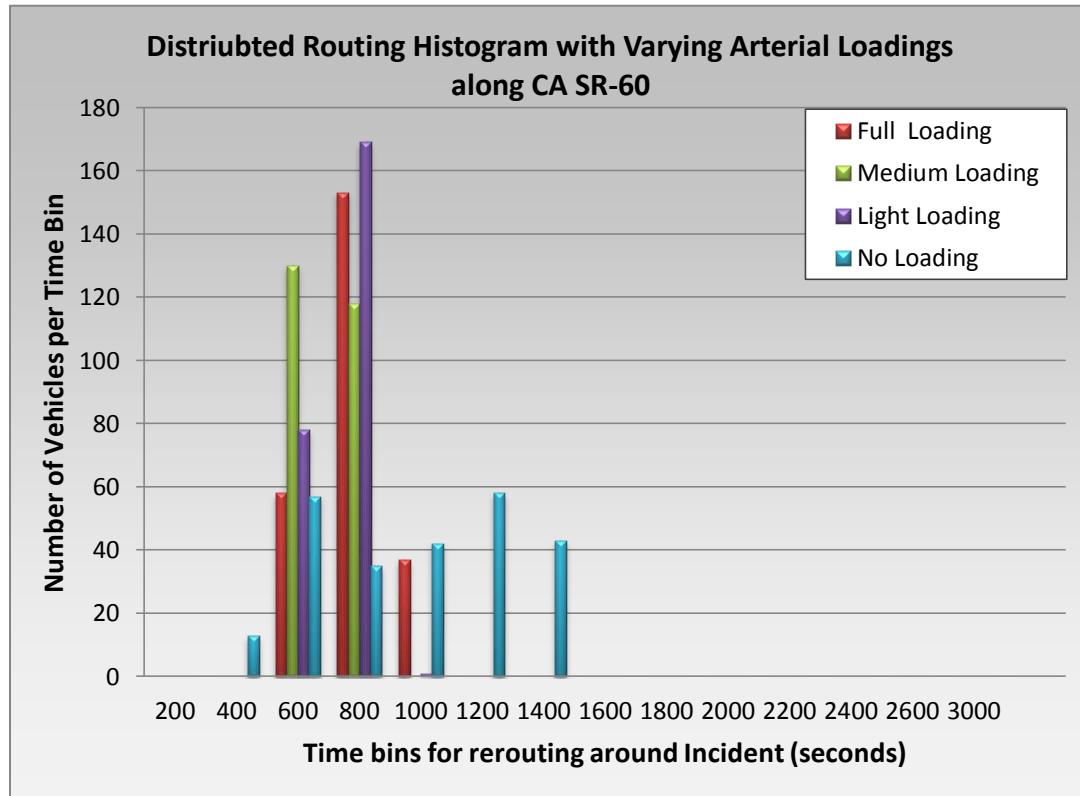
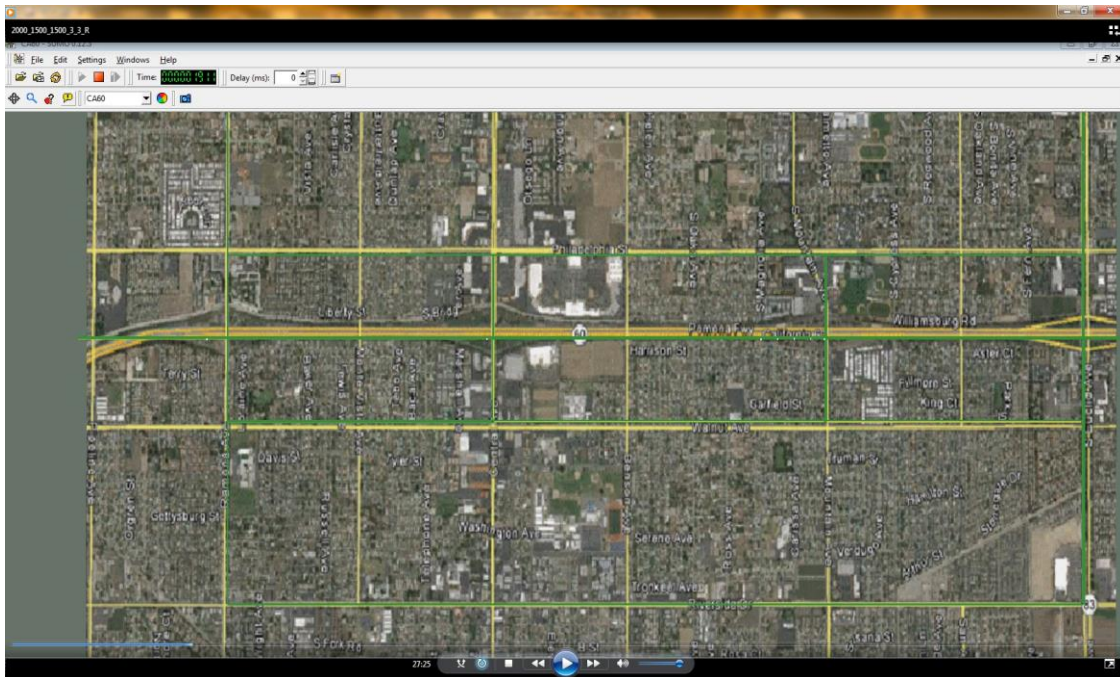


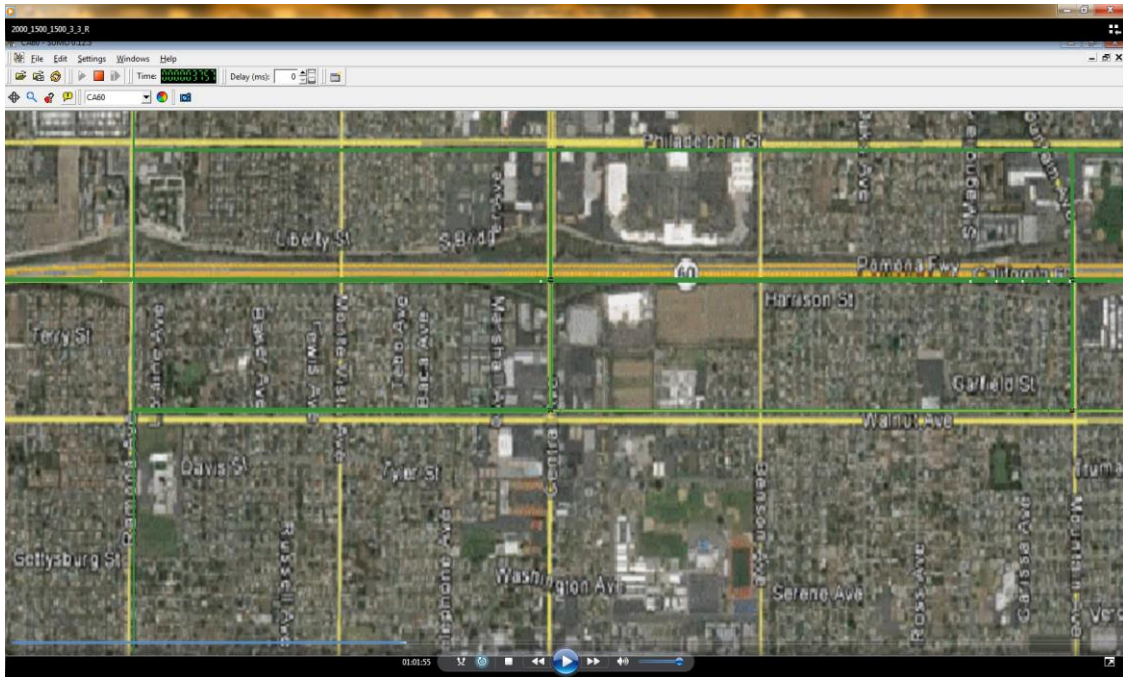
Figure 6-24 Average Travel Times with Decentralized route information methods for rerouting with various loading

These results of the CA SR-60 corridor are similar to the hypothetical results obtained. When the vehicles are within range of one another they are able to take alternative routes thus decreasing the overall congestions to the arterial networks. In this example the cars are able to reroute to Walnut Ave, while the trucks reroute to Francis St. Riverside becomes a secondary route a needed for the trucks. The

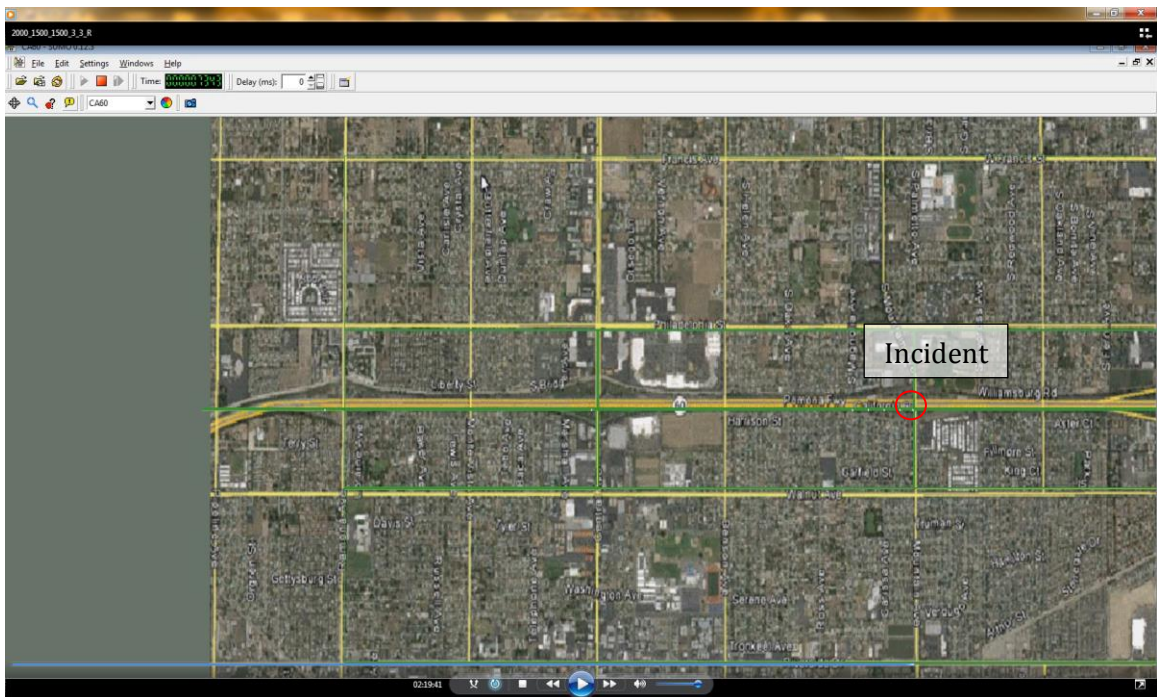
screen captures for this simulation are shown in Figure 6-25 a-c. Note that during this screen capture the road network loading is either green or a medium yellow.



{a}



{b}



{c}

Figure 6-25 a-c Simulation Capture of Hypothetical Road Network using a Centralized ITS Architecture. The location of the incident is indicated by the red circle.

6.5.4 CA SR-60 MIXED CENTRALIZED AND DECENTRALIZED ITS BASED RESULTS

Shown in Figure 6-26 are the results for the simulation that includes both the centralized and decentralized was performed. This described using cell-5 in the matrix. The results for this simulation show that it does better than the centralized by allowing the vehicles to communication with one another. What is interesting about this histogram is that when comparing this to the hypothetical, when the spacing between the vehicles can become sparser, then the range of the radio set to 1000 meters can become a potential issue. The mix of vehicles that use the vehicle to vehicle communication is 50% to that which uses the centralized information.

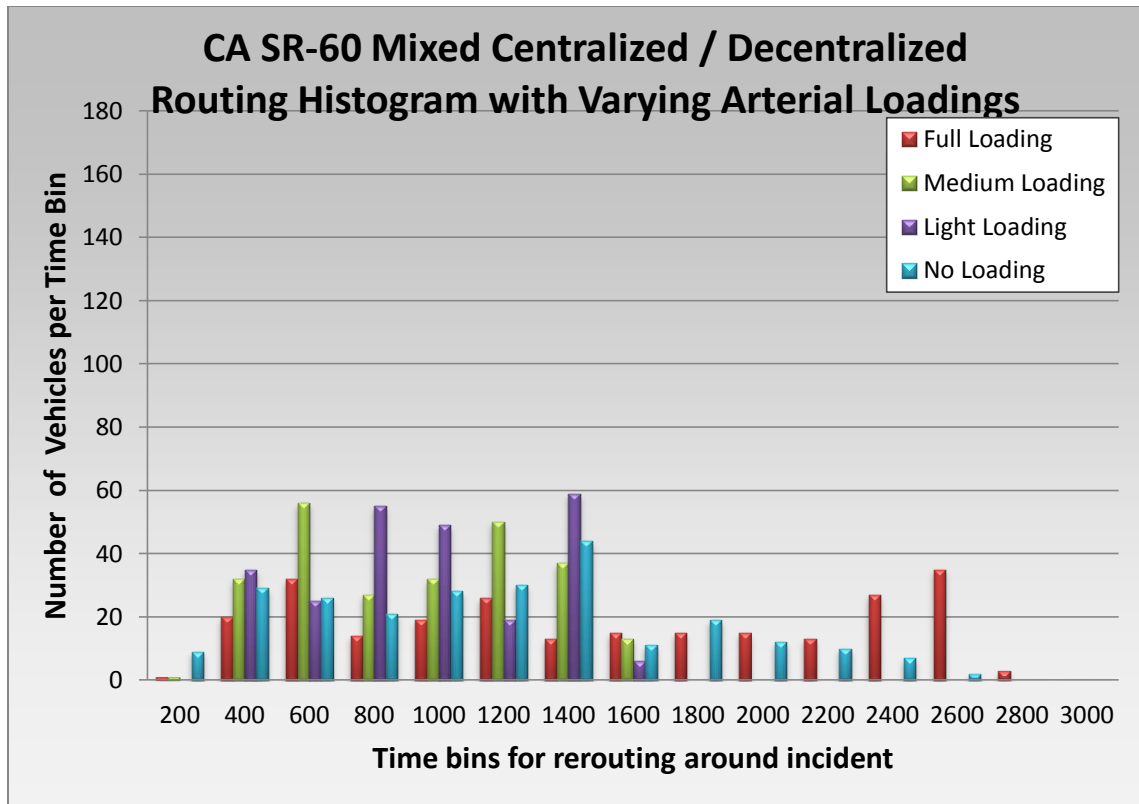


Figure 6-26 Average Travel Times with Centralized and Decentralized route information methods for rerouting with various loading

6.5.5 CA SR-60 ITS BASED RESULTS

To summarize the simulation performed using the CA SR-60 corridor routes, the distributed performed the best providing the quickest route times regardless of the loading effects of the arterial roads. When only 50% of the vehicles are able to communicate with one another, the combined centralized and decentralized performs second best. The next best performing approach is the centralized architecture and finally when the vehicles make their own decisions based solely

upon their own routing without knowledge of the surrounding state of roads, this performs the worst. Figure 6-27 illustrates the histogram and Table 6-2 summarizes the results.

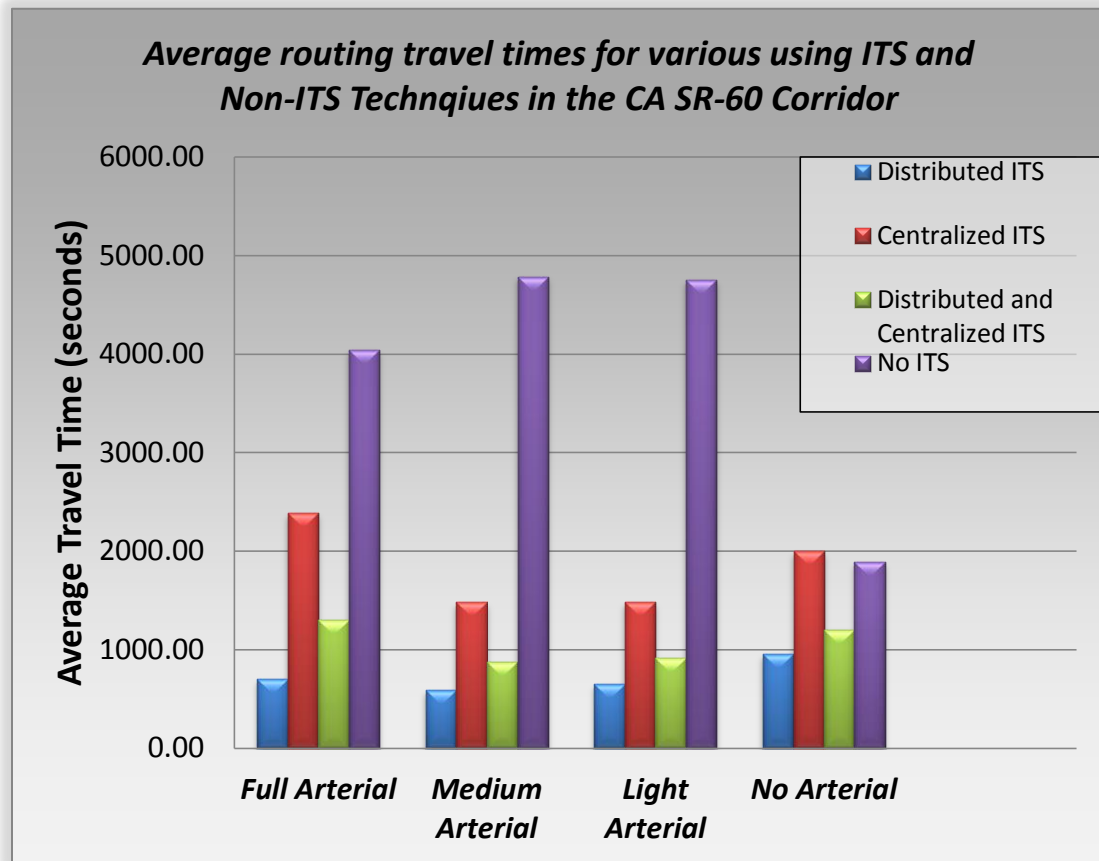


Figure 6-27 Comparing the average travel times through the CA SR-60 corridor

Table 6-2 Average Re-route time vs Technique. FA, MA, LA, NA indicates a fully loaded arterial, medium loaded arterial, lightly loaded arterial, and non-loaded arterial respectively for the CA SR-60 road network.

CA SR-60 Average Route Times				
Technique	FA	MA	LA	NA
Distributed ITS	705.30	586.34	654.36	954.18
Centralized ITS	2384.43	1477.15	1485.03	1998.28
Centralized Decentralized ITS	1302.39	874.53	915.60	1203.26
Greedy - No ITS	4041.29	4786.64	4750.76	1891.90

In summary, a comparison between the hypothetical and CA SR-60 corridor simulations was performed. This analysis was used to ensure that the data captured between both networks for the different possibilities were consistent. Table 6-3 shows the average times for each loading. These are full arterial loading (FA), medium arterial loading (MA), lightly arterial loading (LA), and no arterial loading (NA). The average travel times for each simulation result from the simulation were divided by its corresponding CA SR-60 average travel time. The numbers illustrated in the table indicate that there is a consistency between the simulations. Typically the ratio between the two was around 0.3 except for the simulations without arterial loading, though within themselves, these numbers were consistent.

Table 6-3 Comparing the Hypothetical Road Network with the CA SR-60 Road Network. FA, MA, LA, NA indicates a fully loaded arterial, medium loaded arterial, lightly loaded arterial, and non-loaded arterial respectively.

Hypothetical Network vs Real Network (Hypothetical Network / CA SR-60 Corridor)					
Technique	FA	MA	LA	NA	Avg by Loading
Distributed ITS	0.3083	0.3724	0.3372	0.2344	0.3131
Centralized ITS	0.2750	0.4504	0.3561	0.1119	0.2984
Centralized Decentralized ITS	0.3375	0.4709	0.4079	0.1859	0.3505
Greedy - No ITS	0.1712	0.1847	0.1378	0.2277	0.1803
Average by Type	0.2730	0.3696	0.3098	0.1900	0.2856

Chapter 7 CONCLUSIONS AND FUTURE WORK

Due to the rapid development in technology, ITS technology is able to adapt new techniques in vehicle speed data acquisition and distribution to provide enhanced road network awareness to the drivers. In doing so, this provides them with an ability to improve the process of making decision about the best routes to take. This dissertation has provided two approaches of ITS where one approach uses a centralized technique for data acquisition and processing, as well as an alternative technique where the processing is performed within each vehicle as well as capturing the network data via the individual vehicles. From this the driver is able to select alternative routes that are the most conducive to him as opposed to simply using a device that provides him with the best data without regard to what routes other vehicles are following. By allowing the driver to make "*a more informed decision, drivers* are able to have reduced impact of routing delays when incidents occur.

When investigating the differences between distributed and centralized solutions, the centralized solution carries with it a large infrastructure that requires updating and maintenance. Furthermore, agency responsibility of who provides what data to whom remain in question. However, using a distributed solution alleviates these challenges, but it is noted this technique carries with it its own challenges. Initially

all participants must be willing to participate by providing data to their neighbors, and implement the desired solutions calculated by the vehicle. Also, there needs to be a sufficient vehicle density of vehicles to distribute the data among one another order. One could argue that a combination of a centralized and decentralized could be implemented which would overcome the shortcomings of the distributed method, but at the cost of maintain some type of infrastructure.

Furthermore, implementation of a distributed approach without cooperation of the other vehicles is potentially problematic. The algorithms within each vehicle must be able to address the behaviors of the other vehicles by means on understanding the intended behaviors and routes of the particular vehicles. As such a framework was provided that address how vehicles should interact with one another while addressing a software implementation that best fits the type of asynchronous data streams that each vehicle will experience. By use of Functional Reactive Programming techniques, issues of asynchronous data streams in real time are alleviated while also providing a structure by which programs which much otherwise follow an observer pattern are alleviated from potentially code development issues.

Future work involves further developing the notion of vehicle interaction by means of cooperability. This would incorporate a better understanding of the cost functions involved and their impact on how the cooperability occurs. The idea of agent interaction becomes vital to understand the scope of a multi-agent type distributed solution, however due to the amount of load placed upon a centralized solution, the cooperable distributed solution is more palatable. Also the approach of FRP is still rather new, and some are skeptical about overall claims of usefulness when compared to traditional imperative and object oriented approaches. Traditionally new approaches should be vetted to ensure success, however without first proposing them, the new approaches would never be worked out. Other areas of future work would involve performing a similar body of work, however capture a different application of vehicle interaction such as intersection management, or advanced collision avoidance.

Finally, the overall work and analysis was performed using a simulation environment. Although the simulation provides great insight to the validity of the work, the work presented here should now be applied to actual vehicles that perform their own processing using the algorithms outlined. Initially this could involve several dozen small robotic vehicles scaled down in size to fit in a lab environment. In place of a GPS system, a camera would provide and report position tracking. Other various sensors would be used to replace loop sensors. After

completing this work, the system should be validated using a large sample of real vehicles on freeway systems over the course of many months to confirm the findings presented here.

Chapter 8 BIBLIOGRAPHY

- [1] T. Choe, A. Skabardonis, P. Varaiya, "Freeway Performance Measurement System (PeMS): An Operational Analysis Tool," in *Proc. of the 81st Transportation Research Board Annual Meeting*, Washington D.C., January 2002.
- [2] Rafiq, G.; Talha, B.; Patzold, M.; Gato Luis, J.; Ripa, G.; Carreras, I.; Coviello, C.; Marzorati, S.; Perez Rodriguez, G.; Herrero, G.; Desaegeer, M., "What's New in Intelligent Transportation Systems?: An Overview of European Projects and Initiatives," *Vehicular Technology Magazine, IEEE*, vol.8, no.4, pp.45,69, Dec. 2013
- [3] S. Shladover, "Potential Contributions of Intelligent Vehicle/Highway Systems (IVHS) to Reducing Transportation's Greenhouse Gas Production," *California PATH TECH-MEMO*, 1991.
- [4] <http://www.iteris.com/itsarch/html/mp/mpatms08.htm>, retrieved 6/2014
- [5] H. Bast, S. Funke, D. Matijevic, P. Sanders, D. Schultes, "In Transit to Constant Time Shortest-Path Queries in Road Networks", *Alenex07 Workshop on Algorithm Engineering & Experiments*, New Orleans, LA, pp. 46-59, Jan. 6th, 2007
- [6] Jen-Wen Ding; Fa-Hung Meng; Yueh-Min Huang, "A real-time vehicle guidance system using P2P communication," *Ubi-Media Computing, 2008 First IEEE International Conference on*, vol., no., pp.225,230, July 31 2008-Aug. 1 2008
- [7] <http://www.inrix.com/>
- [8] <http://www.sigalert.com/Map.asp>
- [9] Transportation Management Center Concepts of Operation Implementation Guide. 1999 http://ntl.bts.gov/lib/jpodocs/rept_mis/11494.pdf retrieved 2014
- [10] van Katwijk, R.T.; De Schutter, B.; Hellendoorn, J., "Multi-agent control of traffic networks: Algorithm and case study," *Intelligent Transportation Systems, 2009. ITSC '09. 12th International IEEE Conference on*, vol., no., pp.1,6, 4-7 Oct. 2009
- [11] van Katwijk, R, van Koningsbruggen, P, "Coordination of traffic management instruments using agent technology," *Transportation Research Part C*. vol 10, pp. 455-471, 2002.
- [12] U.S. Department of Transportation. Research and Innovative Technology Administration. Available online: <http://www.itsoverview.its.dot.gov/> (accessed on July 2014).

- [13] Naranjo, J.E.; Bouraoui, L.; Garcia, R.; Parent, M.; Sotelo, M.A., "Interoperable Control Architecture for Cybercars and Dual-Mode Cars," *Intelligent Transportation Systems, IEEE Transactions on* , vol.10, no.1, pp.146,154, March 2009
- [14] Vi Tran Ngoc Nha; Djahel, S.; Murphy, J., "A comparative study of vehicles' routing algorithms for route planning in smart cities," *Vehicular Traffic Management for Smart Cities (VTM), 2012 First International Workshop on* , vol., no., pp.1,6, 20-20 Nov. 2012
- [15] Zhan, F.B.; Noon, C.E. Shortest Path Algorithms: An Evaluation Using Real Road Networks. *Transportation Science* 1998, 32, 65–73.
- [16]. Cao, L.; Shi, Z.; Bao, P. Self-adaptive Length Genetic Algorithm for Urban Rerouting Problem. In *Advances in Natural Computation*; Jiao, L.; Wang, L.; Gao, X.b.; Liu, J.; Wu, F., Eds.; Springer Berlin Heidelberg, 2006; Vol. 4221, Lecture Notes in Computer Science, pp. 726–729.
- [17]. Dorigo, M.; Stützle, T. Ant Colony Optimization; MIT Press, Cambridge MA, 2004
- [18] Glover, F.; Laguna, M. Tabu Search. *Journal of computational biology a journal of computational molecular cell biology* 1997, 16, 1689–703.
- [19] Frazzoli, E.; Bullo, F., "Decentralized algorithms for vehicle routing in a stochastic time-varying environment," *Decision and Control, 2004. CDC. 43rd IEEE Conference on* , vol.4, no., pp.3357,3363 Vol.4, 14-17 Dec. 2004
- [20] Francesco Bullo, Jorge Cortés, Sonia Martínez, Distributed Control of Robotic Networks A Mathematical Approach to Motion Coordination Algorithms - Chapter 2: Geometric models and optimization, Princeton University Press, 2009
- [21] Lujak, M.; Giordani, S.; Ossowski, S., "Fair route guidance: Bridging system and user optimization," *Intelligent Transportation Systems (ITSC), 2014 IEEE 17th International Conference on* , vol., no., pp.1415,1422, 8-11 Oct. 2014.
- [22] Alesiani, F.; Maslekar, N., "Optimization of Charging Stops for Fleet of Electric Vehicles: A Genetic Approach," *Intelligent Transportation Systems Magazine, IEEE* , vol.6, no.3, pp.10,21, Fall 2014.
- [23] Garg, A.; Pandey, K.; Singh, B., "Hierarchical map-based location service for VANETs in urban environments," *Contemporary Computing (IC3), 2014 Seventh International Conference on* , vol., no., pp.199,205, 7-9 Aug. 2014.

- [24] Tonguz, O.K.; Viriyasitavat, W.; Fan Bai, "Modeling urban traffic: A cellular automata approach," *Communications Magazine, IEEE* , vol.47, no.5, pp.142,150, May 2009
- [25] R.R. Murphy Introduction to AI Robotics. The MIT Press Cambridge, MA 2000
- [26] R. C. Arkin. Behavior-based Robotics. The MIT Press Cambridge, MA, 1998
- [27] De Silva, L.; Ekanayake, H., "Behavior-based Robotics And The Reactive Paradigm A Survey," *Computer and Information Technology, 2008. ICCIT 2008. 11th International Conference on* , vol., no., pp.36,43, 24-27 Dec. 2008
- [28] The Frame Problem. <http://plato.stanford.edu/entries/frame-problem/>.
- [29] H. J. Chiel, L. S. Sterling, and R. D. Beer. A biological perspective on autonomous agent design. In *Designing Autonomous Agents: Theory and Practice from Biology to Engineering and Back*. The MIT Press Cambridge, Massachusetts, 1990 {2.3.5}
- [30] Balch, T.; Arkin, R.C., "Behavior-based formation control for multirobot teams," *Robotics and Automation, IEEE Transactions on* , vol.14, no.6, pp.926,939, Dec 1998
- [31] Craig Renolds Flocks, herds and schools: A distributed behavior model. *Computer graphics* 21(4):34, 1987
- [32] M. Mataric Designing control laws for cooperative agent teams in *Proceedings of the 1993 IEEE international conference on Robotics and Automation*, pp 830-835, Nice, France, May 1992
- [33] Minhas, U.F.; Jie Zhang; Tran, T.; Cohen, R., "A Multifaceted Approach to Modeling Agent Trust for Effective Communication in the Application of Mobile Ad Hoc Vehicular Networks," *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on* , vol.41, no.3, pp.407,420, May 2011.
- [34] Eichler, S.; Ostermaier, B.; Schroth, C.; Kosch, T., "Simulation of car-to-car messaging: analyzing the impact on road traffic," *Modeling, Analysis, and Simulation of Computer and Telecommunication Systems, 2005. 13th IEEE International Symposium on* , vol., no., pp.507,510, 27-29 Sept. 2005.
- [35] Hoe Kyoung Kim; Hunter, M.P.; Fujimoto, R.M., "A simulation-based investigation of a dynamic Advanced Traveler Information System," *Simulation Conference (WSC), Proceedings of the 2009 Winter* , vol., no., pp.1124,1135, 13-16 Dec. 2009

- [36] Bo Chen; Cheng, H.H., "A Review of the Applications of Agent Technology in Traffic and Transportation Systems," *Intelligent Transportation Systems, IEEE Transactions on* , vol.11, no.2, pp.485,497, June 2010
- [37] Chisalita, I.; Shahmehri, N., "On the Design of Safety Communication Systems for Vehicles," *Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on* , vol.37, no.6, pp.933,945, Nov. 2007
- [38] G. H. Cooper and S. Krishnamurthi. Embedding dynamic dataflow in a call-by-value language. In Proceedings of the 15th European conference on Programming Languages and Systems, ESOP'06, pages 294–308, Berlin, Heidelberg, 2006. Springer-Verlag.
- [39] I. Maier, T. Rompf, and M. Odersky. Deprecating the Observer Pattern. Technical report, 2010.
- [40] 2.4.3 C. Elliott and P. Hudak. Functional reactive animation. In Proceedings of the second ACM SIGPLAN international conference on Functional programming, ICFP '97, pages 263–273, New York, NY, USA, 1997. ACM.
- [41] H. Nilsson, A. Courtney, and J. Peterson. Functional reactive programming, continued. In Proceedings of the 2002 ACM SIGPLAN workshop on Haskell, Haskell '02, pages 51–64, New York, NY, USA, 2002. ACM.
- [42] Peterson, J.; Hager, G.D.; Hudak, P., "A language for declarative robotic programming," *Robotics and Automation, 1999. Proceedings. 1999 IEEE International Conference on* , vol.2, no., pp.1144,1151 vol.2, 1999
- [43] Byeong-Joon Lee; Dong-Ju Lee; Gyun Woo, "Functional Reactive Program Translator for Controlling Robot Systems," *Convergence Information Technology, 2007. International Conference on* , vol., no., pp.1322,1325, 21-23 Nov. 2007
- [44] <http://benjchristensen.com/2013/05/01/functional-reactive-in-the-netflix-api-with-rxjava/> retrieved 7/12/2014
- [45] Naranjo, J.E.; Bouraoui, L.; Garcia, R.; Parent, M.; Sotelo, M.A., "Interoperable Control Architecture for Cybercars and Dual-Mode Cars," *Intelligent Transportation Systems, IEEE Transactions on* , vol.10, no.1, pp.146,154, March 2009

- [46] Neil Sculthorpe and Henrik Nilsson. 2009. Safe functional reactive programming through dependent types. *SIGPLAN Not.* 44, 9 (August 2009), 23-34.
- [47] Neelakantan R. Krishnaswami. 2013. Higher-order functional reactive programming without spacetime leaks. *SIGPLAN Not.* 48, 9 (September 2013), 221-232
- [48] Chi-Chung Tao, "Dynamic Taxi-Sharing Service Using Intelligent Transportation System Technologies," *Wireless Communications, Networking and Mobile Computing, 2007. WiCom 2007. International Conference on* , vol., no., pp.3209,3212, 21-25 Sept. 2007
- [49] Bazzi, A.; Masini, B.M., "Taking advantage of V2V communications for traffic management," *Intelligent Vehicles Symposium (IV), 2011 IEEE* , vol., no., pp.504,509, 5-9 June 2011.
- [50] Turing, Godel, Curch. Computability. Edited by B. Jack Copeland, Carl J. Posy and Oron Shagrir, MIT Press, Cambridge MA, June 2013
- [51] <http://haskell.cs.yale.edu/wp-content/uploads/2011/02/rt-frp1.pdf>, retrieved 5/13/2014
- [52] Alessandro Margara and Guido Salvaneschi. 2014. We have a DREAM: distributed reactive programming with consistency guarantees. In *Proceedings of the 8th ACM International Conference on Distributed Event-Based Systems (DEBS '14)*. ACM, New York, NY, USA, 142-153
- [55] Engineer Bainomugisha, Andoni Lombide Carreton, Tom van Cutsem, Stijn Mostinckx, and Wolfgang de Meuter. 2013. A survey on reactive programming. *ACM Comput. Surv.* 45, 4, Article 52 (August 2013), 34 pages.
- [56] Belwal, C.; Cheng, A.M.K., "Feasibility Interval for the Transactional Event Handlers of P-FRP," *Trust, Security and Privacy in Computing and Communications (TrustCom), 2011 IEEE 10th International Conference on* , vol., no., pp.966,973, 16-18 Nov. 2011
- [57] Ras, J.; Cheng, A.M.K., "Response Time Analysis for the Abort-and-Restart Event Handlers of the Priority-Based Functional Reactive Programming (P-FRP) Paradigm," *Embedded and Real-Time Computing Systems and Applications, 2009. RTCSA '09. 15th IEEE International Conference on* , vol., no., pp.305,314, 24-26 Aug. 2009
- [58] <https://pypi.python.org/pypi/Trellis>

[59] http://www.dlr.de/ts/en/desktopdefault.aspx/tabid-9883/16931_read-41000/