Adaptive Routing of Autonomous Vehicles using Neighborhood Traffic

by

Shanthini Rajendran

M.Tech, VIT University, Vellore, India, 2011

THESIS SUBMITTED IN PARTIAL FULFILLMENT OF

THE REQUIREMENTS FOR THE DEGREE OF

MASTER OF SCIENCE

IN

COMPUTER SCIENCE

THE UNIVERSITY OF NORTHERN BRITISH COLUMBIA

Jun, 2018

Abstract

As research on autonomous vehicles increases, automotive manufacturers and researchers are developing coordination techniques to enable safe passage of vehicles through intersections. These techniques are called Autonomous Intersection Management (AIM). Even though AIM techniques improve intersection throughput, they do not effectively reduce congestion. Real life urban roads comprise of a networks of multiple intersections. In such scenarios, communicating traffic information between intersections is essential for reducing congestion on the roads.

To achieve this, we propose an adaptive routing algorithm that incorporates a fusion of vehicle-to-intersection (V2I) communication and intersection-to-intersection (I2I) communication in order to bring about significant reductions in congestion. To implement this algorithm, we constructed the Enhanced AIM simulation framework as an extension of AIM simulator (University of Texas, Austin). We demonstrate with simulation experiments that our proposed routing algorithm shows reduced congestion and wait-time, and improved user experience.

Yet only time keeps us apart,

you are in the shadows of my heart

Dearest Mummy

Publications contributed to this Thesis

1. Shanthini Rajendran, Suresh Rathnaraj Chelladurai, and Alex Aravind. 2016. An Adaptive Road Traffic Regulation with Simulation and Internet of Things. In Proceedings of the 2016 annual ACM Conference on SIGSIM Principles of Advanced Discrete Simulation (SIGSIM-PADS '16). ACM, New York, NY, USA, 3-11.

2. Shanthini Rajendran and Alex Aravind. 2018. Adaptive Routing of Autonomous Vehicles using Neighborhood Traffic. Manuscript in preparation for publication.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

"Right now the phone is an accessory to the car, but soon the car is going to be an accessory to the phone." - Mark Andreessen, Co-Founder (Andreesen Horowitz, Netscape) [2]

1.1 Background

Advances in the field of artificial intelligence have led to a new era of intelligent, knowledge-based, mass transportation technology. Specifically, Intelligent Transportation Systems (ITS) is the field that specializes in integrating information technology with vehicles and transportation infrastructure, to make transportation safer, cheaper, and more efficient [3]. Given the advancements in ITS in the past few years, it is becoming easier to envision a future in which vehicles are increasingly able to handle the majority of the driving themselves. The rise in technology surrounding self driving cars or autonomous vehicles suggests significant reduction in traffic congestion and minimal wait times at intersections as an alternative reality. As the number of autonomous vehicles (AVs) on the roads continue to increase, the benefits of smoother, steady traffic flow will be realized to their fullest potential.

In 2017, INRIX analysed congestion in 296 cities across the United States. They reported that traffic congestion cost drivers more than \$305 billion in direct and indirect costs in 2017 alone, a \$10 billion value increase compared to 2016. The direct costs include the cost of the time spent in congestion, plus the additional fuel cost and the social and environmental cost of emissions released by the vehicle. The indirect costs are borne by households through the increase in the prices of goods and services due to congestion faced by businesses [4]. In the UK, INRIX analyzed congestion in 111 cities cost £37 billion in 2017, an increase of 28 percent in direct and indirect costs compared to 2016 [4].

Many reasons contribute to traffic congestion. Increasing number of personal vehicles is one of the main reasons. Poor driving skills and improper road infrastructure also contribute to traffic congestion [5]. Another important reason that contributes to traffic congestion is intersections, especially the ones that are coordinated by traffic signals.

Traffic signals are traditionally controlled by centralized timers that schedule the timing cycles of the signals, based on location and peak-hours of traffic. Normally, these timers are updated and adjusted every two years based on the average traffic flow of the city. However, the timing cycles of these traffic signals are not adjusted based on the real-time traffic situation, and this causes unnecessarily long wait times at intersections [6], [7].

Many attempts have been made to increase the accuracy and real-time responsiveness of signalized intersections. Signalized intersections were built with inductive loops, and motion and vision sensors in order to monitor traffic, weather conditions, road conditions, and control signal timing accordingly. On the vehicle side, initiatives like the Intelligent Car Initiative, spearheaded by the European Union, make extensive use of electronic devices such as sensors, microcontrollers, and actuators in cars for speed control, real time traffic control and for sensing dangerous situations and safely avoiding them [8].

Autonomous vehicles (AV) are predicted to be the future of mobility [9]. This reality is coming faster than we think it is. In order to give the context of where the AV technology stands today, the trends in development of AVs have been classified into four levels of driving according to [10]. This can be summarized as follows:

1. *Level 1 - Function Specific Automation*: This level focuses on specific functions such as adaptive cruise control, parking assist, and lane guidance, etc. The human driver is completely responsible for control of the vehicle.

2. *Level 2 - Combined Function Automation*: This level of technology allows the driver to partially disengage (hands off the steering wheel) only under certain conditions, while most of the time the driver is still in control and is responsible for monitoring the road. An example would be adaptive cruise control with lane centering.

3. *Level 3 - Limited Self-Driving Automation*: This level of technology allows the driver to hand over control of most of the safety-critical applications, and rely on the vehicle to monitor changes in those conditions that will require transition of control back to the driver. There is no necessity for the driver to pay attention to the environment continuously.

4. *Level 4 - Full Self-Driving Automation*: In this level, the vehicle can perform all driving functions, and can monitor traffic conditions for the entire trip. The vehicle can then be operated without human intervention.

Currently, only Level 2 automation has been made available for public use. Most pilot projects put forth by major car manufacturers fall in Level 3, which is the state of the art now. Technology surrounding AVs needs to develop at a greater pace in order to attain Level 4 automation. Figure 1.1 gives the overview of the timeline when AVs will be made available for general public use.

In the following subsection, we will provide a historical account of autonomous vehicles and the benefits and challenges they impose to the society.

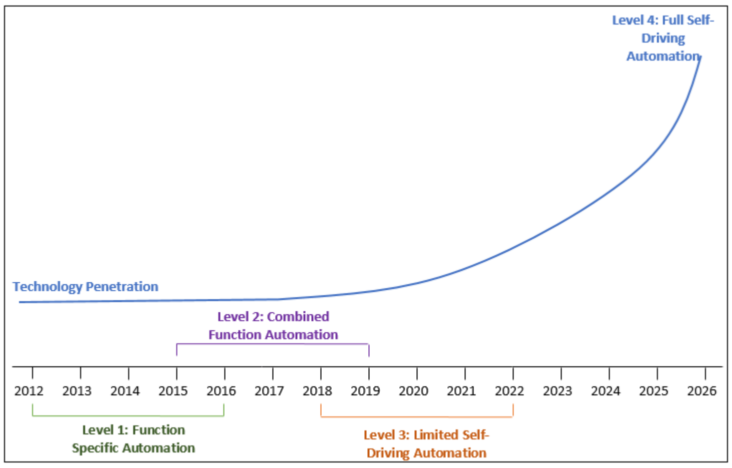


Figure 1.1: Predicted Timeline for Autonomous Vehicles [1]

1.1.1 A Brief History

The first attempts to make AVs began in the 1980s in Germany, by Ernst Dickmanns and his group at Bundeswehr University (UniBW) in Munich. Some of these UniBW cars would drive as fast as 96 km/h on empty streets. This was followed by the largest AV project ever: the pan-European Prometheus project worth almost $1 billion. It involved UniBW and many other research groups developing AVs across Europe between 1987 and 1995 [11]. One of these cars, the "VAmP" (a Mercedes 500 SEL) guided by vision sensors, drove in Paris traffic in 1994, tracking up to 12 other cars simultaneously. It drove more than 1000 km on the Paris multi-lane ring, up to 130 km/h, automatically passing slower cars in the left lane. In 1995, an S-class car of Dickmanns and UniBW, autonomously drove a round trip of 1678 km on a public highway from Munich to Denmark, at up to 180 km/h, passing other cars autonomously [11].

In 2005, in the USA, DARPA started its "Grand Challenge" in the desert. The course was 211 km long and the fastest team to win was from Stanford University, who did the whole course in almost 7 hours [12]. This was followed by a similar demonstration in Europe in 2006, called ELROB (European Land Robot Trials) which was conducted with autonomous off-road vehicles. In 2007, there was another DARPA Grand Challenge for an urban traffic scenario, where the driverless cars would try to complete missions given to them within a specifed time period. Several teams successfully developed a vehicle that has the ability to drive itself and achieve the assigned mission [12]. Most successful teams employed high-end laser scanners coupled with radars for perception and high-precision GPS/INS for localization [13], [14].

The Grand Cooperative Driving Challenge (GCDC), 2011 was the first international competition to implement highway platooning scenarios of cooperating vehicles connected with communication devices [15]. In July 2013, a team from University of Parma, Italy [16] performed another impressive autonomous driving experiment in public. Interesting work that aims for autonomous driving with close-to-production vehicles is presented in [17]. Already several autonomous vehicles have been demonstrated by Google [18], and Tesla [19]. Tesla's Autopilot features automated steering and acceleration in limited conditions [20]. Waymo and Uber announced intentions to begin testing autonomous taxi services [21], [22]. Despite this progress, significant technical progress is needed before vehicles can drive themselves under all normal conditions [23].

1.1.2 Benefits and Impacts of Autonomous Vehicles

The benefits of autonomous vehicles are manifold [9]. The first and the most important one is safety. Autonomous vehicles have the potential to reduce crashes dramatically. The number one reason for traffic accidents worldwide is human error. Over 40% of these fatal crashes involve distraction, drug influence, or fatigue. Fully autonomous vehicles can guarantee 40% reduction in fatal crashes, assuming malfunctions in autonomous vehicles are well within the allowable error margin.

In addition to safety, autonomous vehicles open up the possibility for efficient autonomous intersection management technologies to be deployed. The result is lower traffic delays compared to current technologies such as traffic lights and stop signs. Reduced traffic congestion will invariably reduce fuel consumption as well.

Autonomous vehicles are also expected to use existing lanes and intersections more efficiently through shorter gaps between vehicles, coordinated platoons, and more efficient route choices [24]. The improved safety and reduced congestion benefits of autonomous vehicles have the potential to impact travel behavior significantly. Autonomous vehicles will open up opportunities for the young, elderly, and disabled population to use roadways more effectively. Additional fuel savings can be obtained by allowing the vehicles to communicate with parking infrastructure to enable driverless dropoffs and pickups.

1.2 Motivation

In the previous subsection, we saw how advances in ITS and autonomous vehicles are going to impact the future. In this subsection, we give a brief overview of the research area that forms the focus of this thesis.

When fully autonomous vehicles become the norm of road transportation in the future, the bottleneck that causes congestion will shift from inefficient road infrastructure, traffic lights and human drivers, to intersections. On freeways, there are usually no pedestrians or cyclists and vehicles travel in the same direction with similar velocities. There is no need for more than a simple reactive behavior to keep the vehicle in the same lane and to maintain reasonable distance between vehicles.

Intersections are a completely different story. Vehicles constantly cross paths, in many different directions inside an intersection. A vehicle approaching an intersection can quickly find itself in a situation where collision is unavoidable, even when it has acted optimally. Traffic statistics support the sensitive nature of intersections. Vehicle collisions at intersections account for anywhere between 25% and 45% of all collisions [5]. As intersections occupy only a small portion of the roadway, this is a disproportionate number.

Given the sensitive nature of intersections, it is important then to be able to coordinate traffic safely and efficiently at intersections. The techniques or algorithms or policies used to do the same are called Autonomous Intersection Management (AIM) techniques. AIM techniques solely focus on avoiding collisions and coordinating autonomous traffic effectively at intersections. In order for AIM techniques to work efficiently, we must assume the autonomous vehicles operate in a connected environment where each vehicles possesses the capability to communicate with each other and with the road side infrastructure provided at intersections.

Dresner and Stone [25] proposed an AIM technique based on a reservation system for reducing traffic congestion, specifically at intersections. The simulation results showed that their AIM technique outperformed traffic lights in terms of congestion and wait-time at an intersection. A similar conclusion was drawn by Fajardo et al. [26]. Consequently, it is becoming clear that AV technology might have a positive impact on traffic congestion. However, there is also a threat that this technology can induce additional demand, thus adding more pressure to an already congested network. The impact of the AV technology on traffic congestion is an area yet to be explored.

Even though AIM techniques improve throughput at intersections, they do not guarantee reduced congestion in the overall network. Real life urban roads are comprised of a networks of multiple intersections. In such a scenario, communicating traffic information between intersections is essential for reducing congestion on the roads. In this context, we reason that the presence of Intersection to Intersection (I2I) communication will allow smart intersections or intersection managers (IM) to communicate real time traffic density information, as well as emergency/trauma related information. Each intersection will communicate with its immediate connected neighbors. This way, we will be aware of the traffic density status of a local neighborhood at all times. We propose a routing algorithm that uses this local neighborhood information to route traffic in an adaptive manner.

1.3 Research Problem

The goal of this thesis is two-fold. First, we aim to devise an adaptive routing algorithm for AVs that seeks to minimize congestion in the network. Second, we aim to study how this adaptive routing algorithm affects user experience. Based on the given description, the primary research question for this thesis is as follows.

> *How do we develop an adaptive routing algorithm that is capable of routing autonomous vehicles using I2I communication?*

While coordination of traffic at an intersection represents a local problem, its combination with other intersections and the topography of a city makes routing autonomous vehicles - a difficult challenge to address. The sub research questions are as follows:

- What are the impacts (advantages and disadvantages) of the proposed adaptive routing algorithm?
- How does this adaptive routing algorithm affect congestion and wait times in a given traffic scenario?
- What impact does routing have on a user's experience during their journey?

1.3.1 Need for Computer Simulation

Constructing a real system to answer these questions is not feasible. We need a simulation framework that can help users design various scenarios and measure their characteristics. Computer simulation generated output can be used to infer answers for these questions. In this thesis, we attempt to answer the above questions using Enhanced AIM, an adaptive simulation framework for autonomous vehicles. This framework was constructed by extending the AIM simulation framework developed by Peter Stone and Kurt Dresner et al. at the University of Texas, Austin [27].

1.4 Thesis Contributions

The thesis has three main contributions:

1. *Intersection to Intersection Communication Module:* We believe I2I messages will play a central role in a connected and autonomous vehicle scenario. In this thesis, we add the I2I communication module as an enhancement to the AIM simulator. We also propose a communication protocol standard for messaging between neighboring intersections.

2. *Messaging Middleware:* We use a message-oriented middleware that integrates simulation and analytic engine. The messaging middleware is responsible for connecting these components and facilitating message transfers in near real-time. The middleware is completely reproducible and is built from scratch using open source off-the-shelf components.

3. *Analytic Engine:* We implement a mining repository and a data store that enables storage of all the interactions between various agents within the simulator and supports offline as well as online data analysis.

Simulation experiments were conducted and the results are reported.

1.5 Structure of the thesis

The rest of this thesis is structured as follows. Chapter 2 provides a literature survey. It includes a background about traffic simulation of AVs and an overview of various traffic simulators used for the study of AVs. It also presents an account of existing autonomous intersection management approaches, their advantages and disadvantages. Design and implementation details of the proposed adaptive routing algorithm using I2I messages is discussed in Chapter 3. Chapter 4 describes the key features present in the AIM simulator as well as the Enhanced AIM simulator. Chapter 5 describes the experiments conducted in order to evaluate the system. In Chapter

6, we describe some ethical aspects surrounding autonomous vehicle technology that needs to be considered. Finally, in Chapter 7, we conclude the thesis and provide future directions to extend the work carried out in this thesis.

Chapter 2

Literature Survey

Today's cars are parked 95% of the time.

- Paul Barter, Urban Transport Researcher, National University of Singapore, Murdoch University [28]

This chapter is divided into three sections. In Section 2.1, we discuss how transportation systems can be modeled as a multiagent system. This section also includes a brief overview of the various modeling and simulation techniques that are used to model autonomous vehicles. In Section 2.2, we review the various simulation tools available for autonomous intersection management and the rationale behind choosing AIM simulation framework for this thesis. Section 2.3 provides a review of the existing AIM techniques present in literature.

2.1 Background

2.1.1 Modeling Road Traffic

For many decades, modeling and simulation have been extensively used for studying the flow of road traffic [29]. Traffic modeling is one of the core research areas of ITS and its goal is to develop mathematical tools describing real-world traffic with desired accuracy [30]. Modeling gives us the flexibility to represent individual entities of the traffic system either in an abstract or detailed manner. Traffic can be modeled in various ways depending on the level of detail needed to represent the traffic system: from a microscopic level (each car modeled as a separate entity or agent) to a macroscopic level (traffic described by relations between aggregated values, e.g., speed, flow, and density). A comprehensive review of the various traffic models used in literature is provided in [31].

In this thesis, we use a microscopic model for modeling autonomous vehicles. Microscopic modeling of autonomous vehicles has been developed alongside autonomous cars since the 1960s [32]. Modeling involves deciding how to represent the various interacting entities of the model (driver, vehicles, intersections, crosswalk, pedestrians, etc.). We then have to identify and define characteristics for each entity and define the simulation environment. Traffic models that showed the impact of V2X communication were more prevalent in the early 2000's. Some examples of such models include lane changing, overtaking and cooperative driving on highways [33], and synchronized driving through intersections with traffic lights [7] and without traffic lights [34].

2.1.1.1 Modeling Vehicles

In many models, vehicles and drivers are modeled as a single component. However, this does not represent the real world accurately. A vehicle is modeled with features like speed, acceleration, heading, deceleration, length etc. During simulation, current position and direction of the vehicle in the environment are required to keep track of the current state [35].

2.1.1.2 Modeling Drivers

The authors in [36] suggest that most of the decisions made by the driver can be classified as a macro goal or a micro goal. Macro goals are decisions that affect the destination and route taken, while micro goals involve localized decisions at each point of time in the interest of achieving the macro goal. The macro goal involves daily planning and route generation; often influenced by the origin-destination matrix. Micro goals are decisions that govern the control of the vehicle, such as desired speed, overtaking and turning. Drivers all have different driving styles, which are governed by their individual characteristics such as aggressiveness, confidence and driving experience [37], [36].

2.1.1.3 Modeling the Environment

In a traffic system, the environment in which vehicles drive is a road network which is made up of link segments and nodes. The network of roads is usually modeled as an undirected graph for simplicity [38]. Nodes form the intersection. Each link can have one or more lanes, and may operate in one or both directions. Links have properties such as length, number of lanes, speed limit etc. [25].

2.1.2 Agent-Based Modeling of Road Traffic

Each vehicle in a microscopic model interacts with others in a certain way. This means that at every simulation step, each vehicle is considered as an individual agent, and the parameters associated with that vehicle agent such as position, velocity, acceleration, heading, etc. are updated [30]. There are several different types of commonly used traffic simulation models: vehicle-following (VF) models, Cellular Automata (CA) models and the multiagent (MA) models.

In the past, single-lane car-following models have been successfully applied to describe traffic dynamics [39]. Models that use CA have been studied in the past decade, but they do not realistically reflect driver behaviour. With CA models, vehicles are more or less modeled as entities with irregular acceleration and deceleration rates [40], [41], [42]. The agent metaphor has proven to be a promising choice for complex models such as road networks because it allows abstraction at a conceptual level. The abstraction approach of multi-agent systems (MAS), consists of representing a traffic system by multiple agents that exist in a common environment, and interact in order to achieve specific goals [43]. These agents exhibit intelligence, autonomy, and some social ability. They pursue individual or collective goals, and interact with one another and the environment, as well [37].

In a traffic simulation environment, the vehicle agent senses the environment to know how many vehicles there are on the road and their driving behavior. Each vehicle agent looks at other vehicles on the road periodically, and moves to reach its destination safely in the fastest possible way. The adaptability and flexibility of an agent makes it possible to control various types of vehicles with different driving styles. This way, the simulated vehicles will behave in a manner close to the real system, and the interaction between multiple vehicle agents can be studied. The MA models allow solving problems collaboratively by coordinating the knowledge, goals and plans of autonomous intelligent agents. It offers advantages such as faster response, increased flexibility, robustness, resource sharing, graceful degradation, and better adaptability of integrating pre-existing and stand-alone systems [44].

2.1.3 Simulation of Road traffic

Simulation is used to study the change in behavior of traffic systems under various scenarios with different parameters. Simulation is proven to be an easier, more flexible, and cheaper alternative to perform cause-effect analysis, identify bottlenecks, and study various factors about a system as compared to a real life system. In recent years, with computer technology quickly advancing, simulation is being used more and more for AV research, and to explore ideas of controlling the traffic of the future. Over the last few decades, traffic simulators were widely used as a tool to assist in making decisions in mobility and infrastructure planning. These tools help in analyzing congestion issues, and are able to predict the consequences of changes to the system (or a road network), like adding an extra lane, adding a new road, increasing the maximum speed limit, etc. [45]. Simulations also allow testing the system with scenarios that are not possible, or that are too dangerous to involve human drivers.

In this section, we discussed briefly how modeling and simulation is used for studying traffic systems. In the next section, we will discuss some of the traffic simulators that are used to study autonomous vehicles and future road networks.

2.2 AIM Simulators - A Review

Traffic simulation software has become increasingly used as a tool for studying and analyzing road traffic patterns. In this thesis, we are only concerned with a urban traffic system that supports autonomous vehicles. Keeping this requirement as the primary filter, a study was done to find the simulators that can support the necessary characteristics of the traffic system of interest. By applying this filter, we find the following simulators present in literature that best fit our needs. They are given below.

2.2.1 VISSIM

VISSIM is a commercial traffic simulator developed by PTV in Germany [46]. It is a popular microscopic traffic simulation software used in research and in the transportation industry for planning and analysis purposes. VISSIM provides support for modeling various types of transportation modes such as road, rail, cyclists, pedestrians etc. Various types of vehicles such as cars, trucks, buses, and airplanes can be modeled with VISSIM. VISSIM uses both a psycho-physical vehicle movement model and a rule-based model for modeling vehicles. Simulation of vehicle movements is dependent on the psycho-physical model whereas a rule based model is used for optimizing lane changes on the road network.

Driver behavior can be configured for each simulation. Users are able to configure driver behavior with the help of driver vehicle classes. This gives the user flexibility to select and apply the most suitable method for their analysis. VISSIM offers various ways of visualizing the route choices done by the vehicles. VISSIM also supports intersections of all types including signalized, uncontrolled intersections with applicable right-of-way rules, and autonomous intersections. Almost all the simulation results can be stored as text files or in a database and can be used for analysis.

2.2.2 AIMSUN

AIMSUN (Advanced Interactive Microscopic Simulator for Urban and Non Urban Networks) is widely used for developing traffic management and planning decisions in urban areas [47]. AIMSUM supports various types of traffic modeling such as macroscopic, microscopic, and mesoscopic in order to make a wide variety of analysis possible. The mesoscopic approach is useful for studying scenarios such as adaptive traffic control, or how changing bus stops affects the traffic flow in an urban road network. The microscopic approach is useful for studying traffic dynamics in great detail. For example, the microscopic approach would be extremely useful to study environmental impacts of congestion at an intersection. However, the major limitation of using the microscopic approach would be the need for accurate calibration and computational overhead. A hybrid meso-micro model would overcome the individual limitations of the above models. This is one of the main features to be included in AIMSUN Next, the upcoming, improved version of this tool. AIMSUN is a comprehensive tool that provides the user with a variety of data sets and options for exporting the output of the analysis in various forms making it an excellent choice for analysis purposes.

2.2.3 MITSIMLab

MITSIMLab (microscopic traffic simulation laboratory) is a microscopic traffic simulation model that is used for evaluation of advanced traffic management systems, public transportation systems, route guidance systems, etc. [48]. It was developed by MIT's Intelligent Transportation Systems (ITS) Program. MITSIMLab is an open-source application where its core models have been written in C++ and are fully available. It has been successfully applied in several traffic and research studies in the USA, UK, Sweden, Italy, Switzerland, Japan, Korea, Malaysia and Portugal [46].

MITSIMLab has a repository of traffic models available to the user. A distinguishing feature in this simulator is that users can view the reaction of driver agents to real time traffic information. To allow maximum flexibility in the system, MITSIMLab is implemented in three separate modules as given below.

1. *Microscopic Traffic Simulator (MITSIM)*
2. *Traffic Management Simulator (TMS)*
3. *Graphical User Interface (GUI)*

Microscopic simulation approach is used for modeling traffic flow in the traffic flow simulator (MITSIM). The road network, roads, intersections, and lanes are all represented at the microscopic level. MITSIM accepts time-dependent origin destination trip tables as inputs. For each vehicle then, driving behavior such as speed, aggressiveness and other general vehicle characteristics are specified. MITSIM moves vehicles according to car-following and lane-changing models. The car-following model captures the response of a driver to conditions ahead, as a function of relative speed, headway and other traffic measures. The lane changing model distinguishes between mandatory and discretionary lane changes. Merging, drivers' responses to traffic signals, speed limits, incidents, and toll booths are also captured [48].

The TMS is responsible for the traffic control part of the simulator. The TMS can be used for evaluating a wide range of features such as ramp control, freeway control, intersection control, and external route guidance systems such as variable message signs, and in-vehicle route guidance systems. The input data is given to the TMS, which is then responsible for choosing the route strategy. The control and routing strategies generated by the TMS determines the state of traffic control and route guidance devices. These settings are transferred to the TS. The simulated drivers respond to the various traffic controls and guidance, while interacting with each other. TMS has a generic structure that can represent different designs of such systems with logic, at varying levels of sophistication (from pre-timed to responsive). The TMS is a virtual transportation system operation control center, processing performance data from the sensor network, and generating a strategy. The TMS also simulates a wide range of transit operations control strategies (e.g., transit signal priority and holding for service restoration) defined by the user. The simulation output can be obtained as numerical data tables and via the graphical user interface (GUI), which visualizes traffic impacts through vehicle animation.

MITSIMLab generates various output reports that may be used to evaluate the performance of potential ITS strategies. Travel demand is represented by time-dependent origin-to-destination (OD) trip tables, which show expected conditions or are defined as part of a scenario for evaluation. Based on these tables, individual vehicles are generated. The generated vehicles are assigned driver characteristics (e.g. aggressiveness, planning capability, look-ahead distance, level of compliance with various signs and regulations) and vehicle attributes (e.g. acceleration and speed capabilities and the impact of grade, on these capabilities) based on pre-determined distributions. Route choices are based on a probabilistic model that captures the impact of travel times and biases toward routes that use freeways over urban streets. The impact of real-time information on routing decisions is captured by a route-switching model in which informed drivers re-evaluate their pre-trip route choices, based on the traffic conditions observed en route. MITSIMLab is a time-based simulation model with time steps that may differ for various functions from 0.1 to 1.0 s. It also incorporates event-based approaches for situations such as crash avoidance and responses to changes in traffic controls, and information settings.

2.2.4 MATSim

MATSim (Multi-Agent Transport Simulation) is another major traffic simulator. Its focus is on the bigger picture of evaluating how people choose different aspects of their trips —where to go, when to leave, what route to take —rather than simulating detailed traffic conditions [49]. MATSim can be considered more of a mobility simulator than that of a traffic simulator. MATSim supports two traffic flow models, the newer of which uses discrete-event time. Equations give times at which links will change state (meaning a driver exits or enters a road). Effects like congestion spillback have to be explicitly modeled. MATSim is agent-based in the sense that each driver updates its trip plan after getting a score from simulation. Modeling new control policies for intersections for instance, would not be straightforward in MATSim. As such, further comparison is difficult. For reference, MATSim 0.5 consists of 140,000 lines of well-organized and documented Java. The tutorials make it quick to start using, but getting visualization components to run takes more work.

2.2.5 SUMO

Simulation of Urban Mobility (SUMO) is an open source, highly portable, microscopic road traffic simulation package, designed to handle large road networks [46]. It is licensed under the GPL. Its features include collision free vehicle movement, multi-lane streets with lane changing, fast execution speed, dynamic user assignment, and others.

Parameters for simulation can be set using XML files for demand data, routes, turn definitions at junctions, the map, traffic signal timings, and so on. However, its numerous capabilities come at the cost of a massive code-base of 125,000 lines of C++. Upon rough inspection, the code is not easy to understand.

SUMO also has a high start up cost. There are tutorials provided but they are not very user-friendly. SUMO is a much more powerful simulator, but it comes at the cost of a more complex code-base and is a difficult user experience for beginners. An argument may be made for the choice of C++ for speed, but there are doubts.

2.2.6 AIM Simulator

Dresner and Stone have developed an open-source simulator written in Java, which they call the AIM Simulator [27]. The AIM simulator was mainly built to evaluate the performance of the AIM protocol. The simulator is small, flexible and built in a modular fashion. For a single intersection, the simulation area is modeled as a 250 m x 250 m area, having the intersection at the center of that area.

During each time step, the following events take place.

- *New vehicles are spawned based on a probability distribution*
- *Provides sensor input through vehicles sensors/actuators to all vehicles*
- *Allows driver agents controlling the vehicles to act*
- *Updates vehicle's positions based on a physical model*
- *Removes vehicles after reaching the end of the simulation area*

A driver agent is a computer software that controls and pilots an autonomous vehicle, taking the role of a human driver. In order for driver agents to take the wheel, they have to access the vehicle's properties (e.g. VIN, size, and acceleration capabilities), and state variables (e.g. velocity, heading and acceleration). In addition, they have access to a set of simulated external sensors. One of those sensors is a simulated laser range finder, which determines close-by vehicles, and provides the distance and angle to a point on the nearby vehicle closest to the sensing vehicle. This provides the driver agent with the information needed to control the vehicle so that it does not hit vehicles in front. Vehicles' positions are updated at each time step, based on a physical model.

The Autonomous Intersection Management (AIM) project introduces an intersection control scheme designed for autonomous vehicles. As drivers approach an intersection, they request their desired movement from an intersection agent. This intersection manager determines when the driver can safely make the turn and sends back a reply. The manager predicts potential collisions by dividing the intersection into a grid and reserving space-time tiles for each vehicle. The buffer around a vehicle can be adjusted to account for mechanical inaccuracies, in case the autonomous vehicle cannot exactly follow its intended path.

AIM is not a general traffic simulator; instead, it focuses on interactions between vehicles and infrastructure at one intersection. AIM has been extended to cover a small network of linked intersections. As a point of comparison, the AIM 1.0.4 simulator consists of 40,000 lines of Java.

2.2.7 AORTA

Approximately Orchestrated Routing and Transportation Analyzer (AORTA) is a simulation platform exclusively designed for evaluating intersection policies and testing AV behaviors [50]. One of the key features of using AORTA is that simulations can be run on OSM maps which are generated from real road data. A map for any desired city in the world can be downloaded, and then parsed by AORTA to set up a scale simulation of the real world in a few minutes.

AORTA is an open-source simulator and is easily extensible, making it easy for users to test out a number of agent behaviors and intersection policies in a short time span. AORTA is divided into three modular components: the map model, micro-simulation engine, and user interface (UI). The map model transforms OSM maps into AORTA graphs, then answers path finding and geometry queries. The simulation engine adds a notion of agents, vehicle dynamics, and collisions. Finally, the UI interactively renders the map and agents. A headless mode also exists to run experiments without the overhead of visualization.

2.2.8 Discussion

Kokkinogenis et. al. provide a comprehensive survey on the various types of agent based autonomous vehicle simulators, classifying them by several factors [51]. Two factors were taken into consideration when deciding the type of simulation framework that would be suitable for this thesis. Firstly, the software must be open source, and secondly, it should support modeling vehicles as fully autonomous vehicles. The first two simulators discussed, VISSIM and AIMSUN, are commercial simulators, and therefore we refrained from using them. The other simulators discussed above are open source simulators. On applying the second filter, we were left with two options that were closest to our objectives in terms of functionality and code complexity:

AIM simulator and AORTA.

It is possible to implement the adaptive routing algorithm in AIM simulator as well as AORTA. However, AIM simulator gives us the opportunity to do so with a relatively simpler and well documented code base. Currently, AIM supports only 25 intersections. The fact that AIM was the inspiration for AORTA, helped us rule out AORTA. After careful analysis of the available simulators, AIM simulator was selected to be the base platform for building Enhanced AIM [52]. Because the simulator is implemented in JAVA, which I was already familiar with, together with the fact that it is an open source tool, it was a reasonable option. Additionally, its programming is structured very well, which makes it relatively easy for us to adapt the environment to our own needs.

2.3 Autonomous Intersection Management Techniques

One of the earliest research works in the area of intersection management for autonomous vehicles was established by Dresner and Stone at the University of Texas in 2004 [25]. They created a First Come First Served (FCFS) reservation-based AIM techniques or FCFS for short, that can be used by autonomous vehicles to pass through intersections. They also showed that the FCFS technique clearly outperforms current intersection management technologies such as traffic lights and stop signs [27]. The FCFS technique is centralized and vehicles communicate with intersection managers placed at each intersection.

Vehicles planning to enter the intersection try to reserve a space-time block in the intersection, by sending information to the intersection manager about their time of arrival, velocity of arrival, and their capabilities such as their maximum acceleration/deceleration and size. This information is sent as a package, called a proposal. The intersection manager runs an intersection control policy to compute whether a reservation should be granted or rejected depending on reservations that were already granted. This is done on a First Come, First Serve (FCFS) basis; and sends the result

—accept or reject, back to the requesting vehicle. A vehicle cannot enter the intersection unless it receives a confirmation message about its reservation request. If the vehicle does not receive a reply from the intersection manager, it stops before entering the intersection and keeps on sending requests until one of them gets accepted.

The reservation concept was widely accepted across the research community because it showed significant improvements in terms of throughput, average speed and average wait time at intersections. The FCFS intersection control policy in AIM was changed to a look-ahead intersection control policy (LICP) by another research group [53]. In LICP, the main concept is that if the average intersection delay would be improved by delaying or canceling a reservation, then this will be done even if a vehicle had a higher priority than all the other conflicting vehicles. This is usually done when a vehicle has conflicting trajectories with many other vehicles; so it would be better for all the other vehicles if this vehicle were denied access to the intersection for some time. The LICP policy is shown by the authors to make around 25% average performance improvement on intersection delay compared to the FCFS policy. The authors also address the issue of fairness in their policy, by allowing vehicles that have waited for a long time to pass through the intersection even if this will negatively affect the average intersection delay. De La Fortelle [54] also proposes a reservation-based AIM algorithm, and focuses on accepting reservations in a heuristically efficient order.

Many approaches that were claimed to improve on the reservation approach for intersection management were proposed. The authors in [55] classify these approaches in two categories.

1. *Planning based approaches*
2. *Hybrid approaches*

2.3.1 Planning Based Approaches

In planning-based approaches, it is the job of an IM to find collision-free trajectories for all vehicles. Then the vehicles should follow the trajectories to cross the intersection. If the vehicles fail to follow the planned trajectories, collisions may become unavoidable during simulation. The main limitation of this method is the amount of computation that happens at the IM for generating collision free trajectories for all the cars. Lee and Park argued that trajectory generation is a complex, non-linear, constrained optimisation problem [56]. This technique computes the entire trajectory of each vehicle through the intersection, and if an unexpected event (e.g. a mechanical breakdown) occurs, then those trajectories are invalidated. Planning based approaches do not offer the flexibility to consider the unique needs of each vehicle as this will necessarily introduce a large number of parameters, adding more complexity to an already complex system. To compute optimal trajectories that do not collide with each other, it is necessary to use optimization tools such as such as Active Set Method, Interior Point Method, and Genetic Algorithms.

To address this complexity issue, Kamal et al. [57] formulated a model predictive control (MPC) problem that generates the vehicle trajectories for a given duration in the future. The MPC problem is solved in a receding horizon fashion to take into account changes of the environment. The above approaches first determine the trajectories of the vehicles in a centralised entity and then, in a second phase, require vehicles to follow their assigned trajectories. A slightly different approach, one that allows partial decentralisation, are proposed in [58–60].

The authors in [61] established an autonomous intersection model where vehicles entering a cooperative area inform the intersection manager of their arrival. The IM maintains a queue and adds that vehicle to its waiting queue. Simultaneously, the IM also maintains a permission list. At every time step, the IM will add vehicles from the waiting queue to the permissions list. After this the vehicles in the permission list are permitted to pass through the intersection. Passing through the intersection, they will be released from the permission sequence. In addition, by adjusting the speed, the vehicles can pass through the intersection area without waiting; improving passing efficiency.

2.3.2 Hybrid Approaches

Hybrid approaches introduce some level of priority in the order of vehicles accessing passage through an intersection. The authors in [58] propose a priority approach based on navigation functions. When two cars are both in the crossroad area, the vehicle's navigation function will send their own route information to the intersection controller. Therefore, the probability of collision between the two cars will be calculated according to the vehicles' locations, driving directions, and speed. If there is a possibility of collision, the vehicle nearer the collision point will be assigned greater priority and be told to accelerate while the other one will slow down to avoid a crash [55].

A similar approach is found in [59] where a crossing order is decided in advance for incoming vehicles. Each vehicle is then locally controlled by an MPC scheme that generates a feasible trajectory in a receding horizon fashion. In [60], incoming vehicles decide their desired trajectories using optimization techniques, according to a predefined decision order. The authors [62] propose to let vehicles choose any possible control scheme, unless this would lead the system into an unsafe state. In that case, the control of one or several vehicles is overridden by a centralized controller, to prevent entering this state.

2.3.3 Slot Based Approaches

A more recent approach towards autonomous traffic management is the slot based approach. This was proposed by the authors Tachet et al. at MIT [63]. Slot based systems are inspired from air traffic coordination systems. A common way of coordinating traffic is to exclusively and, in an alternate fahion, give access to vehicles traveling in one direction through the intersection. In contrast, slot-based systems consider the trajectory of multiple vehicles, and prevent collisions by coordinating the time slot in which the intersection can be crossed safely (simultaneously for multiple traffic directions) [63].

The studies done in [63] shows us that traffic flow in Slot Based Systems is more smooth. Forming platoons of vehicles and serving all vehicles in the platoon before giving way to a conflicting flow, is more efficient from a capacity point of view. Based on the generalized queue theory, the researchers found that the Slot-Based Systems capacity can be doubled and the delay can be significantly reduced, compared to the traffic-light controls [63]. But the premise of Slot-Based Systems is that the roads are basically occupied by autonomous cars, thus there may be decades before this can be tested in real life.

2.4 Summary

Based on what we reviewed in this chapter, it is very clear that research in the field of autonomous vehicles is growing in multiple directions. On one hand, we see an increase in the number of traffic simulators that support autonomous vehicles. There has been a lot of research done in building realistic test beds and simulators that support autonomous vehicles to evaluate the performance of these vehicles. On the other hand, many researchers have contributed to researching different types of autonomous intersection management techniques/policies. A review of relevant literature indicates that improving traffic flow at an intersection should, in modeled traffic, have a significant impact on the overall traffic flow. In the next chapter, we will discuss the research related to routing of autonomous vehicles.

Chapter 3

Adaptive Routing Algorithm using I2I Messages

In this chapter, we describe the design and development of an adaptive routing algorithm for routing autonomous vehicles.

3.1 System Assumptions

For this thesis, we use a road network consisting of multiple intersections similar to the road network proposed in [27]. The multiple intersection model that we consider contains 9 connected intersections with 3 incoming roads and 3 outgoing roads. Each road is four lanes wide and has two lanes in each direction. We use this configuration to retain simplicity and demonstrate proof of concept. The AIM simulator provides capability to model upto 25 connected intersections in the network. In this respect, let us look at some assumptions on which the simulation system is based. Figure 3.1 illustrates the intersection and lane model that is used in this thesis.

- Only autonomous vehicles are present in the system.
- All the vehicles passing through the intersection will participate in the intersection control policy employed by the IM.

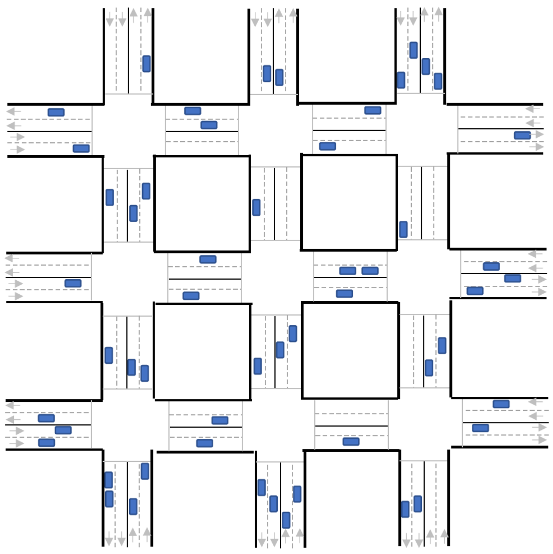


Figure 3.1: Intersection and Lane Model for Enhanced AIM

- Changing lanes are not allowed in the intersection space.
- All vehicles have the ability to detect whether they are entering an intersection or exiting one. They also have the ability to send a request for crossing message to an upcoming IM, when they come within a specified communication range with that IM.
- All the autonomous vehicles are able to communicate with each other and with the IM using wireless communication.
- Reliable communication between the vehicle and the intersection agents is assumed. We assume that there is no explicit delay in processing and transferring and receiving the messages.
- Intersection policy employed by the IM will remain constant throughout the duration of the simulation.
- Finally, we assume that the vehicles have sufficiently powerful brakes to enable a vehicle traveling at the speed limit to stop as soon as the entrance of an

intersection is detected, and not stop in the middle of an intersection, and that the lanes have a sufficiently-wide buffer zones on either side.

3.1.1 Design Criteria

The design criteria for our system is based on the design criteria presented in [27] and aims to satisfy the following properties.

1. *Autonomy*: Each vehicle is an autonomous agent.
2. *Low Communication Complexity*: The number of messages and amount of information transmitted within the system is kept to a minimum.
3. *Sensor Model Realism*: Each agent has access to sensors that are feasible with current-day technology.
4. *Protocol Standardization*: The system employs a simple, standardized protocol for communication between agents.
5. *Deadlock/Starvation Avoidance*: Every vehicle approaching an intersection should eventually pass through the intersection.
6. *Safety*: Vehicles should never collide in the intersection.
7. *Efficiency*: Vehicles should pass through the intersection in as little time as possible.

3.2 Related Work

As seen in the previous chapter, most of the approaches for autonomous intersection management can be categorized into either centralized approaches or decentralized approaches. Centralized approaches are very efficient when considering a road network consisting only of autonomous vehicles, provided those vehicles follow the route proposed by the IM to the rule [24]. We will elaborate briefly.

Imagine a future road network, which consists of intersections and roads with multiple lanes connecting these intersections. We consider only autonomous traffic in this environment. At the beginning of a journey, passengers board an AV and specify a destination. It is the vehicle agents' responsibility to make sure that passengers are transported safely and efficiently, with minimal delay from their respective source to the destination [64]. Each vehicle agent is aware of the map (network), the location of various intersections, etc. This information is used by the vehicle agent to plan their trips respectively. Now, in the present simulation scenario, every IM in the network employs the same intersection control policy. Each IM is responsible for making sure vehicles can have safe, conflict free passage through that intersection. Each IM is also responsible for achieving superior performance by increasing the throughput at that intersection [52].

Current centralized approaches leave most or all of the decisions with respect to coordinating traffic at the intersection to the IM. The IM, however, has limited knowledge about the global traffic distribution in the network. The IM is only concerned with collision-free access in the intersection, which is very important, however not always very efficient. Also, the IM is not fault-tolerant meaning no other IM will know if an IM failed. The vehicles will figure it out eventually, when they send a request message and do not receive any response. This is done at the expense of increasing the wait time of the vehicles approaching that intersection, leading to a bad traffic jam. This whole scenario plays out quite opposite to the motives and the vision behind using self driving cars in the first place.

We present an argument that the current simulation scenario works efficiently for a single intersection but not for a network of intersections. Without any information about the global traffic scenario, there is a high possibility that certain roads in the network will be extremely clogged while others may not suffer heavy traffic volumes. We believe that global information about the network would help in distributing the traffic volume evenly across the network while maintaining efficiency.

Wuthishuwong et. al. present a similar routing algorithm based on I2I messages [38]. They divide the control present in the IM into two levels: network level and the control level. Network level is concerned with density information being transmitted to local, neighboring intersections through I2I messages periodically. The control layer was responsible for translating this data into routing messages. The authors also modeled the traffic flow in the network based on the Greenshield model. The Greenshield model is a popular traffic model that assumes a linear speed-density relationship. Each road has a maximum capacity, and the entire network of roads was routed in such a way that none of the roads reached maximum capacity at any time. A major limitation of this approach is that each vehicle's velocity is controlled so that at no point will a particular road reach its capacity. In simpler terms, during congestion scenarios, vehicles may have to follow the speed limit posted by the IM so that it will reach the intersection at a time that will not cause a jam in the outgoing street. The common goal between the work presented in [38] and ours is reduced wait times and reduced congestion in the network. While there are many ways to achieve this, driving at erratic speeds is certainly not a realistic approach.

3.3 Proposed Method

In this thesis, we propose an adaptive routing policy that will re-route the vehicles based on the traffic density in the connected local neighborhood intersections. For this, we will use I2I messages to share traffic density information of the local connected neighborhood intersections with each intersection manager. The vehicle agents will send proposals (by calculating shortest possible paths) to the IM. Each vehicle agent is able to send multiple proposals in a single request. For simplicity, we limit the number of proposals to a maximum of two.

More information about the request proposal and its structure and the specifics of the I2I communication protocol is given in Appendix A. The enhanced aim simulator uses A* search algorithm for calculating the shortest path proposals. The simulator for does not support lanes of different lengths, hence both the shortest paths to the same destination are of equal length. This is a limitation in the simulator because we assume grid network. In future work, we can explore networks that support lanes of different lengths.

The IM, when evaluating the request, has additional information about the vehicle (destination and arrival time), and based on the proposals the vehicle provides, it can send an accurate route acknowledgement to the vehicle. When the IM receives a request message, the IM will calculate how dense the traffic will be on the connected streets if one more vehicle is to be added. The IM will perform this calculation for both the proposals. In addition, since the IM has density information of all the three outgoing streets, it has the flexibility to choose the path of least congestion. We wont face the problem creating loops in the simulator because vehicles do not have the capability of going in the reverse direction. We demonstrate in our experiments that this routing model will maintain the traffic congestion in the network at an efficient rate while still allowing the vehicle agents to take the shortest possible routes. In this simulation, we assume reliable transmission and reception of messages.

We believe our approach is more realistic in the sense that —

1. each vehicle agent has the freedom to calculate multiple proposals based on its individual goals and

2. each intersection agent has the freedom to choose which proposal to accept based on its individual goals.

In addition, we focus on the problem of solving traffic congestion in a fully autonomous system considering the scenario if one of the IMs fails. Most of the traffic simulators used for studying autonomous vehicles assumes the system will almost never fail. This assumption is even stronger when talking about the communication protocols used in such a system. In this thesis, we assume the alternative, saying how we can efficiently route vehicles when an intersection fails. How can an intersection communicate this information with others in the network and avoid unnecessary congestion or bottlenecks?

3.4 Key Steps

The adaptive routing policy presented in this thesis attempts to solve the routing problem of the IMs in a more efficient and realistic manner. Two different operations happen in the IM. At first we update the active neighbor list for each IM. Second we re-route vehicles based on density information. Each IM has a list of its active neighbors for that update interval. Whenever a vehicle sends a message to the IM, the IM checks the traffic density for both routes given by the vehicle in its proposal. The IM then checks if the destination in the IM specified route is available and, if it is available, it checks for collisions through the intersection. If there are no collisions, then the IM replies back to the vehicle with a CONFIRM message. If the destination IM is not available for both the routes then the IM sends a REJECT message to the vehicle. The order of priority at the IM is as follows.

- *Choose route with lowest density*
- *Check if destination IM is available*
- *Check for collisions*

3.5 Algorithm

Algorithm 1 presents the order of events that happen when an I2I message is received by a neighboring IM.

Algorithm 2 presents the schedule of events that takes place when an IM receives a message from a vehicle agent.

Algorithm 1: IM-IM modules exchange *density* message between neighboring IMs.

```
1  AN;                                                  // List of Active Neighbors
2  Status_AN;                                // Density Status of Active Neighbors
3  Status_CurrentIM;                             // Density Status of Current IM
4  Neighbour_Status_Map <AN_Id,Status_AN>;                   // Map that contain
     density status of all the neighbors
5  Receiver_Queue;                                    // holds received messages
6  while true do
7      if (!Empty(Receiver_Queue)) then
8          Neighbor_Status_Map <— AN_Id, Status_AN;         // Update density
             information in the neighbor map
9      Calculate_Current_Traffic_Density();
10     Process Status_CurrentIM ();
11     Broadcast Status_CurrentIM;          // send current IM density information
12 end
13 Calculate_Current_Traffic_Density()
15     List<Roads> roads = GetEntryAndExitRoads();
17     foreach r:roads do
18         active_vehicles = GetListofActiveVehiclesonRoad();
19     end
```

Algorithm 2: IM receives *message* from V and responds to it using the reservation policy P.

```
Receiver_Queue;                                   // holds received messages
while !$Empty(Receiver\_Queue)$ do
    Dequeue message $M$;
    if ($M.type == Request$) then
        if (Traffic_Density($M.R1$) $\leq$ Traffic_Density($M.R2$)) then
            if N(R1) $\in$ AN & (P($M.R1$)) then Send route $R1$
            else if (N(R2)$\in$ AN & P($M.R2$)) then Send route $R2$
            else Send $Reject$
        else
            if (N(R2)$\in$ AN & P($M.R2$)) then Send route $R2$
            else if N(R1)$\in$ AN & (P($M.R1$)) then Send route $R1$
            else Send $Reject$
        end
    else if ($M.type == Cancel$) then
        Process $Cancel$;
        Send $Acknowledgement$;
    else if ($M.type == Done$) then
        Process $Done$;
        Send $Acknowledgement$;
    end
end
```

3.6 Criteria for Evaluation

We would like to evaluate the impact of the adaptive routing algorithm using I2I messages from two different perspectives. Since recent developments in self driving cars are centered around the user, let us consider the user perspective first. For the second perspective, we take a more traditional approach by considering how the proposed routing strategy and increase in the number of messages in the simulation system is going to affect the system. For both these perspectives, we will provide the advantages and limitations.

3.6.1 User Perspective

Imagine sitting in an AV and choosing your destination. During the course of the journey, an in-car display will show the passengers' information such as route and expected time of arrival. It will also show them what the car can see through its visual system, including its camera and other sensors like LIDAR and RADAR. This is designed in such a way that it will be comforting and educational —so that "riders can understand what the vehicle is perceiving and responding to, and be confident in the vehicle's capabilities" [65]. The whole point of opting for self driving cars is that the car drives itself with little or preferably no human interaction. However, gaining users trust is going to be a big challenge.

Many researchers have studied the relationship between trust and the level of automation from various perspectives [66]. After all, no human user would like to be stuck in a car without any control. In the case of uneasiness or trauma experienced by a user, Waymo has provided a "pull over" button in their cars for such situations. The vehicle will then identify the closest safe place to pull over so that riders can exit. The pull over button is mostly for emergency situations. We believe if we make the system interactive and educational, users will be able to trust the car better and make better choices. For this we propose to display the two route proposals decided by the vehicle, every time it updates its route. We would also display the proposal chosen by the IM at each intersection, an Estimated Time of Arrival and the average congestion rate. Major limitations of this approach are cost and complexity.

3.6.2 System Perspective

From a system perspective, using I2I messages for adaptive routing of vehicles has two advantages. The first advantage is that vehicles can be routed by the IM for the best case scenario most of the time given the circumstances/road conditions present at that time. This represents a traffic system in a realistic manner. In the real world, when we are faced with a road block, there are not many things we can do until the road block is cleared. Although using autonomous vehicles promises shorter wait times and faster travel times, we cannot assume that unexpected delays or even faulty intersections will never happen. Using I2I messages, an IM is able to understand the traffic information within the neighborhood area and is able to make an informed routing decision. The main limitations, however, would be the computational overhead. The simulation becomes slower as the number of calculations done at each IM increases. This makes testing the system with a large number of vehicles infeasible. In our simulation system, we assume communication is reliable. The limitation is that we do not study how the system will be impacted when message loss occurs.

3.7 Summary

In this chapter, we discussed the adaptive routing algorithm and its operation in detail. In the next chapter, we will discuss the features present in the AIM simulator and the enhancements done in our proposed Enhanced AIM simulator in greater detail.

Chapter 4

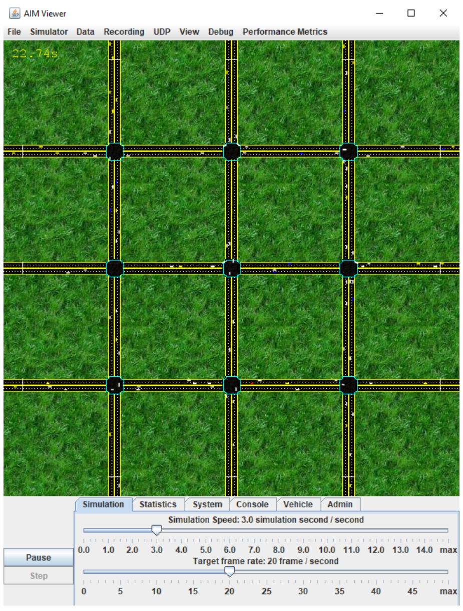
Enhanced AIM Simulator

In this chapter, we see the motivation and the need for Enhanced AIM, our proposed simulation framework. First, we will discuss in detail the AIM simulator and its main features. Next, we will describe the updates done for Enhanced AIM. Finally, we describe in detail the architecture and implementation of Enhanced AIM.

4.1 AIM Simulator

The AIM simulator was mainly built to empirically evaluate the AIM communication protocol as well as the FCFS AIM technique [27]. The AIM protocol defines the message types and the actions involved in establishing a reliable communication between the vehicle and intersection agents. AIM simulator is a time-based simulator.

The simulator models an area that is 250 m X 250 m. If we assume a single intersection, then that intersection is located at the center of the simulation canvas area and its size is determined by the number of lanes traveling in each direction. The number of lanes is variable and we can have a maximum of 6 lanes running in each direction. For experiment purposes, we will assume 2 lanes in each direction. We also assume that vehicles drive on the right side of the road throughout simulation and that at the intersection, each vehicle is capable of turning right or left or going



Simulation Speed: 3.0 simulation second / second

Target frame rate: 20 frame / second

Figure 4.1: A screen shot of the AIM simulator in action

straight depending on which lane they are traveling on respectively. This assumption, however, is not required for the simulator to work properly [27].

Figure 4.1 shows a screen shot of the simulator's graphical display.

At each time step, the following operations take place within the AIM simulator.

1. *Spawn new vehicles in a probabilistic manner at each lane.*
2. *Initialize vehicles sensors/actuators.*
3. *Allow driver agents to control the vehicles.*
4. *Update vehicle positions based on a physical model.*
5. *Removes vehicles after they reach the end of the simulation area.*

There are two main agents in AIM namely the vehicle or driver agent and the intersection agent. We will discuss these two agents in detail below.

4.1.1 Vehicle Agent or Driver Agent

A vehicle agent controls the position, location and navigation of the vehicle at all times. To do this, it has access to the vehicle's properties (e.g. VIN, size, and acceleration capabilities), and state variables (e.g. velocity, heading and acceleration). In addition, it also has access to the vehicle's sensors. For example, each vehicle has a simulated laser range finder sensor that allows the vehicles to determine its proximity to other obstacles and vehicles in their surroundings. This sensor is crucial for the vehicle and it provides information needed for the vehicle agent in order to maintain a safe following distance with other vehicles on the road.

Vehicle agents in the AIM simulator has the following properties —Vehicle Identification Number (VIN), Length, Width, Distance from front of vehicle to front axle, Distance from front of vehicle to rear axle, Maximum velocity, Maximum acceleration, Minimum acceleration, Maximum steering angle, Sensor range and the following state variables —Position, Velocity, Heading, Acceleration, and Steering angle.

The driver agent assigned for navigation of the vehicle may access each of these quantities depending on how the simulator is configured. The driver agent estimates the time and velocity at which it will reach the intersection, and requests an appropriate reservation. If granted a reservation by the IM, it attempts to arrive on schedule. If it determines that it is unable to keep the reservation, it cancels the reservation. If it believes it will be substantially early, it attempts to change to an earlier reservation. If it is unable to get a reservation, it decelerates (down to a minimum velocity) and requests again. It does not enter the intersection without a reservation. On the open road, the driver agent employs a simple lane-following algorithm, and maintains a following distance of one second between its vehicle and the vehicle in front of it. A detailed explanation of the lane following algorithm used in the AIM simulator is given in [27].

4.1.2 Intersection Agent or Intersection Manager

At each intersection, there is an intersection agent that is responsible for calculating trajectories for all the vehicles that are going to use the intersection space at that time. It is also responsible for reserving a space-time block in the intersection for the corresponding vehicles. The intersection manager acts as a stable communication interface between the driver agents and regardless of how the policy makes its decision, the intersection manager must present the same interface to the driver agents [67]. The general intersection manager algorithm is shown in Algorithm 2.

The intersection manager treats Cancel messages and Done messages are almost identically. However, when a Done message is received, the intersection manager knows that the policy can erase any information about the related reservation because the vehicle has successfully completed the reservation. The Done message also may contain information that is useful to the intersection manager and policy. For example, when a vehicle sends a Done message, it could include the delay it experienced crossing the intersection, providing the intersection manager with a sort of reward signal, by which it can judge its performance [67].

Next, we list the main packages in the AIM-Simulator source-code, describing the functionality of the main classes and methods.

4.1.3 AIM Simulator - Main Packages

A detailed documentation of the various classes present in the AIM simulator is given in its API documentation that can be accessed [68]. The following list gives an overview of the main packages within the AIM software.

1. *Config:* This package contains all the configuration settings required for the AIM simulator such as time step, cycles per second, vehicle spawn time, vehicle spawn distribution, vehicle arrival rate, default stop distance before intersection etc.

2. *Driver:* Implements all the driver agent functionality. Navigator, coordinator, and pilot are three important interfaces that contain the methods needed to position and navigate a vehicle, with respect to the safety constraints defined in the simulator. The AIM simulator facilitates two types of driver agents, autonomous driver (also known as auto driver) and human driver agents. Each of these driver agents use functionality implemented in the coordinator, navigator and pilot interfaces.

   Coordinator interface: controls the coordination of an auto vehicle driver agent with other vehicles and with intersection managers. The coordination process includes methods to facilitate sending various messages between vehicles and intersection managers, as well as altering the state of the driver agent.

   Navigator interface: contains methods that are used by the driver agent to choose which way the vehicle should go.

   Pilot interface: contains methods that pilots a driver agent (vehicle) autonomously.

3. *GUI:* Implements the Graphical User Interface (GUI) of the simulator. It includes the main visual area or canvas of the simulator on which the fixed and moving elements, such as roads, intersection and vehicles are drawn. In addition, the GUI package includes a **Viewer** class allowing real time user interaction with the simulator. Finally, the package includes two panels, one for simulation setup and the other for showing statistics and status of the simulator.

4. *Intersection Manager(IM):* Main classes of this package include.

   V2I manager: This class manages access requests sent by vehicles to the IM and coordinates their movement in the intersection making sure there are no collisions with other vehicles. It uses an intersection control policy for its decisions.

   Policy: This class implements the intersection management policy.

Reservation: This class accepts/rejects reservation requests made by vehicles, after running a calculation to find out if the requesting vehicle's trajectory collides with that of another vehicle present in the intersection at that time.

5. *Intersections:* This class deals with properties of the intersection, such as intersection area, roads and lanes.

6. *Map:* This package implements the map of the simulator that is used by the simulator's GUI.

7. *Messages(msg):* The msg package creates different types of messages specified by the AIM protocol; these include messages sent from vehicles to intersection managers and messages sent from intersection managers to vehicles.

8. *Simulator(sim):* The simulator package is the entry point for the AIM simulator. It launches the GUI and allows the user to set up simulation parameters and runs the simulation. It invokes a sequence of functions in order, during each time step of the simulator.

9. *Vehicle:* This package implements the vehicle model in the simulator. The vehicle model describes the vehicle's current movement (velocity, acceleration, heading, steering angle) and the vehicle's specifications (maximum acceleration, length, width, maximum steering angle, etc.)

In the next section, we describe the enhancements done in developing Enhanced-AIM based on the AIM Simulator.

4.2 Enhanced AIM Simulator - Architecture

Among the existing simulation software used for autonomous intersection management studies, only some provide a high level of customization. Most of the simulation software limits its use to only the models and features supported by the tool. The AIM simulation framework has been designed in a very modular fashion. We decided

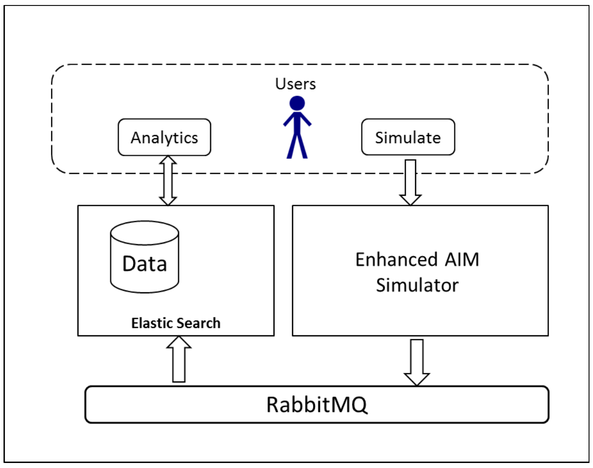


Figure 4.2: Enhanced AIM Architecture

to use the existing AIM simulation framework and make improvements to it to make Enhanced AIM extensible and easy to use. There are myriad possibilities available to make a software better. For the sake of simplicity and proof of concept, we will stick to data analysis as a primary initiative. Enhanced AIM uses a distributed messaging middleware system called RabbitMQ to log events that happen in the simulator. These events are then consumed by the mining repository and are available for users to view. The data produced from simulation can now be stored and analyzed separately and need not be tightly coupled with the simulation tool. This allows the users to perform online as well as offline data analysis.

The architecture of the Enhanced AIM simulation framework is described in this section as shown in Figure 4.2. The proposed simulation framework contains three essential elements.

1. Enhanced AIM simulator
2. Messaging middleware (RabbitMQ)
3. Mining Repository (Elastic Search)

4.2.1 Enhanced AIM Simulator

Since the adaptive routing algorithm requires communication between intersections, our first enhancement to the existing AIM simulator was to add the I2I communication module. We also added the I2I communication protocol to the existing AIM protocol. Secondly, the focus of our study was to understand how using local neighborhood information for routing vehicles in the simulator affects user experience. In order to be able to study user experience it became essential for us to log interactions between various entities within the simulator. We also wanted to be able to analyze the interactions in detail. The simulator had some built-in support for animation. We added some support for analysis. We reasoned, however, that the presence of a messaging middleware and a visualization engine specifically for analytic purposes would help us separate concerns and keep the simulator lightweight and efficient. Hence, we implemented the messaging middleware for logging events and a visualization engine specifically for analysis purposes.

Following is a list of packages that we have developed for Enhanced-AIM, and the modifications we did for some AIM-Simulator classes and methods.

- Newly added classes:
  - *I2I message:* Based on the I2I communication protocol specification we have developed a message class that creates the *status* messages sent between intersections.
  - *I2I manager:* Added the adaptive routing algorithm that calculates density and sends state information from one intersection to its connected neighbors and other helper classes.
  - *SenderQueue:* Implements the producer interface of the middleware. This class is responsible for configuring and establishing a connection between the producer and the message queue.
  - *Consumer:* This package configures and sets up the consumer.

- Existing classes modified:
  - *V2I message:* Changed the format of the request message, so that it contains the two shortest path proposals.
  - *V2I manager:* Other helper classes needed to support the functionality of the adaptive routing algorithm.
  - *Simulator:* We have made modifications to the simulator package in order to take into account the adaptive routing algorithm that will be employed after the reception of I2I messages.
  - *Intersection Manager:* We modified the IM package to be able to receive I2I messages as well as execute the adaptive routing algorithm, and to make reservation decisions based on density information as well as availability of neighboring intersections.
  - *Util:* We had to make additional modifications in the util package in order to support the necessary changes and improvements done in the simulator.

4.2.2 Messaging Middleware

Research on middleware systems has been gaining momentum over the years. One of the important advantages of a middleware system is its ability to provide seamless interoperability between various components [69]. The middleware is the glue that holds various subsystems together. This allows the programmer to focus on building standardized, adaptable and effective solutions rather than worrying about the finer details of the underlying layers [70]. A complete list of the advantages and disadvantages of using a message-oriented middleware is discussed here [71].

There are various standards and protocols for building message- oriented middleware systems. We use Advance Message Queuing Protocol (AMQP). At the time of writing, AMQP and its various open source implementations are in use in some of the most critical systems running in the world, especially in the finance industry. AMQP was developed by John O'Hara of JP Morgan Chase Inc., and is a binary wire transmission protocol. AMQP originated in the finance industry as a solution to the problem of seamlessly connecting different processing platforms together. In order to attain this effortless interoperability, AMQP boasts a well-defined, structured set of rules or behaviors for sending and receiving messages. These rules use a combination of techniques including store and forward, publish and subscribe, peer to peer, request/response, clustering, transaction management and security among many. Because of this, AMQP has become valuable for communication across various operating systems, programming platforms, integration services, and hardware devices without compromising on performance [72].

We use RabbitMQ as the messaging middleware in Enhanced AIM. In the following subsection, we will explain why we made this choice in detail.

4.2.2.1 RabbitMQ

RabbitMQ is an open source implementation of the standard AMQP 0-9-1 and is programmed in Erlang. It provides support for all major operating systems and is also available in languages such as Python, Java, Ruby and .Net. RabbitMQ is extensible and provides a number of plugins to allow communication with other web protocols such as HTTP, XMPP, SMTP and STOMP [72]. It stores messages in queues and acts as a broker between two types of processes, producers and consumers. There are two core units that form RabbitMQ, they are Queues and Exchanges/Router.

In simple terms, every message that is passed through RabbitMQ has to be placed in a queue. The main function of the router is to route the messages from the appropriate producer to the appropriate consumer. Each message consists of a simple header, specifying where it is heading to. The router doesn't read or process the message, it simply delivers the message to the appropriate queues like a letter carrier.

The producers generate messages, which are then pushed to the exchanges. The exchanges apply some routing rules on these messages and push each message to the appropriate queues, thus providing a delivery service. The messages can either be directly delivered, or they can be delivered because of an existing subscription system. The consumers, on the other hand, can either subscribe to a particular message or keep polling the queue to see if a message is received.

We chose RabbitMQ as our messaging service mainly because of two reasons.

1. It supports a standard messaging protocol (AMQP), so we are not confined by any proprietary, client or industry-specific messaging protocol.
2. All the messages are collected by the RabbitMQ. This type of message storage pattern is very similar to a push-style data flow. All the messages move from where they are produced to where they are consumed in a fluid manner, without having to periodically pull messages at various end points.

In the Enhanced-AIM simulator, we also have a common queue that stores all incoming messages to all exchanges in their order of arrival. This common queue is what the mining repository subscribes to. All operations inside RabbitMQ are done in memory. All the messages in the simulator are time-stamped and their order is maintained consistently throughout the simulation.

4.2.3 Mining Repository

A smart traffic system would greatly benefit by the presence of a mining repository that provides ongoing, live support for growing near-real time data. Such repositories are in practice now. In a nutshell, we envision that simulation and mining repository connected by an efficient messaging middleware can play a fundamental role in the advancement of automation integrated future traffic systems.

The data store or the database is used to store data generated by each simulation run. The data store is implemented using a NoSQL (Not Only Structured Query Language) database, since the data generated by each simulation run can be different and there are no complicated relationships among the data generated. Some advantages of using NoSQL over SQL(Structured Query Language) are:

- *Schema-less:* In NoSQL, columns(attributes) can be added in fly. The data can be structured, semi-structured or unstructured. This helps the simulation to cope with new attributes added to the model.

- *Better query performance:* Generally, NoSQL gives better performance over SQL when no relational queries are performed.

- *Object oriented support:* NoSQL provides the capacity to store data without relationship, therefore, the object associated with model can be directly stored with usage of ORM library (Object-relationship mapping). The NoSQL data store is a MongoDB instance [73].

We use MongoDB as the database and Elasticsearch as the search anlytics engine in Enhanced AIM. In the following subsection, we will explain why we made this choice in detail.

4.2.3.1 Search Analytics

The most important requirement of a search analytic service is the ability to access big data in near real-time, and support data growth and updates. A near real-time search engine with standardized API is its main attraction. Most databases that store large volumes of data require some sorting, filtering and other capabilities to segregate and organize that data. That way it is easy to write queries. In this case, an offline analysis is the only solution.

On the contrary, a simulator would greatly benefit by the presence of a search analytic service that provides on-going, real time support for data. For example, Elasticsearch used by organizations worldwide including Netflix, Facebook, GitHub, etc. have such characteristics. Elasticsearch can be used to perform near real-time search, data analytics, and visualization [73]. It is an open source software, and that makes it easier to integrate with any application.

In our simulator, we use MongoDB as our database storing all the events occurring during the simulation and Elasticsearch to provide support for data analytics.

MongoDB is a document oriented database, that stores object models as a document. A document is a <key,value> pair of the object attributes and its associated data. MongoDB can store huge volumes of data. We can query without performance degradation, compared to relational databases. It supports good integration with the Elasticsearch engine. Each event is time-stamped and is stored on our servers in JSON (Javascript Object Notation) format. By querying the events using the appropriate message, we can get real-time analysis. Elasticsearch provides great visualization capability with the help of Kibana [74]. Kibana provides real-time summarization and charting of data. Users can create custom graphs and visualization without the need for programming.

We use Elasticsearch in our proposed framework for the following reasons [74]:

1. Scalability: When it comes to data analytics on a massive scale, elastic search provides incredible support. Elasticsearch can be run as a single instance or multiple instances and is transparent to other services using it.

2. Visualization: Elasticsearch provides great visualization capability with the help of Kibana [74]. Kibana provides real-time summary and charting of data. Users can create custom graphs and visualization without the need for programming.

3. RESTful API: Since Elasticsearch is a RESTful server, the most widely used way to communicate with it is through its REST API. A client typically opens a connection with the Elasticsearch server, posts a JSON Object as a request and receives a JSON object as a response. This is very useful because there is no restriction on the type of client and the programming language used. Any client which can communicate with HTTP requests can communicate with the Elasticsearch server.

4.3 Summary

In this chapter, we reviewed the main features of AIM simulator. We also discussed the need for Enhanced AIM simulation framework, the modifications that were made to the AIM simulator in order to develop Enhanced AIM. In the next chapter, we will present the experiments done in order to study the effectiveness of the adaptive routing algorithm proposed in this thesis and implemented in the Enhanced AIM simulation framework.

Chapter 5

Experimental Evaluation

In this chapter, we study how the adaptive routing algorithm, employed over FCFS in the Enhanced AIM simulation framework affects congestion and wait-time in the system. FCFS policy is the AIM technique used at the intersections. The experiments are conducted in two scenarios: 1. FCFS policy without adaptive routing and 2. FCFS policy with adaptive routing. Suitable simulation experiments were conducted to study both performance as well as user experience aspects in the given two scenarios.

5.1 Simulator Background

In Enhanced AIM, every simulation step increments 0.02 simulation seconds that is displayed at the left top corner of the animation screen during simulation. This means that the simulation takes 50 steps per simulation second. Enhanced AIM uses rectangular grids as the road network. Vehicles are introduced at special locations called **spawn points** in the boundary that intersects with the roads in the network.

In each direction, vehicles are randomly spawned with a predefined probability. Once a vehicle is spawned, it is placed uniformly at random in one of the lanes traveling in that direction. If placing the vehicle in that lane and direction would cause the vehicle to be following another vehicle too closely (within 1 second or 1 meter), the vehicle is not spawned. Our simulator spawns all vehicles traveling at the speed limit and never spawns a vehicle where it would be in danger of colliding with another vehicle. We retain the assumptions made in AIM for vehicle mobility constraints and simulation setup screen controls.

While running a simulation, the user can cause an intersection failure by clicking the mouse pointer at that intersection and pressing the *Break Intersection* button for a given time period, and then make it function again by clicking the mouse pointer and pressing the *Resume Intersection* button.

For animation purposes, simulation speed is defined as the number of simulation seconds per real second. Simulation speed can be controlled using a simulation speed slider. When a simulation second is set to a real second, the animation is expected to show vehicles moving in real time. The simulation can be paused at any moment by clicking the *Pause* button to take a snapshot of the simulation. The simulation can be observed step by step using *Step* button. It can be resumed to run normally using *Resume* button. The accuracy of the animation is controlled by the frame rate slider. Since updating the simulation screen can consume lots of CPU cycles, the simulation speed and screen frame rates can be adjusted depending on the necessity (faster simulation or realistic animation).

5.2 Experimental Setup

The parameters selection and their value are partially based on the experimental setup described in [27]. Each simulation experiment is run for 3600 simulation seconds resulting in 180,000 simulation steps, considering 500 ss as the warm-up period. We set simulation seconds (ss) equal to real time second (s), which approximately corresponds to 1 hour of simulation time for each experiment including the warm-up period.

The maximum velocity of the vehicle at spawn point is set at 55 m/ss. The maximum vehicle velocity that is considered here is not reflective of realistic road simulation. We chose this velocity mainly because, in future road transport systems, fully autonomous vehicles will be able to travel at faster speeds safely. We also decided the above settings because if the vehicles in the simulation moves fast, more vehicles will be introduced in the simulation that allows us to test the efficiency of the adaptive routing algorithm. The Enhanced AIM simulator provides capability to vary the average velocity of the vehicles and perform experiments accordingly.

We use 9 intersections (3×3 rectangular grid). With 9 intersections, we have 12 spawn points from which the input traffic can flow. The traffic flow level is fixed at a corresponding flow rate of 1500 vehicles/hr/lane. As soon as each vehicle is spawned, it will pick a destination road uniformly and at random from the given map. The traveling routes between the spawn points and the destination roads are calculated by the vehicle agent periodically.

The simulation parameters are summarized in TABLE 5.1.

Table 5.1: Simulation Parameters

Parameter	Value
Total Simulation Duration	3600 ss
Warm-up Period	500 ss
Maximum Vehicle Velocity	55 m/ss
Traffic Flow	2 vehicles/lane/ss
Number of Intersections	9
Number of Lanes	2
Failure Period	500 ss if an intersection fails

We kept the input simulation parameters constant and performed the experiments multiple times. The results shown here are an average of 5 simulation runs which is then used to calculate 95% confidence interval using T value distribution.

5.3 Performance Metrics

We study the adaptive routing algorithm from two perspectives: performance and user experience.

For performance study, we study the average delay (i.e., average wait time) experienced by the vehicles at the intersections. We also compute the number of vehicles that completed their trips at any moment during the simulation and the number of vehicles in the system at any moment. The latter metric illustrates the traffic present in the system at any given time.

It is difficult to quantitatively measure user experience. As a preliminary approach, we use two basic metrics: the total number of stops that the vehicle encounters and the speed fluctuation during its trip. These two parameters, we believe, could hugely impact user experience. Experiencing more stops during a trip is certainly not a positive aspect of a travel. Similarly, speed fluctuation of the vehicle is not a comfortable travel experience. It is stated in [75] that car passengers start experiencing discomfort at lower rates of acceleration than car drivers. This discomforting experience can also be attributed equally to passengers traveling in an autonomous vehicles, if not more.

5.4 Experiments for Performance Comparison

To compare the performance of the FCFS without adaptive routing and FCFS with adaptive routing, we conducted the following experiments. All the experiments were conducted under two conditions; without intersection failure and with intersection failure. Following, we describe these experiments with results and observations one by one.

5.4.1 Wait Time Analysis

The objective of this experiment is to show how adaptive routing algorithm reduces the wait time at the intersections compared to that of FCFS without adaptive routing. In this experiment, the average wait time per vehicle at intersections is computed at different simulation time instances assuming no intersection failure condition. We calculate the wait time as the delay between receiving a confirmation of access to the intersection, and the time the vehicle actually passes through the intersection and sends a Done message. The average wait time per vehicle is shown in Fig.5.1.

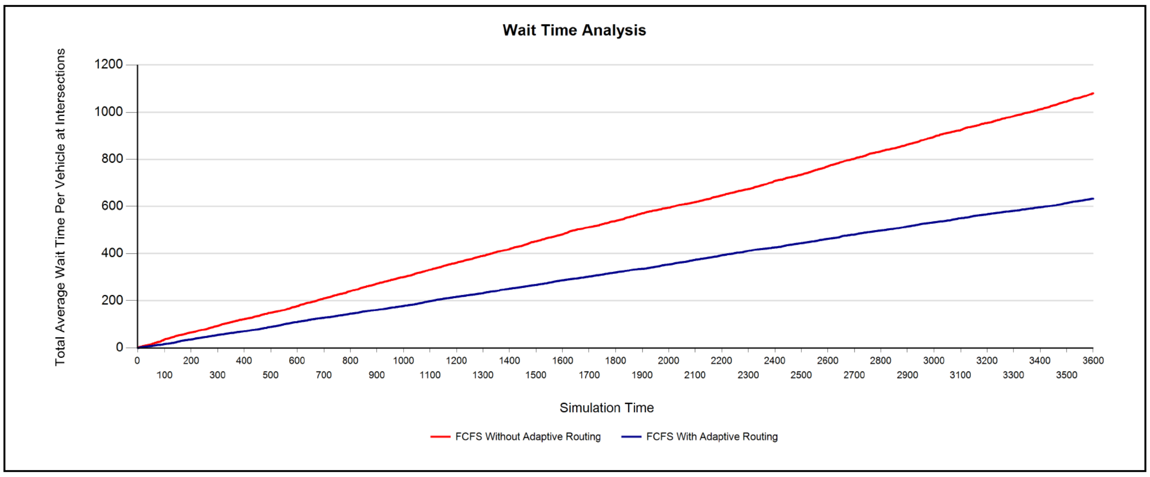


Figure 5.1: Wait Time Analysis

Observation

It is clear from the graph given in Fig. 5.1 that the average wait time per vehicle incurred in the FCFS with adaptive routing at any moment is lower than that of the FCFS without adaptive routing because of I2I messages. Also, as the time progresses the difference in average wait time per vehicle increases. Table 5.2 shows the slopes of the graphs for this experiment.

Table 5.3 shows confidence interval for this experiment.

Table 5.2: Wait Time Analysis Normal Scenario - Slopes

Normal Scenario	Slope Value
FCFS without adaptive routing	0.299
FCFS with adaptive routing	0.174

Table 5.3: Wait Time Analysis Normal Scenario using T Value

	FCFS without adaptive routing	FCFS with adaptive routing
Mean	1080	633
Variance	7056	3364
Standard Deviation	84	58
95% Confidence Interval	[975 - 1184]	[560 - 705]

5.4.2 Wait Time Analysis with Intersection Failure

The objective of this experiment is to show, in addition to the difference of wait time, how FCFS with adaptive routing algorithm handles the traffic by suitably diverting it to the neighboring intersections, compared to that of FCFS without adaptive routing algorithm. For this experiment also we compute the average wait time per vehicle at the intersections for intersection failure condition and the result is shown in Fig. 5.2.

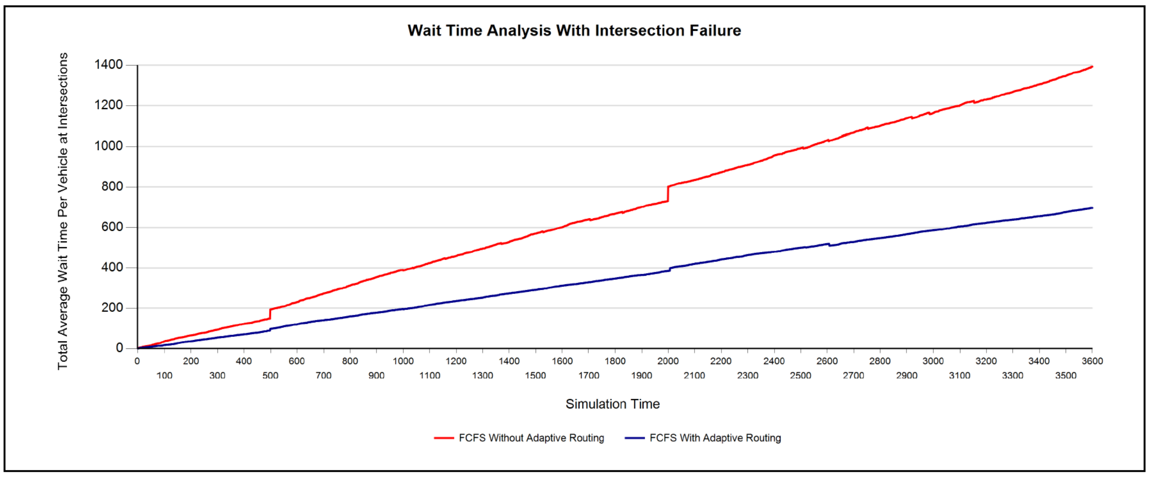


Figure 5.2: Wait Time Analysis with Intersection Failure

Observation

When an intersection fails, the average wait time per vehicle shoots up suddenly in case of FCFS without adaptive routing algorithm. In case of FCFS with adaptive routing algorithm the increase in the average wait time per vehicle is gradual and almost unnoticeable. From the graph, we can notice that there is a sudden increase in wait time at 2000ss for FCFS with adaptive routing policy. This is because at this point of time in simulation, the vehicles that are stopped in front of the broken intersections will not be allowed to pass through and only future vehicles will be re-routed. This is very important from a traffic management perspective. Since the adaptive routing algorithm diverts the traffic away from the failed intersection, the possibility of eventual traffic jam at the failed intersection is avoided. This in turn alleviates the impact in the overall system throughput. Table 5.4 shows the slopes of the graphs for this experiment.

Table 5.5 shows the confidence interval for this experiment.

Table 5.4: Wait Time Analysis Failure Scenario - Slopes

Failure Scenario	Slope Value
FCFS without adaptive routing	0.406
FCFS with adaptive routing	0.195

Table 5.5: Wait Time Analysis Failure Scenario using T Value

	FCFS without adaptive routing	FCFS with adaptive routing
Mean	1393	696
Variance	6241	3969
Standard Deviation	79	63
95% Confidence Interval	[1294 - 1491]	[617 - 774]

5.4.3 Trip Completion Analysis

The objective of this experiment is to show how FCFS with adaptive routing algorithm increases the system throughput compared to that of FCFS without adaptive routing algorithm. This experiment is performed under no intersection failure scenario. We define throughput as the number of vehicles reached their destination or trip. In this experiment, throughput is computed at different simulation time instances and the result is shown in Fig. 5.3.

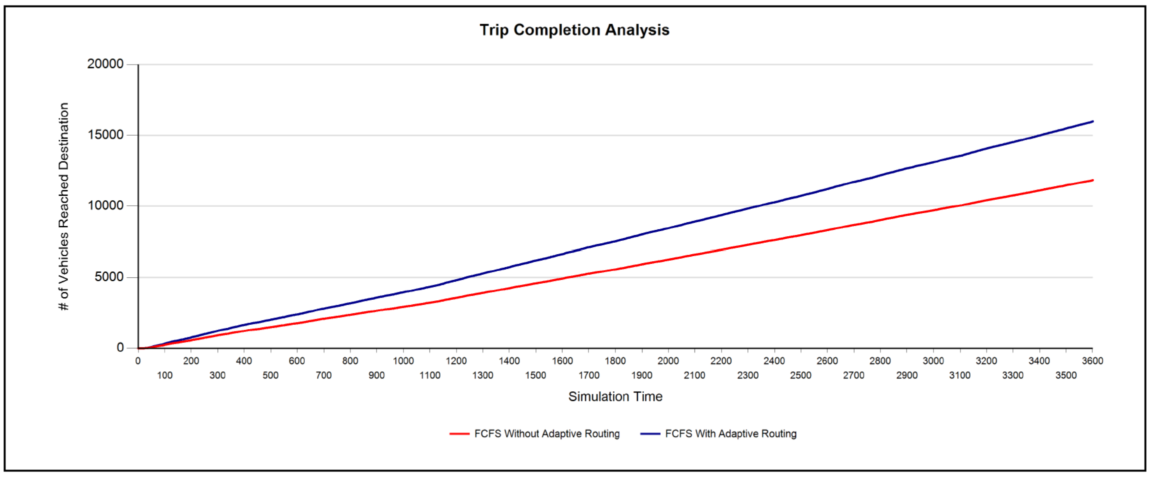


Figure 5.3: Trip Completion Analysis

Observation

It is clear from the performance graph given in Fig. 5.3 that the number of vehicles that completed the trip using FCFS with adaptive routing algorithm at any moment is higher than FCFS without adaptive routing algorithm. Also, as the time progresses the difference in system throughput increases.

Table 5.6 shows the slopes of the graphs for this experiment.

Table 5.6: Trip Completion Analysis Normal Scenario - Slopes

Normal Scenario	Slope Value
FCFS without adaptive routing	3.474
FCFS with adaptive routing	4.690

Table 5.7 shows the confidence interval for this experiment.

Table 5.7: Trip Completion Analysis Normal Scenario using T Value

	FCFS without adaptive routing	FCFS with adaptive routing
Mean	11841	15985
Variance	55225	33856
Standard Deviation	235	184
95% Confidence Interval	[11549 - 12132]	[15756 - 16213]

5.4.4 Trip Completion Analysis with Intersection Failure

The objective of this experiment is to show, how FCFS with adaptive routing algorithm handles throughput compared to that of FCFS without adaptive routing algorithm under intersection failure scenario. In this experiment, we also compute the number of vehicles that reached their destination. The results are shown in Fig. 5.4.

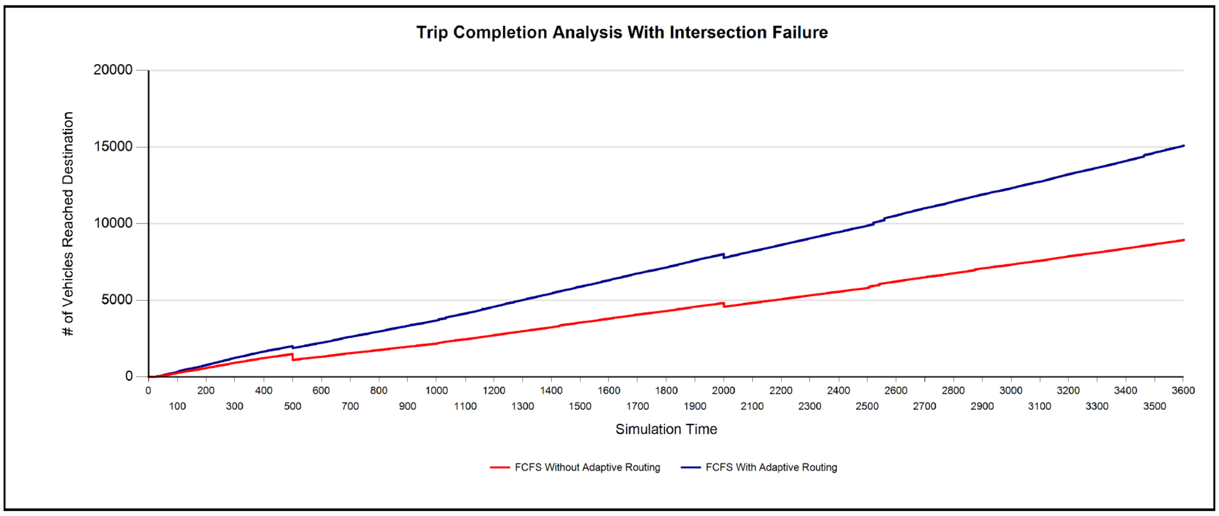


Figure 5.4: Trip Completion Analysis with Intersection Failure

Observation

When an intersection fails, the system throughput (i.e., the number of trips completed) drops suddenly in case of FCFS without adaptive routing algorithm. In case of FCFS with adaptive routing algorithm, decrease in throughput is gradual and almost unnoticeable.

Table 5.8 shows the slopes of the graphs for this experiment.

Table 5.8: Trip Completion Analysis Failure Scenario - Slopes

Failure Scenario	Slope Value
FCFS without adaptive routing	2.571
FCFS with adaptive routing	4.409

Table 5.9 shows the confidence interval for this experiment.

Table 5.9: Trip Completion Analysis Failure Scenario using T Value

	FCFS without adaptive routing	FCFS with adaptive routing
Mean	8940	15106
Variance	53361	35721
Standard Deviation	231	189
95% Confidence Interval	[8652 - 9225]	[14871 - 15340]

5.4.5 Traffic Analysis

The objective of this experiment is to show how the FCFS with adaptive routing algorithm manages the traffic (the number of vehicles) in the system compared to that of FCFS without adaptive routing algorithm. In this experiment, the number of vehicles in the system is computed at different simulation time instances for no intersection failure scenario and the result is shown in Fig. 5.5.

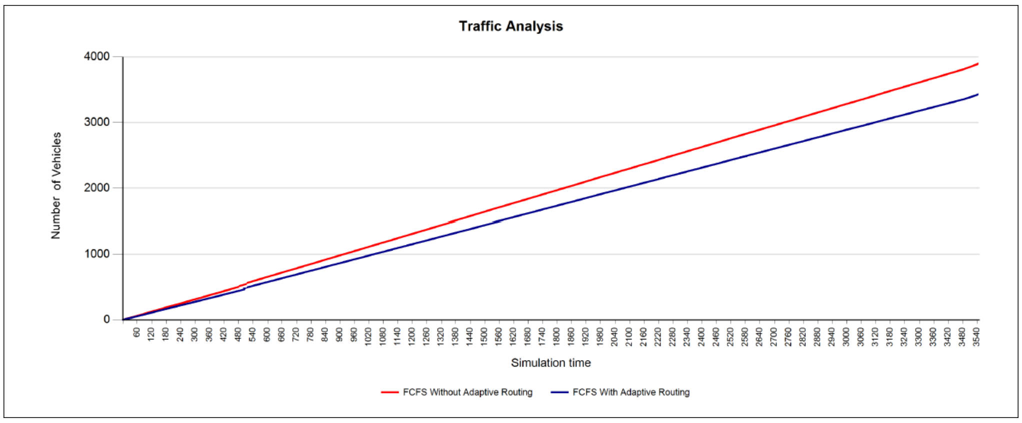


Figure 5.5: Traffic Analysis

5.4.5.1 Observation

Fig. 5.5 clearly demonstrates that the number of vehicles in simulation in case of FCFS with adaptive routing algorithm at any moment is lower than that of the FCFS without adaptive routing algorithm. Note, the slight increase at 500ss for FCFS with adaptive routing shows that the routing algorithm consistently outperforms FCFS without adaptive routing algorithm in terms of traffic in the system. Again, as the time progresses this difference in system traffic increases.

Table 5.10 shows the slopes of the graphs for this experiment.

Table 5.10: Traffic Analysis Normal Scenario - Slopes

Normal Scenario	Slope Value
FCFS without adaptive routing	1.104
FCFS with adaptive routing	0.625

Table 5.11 shows the confidence interval for this experiment.

Table 5.11: Traffic Analysis Normal Scenario using T Value

	FCFS without adaptive routing	FCFS with adaptive routing
Mean	3950	2100
Variance	7569	2209
Standard Deviation	87	47
95% Confidence Interval	[3841 - 4058]	[2041 - 2158]

5.4.6 Traffic Analysis with Intersection Failure

The objective of this experiment is to show how FCFS with adaptive routing algorithm manages the traffic (the number of vehicles) in the system, compared to that of FCFS without adaptive routing algorithm. In this experiment, the number of vehicles in the system is computed at different simulation time instances for intersection failure scenario and the result is shown in Fig. 5.6.

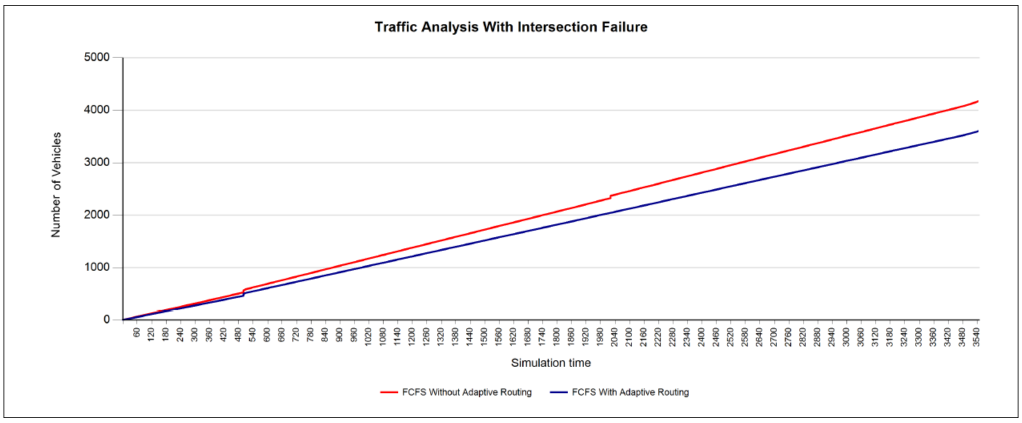


Figure 5.6: Traffic Analysis with Intersection Failure

5.4.6.1 Observation

Fig. 5.6 clearly demonstrates that the traffic in case of FCFS with adaptive routing algorithm at any moment is lower than that of the FCFS without adaptive routing algorithm. Here, the sudden slight increase in traffic around 500 ss and 2000ss for FCFS without adaptive routing is a result of more vehicles waiting in front of the failed intersection for the failure period. Overall, FCFS with adaptive routing algorithm consistently outperforms FCFS without adaptive routing algorithm in terms of traffic in the system. Again, as the time progresses this difference in system traffic increases.

Table 5.12 shows the slopes of the graphs for this experiment.

Table 5.12: Traffic Analysis Failure Scenario - Slopes

Failure Scenario	Slope Value
FCFS without adaptive routing	3.571
FCFS with adaptive routing	1

Table 5.13 shows the confidence interval for this experiment.

Table 5.13: Traffic Analysis Failure Scenario using T Value

	FCFS without adaptive routing	FCFS with adaptive routing
Mean	12072	3800
Variance	63001	6889
Standard Deviation	251	83
95% Confidence Interval	[11760 - 12383]	[3696 - 3903]

5.5 Experiments for User Experience Comparison

To compare the user experience for the two scenarios, FCFS with adaptive routing algorithm and FCFS without adaptive routing algorithm, we conducted the following experiments under two conditions: 1. no intersection failure and 2. intersection failure.

5.5.1 Trip Interruption Analysis

The objective of this experiment is to show how the number of stops experienced by a vehicle during a trip is reduced for FCFS with adaptive routing algorithm scenario compared to that of FCFS without adaptive routing algorithm scenario. This experiment is done under no intersection failure condition. Every time a vehicle's speed is 0, we count it as a stop. We compute the total number of stops. The result is shown in Fig. 5.7.

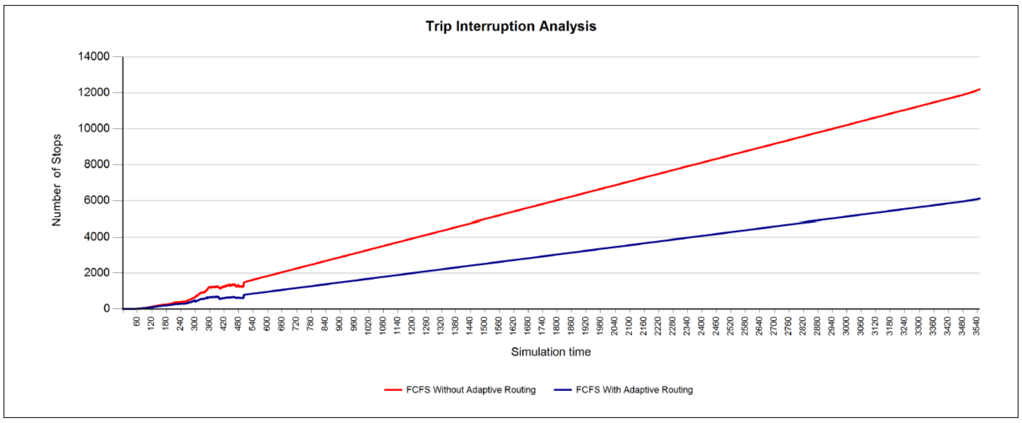


Figure 5.7: Trip Interruption Analysis

5.5.1.1 Observation

Fig. 5.7 shows that the number of stops for FCFS with adaptive routing algorithm at any moment is significantly lower than that of the FCFS without adaptive routing algorithm. At the beginning of the simulation, the intersections near the spawn point gets lots of traffic and the traffic is not equally distributed throughout the network. This might be a probable reason for the jitters observed till 500ss. As time progresses the difference in the number of stops encountered by the two scenarios increases.

Table 5.14 shows the slopes of the graphs for this experiment.

Table 5.15 shows the confidence interval for this experiment.

Table 5.14: Trip Interruption Analysis Normal Scenario - Slopes

Normal Scenario	Slope Value
FCFS without adaptive routing	3.688
FCFS with adaptive routing	1.670

Table 5.15: Trip Interruption Analysis Normal Scenario using T Value

	FCFS without adaptive routing	FCFS with adaptive routing
Mean	12500	6040
Variance	34969	11664
Standard Deviation	187	108
95% Confidence Interval	[12267 - 12732]	[5905 - 6174]

The average number of stops per vehicle for FCFS without adaptive routing policy was 0.791 and for FCFS with adaptive routing policy was 0.333.

5.5.2 Trip Interruption Analysis with Intersection Failure

The objective of this experiment is to show, in addition to the reduction in the number of stops encountered, how FCFS with adaptive routing algorithm gracefully handles the traffic by suitably diverting at the neighboring intersections compared to that of FCFS without adaptive routing algorithm. For this experiment also we compute the number of stops encountered intersection failure scenario and the result is shown in Fig. 5.8.

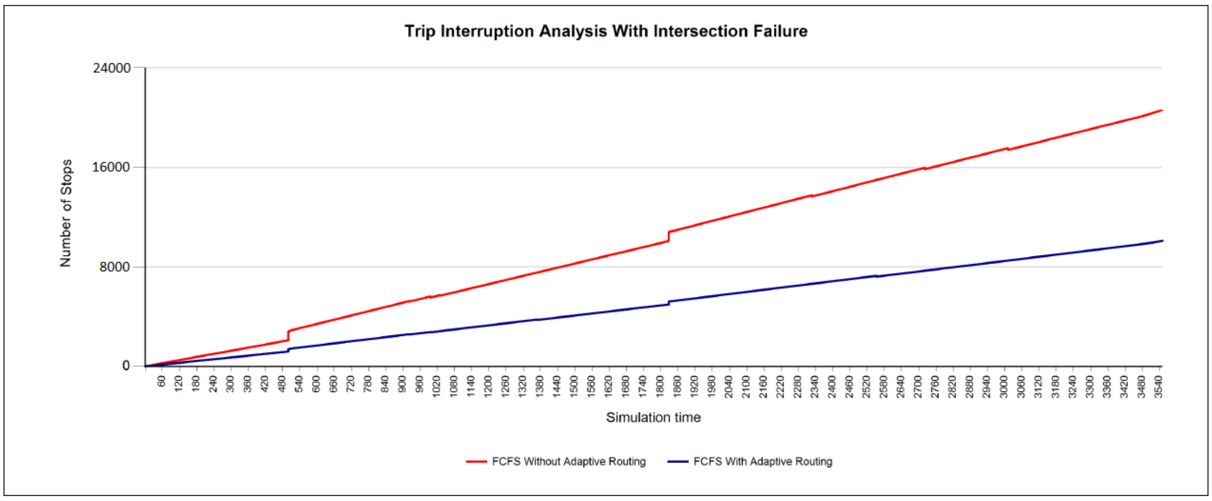


Figure 5.8: Trip Interruption Analysis with Intersection Failure

5.5.2.1 Observation

When an intersection fails, the number of stops increases significantly in case of FCFS without adaptive routing algorithm. In case of FCFS with adaptive routing algorithm, the increase in the number of stops is gradual and almost unnoticeable.

Table 5.16 shows the slopes of the graphs for this experiment.

Table 5.16: Trip Interruption Analysis Failure Scenario - Slopes

Failure Scenario	Slope Value
FCFS without adaptive routing	8.741
FCFS with adaptive routing	2.013

Table 5.17 shows the confidence interval for this experiment.

The average number of stops per vehicle for FCFS without adaptive routing policy was 1.426 and for FCFS with adaptive routing policy was 0.431.

Table 5.17: Trip Interruption Analysis Failure Scenario using T Value

	FCFS without adaptive routing	FCFS with adaptive routing
Mean	29980	8153
Variance	280900	24649
Standard Deviation	530	157
95% Confidence Interval	[29321 - 30638]	[7958 - 8347]

5.5.3 Speed Fluctuation Analysis

The objective of this experiment is to show how FCFS with adaptive routing algorithm encounters speed fluctuation compared to that of FCFS without adaptive routing algorithm. In this experiment, we measure the average speed of the vehicles currently in simulation at different simulation time instances for no intersection failure scenario, and the result is shown in Fig. 5.9.

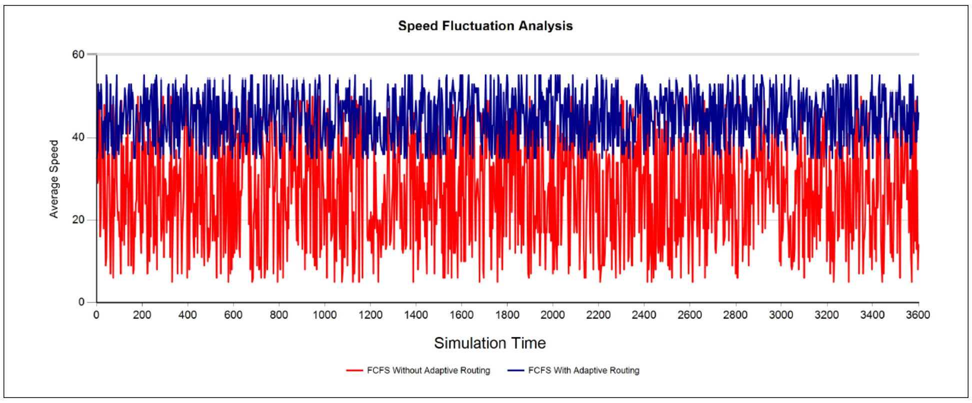


Figure 5.9: Speed Fluctuation Analysis

5.5.3.1 Observation

From Figure 5.9, it is very obvious that the average speed of the vehicles fluctuates greatly for FCFS without adaptive routing algorithm as compared to the FCFS with adaptive routing algorithm. The adaptive routing algorithm ensures all the vehicles in the simulation are evenly distributed. This results in lower delay and therefore a stable speed profile.

Table 5.18 shows the confidence interval for this experiment.

Table 5.18: Speed Fluctuation Analysis Normal Scenario using T Value

	FCFS without adaptive routing	FCFS with adaptive routing
Mean	24.8	46.8
Variance	61.62	26.01
Standard Deviation	7.85	5.1
95% Confidence Interval	[34.54 - 15.06]	[53.13 - 40.47]

5.5.4 Speed Fluctuation Analysis with Intersection Failure

The objective of this experiment is to show how FCFS with adaptive routing algorithm encounters speed fluctuation compared to that of FCFS without adaptive routing algorithm. This experiment is performed under intersection failure condition. We measure the average speed of the vehicles currently in simulation at different simulation time instances, and the result is shown in Fig. 5.10.

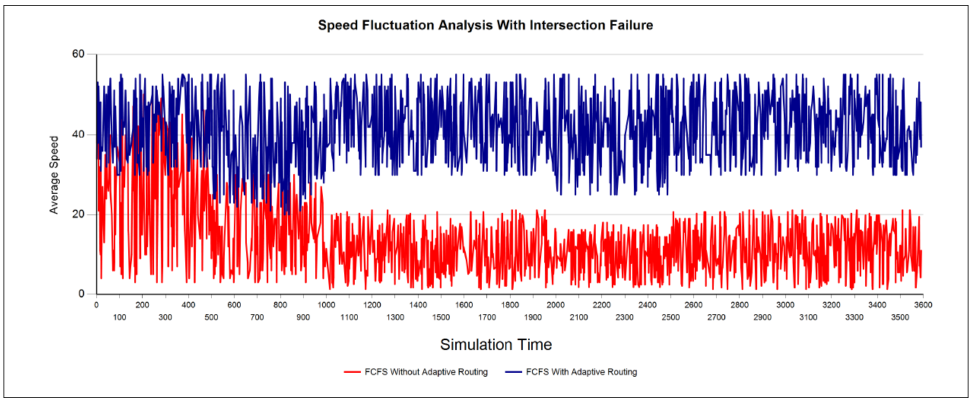


Figure 5.10: Speed Fluctuation Analysis with Intersection Failure

5.5.4.1 Observation

Figure 5.10 shows the average speed of the vehicles currently in simulation under failure scenario. We can see that during failure period, the average speed of vehicles drastically reduces to a minimum for FCFS without adaptive routing algorithm, whereas for FCFS with adaptive routing algorithm, there is a slight dip, but the speed stabilizes very quickly.

Table 5.19 shows the confidence interval for this experiment.

Table 5.19: Speed Fluctuation Analysis Failure Scenario using T Value

	FCFS without adaptive routing	FCFS with adaptive routing
Mean	17.5	42.5
Variance	66.3	57.7
Standard Deviation	8.14	7.6
95% Confidence Interval	[27.60 - 7.40]	[51.93 - 33.07]

In the next chapter, we will discuss some ethical dilemmas that needs to be addressed for furthering the research in this field.

Chapter 6

Future Outlook - Ethics

As mentioned in the introduction chapter, autonomous vehicles are more of a boon than a bane. Vehicle manufacturers are gradually adding autonomous features to present day vehicles such as assisted parking, lane following, adaptive cruise control, and even automatic overtaking. When self-driving cars become an everyday reality, they should be able to drive with a sense of responsibility. By this, I mean they are expected to replicate how human beings make decisions, or even perform better than humans in this area. Most of the decisions performed inside the brain of a self-driving car are done at an algorithmic level. Some decisions need to be quick and they can have fatal consequences. Such decisions seem to require a sense of ethics and it can be incredibly difficult to translate this into algorithms that the car can simply follow [76]. In this chapter, we explain why ethics matter for autonomous vehicles. In the following sections, we identify different ethical aspects related to autonomous vehicles.

6.1 Ethics related to Autonomous Vehicles

6.1.1 Safety

The first and the foremost issue to consider is safety. Safety of a self-driving vehicle is closely associated with how the car is programmed. More specifically, how should the car be programmed to act in the event of an unavoidable accident? Should it focus on protecting the life of pedestrians and other road users or should it just selfishly protect its passengers? The answers to these questions are important because they will have an impact on how this technology is going to be accepted in the world [77]. If people are not convinced to trust their lives to an autonomous vehicle, they would not subscribe to use them.

Let us consider a scenario. A little boy and a grandfather are crossing an intersection, a few feet apart from each other on a red light. The autonomous vehicle now is put in a sticky situation. If it swerves left, it will hit the boy. If it swerves right, it will hit the grandfather. How should the vehicle respond to this situation?

One could consider the grandfather's life less valuable compared to the little boy as he has lived a life filled with many experiences. The little boy, on the contrary, has a whole life ahead of him to live. If the car was to choose either one to kill, it would be an unethical choice. Among its many pledges, the Institute of Electrical and Electronics Engineers (IEEE), for instance, commits itself and its 430,000+ members "to treat fairly all persons and to not engage in acts of discrimination based on race, religion, gender, disability, age, national origin, sexual orientation, gender identity, or gender expression" [78]. Therefore, to treat individuals differently on the basis of their age, when age is not a relevant factor, is a discriminatory action by the car [77].

Now let us add another dimension to this scenario. What if the choice to hit was between a child and another car. This time, it would make sense then to hit the other car in such a way that impact could be minimized. This decision only makes sense if the autonomous vehicle was programmed to protect its occupants over anybody at any cost. Now, if the choice was between two vehicles, then the car should be programmed to collide with the vehicle that suffers the lowest impact [77]. This knowledge could be acquired if we have V2V communication and V2I communication in place. This strategy may be both legally and ethically better than the previous one of jealously protecting the car's own occupants. It could minimize lawsuits because any injury to others would be less severe than fatal accidents. Techniques that decide how vehicles will crash in a scenario are called crash optimization techniques.

We could possibly imagine a wide variety of scenarios similar to the ones mentioned above and then design crash optimization techniques to reduce impact and minimize loss of lives. However, it is important to note that the reaction time available for a human or a vehicle to make a decision in such a situation is very short time, a few seconds at the most. Humans in such situations operate instinctively. Autonomous vehicles are not able to operate with instinct and would be required to activate algorithms that calculate the cost of various crash options and then make a decision that would lead to minimal loss of life as well as reduced impact. This is a core ethical issue and it demands more attention than what is currently offered.

6.1.2 Security

Security is another ethical aspect that is of utmost importance for autonomous cars. One of the biggest threats that we as a society will face in the future is vehicle cyber-security. A malicious cyber attack on an autonomous vehicle has not yet happened. But the possibility of such an attack was demonstrated in 2015 when hackers took control of a Jeep Cherokee and cut its transmission on the highway [79]. This incident resulted in Chrysler recalling 1.4 million vehicles in order to update security features on these vehicles [79].

Cars today have advanced electronic control units (ECU) that are connected to an internal network. If a hacker manages to take control of an ECU, they might easily be able to take control of other safety-critical ECUs as well. A further complication is that these ECUs are programmed by third party vendors and not the auto manufacturers themselves. That means too many people have access to safety-critical code. Car hacking is a real threat at the present time. In 2014, more than half of the vehicles sold in the United States were connected, meaning that they are vulnerable to cyber attacks [79]. Most of the vehicles with advanced sensor features are prone to hacking. The threat will loom larger as our society transitions to autonomous vehicles.

To avoid this, vehicle cyber-security must be addressed at multiple layers similar to network security. At the lowermost layer, ECUs must be protected especially those ECUs in charge of safety critical applications such as braking, transmission, etc. Above that, security software that can protect the vehicle's internal network as a whole must be installed. In addition, we need software that will encrypt any network communication that arises from the vehicle and vice versa. Next, it is very critical to secure solutions that deal particularly with in-car entertainment, such as Apple Car Play or Android Auto as these applications connect the car network to the internet. Finally, cloud security services can detect and correct threats before they reach the vehicle. They can also send the vehicle over-the-air updates and intelligence in real time.

In addition to software protection, auto manufacturers must pay attention to secure hardware components and make sure they only purchase ECUs from trusted vendors. Society is often reactive rather than proactive with security issues, adopting serious preventive measures only after a major incident has occurred. A number of OEMs are currently undertaking proactive measures including Tesla, Fiat Chrysler and GM. These manufacturers have recently established "bug bounty" programs to reward individuals that find and report security flaws in their cars' software, an effort to further fortify their systems against vulnerabilities [80].

6.1.3 Privacy

Self-driving cars collect and transmit lots of data. They are periodically storing information related to the environment they are in, they communicate with other cars and roadside infrastructure. As the autonomous vehicle collects more data, more concerns are raised regarding its protection [81]. Even data from a single sensor in the car could invade privacy of an individual if it is collected without consent. The broader questions that need to be answered by auto manufacturers surrounding this issue include but are not limited to the following:

- How much data is the car supposed to collect for learning purposes? How much data is too much data?
- Who owns this data?
- Who has access to this data?
- Where is this data stored? Is it encrypted before storage? Does it reveal personal identity information?
- When will this data be destroyed?
- How much data is actually used for evaluation? Is it anonymous?
- Does it contain more data than "just" the position and travel details of the autonomous vehicle?
- Can it be connected to other types of data like a phone number, a bank account, credit cards, personal details, or health data?
- If distributing data to other external services such as traffic information or navigation data, which are used in the calculation of the route, how trustworthy are those data sources (e.g., GPS, map data, external devices, other vehicles)?

Research on data privacy related to autonomous vehicles is in its early stage. It is important to accelerate research in this area at the development stage of this technology. This way, we can regulate industry as well as guide research practices before any intentional or unintentional privacy invasion becomes a part of the deployed technology. Data collected solely for safety and navigation purposes must be well regulated. Passengers must be able to opt out of atleaset some parts of the data that is being collected to enhance their travel experience if they feel they are being tracked [82].

6.1.4 Social and Economic Challenges

Driverless vehicles will totally change the traditional car ownership model. At the beginning of Chapter 2, we note that cars presently in the world remain parked almost 95% of the time. Driver-less cars may come in the form of cost-effective transport models or taxi services. This will motivate people to use taxi services rather than owning personal vehicles. This will, in turn, affect land usage in a positive way. Driverless cars can be summoned from anywhere and need not be parked in close proximity to the users. The need for parking garages will rise but the land that is being used for parking lots now could be transformed into a green belt affecting the environment in a positive way. Beyond economic benefits, driverless cars also affect society in a broad way making access to cars very inclusive. People would not need to own driving licenses in order to get around from one place to another.

6.2 Ethics related to Intersection Infrastructure

In this section, we will discuss the ethical questions that need to be considered before deploying road side infrastructure for autonomous vehicles, especially at intersections. Inside the intersection, vehicles move at high speeds in all different directions. Therefore, we need to address safety-critical questions such as, "What happens when a driver agent realizes that it will not make its reservation exactly on time but is close enough to the intersection that it is not possible to stop before entering the intersection?

In such a case, it is imperative to have some sort of a safety buffer. In a simulation setting, two types of buffers are most natural: static buffers and time buffers. Static buffers have constant sizes for safety purposes. Before approving a reservation, the

IM makes sure no two vehicles' static buffers intersect with each other. It is very important to make sure the size of the static buffer around the vehicles is not too large, otherwise, two vehicles that potentially would not have resulted in a collision will not be granted access to their reservation, simply because their large static buffers intersect [27].

Time buffers, on the other hand, do take into account the motion of the vehicles. If the intersection manager assumes that the vehicle might be early or late, the actual area restricted by this buffer will shrink or grow with the vehicle's velocity. Thus, if two vehicles are traveling along parallel lines, the time buffers for those vehicles should not interfere unless those vehicles could potentially collide (they are in the same lane or the lanes are too close together for the vehicles' width) [27]. Time buffers alone do not sufficiently guarantee safety. A small error in the direction of the motion of the vehicles may still cause a collision.

A more practical and a safe solution would be a hybrid buffer. A hybrid buffer consists of a time buffer that scales with velocity, as well as a small static buffer that protects against lateral positioning errors. It also serves as a minimum buffer for slow-moving vehicles or heavy vehicles. Detailed explanation of the implementation details of these buffers in the AIM simulator is given in [27].

6.3 Compatibility With Human Drivers

While AIM techniques for autonomous vehicles will someday be a reality, it is more practical to assume there will always be people who enjoy driving for the foreseeable future. The transition period from human drivers to fully autonomous vehicles will include a few years where self-driving cars would have to share the road with human drivers, pedestrians, and cyclists. This can be employed by having dedicated lanes for self driving cars or restricting the time or hours of the day when self driving cars should be on the road. Either way, we need to be able to support AIM policies that are able to accommodate human drivers or pedestrians/cyclists.

An extended work of the AIM simulator focuses on this. The authors in [83] proposed an AIM protocol that works with human drivers. In order to accommodate human drivers, the AIM policy in effect must be able to direct both human and autonomous vehicles, while coordinating them, despite having much less control and information about where and when the human drivers will be.

The simplest and best solution is to use something human drivers already know and understand —traffic lights. There are protocols like FCFS-Light that demonstrate how AIM policy works in conjunction with human drivers. The human drivers can be simulated in ALL the lanes or in a single lane based on what we want to measure.

6.3.1 Emergency Vehicles

In order to accommodate emergency vehicles, the intersection manager must first be able to detect their presence. The easiest way to accomplish this is to add a new field to all request messages. This indicates to the intersection manager that the requesting vehicle is an emergency vehicle in an emergency situation (lights flashing and siren blaring). Implementing emergency vehicles is out of the scope for this thesis.

Now that the intersection manager can detect emergency vehicles, it can process reservation requests while giving priority to the emergency vehicles. To do this the IM keeps track of which lanes currently contain approaching emergency vehicles. As long as at least one emergency vehicle is approaching the intersection, the policy only grants reservations to vehicles in those lanes. This ensures that vehicles in front of the emergency vehicles will also receive priority. Due to this increase in priority, lanes with emergency vehicles tend to empty very rapidly, allowing emergency vehicles to proceed relatively unhindered.

6.3.2 Remote to Urban Scenario Transition

Finally, we discuss the factors that need to be considered when autonomous vehicles travel over long distances and switch between remote and urban scenarios.

When dealing with rural scenarios, we do not have networked intersections for very long stretches. Therefore for most of a rural setting, we are dealing with highway driving. Highway driving, platooning, and ramp metering of autonomous vehicles is more or less a solved problem. AVs on the highway need to make sure they have safe following distances from other vehicles. They also need to overtake carefully. An urban setup, however, is a more complex environment with many opportunities for distraction.

With current highway infrastructure in place, AVs may face a problem in establishing communication with intersections in small towns that occur in the bypass. There is not enough research available to address this scenario in a satisfactory manner. The safety and security aspects of this transition from a remote to urban scenario and vice versa of AVs is a potential research problem for the future.

6.4 Summary

No complex technology we have created has been infallible. And just about every computing device we have created has been hacked before or can be hacked in the future, including neural implants and military systems [81]. When it comes to self driving cars, software errors, malfunctioned sensors, cyber attacks, and bad weather can contribute to collisions. If ethics are ignored and the car behaves badly, a powerful case could be made that auto manufacturers were negligent in the design of their product, and that opens them up to tremendous legal liability, should such an event occur.

In my opinion, there are two sides to this challenge. On the one hand, the government, transportation agencies, regulatory bodies, safety boards and automotive manufacturers need to work together in creating and maintaining an ethical framework where decisions made by autonomous vehicles can be evaluated. It is important, therefore, to think through ethical dilemmas along with every innovative outburst in this field. On the other side, auto manufacturers need to prepare the future road users before they find themselves surprised in bad ways by autonomous cars. Auto manufacturers should not just be focused on technology penetration rate and market acceptance rate of the vehicles but also communicate to the general public how this might positively or negatively impact their life. Whatever answer to an ethical dilemma that industry might lean towards will not be satisfying to everyone. Ethics and expectations are challenges common to all automotive manufacturers and tier-one suppliers who want to play in this emerging field, not just particular popular companies.

Automated cars promise great benefits and unintended effects that are difficult to predict, and the technology is coming either way. Change is inescapable and not necessarily a bad thing in itself. But major disruptions and new harms should be anticipated and avoided where possible. That is the role of ethics in innovation policy: it can pave the way for a better future while enabling beneficial technologies. Without looking at ethics, we are driving with one eye closed [76].

Chapter 7

Conclusion And Future Work

With the recent advances in artificial intelligence and Internet of Things, road transportation going forward has a greater expectation to include autonomous vehicles. An important component of urban road transportation infrastructure is intersections. A lot of work has been done towards development of coordination techniques to enable safe and collision-free passage of vehicles through intersections both in industry as well as the research community. This is called Autonomous Intersection Management (AIM). Current AIM techniques proposed in the literature will work efficiently for a single intersection. In real life, however, urban roads are comprised of a network of multiple intersections. The intersections are not connected with each other and have no information about how congested the rest of the road network is. In such a case, achieving optimal throughput of vehicles inside an intersection alone does not guarantee an overall reduced congestion outcome in the network.

In this thesis, we focus on how to effectively and adaptively route autonomous vehicles in a road network based on local neighborhood information. To achieve this, we designed and developed an adaptive routing algorithm that uses intersection-to-intersection (I2I) communications. I2I communication allow intersections to communicate with their connected, neighboring intersections, and route traffic based on the density information. To implement and study this algorithm, we constructed the

Enhanced AIM simulation framework with an added I2I communication module by extending the AIM simulator. We also added a messaging middleware and a user-friendly visualization engine for message logging and data analysis as a part of the framework. By adding the I2I communication module and the routing algorithm to the Enhanced AIM simulator, we are able route traffic in an adaptive manner regularly as well as in emergency scenarios such as intersection failure. We also conducted simulation experiments that demonstrate a reduction in congestion and wait-time. The experiments also show a better user experience achieved when adaptive routing is enabled in the system compared to a scenario without adaptive routing invoked.

Future Work

There are multiple ways in which the research done for this thesis can be extended. Some important ones are listed below:

1. Currently, Enhanced AIM simulation framework does not take into account the presence of V2V messages. With V2V messages present, we can disperse density information from the intersection to the vehicles periodically so that they can adjust their own goals based on the congestion condition in the network. The current state of the simulator has vehicles that have no knowledge about how congested the road network is.

2. Adaptive routing algorithm has only been tested over FCFS policy for autonomous intersection management. We could test and study the effectiveness of the adaptive routing algorithm with other policies like priority based, auction based etc.

3. The middleware currently serves as an interface between the simulator and the analytic engine. The middleware can be rewritten to act as an interface between the various agents in the simulator. This enhancement to the middleware would also allow testing the simulation system with a real autonomous car, where a self-driving car's movements can be monitored and controlled from within the

simulation environment in real time.

Appendix

A.1 Communication Protocol Specification

This section is divided into two parts: The first part explains some of the auxiliary actions taken up by by both vehicle and intersection agent. The second part explains the various types of messages used by both vehicle and intersection agent.

A.1.1 Protocol Actions

Both vehicle and intersection agent must follow a set of rules. We call these set of rules as protocol actions [27].

A.1.1.1 Vehicle Actions

These are the rules that the vehicles are expected to follow in order to allow the intersection to function efficiently.

- A vehicle is not supposed to enter the intersection if it does not have a reservation.
- When a vehicle receives a *Confirm* message, it is considered to have a reservation.
- If a vehicle is passing through the intersection, it must follow the parameters included in the most recent *Confirm* message received from the intersection.
- If another *Request* message is sent by the vehicle before the intersection manager has sent a response to the previous one, the intersection manager may choose to ignore it.
- If a vehicle has not yet entered the intersection and does not have a reservation, it may send a *Request* message. If it has not yet entered the intersection and does have a reservation, it may send either a *Change-Request* or a *Cancel* message.

- When a vehicle has successfully passed through the intersection, it sends a *Done* message.

A.1.1.2 Intersection Actions

These are the rules that the intersection manager has to follow [52].

- When an intersection manager receives a *Request* message, it must respond with either a *Confirm* or a *Reject* message. If it responds with a *Confirm* message, the intersection manager should make sure that the requesting vehicle has no conflicting trajectories with the other vehicles occupying the intersection space at that time.
- When an intersection manager receives a *Change-Request* message, it must respond with either a *Confirm* or a *Reject* message.
- When an intersection manager receives a *Cancel* message, it must respond with an *Acknowledge* message.

A.1.2 Types of Messages

The vehicles and intersection manager are each restricted to a few types of messages with which they must coordinate.

A.1.2.1 Vehicle to Intersection

There are four types of messages that can be sent from a vehicle to the intersection manager.

1. *Request*: The *Request* message contains an array of proposals with the following fields:

   Proposal id: a unique identifier for each route proposal.

   vehicle id: a unique identifier for the vehicle.

arrival time: the time at which the vehicle will arrive at the intersection.

arrival lane: an id for the lane in which the vehicle will be when it arrives at the intersection.

turn: the vehicle's turn when it reaches the intersection.

arrival velocity: the velocity at which the vehicle is traveling when it arrives at the intersection.

maximum velocity: the maximum velocity at which the vehicle can travel.

maximum acceleration: the maximum rate at which the vehicle can accelerate.

minimum acceleration: the minimum rate at which the vehicle can accelerate (i.e. negative number representing maximum deceleration).

vehicle length: the length of the vehicle.

vehicle width : the width of the vehicle.

front wheel displacement: the distance between the front of the vehicle and the front axle.

rear wheel displacement: the distance between the front of the vehicle and the rear axle.

max steering angle: the maximum steering angle of the front wheels.

max turn per second: the rate at which the vehicle can turn its front wheels.

emergency: whether or not this is an emergency vehicle in an emergency situation.

2. *Change-Request*: This is the message a vehicle sends when it has a reservation, but would like to switch to a different set of parameters. If the new parameters are not acceptable to the intersection, the vehicle may keep its old reservation. It is identical to the Request message, except that it includes a unique reservation ID for the reservation the vehicle currently has. This message is identical to the *Request* message, except for one added field:

reservation id: an identifier for the reservation to be changed.

3. *Cancel*: This is the message a vehicle sends when it no longer desires its current reservation. It has 2 fields:

   vehicle id: a unique identifier for the vehicle.

   reservation id: an identifier for the reservation to be canceled.

4. *Done*: This message is sent after a vehicle passed through an intersection space. This message is mainly for analysis purposes. It has 2 fields:

   vehicle id : a unique identifier for the vehicle.

   reservation id : an identifier for the reservation that was just completed.

A.1.2.2 Intersection to Vehicle

There are three types of messages that can be sent by the intersection manager to a vehicle.

1. *Confirm*: This message is a response to a vehicle's *Request* (or *Change-Request*) message. This message has 7 fields [27]:

   reservation id: a unique identifier for the reservation just created.

   arrival time: the time at which the vehicle is expected to arrive at the intersection.

   early error: the tolerable error if the vehicle arrives earlier than the estimated arrival.

   late error: the tolerable error if the vehicle arrives later than the estimated arrival time.

   arrival lane: an id for the lane in which the vehicle will be when it arrives at the intersection.

   arrival velocity: the traveling velocity of the vehicle at arrival.

accelerations: the expected acceleration of the vehicle as it travels through the intersection.

2. *Reject*: By sending this message, an intersection can inform a vehicle that the parameters sent in the latest *Request* (or *Change-Request*) were not acceptable. This message also indicates whether or not the rejection was because the reservation manager requires the vehicle to stop at the intersection before entering. This lets the driver agent know that it should not attempt any more reservations until it reaches the intersection. This message has one field:

   stop required: a boolean value indicating whether the vehicle must first come to a full stop before entering the intersection.

3. *Acknowledge*: This message acknowledges the receipt of a *Cancel* or *Done* message. It has one field:

   reservation id: a unique identifier for the reservation just canceled or completed.

   Emergency-Stop: This message is only sent when the intersection manager has determined that a collision or similar problem has occurred in the intersection. This message informs the receiving driver agent that no further reservation requests will be granted, and if possible, the vehicle should attempt to stop instead of entering the intersection, even if it has a reservation. The specifics of how this message is used are out of scope of this thesis.

A.1.2.3 Intersection to Intersection

We implemented the Intersection to Intersection (I2I) messages in order to make local neighborhood knowledge available. At each update interval, the current intersection manager receives the traffic density information from each of the neighbor intersection managers. We can consider this as an exchange of state information from one intersection manager to its neighbors at each interval. The intersection manager now has the capability to make a knowledge based routing decision in real time with the state information from the neighboring intersection managers. The communication of each intersection agent is limited only within its connected neighborhood. The broadcast communication with the User Datagram Protocol (UDP) is applied for I2I. The information will transmit and receive asynchronously every update interval. We set the update interval as 0.1 s for our experiments. The following message is sent from one intersection to another intersection. They are:

1. *Status*: This message contains the state of an intersection. This message has 3 fields:

   IM_Id: ID of the sending intersection manager

   incoming street density: this is the amount of vehicles present on the incoming street.

   outgoing street density: this is the amount of vehicles present on the outgoing street.

A.1.3 Message Corruption and Loss

The AIM protocol assumes that messages are digitally signed. In such case, the possibility of message corruption is very small. AIM protocol is specifically designed to be robust to message loss [25]. In this thesis, we have built the I2I module and the communication protocol using AIM protocol's standards. If a message is sent but not received, the worst possible event then to happen would be additional delay. No collisions can occur due to lost messages. When a vehicle makes a reservation request, it does not assume the space is reserved until it receives a confirmation from the intersection manager. If a Request message is dropped, no Confirm message will follow. If a Confirm or Reject message is dropped, the vehicle will simply try again —it won't assume that it has a valid reservation.

Results of Experiment 2

Bibliography

[1] http://www.reinventingparking.org/2013/02/cars-are-parked-95-of-time-lets-check.html.

[2] Alysson Shontell. The future of self-driving cars - An interview with Chris Dixon.

[3] David Schrank, Bill Eisele, Tim Lomax, and Jim Bak. 2015 Urban Mobility Scorecard. Texas A&M Transportation Institute, 39:5, August 2015.

[4] Graham Cookson and Bob Pishue. INRIX Global Traffic Scorecard. Inrix Global Traffic Scorecard, February 2017.

[5] National agenda for intersection safety. Technical report, US Department of Transportation - Federal Highway Administration, Transportation Officials Campus, Madison.

[6] Li Junce and Zhang Huazhong. Study on optimal control and simulation for urban traffic based on fuzzy logic. In 2008 International Conference on Intelligent Computation Technology and Automation (ICICTA), volume 1, pages 936–940, Oct 2008.

[7] Noah Goodall, Brian Smith, and Byungkyu Park. Traffic signal control with connected vehicles. Transportation Research Record: Journal of the Transportation Research Board, 2381:65–72, 2013.

[8] Intelligent Car Initiative. Technical report, European Commission Information Society and Media, 2010.

[9] Daniel J Fagnant and Kara Kockelman. Preparing a nation for autonomous vehicles: opportunities, barriers and policy recommendations. Transportation Research Part A: Policy and Practice, 77:167–181, 2015.

[10] National Highway Traffic Safety Administration Preliminary Statement of Policy Concerning Automated Vehicles. Technical report, NHTSA, 2013.

[11] José L. F. Pereira and Rosaldo J. F. Rossetti. An integrated architecture for autonomous vehicles simulation. Proceedings of the 27th Annual ACM Symposium on Applied Computing - SAC '12, pages 286–292, June 2012.

[12] Miguel Cordeiro Figueiredo. An Approach to Simulation of Autonomous Vehicles in Intens
PhD thesis, 2009.

[13] Chris Urmson, Joshua Anhalt, Drew Bagnell, Christophe Baker, Robert Bittner, M. N. Clark, John Dolan, Dave Duggins, Tugrul Galatali, Chris Geyer, Michele Gittleman, Sam Harbaugh, Martial Hebert, Thomas M. Howard, Sascha Kolski, Alonzo Kelly, Maxim Likhachev, Matt McNaughton, Nick Miller, Kevin Peterson, Brian Pilnick, Raj Rajkumar, Paul Rybski, Bryan Salesky, Young-Woo Seo, Sanjiv Singh, Jarrod Snider, Anthony Stentz, William "Red" Whittaker, Ziv Wolkowicki, Jason Ziglar, Hong Bae, Thomas Brown, Daniel Demitrish, Bakhtiar Litkouhi, Jim Nickolaou, Varsha Sadekar, Wende Zhang, Joshua Struble, Michael Taylor, Michael Darms, and Dave Ferguson. Autonomous driving in urban environments: Boss and the urban challenge. Journal of Field Robotics, 25(8):425–466, 2008.

[14] SÃűren Kammel, Julius Ziegler, Benjamin Pitzer, Moritz Werling, Tobias Gindele, Daniel Jagzent, Joachim SchrÃűder, Michael Thuy, Matthias Goebl, Felix von Hundelshausen, Oliver Pink, Christian Frese, and Christoph Stiller. Team annieway's autonomous system for the 2007 darpa urban challenge. Journal of Field Robotics, 25(9):615–639, 2008.

[15] Andreas Geiger, Martin Lauer, Frank Moosmann, Benjaminm Ranft, Holger Rapp, Christoph Stiller, and Julius Ziegler. Team annieway's entry to the

2011 grand cooperative driving challenge. IEEE Transactions on Intelligent Transportation Systems, 13(3):1008–1017, Sept 2012.

[16] vislab. Accessible at: vislab.it/proud/, 2013.

[17] Junqing Wei, Jarrod M. Snider, Junsung Kim, John M. Dolan, Raj Rajkumar, and Bakhtiar Litkouhi. Towards a viable autonomous driving research platform. In Intelligent Vehicles Symposium, pages 763–770. IEEE, 2013.

[18] Vladyslav Novak. Google self-driving car. PhD thesis, 2016.

[19] Mica R. Endsley. Autonomous driving systems: A preliminary naturalistic study of the tesla model s. Journal of Cognitive Engineering and Decision Making, 2017.

[20] Andrew Hawkins. Tesla's autopilot is supposed to deliver full self-driving, so why does it feel stuck in the past? The Verge accessible at: =https://www.theverge.com/2017/10/24/16504038/tesla-autopilot-self-driving-update-elon-musk, 2017.

[21] Mark Bergen. Alphabet launches the first taxi service with no human drivers. Bloomberg Technology accessible at: https://bloom.bg/2Ea2dml, 2017.

[22] Fully driverless cars could be months away. google's self-driving car unit prepares to launch a taxi service near phoenix. ArsTechnica accessible at: http://bit.ly/2DkP3FO.

[23] Timothy Lee. Prepare to be underwhelmed by 2021's autonomous cars: Ford, uber, and bmw promise fully self-driving cars in five years - but they will probably only work in very limited areas. MIT Technology Review accessible at: https://www.technologyreview.com/s/602210/prepare-to-be-underwhelmed-by-2021s-autonomous-cars/, 2016. Accessed: 2018-05-03.

[24] Alexander Hars. Autonomous cars: The next revolution looms. Thinking outside the box: Inventivio Innovation Briefs, page 4, January 2010.

[25] Kurt Dresner and Peter Stone. Making Autonomous Intersection Management Backwards-Compatible. Association for the Advancement of Artificial Intelligence, pages 1865–1866, 2006.

[26] David Fajardo, Tsz-Chiu Au, S. Waller, Peter Stone, and David Yang. Automated intersection control: Performance of future innovation versus current traffic signal control. Transportation Research Record: Journal of the Transportation Research Board, (2259):223–232, 2011.

[27] Kurt Dresner and Peter Stone. A Multiagent Approach to Autonomous Intersection Management. Journal of Artificial Intelligence Research, 31:591–653, 2008.

[28] Reinventing parking. Accessible at: `http://www.reinventingparking.org/2013/02/cars-are-parked-95-of-time-lets-check.html`.

[29] Luciano Bononi, Marco Di Felice, Gabriele D'Angelo, Michele Bracuto, and Lorenzo Donatiello. Moves: A framework for parallel and distributed simulation of wireless vehicular ad hoc networks. Computer Networks, 52(1):155–179, January 2008.

[30] Pawel Gora and Inga Rüb. Traffic models for self-driving connected cars. Transportation Research Procedia, 14(14):2207–2216, 2016.

[31] Serge P. Hoogendoorn and P. H. L. Bovy. State-of-the-art of vehicular traffic flow modelling. Proceedings of the Institution of Mechanical Engineers, Part I: Journal of Systems and Control Engineering, 215(4):283–303, 2001.

[32] Douglas Reece and Steven Shafer. A computational model of driving for autonomous vehicles. Technical Report CMU-CS-91-122, Pittsburgh, PA, April 1991.

[33] Jiajun Hu, Linghe Kong, Wei Shu, and Min-You Wu. Scheduling of connected autonomous vehicles on highway lanes. In 2012 IEEE Global Communications Conference (GLOBECOM), pages 5556–5561, Dec 2012.

[34] Yaser Chaaban. A methodology for designing robust central/self-organising multi-agent systems. 2013.

[35] Thomas Schulze and Thomas Fliess. Urban traffic simulation with psycho-physical vehicle-following models. In Proceedings of the 29th conference on Winter simulation, pages 1222–1229. IEEE Computer Society, 1997.

[36] Praveen Paruchuri, Alok Reddy Pullalarevu, and Kamalakar Karlapalem. Multi agent simulation of unorganized traffic. In Proceedings of the first international joint conference on Autonomous agents and multiagent systems: part 1, pages 176–183. ACM, 2002.

[37] Karima Benhamza, Salah Ellagoune, Hamid Seridi, and Herman Akdag. Agent-Based Modeling for Traffic Simulation. pages 51–56, 2012.

[38] Chairit Wuthishuwong and Ansgar Traechtler. Consensus-based local information coordination for the networked control of the autonomous intersection management. Complex & Intelligent Systems, 3(1):17–32, 2017.

[39] Dirk Helbing. Traffic and related self-driven many-particle systems. Reviews of modern physics, 73(4):1067, 2001.

[40] Sven Maerivoet and Bart De Moor. Cellular automata models of road traffic. Physics Reports, 419(1):1–64, 2005.

[41] Kai Nagel and Michael Schreckenberg. A cellular automaton model for freeway traffic. Journal de physique I, 2(12):2221–2229, 1992.

[42] Bastien Chopard, Pascal O. Luthi, and Pierre-Antoine Queloz. Cellular automata model of car traffic in a two-dimensional street network. Journal of Physics A: Mathematical and General, 29(10):2325, 1996.

[43] Monique R. Evans and Debra S. Elston. A Primer for Agent-Based Simulation and Modeling in Transportation Applications. Technical report, University of Arizona, Tuscon, 2013.

[44] Ru Cicortas and Norbert Somosi. Multi-agent system model for urban traffic simulation, 2005.

[45] A S Van Leeuwen. Static Traffic Assignment with Queuing. PhD thesis, 2011.

[46] Martin Fellendorf and Peter Vortisch. Fundamentals of Traffic Simulation, volume 145. 2010.

[47] Perarnau Josep Casas, Jordi and Alex Torday. The need to combine different traffic modelling levels for effectively tackling large-scale projects adding a hybrid meso/micro approach. Procedia - Social and Behavioral Sciences, 20:251 – 262, 2011.

[48] Mustapha Saidallah, Abdeslam El Fergougui, and Abdelbaki Elbelrhiti Elalaoui. A Comparative Study of Urban Road Traffic Simulators. MATEC Web of Conferences, 81(International Conference on Transportation and Traffic Engineering [ICTTE 2016]), 2016.

[49] Andre Maia Pereira, Hossam Anany, Ondrej Pribyl, Prikryl, and Jan. Automated Vehicles in Smart Urban Enviornment: A Review. In Smart Cities Symposium, Prague, 2017.

[50] Dustin Carlino, Mike Depinet, Piyush Khandelwal, and Peter Stone. Approximately Orchestrated Routing and Transportation Analyzer: Large-scale Traffic Simulation for Autonomous Vehicles. In 15th IEEE Intelligent Transportation Systems Conference (ITSC), Anchorage, Alaska, USA, 2012. IEEE.

[51] Zafeiris Kokkinogenis, Lúcio Sanchez Passos, Rosaldo Rossetti, and Joaquim Gabriel. Towards the next-generation traffic simulation tools: a first evaluation. In 6th Iberian Conference on Information Systems and Technology, pages 15–18, 2011.

[52] Kurt Mauro Dresner. Autonomous intersection management. 2009.

[53] Minjie Zhu, Xu Li, Hongyu Huang, Linghe Kong, Minglu Li, and Min-You Wu. Licp: A look-ahead intersection control policy with intelligent vehicles. In 2009 IEEE 6th International Conference on Mobile Adhoc and Sensor Systems, pages 633–638, Oct 2009.

[54] Arnaud de La Fortelle. Analysis of reservation algorithms for cooperative planning at intersections. In 2010 13th International IEEE Conference on Intelligent Transportation Systems (ITSC), pages 445–449. IEEE, 2010.

[55] Chen, Tingting, Chen, Fei, and Chen, Cheng. Review on driverless traffic from management perspective. MATEC Web Conference, 124:03002, 2017.

[56] Joyoung Lee, Byungkyu Brian Park, Kristin Malakorn, and Jaehyun Jason So. Sustainability assessments of cooperative vehicle intersection control at an urban corridor. Transportation Research Part C: Emerging Technologies, 32:193–206, 2013.

[57] Md Abdus Samad Kamal, Jun-ichi Imura, Tomohisa Hayakawa, Akira Ohata, and Kazuyuki Aihara. A vehicle-intersection coordination scheme for smooth flows of traffic without using traffic lights. IEEE Transactions on Intelligent Transportation Systems, 16(3):1136–1147, 2015.

[58] Laleh Makarem, Minh-Hai Pham, André-Gilles Dumont, and Denis Gillet. Microsimulation modeling of coordination of automated guided vehicles at intersections. Transportation Research Record: Journal of the Transportation Research Board, pages 119–124, 2012.

[59] Kyoung-Dae Kim and Panganamala Ramana Kumar. An mpc - based approach to provable system-wide safety and liveness of autonomous ground traffic. IEEE Transactions on Automatic Control, 59(12):3341–3356, 2014.

[60] Gabriel Rodrigues de Campos, Paolo Falcone, and Jonas Sjoberg. Autonomous cooperative driving: a velocity-based negotiation approach for intersection cross-

ing. In Intelligent Transportation Systems-(ITSC), 2013 16th International IEEE Conference on, pages 1456–1461. IEEE, 2013.

[61] A. de La Fortelle. Analysis of reservation algorithms for cooperative planning at intersections. In 13th International IEEE Conference on Intelligent Transportation Systems, pages 445–449, Sept 2010.

[62] Alessandro Colombo and Domitilla Del Vecchio. Least restrictive supervisors for intersection collision avoidance: A scheduling approach. IEEE Transactions on Automatic Control, 60(6):1515–1527, 2015.

[63] Remi Tachet, Paolo Santi, Stanislav Sobolevsky, Luis Ignacio Reyes-Castro, Emilio Frazzoli, Dirk Helbing, and Carlo Ratti. Revisiting street intersections using slot-based systems. PLOS ONE, 11(3):1–9, 03 2016.

[64] Sayed (Reza) Azimi, gaurav Bhatia, Rajkumar Raghunathan, and Priyantha Mudalige. Reliable Intersection Protocols Using Vehicular Networks. In ICCPS, Philadelpia, 2013. ACM.

[65] On the Road to Fully Self-Driving. Waymo Safety Report, 2017.

[66] Timothy J. Wright, William J. Horrey, Mary F. Lesch, and M. Mahmudur Rahman. Drivers trust in an autonomous system: Exploring a covert video-based measure of trust. Proceedings of the Human Factors and Ergonomics Society Annual Meeting, 60(1):1334–1338, 2016.

[67] Kurt Dresner and Peter Stone. Multiagent traffic management: a reservation-based intersection control mechanism. In International Joint Conference on Autonomous Agents and Multiagent Systems, pages 530–537, 2004.

[68] Api documentation for aim simulator. http://www.cs.utexas.edu/~aim/aim4sim/aim4-release-1.0.3/aim4-root/target/site/apidocs/index.html. Accessed: 2016-03-23.

[69] Edward Curry. Message-Oriented Middleware, pages 1–28. John Wiley and Sons, Ltd, 2005.

[70] Koosha Paridel, Engineer Bainomugisha, Yves Vanrompay, Yolande Berbers, and Wolfgang De Meuter. Middleware for the Internet of Things, Design Goals and Challenges. Proceedings of the Third International DisCoTec Workshop on Context-Aware Adaptation Mechanisms for Pervasive and Ubiquitous Services (CAMPUS 2010), 28:1–7, 2010.

[71] Nicolas Nannoni. Message-oriented Middleware for Scalable Data Analytics Architectures. PhD thesis, KTH - Information and Communication Technology School, 2015.

[72] J.O'Hara. Towards a commodity enterprise middleware. Queue, 5(4):48–55, 2007.

[73] Oleksii Kononenko, Olga Baysal, Reid Holmes, David R. Godfrey, and Michael W. Mining Modern Repositories with Elasticsearch. In MSR, 2014.

[74] S. Rajendran, S.R. Chelladurai, and A. Aravind. An adaptive road traffic regulation with simulation and internet of things. In SIGSIM-PADS 2016 - Proceedings of the 2016 Annual ACM Conference on Principles of Advanced Discrete Simulation, 2016.

[75] Zolfaghari Alireza Le Vine, Scott and John Polak. Autonomous cars: The tension between occupant experience and intersection capacity. Transportation Research Part C: Emerging Technologies, 52:1–14, March 2015.

[76] Markus Maurer and J Christian Gerdes. Autonomous Driving: Technical, Legal and Social Aspects. 2016.

[77] Noah J Goodall. Vehicle Automation and the Duty to Act. Detroit, Michigan, 2014.

[78] IEEE. IEEE Code of Ethics. http://www.ieee.org/portal/pages/iportals/aboutus/ethics/code.html. Accessed: 2018-05-18.

[79] Alexander M Wyglinski, Xinming Huang, Taskin Padir, Lifeng Lai, Thomas R Eisenbarth, and Krishna Venkatasubramanian. Security of autonomous systems employing embedded computing and sensors. IEEE micro, 33(1):80–86, 2013.

[80] Inside the minds of bug bounty researchers. https://www.itspmagazine.com/from-the-newsroom/inside-the-minds-of-bug-bounty-researchers. Accessed: 2018-05-03.

[81] Tobias Holstein, Gordana Dodig-Crnkovic, and Patrizio Pelliccione. Ethical and Social Aspects of Self-Driving Cars. In ARXIV, 2018.

[82] Cara Bloom, Joshua Tan, Javed Ramjohn, and Lujo Bauer. Self-driving cars and data collection: Privacy perceptions of networked autonomous vehicles. In Symposium on Usable Privacy and Security (SOUPS), 2017.

[83] Guni Sharon and Peter Stone. A protocol for mixed autonomous and human-operated vehicles at intersections. In Autonomous Agents and Multiagent Systems - AAMAS 2017 Workshops, Best Papers, volume 10642 of Lecture Notes in Artificial Intelligence, pages 151–167. Springer International Publishing, 2017.