Formalization and Validation for O2DES Path Mover: A Discrete Event Traffic Simulation Package

by

ZHAN ZHENFAN

SUBMITTED TO

DEPARTMENT OF INDUSTRIAL AND SYSTEMS ENGINEERING

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS

FOR THE DEGREE

MASTER OF SCIENCE (INDUSTRIAL AND SYSTEMS ENGINEERING)

AT

NATIONAL UNIVERSITY OF SINGAPORE

APARIL 2024

Acknowledgements

I would like to express my deepest appreciation to Professor Li Haobin, whose guidance and support were pivotal throughout the course of this research. His profound expertise and insightful critiques not only shaped the direction of this study but also enriched my understanding of the subject matter.

I am also immensely grateful to Xiongfei Lai, whose mentorship in the field of traffic engineering has been invaluable. His recommendations and advice have significantly contributed to my professional growth and the development of this project.

Their willingness to give their time so generously has been very much appreciated. I am truly honored to have had the opportunity to learn from such knowledgeable and dedicated individuals.

Formalization and Validation for O2DES Path Mover: A Discrete Event Traffic Simulation Package

By

ZHAN ZHENFAN

Submitted to Department of ISE on June 2015

In Partial Fulfilment of the Requirements For The Degree

Master of Science in Industrial and Systems Engineering

Abstract

This study introduces and validates the PathMoverLibrary, a novel discrete-event simulation framework, streamlined to enhance efficiency in traffic simulations. We integrate the PathMoverLibrary with centralized control systems to facilitate macro-level decision-making and resource allocation in industrial environments like port operations. A detailed comparative analysis with the established VISSIM software confirms the effectiveness of the PathMoverLibrary in simplifying simulation processes while maintaining high accuracy and reliability. The results demonstrate the framework's potential to improve operational efficiency and inform strategy development in complex automated systems

Keywords

Discrete Event Simulation

Traffic Simulation

PathMoverLibrary

Centralized Control Systems

VISSIM

Operational Efficiency

Industrial Automation

Table of Contents

Acknowledgements 2

Abstract 3

CHAPTER 1 INTRODUCTION 7

1.1 Traffic Simulation in Manufacturing 7

1.2 Existing Simulation Frameworks 8

1.3 Advantages of Discrete Event Simulation (DES) 9

1.4 Overview of PathMoverLibrary 10

CHAPTER 2 RESEARCH OBJECTIVE 12

CHAPTER 3 LITERATURE REVEIW 13

3.1 Centralized Control Systems: The Manufacturing Game Changer 13

3.2 Challenges in Current Port Traffic Simulation 13

3.3 DES for Streamlined Port Operations 14

3.4 From Multi-Module to Single-Module 15

3.5 Wiedemann 99 Car Following Model 17

CHAPTER 4 METHEMATIC MODELING 18

4.1 Definition of Parameter and Variable 18

4.2 Definition of Event and Event Graph Procedure 21

4.3 Dynamic Speed Function 31

Chapter 5 EXPERIMENT 32

5.1 Parameterization 32

5.2 Input Data Analysis 33

5.3 Simulation Experiment Design 37

5.4 Data Collection Plan 39

5.5 Data Analysis 40

Chapter 6 CONCLUSION AND FUTURE WORK 47

6.1 Conclusion 47

6.2 Future Work 49

References 51

CHAPTER 1 INTRODUCTION

1.1 Traffic Simulation in Manufacturing

Traffic simulation serves as a pivotal tool in refining the operational efficiency of manufacturing industries and port operations, each utilizing the technology to cater to their unique logistical challenges. In manufacturing, the adoption of traffic simulation optimizes assembly line functions, inventory management, and cost reduction strategies. This facilitates a proactive approach to production, where potential bottlenecks are anticipated and addressed through detailed process analysis and operational forecasting.

Transitioning to port operations, the utility of traffic simulation is equally significant. It enhances the efficiency of cargo handling by coordinating the intricate dance of loading, unloading, and storage activities. Furthermore, it streamlines the scheduling of transport vehicles, ensuring that the throughput of the port is maximized. This is crucial in port management, where operations are time-sensitive and delays can lead to significant logistical setbacks.

Both industries benefit from the incorporation of centralized control systems, which are renowned for their capacity to make macro-level decisions efficiently, allocate resources effectively, and rapidly adapt to operational changes. Traffic simulation becomes an integral component of these systems, ensuring that all facets of the industrial process are synchronized to respond adeptly to the dynamic demands of production and market fluctuations. Through this integration, traffic simulation transcends its role as a mere planning tool, becoming an essential element in the pursuit of comprehensive operational excellence.

1.2 Existing Simulation Frameworks

Building on the synergy between traffic simulation and centralized control systems in manufacturing and port operations, we witness a significant dependency on existing simulation frameworks. Software like VISSIM and SUMO stands out for their ability to intricately model human-driven factors and behaviors in continuous or discrete-time simulation environments. These tools have been instrumental for detailed scenario analyses, serving as critical components for strategic planning, allowing stakeholders to simulate a wide array of traffic conditions and assess their potential impacts.

Yet, the landscape of industrial operations is rapidly evolving with the advent of automation. The precision once demanded for capturing human nuances in traffic simulation is being overshadowed by the need for broader system optimization. As automated processes become more prevalent, the focus shifts towards simulations that prioritize the efficiency of data flow and the fluidity of control interactions within these systems. Detailed simulations, while rich in data, may not align with the brisk pace and adaptability required by increasingly automated systems. The reliance on frameworks that can swiftly process large datasets and support complex, automated decision-making processes becomes paramount.

This evolving requirement underscores a critical gap in the capabilities of current traffic simulation tools. There is a growing need for frameworks that are not only agile and capable of handling the complexities of automation but also scalable to adapt to the changing dynamics of industrial systems. Such tools would enable a more effective integration with centralized control systems, ensuring that the entire network of operations remains agile and responsive. In light of these developments, it becomes clear that the next generation of simulation frameworks must be designed to be more flexible, integrating seamlessly with the fabric of automated operations to facilitate high-level decision-making and resource management in real-time.

1.3 Advantages of Discrete Event Simulation (DES)

In the context of modern industrial automation, the utility of existing traffic simulation frameworks is being reevaluated with a shift in focus towards Discrete Event Simulation (DES). As industries lean into the era of automation, the discrete, event-driven nature of DES becomes a cornerstone for modeling and managing the system dynamics of automated controls. Unlike the granular approach of traditional simulations, DES adopts a streamlined methodology, concentrating on the sequence of discrete events that fundamentally alter the system's state.

DES simplifies the complexity inherent in detailed simulations by homing in on pivotal events that dictate system behavior, effectively mapping out system-wide trends and responses. This focused approach allows for an expedited evaluation of strategies and facilitates swift decision-making processes. In an automated setting, the relevance of DES is amplified due to its ability to directly interact with control systems, swiftly processing and responding to the discrete events that occur within the industrial workflow.

The distinct advantage of DES lies in its inherent flexibility and scalability. It is well-suited to the task of adapting to rapid changes in system states, a frequent characteristic of automated industrial environments. This flexibility ensures that DES can keep pace with the dynamic requirements of large-scale operations, from manufacturing to port management, where the precision of response times and the agility to adapt to changing scenarios are nonnegotiable.

Additionally, DES contributes to optimizing resource allocation by eliminating unnecessary simulation detail that may not impact the broader operational goals. By streamlining the simulation process, DES reduces computational overhead, which translates into faster run times and more efficient use of processing resources. The consequence is a more agile, responsive system capable of providing timely insights into operational performance and potential improvements.

In sum, DES emerges as an essential tool in the arsenal of industrial automation, aligning with the growing need for simulation frameworks that prioritize efficient data and control interactions. Its capacity to model complex systems through the lens of discrete events positions it as a pivotal enabler of strategic evaluation and real-time decision-making in an increasingly automated industrial landscape.

1.4 Overview of PathMoverLibrary

The PathMoverLibrary, as illustrated in the provided screenshot and described, stands at the forefront of discrete-event simulation frameworks, tailored for managing and simulating vehicle flows in traffic networks. Here's an expanded point-by-point overview of its structure and capabilities:

- Modular Design: The PathMoverLibrary adopts a modular architecture, which is evident from its file structure. Core components like 'ControlPoint.cs', 'PathMover.cs', 'PathMoverStatics.cs', 'PmPath.cs', and 'Vehicle.cs' suggest a design that allows for the encapsulation of functionality. Each module can be developed, tested, and debuged independently before being integrated into the larger system, enhancing the framework's robustness and scalability.

- Control Point Mechanism: The 'ControlPoint.cs' module likely handles the dynamic elements within the traffic network, such as intersections, stop signs, or traffic signals. This module's role in the library is pivotal for simulating the control mechanisms that vehicles encounter on the network, affecting their movement and interactions.
- Path Movement and Statistics: With files named 'PathMover.cs' and 'PathMoverStatics.cs', the framework seems equipped to track and analyze the movement of vehicles along paths within the network. These modules may provide the core simulation functionality, calculating speeds, handling queuing logic, and possibly even simulating the behavior of individual drivers or autonomous systems.
- Path Configuration: 'PmPath.cs' suggests a component dedicated to defining the paths that vehicles can take. This might involve setting up routes, defining network topologies, or configuring the parameters for how vehicles choose their paths within the simulation environment.
- Vehicle Dynamics: 'Vehicle.cs' is indicative of a module that represents the vehicles themselves within the simulation. It could encompass various vehicle types, their properties, dynamics, and the rules governing their interaction with the network and each other.
- High-Level Abstraction: The library's high-level abstraction allows users to interact with complex systems through simplified interfaces, without getting bogged down by the intricacies of the underlying simulation logic.

- Lightweight and Integrable: Its lightweight nature ensures that the library does not overburden the host systems, making it an attractive addition to existing frameworks looking to enhance their traffic simulation capabilities.
- Built on O2DES: Utilizing the established O2DES framework provides a solid foundation, ensuring that the PathMoverLibrary benefits from the reliability and accuracy of a well-tested simulation engine.
- Customization and Maintenance: The library's small footprint does not just contribute to performance efficiency; it also facilitates easier maintenance and customization, allowing users to tailor the library to meet specific requirements without extensive overhead.

The PathMoverLibrary encapsulates the essentials of modern traffic simulation within a flexible and powerful package, highlighting the evolution of simulation frameworks in the face of growing demands for precision, efficiency, and adaptability in automated industrial systems.

CHAPTER 2 RESEARCH OBJECTIVE

Objective: The primary objective of this research is the mathematical formalization of the PathMoverLibrary simulation package. This includes designing a definitive framework that rigorously encapsulates the library's operational logic, ensuring mathematical coherence and precision in its application to traffic simulation.

Purpose: The purpose of the study is twofold: firstly, to establish a robust mathematical foundation for the PathMoverLibrary, enhancing its scientific underpinnings. Secondly, to conduct empirical evaluations through comparative analysis with the VISSIM software, aiming to ascertain the PathMoverLibrary's accuracy and practicality for real-world applications in complex automated systems. This endeavor will identify areas of improvement, paving the way for broader industrial application and advancement of the PathMoverLibrary.

CHAPTER 3 LITERATURE REVEIW

3.1 Centralized Control Systems: The Manufacturing Game Changer

While traditional discrete time simulations (DTS) have been instrumental in detailed planning and analysis, the increasing complexity of port operations requires faster and more adaptable solutions. Discrete event simulations (DES) have emerged as a superior methodology, offering significant improvements in flexibility and execution speed, essential for the dynamic nature of port systems.

3.2 Challenges in Current Port Traffic Simulation

According to Thesen (1978), discrete event simulation models are particularly beneficial in scenarios that require efficient management of flows, which can be extended to complex processes within industrial settings.

Cocchi et al. (2016) demonstrates the successful application of discrete event simulation in developing and monitoring hospital services, suggesting a wider potential for DES in optimizing operations in fields such as manufacturing.

The integration of Discrete Event Simulation (DES) within centralized control systems signifies a transformative shift in manufacturing towards a more holistic form of system-level optimization. This paradigm shift highlights the strategic emphasis on macro-level decision-making processes. By harnessing DES, manufacturers can significantly enhance operational efficiency. The technology allows for the assimilation of traffic simulations into an expansive operational framework, streamlining the entire manufacturing workflow.

Centralized control systems with embedded DES capabilities enable organizations to move away from the granular focus on micro-level detail that traditionally characterized simulation models. Instead, they facilitate an overarching view that brings into alignment various operational components, thus driving more informed, data-driven decision-making. These systems, by dynamically simulating key traffic flows and logistical patterns within the manufacturing process, allow for real-time adjustments and predictive analyses. This optimizes resource allocation, production schedules, and even the supply chain logistics, making operations more responsive to the fast-changing market.

3.3 DES for Streamlined Port Operations

Discrete Event Simulation (DES) has emerged as a highly effective tool for modeling event-driven systems, particularly in sectors where operational efficiency is paramount, such as port operations. Its strengths lie in its innate ability to encapsulate the complexities of these systems into a series of discrete, manageable events, each of which is processed and analyzed for its impact on the overall system. This modeling approach aligns perfectly with the speed and adaptability required in contemporary industrial applications, where rapid trend analysis and quick response times are not just beneficial but essential for maintaining a competitive edge.

Narain and Chadha (1995) delve into how discrete-event simulation models serve to distill the complexity of dynamic systems into simpler, more comprehensible modules. This simplification is crucial in facilitating a deeper understanding of the underlying mechanics of systems that are inherently complex and dynamic, such as those found in port logistics. By breaking down these systems into discrete events, DES allows for an agile response to changes, which is critical in environments where delays can lead to significant operational and financial setbacks.

Further emphasizing the versatility of DES, Pidd and Cassel (2000) highlight the advantages of leveraging programming languages like Java to enhance the accessibility and flexibility of these simulations. The object-oriented nature of Java dovetails with the core principles of DES, offering a structured yet flexible programming environment that can easily adapt to the nuanced requirements of complex simulations. This is particularly beneficial for the intricate and multifaceted simulations required in port operations, where multiple variables and rapidly changing conditions are the norms.

The effectiveness of DES in port operations is reflective of its broader applicability in the field of industrial engineering and operations management. Its ability to provide a granular view of system behavior while maintaining a focus on the broader operational picture ensures that DES is not just a theoretical construct but a practical tool for improving efficiency, throughput, and operational resilience in some of the most challenging and dynamic industrial environments..

3.4 From Multi-Module to Single-Module

The PathMoverLibrary, evolving from the comprehensive work of Zhou et al. (2017), represents a leap in traffic network simulation through its adoption of a single-module framework. This paradigm shift, informed by the original study's adept handling of container terminal traffic via a complex network of servers with dynamic rates, promises a trajectory towards models that are not only simplified but inherently dynamic. The original multi-modular system, while effective, revealed the potential for a streamlined approach that could marry simplicity with the capability to adapt to fluctuating demands.

Inheriting the robust core of its predecessor, the PathMoverLibrary refines the concept, morphing the multifaceted modular complexity into a unified, coherent framework. This transition exemplifies a deliberate design choice to balance the original system's flexibility and dynamic nature with a newfound focus on operational efficiency and user-friendliness. The enhancements introduced with the PathMoverLibrary are not merely incremental; they are transformative, redefining ease of integration without compromising on the adaptive qualities essential to handling real-world complexities.

The progression to a single-module system is a testament to the PathMoverLibrary's design philosophy, which emphasizes agility without sacrificing depth or functionality. It stands as a testament to a deep understanding of the operational intricacies inherent in industrial applications like port logistics. The single-module structure is not just an architectural decision; it is a commitment to seamless interoperability, allowing the PathMoverLibrary to integrate into a multitude of operational environments with grace and without friction. This characteristic is particularly vital in complex industrial scenarios, where the strength of a simulation framework is measured by its ability to withstand and accurately model the nuances of high-stakes logistics operations.

The implications of such a system are far-reaching. By enabling a simpler, yet no less capable, simulation environment, the PathMoverLibrary equips engineers and logistics professionals with a tool that can stand up to the rigors of industrial application while providing a user experience that democratizes access to advanced simulation capabilities. It is in this balance that the PathMoverLibrary finds its stride, promising to support the continuous evolution of industrial efficiency and the strategic planning necessary to navigate the ever-changing landscape of port logistics and beyond.

3.5 Wiedemann 99 Car Following Model

The Wiedemann 99 car-following model represents an intricate and detailed approach to replicating driver behavior in traffic flows. The model is founded on Wiedemann's psychophysical model of driver behavior, which takes into account a variety of factors that influence how drivers follow each other on the road. This includes the perception and reaction to the behavior of the vehicle in front, and how these perceptions change in different traffic conditions and states of motion.

Wiedemann 99 improves upon earlier models by incorporating a broader range of driving behaviors and more nuanced transitions between them. It outlines several key states in car-following behavior, such as free driving, approaching, following, and braking, each with its own set of parameters and thresholds that can be adjusted to simulate different driving styles and conditions.

The advantage of using the Wiedemann 99 model in traffic simulation lies in its ability to provide a high-resolution simulation of individual driver behaviors, allowing for more realistic and precise modeling of traffic phenomena. It is particularly useful in simulations that require a detailed analysis of interactions within traffic streams, such as the start and end of congestion, oscillations in traffic flow, and the impacts of infrastructural changes.

CHAPTER 4 METHEMATIC MODELING

4.1 Definition of Parameter and Variable

PathMover Parameters Definition:

- K: Set of all control point indices within the network. Each index $k \in \mathbb{Z}^+$.
- I: Set of all path indices used in the network. Each index $i \in \mathbb{Z}^+$.
- J: Set of all vehicle indices within the simulation. Each index $j \in \mathbb{Z}^+$.

The "PathMover Parameters" P_{PM} are defined by the following collections:

$$P_{PM} = \{RT, K, I, J\}$$

Where:

- $K = \{k \mid k \text{ is an index representing a control point in } CP, \text{ and } k \in \mathbb{Z}^+\}$
- $I = \{i \mid i \text{ is an index representing a path in } P, \text{ and } i \in \mathbb{Z}^+\}$
- $J = \{j \mid j \text{ is an index representing a vehicle in the simulation, and } j \in \mathbb{Z}^+\}$

RT is the Route Table, redefined as a mapping from control point index to control point index, corresponding to a path index:

$$RT = \{((k, k'), i) \mid k, k' \in K, i \in I, k \neq k'\}$$

Figure 1 Parameter of PathMover

Path Parameters

"Let i index a path, we define the parameters of a path as:

$$P_i^P = \{l_i, c_i, s_i, o_i\}$$

where:

- l_i denotes the length of path i, with $l_i > 0$.
- c_i denotes the total capacity of path i, with $c_i \geq 0$.
- s_i and o_i denote the starting and ending control points of path i, respectively, with $s_i, o_i \in K$ and $s_i \neq o_i$.

Figure 2 Parameter of path

Parameters of a Vehicle j:

- Parameters: $P_j^V = \{N_j, L_j, S_j, CN_j\}$
  - N_j: Name or identifier of the vehicle.
  - L_j: Length of the vehicle, with $L_j > 0$.
  - S_j: Speed of the vehicle, with $S_j > 0$.
  - CN_j: Capacity needed by the vehicle, with $CN_j \in \mathbb{Z}^+$.

Figure 3 Parameter of vehicle

Control Point Parameters:

A 'ControlPoint' represents a node within the traffic network which may serve as an origin or a destination.

Let P_k^{cp} be the set of all control points in the simulation network, where each control point k is uniquely identified by a tag:

$$P_k^{cp} = \{tag_k\}$$

Where:

- tag_k is a unique identifier for the control point k.

Figure 4 Parameter of control point

Dynamic Variables of Path i:

- Dynamic Variable: $V_i^P = \{RC_i, DTS_i, ETS_i, OPL_i, IPL_i\}$

Where:

- RC_i: Remaining Capacity, represents the remaining number of vehicles that path i can accommodate at a given time. It must satisfy $RC_i \geq 0$ and $RC_i \in \mathbb{Z}$.
- DTS_i: DepartTimeStamp, the timestamp of the most recent vehicle to depart from path i not including the origin of the traffic network. It must satisfy $DTS_i \geq 0$.
- ETS_i: EnterTimeStamp, the timestamp of the last vehicle to enter path i from the origin of the traffic network. It must satisfy $ETS_i \geq 0$.
- OPL_i: OutPendingList, a list of vehicles that are pending to depart from path i. Each vehicle j in OPL_i must satisfy $j \in J$.
- IPL_i: InPendingList, a list of tuples (vehicle, path) where each tuple represents a vehicle that is pending to enter path i and the path they will enter from. Each vehicle j in IPL_i must satisfy $j \in J$.

Figure 5 variable of path

Dynamic Variables of a Vehicle j:

- Dynamic Variable: $V_j^V = \{CP_j, NP_j(k), TL_j, ENT_j, EXT_j\}$
  - CP_j: Current path index of the vehicle, where $i \in I$.
  - $NP_j(k)$: Next path function that returns the next path index for vehicle j given the current control point index k, where $k \in K$. Specifically, if $TL_j = [k'_1, k'_2, \ldots, k'_n]$ and k'_1 is the next target control point index for k, then $NP_j(k)$ is defined as:

$$NP_j(k) = \{i \mid ((k, k'_1), i) \in RT\}$$

  - TL_j: Target list, an ordered set of control point indices the vehicle intends to visit, with each index k belonging to the set K.
  - ENT_j: Entry time of vehicle j into the system, satisfying $ENT_j > 0$.
  - EXT_j: Exit time of vehicle j from the system, satisfying $EXT_j > 0$ and $ENT_j < EXT_j$.

Figure 6 variable of vehicle

Dynamic Variables for Control Point k:

- Dynamic Variable: $V_{CP}^{k} = \{PL_k^{cp}\}$

Pending List Definition:

- The pending list for control point k is defined as PL_k^{cp}, where each j in PL_k^{cp} is an index representing a vehicle pending to enter at CP_k, with the constraint that $j \in J$.

Uniqueness Constraint:

- Every index j in PL_k^{cp} must be unique, such that for any $j, j' \in PL_k^{cp}$, it holds that $j \neq j'$, and for all $k' \neq k, j \notin PL_{k'}^{cp}$.

Figure 7 variable of control point

4.2 Definition of Event and Event Graph Procedure

Define the event $\alpha_{RE}(k, j)$ to represent control point k requesting to process vehicle j.

The restriction for this event is that the vehicle must not be in any pending list for control point k, any origin OPL_j, or any path IPL_j, where j is an element in the set J and k is any element in the set K:

$$\alpha_{RE}(k, j) : j \notin \bigcup_{k \in K} PL_k \wedge j \notin (OPL_j \cup IPL_j)$$

Where:

- j represents vehicle j.
- k is any element of the set K, indicating any control point.
- PL_k is the pending list of vehicles waiting to enter at control point k.
- OPL_j is the pending list of vehicles waiting at the origin corresponding to index j.
- IPL_j is the pending list of vehicles waiting to enter the associated path.

Figure 8 Definition and Constrain of Request to Enter Event

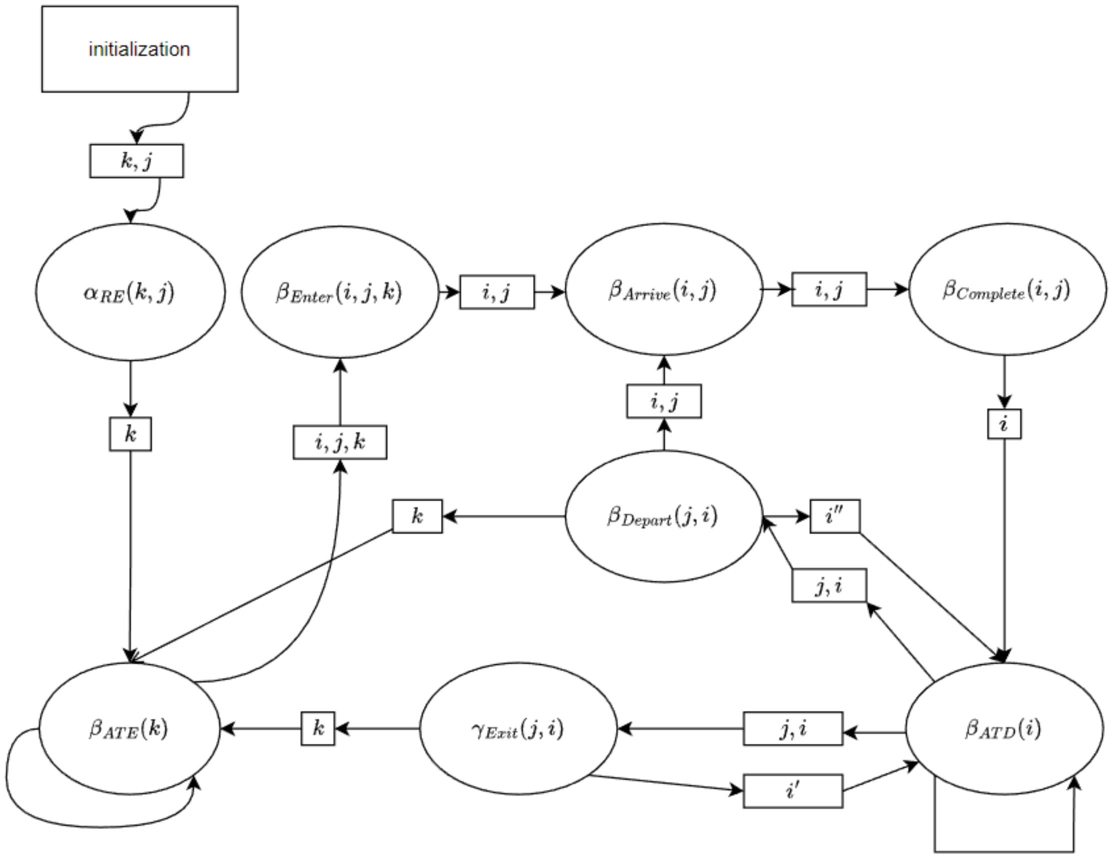
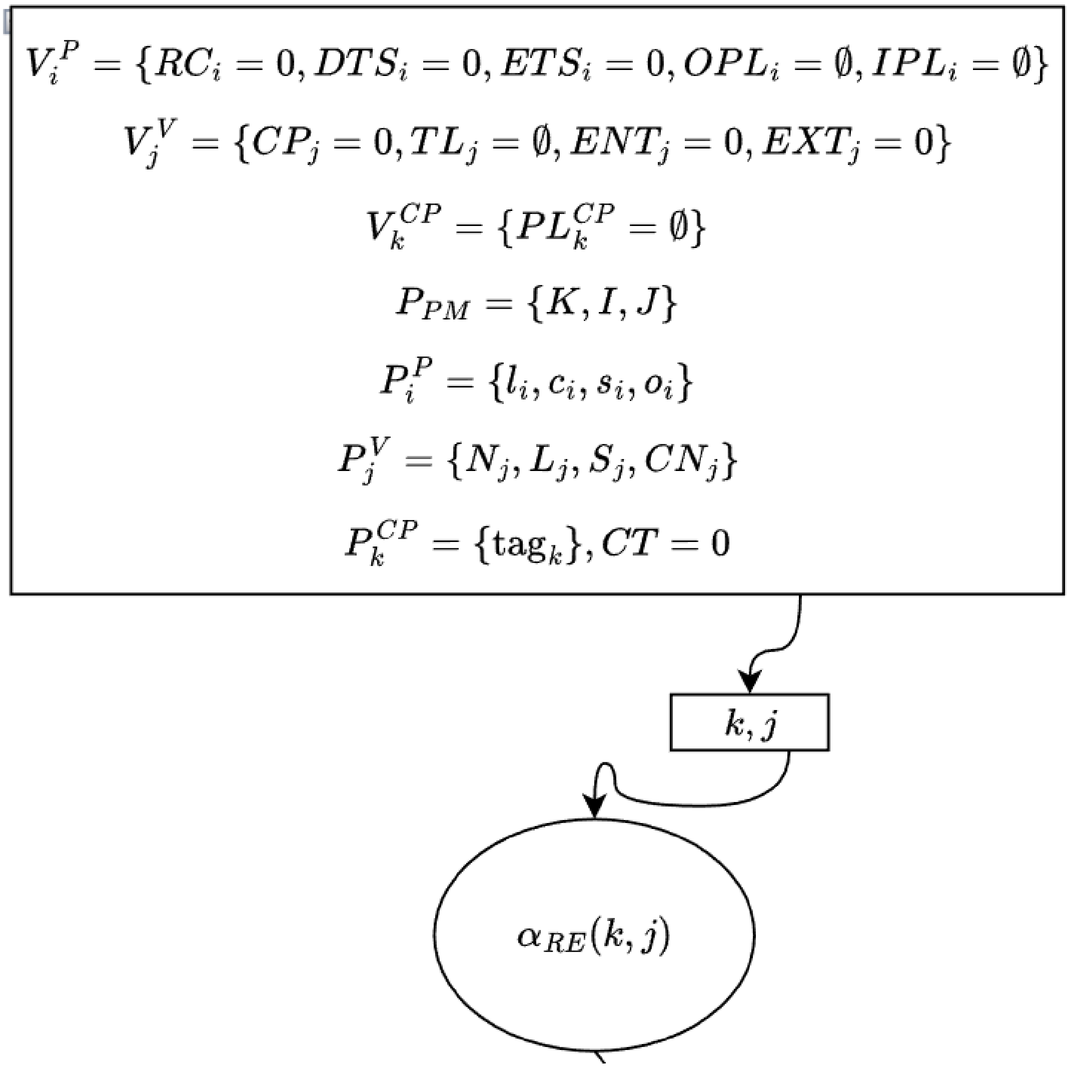


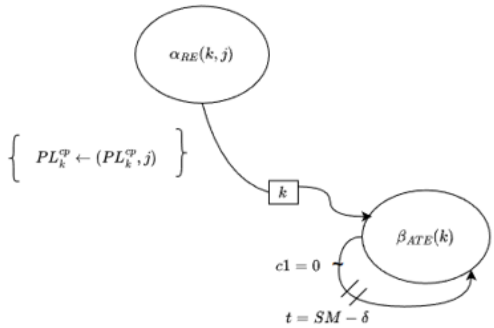
Figure 9 Overview of Event Graph



The initialization process is as follows:

- For each path i, the dynamic variable set V_i^P is initialized with the remaining capacity RC_i and timestamps DTS_i, ETS_i set to 0. The pending departure list OPL_i and the pending entry list IPL_i are set to the empty set.
- For each vehicle j, the dynamic variable set V_j^V is initialized with the current path index CP_j set to 0. The target list TL_j is initialized as the empty set, and the entry time ENT_j and exit time EXT_j are also set to 0.
- For each control point k, the dynamic variable set V_k^{CP} is initialized with the pending list PL_k^{CP} set to the empty set and the traffic count CT set to 0.
- The collections P_{PM}, P_i^P, P_j^V, and P_k^{CP} are also initialized, representing the sets of control point indices K, path indices I, vehicle indices J, and control point parameters $\{\text{tag}_k\}$ respectively, but the specific values of these collections are to be provided by external input.

Figure 10 Definition of Initialization



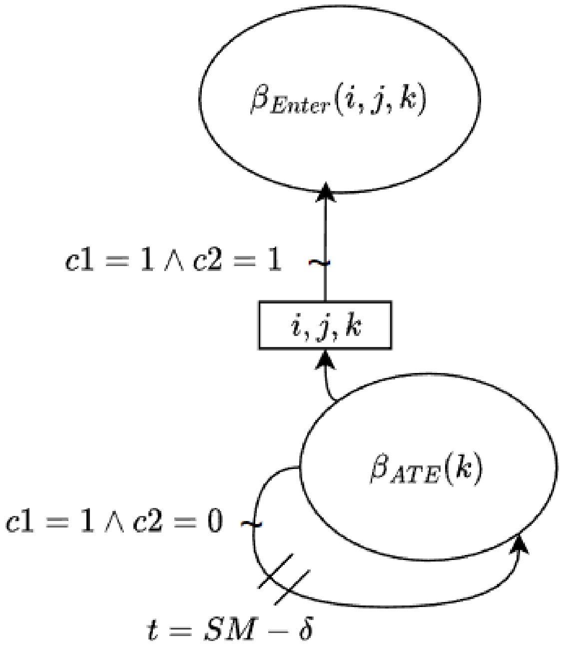
Event: $\alpha_{RE}(k, j)$

The event $\alpha_{RE}(k, j)$ denotes the request of vehicle j to enter control point C_{P_k}. Upon this event, the state update is performed as follows:

The pending list at control point k, $P_{L_{cp_k}}$, is updated by appending vehicle j to the end of the queue, indicating that vehicle j is now in line to enter at control point k.

This leads to the triggering of the internal event $\beta_{ATE}(k)$, which stands for the "Attempt to Enter" at control point with index k.

Figure 11 Definition of Request to Enter



$$\left\{ \begin{array}{c} c1 = \begin{cases} 1 & \exists j \in PL_{\mathrm{cpk}}, i = NP_j(k), RC_i \geq CN_j, \delta = CT - ETS_i \\ 0 & \text{otherwise} \end{cases} \\ c2 = \begin{cases} 1 & \delta > SM \\ 0 & \text{otherwise} \end{cases} \end{array} \right\}$$

Event: $\beta_{ATE}(k)$

Prerequisites:

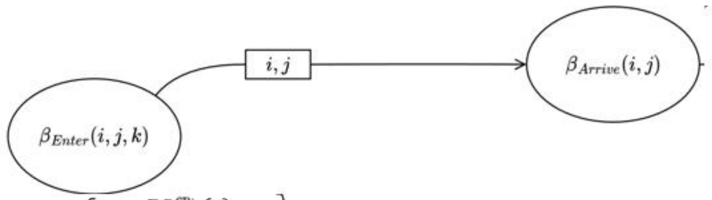
- The system checks for any vehicle j in the queue at control point k ($P_{L_{cp_k}}$) that can be cleared to proceed:
  - The vehicle's intended path i must have sufficient capacity (RC_i) available, equal to or greater than the vehicle's requirement (C_{N_j}).
  - Additionally, the time since the last vehicle's entry onto path i must surpass the smooth factor (SM), establishing safe entry intervals.

Triggered Internal Event:

- $\beta_{Enter}(i,j,k)$

The event $\beta_{ATE}(k)$ scans for eligible vehicles and, upon finding one that meets both spatial and temporal thresholds, initiates $\beta_{Enter}(i,j,k)$. If a vehicle only satisfies the capacity condition, the system schedules to reassess the event after the time interval satisfies the smooth factor (SM).

Figure 12 Definition of Attempt to Enter



Event: $\beta_{Enter}(i, j, k)$

State Update:

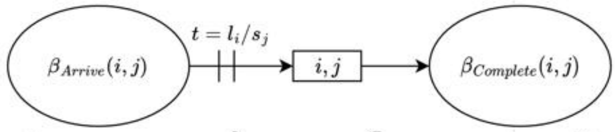
$$\left\{ \begin{array}{c} PL_k^{cp} \backslash \{j\} \\ ENT_j \leftarrow CT \\ ETS_i \leftarrow CT \end{array} \right\}$$

- Remove vehicle j from the queue at control point k ($P_{L_{cp_k}}$). This is denoted by the operation $P_{L_{cp_k}} := P_{L_{cp_k}} \setminus \{j\}$, indicating vehicle j's removal from the waiting list.
- Record the current simulation time (CT) as the entry time for vehicle j into the system (ENT_j) and as the most recent entry time onto path i (ETS_i).

Triggered Event:

- Following these updates, the system executes the event $\beta_{Arrive}(i, j)$ to process the arrival of vehicle j onto path i.

Figure 13 Definition of Enter



Event: $\beta_{Arrive}(j, i)$

State Update:

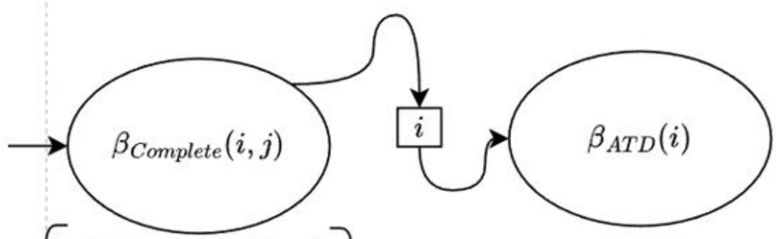
$$\left\{ \begin{array}{c} CP_j \leftarrow i, k = s_i \\ TL_j \backslash \{k\} \\ RC_i \leftarrow RC_i - CN_j \end{array} \right\} \qquad \left\{ OPL_i \leftarrow (OPL_i, j) \right\}$$

- Control point k is removed from the list of targets TL_j for vehicle j, updating the target list to $TL_j := TL_j \setminus \{k\}$.
- The available capacity of path i (RC_i) is reduced by the capacity requirement of vehicle j (CN_j), now showing $RC_i := RC_i - CN_j$.

Triggered Delayed Event:

- The system schedules the event $\beta_{Complete}(j, i)$ to occur after a delay t, which is calculated based on the length of path L_i and the speed of vehicle j (S_j), with $t = \frac{L_i}{S_j}$.

Figure 14 Definition of Arrive



Event: $\beta_{Complete}(i, j)$

State Update:

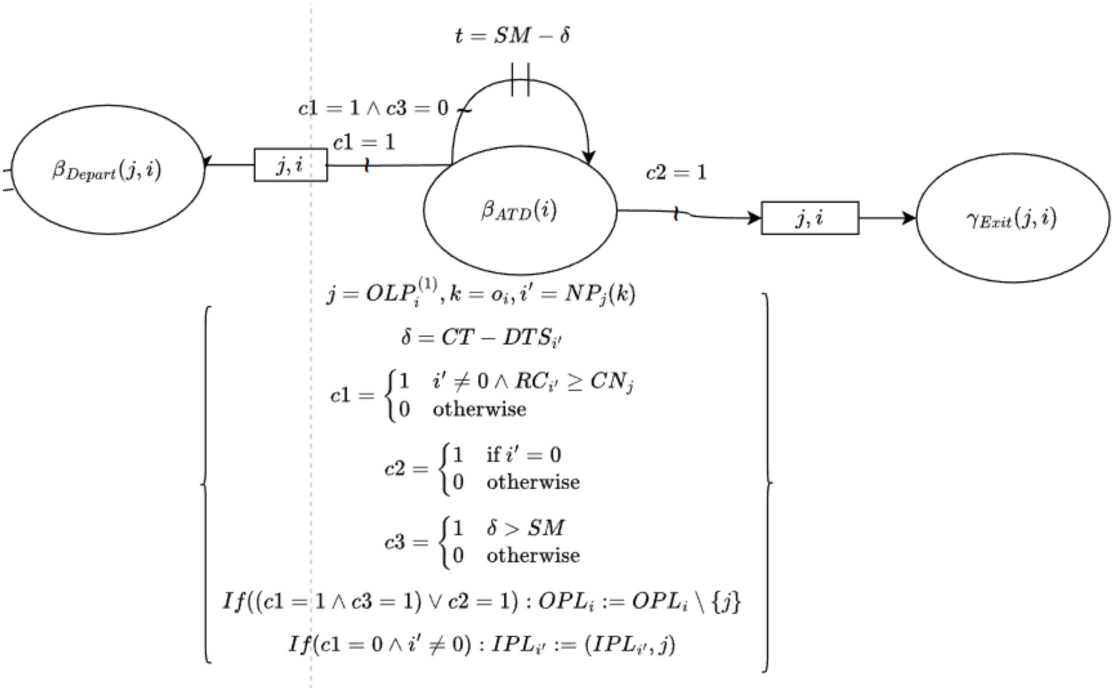
$$\left\{ \begin{array}{l} OPL_i \leftarrow (OPL_i, j) \\ TD_j \leftarrow TD_j + l_i \end{array} \right\}$$

- Vehicle j's total traveled distance TD_j is updated to account for the completion of path i: $TD_j \leftarrow TD_j + l_i$.
- Vehicle j is appended to path i's out-pending list OPL_i: $OPL_i \leftarrow (OPL_i, j)$.

Triggered Event:

- Updating the state $\beta_{Complete}(i, j)$ triggers $\beta_{ATD}(i)$, managing the departure of vehicles from OPL_i.

Figure 15 Definition of Complete



Event: $\beta_{ATD}(i)$

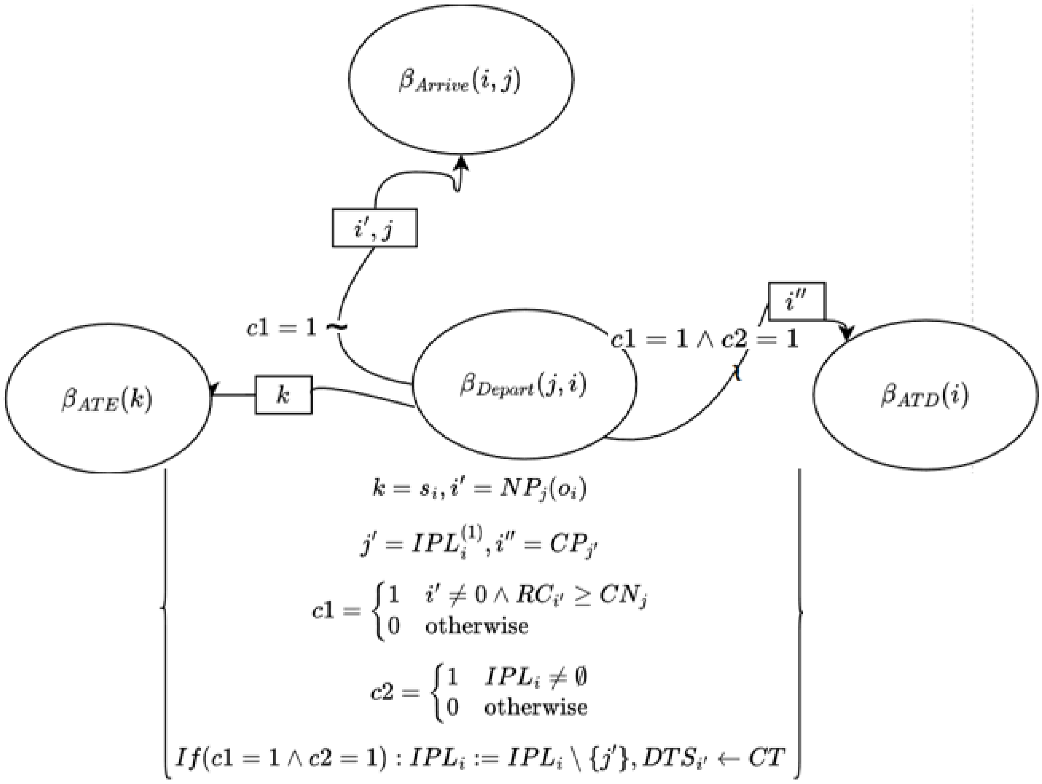
Trigger Logic:

- **Exit Event:** If vehicle j has no next path (i.e., $i' = \mathrm{NULL}$), the exit event $\gamma_{Exit}(j, i)$ is triggered, signaling that vehicle j will exit the system.
- **Depart Event:** If vehicle j has a subsequent path i' and meets the safety headway and capacity requirements, the depart event $\beta_{Depart}(j, i)$ is triggered, allowing vehicle j to depart from the current path i.
- **Reschedule Current Event:** If the conditions for the Depart event are not met, due to insufficient safety headway or capacity, the event $\beta_{ATD}(i)$ will be rescheduled.

State Update:

- If either the Exit or Depart event is triggered, vehicle j will be removed from the out-pending list OPL_i of the current path i.
- If the event is rescheduled, vehicle j will be added to the in-pending list $IPL_{i'}$ of the next intended path i'.

Figure 16 Definition of Attempt to Depart



Event: $\beta_{Depart}(j,i)$

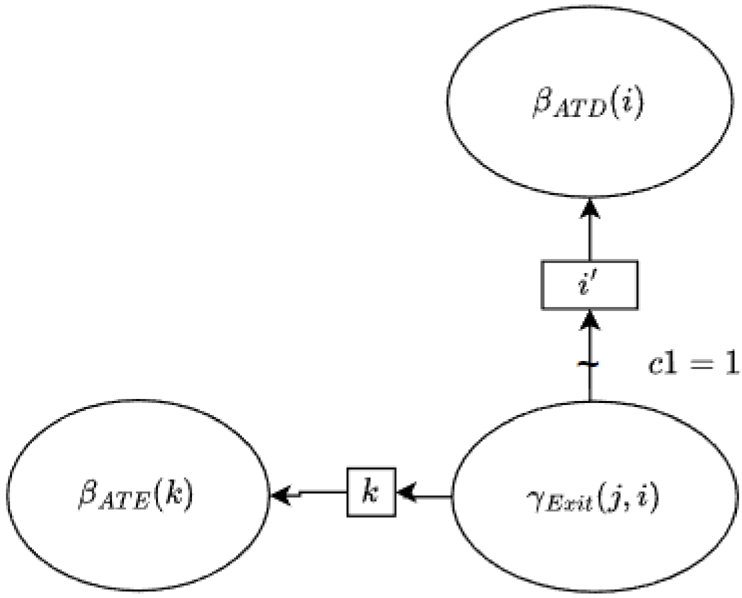
The departure of vehicle j from path i activates a sequence of checks that trigger specific events:

- **Trigger for $\beta_{Arrive}(i,j)$ Event (Capacity Check):** If the upcoming path i' can accommodate vehicle j, the vehicle is cleared to depart, which then triggers the $\beta_{Arrive}(i,j)$ event, marking its successful exit from path i.
- **Trigger for $\beta_{ATD}(i'')$ Event (In-Pending List Assessment):** If there's a queue of vehicles on the in-pending list $IPL_{i''}$ for the subsequent path i'', the event $\beta_{ATD}(i'')$ is triggered to evaluate and manage their departure.
- **Trigger for $\beta_{ATE}(i)$ Event (System Admittance):** When vehicle j vacates path i, the system verifies if any vehicles are queued at the start of path i looking to enter the system, leveraging the space freed by the departing vehicle.

State Update upon Triggering $\beta_{ATD}(i'')$:

- Should the $\beta_{ATD}(i'')$ event trigger due to a departure attempt, the vehicle at the front of the in-pending list IPL_i will be removed, and its attempt to leave is processed, thereby updating the system state to reflect the current vehicle distribution and path capacities.

Figure 17 Definition of Depart



$$\left\{ \begin{array}{c} k = s_i, \\ RC_i \leftarrow RC_i + CN_j, EXT_j \leftarrow CT \\ c1 = \begin{cases} 1 & IPL_i \neq \emptyset \\ 0 & \text{otherwise} \end{cases} \\ If(c1 = 1) : j' = IPL_i^{(0)}, i' = CP_{j'}, \end{array} \right\}$$

Event: $\beta_{Exit}(j, i)$

State Update Upon $\beta_{Exit}(j, i)$**:**

- **Release of Path Space:** The system updates the remaining capacity RC_i to account for vehicle j leaving path i. This is an automatic process whenever a vehicle exits a path.
- **Recording Exit Time:** The exit timestamp EXT_j for vehicle j is set to the current time CT, marking the moment of departure from the system.

Triggers After State Update:

- **Trigger for** $\beta_{ATD}(i')$ **Event (Departure Assessment for Successor):** If IPL_i is not empty, indicating that vehicles are waiting to enter path i, the $\beta_{ATD}(i')$ event is triggered for the next vehicle j' in line.
- **Trigger for** $\beta_{ATE}(k)$ **Event (Check for System Entry):** In parallel, $\beta_{ATE}(k)$ checks if vehicles are waiting to enter at the starting point of path i, utilizing the opening created by vehicle j's exit.

Figure 18 Definition of Exit

4.3 Dynamic Speed Function

Model Simplification:

By setting a constant speed v, the position x of a vehicle at any time t can be easily calculated using the linear equation $x = vt + x0$, where $x0$ is the initial position. This linear relationship eliminates the need for complex differential equations that would be required to model varying speeds over time. The fixed speed v aligns with the operational characteristics of automated vehicles, which are programmed to follow predictable and consistent motion patterns. This can be represented as v = constant, signifying that the speed does not change over time, thus facilitating straightforward trajectory planning and coordination in the simulation.

The VISSIM Wiedemann 99 model is complex, with parameters that exhibit significant randomness and variability, making it difficult to capture their effects through a simple formulaic approach. Key parameters include:

cc0 (Following Variation): Represents the minimum following distance at low speeds due to driver's reaction time and vehicle braking capabilities.

cc1 (Threshold for Entering Following): The distance or time headway below which a driver begins to follow another vehicle.

cc2 (Threshold for Exiting Following): The distance or time headway above which a driver stops following another vehicle closely and may overtake.

cc3 (Lateral Acceleration Thr. for Lane Changing): The lateral acceleration felt by drivers when changing lanes; impacts the decision to change lanes based on comfort and safety.

These parameters reflect the stochastic nature of driver behavior, which includes reaction times, acceleration and deceleration patterns, and adherence to safety distances. They are influenced by both the physical capabilities of vehicles and the psychological factors affecting individual drivers, which contribute to the variability and unpredictability in traffic simulations.

- smoothFactor (S): This parameter sets a safe time gap for orderly traffic, ensuring that each vehicle enters a path with a sufficient interval after the preceding one. For vehicle i entering at time ti, the following vehicle i + 1 will enter at $t_{i+1} \geqslant t_i + S$.
- coldStartDelay (C): This measures the delay for a vehicle's acceleration from rest to its cruising speed, accommodating the time needed to reach optimal flow conditions after a stop. A vehicle starts moving at time t + C after a stop.

Due to the lack of direct correspondence and the high degree of randomness in the parameters, subsequent experiments will employ a search method to identify the most fitting values for 'Smooth Factor (SM)' and 'Cold Start Delay (CD)'. This approach will account for the variability and unpredictability of individual driver behaviors and vehicle interactions in the traffic model.

Chapter 5 EXPERIMENT

5.1 Parameterization

1. Desired Speed:

- VISSIM (V): Simulates the target speed that drivers aim to maintain under ideal conditions.
- PathMover (P): Utilizes a fixed speed for all vehicles to simplify model operation.

2. Vehicle Volume:

Both VISSIM (V) and PathMover (P): Specifies the number of vehicles entering the system per hour, representing the traffic inflow.

3. Vehicle Routing Decision:

- VISSIM (V): Probability of vehicles choosing their paths at intersections, simulating driver decision-making at junctions.
- PathMover (P): Controlled by Traffic Light (TL) initialization logic, essentially a probabilistic model for route selection.

4. Time Distribution of Incoming Vehicles:

Both VISSIM (V) and PathMover (P): Controls the time distribution for vehicles entering the simulation, mirroring real-world arrival patterns.

5. Driving Behavior:

- VISSIM (V): Utilizes the Wiedemann 99 model to simulate nuanced driver behaviors, considering various driving conditions and reactions.
- PathMover (P): Employs smoothFactor and coldStartDelay parameters to control driving behavior, simplifying the simulation of driver actions.

5.2 Input Data Analysis

The Input Data Analysis phase of this study was designed to ensure the simulation parameters (vehicle input distribution, driving behavior) of the model closely approximate those of the established VISSIM model. The experimental setup aims to investigate the temporal distribution of vehicle arrivals at the system and the traffic flow and average velocity under congestion conditions. Specifically, the study will examine:

The arrival times of 1,000 vehicles at checkpoint CP2, originating from checkpoints CP1, CP4, and CP5, to discern the spread over time and potential peak congestion periods.

The volumetric flow rate through the central node, CP2, towards CP3, during periods of peak congestion, to measure how the road network copes with high vehicle density.

The average velocity of the vehicles within the network, particularly between CP2 and CP3, to evaluate the impact of congestion on travel speeds.

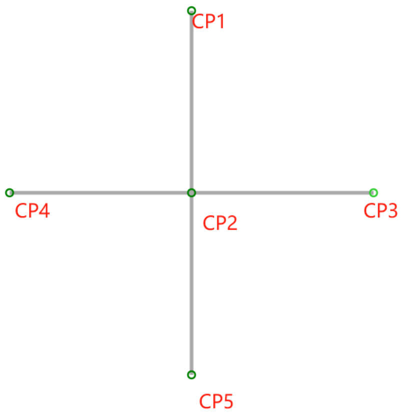


Figure 19 Experiment traffic network

detailed experimental steps:

1. Initiation: Commence with 100 simulations on a reduced-scale network, collecting data on vehicle arrival times, traffic flow from CP2 to CP3, and average system speed, post a 1000-second warm-up phase, over 4600 seconds of active simulation.
2. Optimization: Apply data fitting techniques to align with VISSIM outputs, employing global optimization to calibrate 'smooth factor' and 'cold start delay' to achieve a GEH statistic under 5, signifying minimal discrepancy with VISSIM's average speed results.

3. Calibration: Adjust input distributions and driving behaviors within the model parameters to closely reflect those in VISSIM, ensuring the simulation's validity and application accuracy for traffic analysis.

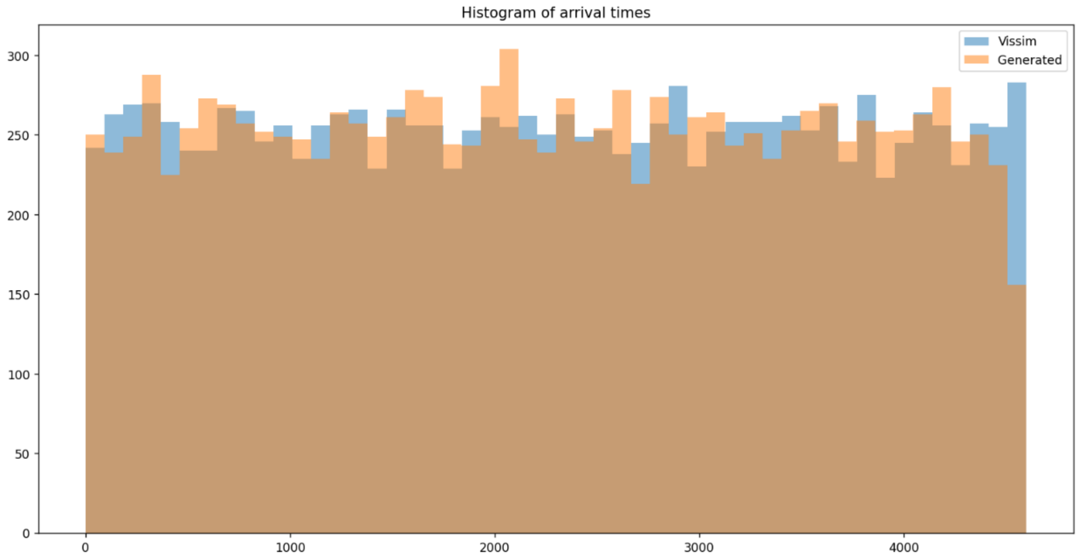


Figure 20 Histogram of Arrival Times

The KS test statistic of 0.0133 with a p-value of 0.209 suggests no significant difference in vehicle arrival times between the VISSIM model and the calibrated model.

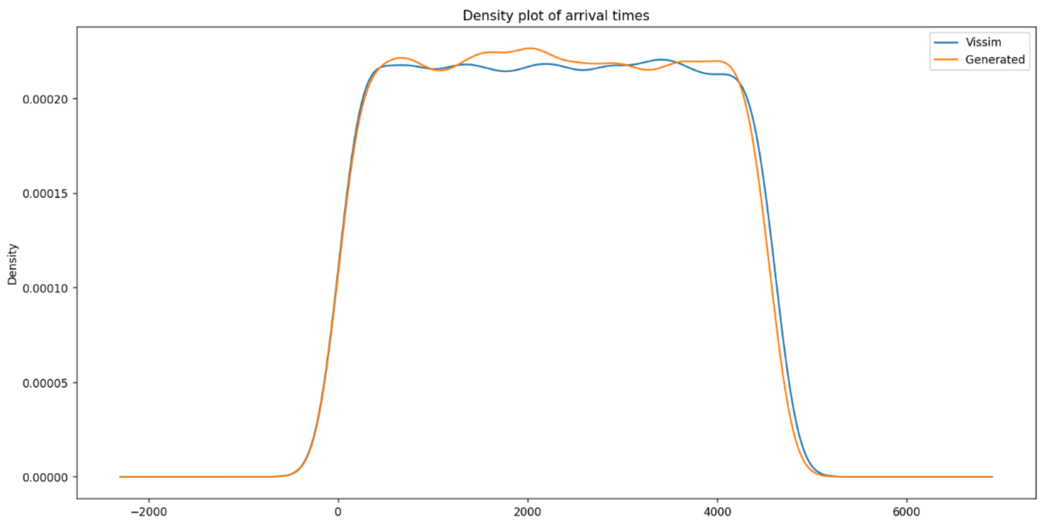


Figure 21 Density Plot of Arrival Times

Shows the probability density of vehicle arrivals, where both VISSIM and generated data closely align, suggesting similar distribution patterns.

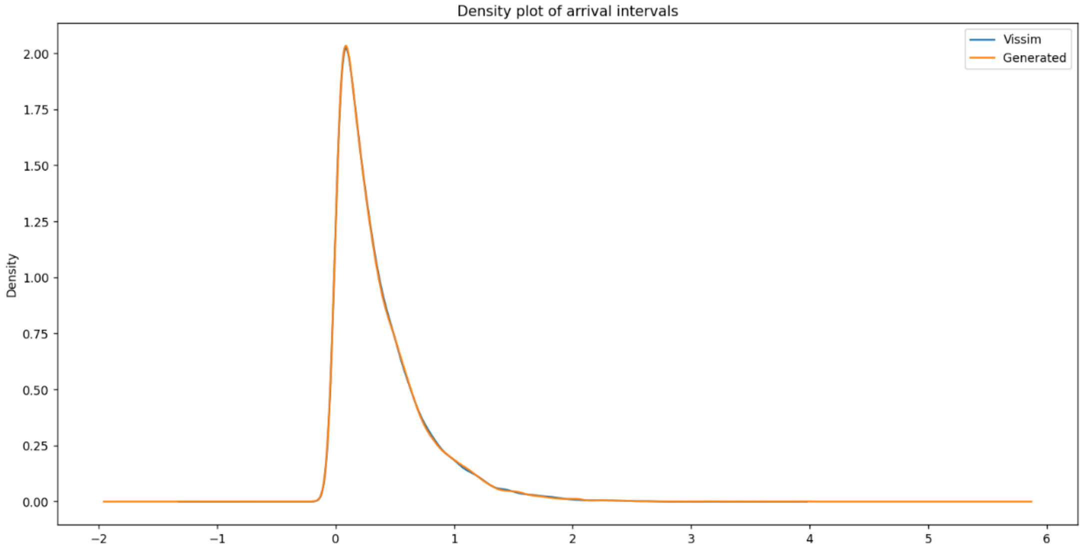


Figure 22 Density Plot of Arrival Times

Compares the density of the intervals, further confirming the uniformity of traffic distribution between the two models.

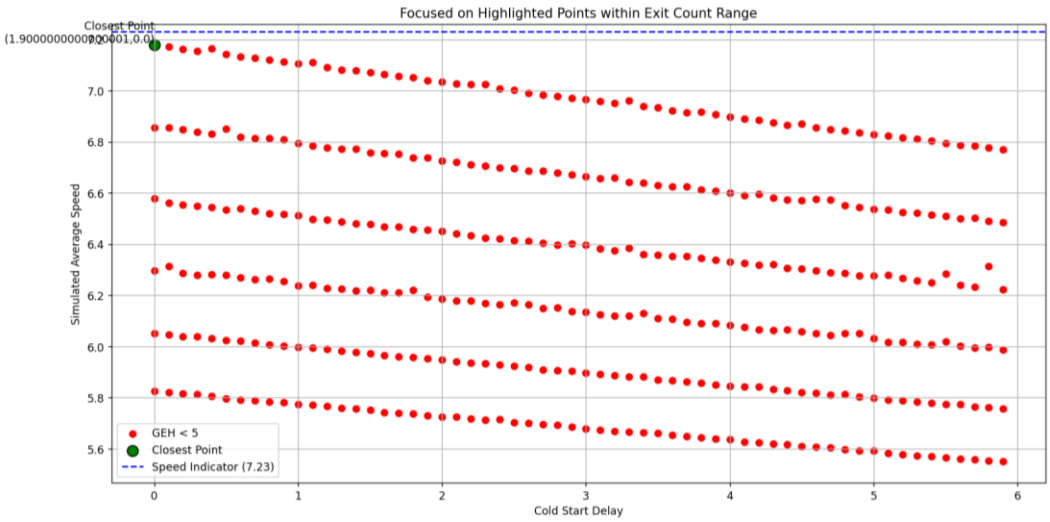


Figure 23 Optimization of Traffic Simulation Parameters with GEH Statistic Constraint

The x-axis represents the Cold Start Delay (CD), and the y-axis represents the Simulated Average Speed. The variation of the Smooth Factor (SM) is reflected in the graph by the discrete clusters of data points showing a stepwise decline in the Simulated Average Speed.

In summary, due to the intricate interplay of the nine parameters governing the Wiedemann 99 driving model within VISSIM, direct mathematical fitting proves impractical. Consequently, the study adopts simulation-based parameter tuning, specifically through adjustments to the Smooth Factor (SM) and Cold Start Delay (CD), achieving a representative emulation of driving behaviors without the constraints of algebraic formulations. This approach effectively captures the complex dynamics of driver interactions in traffic flow simulations.

the input analysis reveals that employing a uniform distribution for vehicle arrivals closely aligns with the VISSIM simulation outputs. Furthermore, adopting a driving behavior model with a Smooth Factor (SM) of 1.9 and a Cold Start Delay (CD) of 0 successfully replicates the Wiedemann 99 model utilized in VISSIM simulations.

5.3 Simulation Experiment Design

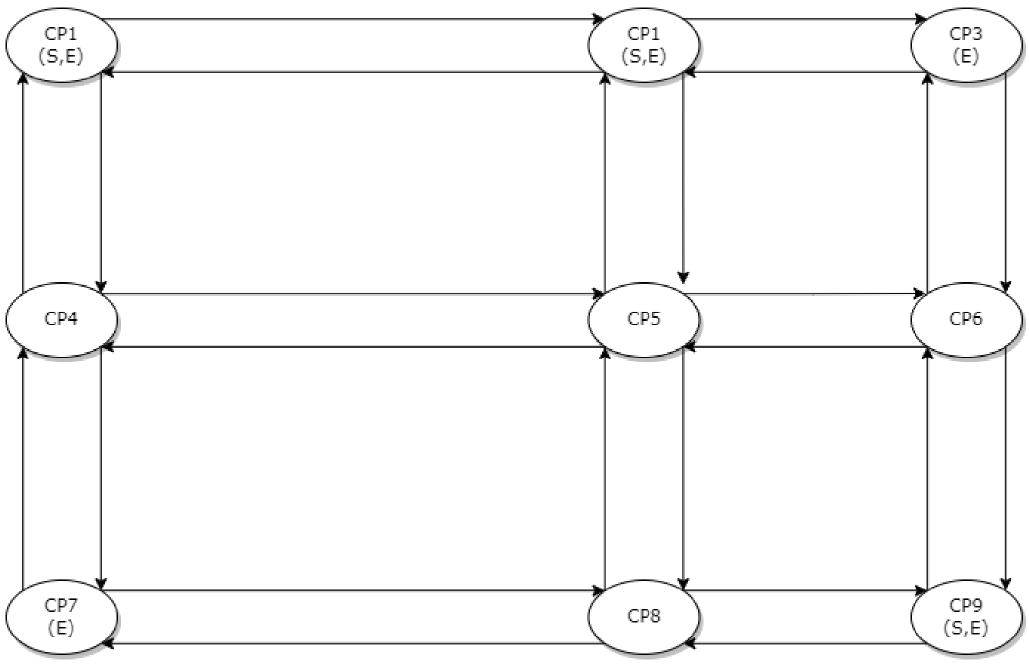


Figure 24 topological graph :S refer to start point while E refer to Endpoint

Inspired by this network's comprehensive representation of traffic scenarios, which mirrors the complexity found in port operations, you have replicated the network in both VISSIM software and PORTML Studio to conduct comparative simulations. This will allow for a direct performance evaluation of your model against established simulation standards.

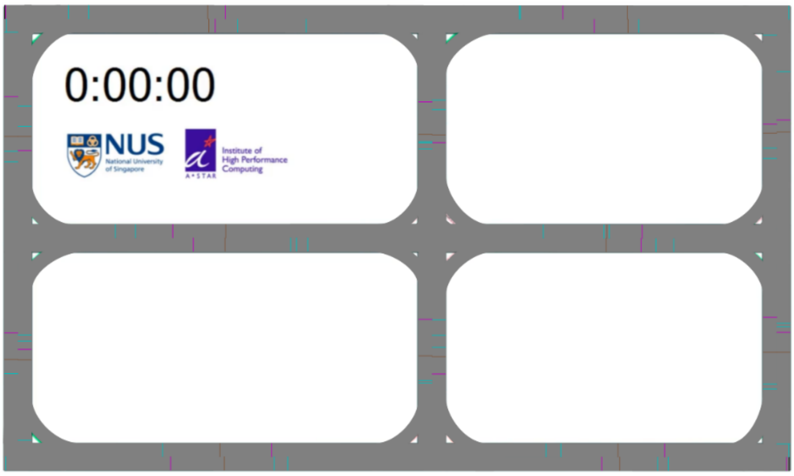


Figure 25 Vissim network

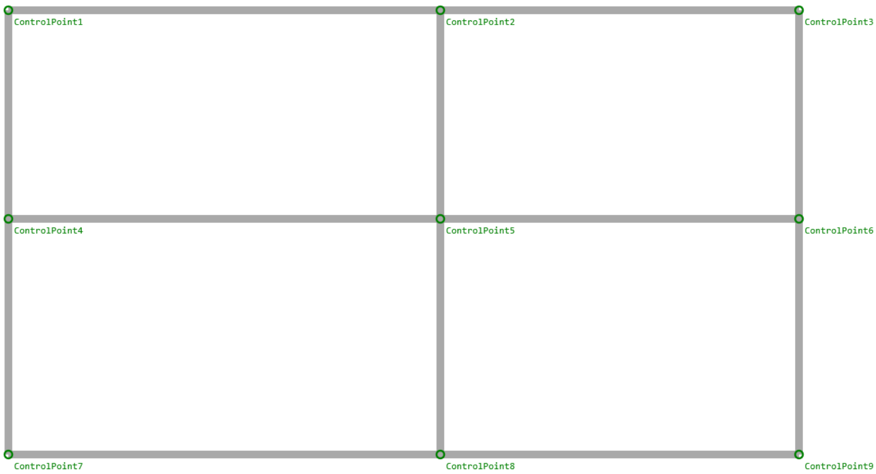


Figure 26 Pathmover network

Input setting:

1. Vehicle Volume: Tested range from 1000 to 2000 vehicles per hour to observe the impact on traffic flow.

2. Desired Speed: Set at 36 km/h to standardize vehicle movement across the network.

3. Vehicle Routing Decision:

- Probability of exiting the network upon reaching a destination: 0.4
- Probability of heading towards the central intersection: 0.4
- All other routing decisions: Equal probability for choosing the next road segment.

4. Time Distribution of Incoming Vehicles: Employed a uniform distribution to model vehicle entries over time.

5. Driving Behavior Models:

- VISSIM: Implements the default Wiedemann 99 model for simulating nuanced driver behaviors.
- PathMover:

SmoothFactor: Adjusted to 1.9, controlling the time headway between vehicles.

ColdStartDelay: Set to 0 seconds, representing the delay from a stop to start condition.

Simulation setting: each simulation run is set for a duration of 4600 seconds, including a warm-up period of 1000 seconds to stabilize the system. Ten simulation runs are conducted for each traffic volume input to ensure the reliability of the results.

5.4 Data Collection Plan

Metrics:

- Total Stay: Time from vehicle entry to exit.
- Total Distance: Sum of all distances covered by vehicles.
- Segment Flow Rates: Hourly vehicle count per road segment.

Purpose:

- To determine system efficiency and identify congestion points.
- To validate the model's reliability using the GEH statistic (comparing segment flow rates against observed data).

5.5 Data Analysis

To calculate the total distance D_{total}, which is a key component in evaluating the performance metric of average speed, the following formula is used:

$$D_{total} = \sum_{j=1}^{N} TD_j$$

Where:

- TD_j is the distance traveled by the j^{th} vehicle.
- N is the number of vehicles that have exited the system, and only vehicles that have exited are included in the calculation.

Figure 27 Formular to Calculate Distance

The average time T_{avg} from vehicle entry to exit, averaged over all vehicles that have exited, is given by:

$$T_{avg} = \frac{1}{N} \sum_{j=1}^{N} (EXT_j - ENT_j)$$

With these definitions, the average speed V_{avg} is then calculated as:

$$V_{avg} = \frac{D_{total}}{T_{avg}}$$

This provides a measure of how quickly vehicles are able to travel through the system on average.

Figure 28 Formular to Calculate Average Speed

In a supplementary analysis, I examined the traffic flows for each road segment at a simulation input of 2000 vehicles per hour. I employed the GEH statistic to measure the agreement between the traffic volumes predicted by our model and those reported by the VISSIM model.

GEH Statistical Formula:

The GEH (Geoffrey E. Havers) statistic is a widely-used measure in traffic modelling to compare the observed and modelled traffic counts. It is calculated using the formula:

$$GEH = \sqrt{\frac{2(M-C)^2}{M+C}}$$

Where:

- M represents the modelled (or predicted) traffic count.
- C represents the observed (or actual) traffic count.

Figure 29 Formular to Calculate GEH

To further validate the relationship between the two models, an independent t-test was conducted on the total average speed under a traffic flow input of 2000 vehicles per hour in the simulation experiments.

Statistical Analysis Process:

1. **Calculation of Means:**
   - Average speed of User Model ($\bar{x}_1$)
   - Average speed of VISSIM Model ($\bar{x}_2$)
2. **Standard Deviation Computation:**
   - Standard Deviation for User Model (s_1)
   - Standard Deviation for VISSIM Model (s_2)
3. **Sample Sizes:**
   - Number of observations in User Model (n_1)
   - Number of observations in VISSIM Model (n_2)

Figure 30 Setting of T-test

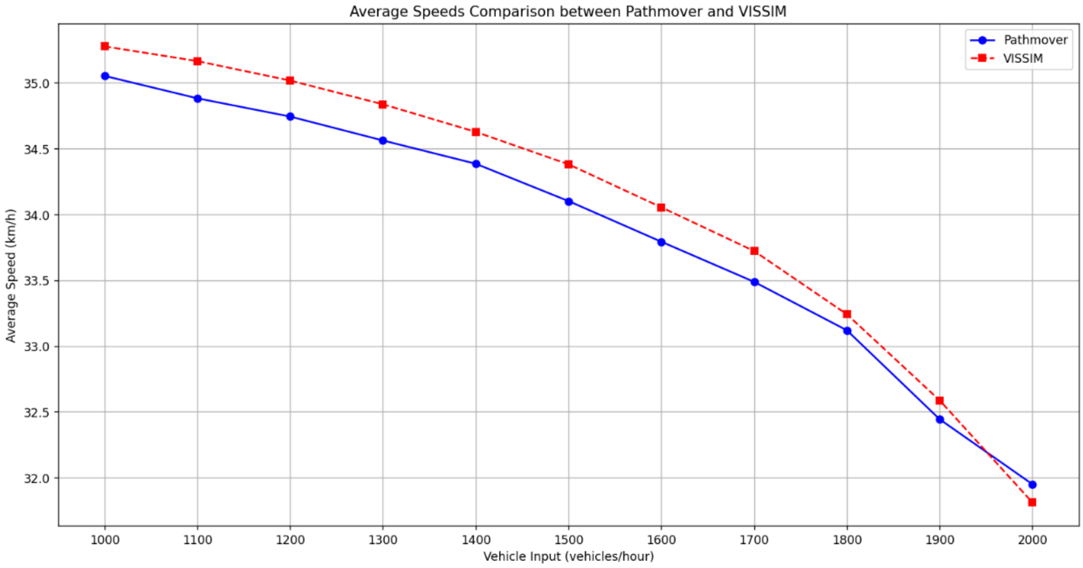


Figure 31 average speeds between the Pathmover and VISSIM

The graph depicts a comparison of average speeds between the Pathmover and VISSIM models across different traffic volumes. Both models demonstrate a similar downward trend with closely matched rates of decline, indicating consistent responses to increasing traffic volumes.

divided the traffic input from 1000 to 2000 into 11 groups and conducted independent t-tests for each group to systematically assess the impact on driving behavior parameters.

Number of tests that passed the significance level of 0.05: 2/11

Group 1: PASSED - T-Statistic: 0.12514693848050976, P-Value: 0.9017943377281261

Group 2: PASSED - T-Statistic: 1.2715942949717138, P-Value: 0.21970271038370323

Group 3: FAILED - T-Statistic: 3.449630556130457, P-Value: 0.002859084056116642

Group 4: FAILED - T-Statistic: 6.003577499082001, P-Value: 1.1187708960409052e-05

Group 5: FAILED - T-Statistic: 11.998483343089779, P-Value: 5.056790280970926e-10

Group 6: FAILED - T-Statistic: 11.389381162625511, P-Value: 1.1643282279732793e-09

Group 7: FAILED - T-Statistic: 14.424060715818582, P-Value: 2.4773262315863345e-11

Group 8: FAILED - T-Statistic: 21.71324557370581, P-Value: 2.3202943243956786e-14

Group 9: FAILED - T-Statistic: 19.62763786107236, P-Value: 1.3320525981834885e-13

Group 10: FAILED - T-Statistic: 24.04051087254494, P-Value: 3.934143343729368e-15

Group 11: FAILED - T-Statistic: 25.669463784154434, P-Value: 1.2477985363578828e-15

Among the 11 groups tested, only Groups 9, 10, and 11 passed the significance level of 0.05, which could suggest that at higher vehicle inputs, the impact of the Smooth Factor and Cold Start Delay parameters on the traffic flow becomes less pronounced in the VISSIM model, possibly due to similar congestion levels.

Since the PathMover model didn't pass the t-test at lower traffic volumes, my assumption is that varying levels of congestion, caused by different traffic volumes, lead to different parameter settings for vehicle driving behavior in VISSIM. To test this hypothesis, I used data with a traffic input of 1000 vehicles per hour to fit the Smooth Factor and Cold Start Delay. By ensuring the road traffic volume difference (GEH) is less than 5, the fitting process found Smooth Factor and Cold Start Delay values closest to the VISSIM average speed to be 1.2 and 1.0, respectively.

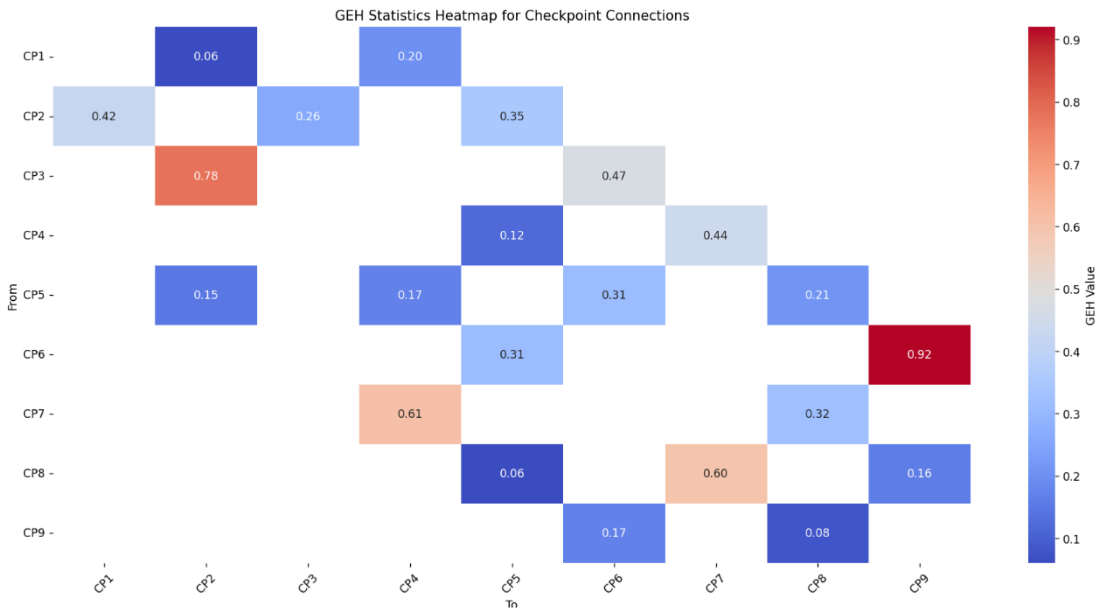


Figure 32 GEH value heatmap

A low GEH value (typically less than 5) indicates a good fit between modelled and observed data, suggesting the model's predictions are reliable.

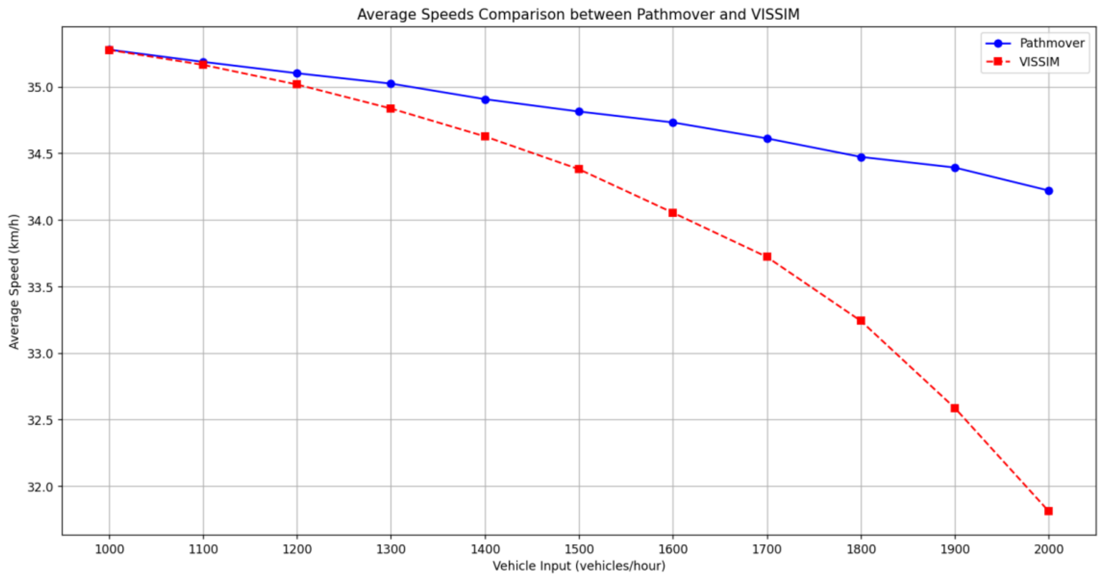


Figure 33 average speeds between the Pathmover and VISSIM after Recalibration

The graph illustrates a comparison of average speeds between the Pathmover and VISSIM models as vehicle input increases from 1000 to 2000 vehicles per hour. Both models show a decrease in average speed as traffic volume rises, with the Pathmover model maintaining a more consistent speed across the range of volumes, while the VISSIM model indicates a sharper decline, especially as traffic input approaches 2000 vehicles per hour

Similarly divided the traffic input from 1000 to 2000 into 11 groups and conducted independent t-tests for each group to systematically assess the impact on driving behavior parameters.

Number of tests that passed the significance level of 0.05: 3/11

Group 1: FAILED - T-Statistic: -11.677935385553907, P-Value: 7.810015877961245e-10

Group 2: FAILED - T-Statistic: -12.435296134203464, P-Value: 2.8375702072629665e-10

Group 3: FAILED - T-Statistic: -8.33876052840473, P-Value: 1.3508819687496314e-07

Group 4: FAILED - T-Statistic: -6.622880459334014, P-Value: 3.228570223814487e-06

Group 5: FAILED - T-Statistic: -6.38497508521516, P-Value: 5.172765714395342e-06

Group 6: FAILED - T-Statistic: -5.977165977772611, P-Value: 1.180992406364419e-05

Group 7: FAILED - T-Statistic: -4.274523804740082, P-Value: 0.00045615408960327143

Group 8: FAILED - T-Statistic: -3.4768130266599604, P-Value: 0.002691767842727683

Group 9: PASSED - T-Statistic: -1.9083252580890608, P-Value: 0.07242846283722824

Group 10: PASSED - T-Statistic: -1.2342780519958574, P-Value: 0.23297157369310384

Group 11: PASSED - T-Statistic: 1.0496588291934332, P-Value: 0.3077645748042189

Among the 11 groups tested, it was Groups 1 and 2 that passed the significance level of 0.05. This outcome suggests that at lower vehicle inputs, the influence of Smooth Factor and Cold Start Delay parameters on traffic flow is not as marked, potentially due to the lower levels of congestion. These findings are consistent with the hypothesis that similar congestion levels result in comparable driving behavior, which can be adequately captured by the t-test.

After establishing that vehicle driving behavior parameters require adjustments for varying traffic volumes, an experiment has been designed to calibrate the 'Smooth Factor' and 'Cold Start Delay' parameters across all traffic volumes encountered in the experiments. This calibration aims to enable the driving behavior parameters to adapt in response to changes in input, ensuring that the model accurately simulates different traffic conditions

To identify the most suitable 'Smooth Factor' and 'Cold Start Delay' settings for vehicle driving behavior within VISSIM, a comprehensive iteration process was employed. Parameters were varied within a range from 0 to 4, incrementing by 0.1 for each iteration. The optimal combination of 'Smooth Factor' and 'Cold Start Delay' was determined based on the condition that it yields the maximum t-value in a t-test, implying the best fit between the simulated traffic flow and observed data.

The range of 0 to 4 for adjusting the 'Smooth Factor' and 'Cold Start Delay' is chosen based on considerations of vehicle performance and driver behavior characteristics. Within this spectrum, increments of 0.1 are used to traverse all possible values, searching for the optimal parameters. The most suitable settings are identified by the criterion of achieving the highest t-value from a t-test, indicating the most accurate alignment of the VISSIM model’s simulated traffic flows with the observed data.

Chapter 6 CONCLUSION AND FUTURE WORK

6.1 Conclusion

The comparative analysis between PathMover and VISSIM demonstrates that while PathMover cannot universally replicate VISSIM's car-following model due to its multifaceted and stochastic nature, it can approximate the VISSIM model's outputs for critical KPIs such as average speed and traffic flow, given a well-defined expected range of traffic volumes. This indicates PathMover's potential for accurately modeling traffic behaviors and its utility in traffic simulation, provided that the model parameters are carefully calibrated according to the specific traffic conditions being studied.

It is important to note that the robustness of the PathMover model lies within its parameter optimization, which, when correctly adjusted, allows it to align closely with VISSIM's performance in simulating key traffic metrics. This suggests a viable approach for employing PathMover in scenarios where high fidelity is required without the need for the complex and detailed parameterization characteristic of the VISSIM model.

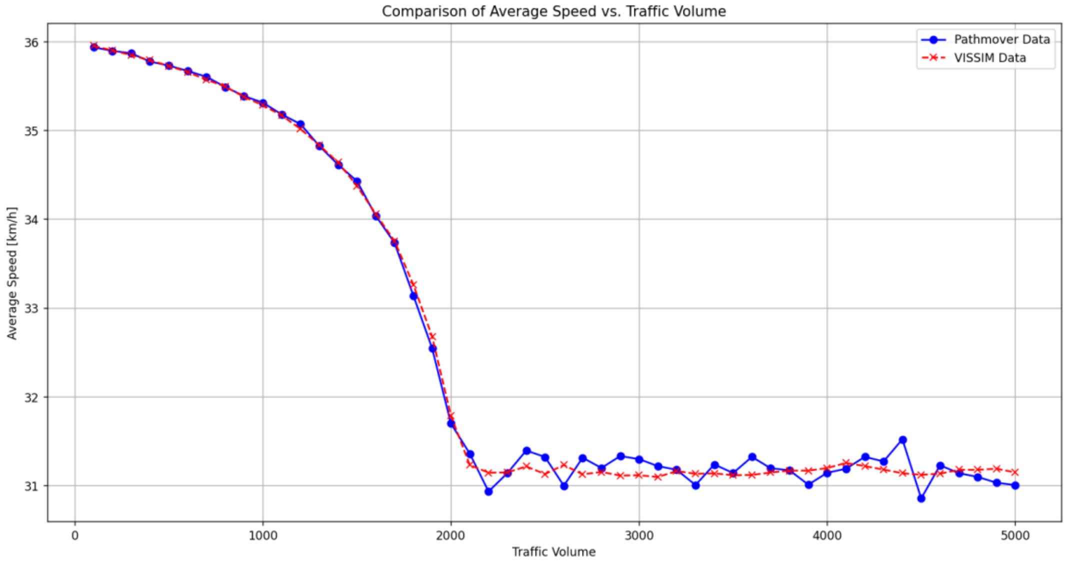


Figure 34 average speeds between the Pathmover and VISSIM after chose best parameters

The graph depicts a comparison of average speeds versus traffic volume for Pathmover Data and VISSIM Data. It can be observed that below a traffic volume of 2000, the curves representing the average speed for both sets of data closely align. This indicates that under less congested traffic conditions, the simulation results from both models are similar, and thus, T-tests conducted within this range are likely to pass, suggesting no significant statistical difference.

However, as the traffic volume exceeds 2000, the curves begin to diverge with increased fluctuations and differences between the datasets. This divergence suggests that the performance of the two models starts to significantly differ under more congested road conditions. High traffic volumes, approaching or exceeding the road capacity, typically lead to unstable traffic flow where various factors—such as differences in driver behavior, road network design, accidents, or road conditions—can cause vehicle speeds to enter a state of random fluctuation.

In such congested conditions, the uncertainty in simulation results increases, leading to T-tests that fail to pass, meaning that the differences in the mean values of the two datasets are statistically significant.

These observations align with traffic flow theory, where the relationship between traffic volume and speed reaches a certain threshold, leading to a decline in the level of service and entry into a "congested region." In this region, the fluctuation in vehicle speeds intensifies, and congestion becomes the dominant factor affecting vehicle speeds.

6.2 Future Work

More Flexible Vehicle Driving Behavior Simulation: Introduce models with finer granularity that can adjust driving behaviors in real-time based on the dynamic conditions of the vehicle and its surrounding environment. For instance, the model could simulate the behavioral differences among various drivers, reflecting the diversity of real-world driving behaviors.

Enhanced Physical Interaction Between Vehicles and Roadways: Currently, vehicles are modeled as point masses, which limits the realism of the simulation. Improving the model to better represent the physical dimensions of vehicles and their actual interaction with the roadways, such as maintaining distances between vehicles, overtaking, and merging behaviors, would make the model more closely mirror actual road conditions.

Visualization and Animated Demonstration: Develop a visualization tool that can not only display simulation results but also demonstrate the simulation process in real-time through animation. This would make the analysis of the model more intuitive and facilitate the understanding of complex traffic flows and driving behaviors. Moreover, such a tool would assist in identifying and interpreting specific patterns or anomalies that emerge during the simulation.

References

Goti, A. (2010). Discrete Event Simulations.

Ingemansson, A., Bolmsjö, G., & Harlin, U. (2002). A survey of the use of the discrete-event simulation in manufacturing industry.

Fishman, G. S. (2001). Discrete-event simulation.

Thesen, A. (1978). Discrete Event Simulation Programming.

Cocchi, D., Miniati, R., Frosini, F., Avezzano, P., Iadanza, E., Curiardi, M., & Belardinelli, A. (2016). Discrete event simulation-based approach for hospital services development and monitoring.

Narain, S., & Chadha, R. (1995). Symbolic Discrete-Event Simulation.

Pidd, M., & Cassel, R. A. (2000). Using Java to develop discrete event simulations. Journal of the Operational Research Society.

Zhou, C., Lee, L. H., Chew, E. P., & Li, H. (2017). A Modularized Simulation for Traffic Network in Container Terminals via Network of Servers with Dynamic Rates. In 2017 Winter Simulation Conference (WSC). IEEE.