Ben-Gurion University of the Negev
Faculty of Engineering Science
School of Electrical and Computer Engineering
Dept. of Communication Systems Engineering

Fourth-Year Engineering Project
Final Report

Autonomous Driving Using Swarms

Project Number:	p-2023-003
Students:	Noy Ella 312252257, Maayan Ben-Gal 311438733
Supervisors:	Prof. Michael Segal
Submitting Date:	30/07/2023

Table of Contents:

1. Project Summary

I. Project Summary (English):

Autonomous Driving Using Swarms

Students Names: Ben-Gal Maayan, Ella Noy

bengalm@post.bgu.ac.il

The future of transportation depends on the development of autonomous vehicles that are required to function in real-time and make decisions according to traffic conditions, when contacting an external factor may take valuable time and cause accidents in cases of a communication failure.

The goal of the project is to develop an algorithm that creates swarms of vehicles with vehicle-to-vehicle (V2V) communication in a decentralized manner. Each swarm is a group of vehicles moving together as one, according to their destination and the vehicle's location on the road in real time. The algorithm must organize the vehicles with a minimal number of steps until the swarms are stabilized.

The concept of communication without an external server is innovative in the context of autonomous vehicles. Beyond the idea of communication without a human factor, the idea of swarms can improve the decision-making of a vehicle.

The method will include a fixed number of vehicles on a road with more than one lane, each vehicle having a color according to its destination. Each car can communicate with vehicles up to 100 meters, to identify its swarm mates. After dealing with this, edge cases will be considered to improve the algorithm and adapt it to reality. The main goal is to develop a single initial algorithm that could be installed on all autonomous vehicle systems and will organize the moving vehicles into swarms in a minimum time.

Keywords – Innovative transportation, Autonomic cars, Autonomous Driving, Distributed algorithms, V2V, Swarms, real-time, SUMO (vehicle simulation software), OMNeT++ (communication simulator).

II. Project Summary (Hebrew):

נהיגה אוטונומית בעזרת להקות

שמות הסטודנטים: בן-גל מעיין, אלה נוי

bengalm@post.bgu.ac.il

עתיד התחבורה תלוי בפיתוח רכבים אוטונומיים הנדרשים לתפקד ולקבל החלטות בזמן-אמת בהתאם לתנאי התנועה והשטח, כאשר יצירת קשר עם גורם חיצוני כגון חדר בקרה עלול לגזול זמן יקר ולגרום לתאונות במקרה של נפילת תקשורת.

מטרת הפרויקט לפתח אלגוריתם המייצר להקות רכבים הנעים בכביש בעזרת תקשורת בין רכבים (V2V) באופן מבוזר. כל להקה היא קבוצת רכבים הנעה יחד כרכב אחד, על פי יעד הנסיעה בהתאם לרכבים הממוקמים על הכביש בזמן אמת. על האלגוריתם לארגן את הרכבים כך שכמות השלבים עד להתייצבות הלהקות בכביש תהיה מינימלית.

הרעיון של תקשורת ללא שרת חיצוני הוא חדשני בהקשר של רכבים אוטונומיים. מעבר לרעיון התקשורת ללא גורם אנושי, רעיון הלהקות יכול לשפר משמעותית את תהליך קבלת ההחלטות של הרכב.

שיטת העבודה תכלול מספר קבוע של רכבים על כביש בעל למעלה מנתיב אחד, כאשר לכל רכב יש צבע על פי יעדו הסופי. לכל מכונית ישנה יכולת לתקשר עם רכבים עד 100 מטר, על מנת לזהות את חבריו העתידיים ללהקה. לאחר התמודדות עם מצב זה נוסיף מקרי קצה במטרה לשפר את האלגוריתם ולהתאימו למציאות. מטרת העל היא לפתח אלגוריתם ראשוני יחיד שיותקן על כל מערכות הרכבים האוטונומיים וייבצע את ארגון הרכבים הנעים ללהקות בזמן מינימלי.

מילות מפתח: תחבורה חדשנית, רכבים אוטונומיים, נהיגה אוטונומית, אלגוריתמים מבוזרים, V2V, להקות, זמן אמת, SUMO (תוכנת סימולציית רכבים), OMNeT++ (תוכנת סימולציה לרשתות תקשורת).

2. Introduction

2.1. Motivation

Autonomous vehicle known as a Self-driving vehicle, meaning a vehicle that can drive in its environment (in the case of a car - road) using auxiliary sensors such as distance and motion sensors, video cameras, radars, etc., and perform tasks according to navigation protocol without human intervention [3]. The main goal of the autonomous car is to go wherever a traditional car goes and do everything an experienced human driver does. Nowadays the Society of Automotive Engineers (SAE) [4] defines 6 levels of driving automation from Level 0 (No automation) to Level 5 (full automation). Those levels are divided into two parts: in levels 0-2 the human monitors the driving environment and in levels 3-5 the automated system monitors the driving environment [5]. As of 2023, autonomous cars are still in the planning, development, and testing stages and many companies such as Google, Mobileye, Tesla, General Motors, and more are in different stages of development and at different levels of automation. Today, fully autonomous cars are not yet available to the general public meaning there are only semi-autonomous cars today on the market [3].

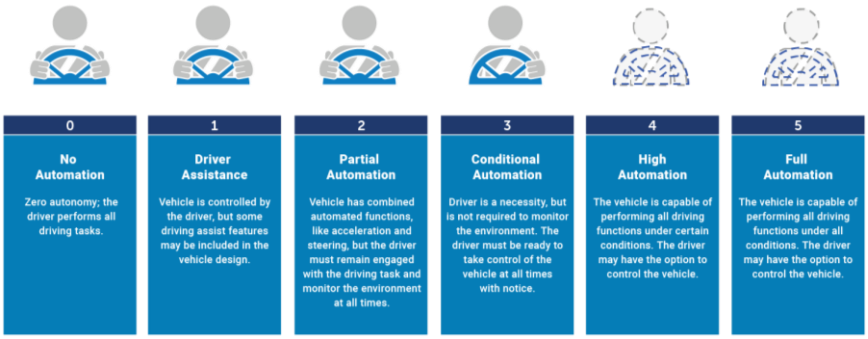

Figure 1: levels of driving automation [5]

Even a fully autonomous vehicle needs to communicate with an external factor such as a remote server, this server can provide information on the road condition, traffic, navigation, and more. The communication between a vehicle and the server will provide the vehicle with the information required to continue the drive and make the most efficient decisions. This type of communication might not be scalable or even stable. Using V2V (vehicle-to-vehicle) communication can reduce the amount of traffic on the server, and still reserve the information transportation to the vehicles. The V2V communication needs to be used carefully so it won't create overload on the vehicles and still be useful. The idea of vehicle swarms is shown in [1] using V2V communication, but maybe we can improve it.

2.2. Project Goal

The project goal is to develop an initial algorithm that maps autonomous vehicles into swarms using vehicular ad hoc networks (VANET) in real-time, in minimum steps. The swarms' partition is based on the vehicle's destination and should be congested in a minimum time.

2.3. Past Work and Our Approach

A previous approach to solve this problem was developing an algorithm that creates swarms according to the physical geographical proximity of the vehicles in real-time**.** That is, the cars on the road will communicate and connect to a collective according to their position on the road at a minimum distance. In prof. Michael Segal article [1] a D-Cut algorithm is presented that generates clusters of vehicles according to their position on the road and chooses a temporary leader for the swarm each time. The advantage of this approach is that the swarm creation and stabilization time is fast. Another advantage is that there is a limit on the number of vehicles in each cluster, which gives a simple intra-swarm communication protocol. A disadvantage of this approach is the instability of the swarm in a situation where a vehicle leaves the group when it reaches its destination. Since the vehicles in the swarm do not necessarily reach the same destination, the group will not remain intact throughout the route.

Another approach is to develop an algorithm to create swarms when the communication between the moving cars is not Peer-2-Peer but through the cloud. The information about each car (velocity, location on the road, etc.) is stored in the cloud, which is updated immediately and shared with all vehicles on the road [2]. The advantage of this approach is an efficient use of memory, instead of each car containing all the information about all the cars and the road conditions, the information is saved once in the cloud with quick access to all the cars. The disadvantage of this approach is the reliance on an external factor (cloud) in real-time systems. The access to cloud memory, however fast, can sometimes be harmful in real-time systems. In case of server crash, all the cars on the road are immediate disabled and require human intervention.

Our approach is to develop an algorithm that maps the vehicles by their destination in order to create swarms that will stay connected for a longer period of time. Therefor reducing the network traffic due to lower 'joining/leaving the swarms' messages.

3. Technical Specifications

To simulate and test the algorithm there are a few basic needs that the simulation should be able to perform, first is the ability to create vehicle traffic in fairly realistic conditions, creating V2V communication according to communication protocols, and more. Unfortunately, one simulator can't fulfill all the requirements, therefore, we use two simulators, framework, API, and protocol to perform the simulation.

3.1. Programs, Framework, and API

3.1.1. Simulation of Urban Mobility (SUMO)

SUMO is an open-source, microscopic, multi-modal traffic simulation. [7] SUMO can simulate a variety of features such as Automated Driving - Integrate automated vehicles in traffic simulations and equip vehicles with a transition of control (ToC) device. Vehicle Communication - By coupling to a communication network simulator. Traffic Management - Model video detectors and induction loops to manage traffic interactively by controlling speed limits, traffic lights, and vehicle behavior. Multimodal Traffic - Combine different modes of transportation and simulate cars, buses, trains, bicycles, pedestrians, public transport, etc. These and more are the useful features SUMO can offer. [8]

Although SUMO can simulate a variety of physical scenarios, it can't simulate V2V message streams or any type of communication, therefore we decided to create the physical terms in SUMO and combine the work with other simulators.

3.1.2. Objective Modular Network Testbed in C++ (OMNeT++)

OMNeT++ is an extensible, modular, component-based C++ simulation library and framework, primarily for building network simulators. “Network” means in a broader sense that includes wired and wireless communication networks, on-chip networks, queueing networks, and so on. Domain-specific functionality such as support for sensor networks, wireless ad-hoc networks, Internet protocols, performance modeling, photonic networks, etc., is provided by model frameworks, developed as independent projects. OMNeT++ offers an Eclipse-based IDE, a graphical runtime environment, and a host of other tools. There are extensions for real-time simulation, network emulation, database integration, System integration, and several other functions. [9]

Using OMNeT++ can only create the communication network, and not enough to create the physical needed terms for our project. This way when SUMO is responsible for the 'physical layer' to be simulated, OMNeT++ is responsible for the 'network layer' to be simulated. To connect these two simulators there is a need for both API and framework to make them able to work together.

3.1.3. Vehicles in Network Simulation (Veins) Framework

Veins is an open-source framework for running vehicular network simulations. It is based on two well-established simulators: OMNeT++, an event-based network simulator, and SUMO, a road traffic simulator. It extends these to offer a comprehensive suite of models for IVC (Inter-Vehicular Communication) simulation. [10]

Using Veins can create realistic simulations where network communication in OMNeT++ interacts with vehicle movements and traffic conditions simulated by SUMO. The framework synchronizes the two simulators, exchanging data at specific intervals to maintain a consistent state throughout the simulation. Veins facilitate the exchange of information, enabling the evaluation of communication protocols and traffic management strategies in these integrated simulations. [10]

3.1.4. Traffic Control Interface (TraCI) API

TraCI API gives access to a running road traffic simulation, it allows to retrieve values of simulated objects and to manipulate their behavior "on-line". [11] TraCI is a getter-setter API that enables updating and receiving information in run-time.

TraCI API acts as a bridge, allowing OMNeT++ to interact with SUMO's traffic simulation and control features. It facilitates bidirectional communication, enabling OMNeT++ to query and control the traffic state and vehicle movements within the SUMO simulation environment. Through TraCI API, researchers can design and execute complex vehicular network simulations in OMNeT++, while simultaneously incorporating SUMO's realistic traffic and mobility models. This integration offers a powerful platform for evaluating communication protocols, traffic management strategies, and various intelligent transportation systems scenarios more accurately and dynamically [11].

In summation, SUMO is used to create the vehicles' network, OMNeT++ is used to create the communication network, and both Veins framework and TraCI API help to connect the two simulators.

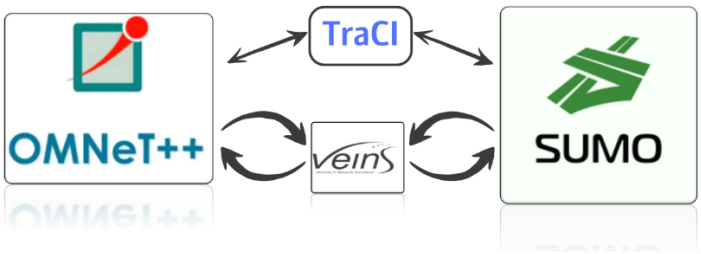

Figure 2: Simulators, Framework, and API's Topology

3.2. Dedicated Short-Range Communications (DSRC) Protocol

Dedicated short-range communications (DSRC) is a technology for direct wireless exchange of vehicle-to-everything (V2X) and other intelligent transportation systems (ITS) data between vehicles, other road users (pedestrians, cyclists, etc.), and roadside infrastructure (traffic signals, electronic message signs, etc.).

DSRC, which can be used for both one- and two-way data exchanges, uses channels in the licensed 5.9 GHz band. DSRC is based on IEEE 802.11p. [12]

By integrating Veins with SUMO, you can use DSRC communication models in your OMNeT++ simulations. Veins offer predefined communication protocols and mechanisms that facilitate DSRC message exchange between vehicles. Researchers and developers can leverage this capability to investigate and evaluate various DSRC-based applications and communication protocols in the context of vehicular networks.

3.3. Changes Occurred During the Year

As represented in the project's previous reports, the initial plan was to use only the SUMO simulator and DSRC communication protocol. During the year we understood as mentioned that SUMO can only simulate the physical vehicles network, and is unable to simulate the communication network, therefore we found the use of OMNeT++ simulator and searched the way to make both simulators work together.

Connecting SUMO and OMNeT++ got us using Veins and TraCI, which was significantly different from the original plane.

4. Approach and Design

The project was built on an algorithm, and its development involved defining a few ground rules and determining what are measurable variables. Our point of view was focused on the message amount and the swarm convergence time, this way we calculated the theoretical equation for the algorithm to detect its efficiency.

4.1. Design and Algorithm

There are some basic ground rules we define for our convenience, the first is that there are two types of communication that the vehicles can do – V2V (vehicle-to-vehicle) communication and V2S (vehicle-to-server) communication. V2V communication is 100 meters radius for both broadcast and unicast messages. The assumption is that V2S communication (with or without partition to swarms) is required once in 10 seconds, meaning the vehicle needs to update the server every 10 seconds and wait for the required update response from the server. Assuming there is no delay due to distance for both V2V and V2S communication, and that V2S can be for any distance required.

The swarm is defined as a group of vehicles with the same destination that moves together, the swarm includes 3 roles:

Swarm Leader – every swarm has one leader that makes the swarm decisions, the leader is the only vehicle from the swarm that communicates with the remote server, and it's responsible to update the other vehicles in the swarm with the relevant information.

Swarm Gateway (GW) – to communicate with vehicles from out the swarm, there are two vehicles in a GW position, they will be located at the front and back edges of the swarm, their role is to report on the existence of the swarm and receive messages from vehicles out of the swarm that wishes to join it.

Swarm Member – a vehicle that doesn't have a special role in the swarm, it can route a message from/to the leader or from/to the GW vehicles.

Figure 3: Swarm leader (mark with the crown), gateway (marks in 'GW'), and members (unmarked).

The full algorithm scheme is represented in Fig. 4. Its explanation will follow the scheme.

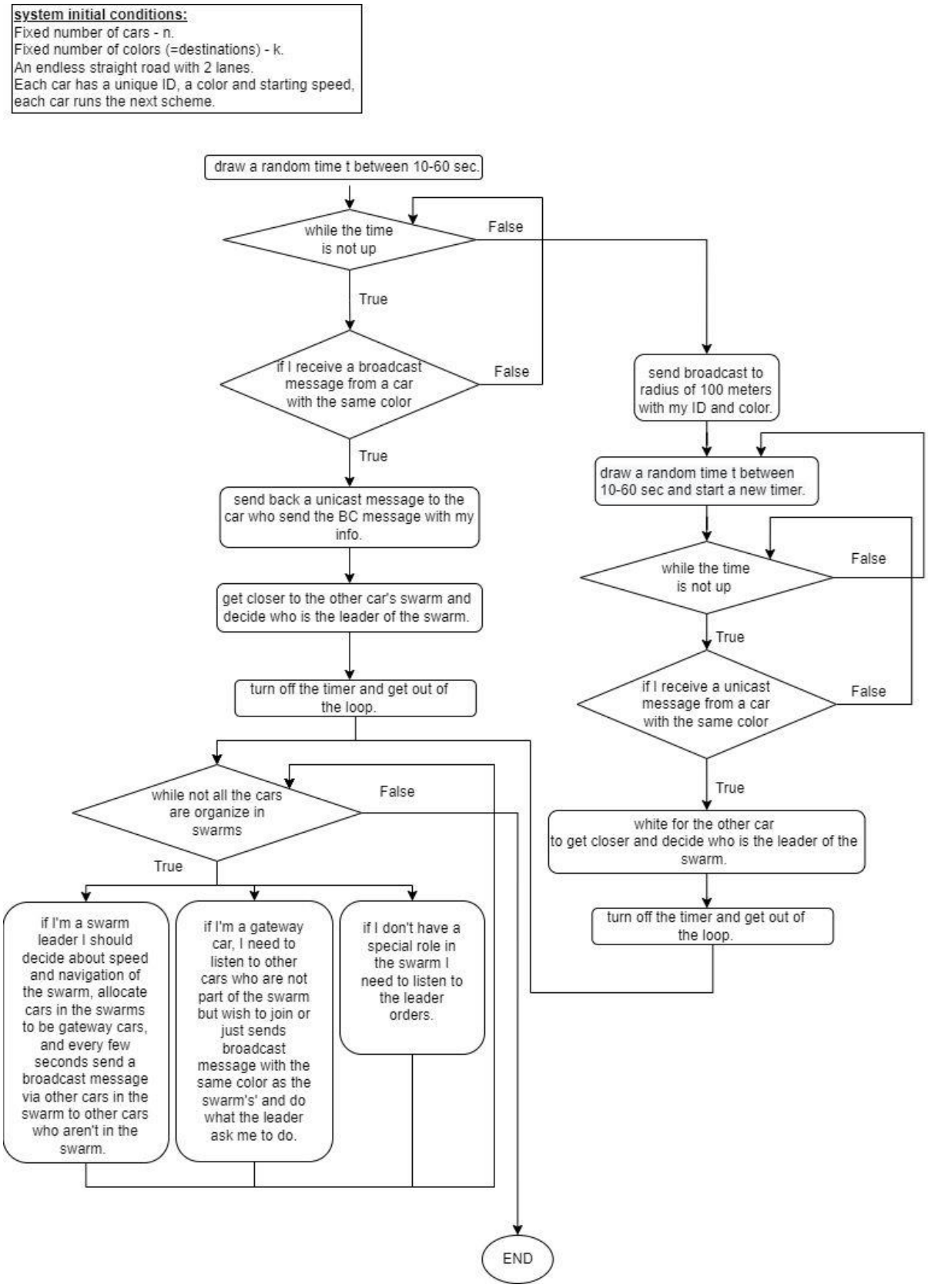

Figure 4: Algorithm Scheme

In the initial system condition represented in Fig. 5, there is a 2-lane loop road with n vehicles and k colors (assuming k<n) that represents different destinations. Each vehicle has an ID, random location, and speed. Notice that the vehicles do not leave or enter the topology during the simulation time. We also assume all the vehicles are congested into swarms before arriving at their destination.

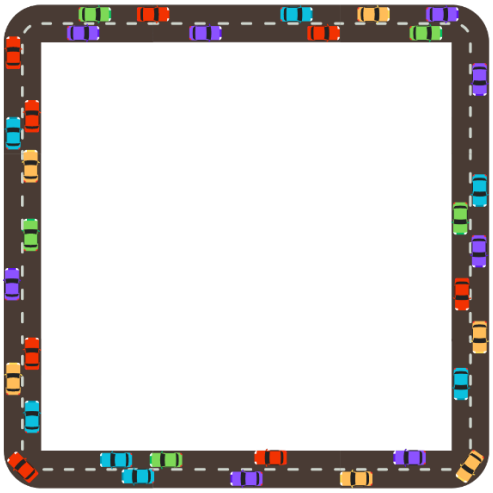

Figure 5: The initial state of the algorithm.

As the algorithm starts every vehicle randomly draws time, up to 30 seconds, and send a broadcast message to 100 meters radius. The first two broadcast messages are up to 30 seconds, after words the broadcast messages frequency is up to every 60 seconds. Broadcast messages include the vehicle's details such as ID, color (= destination), current location, current speed, and any other required information.

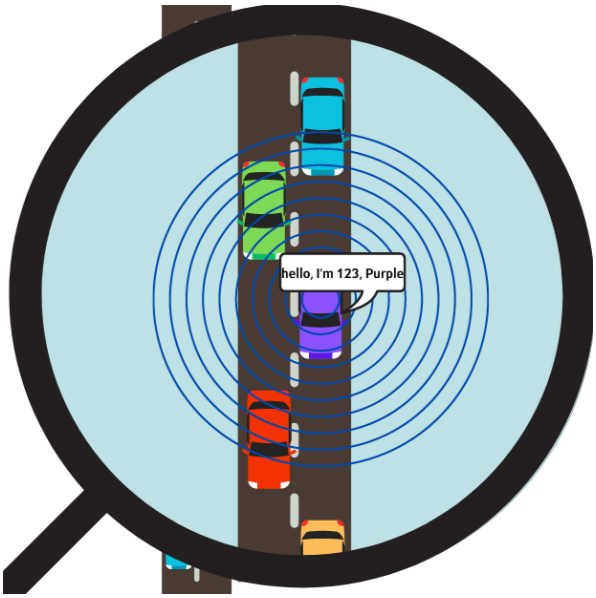

Figure 6: Demonstration of broadcast message sending.

While waiting to send the next broadcast message every vehicle listens to received messages from other vehicles. When identifying other vehicles with the same color, it decides to communicate with the other vehicle with the goal to join into a swarm. In Fig.7. shown that the red vehicle gets messages from few vehicles but decides to communicate with 345 which is also red.

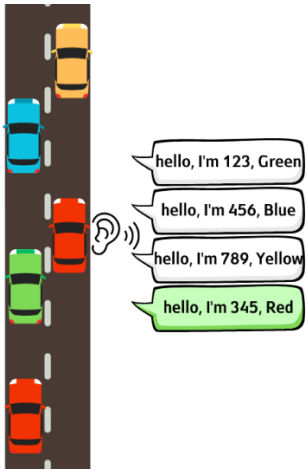

Figure 7: Demonstration of broadcast messages receiving.

To join into a swarm the two vehicles perform a 3-ways handshake, to ease the explanation of this process lets view Fig.8. There are two vehicles, number 111 (referred to vehicle 1) and number 345 (referred as vehicle 2), both red. Vehicle 1 starts the communication by sending message #1 which includes vehicle 1 details, and a joining request. Vehicle 2 returns message #2 with its details, joining confirmation, and the leader of the new swarm (arbitrarily decided as the vehicle with the higher ID, in this case vehicle 2). Now vehicle 1 replays with message #3 with acknowledge of the swarm and its leader. Similar 3-ways handshake might happen when two vehicles when one (or both) is already part of a swarm.

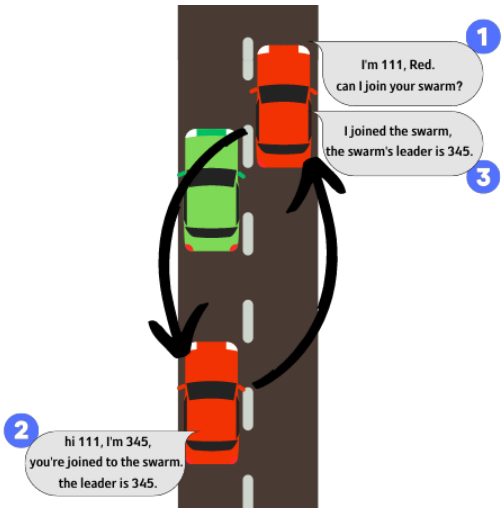

Figure 8: Demonstration of 3-ways handshake.

While not all the vehicles are in swarms, the algorithm continues. To identify the remain single vehicles, there are two types of vehicles sending broadcast messages, single vehicles and a GW vehicle as represented in Fig.9. When single vehicles send broadcast as represented before, the GW vehicles send broadcast including in addition to their details, the swarm details such as color, swarm ID, and leader ID.

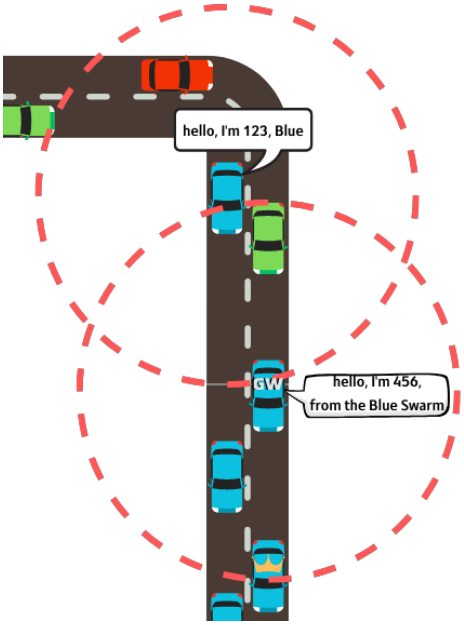

Figure 9: Individual vehicle vs. GW vehicle broadcast sending.

Finally, when all the vehicles congest into swarms, the algorithm is done!

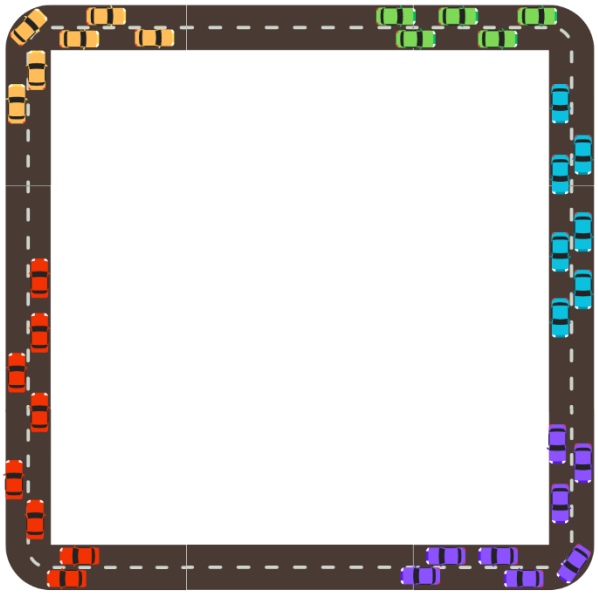

Figure 10: Final required state.

4.2. Mathematical Approach and Calculations

To indicate the requested results there was a need for numerical estimation, to calculate such estimation the first step is define variables, define numeric values if possible and identify the measured variable.

The following list define all the important variables, their numeric value, and their meaning:

- n – number of vehicles, can be any value needed.
- k – number of swarms\possible destinations, can be any value needed, but must be less or even to n.
- VS – number of messages of Vehicle-Server communication. Assuming the vehicle needs to send update message to the server every 10 seconds and receive respond immediately, meaning $6\left[\frac{msg}{min}\right]$ from the vehicle to the server and $6\left[\frac{msg}{min}\right]$ from server to vehicle the VS value is – $12\left[\frac{msg}{min}\right]$
- BC – number of broadcast messages from vehicle to 100m radius, after the first two broadcast messages - $1\left[\frac{msg}{min}\right]$
- BC_{start} - number of broadcast messages from vehicle in the first minute, meaning drawing twice up to 30 seconds - $2\left[\frac{msg}{min}\right]$
- UC – number of messages in the 3-ways handshake between 2 vehicles - $3[msg]$
- MC – number of update messages from the leader to its swarm, meaning a message after every update from the server to the leader - $6\left[\frac{msg}{min}\right]$
- GW – number of broadcast messages from gateways of a swarm to the other vehicles, these messages frequency is like regular broadcast messages - $k * BC = k * 1\ \left[\frac{msg}{min}\right]$
- T – time that takes all the vehicles to reach their destination.
- t – congestion time – the time it takes all the vehicles to congest into swarms. This variable will be the main measure variable.

After defining the important variables, we need to build an equation. Before calculating the equation of the algorithm, let us review the scenario where there is no use of partition to swarms. In such case the messages number will be constant and depends only on T as follows:

$$M_{no\ swarms} = n \cdot VS \cdot T$$

Equation 1

To build the full equation we split the algorithm into three parts – before congestion, during congestion, and after congestion as follows.

Before congestion: this part calculation is for two possible scenarios, if $0 < t < 1[min]$ in this case we can refer only to BC_{start} , but if $t \geq 1[min]$ there is both BC_{start} and BC that needs to be taken into consideration.

Let $m_1(t)$ equation be the number of messages before congestion, therefore:

$$m_1(t) = \begin{cases} n \cdot t(VS + BC_{start}), & 0 < t < 1[min] \\ n(t \cdot VS + BC_{start} + BC(t-1)), & t \geq 1[min] \end{cases}$$

During congestion: this part refers to the number of messages needed to congest, let us mark it with m_2:

$$m_2 = UC \cdot n$$

After congestion: this part refers to the time from t until T, let us mark it with $m_3(t)$:

$$m_3(t) = k(T-t)[MC + BC + VS]$$

At last, all we should do is to add up all the stages and receive the final full equation as follows:

$$M(t) = m_1(t) + m_2 + m_3(t)$$

$$\boxed{M(t) = \begin{cases} n \cdot t(VS + BC_{start}) + UC \cdot n + k(T-t)[MC + BC + VS], & 0 < t < 1 \\ n\big(t \cdot VS + BC_{start} + BC(t-1)\big) + UC \cdot n + k(T-t)[MC + BC + VS], & t \geq 1 \end{cases}}$$

Equation 2

Inserting the numeric values, we will receive:

$$\boxed{M(t) = \begin{cases} (15n - 19k)t + 2n + 19kT & for\ 1 \leq t \\ (14n - 19k)t + 3n + 19kT & for\ 0 < t < 1 \end{cases}}$$

Equation 3

5. Results

It is shown from the final representation of $M(t)$ (in Eq.2) that it depends on t, T, n, k assuming T is constant, and t is the measured variable, there are two scenarios possible: first the number of vehicles is constant and the number of swarms changes, second the number of swarms is constant and the number of vehicles changes.

In Fig.11. and Fig.12. is the scenario when n is constant and k changes.

When comparing the two conditions - division into swarms (represented in Eq.2.) and no division into swarms (represented in Eq.1.) - interesting patterns emerge, shedding light on the dynamics of the system. The phenomenon becomes evident as we analyze the convergence of the meeting points of the two graphs. Surprisingly, despite the variation in the number of swarms in each scenario, a convergence of the meeting points appears, occurring around 7.83 minutes. This convergence point hinting at a critical moment in the results' behavior.

Even when we subject the system to drastic shifts in the value of K, ranging from as low as 5 to as high as 50, the convergence point remains stable and maintains its presence around the 7.83-minute mark. This remarkable resilience of the convergence point suggests that there might be underlying principles governing the system's behavior that transcend the specific number of swarms.

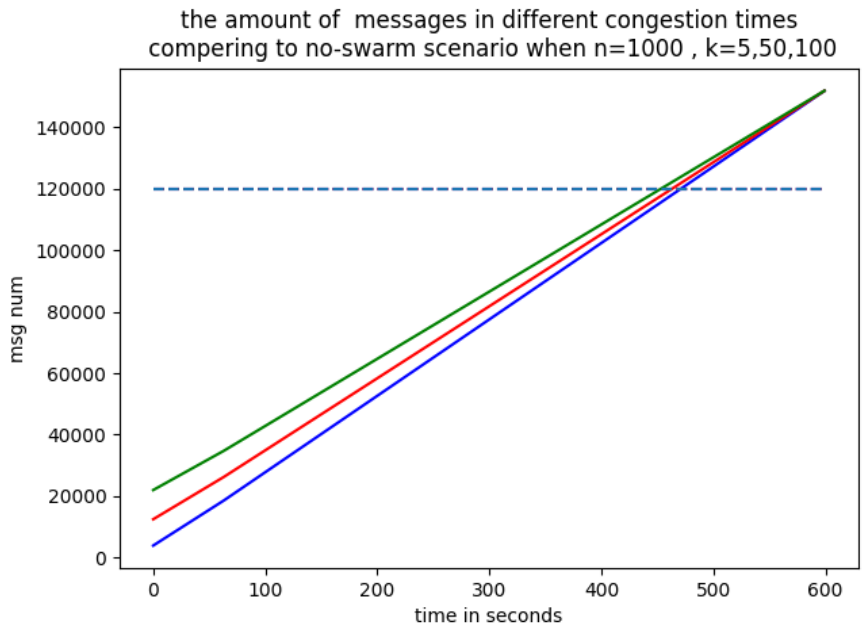

Figure 11: Graph of messages number in different congestion times compered between $M(t)$ and $M_{no\ swarms}$ when n = 1000 and k = 5,50,100.

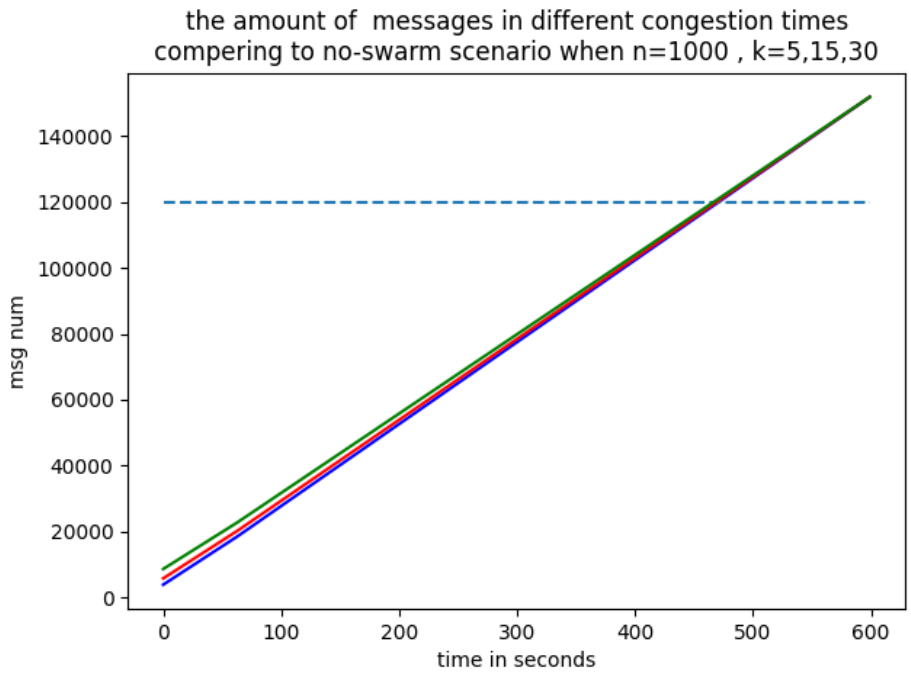

Figure 12: Graph of messages number in different congestion times compered between $M(t)$ and $M_{no\ swarms}$ when n = 1000 and k = 5,15,30.

Fig.13. and Fig.14. is the scenario when n is constant and k changes.

In the current investigation, the focus shifts to examining the impact of varying vehicle numbers (n) while keeping the number of swarms (k) constant. By carefully analyzing the results, a pattern emerges when comparing scenarios where vehicles do not divide into swarms with those where they do, all with different quantities of vehicles. Remarkably, despite the changes in the number of vehicles within each case, we consistently observe the convergence of the meeting points between the two corresponding graphs at approximately 7.83 minutes. This phenomenon remains consistent even when we extend the examination to encompass a range of 'n' values, spanning from 1000 to 2000, implying a consistency in the system's behavior.

Notably, the investigation further explores the dynamics by varying the number of swarms, with an increase from 5 (in Fig.14) to 50 (in Fig.13). Surprisingly, the graphs continue to converge to the same meeting point at 7.83 minutes, showcasing the independence of this critical convergence on the specific number of swarms present. These findings offer compelling evidence that the behavior of the system and the resultant convergence are governed by fundamental principles that transcend both the number of swarms and the quantity of vehicles.

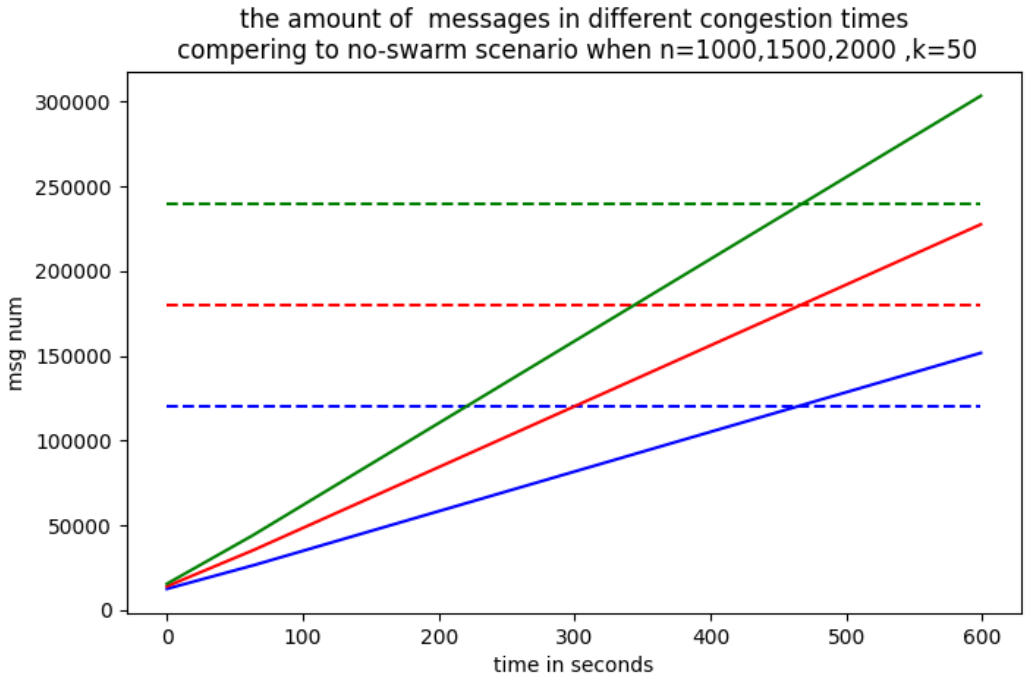

Figure 13: Graph of messages number in different congestion times compered between $M(t)$ and $M_{no\ swarms}$ when n = 1000,1500,2000 and k = 50.

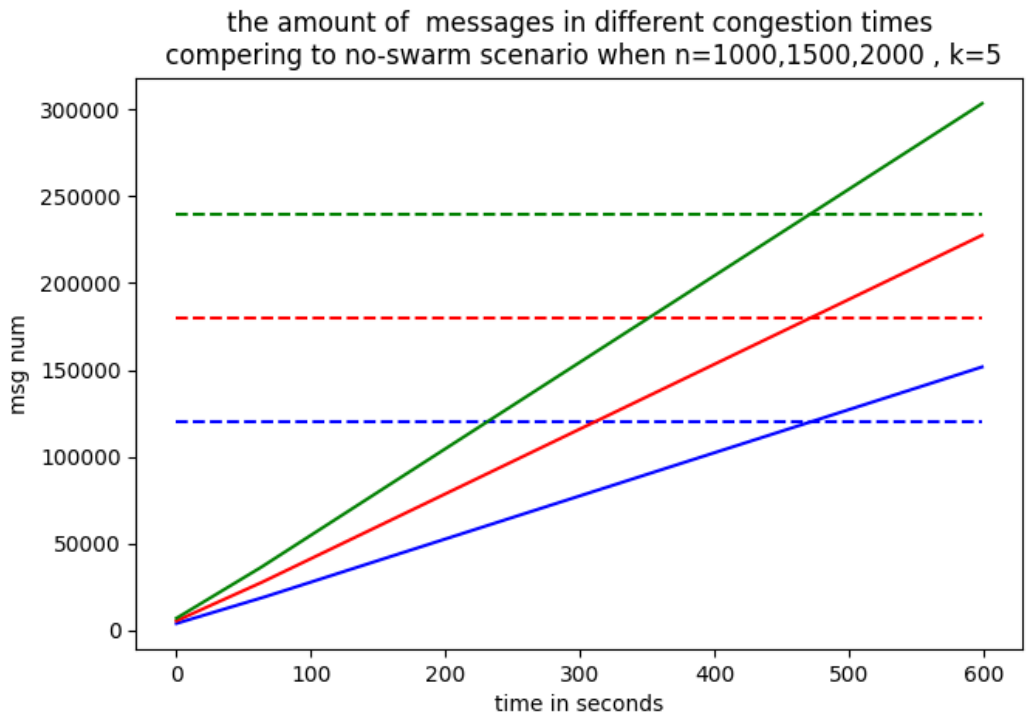

Figure 14: Graph of messages number in different congestion times compered between $M(t)$ and $M_{no\ swarms}$ when n = 1000,1500,2000 and k = 5.

In both scenarios, where the number of vehicles and the number of swarms are independently changed, an intriguing consistency emerges as they converge to the same meeting point at 7.83 minutes. This convergence point holds significant implications for the algorithm implemented in the project. When the algorithm's congestion time is less than 7.83 minutes, it demonstrates greater efficiency compared to the original state without swarm division. However, when the algorithm's congestion time exceeds 7.83 minutes, it not only becomes less efficient than the original configuration but exhibits a substantially deteriorated performance.

To gain further insights into the algorithm's efficiency, a comparison was conducted with a test involving 100 vehicles divided into 5 swarms, as opposed to a scenario with the same number of vehicles but no swarm division. The results are as follows:

For a remarkably optimistic congestion time of 0.75 minutes (45 seconds), the presented algorithm proved to be 81.4% more efficient than the original state in terms of reduced message transmission during the simulation time.

With an optimistic congestion time of 2 minutes, the presented algorithm displayed 67% higher efficiency compared to the original state.

As the congestion time increased to 4 minutes, the presented algorithm still exhibited 43.6% higher efficiency than the original state.

Even with a congestion time of 6 minutes, the presented algorithm retained its advantage, being 20.2% more efficient than the original state in terms of message reduction.

These findings underscore the significance of the 7.83-minute meeting point and emphasize the algorithm's performance dependency on its congestion time. The provided data showcases how the algorithm's efficiency can outperform the original state under certain conditions but can suffer a substantial decline when congestion times exceed the critical threshold.

6. Challenges

6.1. Workspace

One of the main challenges was finding the right workspace. The initial assumption was that the algorithm can be simulated by SUMO alone, which could provide the road topology, the routing, the vehicles, and the random speed and location.

Unfortunately, SUMO couldn't provide the V2V of any other communication needed to simulate the algorithm, and the need for a communication simulator that can be combined with SUMO emerged. OMNeT++ was an obvious simulator due to its versatility and variety of factuality such as wireless communication, ability to construct message type, and implementation of communication protocols.

The choice of OMNeT++ led to another challenge, how to make SUMO and OMNeT++ work together? After research, and with the help of open-source code, the tools Veins and TraCI became the most accurate solution for our algorithm's simulations.

6.2. Implementation within the OMNeT++ framework

OMNeT++ is a complex program with many conditions, files, and a unique coding language, which raises programming challenge. While OMNeT++ is a common simulation tool, there are little information and sources available to learn from, especially while using another simulator (SUMO), and a specific combination of different programing tools such as Veins and TraCI.

Finding open-source code guides, using the OMNeT++ manual, and using the information in the official website of SUMO [13] and Veins [10] were the only way to deal with the complexity of OMNeT++ and SUMO combination.

6.3. Assembling an Equation

The equation was not intuitive at first, the challenge was detecting the patterns, stages in the algorithm, and the relation needed to construct the equation.

The assembling of the equation started from the variable's definition, and their numeric value. Observing the different stages – the messages before congestion into swarms of vehicles, the messages needed for the vehicles to preform 3-ways handshake, and the messages post congestion – was the key to develop mathematical rules, that developed the equations for each stage, and led to the assembling of the main equation (Eq.2 and Eq.3 which is the numeric form of Eq.2.).

7. Conclusions

7.1. Work Summary

Working on this project was a challenging yet rewarding experience. We entered a journey to demonstrate the feasibility of our algorithm within our specific environment. The solution we proposed required careful consideration of numerous parameters before conducting the simulation. The algorithm creation was not trivial, as it demanded thorough analysis and thoughtful decision-making to ensure we created the right work environment for our model to thrive.

Ultimately, we succeeded in showcasing the viability of our model within our chosen environment. The sense of satisfaction and fulfillment derived from seeing our hard work bear fruit was immeasurable. This project not only tested our abilities but also taught us valuable lessons about resilience, adaptability, and the importance of meticulous planning.

7.2. Follow-up Project Suggestions

The completion of the project marks a significant milestone, although it's important to note that fully autonomous vehicles are not yet a reality. Our algorithm, built upon fundamental ground rules, showcases promising results. However, there remains much room for expansion to enhance its accuracy further. The complexity of the real world introduces many challenges, prompting us to raise essential questions not only about autonomous vehicles as a whole but also about the specific nuances of our project. As a result, we propose these ideas as a foundation for follow-up projects that can go deeper into addressing these complexities and advancing the development of autonomous vehicles.

7.2.1. Follow-up Project – Improve and Develop the Algorithm

A compelling follow-up project could involve examining the algorithm's performance in diverse and challenging edge cases. These scenarios could encompass various complex conditions, such as intersections with traffic lights, intricate road networks, and more. Additionally, exploring extreme cases like communication failures or road accidents could provide valuable insights into the algorithm's resilience and adaptability. Investigating how the algorithm responds to unexpected situations, such as a car deciding to change its destination during the trip or handling emergency stops for obstacles like a child running onto the road, would offer crucial knowledge for enhancing the algorithm's real-world applicability and safety measures.

7.2.2. Follow-up Project - Enhancing Autonomous Vehicle Interactions

A promising follow-up project could involve developing algorithms that govern the interactions between autonomous vehicles and emergency vehicles, traffic lights, and other external elements that establish communication with the autonomous car. This project would explore the dynamics of communication between the autonomous vehicle and traffic lights or any other entity that creates a network of interactions. The project may encompass various aspects of the Internet of Things (IoT) and simulate the behavior of the autonomous vehicle, similar to that of a smart home, where human intervention is unnecessary, and it possesses capabilities beyond those of a human driver. For instance, the autonomous vehicle could request the traffic light to switch to green or receive real-time updates from an ambulance, allowing it to make informed decisions on how to give way and facilitate its passage safely. This endeavor holds the potential to advance the seamless integration of autonomous vehicles into the broader transportation ecosystem, making them more efficient and responsive to external stimuli.

7.2.3. Follow-up Project - V2V-Driven Mobility

The follow-up project aims to implement communication-independent movement for autonomous vehicles. The initial phase involves the existence of a remote server, which, after a specified time or possibly after the formation of vehicle swarms, will become non-operational. Subsequently, all communication will be established solely between the vehicles. The main objective is to initiate the journey with communication solely through V2V (Vehicle-to-Vehicle) communication, eliminating the need for any connection to a remote server. The project seeks to examine whether the absence of remote server utilization leads to increased efficiency or inefficiency in the system's operation. By exploring this communication-independent approach, the project aims to enhance the self-reliance and autonomy of the autonomous vehicles, opening up new possibilities for their real-world deployment.

7.2.4. Follow-up Project - Multi-Level Swarms

The goal of this project is to create a hierarchical structure of destinations and vehicle swarms. After the initial division of vehicles into swarms, a further subdivision will occur - creating sub-swarms within existing swarms. For instance, if a swarm is heading towards a specific city, within that swarm, sub-swarms will form for vehicles traveling to specific neighborhoods, and within those sub-swarms, additional sub-swarms will be created for vehicles heading to specific streets, and so on. This approach allows for the inclusion of subgroups for vehicles going to private residences or, conversely, to events or sports games where a large number of vehicles are heading to the same specific destination. The subdivision into sub-swarms can occur before or after the original swarm splits, providing flexibility in the hierarchical organization. This project aims to explore and optimize the efficiency and coordination of vehicle movement through such multi-level swarm structures, offering new insights into swarm-based transportation systems.

7.3. Advantages and Disadvantages of the Solution

7.3.1. Advantages

The project demonstrates several significant advantages. First and foremost, it proves the feasibility of the proposed model within the specific environment. Despite the complexity of the algorithm, it successfully addresses the fundamental requirements for its functionality, showcasing its potential for real-world application. Moreover, the presentation of an evaluative equation allows for a thorough analysis of the algorithm's properties, enhancing transparency and understanding of its behavior. The elegant equations used in the algorithm design further contribute to its appeal and readability. Additionally, the project's simulation, although challenging, played a crucial role in validating the algorithm's efficacy, providing valuable insights into its performance under various scenarios.

7.3.2. Disadvantages

We showed that our model is feasibility, in our environment. The solution we proposed is not a trivial one, b While the project boasts numerous advantages, there are also some limitations to consider. One notable drawback is the complexity of the algorithm, which might pose challenges in its implementation and maintenance. Furthermore, the observed significant improvement in a specific time range raises uncertainty about its practicality in the real world, where various dynamic factors can influence vehicle movements differently. It is essential to carefully assess the algorithm's applicability beyond this specific range. Additionally, the complexity of the simulation might limit its scalability and efficiency in handling larger-scale scenarios. Nonetheless, despite these limitations, the project marks a significant step towards the development of effective swarm-based transportation systems, demonstrating great promise for future improvements and optimizations.

7.4. Project Goals Achievement

The project has achieved its primary goals by demonstrating the feasibility of the proposed algorithm. Despite the complexity of the algorithm, it successfully covers the essential requirements for its functionality, providing a basis for the envisioned autonomous vehicle system. The presentation of an evaluative equation allows for a comprehensive assessment of the algorithm's properties, enhancing transparency and facilitating further analysis. Although the simulation was challenging, it served as a proof-of-concept (POC) for the algorithm, validating its performance under various conditions. Moreover, the elegant equations used in the algorithm design reflect the efforts in creating an efficient and effective solution that meets specific requirements. While the algorithm displayed significant improvement in a specific time range, there is a need for further exploration to determine its feasibility in the real world, where dynamic factors may influence its performance differently. Overall, the project's achievements lay a solid foundation for the development of an algorithm that meets the desired objectives and leads the way for potential advancements in autonomous vehicle systems.

8. Bibliography

[1] Yair Allouche and Michael Segal. "**A cluster-based beaconing approach in vanets: Near optimal topology via proximity information**". English. In: Mobile Networks and Applications 18.6 (Dec. 2013)

[2] https://carbiketech.com/swarm-intelligence-automated-driving/

[3] Self-driving Car Wikipidia: https://en.wikipedia.org/wiki/Self-driving_car

[4] SAE Official Website: https://www.sae.org/

[5] https://www.synopsys.com/automotive/what-is-autonomous-car.html

[6] https://accoladetechnology.com/5-levels-of-autonomy-l1-and-l2/

[7] SUMO Wikipidia: https://en.wikipedia.org/wiki/Simulation_of_Urban_MObility

[8] SUMO Overview: https://www.eclipse.org/sumo/

[9] OMNeT++ official website: https://omnetpp.org/

[10] Veins official website: https://veins.car2x.org/

[11] TraCI API - SUMO Documentation: https://sumo.dlr.de/docs/TraCI.html

[12] DSRC Wikipedia: https://en.wikipedia.org/wiki/Dedicated_short-range_communications

[13] SUMO Official Website: https://eclipse.dev/sumo/

[14] Our Git Project: https://github.com/maayann102/project-2023-003

9. Final Report Evaluation

המלצת ציון (ע"י מנחה אקדמי) לדו"ח מסכם

אם יש צורך, לכל סטודנט/ית בנפרד

מספר הפרויקט: p-2023-003

שם הפרויקט: נהיגה אוטונומית בעזרת להקות.

שם המנחים מהמחלקה: פרופסור מיכאל סגל.

שם הסטודנטית: נוי אלה ת.ז.: 312252257

שם הסטודנטית: מעיין בן-גל ת.ז.: 311438733

%		חלש 55-64	בינוני 65-74	טוב 75-84	ט"מ 85-94	מצוין 95-100	
20	הצגת הגישה והתכנון ההנדסי						
20	הצגת התוצאות וניתוח השגיאות						
20	הסקת מסקנות						
10	גילוי יוזמה וחריצות						
20	פתרון בעיות, מקוריות ותרומה אישית (מעבר למילוי ההנחיות)						
10	עמידה בלו"ז ורמת הביצוע המעשי						

אם יש כוונה לפרסם/ יפורסם מאמר, שם כתב העת ומועד משוער להגשה:

ציין אם יש כוונה לשקול המלצה כפרויקט מצטיין:

הערות נוספות: