SN Computer Science (2024) 5:80
https://doi.org/10.1007/s42979-023-02391-y

ORIGINAL RESEARCH

Internet of Vehicles-Based Autonomous Vehicle Platooning

Omkar Dokur[1] · Srinivas Katkoori[1]

Received: 9 January 2023 / Accepted: 5 October 2023

Abstract
The Internet of Things (IoT) facilitates vehicle communication using wireless networks to improve safety, mobility, and efficiency in transportation. Autonomous vehicles (AVs) can use IoT to form a platoon and travel cooperatively to a common destination as connected autonomous vehicles (CAVs). In our previous work, we demonstrated platoon negotiation and formation between two vehicles or IoT nodes using Dedicated Short Range Communication (DSRC)-based messages only. This paper extends these algorithms to support multi-vehicle platoon negotiation and formation using DSRC messages for AVs. To achieve this, once two vehicles negotiate and form a platoon, the platoon member (PM) sends a platoon-complete negotiation to the platoon leader (PL) after the string stability is achieved. Once PL receives this message, it makes itself available to receive negotiations from nearby vehicles who are willing to join its platoon. We modified our platoon-ready, pre-negotiation, negotiation resolver, and platoon joiner algorithms from our prior work. Also, PL maintains the PM vehicle IDs and their position so that it can assign a local leader to the future vehicles joining the platoon. Now, the vehicle willing to platoon negotiates with PL to check if their destinations match. If a common destination is found, the new vehicle further negotiates with PL in a series of transactions to join the existing platoon. During these negotiations, PL assigns the last joined PM as a local leader to this newly joined vehicle to follow. Then, PL adds the new PM vehicle ID and its position to the list. Assigning a local leader not only increases the range of the platoon but also decreases the delay in the message exchange. We demonstrated the above algorithms in the CARLA simulator by extending them to support IoT connectivity and platooning. We validated the algorithms by conducting experiments with three-vehicle platooning scenarios.

Keywords Internet of Things (IoT) · Connected vehicles (CVs) · Platoon negotiation · Platoon formation · CARLA

Introduction and Motivation

The term "Internet of Things" (IoT) refers to the data sharing between physical things connected via the internet. The number of physical things getting connected in the IoT field is rising quickly [1]. There have been numerous IoT architectural proposals thus far. In their book, Hanes et al. [2] introduced a simplified IoT architecture composed of edge, fog, and cloud layers, which is briefly covered here below:

1. Edge Layer: Contains edge nodes, which are smart objects, sensors, and actuators. IoT devices here, also known as edge computing, carry out the computation.
2. Fog Layer: Contains network-connected communication and processing units also known as fog nodes. Using fog nodes, data from edge nodes are uploaded to the cloud which is also known as fog computing.
3. Cloud Layer: In this layer, applications and analytics are built using data from fog nodes.

As shown in Fig. 1, the IoT concept mentioned above can be applied to autonomous vehicles (AVs) in the transportation sector where each AV can function as an edge node to address a number of issues [3]. Notably, the American Transportation Research Institute (ATRI) examined the operational costs of trucking based on data from 2021

This article is part of the topical collection "Multidisciplinary Research Perspectives for IoT Systems" guest edited by Luis Camarinha-Matos, Luis Ribeiro, Paul Havinga and Srinivas Katkoori.

Omkar Dokur
omkardokur@usf.edu

Srinivas Katkoori
katkoori@usf.edu

[1] Department of Computer Science and Engineering, University of South Florida, 4202 East Fowler Avenue, Tampa 33620, FL, USA

Published online: 09 December 2023

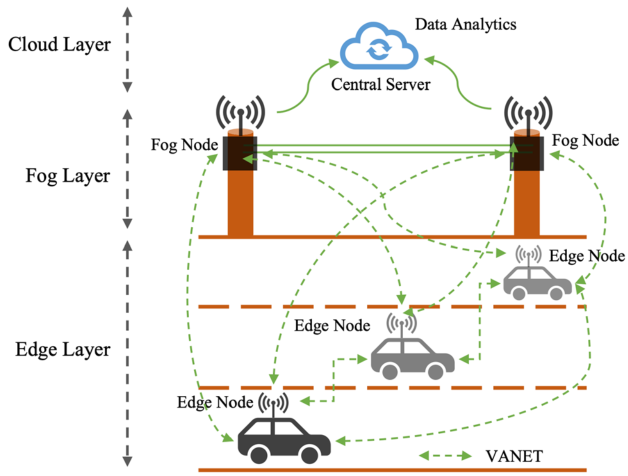


Fig. 1 Simplified IoT architecture vehicles connected through VANET

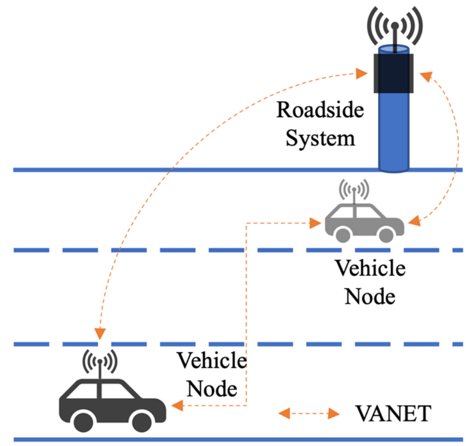


Fig. 2 Vehicles and roadside system connected through VANET

in its 2022 edition. This report showed that the fuel efficiency in terms of miles per gallon (MPG) increased very slightly from 6.535 MPG in 2020 to 6.652 MPG in 2021 due to the use of speed governors [4]. Additionally, the American Trucks Association (ATA) reports that in 2020, trucks used 9.0 billion gallons of gasoline and 35.8 billion gallons of diesel fuel leading to large amounts of greenhouse gases [5]. According to the US Environmental Protection Agency (EPA), transportation is the major source of greenhouse gas emissions, accounting for 27% of all emissions in 2020 [6]. As a result, there are significant fuel costs and greenhouse gas emissions, both of which can be reduced if fuel economy is further enhanced. This can be achieved through vehicle platooning, where a group of vehicles drive closely together with smaller gaps between them to achieve lower aerodynamic resistance, can boost fuel economy [7].

Many companies are striving to make autonomous driving technology accessible to the general public, including Tesla, Waymo, Uber, and others. The connected vehicle (CV) technology is being tested by government agencies through a number of experimental deployment programs [8]. To increase transportation efficiency and safety, these two technologies are combined. There are many vehicle applications being developed, and platooning is undoubtedly one of them [9]. Platooning is achieved through the Internet of Vehicles (IoV) which is a subset of the IoT where vehicles and roadside systems are connected through vehicular ad hoc networks (VANETs) as shown in Fig. 2 [10]. Using vehicle-to-everything (V2X) connections, each vehicle in the IoV is considered a smart object that is installed with sensor platforms, processing facilities, control units, and storage. A wide range of industry sectors, including transportation, automobile manufacturing, energy, automation, software, and information and communication technology, are all significantly impacted by this technology [11].

The implementation of platooning technology remains limited in practice, prompting collaborative efforts between government entities and manufacturers to explore its potential. Notably, the Florida Department of Transportation (FDOT) led the Driver Assistive Truck Platooning (DATP) Pilot project, focusing on specific weather conditions to showcase platoon operations [12, 13]. While initiatives like Konvoi and SARTRE did not comprehensively address diverse weather scenarios [14], the multifaceted weather conditions present in the United States, including snow, rain, and fog, pose challenges for platooning. In response, our approach introduces a relative position methodology from our prior work [15] to enhance existing platoon design. This approach stands apart from conventional methods that rely on cameras or LIDAR, susceptible to adverse weather conditions, as we utilize Dedicated Short Range Communication (DSRC), resilient to weather.

Furthermore, existing algorithms presuppose active platooning systems with a platoon leader (PL) and one platoon member (PM), rendering them incapable of supporting the formation of new platoons. To address this, we propose IoV-based platoon algorithms that enable two vehicles to negotiate and initiate a non-existing platoon through Basic Safety Messages (BSMs). After successful negotiations, the PL and PM are designated. Subsequently, the PM uses the platoon formation algorithm to compute the relative position of the PL to generate target velocity and destination. These are used as inputs to compute the vehicle commands for PMs to form a new platoon. Once a platoon is formed, new vehicles can join the platoon using the same proposed algorithms.

The paper's focus is on enhancing platooning technology for vehicles using the IoV approach. The proposed IoV-based platoon algorithms address the initiation of non-existing platoons, allowing vehicles to negotiate and form platoons using Basic Safety Messages (BSM) over Dedicated Short Range Communication (DSRC). The significant contributions of our work encompass:

1. Initiating platoons via BSM-based communication over DSRC with nearby vehicles;
2. Introducing an IoV-based multi-vehicle non-existing platoon negotiation algorithm capable of handling diverse messages;
3. Formulating an IoV-based multi-vehicle platoon formation algorithm;
4. Extending the CARLA simulator to incorporate IoT connectivity; and
5. Validating the aforementioned platooning algorithms through CARLA simulations.

Initially, the efficacy of the proposed platoon negotiation and formation algorithms is demonstrated via a two-vehicle scenario, followed by a three-vehicle scenario. The remaining sections of the paper are organized as follows. "Overview" provides an overview of AVs, connected autonomous vehicles (CAVs), an IoT vision for platooning, and simulation tools. "Related work" summarizes the related work on IoT-based vehicle platooning systems and their challenges. "Edge node platooning algorithms" explores two-vehicle negotiation algorithms as well as the modified platoon negotiation and formation algorithms to support multi-vehicle platooning. "Experimental results" presents the experimental findings of the proposed platooning algorithms from the CARLA Connect simulator. Finally, "Conclusion" summarizes our experimental results and presents future work.

Overview

This section provides an overview of CAVs, platooning, and VANET simulation tools.

Automated Driving System

According to Rajasekhar et al. [16], an AV is a smart vehicle that can sense its environment, choose the best route to its destination, and then drive by itself using Automated Driving System (ADS). The Society of Automotive Engineers (SAE) defines six levels of autonomy, with level zero meaning no autonomy and level five meaning complete autonomy [17]. An ADS can operate in every driving scenario without assistance from the driver at level five. Despite the fact that ADS ensures a safe, efficient, and comfortable driving experience, the number of casualties is growing. Although AV technology has advanced significantly to this point, several issues still exist [18]. Autonomy at level four and up still remains an open and challenging problem. Additionally, most current ADS solutions are ego-only approaches. In ego-only approach, vehicles operate independently, relying solely on their sensor and internal systems to navigate and make decisions, without active communication or coordination with other vehicles [19]. The technical, ethical, and legal challenges currently faced by ADS are thoroughly discussed in [20].

Connected Autonomous Vehicles (CAVs)

As discussed in "Automated driving system", AVs currently use an ego-only technique in which the vehicle alone handles all aspects of driving. But, an AV can make use of the CV technology to rely on the road infrastructure and other nearby vehicles to accomplish driving responsibilities and perform the functionalities of an ADS. These vehicles will have access to a vast amount of data through vehicle-to-vehicle (V2V), vehicle-to-infrastructure (V2I), and vehicle-to-everything (V2X) communications. This could be used to fix the problems with the existing ego-only approach and is believed to be autonomous driving's future [19, 21]. The phrase "Connected Vehicles" refers to smart transportation, which involves wireless communication between vehicles and their surroundings. These include V2V, V2I, and V2X communications [22]. The CV network as a whole falls into the category of ad hoc networks and is known as VANET [23]. DSRC and Long-Term Evolution (LTE) are the most commonly identified candidate technologies with a VANET's communication layer [24].

Road-Side Units (RSUs) and On-Board Units (OBUs) are the two central units of CV technology where OBUs are deployed within the vehicles while RSUs are installed on the road infrastructure. In the IoT architecture, an OBU represents an edge while the RSU represents a fog node as shown in Fig. 1. The OBU gathers vehicle information like speed, location, heading, etc., and sends it numerous times per second through a wireless network in the form of a BSM to surrounding OBUs or RSUs in the area. OBUs that receive BSMs use this information to keep drivers safe and alert them if there is a chance of an accident [25].

When CV technology is incorporated into an AV, it becomes a CAV and increases safety, effectiveness, and mobility by gaining access to a vast amount of nearby traffic data [21]. This way CAVs will be able to access a significant amount of nearby vehicle data to address the current issues with the ego-only design. Currently, there is not a single working CAV on the road and the future of ADS is believed to be this design [19].

An IoT Vision for Platooning

According to Shladover et al. [26], a platoon is defined as the "spontaneous and dynamic creation of convoys of vehicles".

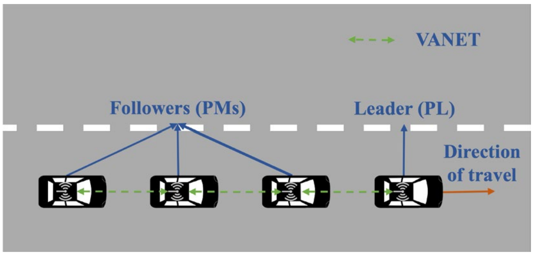


Fig. 3 Platoon illustration through VANET

A platoon at a minimum consists of two vehicles where one vehicle closely follows the one in front of it. Here, each vehicle can be driven autonomously or drivers are assisted to keep a safe distance and stay inside the lane limits, including autonomous braking [27]. As shown in Fig. 3, a standard platoon is made up of a PL and a PM, with the PL leading and the PM following using VANET. PL is responsible for regulating and tracking the platoon's speed, member count, and entry and departure permits. PM follows the directions from PL and then reports to PL on its progress. While PL and PM communication is accomplished via V2V, the inter-vehicle gap is calculated and maintained by Adaptive Cruise Control (ACC) employing LIDAR, radar, camera sensors, etc. A Cooperative ACC (CACC), which combines V2V communication in CV with ACC in ADS, enables platooning [28]. Also from Lesch et al. [29], platooning requires the use of control and communication technologies. Sturm et al. [30] presented various stages involved in a platooning process which are:

1. Finding a Platoon: V2V or V2I communication is used by vehicles to identify available platoons and possible platooning participants.
2. Joining a Platoon: After finding a platoon, the vehicle interacts (negotiates) with the platoon (usually the platoon leader) on how to join it.
3. Maintaining a Platoon: After the vehicle joins a platoon, this takes care of maintaining a safe distance between PMs, managing overtaking procedures, and combining and dividing platoons.
4. Leaving and Dissolving a Platoon: A vehicle may leave the platoon to discontinue platooning or to join another platoon.

A platoon is affected by a variety of vehicle-dependent variables, such as the inter-vehicle gap, platoon-dependent attributes, such as platoon size, and external factors, such as the weather. Sturm et al. [30] covered each of these elements in great detail. This work focuses on the first three stages which are finding a vehicle that is willing to platoon, joining the found vehicle by negotiating with it and maintaining it. To achieve this, we apply the relative position technique for CVs that was proposed in our earlier work [15]. By computing the angle between the host vehicle (HV) heading and a new vector drawn between the HV and remote vehicle (RV) position, the relative angle '*theta*' between HV and RV can be determined. Similarly, the angle between HV and the new vector between HV and PL as illustrated in Fig. 4 can be used to determine the relative angle "*theta*" between HV and PL (an RV here).

Simulation Tools

Many VANET simulation tools are available. Weber et al. [31] provided an updated review on VANET simulators. According to them, mobility and network simulators are the main building block components of VANET simulators. Examples of mobility simulators include SUMO, VISSIM, SimMobility, PARAMICS, and CORSIM and examples of network simulators include OMNeT++, OPNET, JiST/SWANS, NS3, and NS2. NetSim, Veins, Eclipse MOSAIC, EstiNet, ezCar2X, etc., are some of the VANET simulators available. But, all of these simulators are 2D that do not provide proper visualization to validate the algorithms. Hence, we chose CARLA, a 3D simulator that supports autonomous driving to validate the proposed algorithms.

CARLA (Car Learning to Act) is an open-source simulator platform for autonomous driving based on Unreal Engine 4 (UE4) [32]. It supports basic NPC logic, real-time physics, interoperability, and high-quality graphics. A proportional–integral–derivative (PID) controller is utilized here to deliver steering, throttle, and braking signals to drive the simulated automobiles autonomously. The PID controller uses the current vehicle position, speed, and waypoints list to create the vehicular commands. Throughout the simulation, we employ the aforementioned PID controller to create vehicular orders for the simulated automobiles. CARLA is appropriate for platooning testing because of its comprehensive sensor suite and autonomous driving capabilities. CARLA does not support vehicle connectivity and platooning and we extended it to CARLA Connect to support these features [33].

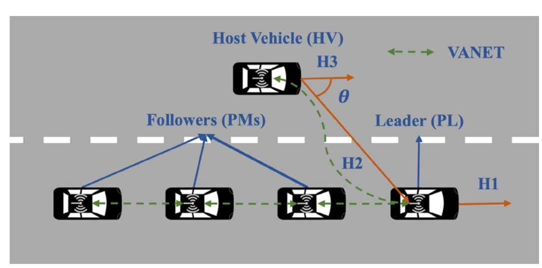


Fig. 4 Relative position approach for platooning

Related Work

There have been several platoon research projects, some of which are still continuing. They all have different goals, such as truck platooning, mixed vehicle platooning, and so on. They either presume that vehicles are entirely autonomous or that PL is driving manually [34]. CARMA is one such initiative (Cooperative Automation Research Mobility Applications). CARMA is a Federal Highway Administration (FHWA) program of the United States Department of Transportation (USDOT) [35]. CARMA3 is the most recent version of CARMA, and it includes ADS and V2X capabilities to accomplish Cooperative Driving Automation (CDA). CARMA's CDA features include cruising, yielding, lane changing and merging, speed harmonization, and platooning [9]. CARMA enables the development and testing of CDA features using an open-source methodology in collaboration with academic institutions. While CARMA is opening the way for more CDA research and development (R &D), several issues remain unresolved.

There is a lot of research being conducted in the IoT domain to enable vehicle platooning. Chakraborty et al. [36] offered a unique Adaptive Cruise Control (ACC) system for self-driving platoons based on Cloud, Cloudlet, and Sensor Fusion via Adaptive Kalman Filter. Here, the authors proposed a novel IoT architecture where Sensor Fusion and ACC algorithms are stored in a Cloudlet. Information such as road, weather, etc., and vehicle IoT services are provided by the Cloud system. To reduce the overhead of data processing at edge nodes, Petrov et al. [37] has created a framework for message exchange and relay-based data processing. Here, the authors make use of crowdsensing, where several devices (vehicles) work together to process required data, using Narrowband IoT Technology (NB-IoT). Sodhro et al. [38] study intends to address the decline in Quality of Service (QoS) as a result of the high mobility of vehicles as nodes in an IoT network. The authors created a QoS-aware, green, sustainable, reliable, and available (QGSRA) algorithm to decrease message exchange latency, consume less energy, and enhance service reliability.

Niu et al. [39] proposed a Space-Air-Ground integrated vehicular network (SAGiven) architecture to support connected, automated, and intelligent transportation systems in the future. Wang et al. [40] from Ibaraki University in Japan discussed the effect of message exchange latency on platooning safety. The authors found that using a directional antenna reduced message delay in V2V communication. Directional antenna utilization has been demonstrated to be helpful for vehicles or beacons that are close to one another, despite the fact that it is still effective for high traffic. It is also crucial to draw attention to the cutting-edge technologies that might significantly enhance the overall quality of not only the proposed V2V application but also of a wide range of other IoT-related technologies. Blockchain technology and Artificial Intelligence (AI) application may also enhance vehicular IoT applications, enabling decentralized network management, node-to-node interoperability, and traceability and dependability of the data being exchanged [41]. IoT can benefit from Artificial Intelligence of Things (AIoT) in terms of data processing and analytics, which will improve the reliability and efficiency of platoon formation, negotiation, and maintenance [42].

So far, extensive research has been conducted in platooning that depends on ADS onboard sensors like radar, LIDAR, cameras, etc. However, these sensors may fail during inclement weather [43]. Adverse weather conditions such as rain, snow, or fog may cause limited visibility for platooning [34]. Table 1 compares exteroceptive sensors used in ADS to better understand how they work in low-light and adverse weather conditions [19]. Except for radar and ultrasonic, all sensors are impacted by either light or weather. And one cannot rely on these two sensors since they only serve to determine distances between objects ahead but do not offer lane information or the platoon's relative position. According to [44], the change in air density that occurs during foggy weather has no impact on DSRC transmission. The Safety Pilot Model Deployment (SPMD) project directed by the University of Michigan Transportation Research Institute (UMTRI) consisted of around 2800 cars outfitted with OBU devices and 25 RSU devices [45]. This study's data are utilized to evaluate the influence of weather conditions on DSRC transmission. A study of 2,581 clear weather days, 114 rainy days, and 227 snowy days revealed that severe weather conditions have no effect on DSRC performance. Based on this study, DSRC is a promising and dependable technology in every weather condition.

There have been several platooning algorithms proposed so far. Rajamani et al. [46] recorded the lateral and longitudinal control systems used in the National Automated

Table 1 Comparison of ADS sensors

Sensor	Influenced by light	Influenced by weather
LIDAR	No	Yes
Radar	No	No
Ultrasonic	No	No
Camera	Yes	Yes
Stereo Camera	Yes	Yes
Flash Camera	Yes	Yes
Event Camera	Limited	Yes
Thermal Camera	No	Yes

Highway System Consortium (NAHSC) demonstration to display an eight-car platoon. Here, the vehicles were kept within the lane using magnets placed in the middle of the lane and magnetometers mounted on both the front and rear of the vehicles. Their longitudinal control system is based on a radar sensor that achieves string stability and performs platoon merge/split operations using radio transmission. Although the demonstration showed that vehicle platooning is feasible, installing magnets in lanes is quite expensive. Amoozadeh et al. [47] created a platoon management protocol for CACC vehicles using VANET that includes a merge, split, and lane-change actions. Saeednia et al. [48] presented a hybrid approach that combines catch-up and slow-down strategies during the formation phase of platooning, assuming the connectivity between vehicles. Here, the authors discussed only longitudinal movement, while there is no mention of the lateral movement of the vehicles during platoon formation. Back et al. [49] provided a technique to re-elect the platoon leader vehicle automatically based on the scenario to change a fixed leader into a flexible leader vehicle. They utilized the Raft algorithm to select a suitable leader vehicle based on numerical values of vehicle performance by monitoring the leader vehicle's condition in real time and responding to diverse scenarios.

Choi et al. [50] addressed the issue of traffic overload and time delay caused by the existing cluster-run approach by introducing the priority of cluster candidate leader vehicle. Here, if the leader needs to be changed due to traffic circumstances or vehicle characteristics, the leader may be changed rapidly without splitting the cluster thereby allowing for flexible platooning. But, both of the above works assume an active existing platoon that is not practical. Burov et al. [51] suggested a platoon formation method to save travel time, but it only functions for active or established platoons. A technique was presented by Ganaoui-Mourlan et al. [52] that enables a vehicle to join a platoon autonomously and with the best route possible that makes the optimal use of the road space. In their work, PL determines the path that PM will take to get to the platoon position. This could be computationally expensive for larger platoons and may cause a delay in platoon formation times.

Table 2 offers a comprehensive qualitative comparison between existing platooning algorithms and our proposed work. This comparison underscores the unique contributions and limitations of each approach, highlighting the distinct features that set our proposed work apart. Our motivation to develop non-existing multi-vehicle platoon negotiation and formation algorithms was fueled by the insights gained from this comparative analysis. Recognizing the limitations of existing approaches, we embarked on a path to address critical gaps in platooning research. By actively incorporating negotiation, platoon formation, and the initiation of new platoons, our approach aims to provide a holistic solution that goes beyond the scope of previous works. Through this research, we strive to push the boundaries of platoon algorithm research and contribute to the safer and more efficient transportation system.

Table 2 Qualitative comparison of existing work with proposed work

Work	Negotiation	Formation	New platoon
Rajamani et al. [46]	Yes	Not supported	Out of scope
Amoozadeh et al. [47]	Yes	Not supported	Out of scope
Saeednia et al. [48]	Not supported	Yes	Out of scope
Back et al. [49]	Not supported	Not supported	Out of scope
Choi et al. [50]	Not supported	Not supported	Out of scope
Burov et al. [51]	Yes	Yes	Out of scope
Ganaoui et al. [52]	Not supported	Yes	Out of scope
Proposed Work	Yes	Yes	Yes

Edge Node Platooning Algorithms

In this section, we extend the platoon negotiation and formation algorithms proposed in our prior work [53, 54] to support multi-vehicle platooning, where the term "multi-vehicle" refers to a platoon consisting of more than two vehicles. The algorithms are designed to function effectively on roads which has more than 2 lanes or multiple-lane roads that can accommodate both left-side and right-side driving scenarios, even considering instances where heavy goods vehicles (HGVs) might not be allowed in the third lane. Here, the negotiation algorithms help vehicles to find and negotiate while the formation algorithms provide inputs required to generate the vehicle commands to join and maintain the platoon. These algorithms execute independent on each vehicle and process data from the BSM messages. We assume that each vehicle transmits multiple BSMs per second that contains vehicle information such as speed, location, acceleration, heading, etc., as well as platoon negotiation consisting of negotiation, sender id, and receiver id. When another vehicle receives a BSM containing a negotiation, the negotiation algorithms are developed in such a way that a negotiation is processed only if the receiver id matches the current vehicle id.

Further, the negotiation process in our approach involves a combination of coordination and decision-making among the platoon agents. While the term "negotiation" is used, it is more accurately described as a coordination mechanism to determine the roles and positions within the platoon. When a vehicle seeks to join an existing platoon or form a new one, it broadcasts its intent and destination to nearby vehicles using BSMs. Vehicles which are nearby receive these messages and evaluate the negotiation requests based on predefined criteria such as vehicle speed, proximity, and destination. The negotiation resolver plays a central role in analyzing these requests and making decisions. In this work, we use five different types of negotiations out of which four were introduced in our earlier work and added Platoon Complete (PC) as a fifth negotiation. We have added this extra negotiation to acknowledge the leader that the current joining member has completed the platoon. A brief description of each of them is provided below:

1. Platoon Join Request (PJRQ): Once a vehicle is ready to platoon, this negotiation is used to request another vehicle to form a platoon.
2. Platoon Accepted (PA): Upon receiving a PJRQ negotiation, a PA negotiation is sent to the requesting vehicle if the current vehicle accepts to form a platoon.
3. Platoon Join Ready (PJRY): Upon receiving a PA negotiation, a PJRY negotiation is sent back to the platoon accepted vehicle to acknowledge that the current vehicle is ready to join a platoon.
4. Platoon Leader Request (PLR): Upon receiving a PJRY negotiation, a PLR negotiation is sent back to the acknowledged vehicle by the current vehicle requesting to be a platoon leader.
5. Platoon Complete: After negotiating and forming a platoon, PM sends a PC negotiation to PL that the platoon formation is complete.

Using these negotiations and the negotiation algorithms, two vehicles negotiate with each other to form a platoon initially. Once the platoon is formed, the PM will send a PC negotiation to the PL. Upon receiving this, PL makes itself available to accept future platoon requests. In the next section, we provide a brief description of each of the above works.

Two-Vehicle Platoon

Platoon-ready, pre-negotiation, negotiation resolver, and PM algorithms were developed in our earlier work [53] to initiate and start a platoon based on BSMs only. Figure 5 illustrates the relationship between these algorithms and the order in which they are executed.

Platoon-Ready Algorithm

The platoon-ready algorithm determines if the individual vehicle is ready to platoon after certain requirements are met. Several criteria can be established to do this. The requirement we put here is that the platoon be switched on and the vehicle is moving on a road at a specified speed, for example, more than 40 km/h. If these requirements are met, the vehicle is suitable for platooning. The flowchart depicting the platoon-ready state is shown in Fig. 6. Once the vehicle is platoon-ready, pre-negotiation transactions occur before the actual negotiations.

Pre-negotiation Transactions

Pre-negotiation transactions take place between the vehicles through BSMs once they are platoon-ready to check if their destinations match. If the destinations are different, this algorithm evaluates their global routes to get the closest match. A negotiation is then sent to the vehicle with the matching destination once a common destination has been identified. Fig. 7 clearly depicts pre-negotiation transactions that occur between Vehicle 1 and Vehicle 2 once they are platoon-ready. Once Vehicle 1 is platoon-ready, it broadcasts its destination and path to Vehicle 2 through BSM. Vehicle 1 now constantly checks to see if Vehicle 2's destination and route are available. Once ready, Vehicle 1 compares its destination to the destination of Vehicle 2. If they do not match, their global routes are compared to identify the best possible match. After finding a match, Vehicle 1 begins negotiating by issuing a PJRQ. The same logic is executed in Vehicle 2 (shown on the right side of Fig. 7), which begins negotiating once a match is identified. Once the vehicles begin negotiating, the negotiations are handled using the negotiation resolver algorithm.

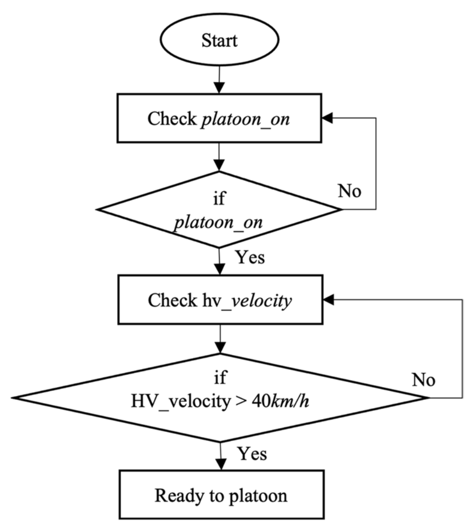
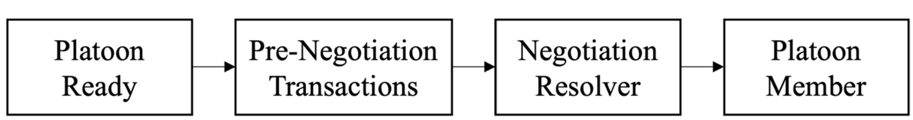


Fig. 6 Platoon-ready flowchart

Fig. 5 Relationship between the two-vehicle platoon algorithms



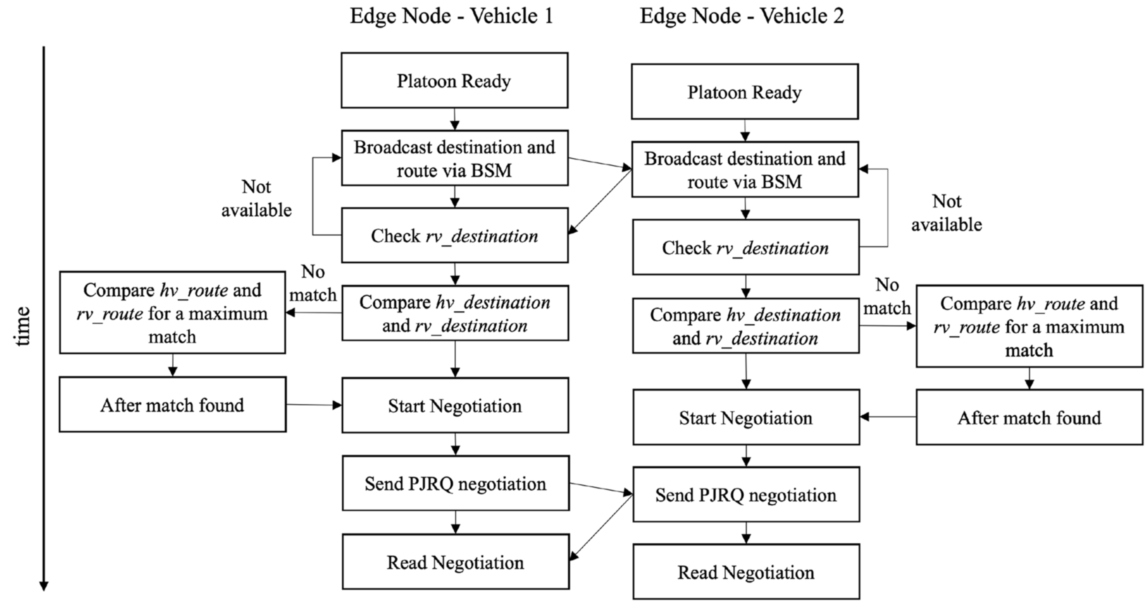


Fig. 7 Pre-negotiation transactions

Negotiation Resolver Algorithm

The negotiation resolver algorithm resolves the negotiations between two vehicles over time as shown in Fig. 8. Once the vehicle is ready to platoon and begins to negotiate, a PJRQ negotiation is sent to other vehicles through BSMs. The negotiation structure includes a *negotiation*, *sender_id*, and *receiver_id* and the *negotiation* may be either PJRQ, PA, PJRY, or PLR. Other variables *sender_id* and *receiver_id* represent HV id and RV id respectively. This way, only the vehicles interacting can process the negotiation messages. The negotiation resolver, which runs independently on each vehicle, processes negotiations received through the BSMs. As shown in Fig. 8,

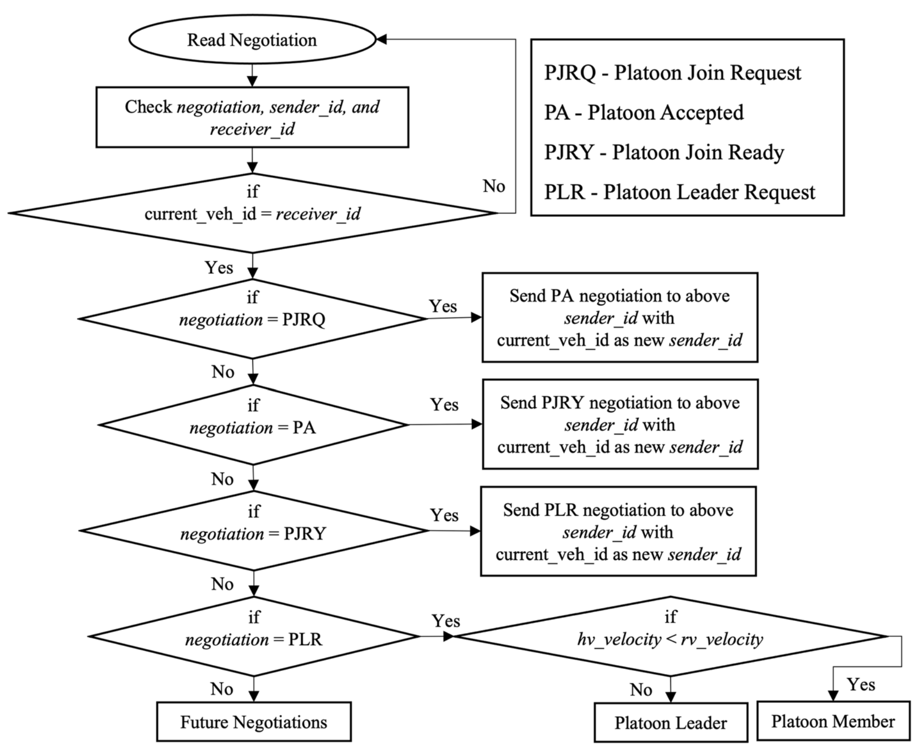


Fig. 8 Negotiation Resolver Flowchart

the resolver reads these negotiations and handles them accordingly.

Vehicles process received BSMs, and when a negotiation is read from the BSM, the *receiver_id* in the BSM is compared to the current vehicle's id 'current_veh_id' or HV id that is processing the negotiation. If the current_veh_id matches the *receiver_id*, the received negotiation is processed further. If the *negotiation* is a PJRQ, the algorithm accepts it and sends a PA negotiation back to the above *sender_id* or RV id. The current HV id becomes the new *sender_id* and the RV id becomes the new *receiver_id* when sending a PA negotiation back. This is a similar concept to the remainder of the negotiations. Following that, if the *negotiation* is a PA, a PJRY negotiation is acknowledged back to the above *sender_id* or RV id. At this point, both vehicles have acknowledged that they are ready to form a platoon. The next negotiations would be to establish who would head the platoon. For this, we assume that each vehicle declares itself to be PL and sends PLR negotiations. If the negotiation is a PLR, the current vehicle or HV compares its velocity to that of the sender vehicle or RV during processing. If *hv_velocity* is less than *rv_velocity*, meaning that it is driving slower than the remote, it becomes a PM.

The criteria set to become a PL here are arbitrary and can be changed according to the requirements. This way a negotiation resolver is implemented to settle the negotiations between the vehicles to initiate platooning.

Platoon Member Algorithm

Once PM and PL are determined, this logic is used by PM to follow PL based on the relative angle 'θ'. Figures 9, 10, 11 shows when PL is on the right, left, and same lanes of PM, respectively. Here, we assume that H1 is the heading angle of PM, H2 is the heading angle of the new vector AB joining PM and PL, and H3 is the heading angle of PL. Relative angle θ is the difference between H1 and H2. First, we check if both PM and PL are heading in the same direction by calculating 'α', the difference in their heading angles (H1–H3). When mod α is less than 5° meaning PM and PL are heading in the same direction, the relative angle θ between PM and PL is computed. PM and PL are in the same lane when mod θ is less than 1°. In this scenario, the PM will follow

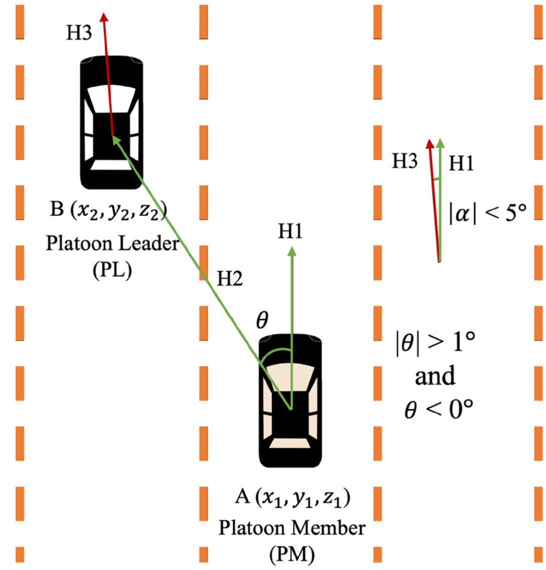


Fig. 10 PM illustration when PL in left lane

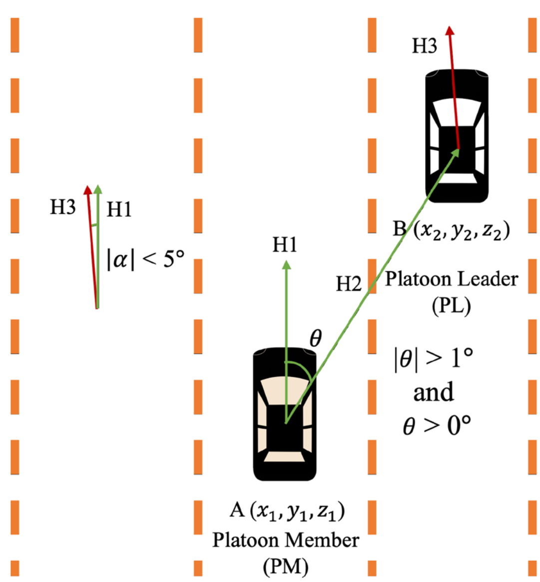


Fig. 9 PM illustration when PL in right lane

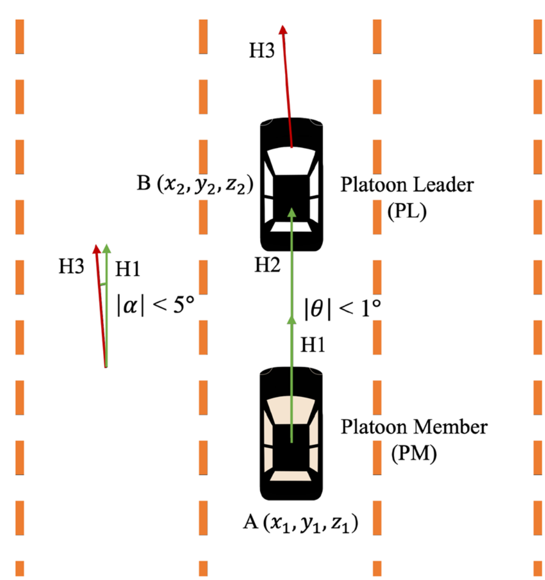


Fig. 11 PM illustration when PL in the same lane

PL in the same lane as shown in Fig. 11. PM and PL are in different lanes when mod θ is greater than $1°$. PL is on the right lane of PM when θ is positive or greater than $0°$. In this case, the PM will drive to its right lane to follow PL as shown in Fig. 9. Lastly, PL is on the left lane of PM when θ is negative or less than $0°$. Here, the PM will drive to its left lane to follow PL as shown in Fig. 10. Figure 12 shows the above logic in a flowchart.

This way vehicles can negotiate with each other to find a platoon and then join using BSM information only. Further, we have conducted experiments to evaluate the above algorithm and the results are shown in "Experimental results". We used PM algorithms as a base to develop platoon formation and maintenance algorithms for a non-existing two-vehicle platoon in our latest work [54].

Multi-vehicle Platoon

In this section, we extend the two-vehicle platoon algorithms presented in "Two-vehicle platoon" to support a multi-vehicle platoon. When compared to the two-vehicle platoon we

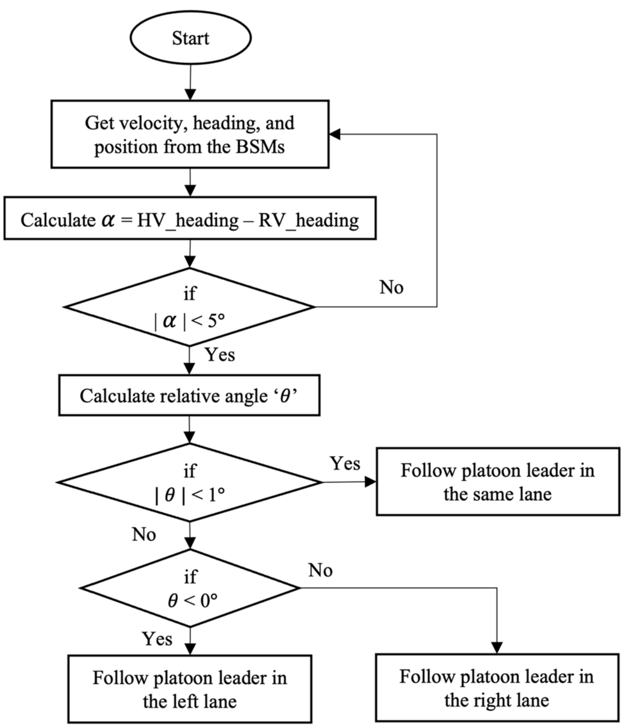


Fig. 12 Platoon Member Flowchart

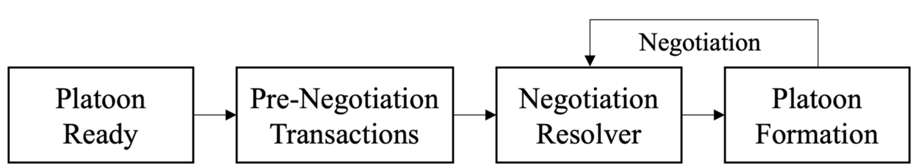
published earlier, we encountered several challenges while implementing multi-vehicle platoon negotiation and formation algorithms as it did not meet the requirements. This is mainly due to the increased member vehicles leading to an increase in the platoon size. As the platoon size increased, it led to more complexity in communication and coordination among the vehicles in the platoon. We applied several strategies to the existing two-vehicle platoon algorithms to overcome these challenges, including the introduction of global and local leader roles. This concept helps in organizing and coordinating the platoon's activities. Here, PL is assigned as a global leader which is responsible to oversee the entire platoon's operations such as initiating negotiations, coordinating communication, and making decisions for the platoon as a whole. On the other hand, the local leader is assigned to each individual PM to facilitate communication and coordination between global leader and its associated PM.

With this, there is no change to the platoon-ready algorithm while pre-negotiation, negotiation resolver, and platoon member algorithms are updated to support multi-vehicle platooning. Also, we renamed the "Platoon Member" algorithm to the "Platoon Formation" algorithm to accurately reflect the expanded functionality of this algorithm, which now encompasses the broader platoon formation process. The relationship between these algorithms and the order in which they are executed is illustrated in Fig. 13. Using these, two vehicles can negotiate and start a platoon where we assign PL as the global and the local leader for the first PM. Once the platoon is complete between these two vehicles, the PL makes itself available to negotiate with nearby vehicles. Any future PM can now negotiate with this PL to join its platoon. After their negotiations, we assign PL as the global leader and the existing PM as the local leader to the newly joined PM. A clear explanation of each of the updated algorithms is presented in this section.

Multi-vehicle Pre-negotiation Transactions

When the vehicles are platoon-ready, they broadcast their intent to platoon through BSM containing their destination and route. When nearby vehicles receive BSMs, they check if the vehicle that sent the BSM is platoon-ready and willing to platoon. If yes, their destinations are compared to check if they both are traveling to the same destination. If their destinations do not match, a maximum global match is found. Once a common destination is identified, the vehicles check if the RV is available for

Fig. 13 Relationship between the multi-vehicle platoon algorithms



negotiation. If the RV is available to negotiate, the HV then checks if there is an active platoon. If there is no active platoon, HV marks itself as busy and starts negotiating. If there is an active platoon, HV waits until it receives a BSM from PL of the active platoon, marks itself as busy, and starts to negotiate with PL.

Pre-negotiation transactions happening between edge nodes Vehicle 1 and Vehicle 2 over time are depicted in detail in Fig. 14 after they are platoon-ready. Vehicle 1 broadcasts its destination and route to Vehicle 2 through a BSM and continuously checks if the destination and route for Vehicle 2 are available through incoming BSMs. When available, Vehicle 1 compares both destinations to find a common match. If a common destination is present, Vehicle 1 checks if Vehicle 2 is available to negotiate. If available, then it checks if there is an active platoon system. If an active platoon is present, it checks if Vehicle 2 is a platoon leader. If Vehicle 2 is not a leader, Vehicle 1 changes its status to busy and starts to negotiate by sending a Platoon Join Request (PJRQ) negotiation. Vehicle 2, which is shown on the right side of Fig. 14, follows the same logic and begins negotiating as soon as a common destination is identified and status is available. The negotiation resolver algorithm, which is described in "Multi-vehicle negotiation resolver algorithm", is used to resolve the negotiations.

Multi-vehicle Negotiation Resolver Algorithm

At a high level, the negotiation resolver executes three types of logic. The first type resolves negotiations between two vehicles to initialize a non-existing platoon. Once initialized, the other two support negotiations between PL and any future PMs and vice versa. For this, we initialize '*join_ready*', '*active_platoon*', and '*leader*' variables to false. Here, *join_ready* is set to true after successful platoon negotiations for all vehicles, *active_platoon* to true once there is any active PL, and *leader* is set to true for PL. Variables '*global_leader*' and '*local_leader*' are initialized to 'None', where *global_leader* will be set to PL and *local_leader* as the last joined PM in the platoon. Lastly, variables '*member_list*' and '*member_count*' are set to 'None' that are used by PL to track PMs. Note that the same algorithm runs in every vehicle and logic is executed based on the situation. With this background, we now explain the negotiation resolver algorithm as shown in Fig. 15.

After pre-negotiation transactions, the vehicle marks itself busy and will not accept negotiations from any vehicles other than the vehicle it is negotiating with. We now assume that there is no active platoon and two vehicles are exchanging IoT-based BSMs containing vehicle information accompanied by a negotiation. Now, the negotiation resolver running independently in each vehicle processes the BSM and reads the negotiation. The structure of the negotiation contains

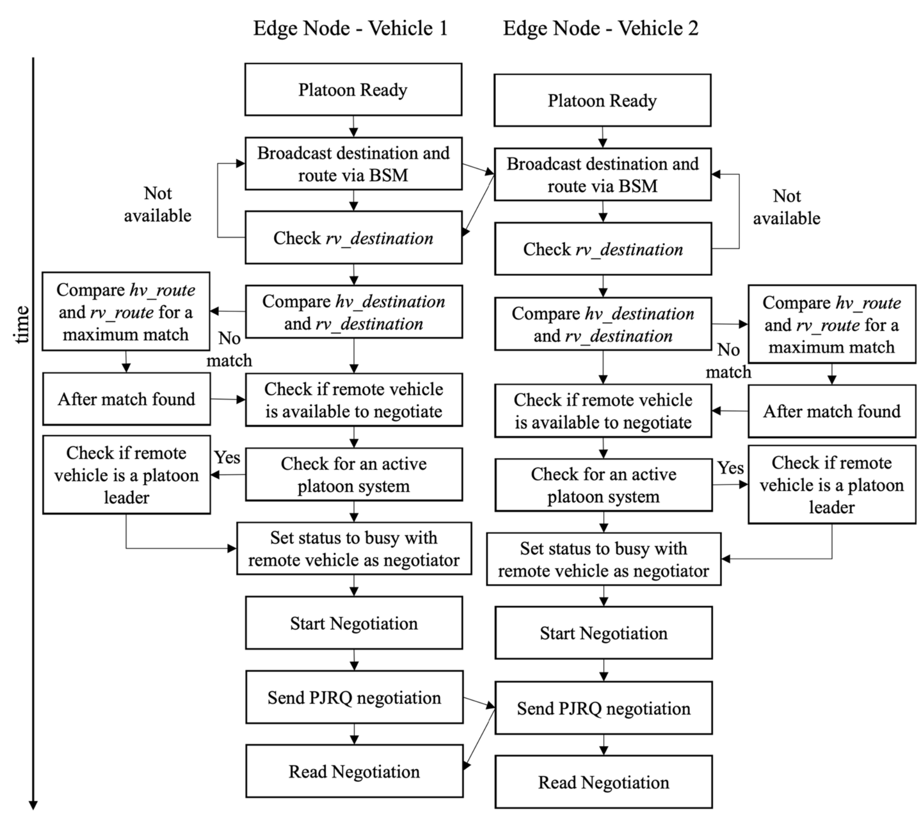


Fig. 14 Multi-vehicle pre-negotiation transactions

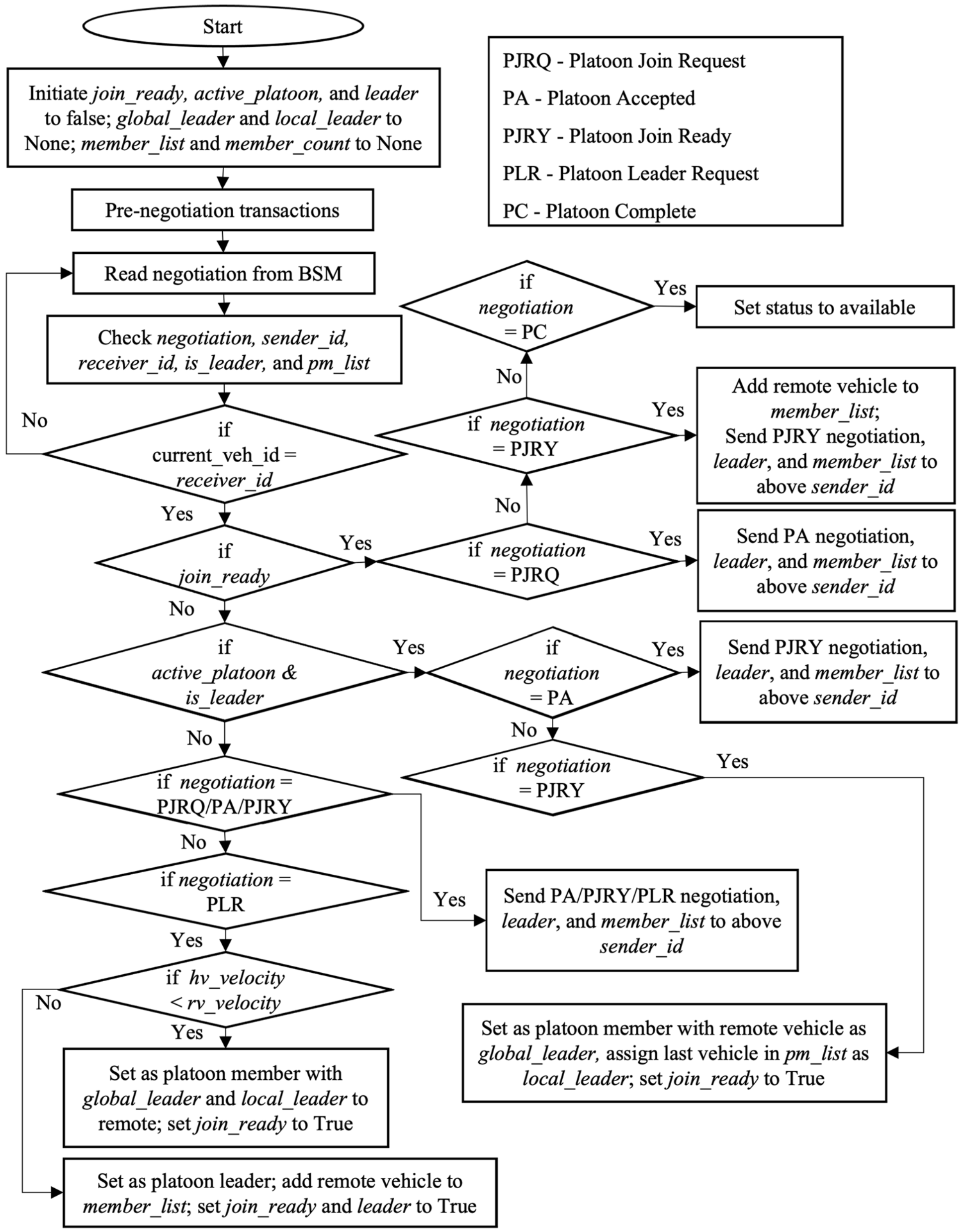


Fig. 15 Multi-vehicle negotiation resolver flowchart

a '*negotiation*', '*sender_id*', '*receiver_id*', '*is_leader*', and '*pm_list*' where a vehicle with id '*sender_id*' is sending a negotiation '*negotiation*' to a vehicle with id '*receiver_id*'. Here, *is_leader* indicates whether the vehicle sending the negotiation is a platoon leader or not and *pm_list* contains the list of PMs that can be used by a future PM to find its local leader.

After receiving and reading the negotiation, the algorithm checks if the negotiation is intended for the current vehicle. If it is, then it checks for the variable *join_ready*. Since this variable is initialized to false, the algorithm checks for the variables *is_leader* from negotiation and *active_platoon*. Since these two variables will be false initially, negotiations are carried out until one of them becomes a PM and the other PL as explained in our prior work [53]. For PM, PL is assigned as the *global_leader* and *local_leader* while *join_ready* is set to true. For PL, PM is added to the *member_list*, *member_count* incremented by one, and variables *join_ready* and *leader* are set to true. Using the platoon formation algorithm presented in "Multi-vehicle platoon formation algorithm", PM joins in PL's lane and then sends a Platoon Complete (PC) negotiation to the PL. When PL receives this negotiation and processes, it enters into a second type of logic from here onwards since *join_ready* is set to true. Here, if the negotiation is PC, PL marks itself as available to negotiate with nearby vehicles.

The new vehicle that is willing to platoon sets *active_platoon* to true if there is an active platoon and initializes pre-negotiation transactions. If their destinations match, the new vehicle sets itself busy and sends a PJRQ negotiation to the leader. When PL receives this negotiation, the algorithm sends a Platoon Accept (PA) negotiation back to the new vehicle along with *leader* as true. When the new vehicle receives this negotiation, the negotiation resolver enters into the third type of logic since there is already an active platoon and the negotiation is from the leader itself. If the received negotiation is PA, a Platoon Join Ready (PJRY) negotiation is sent back to PL. When PL receives PJRY negotiation, it adds the new vehicle into its *member_list* and sends a PJRY negotiation back. If the new vehicle receives PJRY as negotiation, it sets itself as platoon member, *global_leader* as PL, *local_leader* as last joined PM before it from *pm_list*, and sets *join_ready* to true. Using the platoon formation algorithm, a new PM joins the platoon and sends a PC negotiation back to PL. Any future vehicle can join an existing platoon similarly using the second and third types of logic.

Multi-vehicle Platoon Formation Algorithm

Once PM and PL are determined, PM uses the platoon formation algorithm shown in Fig. 16 to join the PL with more technical details provided in Fig. 17. This algorithm checks if a PM is *join_ready* and that the received BSM is from its *local_leader*. After verifying, PM joins PL in its lane as discussed in our latest work [54]. Once PM reaches the *platoon_maintain* state, it sends a PC negotiation to the global leader. These are the changes made to the platoon formation algorithm to support multi-vehicle platooning.

Experimental Results

To validate the algorithms proposed in this work, we implemented and tested them within the CARLA Connect simulator, an extension of CARLA. Vehicle connectivity and platooning are not supported in CARLA by default. For this reason, a DSRC agent is created in CARLA Connect to enable connectivity between the vehicles in simulation. In order to obtain all the necessary vehicle information, we created a custom sensor called the OBU sensor on the server side of the simulator. This sensor is attached to every spawn vehicle to collect information such as the vehicle's position, speed, heading, etc. Using the DSRC agent, vehicles deployed with the custom OBU sensor can now send and receive BSMs. In our study, the range for simulated BSMs exchange is set to 1000 m, which is managed by the DSRC agent. This setting carefully emulates realistic communication distances within the context of the simulation environment. The selected range is based on the capabilities of DSRC technology and is designed to ensure effective message exchange between vehicles.

To enable platooning, a platoon agent (PA) is created on the client side of the simulator where all the proposed

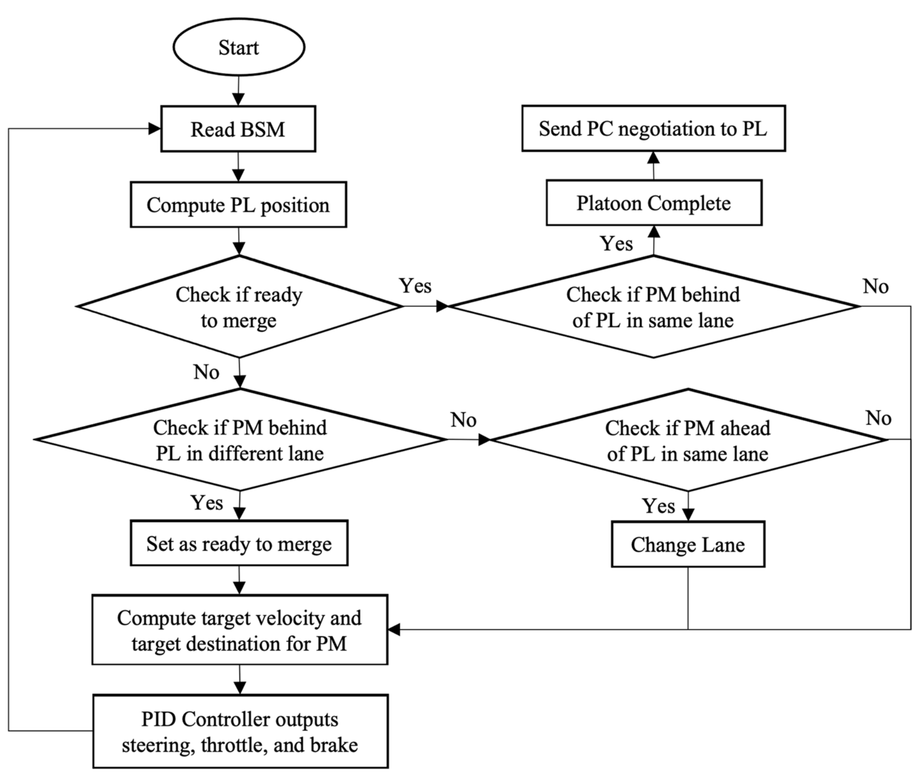


Fig. 16 High-level platoon formation flowchart

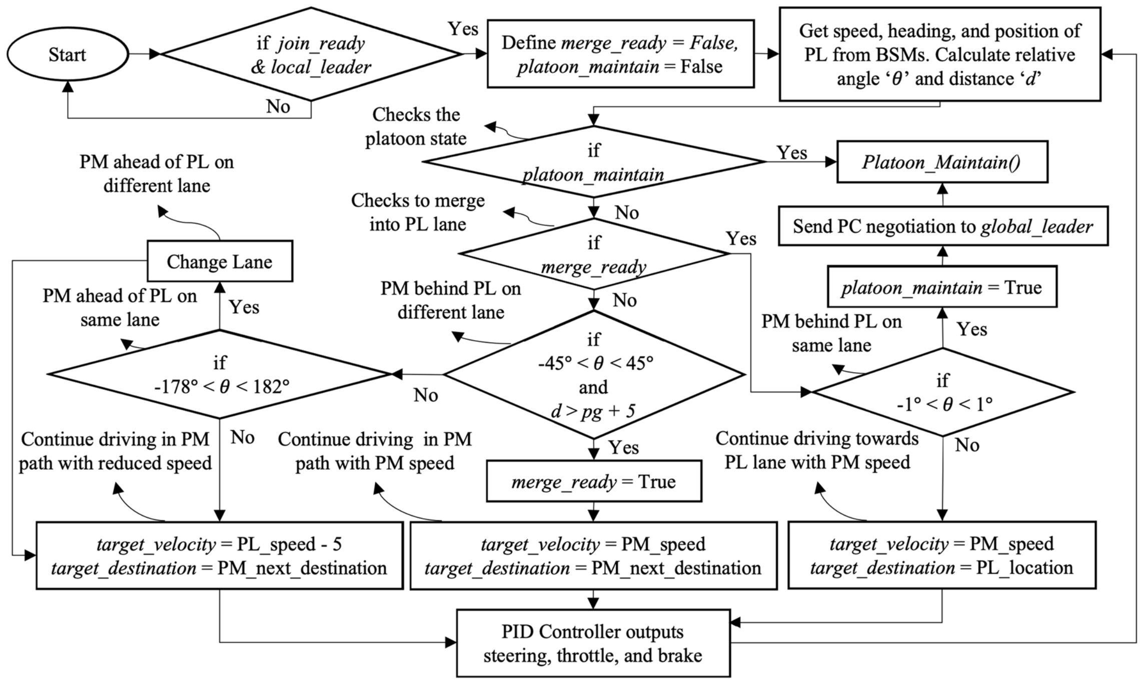


Fig. 17 Platoon formation flowchart

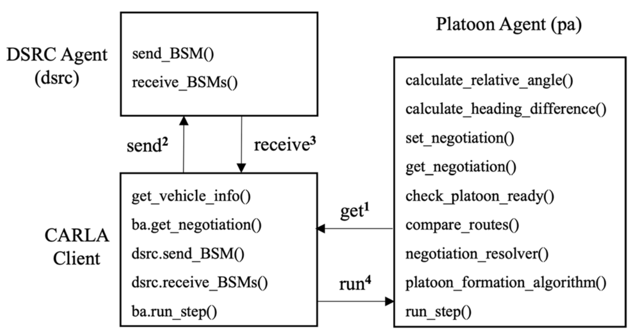


Fig. 18 Connectivity architecture implemented in CARLA

platoon algorithms are implemented. This agent is responsible for carrying out the driving functionalities of the simulated vehicle from the source to the destination. Fig. 18 provides a high-level picture of how platooning and connectivity are implemented in CARLA Connect. First, each vehicle on the client side reads its own negotiation from the PA agent. Now, the vehicles broadcast their information along with their negotiation to the DSRC agent and then receive BSMs from other vehicles. Vehicles now send this information to the PA objects which run the platoon algorithms. A detailed explanation of CARLA Connect is provided in our work [33].

This simulator has a wide variety of maps from a simple town to an urban environment. Out of these, we used CARLA Town06 to test our algorithms, as it has extensive roadways and several highway entrances and exits that are practical for this application. We chose a long highway in this map to simulate platooning and deployed each vehicle with an OBU sensor attached to it. Other simulation parameters which we have used are the maximum vehicle speed of 70 km/h, a platoon gap of 10 m, and a braking distance of 5 m. With this, we have conducted experiments with two and three vehicles to negotiate and form a platoon. In a three-vehicle scenario, we also experimented with stop and lane-change maneuvers. A video demonstration of this experiment is available on YouTube at [55].

Same Destination Experiment

In this experiment, two vehicles with the same destination are spawned in two distinct lanes as shown in Fig. 19. The leading vehicle is an RV while the following vehicle on the left lane to the RV is HV. RV started off first and achieved a speed of 40 km/h before HV. At this point, the RV is ready to platoon and began to broadcast its destination. Similarly, once the speed criteria is met by HV, it started to broadcast its destination. At this point, both the HV and the RV exchanged both of their respective destinations. As their destinations are the same, a PJRQ negotiation is set in its PA object, which can later be retrieved by the CARLA client. First, RV reads its negotiation for HV and broadcasts it to the DSRC agent through the BSM, and the same is done by HV. Both HV and RV now receive their BSMs from the

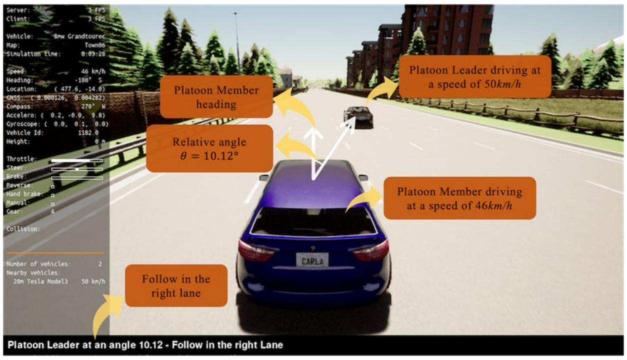


Fig. 19 PM driving at speed of 46 km/h to join PL on the right lane at an angle of $\theta = 10.12°$ ahead of PM

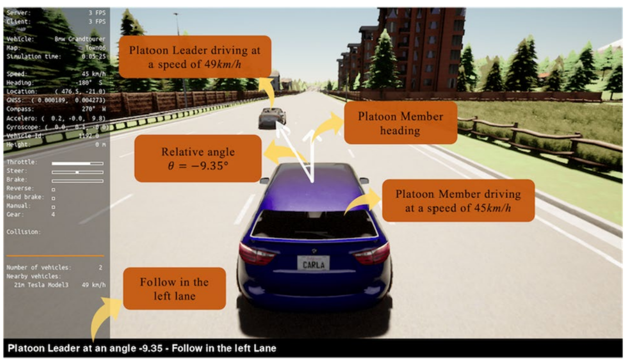


Fig. 20 PM driving at speed of 45 km/h to join PL on the left lane at an angle of $\theta = -9.35°$ ahead of PM

DSRC agent and send them to their PA object, which now includes negotiations.

The negotiation resolver reads the negotiation and analyzes it using the logic described in Fig. 8. When both vehicles are in the PJRY condition, a PLR request is made, and the vehicle with the higher speed becomes the PL. RV speed is higher than HV in this scenario and is assigned as PL, while HV is assigned as PM. After determining PL and PM, the heading angle difference *alpha* and relative angle *theta* are compared. After verifying, PM drives to the right lane to join the platoon with PL, as shown in Fig. 19.

Different Destination Experiment

This experiment is identical to the same destination experiment, except that the vehicles are spawned with different destinations. In this case, the *compare_routes*() method in the individual PA object is used to find a common match. Following successful discussions, RV becomes PL and HV becomes PM. After determining PL and PM, the heading angle difference α and relative angle θ are compared. After verifying, PM drives to the left lane to join the platoon with PL, as shown in Fig. 20.

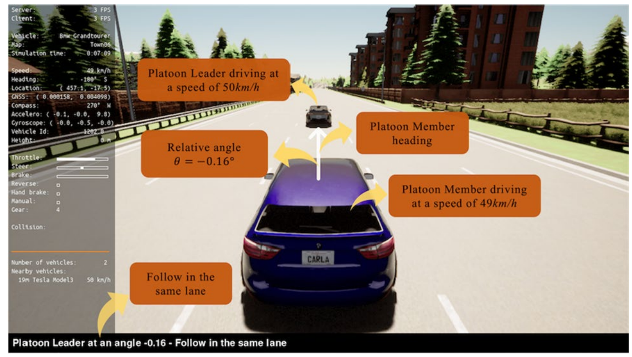


Fig. 21 PM driving at speed of 49 km/h to join PL on the same lane at an angle of $\theta = -0.16°$ ahead of PM

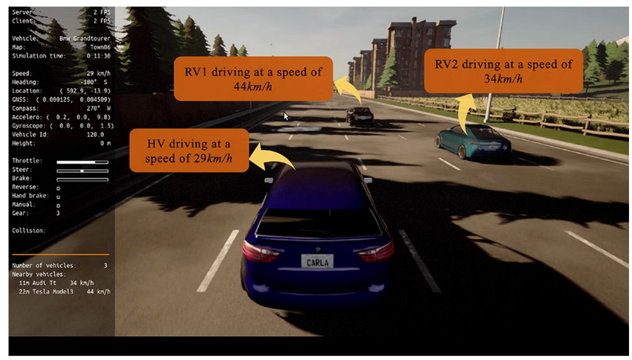


Fig. 22 Three vehicles spawned in CARLA town

Same Lane Experiment

This experiment is identical to the same destination experiment in "Same destination experiment" except that the vehicles are spawned in the same lanes. Vehicles began negotiating after their locations are matched in this scenario. After successful discussions, RV becomes PL and HV becomes PM. After determining PL and PM, the heading angle difference α and relative angle θ are compared. PM joins the platoon with PL after confirming by driving in the same lane as shown in Fig. 21.

Three-Vehicle Platoon Experiment

To validate the proposed algorithms, we have spawned three vehicles in three different lanes heading to the same destination as shown in Fig. 22. The vehicle in the last is HV, while the other two are RVs. The condition for the vehicles to start platoon is to reach a speed greater than 40 km/h. First, RV1 reached a speed greater than 40 km/h. Once reached, it is ready to platoon and started to broadcast its destination to the other two vehicles. RV2 was traveling faster than HV and started to broadcast its destination once reaching the set condition. After RV1 and RV2 exchanged their destinations, negotiations were carried out between these two vehicles. Once the negotiations were complete, RV1 became PL and RV2 became PM. PM started to follow PL in its lane and sent a PC negotiation to PL once completing the platoon. Reading this message, PL set itself to available.

HV starts to broadcast its destination and its intent to platoon once reaching the set condition. Since the other two vehicles were busy, HV could not negotiate with them during this entire time. Once PL or RV1 set itself to available, it reads HV's destination and intent to platoon. Now, both PL and HV mark themselves busy and negotiate with each other. After successful negotiations, HV is assigned RV1 as its global leader and RV2 as its local leader. HV or second PM joins the platoon and sends a PC negotiation to the global leader which then sets itself as available. In this situation, RV2 receives platoon instructions from RV1 while HV receives instructions from RV2. This way, we have successfully demonstrated a three-vehicle platoon as shown in Fig. 23.

Once three vehicles formed the platoon, we further experimented with stop and lane-change maneuvers. As PL made a stop at the right light, both the PMs came to a complete stop as shown in Fig. 24. Similarly, when PL made a lane change maneuver, both the PMs started to change lanes with RV2 first followed by HV as shown in Fig. 25

Conclusion

The Internet of Things (IoT) has a wide range of uses. This study highlights how the Internet of Vehicles (IoV), a subset of IoT, can address transportation challenges. We built upon our previous work to enhance platoon negotiation and formation algorithms for multi-vehicle groups. Our proposed algorithms were successfully tested with three-vehicle platoon experiments, including maneuvers like stopping and lane changes. Unlike traditional methods relying on cameras and LIDARs that can struggle in bad weather, our algorithm based on BSMs is resilient even in adverse conditions. However, it is important to clarify that our approach is not intended to replace sensors like cameras; rather, it complements their capabilities. It is also worth noting that our approach is distinct from Cooperative Adaptive Cruise Control (CACC), which relies on radar and cameras to detect preceding vehicles-our method does not rely on these sensors.

Considering potential enhancements, it may be feasible to incorporate positioning data from Global Navigation Satellite Systems (GNSS) in conjunction with inertial navigation systems for scenarios such as tunnels. This possibility, however, necessitates a comprehensive discussion inclusive of the associated challenges in terms of positioning errors and accuracy. There are certain limitations to our work. The relative position methodology is developed and tested in a simulation environment, which may differ from real-world conditions. Another limitation is that our current algorithms do not accommodate sharp turns; however, this aspect will be addressed in future research. The future work also involves extending the algorithms to support platoon merge and split maneuvers. Also, the current focus of this work encompasses the entire process of platoon formation, while aspects like engagement with non-platoon traffic will be covered in subsequent research.

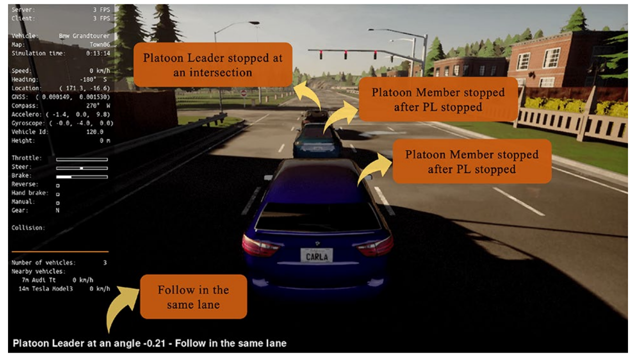


Fig. 24 Platoon stopped after PL stopped at red light

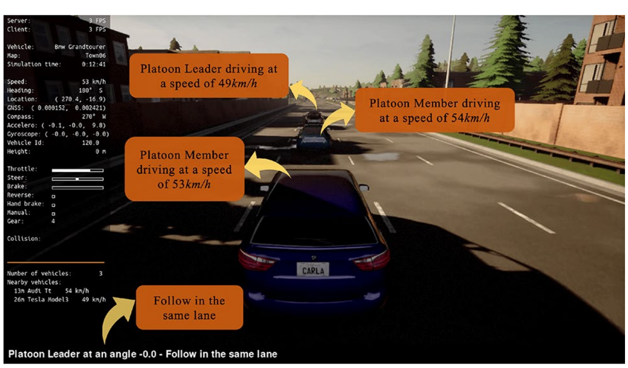


Fig. 23 PM driving at speed of 53 km/h after forming a platoon

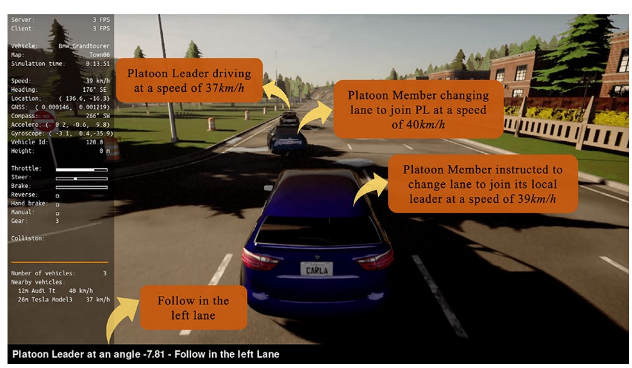


Fig. 25 PM driving at speed of 39 km/h instructed to join its local leader in left lane

Declarations

Conflict of interest On behalf of all authors, the corresponding author states that there is no conflict of interest.

References

1. Al-Fuqaha A, Guizani M, Mohammadi M, Aledhari M, Ayyash M. Internet of things: a survey on enabling technologies, protocols, and applications. IEEE Commun Surv Tutor. 2015;17(4):2347–76.
2. Hanes D, Salgueiro G, Grossetete P, Barton R, Henry J, Trollope R. IOT fundamentals networking technologies, protocols, and use cases for the internet of things. Cisco Press. 2017.
3. Pawar N, Bourgeau T, Chaouchi H. Study of IoT architecture and application invariant functionalities. In: 2021 IFIP/IEEE International Symposium on Integrated Network Management (IM), pp. 667–671. 2021.
4. Leslie A, Murray D. An analysis of the operational costs of trucking: 2022 update. Am Transp Res Inst (ATRI). 2022.
5. American Trucking Associations. Economics and Industry Data. Accessed 20 Dec. 2022.
6. U. S. Environmental Protection Agency. Sources of Greenhouse Gas Emissions. Accessed 21 Dec. 2022. 2022.
7. Bergenhem C, Shladover S, Coelingh E, Englund C, Tsugawa S. Overview of platooning systems. In: Proceedings of the 19th ITS World Congress, Oct 22-26, Vienna, Austria (2012). 2012.
8. U.S. Department of Transportation. Connected Vehicle Pilot Deployment Program. Accessed: 2022-02-06.
9. Lochrane T, Dailey L, Tucker C. CarmaSM: driving innovation. Pub Roads. 2020;83(4):28–31.
10. Oquendo F. Architecting exogenous software-intensive systems-of-systems on the internet-of-vehicles with sosadl. Syst Eng. 2019;22(6):502–18.
11. Shen X, Fantacci R, Chen S. Internet of vehicles [scanning the issue]. Proc IEEE. 2020;108(2):242–5.
12. Florida Department of Transportation. Driver Assisted Truck Platooning (DATP) Pilot. Accessed: 2022-02-07.
13. Crane C, Bridge J, Bishop R. Driver assistive truck platooning: considerations for florida state agencies. Accessed: 2022-02-06. 2018.
14. Foreman C, Keen M, Petrella M, Plotnick S. FHWA research and technology evaluation: truck platooning. FHWA-HRT-20-071, Tech Report. 2021.
15. Dokur O, Katkoori S. Three connected V2V applications based on DSRC basic safety messages. In: 2022 International Conference on Connected Vehicle and Expo (ICCVE), pp. 1–6. 2022.
16. Rajasekhar M V, Jaswal A K. Autonomous vehicles: the future of automobiles. In: 2015 IEEE International Transportation Electrification Conference (ITEC), pp. 1–6. 2015.
17. SAE. Taxonomy and definitions for terms related to driving automation systems for on-road motor vehicles. SAE J3016. Tech. Rep. 2018.
18. Badue C, Guidolini R, Carneiro RV, Azevedo P, Cardoso VB, Forechi A, Jesus L, Paixão Berriel R, T M, Mutz F, de Paula Veronese L, Oliveira-Santos T, De Souza A F,. Self-driving cars: a survey. Expert Syst Appl. 2021;165: 113816.
19. Yurtsever E, Lambert J, Carballo A, Takeda K. A survey of autonomous driving: common practices and emerging technologies. IEEE Access. 2020;8:58443–69.
20. Thakurdesai H M, Aghav J V,. Autonomous cars: technical challenges and a solution to blind spot. In: Advances in Computational Intelligence and Communication Technology, pp. 533–547, Springer. 2021.
21. Kim N, Karbowski D, Jeong J, Rousseau A. Simulation of heavy-duty vehicles in platooning scenarios. In: 2018 21st International Conference on Intelligent Transportation Systems (ITSC), pp. 1604–1610. 2018.
22. Vasili A, Moreno W. A. Applications and trends in connected vehicles: debates and conclusions. In: 2019 IEEE International Conference on Connected Vehicles and Expo (ICCVE), pp. 1–5. 2019.
23. Hartenstein H, Laberteaux L. A tutorial survey on vehicular ad hoc networks. IEEE Commun Mag. 2008;46(6):164–71.
24. Hameed Mir Z, Filali F. Lte and ieee 80.211 p for vehicular networking: a performance evaluation. EURASIP J Wirel Commun Network. 2014;2014(1):1–15.
25. Dow C.-R, Ho M, Lee Y.-H, Hwang S.-F. Design and implementation of a dsrc based vehicular warning and notification system. In: 2011 IEEE International Conference on High Performance Computing and Communications, pp. 960–965, IEEE, 2011.
26. Shladover SE. Path at 20-history and major milestones. IEEE Trans Intell Transp Syst. 2007;8(4):584–92.
27. Krupitzer C, Segata M, Breitbach M, El-Tawab S, Tomforde S, Becker C. Towards infrastructure-aided self-organized hybrid platooning. In: 2018 IEEE Global Conference on Internet of Things (GCIoT), pp. 1–6. 2018.
28. Ghosal A, Sagong SU, Halder S, Sahabandu K, Conti M, Poovendran R, Bushnell L. Truck platoon security: state-of-the-art and road ahead. Comput Netw. 2021;185: 107658.
29. Lesch V, Breitbach M, Segata M, Becker C, Kounev S, Krupitzer C. An overview on approaches for coordination of platoons. IEEE Trans Intell Transp Syst. 2021;1–17.
30. Sturm T, Krupitzer C, Segata M, Becker C. A taxonomy of optimization factors for platooning. IEEE Trans Intell Transp Syst. 2020;22(10):6097–114.
31. Weber JS, Neves M, Ferreto T. Vanet simulators: an updated review. J Braz Comput Soc. 2021;27(1):1–31.
32. Dosovitskiy A, Ros G, Codevilla F, Lopez A, Koltun V. CARLA: an open urban driving simulator. In: Proceedings of the 1st Annual Conference on Robot Learning, pp. 1–16. 2017.
33. Dokur O, Katkoori S. CARLA connect: a connected autonomous vehicle (CAV) Driving Simulator. In : 2022 IEEE International Symposium on Smart Electronic Systems (iSES), pp. 656–659. 2022.
34. Sturm T, Krupitzer C, Segata M, Becker C. A taxonomy of optimization factors for platooning. IEEE Trans Intell Transp Syst. 2020;22(10):6097–114.
35. U.S. Department of Transportation, “CARMA.” Accessed: 2022-02-06.
36. Chakraborty T, Yamaguchi S, Datta S. K. Sensor fusion and adaptive cruise control for self driving platoon. In: 2018 IEEE 7th Global Conference on Consumer Electronics (GCCE), pp. 231–235. 2018.
37. Petrov V, Samuylov A, Begishev V, Moltchanov D, Andreev S, Samouylov K, Koucheryavy Y. Vehicle-based relay assistance for opportunistic crowdsensing over narrowband IoT (NB-IoT). IEEE Internet Things J. 2018;5(5):3710–23.
38. Sodhro AH, Obaidat MS, Abbasi QH, Pace P, Pirbhulal S, Yasar A-U-H, Fortino G, Imran MA, Qaraqe M. Quality of service optimization in an iot-driven intelligent transportation system. IEEE Wirel Commun. 2019;26(6):10–7.
39. Niu Z, Shen XS, Zhang Q, Tang Y. Space-air-ground integrated vehicular network for connected and automated vehicles: challenges and solutions. Intell Converg Netw. 2020;1(2):142–69.
40. Wang X, Wang D, Ariyasu N, Umehira M. Better platooning toward autonomous driving: inter-vehicle communications with directional antenna. China Commun. 2021;18(7):44–57.

41. Guo F, Yu FR, Zhang H, Li X, Ji H, Leung VCM. Enabling massive iot toward 6g: a comprehensive survey. IEEE Internet Things J. 2021;8(15):11891–915.
42. Sigov A, Ratkin L, Ivanov L A, Xu L D. Emerging enabling technologies for industry 4.0 and beyond. Inform Syst Front. 2022;1–11.
43. Kouchak S. M, Gaffar A. Determinism in future cars: why autonomous trucks are easier to design. In: 2017 IEEE SmartWorld, Ubiquitous Intelligence Computing, Advanced Trusted Computed, Scalable Computing Communications, Cloud Big Data Computing, Internet of People and Smart City Innovation (SmartWorld/SCALCOM/UIC/ATC/CBDCom/IOP/SCI), pp. 1–6. 2017.
44. El-Said M, Bhuse V, Arendsen A. An empirical study to investigate the effect of air density changes on the DSRC performance. Proc Comput Sci. 2017;114:523–30.
45. Huang X, Zhao D, Peng H. Empirical study of DSRC performance based on safety pilot model deployment data. IEEE Trans Intell Transp Syst. 2017;18(10):2619–28.
46. Rajamani R, Tan H-S, Law BK, Zhang W-B. Demonstration of integrated longitudinal and lateral control for the operation of automated vehicles in platoons. IEEE Trans Control Syst Technol. 2000;8(4):695–708.
47. Amoozadeh M, Deng H, Chuah C-N, Zhang HM, Ghosal D. Platoon management with cooperative adaptive cruise control enabled by VANET. Veh Commun. 2015;2(2):110–23.
48. Saeednia M, Menendez M. Analysis of strategies for truck platooning: hybrid strategy. Transp Res Record. 2016;2547(1):41–8.
49. Back J, Choi S, Shin Y. A study on the election of suitable leader vehicle in vehicle platooning using the raft algorithm. In: 2019 Eleventh International Conference on Ubiquitous and Future Networks (ICUFN), pp. 756–761. 2019.
50. Choi S, Back J, Shin Y. A study on the selection of candidate leaders for flexible platooning in urban road. In: 2019 Eleventh International Conference on Ubiquitous and Future Networks (ICUFN), pp. 725–728. 2019.
51. Burov M, Mehr N, Smith S, Kurzhanskiy A, Arcak M. Platoon formation algorithm for minimizing travel time. In: 2020 IEEE 23rd International Conference on Intelligent Transportation Systems (ITSC), pp. 1–6. 2020.
52. Ganaoui-Mourlan E, Camp S, Hannagan T, Arora V, De Neuville M, Kousournas VA, et al. Path planning for autonomous platoon formation. Sustainability. 2021;13(9):4668.
53. Dokur O, Olenscki G, Katkoori S. An edge computing approach for autonomous vehicle platooning. In: IFIP International Internet of Things Conference, pp. 332–349, Springer. 2022.
54. Dokur O, Olenscki G, Katkoori S. Platoon Formation Based on DSRC Basic Safety Messages. In: 2022 IEEE International Symposium on Smart Electronic Systems (iSES), pp. 700–705. 2022.
55. Dokur O, Katkoori S. Three-vehicle platoon formation based on DSRC basic safety messages. 2023. https://youtu.be/d8e-0-Cm1LA. Accessed on 2023/01/06.