IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received July 2, 2021, accepted July 20, 2021, date of publication August 20, 2021, date of current version September 23, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3106455

A Hierarchical State-Machine-Based Framework for Platoon Manoeuvre Descriptions

JORDAN IVANCHEV[1,2], (Member, IEEE), CORVIN DEBOESER[1], THOMAS BRAUD[1], ALOIS KNOLL[1,2], DAVID ECKHOFF[1,2], (Member, IEEE), AND ALBERTO SANGIOVANNI-VINCENTELLI[3]

[1]TUMCREATE, Singapore 138602
[2]Department of Informatics, Chair of Robotics, Technische Universität München, 80333 München, Germany
[3]Electrical Engineering and Computer Science Department, University of California at Berkeley, Berkeley, CA 94720, USA

Corresponding author: Jordan Ivanchev (jordan.ivanchev@tum-create.edu.sg)

This work was supported by Singapore National Research Foundation through its Campus for Research Excellence And Technological Enterprise (CREATE) Programme.

ABSTRACT This paper introduces the a framework that simplifies the process of designing and describing autonomous vehicle platooning manoeuvres which implements four design principles: Standardisation, Encapsulation, Abstraction, and Decoupling (SEAD). Although a large body of research has been formulating platooning manoeuvres, it is still challenging to design, describe, read, and understand them. This difficulty largely arises from missing formalisation. To fill this gap, we analysed existing ways of describing manoeuvres, derived the causes of difficulty, and designed a framework that simplifies the manoeuvre design process. Alongside, a Manoeuvre Design Language was developed to structurally describe manoeuvres in a machine-readable format. Unlike state-of-the-art manoeuvre descriptions that require one state machine for every participating vehicle, the SEAD framework allows describing any manoeuvre from the single perspective of the platoon leader. We hope that the SEAD framework will pave the way for further research in the area of new manoeuvre design and optimisation by largely simplifying and unifying platooning manoeuvre representation.

INDEX TERMS Autonomous vehicles, platoon manoeuvres, AHS, mixed traffic simulation.

I. INTRODUCTION

Multiple challenges must be solved along the way to a globally optimised and highly automated urban traffic system. While the past has seen a large body of research and technological innovation on Autonomous Vehicles (AVs), platooning as a concept is still in its early stages. With the capabilities of vehicular awareness and communication in place, research can now start building elaborate platooning strategies to leverage on those technological advancements.

A specialised body of research is concerned with formulation of collaborative driving manoeuvres whereas research is split into two major categories. The first category optimises driving behaviour on the vehicle's dynamics level for smooth and energy-efficient driving. The second category, in contrast, centres around collaboration and communication strategies, for instance, how an additional vehicle joins an existing platoon. Within the second category, researchers have investigated specific cases of collaborative driving in platoons. Most papers focus on the specific details of one manoeuvre and optimise it to a great extent in terms of stability (the trait of a manoeuvre not to lead to dangerous traffic situations) or execution time.

Although most papers draw on the description of manoeuvres through finite state machines, there is no consensus or convention among researchers how to represent manoeuvres. This heterogeneity aggravates the comparison of manoeuvres and requires researches to constantly adapt to new conventions. Besides, the description of manoeuvres through mere state machines requires multiple synchronised yet independent state machines, one for each participating vehicle. Designing and reading these state machines can become challenging even for simple manoeuvres. The objective of this work is to provide a framework that simplifies and formalises this description.

The proposed framework follows four principles: Standardisation, Encapsulation, Abstraction, and Decoupling (SEAD). Standardisation ensures a common terminology

The associate editor coordinating the review of this manuscript and approving it for publication was Jesus Felez.

among all areas within the framework and all researchers applying it. To take advantage of the repetitive occurrence of action-sequences in various contexts, the Encapsulation principle allows grouping such repetitive patterns into re-usable modules. Since some patterns may contain sub-patterns, the Abstraction principle leads to recursive encapsulation which allows considering manoeuvres and their building blocks on different levels of detail. The Decoupling principle, finally, separates the control of the manoeuvre from its execution. This allows describing manoeuvres exclusively from the control-perspective of the platoon leader while the framework ensures the correct execution behaviour. The SEAD framework may serve future research as a point of reference and tool to facilitate further manoeuvre investigations. Solutions that implement one of the four principles exist. For example, the openSCENARIO standard certainly offers a form of standartisation, however, it does not provide the benefits of the remaining three principles, namely encapsulation, abstraction, and most critically decoupling.

In summary, the contributions of this article are:

- We survey and structure the complex landscape of AV manoeuvre research, identify shortcomings, and derive requirements for a new framework.
- We present SEAD, a novel AV manoeuvre framework to significantly simplify the process of manoeuvre modelling.
- We make a library of manoeuvres publicly available, in both human-readable and machine-readable formats.

The remainder of this article is organised as follows: First we give an extensive summary of related work in the domain of AV manoeuvre modeling (Section II). From this we derive in Section III requirements and principles that a common framework should fulfill and follow. Section IV-E introduces our new SEAD framework. Finally, in Section V we discuss possible extensions and conclude the article.

II. RELATED WORK

The concept of an Autonomous Highway Systems (AHS) was first introduced by Varaiya in 1993 under the name Intelligent Vehicle/Highway System (IVHS) [1], promising increases in safety and in highway capacity without the need of building new roads. These advantages emerge from two conceptual changes as compared to a conventional highway: 1) the application of vehicle platooning and 2) a global optimisation of traffic flow and travel times through a layered control structure.

A platoon is composed of a leader and at least one follower whereas, in most publications, the leader is the first vehicle in the upstream direction. The major impact of platooning for an increased highway capacity is the decreased inter-vehicle distance within platoons (intra-platoon distance d) as compared to the inter-vehicle distance outside of platoons (inter-platoon distance D) [1]. As shown in [1], platooning may increase road capacity by up to a factor of three.

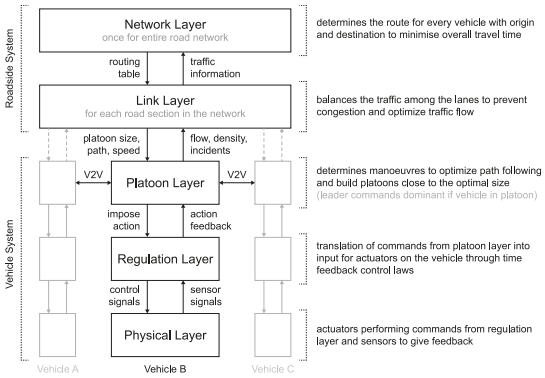


FIGURE 1. Control hierarchies of the platooning control system (summarized from [1]–[3]).

In the platooning logic, the leader sets the platoon speed and coordinates the manoeuvres performed by the platoon, for example splitting a platoon into two platoons (Split manoeuvre) or merging two platoons into one (Merge manoeuvre). The followers follow the preceding vehicle according to their Cooperative Adaptive Cruise Control (CACC) system and react upon commands issued by their platoon leader.

Although platooning itself increases road capacity and vehicle throughput, it may become more effective when applying additional higher-order strategies that leverage on the vehicle's communication capabilities. The interconnection of vehicles and infrastructure allows to control optimal platoon forming, driving, and splitting strategies and to optimise traffic globally instead of locally. The combination of such systems yields an AHS. Due to its complexity, the structure of the AHS is split into five hierarchical layers [1], [2]. Each layer fulfills a mutually exclusive task as shown in Figure 1.

At the highest level of the hierarchy, the Network Layer is responsible for optimizing the overall travel time of all vehicles and the traffic flow in the entire network [2]. It is aware of all autonomous vehicles in the road network and the current and predictable traffic situation on every road [1], [2]. To optimize travel time and traffic flow, it balances the traffic load on each road by determining the ideal path for each vehicle that travels from a defined origin to a defined destination [1], [4]. For the case of IVHS, the network only consists of the highway. The Link Layer is more decentralized and implemented by a controller for each road segment [1]. The controller ensures a smooth traffic flow on its road segment by distributing the traffic vehicles among lanes [2]. Besides determining a dedicated lane for each vehicle or platoon, it also determines the target size and velocity for platoons on that section [3]. For an IVHS, the road segmentation is realized by dividing the highway into segments of equal length. These two layers are implemented in the infrastructure and part of the roadside system. The layers below belong to the vehicle system, meaning that every vehicle is equipped

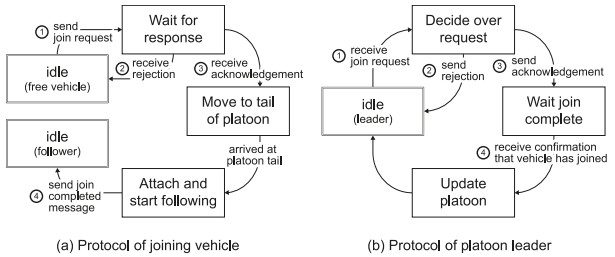


FIGURE 2. Example description of the JoinTail protocol as communicating FSM (simplification of protocol from [3]). (a) behaviour of the joining vehicle, (b) behaviour of the platoon leader. Rectangles constitute the states and the arrows the transitions. States with double-stroke are Idle States (i.e. starting and ending states of a protocol) and the ones with a single stroke are action states (i.e. states where an action is performed).

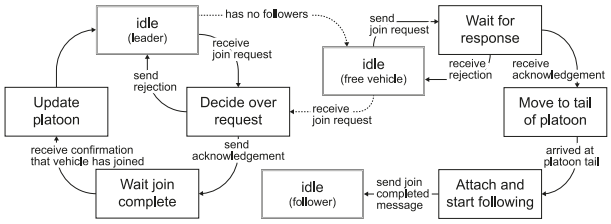


FIGURE 3. Combined FSM that represents the JoinTail-protocol as shown in Figure 9. The dashed arrows establish the connection between the two state machines.

with modules to realize the tasks of the three layers described below.

At the highest hierarchical level of the vehicle system is the Platoon Layer or Planning and Coordination-Layer. As a free vehicle, this layer determines actions to fulfill the path and lane directives imposed by the layers above [3]. Part of this task is to determine lane changes, if a vehicle should join or leave a platoon [2]. As a platoon leader, this layer coordinates the actions with vehicles that are associated with the platoon either as followers or potential joiners [1]. As a platoon follower, the platoon layer collaboratively performs action protocols which are initiated by the platoon leader [1], [3].

The Regulation Layer and the Physical Layer are responsible for realising the trajectories computed by the hierarchically higher layers. Control loops in the Regulation Layer compute actionable commands for the actuators and minimise the errors reported by the sensors in the Physical Layer [3].

The Platooning Layer, which is the focus of this publication, contains collaborative driving logic: Platoon Manoeuvres are collaborative vehicle actions that deal with platoon formation, maintenance, and modification. Manoeuvres encompass the driving and communication behaviour for all participating vehicles in the form of manoeuvre protocols. As shown in the subsequent sections, the description of manoeuvres is complex even for simple ones. Our work, thus, aims to increase the comprehensibility and facilitate the easy formulation of platoon manoeuvres. For readers which are more familiar with the PEGASUS 6-layer model, our work covers, partially, layer 4 of that model.

A common way of describing these manoeuvre-protocols are communicating finite state machines e.g. [5]–[7]. In order to gain a better understanding of different manoeuvre description approaches we present a simplified example of the JOIN TAIL-manoeuvre from [3] shown in Figure 2. The JOIN TAIL-protocol describes the process of a free vehicle joining an existing platoon at its tail through two CFSM.

When a vehicle (Vehicle A) decides to join a platoon, it sends a join request to the leader (Vehicle B) of the platoon (Transition 1, T1). B either rejects (T2) or acknowledges (T3) the request. In the case of rejection, both vehicles return to idle and the protocol terminates. If the request is acknowledged, A will wait for B to join the platoon. A moves to the tail of the platoon. Once arrived at the tail, A attaches to the platoon, starts following the preceding vehicle (i.e. switch into CACC mode), sends a message to B that the join is completed (T4), and transitions into the follower-Idle State. Upon receipt of the join-completion message, B updates the platoon information and returns to idle. The protocol execution is complete and terminates.

This description shows that state machines synchronise through messages that are sent between the vehicles. Since both vehicles operate through the same set of protocols, they can expect the other vehicle to behave synchronously.

The representation in Figure 2 describes the behaviour as two separate FSM, each describing the protocol for one role within the manoeuvre. FSM (a) shows the transition of a vehicle from one Idle State into another. After extending the state machine by the transition from a free vehicle into a platoon leader through platoon formation, the entire behaviour could be explained in one role-agnostic FSM. This FSM operates on both vehicles independently and simultaneously. Both participating vehicles then operate following the same FSM schema, yet they are in different states during the manoeuvre and follow different paths through the FSM.

This approach was shown by [5] and is illustrated in Figure 3 as an extension to the example in Figure 2. The dashed arrows were added to combine the two FSM into one.

In the same manner as in Figure 3, it is possible to describe the entire behaviour of vehicles in one role- and manoeuvre-agnostic FSM, meaning that one FSM would describe the entire platooning behaviour of a vehicle for all roles.

Although FSM are the dominant way to describe manoeuvres in the current state of literature, researchers also applied other ways of illustration. The authors of [3] describe a manoeuvre in a role-agnostic process flow-chart. From this chart, they deduct role-specific state machines that are ultimately formulated in the COSPAN (coordination-specification-analysis) system. COSPAN formalises CFSM to prove certain mathematical state machine properties such as completeness or reachability [8].

In [9], the logical flow of manoeuvres is also described as a role-agnostic flow-chart. Focusing on V2V communication, however, this paper also provides additional information

flow-charts to describe the communication between the participating vehicles. In addition to illustrations, the majority of papers describes the logical flow of events in manoeuvres through textual descriptions e.g. [3], [5], [7].

As mentioned above, most publications describe manoeuvres as FSM whereas only a minority applies other visualization and description principles. However, although most manoeuvre descriptions use the same methodologies, there is still a vast heterogeneity among all existing publications. Two dominant sources of this heterogeneity are: 1) different levels of detail, and 2) verbally differing descriptions to represent identical actions.

A. DIFFERENT LEVELS OF DETAIL

Some manoeuvre descriptions show a level of detail that reaches the regulation layer with states such as Accelerate to Merge [3] or Set Speed to 30 m/s [5]. Other descriptions, or sometimes even the same description, stay on a very high level within the platooning layer with states such as Car Splits [7] or Move To Position [10]. Due to these differences of abstraction level, FSM that potentially describe the same behaviour are, in fact, significantly different.

B. DIFFERING VERBAL DESCRIPTIONS

Even if illustrations describe a manoeuvre on similar levels of detail, they may use different terminologies or different levels of verbal abstraction for the same action. For example, the Join Tail-protocol was described in [10] and in [3] in similar logical ways. In both descriptions for the joining vehicle, a dedicated state equalises the speed of the platoon and the joining vehicle by decelerating or accelerating. While [10] describes this as Set Speed to 30 m/s and Catch-up and merge the platoon from the back, [3] expresses the same process as Accelerate to Merge. This poses a challenge when formalising and comparing manoeuvres across different publications.

Besides these heterogeneities, we identified two additional factors that pose challenges when formulating manoeuvres. First, repetitiveness introduces pseudo-complexity, especially for those descriptions with high levels of conceptual detail. For instance, the process for a vehicle moving to a certain position requires both communication and physical action. The leader orders the joining vehicle to move to a defined position. When at that position, the joining vehicle informs the leader about the arrival. These action-communication-patterns may occur repetitively within a manoeuvre FSM.

Second, since manoeuvres are mostly performed by two or more vehicles collaboratively, the behaviour for every participant is described in dedicated, role-dependent FSM. Reading these coupled state machines requires a substantial effort as the reader must manually synchronize the state machines to understand the collaborative process.

Another very important aspect of platooning is communication. The Join Tail-protocol description in Figure 3 shows the necessity for message transfers between participating vehicles. V2V standards, however, serve only collaborative awareness and are not sufficient for this case of collaborative driving. Researchers have developed various ways of formalising this communication while complying with the existing standards such as C-V2X, ITS G5, DSRC. The authors of [32] propose two protocols: A Minimal Protocol and a Full Platooning Protocol. While the Full Platooning Protocol is equipped for secured two-way communication, the Minimal Protocol is only suitable for one-way messages.

The Minimal Protocol can be used for collaborative awareness as well as manual platooning, meaning that a driver must manually initiate the joining or leaving of a platoon. Within the platooning protocol, the authors differentiate among three types of messages: Service Announcements, Service Requests, and Control Data Messages. Service announcements are sent by platoon leaders to advertise services, for instance, the availability to add another vehicle to the platoon. Service requests are, for instance, sent by free vehicles to platoon leaders to express the willingness to join. Control Data Messages are sent periodically by platooning vehicles to maintain and update the integrity of the platoon.

In [5], the WAVE Short Message Protocol (WSMP) [33] is applied to carry information on the control channel. The messages are of two types. The first type are beacons. These beacons, like cooperative awareness messages, carry information about the vehicle state, e.g. the position, acceleration, and lane. Additionally, the beacon carries the platoon ID and the current vehicle position in the platoon if the vehicle is platooning. The position starts with 0 at the platoon leader and increments with each vehicle in the platoon counting upstream. The second type is micro-commands. These are used to initiate and control platoon manoeuvres. While beacons are usually broadcasted, most micro-command messages are unicasted or, if the manoeuvre involves multiple vehicles, multicasted. The authors provide a set of seventeen micro-commands to enable a multitude of manoeuvres. To give the reader a more complete overview Table 1 summarizes the research efforts involving studying platoon manoeuvres that were considered for this paper.

III. REQUIREMENTS AND PRINCIPLES

To relieve the shortcomings described in the previous section, namely the challenges with comprehensibility, compilation and comparison of manoeuvre-descriptions, we propose a framework for the universal description of manoeuvres that builds on the four principles of Standardisation, Encapsulation, Abstraction, and Decoupling (SEAD). This framework will enable researchers and engineers to easily define new or alternative manoeuvres within the platooning layer shown in Figure 1. First, we will derive requirements for the framework design from the identified limitations, then we will define goals that the framework should fulfil, and postulate the overarching design principles to achieve the design goals. Figure 4 provides an overview of the interrelations among deficits, goals, and principles.

The SEAD design framework aims to resolve the major shortcomings that state-of-the-art manoeuvre descriptions face. The subsequent list summarises the identified deficits

TABLE 1. Research publications regarding platooning manoeuvres.

Reference	Manoeuvres	Additional Information	Cat*
Lu & Hedrick 2003 [11], Lu et al. 2004 [12]	Merge	Design of controller for longitudinal merging, virtual platoon to pre-compute trajectory before manoeuvre	P
Nowakowski et al. 2016 [13]	Merge, Split	Platooning for non-autonomous trucks with CACC technology and V2V communication, manoeuvres performed by drivers	P
Segata et al. 2014 [10]	JoinMiddle (FSM)	Designing and investigating the JoinMiddle manoeuvre with various interference scenarios and communication packet loss	P
Hsu & Liu 2004 [14]	LaneChange	Proposes manoeuvre to LaneChangeAndFollow for higher efficiency	P, R
Hsu & Liu 2008 [15]	LaneChange	Investigates LaneChange options: simultaneously and time delay with varying inter-platoon spacing	P, R
Kazerooni & Ploeg 2015 [16]	Join, LaneReduction, GapOpen, GapClose	Design of interaction protocols for LaneReduction and JoinMiddle, Analysis of velocity profiles	P, R
Rajamani et al. 2000 [7]	Join, Leave, LaneChange (FSM)	Lateral and longitudinal control for Merge and Join, and LaneChange	P, R
Best 2012 [17]	-	Rapid high-speed lane change for obstacle avoidance, proposal of open-loop controller for steering instead of emergency braking	R
Choi & Swaroop 2000 [18]	-	Assessing coordinated emergency braking in platoons	R
Frankel et al. 1996 [19]	Merge, Split, LaneChange	Proposal of safety-ensuring controllers for Merge, Split, and LaneChange as FSM, textual description of manoeuvres	R
Godbole et al. 1995 [2]	Join, Leave (textual)	On- and off-ramp in AHS with dedicated lane for AVs	R
Goli & Eskandarian 2014 [20]	Merge, LaneChange	Multi-merge manoeuvre (multiple vehicles at one), lateral trajectory generation and execution via PID	R
Huang & Chen 2001 [21]	Merge, Split	Investigates the safety of Merge and Split for emergency braking into different stages of the manoeuvres	R
Kato et al. 2002 [22]	Merge, Leave, LaneReduction	Investigation of velocity profiles for merging, obstacle avoidance, leaving, and Stop&Go	R
Li et al. 1997 [23]	-	Longitudinal control laws with focus on safety that can be adjusted through parameters for changing external conditions	R
Murthy & Masrur 2016 [24], Murthy & Masrur 2017 [25]	Emergency Brake	Coordinated emergency braking in platoons considering the weakest vehicle in the platoon	R
Naranjo et al. 2008 [26]	LaneChange	Applying fuzzy controller lateral and longitudinal control in LaneChange manoeuvre for overtaking	R
Rai et al. 2015 [27]	Merge, LaneChange, Overtake	Concept of virtual leader to command synchronised lane changes, collision avoidance through potential-field controller	R
Sun et al. 2003 [28]	LaneChange, PlatoonChange, GapOpen, GapClose	Manoeuvre to directly switch from one platoon to other platoon on adjacent lane, controller design for gap making and closing	R
Swaroop & Yoon 1999 [29]	Emergency Brake and Emergency LaneChange	Controller that couples braking and lane changing in emergency situations through V2V communication for higher safety	R
Usman & Kunwar 2009 [30]	Overtake	Methodology for online generation of driving trajectories for overtaking manoeuvres	R
Wang et al. 2017 [31]	JoinMiddle, GapOpen, GapClose	Trajectories for JoinMiddle for minimizing carbon emissions	R

* Focus of the paper, C = Communication, R = Regulation Layer, P = Platooning Layer

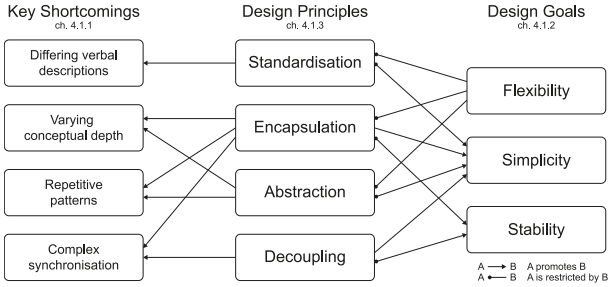


FIGURE 4. Relations of key deficits, design principles, and design goals for creating the SEAD framework.

and defines the scope of problems the proposed framework should solve.

- Varying conceptual depth: Varying levels of conceptual depth within and among manoeuvre descriptions require to re-frame the manoeuvres before comparison.
- Differing verbal descriptions: Differing verbal descriptions for equivalent conceptual components require the manual alignment of terms before comparison.
- Repetitive patterns: Repetitive action-communication-patterns within and among manoeuvres make manoeuvres more complex to read and tedious to be described.
- Complex synchronisation: The complex synchronisation between multiple role-specific FSM in one manoeuvre is challenging. This makes it hard to understand the collaborative aspect of a manoeuvre. Besides, it poses the threat of unstable states due to unforeseen circumstances.

For the SEAD framework to achieve practicality while resolving these deficits, we defined three design goals. The goals outline how we define practicality for the framework and give guidance in making all design decisions. The three qualitative goals are Flexibility, Simplicity, and Stability.

- Flexibility: The framework must be capable of describing all manoeuvres in the current state of literature. Any design decision, thus, must ensure that the flexibility of describing manoeuvres is maintained and that restrictions are minimised.
- Simplicity: The prime use case of the framework is reading and formulating new or alternative manoeuvres. Any design decision, thus, must promote the simplicity of reading and formulating manoeuvres.
- Stability: The stability of manoeuvre designs is crucial, meaning that the design shall not allow for manoeuvres that end in an undefined state if unexpected driving situations occur. Any design decision, thus, must foster the stability of manoeuvres for any possible interruption or exception.

Given the postulated requirements, we derived four major design principles that help to resolve the current shortcomings and to fulfil the design goals. These can be summarised under the four terms Standardisation, Encapsulation, Abstraction, and Decoupling. The subsequent sections explain these principles in greater detail and interrelate them with the established design goals and identified deficits.

A. STANDARDISATION

As stated above, one prime issue in defining manoeuvres is the vastly varying terminology for equal conceptual components. These components can be states within an FSM, V2V platooning messages or inherent reasons for state transitions such as autonomous decisions. Our framework resolves the lack of standardisation by introducing a finite set of symbols for these conceptual components. More specifically, the SEAD framework provides a syntax for describing vehicle actions, messages, and internal state transition triggers. The main requirement for this standardisation is the universal comprehensibility regardless of context or background.

The principle of Standardisation will mainly be affected by the design goals Flexibility and Simplicity. Although Standardisation will introduce syntactic rule sets, the standardised framework must be capable of expressing a sufficient conceptual depth not to obstruct the Flexibility goal. Besides, the Standardisation should also promote Simplicity, meaning that the standardised terms must be intuitively comprehensible and easy to grasp without the need for extensive preparation. The aim is that a person with no prior experience with the SEAD framework must be able to understand the manoeuvres formulated using it.

B. ENCAPSULATION

Throughout manoeuvre descriptions, we identified various repetitive patterns that include vehicle actions and inter-vehicle communication. The principle of Encapsulation will help pack these patterns into reusable blocks whereas one block may contain autonomous activities of one vehicle or synchronised actions of multiple vehicles. Moreover, these sub-manoeuvres will mainly comprise actions and states of equal conceptual depth. This helps in resolving the problems of varying conceptual depth, repetitive patterns, and complex synchronisation.

The Encapsulation will be influenced by all three design goals. Following the design goal of Flexibility, the Encapsulation of intertwined action-communication patterns must not impose manoeuvre-design restrictions. Thus, sub-manoeuvres on the lowest level must express the most fundamental patterns exhaustively. Thereby, Encapsulation will greatly promote Simplicity as it will allow reusing behavioural patterns without the need for redesigning them. This requires equipping sub-manoeuvres with a suitable interface that allows for integrating them into higher-order structures. Lastly, to achieve Stability, the sub-manoeuvres must be designed to always lead to predictable outcomes that can be handled by any context.

C. ABSTRACTION

Extending Encapsulation, the principle of Abstraction will allow to reuse the encapsulated sub-manoeuvres within structures of conceptually higher levels and to encapsulate these higher order structures into reusable blocks. This recursive re-usage allows for arbitrarily complex structures whereas every level of conceptual depth can be designed separately without the need to operate on different levels at once. Abstraction helps to resolve the deficits of varying conceptual depth and handling repetitive patterns.

As every system that re-frames complexity through hierarchical depth, our framework imposes certain restrictions. Nevertheless, it shall fulfil the design goal of Flexibility, meaning that every abstraction will be carefully evaluated. On the other hand, the Abstraction principle fosters Simplicity as the major part of the manoeuvre design process takes place at the higher levels of abstraction. This allows building highly complex manoeuvres with relatively low effort.

D. DECOUPLING

Lastly, the Decoupling principle will allow describing complex manoeuvres entailing two or more vehicles from the perspective of the platoon leader. The SEAD framework, and specifically the Encapsulation principle, ensures that this single-sided description suffices to describe the behaviour of all vehicles. More specifically, the utilized primary-secondary structure leads to a general universal description of the reactive behaviour of the slave (platoon followers or free vehicles) while manoeuvres are only defined and steered by the master (platoon leader). This principle, in consequence, resolves the need for synchronising multiple manoeuvre descriptions.

Decoupling will considerably facilitate the process of reading and describing manoeuvres as both actions must only be performed from the leader perspective. This greatly drives the goal of Simplicity. However, designing the reactive behaviour of the slave-vehicles must be very comprehensive and elaborate in order to fulfil the design goal of Flexibility.

IV. FRAMEWORK BUILDING BLOCKS

In this section, we will give a detailed description of the building blocks that make up the SEAD framework. We present the building blocks and concepts in a bottom-up fashion, starting with the lowest level: action primitives, followed by sub-manoeuvres, manoeuvres, and maneuver wrapping for simulataneous execution. After that, we cover the overarching concepts of states, the universal reactive state machine, and the proactive maneuvering engine. Before we present each building block, we will describe a few properties of the system to facilitate its overall comprehension:

Scope: The described platooning framework does not include high level decision-making processes such as when to form a platoon or when to leave a platoon. This is the task of the Link Layer Interface (LLI) and could either be computed inside an autonomous vehicle or be given in

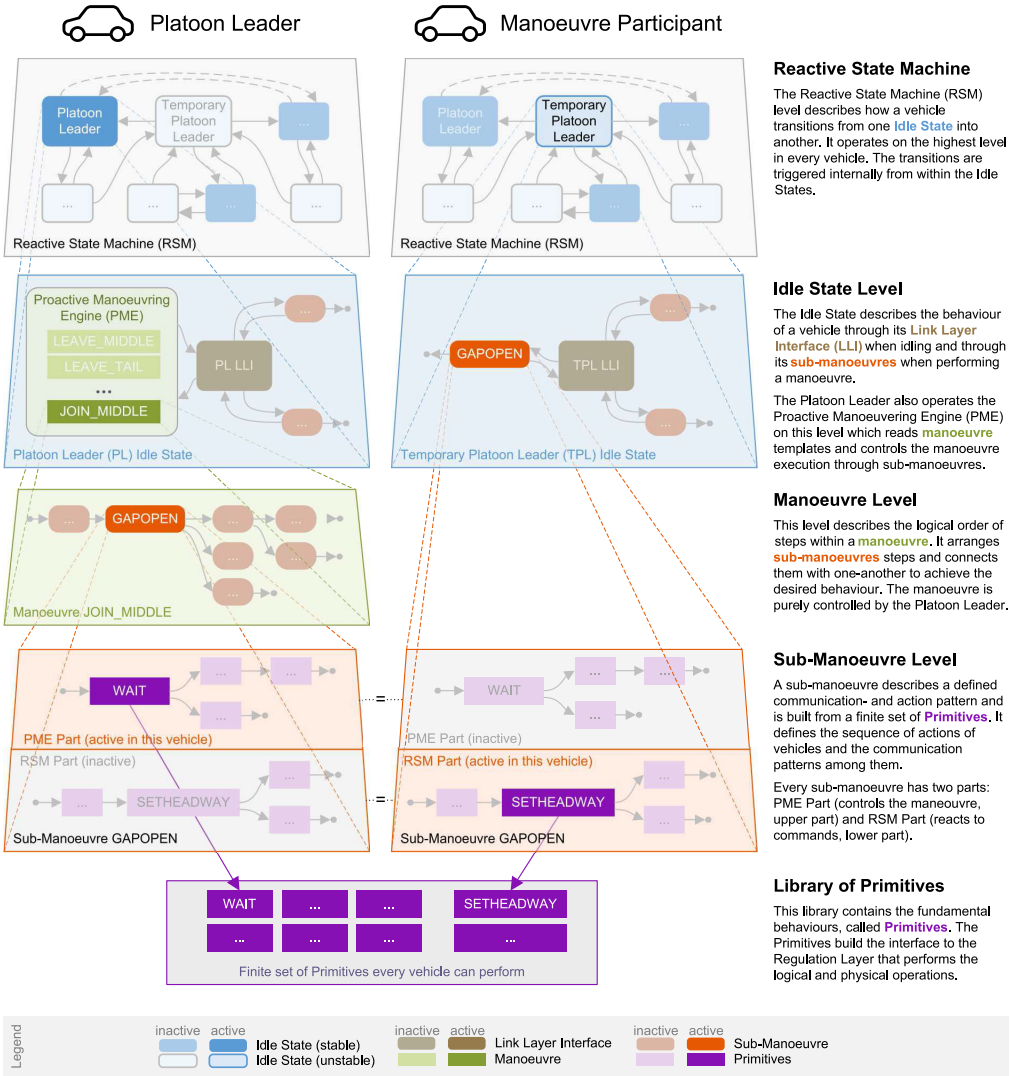


FIGURE 5. Overview of the hierarchical structure of the SEAD framework. The figure provides an example where two vehicles, a platoon leader (left) and a temporary leader (right), perform a collaborative manoeuvre.

the form of commands issued by another controlling entity. The framework at hand will only cover the platooning layer itself.

Stability: To ensure stability of the system, each platooning manoeuvre has to end in a stable state for all involved vehicles, regardless of manoeuvre's success. We refer to these states as "stable idle states" in contrast to "unstable idle states" which are states during a manoeuvre when a vehicle is waiting for an action of another vehicle. These stable idle states can be Platoon Leader (PL), Platoon Follower (PF), or Free Vehicle (FV). When a vehicle is waiting for an instruction or action from another vehicle while performing a manoeuvre their respective unstable idle state would be WPL, WPF, and WFV. Additionally, we make use of the unstable idle state Temporary Platoon Leader (TPL) to increase stability [10] when manoeuvres are aborted. Only when a vehicle is in a stable idle state (i.e., not currently performing a manoeuvre) is it available to receive commands from the LLI.

Control: For every manoeuvre, we assume it to be carried out in a primary-secondary fashion, that is, one vehicle (naturally, the platoon leader) issues orders to the other vehicle(s). This does, however, not mean that manoeuvres can only be initiated by the platoon leader, it merely describes the control flow once the manoeuvre has started.

Communication We assume the existence of an underlying communication system that provides message primitives such as described by [5]. These messages include Requests (REQ), orders (ORD), done-confirmation (DN), abort (ABT), and acceptance/rejection (ACK/NACK). We abstract away from modelling the physical transmission of messages and assume perfect communication. To implement the concept of a Temporary Platoon Leader (TPL), an additional specialised message type TMPL SPLIT forces the split of a TPL and aborts the manoeuvre that is currently being processed. REQ, ORD and DN messages can include additional information as required by the system, e.g., the size of the gap a vehicle is ordered to open. We further assume

that the underlying protocols can ensure that the successful transmission of messages. If this cannot be ensured, then the sub-manoeuvres can be extended by adding respective time-out and abort transitions.

A. FRAMEWORK OVERVIEW

Figure 5 provides an overview of the SEAD framework and illustrates its hierarchical structure according to the paradigm of hierarchical state machines. The entire framework could be expressed as one big state machine, however, the transitions and states are designed in a way to enable all four SEAD principles (Standardisation, Encapsulation, Abstraction, and Decoupling). The different layers can therefore be seen as different zoom levels or views of the platooning behaviour definition for autonomous vehicles. In the following sections, we will introduce and explain the framework in a bottom-up fashion.

B. ACTION PRIMITIVES

The lowest layer of the framework is composed of action primitives which are actions that directly affect the physical or logical state of a vehicle. We differentiate between physical primitives, state primitives, and other primitives. The physical primitives are the direct interface to the regulation layer and affect the physical state of a vehicle, i.e, the vehicle's position or speed. State primitives affect the role of the vehicle, i.e. whether it acts as a platoon leader or platoon follower. They are needed to transition a vehicle from and to different idle states (e.g., at the end of a manoeuvre or when the vehicle needs to wait for instructions from another vehicle). Lastly, other primitives are needed to either send messages, update platoon information, or wait for certain events. Primitives should be designed in an orthogonal fashion, meaning that one primitive cannot be expressed by a combination of any other primitives. Table 2 provides an overview of action primitives that could be found in the state-of-the-art manoeuvre descriptions.

As the SEAD framework describes all actions on the platooning layer, the physical primitives provide directions to the low-level control of the vehicle instead of controlling the longitudinal dynamics directly. For instance, instead of setting the desired speed or acceleration of the vehicle, the primitives set the desired location. The path-following logic on the Regulation Layer will convert the target location into actionable commands (accelerate, decelerate, steer left / right) for the physical layer.

C. FORMULATING SUB-MANOEUVRES

With the list of idle states and action primitives, we can now create sub-manoeuvres. A sub-manoeuvre encapsulates reusable behavioural patterns that involve two or more vehicles and transitions at least one of the participating vehicles from one idle state into another. For each participating vehicle, the behaviour is described through a sequence of primitives which constitute a sub-state machine. Sub-manoeuvres must design action-communication patterns with the smallest reasonable scope to promote re-usability and therefore achieve the goal of Flexibility.

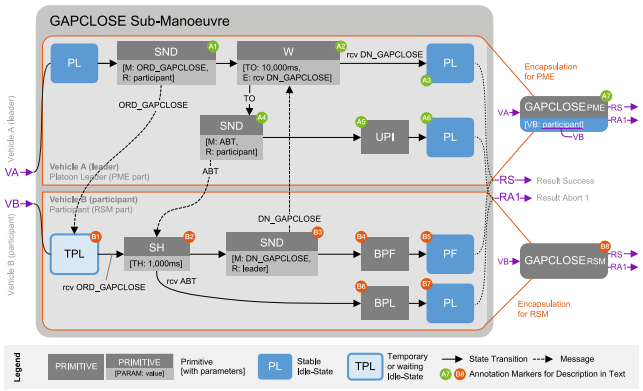


FIGURE 6. Sub-manoeuvre description and encapsulation for GAPCLOSE.

To implement the principle of Decoupling, each sub-manoeuvre is split into two or more sub-state machines, one for each vehicle participating. As SEAD follows the primary-secondary paradigm, one sub-state machine controls the sub-manoeuvre (executed by the platoon leader), the other sub-state machines purely react. The sub-state machines are connected and synchronised through V2V messages. This structure promotes the goal of Simplicity through the principles of Encapsulation and Abstraction: Any sub-manoeuvre can be reused without a thorough understanding of the inner workings once its behaviours and outcomes are defined. We will discuss this separation and decoupling extensively in the Reactive State Machine (RSM) and the Proactive Manoeuvring Engine (PME) sections. The controlling sub-manoeuvre is denoted with PME while the reactive ones are marked with RSM (as in Figure 5).

To fulfil the goal of Stability, sub-manoeuvres must be designed such that any possible scenario (success or abort) leads to a defined outcome for every participating vehicle. Therefore, if one vehicle encounters a situation that will prevent the successful completion of the sub-manoeuvre and causes an abort-result, V2V communication (or time-outs) must cause all other sub-state machines to terminate at the same abort-result. Due to a shared understanding of the sub-manoeuvre among all vehicles, every vehicle is informed about the final state of all participants.

Figure 6 illustrates an example sub-manoeuvre where a platoon leader orders another vehicle in the platoon to close the gap. For stability reasons, the vehicle that closes the gap is a temporary platoon leader, so that when the sub-manoeuvre fails, it will be the platoon leader of all other vehicles behind. The sub-manoeuvre includes two participants, the platoon leader (PL, vehicle A) and the temporary platoon leader (TPL, vehicle B). The sub-manoeuvre can conclude in either Success (RS) or Abort 1 (RA1).

A initiates the sub-manoeuvre through commanding B to close the gap by sending an ORD GAPCLOSE message (A1) via V2V communication. After sending the message, A waits for the completion (A2). The message triggers B to execute

TABLE 2. List of action primitives derived and condensed from the state of the art.

Acronym	Primitive	Parameters	Description	Actor
Physical Primitives				
MTP	Move to position	TR: Target object, RO: Relative offset	Move to a position defined by a relative offset to a target (vehicle) and align speed	FV, PL, TPL
SH	Set headway	TH: Time headway *or* SH: Space headway	Set the desired time or space ahead to preceding vehicle and adjust real distance	PF, TPL
State Primitives				
BFV	Become free vehicle		Transitions the vehicle into a Free Vehicle	All but FV
BPL	Become platoon leader		Transitions the vehicle into a Platoon Leader	All but PL
BPF	Become platoon follower		Transitions vehicle into a Platoon Follower	All but PF
BTL	Become temporary leader		Transitions the vehicle into a Temporary Leader	All but TPL
SW	Set Wait		Sets idling state to corresponding waiting state	FV, PL, PF
USW	Unset Wait		Sets waiting state to corresponding idling state	WFV, WPL, WPF
Other Primitives				
W	Wait	E: Event to wait for, TO: Timeout	Waits for an event to occur, a message, or for a timeout	All
SND	Send Message	M: Message, R: Receiver	Sends a message to the receiver	All
UPI	Update platoon information		Updates information about the platoon (number and order of followers etc.)	PL

*(W)FV: (Waiting) Free vehicle, (W)PF: (Waiting) Platoon Follower, (W)PL: (Waiting) Platoon Leader, TPL: Temporary Platoon Leader

the GAPCLOSE sub-manoeuvre (B1). B sets its headway to the requested intra-platoon distance and the regulation layer starts to decrease the distance until it reaches the desired headway (B2).

In the success scenario, the desired headway is reached, B signals the completion of closing the gap to the PL through sending a DN_GAPCLOSE (B3) and transitions into the stable PF Idle State (B4). This concludes the sub-manoeuvre for B with a success-result RS (B5). A receives the message and concludes the sub-manoeuvre with a success-result RS (A3).

In the abort scenario, if closing the gap is taking too long, a timeout in A aborts the sub-manoeuvre. The timeout causes A to send an ABT message to B (A4) and to update the platoon information (A5). Afterwards, the sub-manoeuvres concludes for A with an abort-result RA1 (A6). B receives the message and will initiate the sequence to split from the original platoon by transitioning into a PL (B6). Once B is a PL, the sub-manoeuvre also concludes for B with an abort-result A1 (B7).

The upper part (the pro-active part) and the lower part (the reactive part) of the sub-manoeuvre is encapsulated into two reusable blocks (A7 and B8, respectively). These buildings blocks can then be used to build manoeuvres for the Proactive Manoeuvring Engine (PME) and the Reactive State Machine (RSM). Due to the structure of the sub-manoeuvres, the initiation of a PME part will always leads to initiation of the RSM part as well. This principle allows defining manoeuvres from the perspective of the platoon leader without the need to describe the participant's behaviour. In the same fashion as in Figure 6, it is possible to define a comprehensive set of sub-manoeuvres that allows assembling complex manoeuvres with limited number of restrictions. We have created an online repository where we have made graphical depictions of manoeuvres and sub manoeuvres as well as their machine readable descriptions (see Section IV-I) publicly available.[1]

D. FORMULATING MANOEUVRES

To promote the design goal of Simplicity and due to the primary-secondary paradigm of SEAD, a manoeuvre only needs to be described from the perspective of the platoon leader. The behaviour of the other participants is defined in the sub-manoeuvres and the initiation of these sub-manoeuvres is done by the platoon leader via V2V messages. A sequential manoeuvre (see Section IV-E how to define simultaneous manoeuvres) is a chain of sub-manoeuvres, where the transition to the next sub-manoeuvre depends on the result of the previous one. Figure 7 shows the formulation of the JOIN TAIL manoeuvre as it was also described in Figure 2.

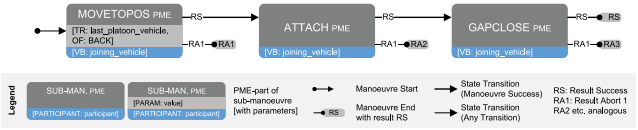


FIGURE 7. Description of the JOIN TAIL manoeuvre according to the PME logic.

In this illustration, the chain of sub-manoeuvres describes the entire manoeuvre. The blue lower part of the sub-manoeuvre boxes specifies the participating vehicles according to the notation in Table 2 whereas vehicle A (VA) is not specified since it is always part of the manoeuvre as the PL, and vehicle B (VB) is the participating vehicle. The JOIN TAIL manoeuvre requires no additional actions in case of an abort and will conclude in a stable state achieved through the abort-architecture within the sub-manoeuvres. More elaborate manoeuvres such as the JOIN MIDDLE require more sophisticated abort structures (e.g., a GAP CLOSE for a platoon follower if the joining vehicle was unable to lane change into the platoon). This is necessary to ensure that no vehicle will be in an unstable idle state once the manoeuvre is finished.

Defining manoeuvres in this fashion largely promotes Simplicity as a manoeuvre can be described from the leader's point of view while the reactive part of all sub manoeuvres (also referred to as the universal RSM) takes care of the participating vehicles' perspective.

E. SIMULTANEOUS MANOEUVRES

Manoeuvres with two or more participating vehicles can potentially benefit from the parallel execution of sub-manoeuvres. To facilitate this, the SEAD framework introduces a wrapper for the simultaneous execution, referred

[1]The library can be found at https://github.com/sead-framework/manoeuvre-catalogue

to as the SIM WRAPPER. This construct can involve an arbitrary number of simultaneous sub-manoeuvres whereas every sub-manoeuvres' controlling part (the upper part in Figure 6) is executed by the same leader and the reactive portion (the lower part in the same figure) is executed by the participants. Two simultaneous sub-manoeuvres, however, cannot involve the same participating vehicle.

To comply with the requirement that a state machine can only be in one state, the SIM WRAPPER can be understood as a product state machine of the controlling portions of all involved sub-manoeuvres. Since the reactive portion is executed in separate vehicles through the Decoupling principle, they occur in separated state machines in distinct systems. The execution result can be any element from the Cartesian product of all execution result sets from all sub-manoeuvres.

F. IDLE STATES AND SUPER-STATES

With the definition of idle states and sub-manoeuvres, it is possible to define them together to derive a state-machine on an abstraction level that clearly shows how the vehicle can transition from one idle state to another via which sub-manoeuvre. We combine the idle state and its associated sub-manoeuvres (i.e., the sub-manoeuvres that a vehicle can execute if it is in the idle state upon reception of a V2V message) into an idle super-state. This concept is shown in Figure 8, where the idle state WFV (Waiting Free Vehicle) and three sub-manoeuvres LC_BPF (Lane Change & Become Platoon Follower), MOVETOPOS (Move to Position), and ATTACH are all combined into a superstate. This superstate can only be left through the successful or unsuccessful execution of sub-manoeuvres or when a time-out occurs.

Every stable Idle State (FV, PF, PL) has an idling superstate. Within this superstate, the idle state will be referred to as a LLI (Link Layer Interface) idle state, because in these states, the vehicle can make (or receive) high-level decisions to carry out collaborative actions, for instance, joining or leaving a platoon, the decisions for which are made in the Link Layer.

When the vehicle is in an unstable idle state (i.e., WFV, WPF, WPL, TPL), thus in a manoeuvre, the LLI is inactive because manoeuvre initiation is only possible when the vehicle is not already performing a manoeuvre. Since this paper focuses on the Platoon Layer and the LLI models the Link Layer, the inner workings of the LLI will not be covered here.

G. REACTIVE STATE MACHINE (RSM)

The combination of all idle superstates into one big interconnected state-machine yields the so-called Reactive State Machine (RSM), which can be seen as a complete behaviour definition for all platooning vehicles except the platoon leader. As the platoon leader coordinates and controls the manoeuvres, this state-machine is called *reactive* as it reacts to what the leader is doing. Figure 9 shows the complete RSM for a platooning system which supports a number of basic sub manoeuvres. For better readability, we chose sub manoeuvres as the abstraction level in this illustration, however, each of

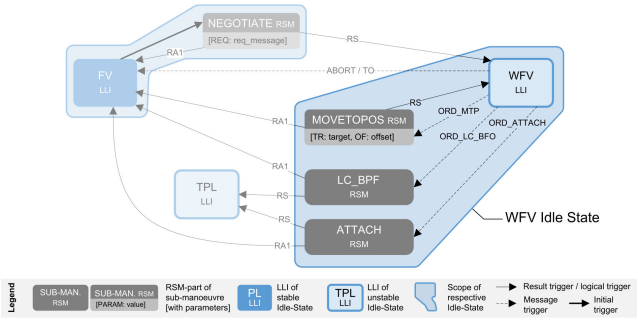


FIGURE 8. Definition of the WFV idle superstate. It contains the WFV LLI Idle state as well as the sub-manoeuvres that can be performed by the WFV to transition into another idle state.

the sub manoeuvre boxes could be replaced by the entire sub manoeuvre definition (e.g., Figure 6).

The RSM describes the universal reactive behaviour of any vehicle for any manoeuvre that can be built using sub-manoeuvres as building blocks. However, the RSM describes only the decoupled reactive behaviour (as indicated by the RSM in any sub-manoeuvre box). It does not implement how the manoeuvre is logically performed, i.e. the sequence of sub-manoeuvres comprising a manoeuvre. This implements the first half of the Decoupling principle. The second half of Decoupling, the formulation and control of manoeuvres, is implemented by a complementary structure that steers manoeuvres, namely the Proactive Manoeuvring Engine (PME).

The introduction of the RSM strongly promotes the goal of Stability. As can be seen in Figure 9, the RSM defines the behaviour in case of an abort for every sub-manoeuvre. This structure, thus, always brings the vehicle to a defined state in whichever way a manoeuvre terminates. According to the Abstraction principle, the RSM reuses sub-manoeuvres, which leads to all building blocks in the RSM being on an equivalent conceptual level. This directly mitigates the two key shortcomings of varying conceptual depth and repetitive patterns. Since the structure and design of the sub-manoeuvres is aligned with the goal of Flexibility, the RSM introduces no further restrictions regarding this concern.

H. PROACTIVE MANOEUVRING ENGINE (PME)

While both proactive and reactive behaviour definitions can be combined into one state-machine, we separate them for the sake of Simplicity and Decoupling. To this end, the Proactive Manoeuvring Engine (PME) complements the reactive behaviour of the RSM (as can be seen in the left bottom of Figure 9 they are indeed connected). The PME is therefore an extension to the PL's RSM behaviour and is responsible for the coordination of platooning manoeuvres.

Figure 10 shows the PL LLI with the connection to the RSM part (Figure 9) on the right side and the entire PME on the left side. The level of abstraction chosen in this figure is on the manoeuvre layer with the example at hand supporting

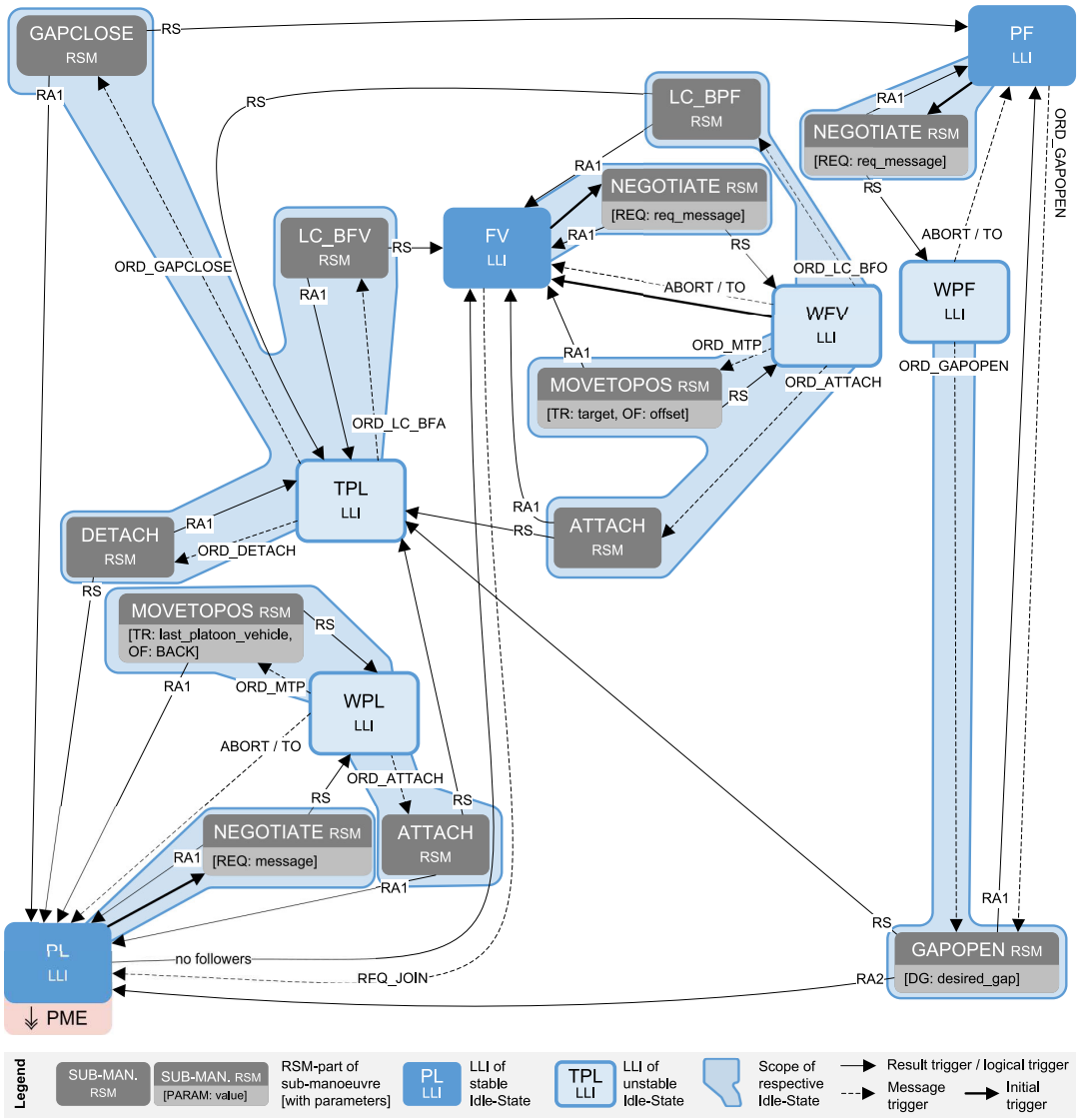


FIGURE 9. Reactive state machine (RSM). It describes the reactive behaviour by combining all sub-manoeuvres.

the JOIN, LEAVE and SPLIT manoeuvres. These manoeuvre boxes could be extended to their respective contained sub-manoeuvres (or even further to include the entire controlling part of each sub-manoeuvre), however, Abstraction and Decoupling allow us to illustrate the platooning system in a more comprehensible way. The PME combines the manoeuvre schemes of all manoeuvres and steers the manoeuvres from the perspective of the PL. Manoeuvres are initiated through the LLI of the PL either directly (direct init. in Figure 10) or through a request REQ that triggers a NEGOTIATE sub-manoeuvre. When adding a new manoeuvre, rules in the LLI must define when it will be called (i.e. which REQ message evokes a NEGOTIATE or which conditions triggers a direct init.) as illustrated in Ffigure 10.

Only manoeuvres that concern exclusively platoon followers can be instantiated directly by the PL. For instance, a PL cannot command an FV to join its platoon without a prior request of the FV to join. This idea imposes that joining is always initiated through an FV. However, by designing an additional sub-manoeuvre where the PL requests an FV to join, the SEAD framework can also adapt to this paradigm.

I. MANOEUVRE DESIGN LANGUAGE

Although the visual description of manoeuvres and sub-manoeuvres is easily comprehensible for humans, machines will not be able to process it. To allow flexibly redesigning manoeuvres and sub-manoeuvres, we have developed a Manoeuvre Design Language (MDL) that directly translates from and into a graphical representation. In the future, a graphical editor to create and export manoeuvres and sub-manoeuvres as JSON MDL files will help to easily design manoeuvres graphically and to directly feed them into simulation systems. The simulation system parses the MDL file and generates the code required for the

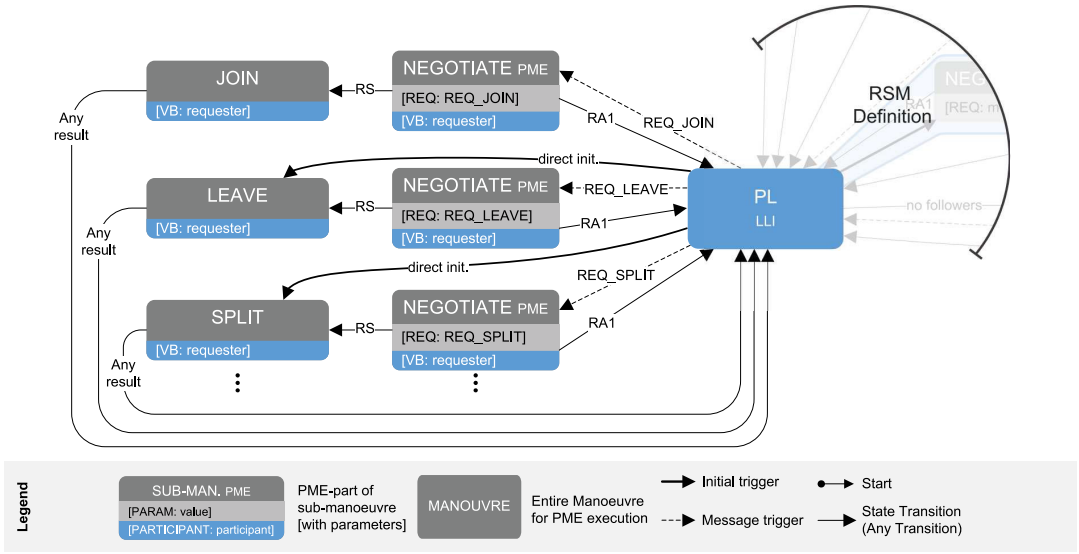


FIGURE 10. Proactive manoeuvring engine (PME). A manoeuvre can either be initiated through an acknowledged request through NEGOTIATE or directly through the LLI of the PL.

execution of manoeuvres according to the SEAD framework. The MDL is based on JSON and was inspired by the syntax and structure of the Amazon States Language. One JSON MDL file has a unique ID (or action ID) and describes a (sub-)manoeuvre following a fixed syntax. The syntax of the language is outside of the scope of this paper but can be accessed in [34].

V. CONCLUSION AND FUTURE WORK

This paper introduces the SEAD framework that simplifies the formulation of manoeuvres for vehicle platooning. It is based on the four principles of Standardisation, Encapsulation, Abstraction and Decoupling (SEAD). As previous research has shown [1], vehicle platooning has the potential to substantially increase road capacity and traffic throughput, providing a potential solution for traffic systems to adapt to the ever-increasing traffic demand. Although vehicle platooning is a promising concept, it remains challenging to define and describe collaborative manoeuvres. This poses a bottleneck in the development pace of the platooning concept and hampers its applicability in real-world scenarios.

The design of the SEAD framework was conceptualised to resolve four key shortcomings of the state-of-the-art description of manoeuvres using coupled state machines: First, as a manoeuvre description consists of one state machine per participant, the reader must synchronise the state machines to understand the manoeuvre. Second, the investigated schemas reveal varying conceptual depth within and among manoeuvre description as well as, third, differing verbal descriptions of equivalent components. This heterogeneity makes it difficult to understand and compare manoeuvres. Finally, various action-communication patterns surfaced in multiple manoeuvre descriptions, making them seem more complex than they are.

The SEAD framework is a first step to formalise the Platoon Layer of an Automated Highway System. It provides the means to conduct further research on the performance of manoeuvre variations, alternatives and platoon forming strategies. There are two main future research fields involving the proposed framework.

A. IMPROVING AND EXTENDING THE SEAD FRAMEWORK

Once the required simulation tools for platooning in urban environments are in place, further building blocks may be designed extending the SEAD framework. The framework will allow formulating complex urban manoeuvres such as LEAVE MIDDLE AND TURN LEFT (or LMTL), yet the further development of primitives, sub-manoeuvres, and communication patterns will be required.

To increase the adoptability and usability of the SEAD framework, future research could focus on developing a stand-alone application-agnostic tool that models the Platoon Layer according to the presented framework. However, since the Platoon Layer and the Regulation Layer are closely interconnected through the Action Primitives, an elaborate communication interface between these two layers needs to be developed to allow for modularisation.

Furthermore, although the proposed framework allows defining elaborate abort structures for manoeuvres and sub-manoeuvres, it does not propose an abort-and-retry structure. Once elaborate models are in place to evaluate if a second attempt could be successful, higher-order manoeuvres and re-modularisation of certain sequential parts of a manoeuvre provide the opportunity to implement such abort-and-retry structures.

B. EMPOWERING FURTHER STUDIES

As mentioned before, the biggest advantage of the SEAD framework is its capability of designing manoeuvre variants and alternatives through re-arranging the sub-manoeuvres once all building blocks are implemented and the RSM

defined all required state transitions. This allows for various dynamic investigations.

For instance, traffic simulators implementing the framework such as BEHAVE [35] could provide a means to identify the most efficient alternative of a manoeuvre through simulating all possible alternatives and measuring the execution time, success rate, and traffic flow influence of the alternatives. Using the same approach, different platoon formation strategies (Weakest-in-Front, Last-in-at-Tail, dynamic contextual strategies etc.) could be investigated regarding their influence on the overall traffic flow. The SEAD architecture has already been implemented and is a critical part of the Autonomous Vehicle Driver Model architecture described in [36].

The proposed framework utilises a timeout-strategy so that manoeuvres are not leading into a deadlock if, for instance, a human-driven vehicle is blocking the way to complete a manoeuvre. The timeout-strategy, however, is a mechanism to ensure deadlock-freeness. Research may investigate further models to identify blocking situations with a low likelihood of manoeuvre success to interrupt the manoeuvre using as a trigger an event rather than a time-out. Otherwise, simulations may allow to numerically optimise the time-out parameters with, for instance, the overall traffic flow as the objective variable to maximise.

After all, platoon manoeuvring is a powerful yet complex technique to fulfil the ever-increasing traffic demand with the given capacities we have. Further research will have to investigate many more gaps and find the most effective strategies to maximise traffic throughput. To unlock the full potential of platooning, the SEAD framework aims to pave the way for this future research by providing a formalisation of the platooning logic and simplifying the way how new manoeuvres are created and existing ones are compared and optimised.

REFERENCES

[1] P. Varaiya, "Smart cars on smart roads: Problems of control," *IEEE Trans. Autom. Control*, vol. 38, no. 2, pp. 195–207, Feb. 1993.

[2] D. N. Godbole, F. Eskafi, E. Singh, and P. Varaiya, "Design of entry and exit maneuvers for IVHS," in *Proc. Amer. Control Conf. (ACC)*, vol. 5, Jun. 1995, pp. 3576–3580.

[3] A. Hsu, F. Eskafi, S. Sachs, and P. Varaiya, "Protocol design for an automated highway system," *Discrete Event Dyn. Syst.*, vol. 2, nos. 3–4, pp. 183–206, Feb. 1993.

[4] J. Ivanchev, D. Zehe, V. Viswanathan, S. Nair, and A. Knoll, "Bisos: Backwards incremental system optimum search algorithm for fast socially optimal traffic assignment," in *Proc. IEEE 19th Int. Conf. Intell. Transp. Syst. (ITSC)*, Nov. 2016, pp. 2137–2142.

[5] M. Amoozadeh, H. Deng, C.-N. Chuah, H. M. Zhang, and D. Ghosal, "Platoon management with cooperative adaptive cruise control enabled by VANET," *Veh. Commun.*, vol. 2, no. 2, pp. 110–123, 2015.

[6] A. Hsu, S. Sachs, F. Eskafi, and P. Varaiya, "The design of platoon maneuvers for IVHS," in *Proc. Amer. Control Conf.*, Jun. 1991, pp. 2545–2550.

[7] R. Rajamani, H.-S. Tan, B. K. Law, and W.-B. Zhang, "Demonstration of integrated longitudinal and lateral control for the operation of automated vehicles in platoons," *IEEE Trans. Control Syst. Technol.*, vol. 8, no. 4, pp. 695–708, Jul. 2000.

[8] E. M. Clarke and R. P. Kurshan, "Computer-aided verification," *IEEE Spectr.*, vol. 33, no. 6, pp. 61–67, Jun. 1996.

[9] H. H. Bengtsson, L. Chen, A. Voronov, and C. Englund, "Interaction protocol for highway platoon merge," in *Proc. IEEE 18th Int. Conf. Intell. Transp. Syst.*, Sep. 2015, pp. 1971–1976.

[10] M. Segata, B. Bloessl, S. Joerer, F. Dressler, and R. L. Cigno, "Supporting platooning maneuvers through IVC: An initial protocol analysis for the JOIN maneuver," in *Proc. 11th Annu. Conf. Wireless On-Demand Netw. Syst. Services (WONS)*, Apr. 2014, pp. 130–137.

[11] X.-Y. Lu and J. K. Hedrick, "Longitudinal control algorithm for automated vehicle merging," *Int. J. Control*, vol. 76, no. 2, pp. 193–202, 2003.

[12] X.-Y. Lu, H.-S. Tan, S. E. Shladover, and J. K. Hedrick, "Automated vehicle merging maneuver implementation for AHS," *Vehicle Syst. Dyn*, vol. 41, no. 2, pp. 85–107, 2004.

[13] C. Nowakowski, D. Thompson, S. E. Shladover, A. Kailas, and X.-Y. Lu, "Operational concepts for truck maneuvers with cooperative adaptive cruise control," *Transp. Res. Rec.*, vol. 2559, no. 1, pp. 57–64, 2016.

[14] H.-H. Hsu and A. Liu, "Platoon lane change maneuvers for automated highway systems," in *Proc. IEEE Conf. Robot., Automat. Mechatronics*, vol. 2, Dec. 2004, pp. 780–785.

[15] H. C.-H. Hsu and A. Liu, "Kinematic design for platoon-lane-change maneuvers," *IEEE Trans. Intell. Transp. Syst.*, vol. 9, no. 1, pp. 185–190, Jan. 2008.

[16] E. S. Kazerooni and J. Ploeg, "Interaction protocols for cooperative merging and lane reduction scenarios," in *Proc. IEEE 18th Int. Conf. Intell. Transp. Syst.*, Sep. 2015, pp. 1964–1970.

[17] M. C. Best, "Optimisation of high-speed crash avoidance in autonomous vehicles," *Int. J. Vehicle Auton. Syst.*, vol. 10, no. 4, pp. 337–354, 2012.

[18] W. Choi and D. Swaroop, "Assessing the safety benefits due to coordination amongst vehicles during an emergency braking maneuver," in *Proc. Amer. Control Conf.*, vol. 3, Jun. 2001, pp. 2099–2104.

[19] J. Frankel, L. Alvarez, R. Horowitz, and P. Li, "Safety oriented maneuvers for IVHS," *Vehicle Syst. Dyn.*, vol. 26, no. 4, pp. 271–299, 1996.

[20] M. Goli and A. Eskandarian, "Evaluation of lateral trajectories with different controllers for multi-vehicle merging in platoon," in *Proc. Int. Conf. Connected Vehicles Expo (ICCVE)*, Nov. 2014, pp. 673–678.

[21] A.-C. Huang and Y. J. Chen, "Safe platoon control of automated highway systems," *Proc. Inst. Mech. Eng., I, J. Syst. Control Eng.*, vol. 215, no. 5, pp. 531–543, Aug. 2001.

[22] S. Kato, S. Tsugawa, K. Tokuda, T. Matsui, and H. Fujii, "Vehicle control algorithms for cooperative driving with automated vehicles and intervehicle communications," *IEEE Trans. Intell. Transp. Syst.*, vol. 3, no. 3, pp. 155–161, Sep. 2002.

[23] P. Li, L. Alvarez, and R. Horowitz, "AHS safe control laws for platoon leaders," *IEEE Trans. Control Syst. Technol.*, vol. 5, no. 6, pp. 614–628, Nov. 1997.

[24] D. K. Murthy and A. Masrur, "Braking in close following platoons: The law of the weakest," in *Proc. Euromicro Conf. Digit. Syst. Design (DSD)*, Aug. 2016, pp. 613–620.

[25] D. K. Murthy and A. Masrur, "A subplatooning strategy for safe braking maneuvers," in *Proc. Euromicro Conf. Digit. Syst. Design (DSD)*, Aug. 2017, pp. 375–382.

[26] J. E. Naranjo, C. Gonzalez, R. Garcia, and T. D. Pedro, "Lane-change fuzzy control in autonomous vehicles for the overtaking maneuver," *IEEE Trans. Intell. Transp. Syst.*, vol. 9, no. 3, pp. 438–450, Sep. 2008.

[27] R. Rai, B. Sharma, and J. Vanualailai, "Real and virtual leader-follower strategies in lane changing, merging and overtaking maneuvers," in *Proc. 2nd Asia–Pacific World Congr. Comput. Sci. Eng. (APWC CSE)*, Dec. 2015, pp. 1–12.

[28] X. Sun, R. Horowitz, and C.-W. Tan, "An efficient lane change maneuver for platoons of vehicles in an automated highway system," in *Proc. ASME Int. Mech. Eng. Congr. Expo.*, Jan. 2003, pp. 355–362.

[29] D. Swaroop and S. M. Yoon, "Integrated lateral and longitudinal vehicle control for an emergency lane change manoeuvre design," *Int. J. Vehicle Des.*, vol. 21, nos. 2–3, pp. 161–174, 1999.

[30] G. Usman and F. Kunwar, "Autonomous vehicle overtaking—An online solution," in *Proc. IEEE Int. Conf. Automat. Logistics*, Aug. 2009, pp. 596–601.

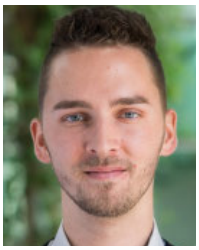
[31] Z. Wang, G. Wu, P. Hao, K. Boriboonsomsin, and M. Barth, "Developing a platoon-wide eco-cooperative adaptive cruise control (CACC) system," in *Proc. IEEE Intell. Vehicles Symp. (IV)*, Jun. 2017, pp. 1256–1261.

[32] C. Bergenhem, "Approaches for facilities layer protocols for platooning," in *Proc. IEEE 18th Int. Conf. Intell. Transp. Syst.*, Sep. 2015, pp. 1989–1994.

[33] *IEEE Guide for Wireless Access in Vehicular Environments (WAVE)—Architecture*, Standard 1609.0-2013, IEEE, 2014.

[34] C. DeBoeser, "Towards a hierarchical state-machine-based framework for platoon manoeuvre descriptions," M.S. thesis, Dept. Inform., Technical Univ. Munich, Munich, Germany, 2019.
[35] J. Ivanchev, T. Braud, D. Eckhoff, D. Zehe, A. Knoll, and A. Sangiovanni-Vincentelli, "On the need for novel tools and models for mixed traffic analysis," in *Proc. 26th Int. Transp. Syst. World Congr.*, Singapore, 2019, pp. 1–8.
[36] T. Braud, J. Ivanchev, C. Deboeser, A. Knoll, D. Eckhoff, and A. Sangiovanni-Vincentelli, "AVDM: A hierarchical command- and-control system architecture for cooperative autonomous vehicles in highways scenario using microscopic simulations," *Auto. Agents Multi-Agent Syst.*, vol. 35, no. 1, pp. 1–30, Apr. 2021.

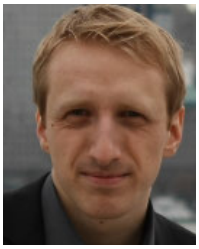
JORDAN IVANCHEV (Member, IEEE) received the Ph.D. degree in computer science from Technische Universität München (TUM), in 2017. In June 2013, he joined TUMCREATE to work on traffic modeling and optimization, where he is currently a Senior Research Fellow. His research interests include mixed traffic modeling and simulation, automated simulation calibration, optimal sensor placement problems, social optimal routing strategies, traffic modeling, and sensing and control (jordan.ivanchev@tum-create.edu.sg).



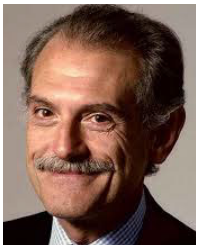
CORVIN DEBOESER received the M.Sc. degree in mechanical engineering from Technische Universität München (TUM), Germany, in 2019, under the supervision of Dr. A. Knoll. His research interests include autonomous vehicle platooning and modeling of electric vehicles (deniscorvin.deboeser@bears-berkeley.sg).

THOMAS BRAUD received the Ph.D. degree in computer science from Sorbonne University, in 2018. In 2018, he joined the Area-Interlinking Design Analysis Group, TUMCREATE, as a Research Fellow. Since 2018, he has been involved in a joint project between TUMCREATE and the University of California, Berkeley. His research interests include mixed traffic modeling and simulation, sensor fusion algorithms, and lastly, modeling autonomous vehicle driving behaviour (thomas.braud@tum-create.edu.sg).

ALOIS KNOLL received the Ph.D. degree in computer science from the Technical University of Berlin. In 1993, he joined the Technical Faculty of the University of Bielefeld, where he was a Full Professor and the Director of the Technical Informatics Research Group, until 2001. Since Autumn 2001, he has been a Professor of computer science with the Computer Science Department, Technische Universität München. His research interests include cognitive, medical and sensor-based robotics, multi-systems, data fusion, adaptive systems, and multimedia information retrieval (knoll@in.tum.de).



DAVID ECKHOFF (Member, IEEE) received the Dr.-Ing. degree (Hons.) in engineering, in 2016. In October 2016, he joined TUMCREATE, Singapore, a joint research institute by TU Munich and Nanyang Technical University, Singapore, where he is currently a Principal Investigator. His research interests include privacy protection, smart cities, vehicular networks, and intelligent transportation systems with a particular focus on modeling and simulation (david.eckhoff@tum-create.edu.sg).



ALBERTO SANGIOVANNI-VINCENTELLI holds Buttner Chair at the Department of EECS, University of California, Berkeley. He is an ACM Fellow and a member of the National Academy of Engineering. He was a recipient of Kaufman Award of the Electronic Design Automation Council for "pioneering contributions to EDA" and the IEEE/RSE Maxwell Medal "for groundbreaking contributions that have had an exceptional impact on the development of electronics and electrical engineering or related fields" (alberto@berkeley.edu).