Formal Verification of Platoon Control Strategies

Adnan Rashid[1], Umair Siddique[2] and Osman Hasan[1]

[1]School of Electrical Engineering and Computer Science (SEECS)
National University of Sciences and Technology (NUST)
Islamabad, Pakistan
{adnan.rashid,osman.hasan}@seecs.nust.edu.pk
[2]Department of Computing and Software
McMaster University, Hamilton, Canada
siddiu3@mcmaster.ca

Abstract. Recent developments in autonomous driving, vehicle-to-vehicle communication and smart traffic controllers have provided a hope to realize platoon formation of vehicles. The main benefits of vehicle platooning include improved safety, enhanced highway utility, efficient fuel consumption and reduced highway accidents. One of the central components of reliable and efficient platoon formation is the underlying control strategies, e.g., constant spacing, variable spacing and dynamic headway. In this paper, we provide a generic formalization of platoon control strategies in higher-order logic. In particular, we formally verify the stability constraints of various strategies using the libraries of multivariate calculus and Laplace transform within the sound core of HOL Light proof assistant. We also illustrate the use of verified stability theorems to develop runtime monitors for each controller, which can be used to automatically detect the violation of stability constraints in a runtime execution or a logged trace of the platoon controller. Our proposed formalization has two main advantages: 1) it provides a framework to combine both static (theorem proving) and dynamic (runtime) verification approaches for platoon controllers; and 2) it is inline with the industrial standards, which explicitly recommend the use of formal methods for functional-safety, e.g., automotive ISO 26262.

Keywords: Autonomous Driving, Platoon Control, Formal Verification

1 Introduction

Autonomous cars seem to be just around the corner, as most of the car manufacturers (e.g., Tesla, BMW, Toyata, Nissan, Ford, Jaguar Land Rover, etc.) and even silicon valley players (e.g., Intel and Nvidia) claim that fully autonomous vehicles will be on the road around 2020 [1,2]. Such a speedy development in autonomous driving is motivated by the fact that the autonomous cars will be more safe and crashless than the human driven cars. For example, the human error is to blame for up to 90% of the 1.2 million deaths that occur each year from car accidents around the world [3]. Like various fields of science and engineering,

the developments in autonomous driving have opened the doors to many other interesting fields, for example, *automated vehicle platooning* is one of the most benefitting fields.

A *platoon* is a group of vehicles (as shown in Figure 1) that travels in a close proximity to one another, nose-to-tail, at highway speeds. Vehicle platoons have

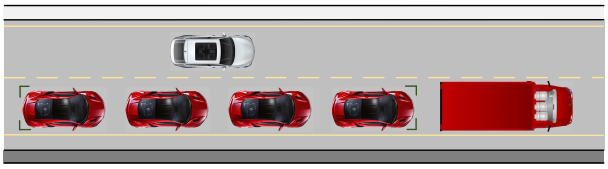

Fig. 1: Platoon of Vehicles

been proposed since at least the early 1980's even before we had wireless communication, global positioning system (GPS) and commercially available radar sensors. However, given the exceptional capabilities and reliability of the autonomous cars, vehicle platooning can become a reality using a mix of available technologies such as drive-by-wire steering [4], radar cruise control [5] and lane keep assist systems [6] to name a few. Some of the main advantages of the vehicle platooning are increased road capacity, reduction in drag and improved fuel economy, improved traffic congestion strategies [7] and reduced roadside accidents due to the autonomous features, e.g., collision detection [8] and automatic emergency braking [9].

The stability of the automated vehicles in a platoon, individually or as a group, depends on the interaction of the vehicles and is vital for an uninterrupted flow of traffic and a better throughput. A *stable platoon* ensures that the vehicles should not collide with each other while maintaining a safe inter-vehicle spacing bound. In practice, the stability of platoon is ensured by two types of controllers, i.e., autonomous and non-autonomous [10]. The autonomous controllers use the on-board sensors for determining the speed and position of the connected vehicles, whereas the non-autonomous controllers are based on other forms of the inter-vehicle communication. Furthermore, communication amongst controllers is either unidirectional or bidirectional, based on the information shared between the neighbouring vehicles. Similarly, various strategies can be used for the platoon control, such as constant spacing, variable spacing and variable time headway.

Traditionally, the platoon controllers are analyzed using informal approaches including paper-and-pencil based proofs [10] and numerical simulations [11]. These informal approaches have known limitations when used in safety-critical domains, for example, missing assumptions and even wrong derivations in hand-driven proofs, and inherent incompleteness of the numerical algorithms, respectively. Considering these facts, it is natural to think about complementing traditional analysis approaches with formal methods for developing reliable pla-

toon controllers. In this direction, model checking has been used to verify the high-level models of the platoon controllers using the temporal logic based properties [12,13,14]. In all these approaches, the authors considered the vehicles platoon and their controllers as a discrete-time system by modeling them as automata and verified their properties, such as safety and inter-vehicle spacing bound properties. Thus, these model checking based analysis lacks the physical analysis of the platoon, which requires modeling and reasoning of control strategies using differential equations and their frequency domain stability analysis using Laplace transform. Similarly, Mashkoor et al. [15] used higher-order logic to formally reason about the cyber-physical transportation system. The authors used random variables to model the unpredictable elements of the system and formally conducted a probabilistic analysis of the transportation system without considering the dynamic aspects of the system. In this paper, we propose a higher-order logic based framework to formally model and verify the stability of various types of platoon controllers using the HOL Light proof assistant [16]. Consequently, we utilize the verified results to construct monitors, which ensure the platoon stability at runtime. The main reasons for using higher-order logic and HOL Light include the expressibility to represent the platoon controllers, which are modeled using differential equations in time-domain and the Laplace transform in frequency-domain. Moreover, HOL Light has the smallest trusted core (i.e., approximately 400 lines of Ocaml code) amongst all other HOL proof assistants and the underlying logic kernel has been verified in the CakeML project [17].

The main contributions of the paper are as follows:

- Deep embedding based formalization of platoon controller types, configurations and strategies along with the associated differential equations based functional models.
- Formal derivation of the Laplace domain transfer functions using the formalized libraries of multivariate calculus [18] and the Laplace transform [19,20] in the HOL Light proof assistant.
- Formal verification of the platoon control strategies based on the formal models of various controllers.
- Development of the stability monitors for each type of the controllers and demonstration of their violation detection capability on a pseudo-randomly generated traces of a platoon controller.

The source code of our HOL Light development is available for download at [21] and thus can be used by the other researchers and engineers interested in the design and verification of the platoon controllers.

The rest of the paper is organized as follows: Section 2 presents an overview of the HOL Light proof assistant along with the formalization of the Laplace transform. Section 3 provides the formal modeling of the platoon controller and its stability. We provide the formal verification of the platoon control strategies and the stability constraints in Section 4. Section 5 describes the construction of the stability monitors. Finally, Section 6 concludes the paper and highlights some future research directions.

2 Preliminaries

This section presents a brief introduction to the HOL Light proof assistant and its multivariate calculus and the Laplace transform theories, which are extensively used in the rest of the paper.

2.1 Theorem Proving and HOL Light Proof Assistant

Theorem proving is a widely adapted formal verification technique, which is concerned with constructing the proofs of the mathematical theorems using a computer program (called *theorem prover or proof assistant*) [22]. Theorem proving systems have been commonly used for verifying the properties of the software and hardware systems. For example, a hardware designer can certify a digital circuit by modeling its behavior by some predicates and verifying its different properties using Boolean algebra. Similarly, a mathematician can verify the transitive property of the ordering of real numbers using some basic axioms of real numbers theory. These properties are expressed as theorems using some logic, such as propositional, first-order or higher-order logic, based on the required expressiveness. For example, using the higher-order logic is advantageous over the first-order logic as it provides the additional quantifiers and is more expressive as well. Moreover, higher-order logic can better describe the complex mathematical concepts including multivariate calculus, transcendental functions and topological spaces. Once such a mathematical theory is developed inside a proof assistant, we say that it is formalized.

HOL Light [16] is an interactive theorem proving environment for constructing the mathematical proofs. The main implementation of HOL Light is done in a functional programming language, Objective CAML (OCaml), which is originally developed to automate the mathematical proofs [23]. The logical kernel of HOL Light is of approximately 400 lines of OCaml code and its main components are its types, terms, theorems, rules of inference, and axioms. A theory in HOL Light consists of types, constants, definitions, axioms and theorems. The HOL Light theories are ordered in a hierarchical fashion and the child theories can inherit the types, definitions and theorems of the parent theories. Every new theorem has to be verified based on the primitive inference rules and basic axioms or already verified theorems present in HOL Light, which ensures the soundness of this technique.

2.2 Multivariable Calculus and Laplace Transform Theories

HOL Light provides an extensive support for the analysis of physical systems based on multivariate calculus theories, which include derivatives, integration, transcendental theory, topology, vector analysis and Laplace transform theory. Table 1 presents some definitions from the Laplace transform theory of HOL Light, which includes the Laplace transform, Laplace existence and the exponential-order conditions, and the differential equation of order n. Interested readers can refer to [19,20] for more details about this theory. It is extensively used in our proposed verification of the platoon control strategies for the automated vehicles.

Table 1: Laplace Transform

Mathematical Form	Formalized Form
Laplace Transform	
$\mathcal{L}[f(t)] = F(s) = \int_0^\infty f(t)e^{-st}dt,\ s \in \mathbb{C}$	⊢ ∀ s f. **laplace_transform** f s = integral {t | &0 <= drop t} (λt. cexp (−−(s ∗ Cx (drop t))) ∗ f t)
Laplace Existence	
f is piecewise smooth and is of exponential order on the positive real line	⊢ ∀ s f. **laplace_exists** f s ⇔ (∀ b. f piecewise_differentiable_on interval [lift (&0),lift b]) ∧ (∃ M a. Re s > drop a ∧ exp_order_cond f M a)
Exponential-order Condition	
There exist a constant a and a positive constant M such that $\|f(t)\| \leq Me^{at}$	⊢ ∀ f M a. **exp_order_cond** f M a ⇔ &0 < M ∧ (∀ t. &0 <= t ⇒ norm (f (lift t)) <= M ∗ exp (drop a ∗ t))
Differential Equation of Order n	
Differential Equation = $\sum_{k=0}^{n} \alpha_k \frac{d^k f}{dt^k}$	⊢ ∀ n f t. **diff_eq_n_order** n lst f t = vsum (0..n) (λk. EL k [$\alpha_1, \alpha_2, ..., \alpha_k$] ∗ higher_order_derivative k f t)

3 Formal Modeling of Platoon Controller and Stability

In this section, we present the formal modeling of a platoon controller based on its types, configurations and the underlying strategies along with the concept of the platoon stability.

3.1 Formalization of Platoon Controller

The connected vehicles in a platoon are widely characterized by the controllers, which are mainly responsible for their automated operation. The platoon controllers are generally of two types namely autonomous and non-autonomous.

- *Autonomous controllers* use the on-board sensors for determining the speed and position of the connected vehicles.
- *Non-autonomous controllers* are based on some other form of the inter-vehicle communication.

The information sharing among the neighbouring vehicles is either unidirectional or bidirectional depending upon the configuration of the platoon controllers.

- *Unidirectional configuration* allows a controller to use the information about the relative distance and velocity of only the preceding vehicle.
- *Bidirectional controller* can access the information about the relative distance and velocity of both the proceeding and preceding vehicles by considering their individual masses.

The autonomous controllers can adapt different strategies to maintain the stable operation of the platoon along the highway. In general, controllers utilize three strategies namely constant spacing, variable spacing and variable time-headway.

- *Constant spacing policy* requires that each vehicle maintains a constant distance (spacing) with its preceding vehicle in a platoon.
- *Variable spacing policy* allows a variable inter-vehicle spacing, which depends on the velocity of the vehicles in a platoon. For example, a faster moving vehicle creates more inter-vehicle space between itself and its proceeding vehicle. It is also known as the *constant time headway spacing*.
- *Variable time headway* policy imposes constraints on the relative velocity rather than the absolute velocity of the vehicle in contrast to the constant time headway spacing policy, which results into large inter-vehicle spaces and thus decreases the throughput of the highway traffic.

In our formalization, we model the types of the controller, its configurations and strategies as enumerated datatype using the built-in define_type mechanism in HOL Light.

```
type controller_type = autonomous | non_autonomous
type configuration = unidirectional | bidirectional
type strategy = constant_spacing | variable_spacing | var_time_headway
```

We model a platoon as a tuple $(x, n, m, k, c, ch, vd, h0, ca, cd)$, where the description and the type of each parameter is given in Table 2. Indeed, these parameters characterize various physical aspects of the vehicles in a platoon (e.g., the horizontal distance x, the number of vehicles in a platoon n and the mass of a vehicle m). In HOL Light, we formalize the platoon tuple $(x, n, m, k, c, ch, vd, h0, ca, cd)$ and controller tuple $(controller\_type, configuration, strategy, platoon)$ as type abbreviations:

```
type_abbrev platoon:(x × n × m × k × c × h × ch × vd × h0 × ca × cd)
type_abbrev controller:(controller_type × configuration × strategy × platoon)
```

It is important to note that `platoon` contains a unique mass m, which implies that we only consider a platoon with identical vehicles as shown in Figure 1.

In order to ensure that the given parameters of a platoon indeed represent a valid platoon, we formalize the associated constraints as a predicate is_valid_platoon (Definition 1). For example, the mass m should always be greater than 0 and the number of vehicles in a platoon should be greater than 1.

Definition 1. Valid Platoon

```
⊢ is_valid_platoon (x,n,m,k,c,h,ch,vd,h0,ca,cd) ⇔ 0 < m ∧ 0 < k ∧ 0 < c ∧
                0 < h ∧ 0 < ch ∧ 0 < vd ∧ 0 < h0 ∧ 0 < ca ∧ 0 < cd ∧ 1 < n
```

Table 2: Data Types for Platoon Parameters

Parameter Description	Type
Horizontal distance	x:$\mathbb{N} \to (\mathbb{R} \to \mathbb{C})$
Number of vehicles	n:$\mathbb{N}$
Mass of a vehicle	m:$\mathbb{R}$
Disturbance constant	k:$\mathbb{R}$
Fluctuations constant	c:$\mathbb{R}$
Time headway	h:$\mathbb{R}$
Fluctuations due to time headway	ch:$\mathbb{R}$
Desired platoon speed	vd:$\mathbb{R}$
Nominal value of time headway	h0:$\mathbb{R}$
Additional fluctuations with respect to platoon leader	ca:$\mathbb{R}$
Additional fluctuations with respect to "virtual" mass	cd:$\mathbb{R}$

3.2 Formalization of the Platoon Stability

The *stability* is an important property of a vehicle platoon, which describes the capability of a platoon to attenuate the oscillations introduced by the leader or any other vehicle in the platoon. In general, such oscillations can be considered in terms of various signals. e.g., the position error between the vehicles or the relative acceleration of connected vehicles. In this paper, we consider the notion of stability with respect to the position error between the vehicles. Formally, a platoon is stable if any oscillation with respect to the position error diminishes out as it propagates towards the tail of the platoon. The platoon stability in longitudinal direction is mathematically expressed as a norm condition on spacing errors in the frequency domain, as given in the following equation:

$$\left|\left| \frac{z_n(i\omega)}{z_{n-1}(i\omega)} \right|\right| < 1, \quad n = 2, 3, 4, ... \tag{1}$$

where z_{n-1} is the spacing error between the vehicle $n-1$ and its proceeding vehicle n, i.e., it is the deviation from the desired inter-vehicle spacing for vehicle $n-1$. If x_{n-1} is the inter-vehicle spacing between the vehicle $n-1$ and its preceding vehicle $n-2$ and x_n is the inter-vehicle spacing between the vehicle n and its preceding vehicle $n-1$, then the spacing error between vehicles $n-1$ and n is given by $z_{n-1} = x_{n-1} - x_n$. Similarly, z_n represents the spacing error between the vehicle n and its proceeding vehicle $n+1$, i.e., $z_n = x_n - x_{n+1}$. In case of all the desired inter-vehicle spacings are same, i.e., $x_n = x_{n-1} = ... = x_1$, then this leads to the zero spacing errors, i.e., $z_n = z_{n-1} = ... = z_1 = 0$. We formalize platoon stability in HOL Light as follows:

Definition 2. Stable Platoon

$$\vdash \forall\, \mathsf{s}\ \mathsf{x}\ \mathsf{y}.\ \textbf{transfer\_function}\ \mathsf{s}\ \mathsf{x}\ \mathsf{y} \Leftrightarrow \frac{\mathsf{laplace\_transform\ y\ s}}{\mathsf{laplace\_transform\ x\ s}}$$

$$\vdash \forall\, \omega\ \mathsf{x}\ \mathsf{y}.\ \textbf{frequency\_response}\ \omega\ \mathsf{x}\ \mathsf{y} \Leftrightarrow \mathsf{transfer\_function}\ (\mathsf{i}\omega)\ \mathsf{x}\ \mathsf{y}$$

$$\vdash \forall\, \omega\ \mathsf{z}.\ \textbf{is\_stable\_platoon}\ \omega\ \mathsf{z}\ \mathsf{n} \Leftrightarrow \left\| \mathsf{frequency\_response}\ \omega\ (\lambda \mathsf{t}.\ \mathsf{z}\ (\mathsf{n}))\ (\lambda \mathsf{t}.\ \mathsf{z}\ (\mathsf{n}\ -\ 1)) \right\| < 1$$

where the predicate is_stable_platoon accepts the parameters $\mathsf{z}{:}\mathbb{N} \rightarrow (\mathbb{R} \rightarrow \mathbb{C})$, which represents the complex-domain representation of the spacing error, angular frequency $\omega{:}\mathbb{R}$ and number of vehicles n, and returns the condition that the complex norm of the transfer function at $s = i\omega$, i.e., $\frac{Z_n(i\omega)}{Z_{n-1}(i\omega)}$ is always less than 1 for every vehicle in the platoon.

This concludes our fundamental formalization of the platoon controller and the corresponding stability. We build upon the concepts, formalized in this section, to formalize various control strategies and verify their correctness in the next section.

4 Formal Verification of the Platoon Control Strategies

In this section, we first present the formalization of an autonomous unidirectional controller with constant spacing policy. Indeed, the main intention is to demonstrate the formalization steps, i.e., formal modeling of the controller dynamics in higher-order logic, formalization of the necessary constraints, and the formal verification of the stability theorem. Building upon these steps, we next present its generalization to all types of controllers along with the verification of a generalized stability theorem.

4.1 Autonomous Unidirectional Controller

Generally, the dynamics of platoon controllers are characterized by a set of differential equations, which interrelate the parameters of the platoon. The schematic representation of the platoon of vehicles having autonomous unidirectional controller with constant spacing policy is depicted in Figure 2. It consists of n interconnected vehicles of identical masses, i.e., $m_1 = m_2 = ... = m_{n-1} = m_n = m$. The parameters k and c are the disturbance and fluctuation constants, representing the control gains on the relative position and velocity, respectively. Similarly, the parameter u represents the force required by the first vehicle to move forward in the platoon. The mathematical representation of this platoon controller's dynamics are given as the following equation set:

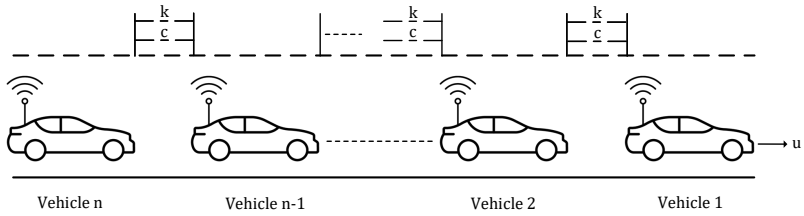

Fig. 2: Autonomous Unidirectional Controller with Constant Spacing

$$\begin{aligned}
\frac{dx_1}{dt} &= v_1, \quad \frac{dv_1}{dt} = \frac{u}{m}, \quad \frac{dx_2}{dt} = v_2 \\
\frac{dv_2}{dt} &= \frac{k}{m}x_1 - \frac{k}{m}x_2 + \frac{c}{m}v_1 - \frac{c}{m}v_2 \\
&\vdots \\
\frac{dx_{n-1}}{dt} &= v_{n-1} \\
\frac{dv_{n-1}}{dt} &= \frac{k}{m}x_{n-2} - \frac{k}{m}x_{n-1} + \frac{c}{m}v_{n-2} - \frac{c}{m}v_{n-1} \\
\frac{dx_n}{dt} &= v_n \\
\frac{dv_n}{dt} &= \frac{k}{m}x_{n-1} - \frac{k}{m}x_n + \frac{c}{m}v_{n-1} - \frac{c}{m}v_n
\end{aligned} \tag{2}$$

where the variables x and v represent the inter-vehicle spacing and velocity of platoon vehicles, respectively. Overall, the set of differential equations characterize the dynamics of n vehicles in the platoon depicted in Figure 2. We can rewrite Equation (2) in a compact form by eliminating the variable v and representing it in the form of spacing error, i.e., z as:

$$\frac{d^2z_n}{dt^2} + \frac{c}{m}\frac{dz_n}{dt} + \frac{k}{m}z_n = \frac{c}{m}\frac{dz_{n-1}}{dt} + \frac{k}{m}z_{n-1}, \quad n = 2, 3, 4, \ldots \tag{3}$$

We formally model this controller in HOL Light as follows:

Definition 3. Unidirectional Controller with Constant Spacing

```
⊢ ∀ k c m ch n vd ca cd h h0 x.
  control_uni_cs (autonomous,unidirectional,constant_spacing,
                                  ((x,n,m,k,c,h,ch,vd,h0,ca,cd):platoon)) t ⇔
    let z_{n-1} = (λt. x (n - 1) t - x (n) t) and
        z_n = (λt. x (n) t - x (n + 1) t) in
              D^2 [k/m; c/m; 1] z_n = D^1 [k/m; c/m] z_{n-1}
```

where the operators $\mathcal{D}^1$ and $\mathcal{D}^2$ represent the first-order and second-order complex-valued derivatives in HOL Light, respectively, and thus can be obtained by instantiating $n = 1$ and $n = 2$ in the predicate diff_eq_n_order, given in Table 1.

We next model some physical constraints associated with the controller model control_uni_cs, which include differentiability, existence of the Laplace transform and zero-initial conditions for parameters z_{n-1} and z_n, as given in Definition 4.

Definition 4. Constraints for a Platoon having Autonomous Unidirectional Controller

```
⊢ ∀ x n s c m k.
    constraints_uni_cs x n s c m k ⇔
      let z_{n-1} = (λt. x (n - 1) t - x (n) t) and
          z_n = (λt. x (n) t - x (n + 1) t) in
            (∀t. differentiable_higher_derivative [2,1] [z_{n-1},z_n]) ∧
              laplace_exists_higher_deriv [2,1] [z_{n-1},z_n] s ∧
              zero_initial_conditions [1,0] [z_{n-1},z_n] ∧
              non_zero_tf_uni_cs z_{n-1} s c m k
```

where the first two conjuncts provide the differentiability and the Laplace existence conditions for the second-order and first-order derivatives of the spacing errors z_{n-1} and z_n, respectively. Similarly, the next conjunct imposes the zero-initial conditions for the spacing errors z_{n-1} and z_n, respectively. Finally, the last conjunct ensures that the transfer function does not include the singularities, i.e., the points at which the denominator of the transfer function becomes infinite or undefined. Mathematically, it is described as $s^2 + \frac{c}{m}\, s + \frac{k}{m} \neq 0$.

Our next step is to formally verify that the platoon controller model control_uni_cs implies the platoon stability for any number of vehicles. The main purpose of this verification is twofold: 1) identify the stability constraints in terms of the platoon parameters, and 2) utilize verified stability constraints to ensure the stability of a given platoon at any time instant. Indeed this step requires the instantiation of platoon parameters with concrete values (i.e., number of vehicles $n = 10$, mass $m = 1000kg$, etc.). We verify the following universally quantified stability theorem in HOL Light.

Theorem 1. Stability of a Platoon having Autonomous Unidirectional Controller

```
⊢ ∀ k c m ch n vd ca cd h h0 x w.
 let s = iω and
     p = ((x,n,m,k,c,h,ch,vd,h0,ca,cd):platoon) and
     z = (λ n t. x (n) t - x (n + 1) t) in
  0 < ω ∧ 2k/m < ω² ∧ valid_platoon p ∧ constraints_uni_cs x n s c k m ∧
  ∀ t. control_uni_cs (autonomous,unidirectional,constant_spacing,p) t
    ⟹ is_stable_platoon ω z n
```

The main proof of Theorem 1 consists of the following steps: 1) rewriting with the Definitions 1-4, 2) complex arithmetic reasoning and 3) the verification of Lemma 1, which transforms the time-domain model of the platoon controller control_uni_cs into its equivalent frequency-domain representation, i.e., transfer function. The verification of Lemma 1 is quite involved due to the reasoning

about the Laplace transform in HOL Light [19]. The formal statement of Lemma 1 is given as follows:

Lemma 1. Model Implies Transfer Function

$$\vdash \forall\ k\ c\ m\ ch\ n\ vd\ ca\ cd\ h\ h0\ x\ s.$$
$$\text{let } p = ((x,n,m,k,c,h,ch,vd,h0,ca,cd):\text{platoon}) \text{ and}$$
$$z_{n-1} = (\lambda t.\ x\ (n - 1)\ t - x\ (n)\ t) \text{ and}$$
$$z_n = (\lambda t.\ x\ (n)\ t - x\ (n + 1)\ t) \text{ in}$$
$$\texttt{valid\_platoon}\ p \wedge \text{constraints\_uni\_cs}\ x\ n\ s\ c\ k\ m \wedge$$
$$\forall\ t.\ \text{control\_uni\_cs}\ (\text{autonomous,unidirectional,constant\_spacing,p})\ t$$
$$\Longrightarrow \text{transfer\_function}\ s\ z_n\ z_{n-1} = \frac{\frac{c}{m}\ s\ +\ \frac{k}{m}}{s^2\ +\ \frac{c}{m}\ s + \frac{k}{m}}$$

4.2 Generalized Platoon Controller

We formally model various types of platoon control strategies as given in Table 3. We also formalize the physical constraints and verify the stability for each control strategy along the same lines as that of autonomous unidirectional controller presented in Section 4.1. Finally, we package them in an inductive predicate gen_platoon_control, which takes two parameters, i.e., controller and time t and returns the predicate describing the physical behavior of the controller. For example, for controller (autonomous,unidirectional,constant_spacing,platoon), the inductive predicate gen_platoon_control returns control_uni_cs[1].

Finally, we verify a general theorem, which describes the stability constraints for any type of the controller *cc*, as follows:

Theorem 2. Stability of a Platoon

$$\vdash \forall\ (cc:\text{controller})\ s.$$
$$\text{let } s = i\omega \text{ and}$$
$$p = (x,n,m,k,c,h,ch,vd,h0,ca,cd):\text{platoon and}$$
$$\texttt{cc = (ct,cg,sg,p)} \text{ and}$$
$$z = (\lambda n\ \ t.\ x\ (n)\ t - x\ (n + 1)\ t) \text{ in}$$
$$\text{gen\_stability\_physical\_constraints cc s } \omega \wedge \forall\ t.\ \text{gen\_platoon\_control cc t}$$
$$\Longrightarrow \texttt{is\_stable\_platoon}\ \omega\ \texttt{z n}$$

where the predicate gen_stability_physical_constraints encapsulates the stability and physical constraints of all types of controllers in our formalization. The formal proof of Theorem 2 is based on induction on cc:controller and further induction on the controller_type, configuration and strategy along with the verified stability theorems for each control strategy (e.g., Theorem 1 for autonomous unidirectional controller presented in Section 4.1).

This concludes our formalization of platoon controllers in the HOL Light proof assistant. In summary, we formalized the basic notions of the platoon controllers

[1] We have omitted the formal definition of gen_platoon_control for the sake of conciseness, however, interested reader can find the formal definition and HOL Light code on the project's webpage [21].

Table 3: Formal Platoon Models considering Various Control Strategies

Autonomous Unidirectional Controller with Speed-dependent Spacing
$\vdash \forall$ k c m ch n vd ca cd h h0 x. **control_uni_vs** (autonomous,unidirectional,variable_spacing, ((x,n,m,k,c,h,ch,vd,h0,ca,cd):platoon)) t $\Leftrightarrow$ let $z_{n-1} = (\lambda t.\ x\ (n - 1)\ t - x\ (n)\ t)$ and $z_n = (\lambda t.\ x\ (n)\ t - x\ (n + 1)\ t)$ in $\mathcal{D}^2\ [\frac{k}{m}; \frac{c+k*h}{m}; 1]\ z_n = \mathcal{D}^1\ [\frac{k}{m}; \frac{c}{m}]\ z_{n-1}$
Autonomous Unidirectional Controller with Variable Time Headway
$\vdash \forall$ k c m ch n vd ca cd h h0 x. **control_uni_vth** (autonomous,unidirectional,var_time_headway, ((x,n,m,k,c,h,ch,vd,h0,ca,cd):platoon)) t $\Leftrightarrow$ let $z_{n-1} = (\lambda t.\ x\ (n - 1)\ t - x\ (n)\ t)$ and $z_n = (\lambda t.\ x\ (n)\ t - x\ (n + 1)\ t)$ in $\mathcal{D}^2\ [\frac{k}{m}; \frac{c+k*h0+k*ch*vd}{m}; 1]\ z_n = \mathcal{D}^1\ [\frac{k}{m}; \frac{c+k*ch*vd}{m}]\ z_{n-1}$
Autonomous Bidirectional Controller with Constant Spacing
$\vdash \forall$ k c m ch n vd ca cd h h0 x. **control_bi_cs** (autonomous,bidirectional,constant_spacing, ((x,n,m,k,c,h,ch,vd,h0,ca,cd):platoon)) t $\Leftrightarrow$ let $z_{n-1} = (\lambda t.\ x\ (n - 1)\ t - x\ (n)\ t)$ and $z_n = (\lambda t.\ x\ (n)\ t - x\ (n + 1)\ t)$ in $\mathcal{D}^2\ [\frac{2*k}{m}; \frac{2*c}{m}; 1]\ z_n = \mathcal{D}^1\ [\frac{k}{m}; \frac{c}{m}]\ z_{n-1}$
Autonomous Bidirectional Controller with Speed-dependent Spacing
$\vdash \forall$ k c m ch n vd ca cd h h0 x. **control_bi_vs** (autonomous,bidirectional,variable_spacing, ((x,n,m,k,c,h,ch,vd,h0,ca,cd):platoon)) t $\Leftrightarrow$ let $z_{n-1} = (\lambda t.\ x\ (n - 1)\ t - x\ (n)\ t)$ and $z_n = (\lambda t.\ x\ (n)\ t - x\ (n + 1)\ t)$ in $\mathcal{D}^2\ [\frac{2*k}{m}; \frac{2*c+k*h}{m}; 1]\ z_n = \mathcal{D}^1\ [\frac{k}{m}; \frac{c}{m}]\ z_{n-1}$
Non-autonomous Controller considering Communication of Leader's Current Velocity
$\vdash \forall$ k c m ch n vd ca cd h h0 x. **control_clcv** non_autonomous ((x,n,m,k,c,h,ch,vd,h0,ca,cd):platoon) t $\Leftrightarrow$ let $z_{n-1} = (\lambda t.\ x\ (n - 1)\ t - x\ (n)\ t)$ and $z_n = (\lambda t.\ x\ (n)\ t - x\ (n + 1)\ t)$ in $\mathcal{D}^2\ [\frac{k}{m}; \frac{c+ca}{m}; 1]\ z_n = \mathcal{D}^1\ [\frac{k}{m}; \frac{c}{m}]\ z_{n-1}$

using the new type definition and corresponding physical and stability constraints. The notable feature of our formalization is its generic nature, as we can model a platoon controller with any number of vehicles composed of basic controller types, configurations and strategies. Moreover, the physical and stability constraints are explicitly present in our formally verified stability theorems, which may get ignored in the conventional platoon analysis and may result into an unstable platoon interrupting the traffic flow on the highways. In the next section, we describe the utilization of our verified results in HOL Light to develop stability monitors for automatically detecting the violations of the stability constraints.

5 From Verified Controller to Stability Monitors

Static formal verification approaches, such as theorem proving, provide an effective way to formally model and verify digital hardware, its underlying software, control and cyber-physical systems at an appropriate abstract level. For example, we employed higher-order logic to formalize various control strategies of a platoon due to the involvement of multivariate calculus (i.e, complex frequency domain and Laplace transform). Moreover, we formally verified some of the most important stability constraints for arbitrary platoon parameters. Indeed, this is one of the main strengths of the interactive proof assistants as compared to the simulation based analysis where verification holds only for the applied test cases and thus cannot be considered as complete. However, the verification of important properties of given system in a proof assistant does not guarantee that the system will behave as expected during the runtime operation. Indeed, the verified results in a proof assistant provide a confidence that the system will behave correctly only when the corresponding conditions are met at all times during the life-time of a system. Actually this falls under the scope of runtime verification approaches, which are light-weight formal methods to monitor the correctness of a given system with respect to a formal specification at runtime.

We demonstrate here the utilization of verified stability theorems for various control strategies to construct monitors, which are capable of detecting the violation on a given execution of the system. Consider that the behavior of a platoon controller at each time instant (called an event) is characterized by the tuple platoon and frequency w, i.e., event = ((x,n,m,k,c,h,ch,vd,h0,ca,cd),w). Thus, an execution of the platoon controller consists of the sequence of events and we model it as an event list in HOL Light. Next, we consider the autonomous unidirectional controller with constant spacing, for which the stability of the platoon is ensured if the following two conditions are met for every event in the controller execution. 1) P_1 : valid_platoon (x,n,m,k,c,h,ch,vd,h0,ca,cd) and 2) P_2 : $0 < \omega \wedge \frac{2k}{m} < \omega^2$. In terms of temporal logic, a formal requirement to ensure the platoon stability is $\Box P_1 \wedge \Box P_2$ where $\Box$ represents *Globally* (G) or an *Always* operator in the linear temporal logic (LTL). We can model this monitor in HOL Light as (ALL P_1 execution) $\wedge$ (ALL P_2 execution) where ALL is a HOL Light function, which ensures the satisfaction of a predicate on each element of the list. Moreover, we developed a tactic MONITOR_TAC, which automatically verifies that both the predicates P_1 and P_2 holds for a given platoon controller execution as a list of events. We tested the efficiency of the MONITOR_TAC on randomly generated executions and it can check the validity in a reasonable time. For example, on average MONITOR_TAC returns the truth (T) in 3 seconds on an execution of 1000 unique events.

The main purpose of the above illustration was to show that the efforts spent during the formalization within an interactive proof assistant can be complemented by the development of the monitors to ensure the correctness of the system operation at runtime, and thus closing the loop from abstract verification to the real-time monitoring on the concrete system. Our illustration only describes the off-line monitoring where the platoon controller execution is given

as a logged data. However, the same monitor can be used for the online monitoring by translating the monitor as a post-condition in the actual system implementation or by generating the monitor using well-known LTL3 [24] or the rewriting-based monitoring approaches [25].

We believe that the stability monitoring can be used for the already available platoon controllers by inspecting the logged traces and by adding monitors in the early controller prototypes for quickly evaluating the correctness of the underlying algorithms. Thus, engineers working on the design and development of the platoon controllers can use the proposed framework without any prior knowledge of theorem proving and gain formally analyzed insights about the given platoon control system.

6 Conclusion and Future Work

This paper provides a framework for analyzing the platoon control strategies using both the static and dynamic verification approaches. It mainly presents the formal modeling of the platoon controller and its stability using higher-order logic. Next, the proposed formalization is used for formally verifying various platoon control strategies and their stability within the sound core of the HOL Light proof assistant. Finally, the formally verified stability theorems are used to develop the runtime monitors for each of the controllers that are used for detecting the violation of any stability constraints.

In future, we plan to formally analyze the platoon considering different connected vehicles (having different masses). We can also incorporate the stability in lateral direction and their physical constraints in our framework for the platoon stability. The other direction is to formally analyze the platoon of connected vehicles, where some of the vehicles act in a malicious manner by changing the control gain and thus the properties of the controllers. Such scenarios can compromise the safety of other vehicles on the highways and result in destabilizing the traffic flow [26]. Finally, it is interesting to consider two-dimensional platoons, which can be analyzed by combining our current framework and formalization of the z-Transform [27], which is already available in the HOL Light proof assistant.

References

1. http://www.driverless-future.com/?page_id=384, 2018.
2. https://www.technologyreview.com/s/602196/2021-may-be-the-year-of-the-fully-autonomous-car/, 2018.
3. https://phys.org/news/2017-09-self-driving-cars-road-toll.html, 2018.
4. Z. Changfu and L. Kai. Development of the Drive-By-Wire Technology. *Automobile Technology*, 3(001), 2006.
5. B. Van A., C. JG Van D., and R. Visser. The Impact of Cooperative Adaptive Cruise Control on Traffic-flow Characteristics. *Transactions on Intelligent Transportation Systems*, 7(4):429–436, 2006.
6. H. Kawazoe, M. Shimakage, O. Sadano, and S. Sato. Lane Keeping Assistance System and Method for Automotive Vehicle, December 10 2002. US Patent 6,493,619.
7. P. Fernandes and U. Nunes. Platooning with IVC-enabled Autonomous Vehicles: Strategies to Mitigate Communication Delays, Improve Safety and Traffic Flow. *Transactions on Intelligent Transportation Systems*, 13(1):91–106, 2012.

8. S. Biswas, R. Tatchikou, and F. Dion. Vehicle-to-vehicle Wireless Communication Protocols for Enhancing Highway Traffic Safety. *Communications Magazine*, 44(1):74–82, 2006.
9. J. Yi, L. Alvarez, R. Horowitz, and C. C De Wit. Adaptive Emergency Braking Control using a Dynamic Tire/road Friction Model. In *Decision and Control*, volume 1, pages 456–461. IEEE, 2000.
10. J Eyre, D Yanakiev, and I Kanellakopoulos. A Simplified Framework for String Stability Analysis of Automated Vehicles. *Vehicle System Dynamics*, 30(5):375–405, 1998.
11. P. Barooah, P. G Mehta, and J. P Hespanha. Mistuning-based Control Design to Improve Closed-loop Stability Margin of Vehicular Platoons. *Transactions on Automatic Control*, 54(9):2100–2113, 2009.
12. M. Kamali, L. A. Dennis, O. McAree, M. Fisher, and S. M. Veres. Formal Verification of Autonomous Vehicle Platooning. *Science of Computer Programming*, 148:88–106, 2017.
13. E. Dolginova. *Safety Verification for Automated Vehicle Maneuvers*. PhD thesis, Massachusetts Institute of Technology, 1998.
14. T. Wongpiromsarn and R. M. Murray. Formal Verification of an Autonomous Vehicle System. In *Conference on Decision and Control*, 2008.
15. A. Mashkoor and O. Hasan. Formal Probabilistic Analysis of Cyber-physical Transportation Systems. In *International Conference on Computational Science and Its Applications*, volume 7335 of *LNCS*, pages 419–434. Springer, 2012.
16. J. Harrison. HOL Light: A Tutorial Introduction. In *Formal Methods in Computer-Aided Design*, volume 1166 of *LNCS*, pages 265–269. Springer, 1996.
17. R. Kumar. Self-compilation and Self-verification. Technical report, University of Cambridge, Computer Laboratory, 2016.
18. J. Harrison. The HOL Light Theory of Euclidean Space. *Journal of Automated Reasoning*, 50(2):173–190, 2013.
19. S. H. Taqdees and O. Hasan. Formalization of Laplace Transform Using the Multivariable Calculus Theory of HOL-Light. In *Logic for Programming, Artificial Intelligence, and Reasoning*, volume 8312 of *LNCS*, pages 744–758. Springer, 2013.
20. A. Rashid and O. Hasan. Formalization of Transform Methods using HOL Light. In *Conference on Intelligent Computer Mathematics*, volume 10383 of *LNAI*, pages 319–332. Springer, 2017.
21. A. Rashid. Formal Verification of Platoon Control Strategies. `http://save.seecs.nust.edu.pk/projects/fvpcs/`, 2018.
22. J. Harrison. *Handbook of Practical Logic and Automated Reasoning*. Cambridge University Press, 2009.
23. A History of OCaml. `http://ocaml.org/learn/history.html`, 2015.
24. A. Bauer, M. Leucker, and C. Schallhart. Runtime Verification for LTL and TLTL. *Transactions on Software Engineering and Methodology*, 20(4):14:1–14:64, September 2011.
25. K. Havelund and G. Rosu. Monitoring Programs using Rewriting. In *Automated Software Engineering*, pages 135–143, 2001.
26. D. D. Dunn. *Attacker-induced Traffic Flow Instability in a Stream of Automated Vehicles*. Utah State University, 2015.
27. U. Siddique, M. Y. Mahmoud, and S. Tahar. On the Formalization of Z-Transform in HOL. In *International Conference on Interactive Theorem Proving*, volume 8558 of *LNCS*, pages 483–498. Springer, 2014.