Modeling and Verification of Continuous Object Behavior and Discrete Control Actions in Situation Awareness Systems*

Stefan Mitsch

Marshall Plan Scholar
Johannes Kepler University Linz
Altenbergerstr. 69, 4040 Linz, Austria

Abstract. Large-scale control systems, as needed in road traffic management, typically deal with highly dynamic environments. They provide vast amounts of information about real-world objects (e.g., tunnels) and events (e.g., accidents). To counteract information overload in such systems, situation awareness aims at supporting human operators in taking appropriate control actions pro-actively, thus preventing critical events. As a pre-requisite for reasoning about the effects of control actions, object and event evolution must be modeled in a fine-grained manner, for instance in terms of describing future positions of objects on the basis of their velocity, or describing traffic flow in terms of vehicle density and flux. The effects of control actions (e.g., set speed limit) can then be described in terms of influencing the behavior of objects and events (e.g., a speed limit reduces the velocity of objects). As ultimate vision, operating procedures including control actions should be implemented within a situation awareness system in order to issue actions automatically. Such situation awareness systems would not be particularly useful, if the lead to incorrect control decisions. Therefore, the validity of the operating procedures must be verified, and certain invariants must be guaranteed within the system (e.g., control actions must not cause accidents).
In this work, we study how CPS technology can help improve control actions in traffic centers by combining local car GPS positioning, traffic center control decisions, and communication to achieve more tightly coupled feedback control in intelligent speed adaptation. In addition, we describe a textual modeling environment for the Eclipse platform, which supports developing hybrid models for CPS and integrates with the theorem prover KeYmaera.

Keywords: Situation Awareness, Cyber-physical system, hybrid system, modeling, verification

1 Introduction

Large-scale control systems, as needed in road traffic management, typically deal with highly dynamic environments. They provide vast amounts of information

* This work has been funded by Marshall Plan Foundation.

from multiple heterogeneous sources about a large number of real-world objects (e.g., tunnels, bridges) and events (e.g., accidents, traffic jams), which are anchored in time and space. In such systems, human operators are vulnerable to information overload and may fail to be aware of the overall meaning of available information and its implications. To counteract information overload, situation awareness aims at supporting human operators in assessing current situations and in predicting possible future ones to take appropriate actions pro-actively, preventing critical events.

Recently, ontology-based situation awareness systems have been proposed, including our prior work in the FIT-IT Semantic Systems project BeAware! [7]. In the course of this project, a qualitative approach to situation prediction has been developed, which augments conceptual neighborhood graphs representing evolution in qualitative spatial relation calculi with coarse-grained evolution events, such as motion, scaling, and rotation thus being able to increase prediction precision [6]. An encoding of these conceptual neighborhood graphs and evolution events in so-called Situation Prediction Nets [8] provides for a reasoner being capable of qualitative prediction and simulation.

In this qualitative prediction approach, the effects of control actions must be mapped onto one of the qualitative evolution possibilities. As a pre-requisite for reasoning about the effects of control actions, however, besides these qualitative options, object and event evolution must be modeled in a more fine-grained manner, for instance in terms of describing future positions of traffic objects on the basis of their velocity (in the case of individual vehicles), or describing traffic flow in terms of vehicle density and flux of vehicles. The effects of control actions (e.g., set speed limit) can then be described in terms of influencing the behavior of objects and events (e.g., a speed limit reduces the velocity of objects), resulting in more expressive predictions of possible future situations. As ultimate vision, not only the effects of possible control actions should be highlighted to human operators, but the operating procedures should be implemented within the system in order to issue actions automatically. As a pre-requisite for this, the validity of the operating procedures must be verified, since certain invariants must be guaranteed (e.g., the control actions must never result in causing an accident). In this work, we discuss the following contributions.

1. Fine-grained modeling of continuous object behavior with differential equations, since many real-world phenomena can be described by functions and their derivatives (e.g., position in terms of velocity)
2. Modeling of control actions as the basis for action decisions of human operators in terms of reactions to real-world preconditions in a discrete manner (e.g., set speed limits)
3. Verifying control action validity and their implementation in terms of discrete controllers; we describe continuous behavior models and control actions in differential dynamic logic [28, 29], and complement these models with system invariants, which altogether are verifiable by the KeYmaera prover.

In the next section, we describe a textual modeling environment for problem specifications in $\mathsf{d}\mathcal{L}$ provable by KeYmaera. This textual modeling environment

targets at supporting fine-grained modeling of continuous behavior, control actions, and system environments, and can load these models into the KeYmaera prover. Basing on this modeling environment, in Sect. 3, we describe a case study of a cooperation between a (semi-)autonomous vehicle and a traffic center. In this case study, the vehicle has to respect speed limits issued by the traffic center, which in turn must guarantee to only issue speed limits that are within the physical reaction boundaries (e.g., braking capability) of the vehicle. This case study was modeled in the described textual modeling environment, and its validity was proven in KeYmaera.

2 Textual Modeling of dℒ Problem Specifications for KeYmaera

In this section, we describe our textual modeling environment *KeYmaera Eclipse editor*. It is built using Eclipse Xtext[1], and bases on the grammar of KeY [24] and a description of the hybrid program syntax for hybrid systems[2]. Figure 1 shows a screenshot of the KeYmaera Eclipse editor.

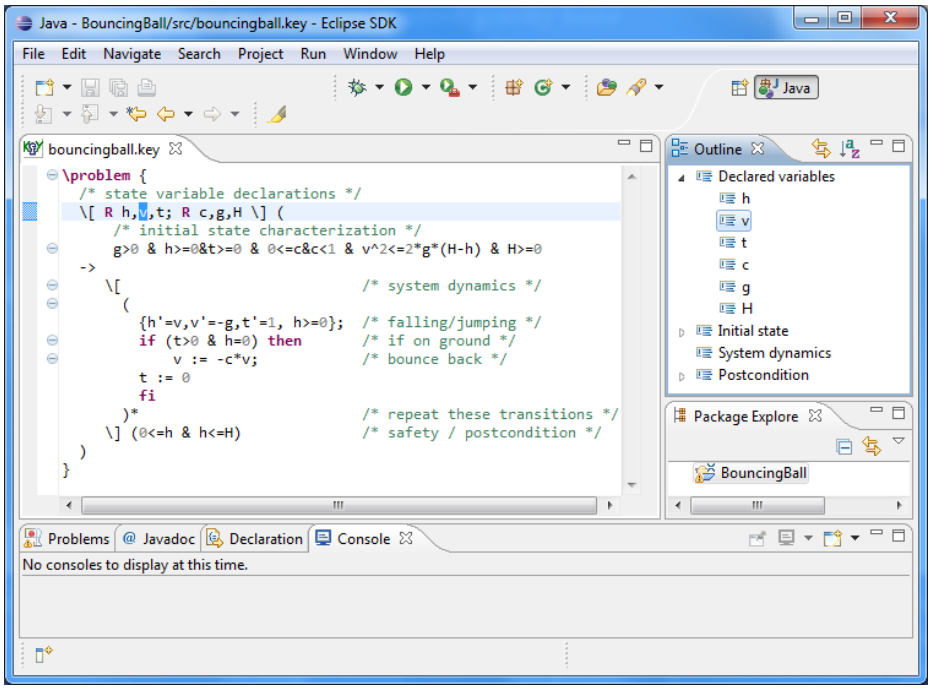


Fig. 1: Syntax-highlighting in the KeYmaera Eclipse editor

[1] www.eclipse.org/Xtext/
[2] www.symbolaris.com/info/KeYmaera-guide.html#keymaerafile

As introduction into hybrid systems modeling, in the following section, differential dynamic logic ($\mathsf{d}\mathcal{L}$) is described, which supports *hybrid programs* [28, 29] as a program notation for hybrid systems.

2.1 Preliminaries: Differential Dynamic Logic

For specifying and verifying correctness statements about hybrid systems, the KeYmaera Eclipse editor bases on *differential dynamic logic* $\mathsf{d}\mathcal{L}$ [28, 29]. The syntax of hybrid programs is summarized together with an informal semantics in Tab. 1.

The sequential composition $\alpha;\ \beta$ expresses that β starts after α finishes (e.g., first let a traffic center choose a maximum speed, then a position for a speed limit area). The nondeterminic choice $\alpha\ \cup\ \beta$ follows either α or β (e.g., let a traffic center decide nondeterministically between keeping an existing speed limit or choosing a new one). The nondeterministic repetition operator α^* repeats α zero or more times (e.g., let a traffic center choose new speed limits arbitrarily often, not just once). Discrete assignment $x := \theta$ instantenously assigns the value of the term θ to the variable x (e.g., let a car choose a particular acceleration), while $x := *$ assigns an arbitrary value to x (e.g., let a car choose any acceleration). $x' = \theta\ \&\ F$ describes a continuous evolution of x within the evolution domain F (e.g., let the velocity of a car change according to its acceleration, but always be greater than zero). The test $?F$ checks that a particular condition expressed by F holds, and aborts if it does not (e.g., check that an arbitrarily chosen acceleration stays within the physical limits of a car because physically impossible accelerations are never considered). Finally, if(F) then α else β is a deterministic choice that executes α if F holds, and β otherwise (e.g., let a traffic center decide upon the position of a car whether or not a speed limit should be issued).

To specify the desired correctness properties of the hybrid programs, differential dynamic logic ($\mathsf{d}\mathcal{L}$) provides modal operators $[\alpha]$ and $\langle\alpha\rangle$, one for each hybrid program α. When ϕ is a $\mathsf{d}\mathcal{L}$ formula (e.g., a simple arithmetic constraint) describing a safe state and α is a hybrid program, then the $\mathsf{d}\mathcal{L}$ formula $[\alpha]\phi$ states that all states reachable by α satisfy ϕ. Dually, formula $\langle\alpha\rangle\phi$ expresses that there is a

Table 1: Statements of hybrid programs

Statement	Effect
$\alpha;\ \beta$	sequential composition, first performs α and then β afterwards
$\alpha\ \cup\ \beta$	nondeterministic choice, following either α or β
α^*	nondeterministic repetition, repeating α $n \geq 0$ times
$x := \theta$	discrete assignment of the value of term θ to variable x (jump)
$x := *$	nondeterministic assignment of an arbitrary real number to x
$\big(x_1' = \theta_1, \ldots,$ $x_n' = \theta_n\ \&\ F\big)$	continuous evolution of x_i along differential equation system $x_i' = \theta_i$, restricted to maximum domain or invariant region F
$?F$	check if formula F holds at current state, abort otherwise
if(F) then α else β	perform α if F holds, perform β otherwise

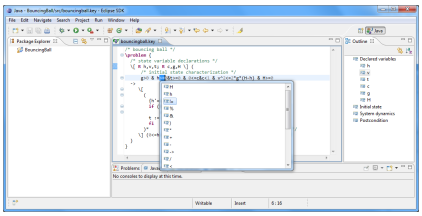
state reachable by the hybrid program α that satisfies formula ϕ. The dL formulas are generated by the following EBNF grammar (where $\sim \in \{<, \leq, =, \geq, >\}$ and θ_1, θ_2 are arithmetic expressions in $+, -, \cdot, /$ over the reals):

$$\phi ::= \theta_1 \sim \theta_2 \mid \neg\phi \mid \phi \wedge \psi \mid \forall x \phi \mid \exists x \phi \mid [\alpha]\phi \mid \langle\alpha\rangle\phi$$

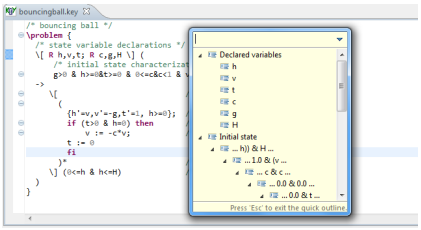
Differential dynamic logic is not only a specification language for hybrid systems (as hybrid programs) and desired correctness properties (as dL formulas), but also comes with a verification technique to prove those correctness properties. The textual modeling environment can load problem specifications expressed in dL into KeYmaera, which is a verification tool that implements Platzer's [28, 29] proof calculus for dL. In the next section, we describe the modeling features of the KeYmaera Eclipse editor.

2.2 The KeYmaera Eclipse Editor

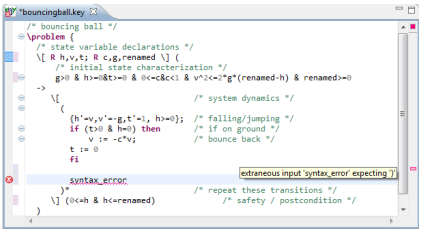
The KeYmaera Eclipse editor supports problem specification file editing with content assists, code folding, and syntax checking. The available content assists, cf. Fig. 2(a), comprises features such as automated code completion, and cross references between variable declarations and variable usages.



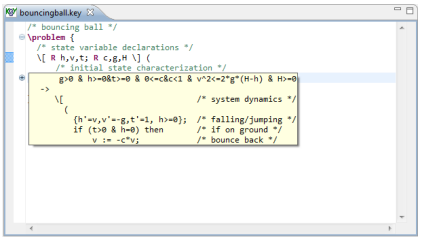
(a) Content assistance, code completion and outline view of a hybrid program



(b) A hierarchical quick outline of the complete problem specification



(c) Syntax checking



(d) Code folding of entire blocks

Fig. 2: Editing capabilities of the KeYmaera Eclipse editor

An outline view, which is in Fig. 2 (a) placed to the right of the text editor, summarizes the main constituents of a hybrid program: declared variables, initial

state description, system dynamics in terms of continuous object behavior and discrete control decisions, and a postcondition defining system invariants. The same outline structure can additionally be superimposed on the text editing view, as shown in Fig. 2 (b). Editing of hybrid programs is supported through syntax checking while typing (cf. Fig. 2 (c)), in order to avoid common mistakes such as mistyped keywords, undeclared variables, as well as missing parentheses and brackets. In large hybrid programs, entire blocks can be folded to save space; the content of currently folded blocks is displayed in a tooltip as shown in Fig. 2 (d).

These and further editor features, such as navigating to the declaration of a variable, opening the quick outline, content assistance, renaming elements, validating the current hybrid program, suggesting quick fixes for syntax problems, finding references to declarations, and running hybrid programs in the KeYmaera prover, are available through the context menu of the KeYmaera Eclipse editor, see Fig. 3.

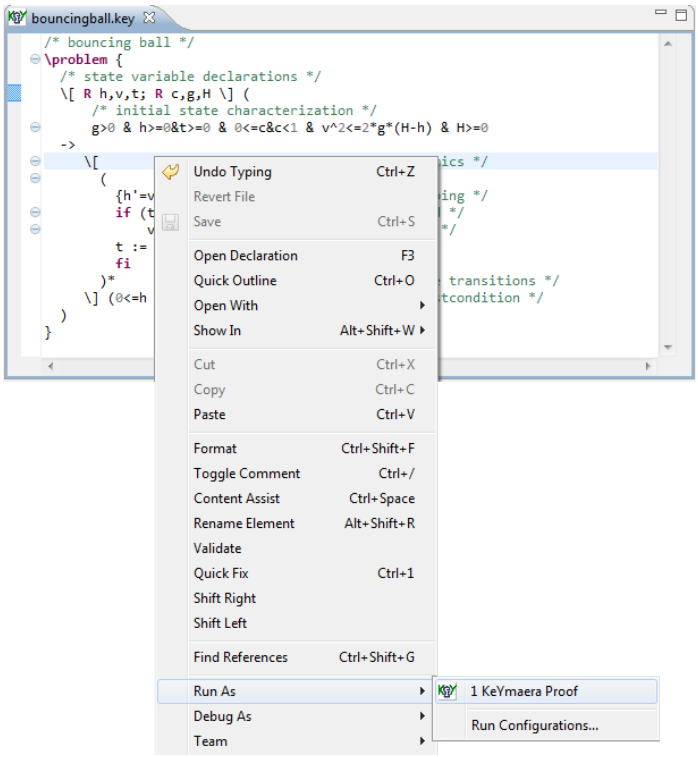
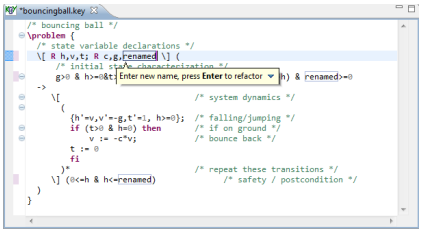


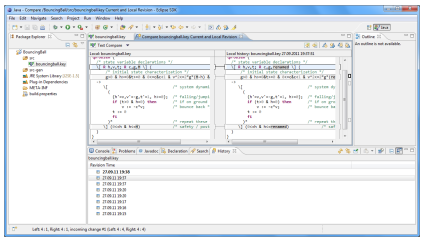
Fig. 3: Features of the KeYmaera Eclipse editor in the context menu

Hybrid program maintenance is supported through consistent renaming of variable declarations and variable usage, see Fig. 4 (a). Eclipse automatically

tracks versions in a local repository, and the KeYmaera Eclipse editor is equipped with a comparison view that highlights differences between the different versions of a single file, and between different files, as shown in Fig. 4b.



(a) Renaming all occurrences of a variable



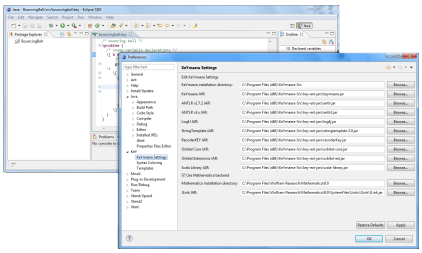
(b) Comparing two problem specification files: in this example, different versions of the same file in the local history are compared

Fig. 4: Refactoring and code browsing in the KeYmaera Eclipse editor

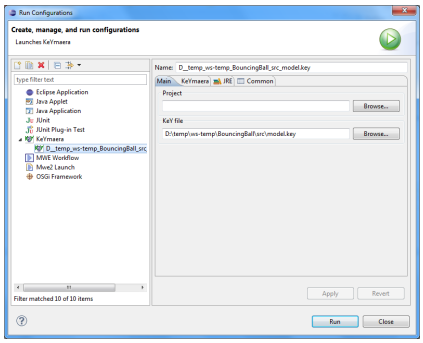
The KeYmaera Eclipse editor can be configured to load hybrid programs and completed proofs from the editor into the KeYmaera prover. As a pre-requisite for this, KeYmaera must be installed separately, and then configured in the KeYmaera Eclipse editor (installation directory and libraries). This configuration can be edited for the complete KeYmaera Eclipse editor in the editor preferences (cf. Fig. 5 (a)), or per loaded proof file on the run configuration page, see Fig. 5 (b).

Each hybrid program or proof file loaded via the run configuration of the KeYmaera Eclipse editor results in a dedicated KeYmaera instance being loaded, see Fig. 5 (c). Every KeYmaera instance is represented with its own console inside the editor. The console view, as any Eclipse console, supports scroll locking, switching between consoles, and forcing the associated KeYmaera instance to shut-down. In order to share a KeYmaera instance between multiple hybrid programs or proof files, as an alternative, the KeYmaera Eclipse editor supports drag-and-drop of hybrid programs and proof files into an already running KeYmaera instance. Previously loaded proofs can be reloaded into KeYmaera with the run history of Eclipse, see Fig. 5 (d).

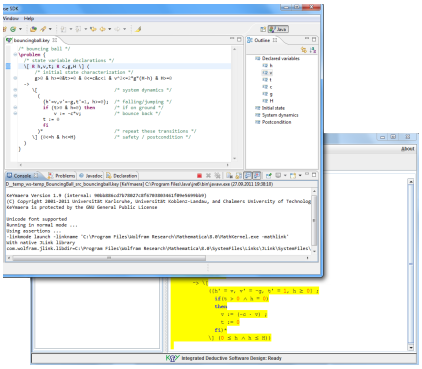
Now that we have seen the modeling of hybrid programs and their execution as proofs in KeYmaera, in the next section, we utilize both $\mathsf{d}\mathcal{L}$ and the textual modeling syntax to specify models of freeway traffic control.



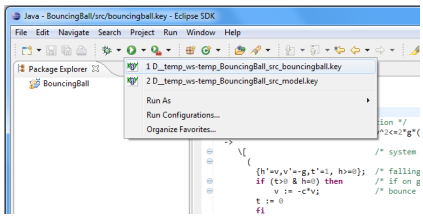
(a) Configuring KeYmaera for multiple runs: installation path and libraries



(b) Configuring a proof run in KeYmaera: project, problem specification, libraries, and KeYmaera installation path



(c) Proof and error output in the KeYmaera console, the KeYmaera prover user interface is depicted in the background. The prover can be stopped at any time through the stop button in the console.



(d) Repeating a previous proof

Fig. 5: Configuring and running KeYmaera from within the KeYmaera Eclipse editor

3 A Case Study in Formal Verification of Traffic Control

In this section, we describe a case study of formal verification of freeway traffic control. A major revision and extension of this section was submitted to the International Conference on Cyber-Physical Systems[3].

Traffic centers have the goal of ensuring global functioning and safety of a freeway or highway network. The available control options comprise, for instance, variable speed limits, ramp metering, lane closures, detours, arterial traffic light control, and warning signs displaying traffic incident information (e.g., traffic jams, construction sites, or driving conditions). A number of theoretical and experimental results have shown that such global highway and freeway traffic control increases safety [2, 21], homogenizes traffic flow [32], and may increase the flow during peak periods [9, 10]. Today's highway and freeway traffic control is centralized in traffic centers (e.g., on a per state level) with little direct influence on the behavior of cars, making it an open-loop control system. Typically, advice to drivers is displayed on dynamic traffic signs mounted on gantries, broadcasted via radio stations, or to GPS navigation systems.

With the advent of more precise and pervasive sensing as well as car-to-car (C2C) and car-to-infrastructure (C2I) communication, a large amount of dynamic traffic information about individual cars becomes available. It is a promising idea to exploit such dynamic traffic information and complement the (geographically) static road infrastructure with dynamic infrastructure-to-car communication. As a result, custom traffic advice could be provided to each car individually, broadcast to all cars in an area, and, in the futre, may even be fed directly as set values into the controllers of (semi-)automatic driver assistance technology in the cars.

At this point, at the latest, the scenario is a prototypical *cyber-physical system* (CPS) case. On the one hand, we find the physics of the movement of a car or a collection of cars down the streets. On the other hand, we have onboard computers embedded in the car and various of its controllers as well as computers in the traffic center that analyze dynamic traffic flow information and support humans with traffic management decisions. In the middle, we find the communication that sends status, traffic, and flow information from the roadside sensor infrastructure and the car GPS's to the traffic centers and the communication that broadcasts, e.g., variable speed limit decisions back to cars and dynamic traffic signs. The CPS is especially interesting when we close the loop and use car information to enhance traffic center decisions and provide traffic center control for individual cars, both connected via C2I communication. The hope is that integrated CPS could direct the fleet of cars more efficiently than a relatively uninformed traffic center without means for direct feedback.

This technology would not be particularly useful if it lead to vastly suboptimal or incorrect control decisions, possibly even endangering safety on the road instead of improving it. For one thing, decision time delays, which may be negligible in more local control scenarios, have a serious impact on the overall CPS

[3] iccps2012.cse.wustl.edu/

dynamics and its behavior over time given the long-range communication and control loop. One particularly interesting challenge to help develop such next generation road traffic control, thus, is the question of how to ensure correct functioning and reliability of such a system. Another challenge is to identify safe margins on the system within which traffic flow can be optimized without endangering safety. First steps towards the verification of safety in road traffic control have been taken by verifying that cars with local adaptive cruise control cannot collide [20]. As a next step, we introduce global control by highway authorities. Our main contribution is a model of a distributed intelligent speed adaptation system and a formal proof that this system correctly disseminates speed limit information and guarantees for cars adhering to the speed limit. For this, we identify constraints on the input and output parameters that car and traffic center controllers need to obey to remain within the safely operable bounds of the system.

These constraints are also relevant in local control loops that replace the traffic center with in-car driver assistance systems, such as traffic sign, pedestrian, or obstacle detectors, picking up control decisions from roadside infrastructure or detecting incidents along the road. The constraints can serve as a basis for precise requirements for driver assistance systems with regard to, for instance, image resolution, focal length of the camera lense, and computation time. In combination with car control, this scenario represents a fully autonomous, active intelligent speed adaptation system [27].

In summary, our contributions are as follows.

1. A model of a distributed intelligent speed adaptation system obtaining speed advice from traffic centers, traffic sign detectors, or obstacle detectors
2. Lower and upper bounds on the position of speed limit area beginnings relative to the position of a car
3. Requirements for the implementation of such systems that directly follow from the bounds

This section is organized as follows. In the next section, we discuss related research concerning global control in traffic centers and local control with in-car driver assistance technology, with a focus on formal verification. In Sect. 3.2 we discuss the challenges in intelligent speed adaptation and, from these, derive the input and output parameters and the general structure of the system. Section 3.3 then presents a model and verification of a lower bound for speed limit choices. Finally, Sect. 3.4 concludes the case study with an outlook on future work.

3.1 Related Work

We discuss related work from the application areas that we focus on: firstly concerning global control in traffic centers from the viewpoint of intelligent speed adaptation, secondly considering advanced driver assistance systems, and, finally, concerning formal verification of traffic control systems.

Lu et al. [22] demonstrate with simulations that higher-level control strategies, such as variable speed limits, can help increase traffic flow and reduce

congestion in bottleneck areas. Dia et al. [13] also used simulation to assess the impact of incident management techniques such as ramp metering, route diversion, and variable speed limits. We verify that such variable speed limits can be disseminated to cars in a safe manner, and that these cars comply with the speed limit at all times. Intelligent speed adaptation [1] and variable speed limit sign systems [2] have increasingly gained attention as a means to increase road traffic safety. Related research in these areas, however,

1. focuses on experiments and simulations of traffic behavior [9, 10] and models for determining optimal speed limits [15],
2. shows the effectiveness of speed adaptation in terms of reducing casualties [2, 26], homogenizing speed [32], increasing compliance with speed limits [1], as well as
3. discusses impacts on travel time and throughput.

We, instead, investigate constraints that implementations of such a system must respect and verify the safety that can be guaranteed under these constraints.

Automated highway control has been the focus, for instance, of the California PATH project (for an overview see [31]), which also investigated the integration of vehicles and roadside infrastructure [23]. Particularly relevant is the work of Ioannou et al. [17] on an integrated roadway/adaptive cruise control system, which was shown to improve travel times and smoothen traffic flow. Similarly, Baskar et al. [5] combined automated vehicle platooning with conventional traffic control in a hierarchy of cooperating controllers with different responsibilities. Again, these works tested safety only partially (mostly using simulation), and do not derive constraints for implementation. Our model is similar, in that it also divides responsibilities between distributed controllers. However, our verification results allow us to derive constraints for the cooperation between higher-level controllers, such as area, regional, or super-regional controllers [5] and highway traffic management control [17] (i.e., traffic centers), and lower-level platoon and vehicle controllers.

With the recent commercialization of advanced driver assistance systems, such as adaptive cruise control, braking assistants, and lane guard systems, also research on traffic sign, crosswalk, and pedestrian detection (cf., for instance, [12, 11, 16, 18, 25]) gained popularity. Such systems are also vital in realizing the vision of fully autonomous vehicles [19]. While investigating detection quality and computation speed, both being undisputedly important characteristics of these systems, none of the works focused on determining the necessary bounds within which such a system can be operated safely.

Verification of safety has been the focus of Loos et al. [20] in their work on adaptive cruise control. Their model focused on mutual safety of cars following each other on a highway. In contrast, we discuss the interplay of cars and roadside infrastructure. Traffic centers disseminating virtual information that may change arbitrarily (i.e., whose positions are not constrained by physical limits and continuity). We, thus, need to find appropriate bounds for traffic center decisions. Moreover, our model allows physical entities on a freeway (e.g., incidents) to move opposite to the driving direction, which was assumed not to

happen in [20]. Movement authorities, which are somewhat similar to speed limits, have been used in verifying the European train control system [30]. They are issued centrally at frequent intervals and trains are not allowed to move without frequent clearance. These results are not applicable to road traffic, because permanent negotiation for movement clearance does not scale to the vast number of cars on a highway (in comparison to the small number of trains on a railroad network). Also, motion was only allowed in accordance with the driving direction of a railroad link. In [3], online verification techniques are presented to derive collision probabilities for autonomous cars. However, deriving bounds and implementation requirements has not been the focus in their work.

3.2 Challenges in Intelligent Speed Adaptation

A typical application of intelligent speed adaptation with variable speed limits is to lower and homogenize speed in the area of traffic incidents [2]. For example, Lu et al. [22] use variable speed limit control to maximize bottleneck flow in an area of a lane drop, such as encountered when lanes merge or lanes are closed due to road work. The nature of traffic incidents in conjunction with the distributed setting of cars and traffic centers, however, poses several challenges on the integration of roadside infrastructure and vehicle control in general, and on the implementation of intelligent speed adaptation systems in particular, as detailed below.

Traffic incidents can not only occur at a geographically fixed position (i.e., be static, such as construction sites), but also change their position (e.g., traffic jams, wrong-way drivers) in the worst case in the opposite direction of traffic on a freeway. Often, motion of traffic incidents can only be approximated using complex models (e.g., shock waves traveling opposite to the driving direction on a freeway [14]). However, estimations about the velocity of incidents can be made, for instance, on the basis of traffic operator experience, or from traffic throughput measurings of induction loop detector arrays.

Nevertheless, the positions and points in time at which incidents occur are completely nondeterministic. As a result, special focus must be laid on the trade off that traffic centers have to make upon occurrence of an incident: as many cars as possible should be warned, which means that speed limits have to be enacted as close as possible to an incident, while at the same time speed limits should only be enacted at safe positions (i.e., at positions that guarantee for cars being able to meet the speed limit).

These matters are even made worse by the fact that the communication delay between a traffic center and a car is non-negligible. During this communication delay, the car moves and the incident changes its position. In order to be effective, variable speed limits must be enacted (from the viewpoint of cars traveling on a freeway) in front of an incident. As a consequence, the traffic center has to estimate a latest speed limit position, which ensures that a car receives its speed limit in any case before it meets an incident.

The promising effects of roadside infrastructure and vehicle integration demonstrated in [17] in terms of better managed traffic with reduced travel times and

smoothened traffic flow, which in turn may lead to improvements in safety and environment, however, make it worthwile to accept these challenges.

The result of this paper is a formally verified model of a straight stretch of highway (which may comprise traffic incidents, such as accidents, construction sites, and traffic jams) controlled by a traffic center and a car following its local control and the variable speed limit issued by the traffic center (i.e., the car does not purposefully violate speed limits). The car has a position, velocity, and acceleration and must obey the laws of physics. The model additionally accounts for sensor and actuator delay within the car, communication between the car and the traffic center, and computation in both. The possible delays caused by communication with central facilities are non-negligible.

Complex maneuvers, such as lane changes, and cars entering and leaving the highway, are not essential for the purpose of our proofs and therefore omitted in the model. It is of utmost importance that the control choices of the car and the traffic center at all times ensure safety of the car, that is, make it possible for the car to meet the speed limit, and safety of the traffic center decision—i.e., set the speed limit at a position on the lane that is between the car and the incident. In the next section, we prove that the speed limit choices of the traffic center can at all times be followed by the car.

3.3 Variable Speed Limit Control

As a first step towards verifying traffic control, the problem that we are solving is: a car on a straight lane can accelerate, coast and brake and we prove that it will not exceed the speed limit set by the traffic center or indicated by a traffic sign detector at any point. This system contains discrete and continuous dynamics, thus it is a hybrid system. Alone, the necessity for issuing a speed limit may arise at any time. As a consequence, both the traffic center and the traffic sign detector can repeatedly issue new speed limits—comprising a maximum speed and a position denoting the speed limit area—or decide to stick with already set ones. Newly issued speed limits are communicated to the car controller (in the case of the traffic center, e.g., wirelessly or via conventional roadside infrastructure), which, anyway, takes time. In the meantime, of course, the car's position evolves according to its velocity and acceleration. As a consequence, the traffic center must take into account the car's position, velocity, and acceleration, and the time needed for communicating to and processing the decision in the car when choosing the maximum speed and position of a speed limit area, in order to avoid issuing speed limits that cannot be complied with (e.g., we cannot demand the car to brake from 30 m/s to 20 m/s within 1 m). Likewise, a traffic sign detector must be able to correctly recognize a speed limit sign at a distance depending on the car's velocity and acceleration, and the time needed for processing the speed limit sign image, communicating to and processing the speed limit in the car controller. At the cost of more conservative decisions and distance/processing bounds, this information demand can be relaxed by assuming generic maximum values for velocity, acceleration, and communication and processing time.

Here, we abstract from the details of traffic centers and traffic sign detectors by modeling the relevant characteristics of the decisions both have to make (i.e., the maximum speed allowed in and the geographical position of the speed limit area), as described in the following paragraph. Formal verification of the model guarantees safety in all considered situations. This allows us to derive from the model the bounds for (i) maximum speed and geographical position relevant for the traffic center, and (ii) distance and processing time of a traffic sign detector. Since we use individual speed limits for each car (realized with direct communication between traffic centers and cars, or with traffic sign detectors inside the car), we can safely simplify our model to a single car.

Modeling Based on the adaptive cruise control model of Loos et al. [20], we develop a formal model of a distributed intelligent speed adaptation system as a hybrid program (HP). The car has state variables describing its current position (x_c), velocity (v_c), and acceleration (a_c). The continuous dynamics of the car is described by the differential equation system of ideal-world dynamics for longitudinal position changes ($x_c' = v_c, v_c' = a_c$). We assume bounds for acceleration a_c in terms of a maximum acceleration $A \geq 0$ and a minimum positive braking power $b > 0$. We introduce a constant ε that provides an upper bound for sensor and actuator delay, communication between the traffic center or traffic sign detector and the car controller, and computation in both. The car controller[4] and the traffic center may react and exchange messages as quickly as they want, but they can take no longer than ε.

The car is allowed to brake at all times through (3) having no precondition, which is also the only option if there is not enough distance between the car and the speed limit area to maintain speed or accelerate. If the car is still at a safe distance from the speed limit area, it may choose its acceleration freely within the bounds of its braking power and acceleration, cf. (4). Safety of the car is given when (7) and (8) are satisfied, i.e., if the car can drive up to ε time units with any choice of acceleration, and still adhere to the speed limit. For this, the distance between the car's current position x_c and the beginning of the speed limit area x_{sl} must account for two components: first, the car may need to brake from its current velocity v_c down to v_{sl}, and in the course of this travel the distance given in (7). Second, since the car may not notice the speed limit up to ε time units, we must additionally take into account the distance that the car may travel with its current velocity and worst-case acceleration A and the distance needed for compensating its potential acceleration of A during that time with braking power b, see (8). In the speed limit area the car may choose its acceleration within its physical limits and depending on the current velocity difference to the speed limit, see (5). Note that for the implementation of a car controller that computes its acceleration only on the basis of the physical boundaries of the car (i.e., A and b), the additional restriction $a_c \leq \frac{v_{sl}-v_c}{\varepsilon}$ can be used as a precondition using maximum acceleration $v_c + A \cdot \varepsilon \leq v_{sl}$. Finally,

[4] Note that the car controller may also be a human driver, in which case the processing time in the car is mostly attributed to the reaction time of the driver.

Model 1 Variable speed limit control (vsl)

$$vsl \equiv (ctrl; dyn)^* \tag{1}$$
$$ctrl \equiv ctrl_{car} || ctrl_{ctr}; \tag{2}$$
$$ctrl_{car} \equiv (a_c := -b) \tag{3}$$
$$\cup \left(?Safe_{\underline{x_{sl}}}; a_c := *; ?(-b \leq a_c \leq A)\right) \tag{4}$$
$$\cup \left(?x_c \geq x_{sl}; a_c := *; ?(-b \leq a_c \leq A \wedge a_c \leq \frac{v_{sl} - v_c}{\varepsilon})\right) \tag{5}$$
$$\cup (?v_c = 0; a_c := 0) \tag{6}$$
$$Safe_{\underline{sl}} \equiv x_c + \frac{v_c^2 - v_{sl}^2}{2 \cdot b} \tag{7}$$
$$+ \left(\frac{A}{b} + 1\right) \cdot \left(\frac{A}{2} \cdot \varepsilon^2 + \varepsilon \cdot v_c\right) \leq x_{sl} \tag{8}$$
$$ctrl_{ctr} \equiv (x_{sl} := x_{sl}; v_{sl} := v_{sl}) \tag{9}$$
$$\cup \left(x_{sl} := *; v_{sl} := *; ?(v_{sl} \geq 0 \wedge Safe_{\underline{sl}})\right) \tag{10}$$
$$dyn \equiv (t := 0; x_c' = v_c, v_c' = a_c, t' = 1 \tag{11}$$
$$\& v_c \geq 0 \wedge t \leq \varepsilon) \tag{12}$$

the car may choose to stand still if its current velocity is zero already (6), since the continuous dynamics, in accordance with freeway traffic rules, do not allow velocities below zero (i.e., driving opposite to the driving direction on a freeway is prohibited).

The traffic center may choose to keep a current speed limit, cf. (9), or set a new speed limit v_{sl} (which, of course, must not force the car to drive backwards) at a new, safe position x_{sl}; see (10). This safe position guarantees, that the car is still able to meet the speed limit even if it does not receive and cannot react on the new speed limit for up to ε time units. For making this decision, it is essential that the controller in the traffic center knows or can estimate the current position, velocity, braking and acceleration capabilities of the car, and the time needed for reaction. Note that it is only mandatory to communicate the current position of the car (allowing, of course, some inaccuracy) to the traffic center. At the expense of a less stringent speed limit area (i.e., the safety distance may be larger than absolutely necessary), worst case estimations can be used for all other values (e.g., general highway speed limits, typical car acceleration, and minimum braking power demanded by law).

Car and control center can repeatedly choose acceleration and speed limit, respectively, which is represented by the nondeterministic repetition operator * in (1). The controllers of the car and the traffic center operate in parallel, cf. (2). Since the controllers are independent with respect to their read and write variables, the parallel operation can also be modeled using a sequential composition. The order of components in a sequential composition is significant:

we model the control of the car followed by the control of the traffic center, and finally the dynamics of the system. As a result, the system evolves before the traffic center decisions reach the car controller at the next iteration, which models communication delay between the traffic center and the car.

The continuous dynamics (11) of the model describe the evolution of the car's position and velocity according to the current acceleration. We use a variable t that evolves with constant slope (i.e., a clock) for measuring time within the upper bound ε, and constrain the evolution of velocity v_c to non-negative values, see (12).

Listing 1.1 in Appendix A shows the hybrid program notation of Model 1 in the syntax of the textual KeYmaera Eclipse editor.

Verification We verify the safety of a speed limit choice as modeled above, using a formal proof calculus for $\mathsf{d}\mathcal{L}$ [28, 29]. In this use case, the car must comply with the speed limit inside a speed limit area at all times. The following condition captures this requirement as an invariant that must hold at all times during the execution of the model:

$$c \searrow sl \equiv \left(v_c \leq v_{sl} \vee x_{sl} \geq x_c + \frac{v_c^2 - v_{sl}^2}{2 \cdot b} \right) \wedge v_c \geq 0 \wedge v_{sl} \geq 0$$

The formula states that a speed limit chosen by the traffic center or detected by the traffic sign detector can be complied with when the car's current velocity is already less or equal to the speed limit, or there is still enough distance for the car to brake (and the car must drive forward, and the speed limit not demand driving backwards).

Proposition 1 (Safety of speed limit). *If a car is at a safe distance from x_{sl} initially, then it will not exceed the speed limit past the beginning of a speed limit area while the car controller and the traffic center or traffic sign detector follow the* `vsl` *control model. Compliance with the speed limit is expressed by the provable formula* $(c \searrow sl) \rightarrow [vsl](x_c \geq x_{sl} \rightarrow v_c \leq v_{sl})$

We proved Proposition 1 using KeYmaera, a theorem prover for hybrid systems. The resulting proof files are available online as projects in KeYmaera[5].

Safe bounds The condition $Safe_{\underline{sl}}$, see (7), provides bounds on the minimum distance of a speed limit area to the current position of a car (assuming a certain maximum speed), as well as the maximum speed (assuming a certain minimum distance) of a variable speed limit area. Concerning a traffic sign detector, the worst case minimum distance that a car needs in order to comply with a speed limit is most interesting. This worst case minimum distance, at which a traffic sign detector must be able to identify a speed limit sign at the latest, is given through (13).

[5] `http://symbolaris.com/info/KeYmaera.html`

$$xsl - x_c \geq \frac{v_c^2 - v_{sl}^2}{2 \cdot b} + \left(\frac{A}{b} + 1\right) \cdot \left(\frac{A}{2} \cdot \varepsilon^2 + \varepsilon \cdot v_c\right) \tag{13}$$

For instance, with a current velocity of 60 km/h, typical values for maximum acceleration (4 m/s^2) and maximum braking power (9 m/s^2), and assuming a speed limit sign that shows 50 km/h, computation time of 50 ms and another 50 ms for communication and reaction, the distance at which the traffic sign detector must start at the latest is about 8 m from the traffic sign. When applying a comfortable braking power of only 2 m/s^2 [4], the distance grows to over 26 m. Taking a look at the camera and resolution used by Deguchi et al. [12], a speed limit sign of 0.5 m width in a distance of 26 m would be represented in the resulting image with a width of 12 pixels[6], which is below the 15–45 pixels image width used in their evaluation. This is an example where formal analysis can be used to infer design decisions of CPS.

3.4 Conclusion and Future Work

Traffic centers focus on the global functioning of a freeway or highway network and, for this, impose dynamic constraints (e.g., variable speed limits) on the control choices of car controllers. At the same time, sensor and driver assistance systems make cars increasingly aware of their environment, and enable them to react autonomously (e.g., adaptive cruise control, or obstacle detection that initiates emergency braking). It is a promising idea to combine global traffic control choices—which could be communicated directly from a traffic center to a car, or sensed by driver assistance systems—and car control into a fully autonomous system. Yet, such a combination is only economically feasible without costly post-deploy upgrades or even possible hazards when its safety can be ensured.

In this work, we presented a distributed intelligent speed adaptation system comprising a car controller and a speed limit controller (e.g., a traffic center or a driver assistance system) with direct communication in-between. We presented formal verification results that guarantee safe operation of cars (i.e., cars always comply with speed limits). In the process of verification, we found important invariants, which are needed to ensure such safe operation if implemented in actual physical controllers. These invariants comprise bounds on the distance between cars and speed limit areas. They can be further transformed into precise requirements and help decide trade-offs even for design parameters of traffic centers and driver assistance systems that are not modeled formally (e.g., image resolution, focal length, and computation time of driver assistance systems).

Future work includes addressing arbitrarily many incidents, and homogenizing speed with consecutively arranged speed limits of decreasing maximum speed. Also, the models discussed in this paper will be further refined by introducing explicit communication channels, which allows multiple control decisions during one communication roundtrip.

[6] Given a chip width (w_{chip}) and focal length (l_{focal}) of both 63 mm and 640 pixels of horizontal image width (w_{image}), using $res = w_{image}/(d \cdot w_{chip}/l_{focal})$, we get a resolution of 24 pixels/m for an object at a distance (d) of 26 m.

Acknowledgments

This work has been partly funded by Marshall-Plan Foundation. This material is based upon work supported by the National Science Foundation under NSF CAREER Award CNS-1054246 and NSF EXPEDITION CNS-0926181 as well as Grant Nos. CNS-1035800, and CNS-0931985.

References

1. N. Agerholm, R. Waagepetersen, N. Tradisauskas, and H. Lahrmann. Intelligent speed adaptation in company vehicles. In *Proceedings of the IEEE Intelligent Vehicles Symposium*, pages 936 –943. IEEE, 2008.
2. P. Allaby, B. Hellinga, and M. Bullock. Variable Speed Limits: Safety and Operational Impacts of a Candidate Control Strategy for Freeway Applications. *IEEE Transactions on Intelligent Transportation Systems*, 8(4):671 –680, 2007.
3. M. Althoff, O. Stursberg, and M. Buss. Safety assessment of autonomous cars using verification techniques. In *Proceedings of the American Control Conference (ACC)*, pages 4154–4159, 2007.
4. G. Anagnostopoulos, M. Coltman, and R. Suever. Compendium of executive summaries from the maglev system concept definition final reports. Technical Report DOT/FRA/NMI-93/02, U.S. Department of Transportation, 1993.
5. L. Baskar, B. De Schutter, and H. Hellendoorn. Dynamic speed limits and on-ramp metering for ivhs using model predictive control. In *Proceedings of the 11th International IEEE Conference on Intelligent Transportation Systems (ITSC)*, pages 821–826. IEEE, 2008.
6. N. Baumgartner, W. Gottesheim, S. Mitsch, W. Retschitzegger, and W. Schwinger. On Optimization of Predictions in Ontology-Driven Situation Awareness. In *Proceedings of the 3rd International Conference on Knowledge Science, Engineering and Management (KSEM)*, pages 297–309. Springer, 2009.
7. N. Baumgartner, W. Gottesheim, S. Mitsch, W. Retschitzegger, and W. Schwinger. BeAware!—Situation Awareness, the Ontology-driven Way. *International Journal of Data and Knowledge Engineering*, 69(11):1181–1193, 2010.
8. N. Baumgartner, W. Gottesheim, S. Mitsch, W. Retschitzegger, and W. Schwinger. Situation Prediction Nets—Playing the Token Game for Ontology-Driven Situation Awareness. In *Proceedings of the 29th International Conference on Conceptual Modeling (ER)*, pages 202–218. Springer, 2010.
9. R. Bertini, S. Boice, and K. Bogenberger. Using ITS data fusion to examine traffic dynamics on a freeway with variable speed limits. In *Proceedings of Intelligent Transportation Systems*, pages 1006–1011. IEEE, 2005.
10. R. L. Bertini, S. Boice, and K. Bogenberger. Dynamics of variable speed limit system surrounding bottleneck on german autobahn. *Transportation Research Record: Journal of the Transportation Research Board*, pages 149–159, 2006.
11. A. Broggi, P. Cerri, S. Ghidoni, P. Grisleri, and H. G. Jung. A new approach to urban pedestrian detection for automatic braking. *Intelligent Transportation Systems, IEEE Transactions on*, 10(4):594 –605, dec 2009.
12. D. Deguchi, M. Shirasuna, K. Doman, I. Ide, and H. Murase. Intelligent traffic sign detector: Adaptive learning based on online gathering of training samples. In *Intelligent Vehicles Symposium (IV), 2011 IEEE*, pages 72–77, 2011.

13. H. Dia, W. Gondwe, and S. Panwai. Traffic impact assessment of incident management strategies. In *Intelligent Transportation Systems, 2008. ITSC 2008. 11th International IEEE Conference on*, pages 441–446, 2008.
14. M. R. Flynn, A. R. Kasimov, J.-C. Nave, R. R. Rosales, and B. Seibold. Self-sustained nonlinear waves in traffic flow. *Physical Review E*, 79:056113, May 2009.
15. R. Gallen, N. Hautiere, and S. Glaser. Advisory speed for intelligent speed adaptation in adverse conditions. In *Intelligent Vehicles Symposium (IV), 2010 IEEE*, pages 107 –114, 2010.
16. A. Haselhoff and A. Kummert. On visual crosswalk detection for driver assistance systems. In *Intelligent Vehicles Symposium (IV), 2010 IEEE*, pages 883–888, 2010.
17. P. Ioannou, Y. Wang, and H. Chang. Integrated roadway/adaptive cruise control system: Safety, performance, environmental and near term deployment considerations. Technical Report UCB-ITS-PRR-2007-08, California PATH program, Institute of transportation studies, University of California, Berkeley, 2007.
18. R. Kastner, T. Michalke, T. Burbach, J. Fritsch, and C. Goerick. Attention-based traffic sign recognition with an array of weak classifiers. In *Proceedings of the 2010 Intelligent Vehicles Symposium (IV)*, pages 333 –339. IEEE, 2010.
19. J. Levinson, J. Askeland, J. Becker, J. Dolson, D. Held, S. Kammel, J. Kolter, D. Langer, O. Pink, V. Pratt, M. Sokolsky, G. Stanek, D. Stavens, A. Teichman, M. Werling, and S. Thrun. Towards fully autonomous driving: Systems and algorithms. In *Intelligent Vehicles Symposium (IV), 2011 IEEE*, pages 163–168, 2011.
20. S. M. Loos, A. Platzer, and L. Nistor. Adaptive cruise control: Hybrid, distributed, and now formally verified. In M. Butler and W. Schulte, editors, *17th International Symposium on Formal Methods (FM)*, volume 6664 of *LNCS*, pages 42–56. Springer, 2011.
21. F. Lu and X. Chen. Analyzing the speed dispersion influence on traffic safety. In *International Conference on Measuring Technology and Mechatronics Automation (ICMTMA)*, volume 3, pages 482–485. IEEE, 2009.
22. X.-Y. Lu, P. Varaiya, R. Horowitz, D. Su, and S. Shladover. A new approach for combined freeway variable speed limits and coordinated ramp metering. In *Intelligent Transportation Systems (ITSC), 2010 13th International IEEE Conference on*, pages 491 –498. IEEE, 2010.
23. J. Misener and S. Shladover. Path investigations in vehicle-roadside cooperation and safety: A foundation for safety and vehicle-infrastructure integration research. In *Proceedings of the Intelligent Transportation Systems Conference (ITSC)*, pages 9–16. IEEE, 2006.
24. W. Mostowski. The key syntax. In B. Beckert, R. Hhnle, and P. Schmitt, editors, *Verification of Object-Oriented Software. The KeY Approach*, volume 4334 of *Lecture Notes in Computer Science*, pages 599–626. Springer, 2007.
25. L. Oliveira and U. Nunes. Context-aware pedestrian detection using lidar. In *Intelligent Vehicles Symposium (IV), 2010 IEEE*, pages 773–778, 2010.
26. M. Paine, D. Paine, and I. J. Faulks. Speed Limiting Trials in Australia. In *Proceedings of the 21st International Technical Conference on the Enhanced Safety of Vehicles*, 2009.
27. M. Paine, D. Paine, M. Griffiths, and G. Germanos. In-vehicle Intelligent Speed Advisory Systems. In *Proceedings of the 20th International Conference on the Enhanced Safety of Vehicles*, 2007.
28. A. Platzer. Differential dynamic logic for hybrid systems. *J. Autom. Reas.*, 41(2):143–189, 2008.

29. A. Platzer. *Logical Analysis of Hybrid Systems: Proving Theorems for Complex Dynamics*. Springer, Heidelberg, 2010.
30. A. Platzer and J.-D. Quesel. European Train Control System: A case study in formal verification. In K. Breitman and A. Cavalcanti, editors, *ICFEM*, volume 5885 of *LNCS*, pages 246–265. Springer, 2009.
31. S. Shladover. PATH at 20—history and major milestones. *Transactions on Intelligent Transportation Systems*, 8(4):584–592, 2007.
32. E. van den Hoogen and S. Smulders. Control by variable speed signs: results of the Dutch experiment. In *7th International Conference on Road Traffic Monitoring and Control*, pages 145–149. IEEE, 1994.

A Variable Speed Limit Control as a Hybrid Program

Listing 1.1: Variable speed limit

```
\problem {
\[
  /* variable declarations */
  R xc, vc, ac, t;  /* car 1 */
  R vsl, xsl;         /* traffic center */
  R B, A, ep;       /* system parameters */
\]
  (  vc >= 0
   & vsl >= 0
   & xc <= xsl
   & 2 * B * (xsl - xc) >= vc^2 - vsl^2
   & A >= 0
   & B > 0
   & ep > 0
    -> \[(
           /* control car */
           /* braking is always allowed */
           (ac := -B)
           /* outside the speed limit, ensure that the car
               can still brake to meet the speed limit */
           ++ (?xsl >= xc + (vc^2 - vsl^2) / (2 * B) + (A / B
                + 1) * (A / 2 * ep^2 + ep * vc);
               ac := *;  ?-B <= ac & ac <= A)
           /* comply with the speed limit by not accelerating
                too much */
           ++ (?xc >= xsl; ac := *; ?-B <= ac & ac <= A & ac
               <= (vc - vsl) / ep);

           /* traffic center */
           /* keep previous speed limit */
           (xsl := xsl; vsl := vsl)
           /* or set a new speed limit */
           ++ (xsl := *; vsl := *; ?vsl >= 0 & xsl >= xc + (
               vc^2 - vsl^2) / (2 * B) + (A / B + 1) * (A / 2
                * ep^2 + ep * vc));

           t := 0;
           /* dynamics */
           {xc' = vc, vc' = ac, t' = 1, vc >= 0, t <= ep}
      )* /* repeat non-deterministically */
      @invariant(vc >= 0 & vsl >= 0 & (vc <= vsl | xsl >= xc
          + (vc^2 - vsl^2) / (2 * B)))
    \] (xc >= xsl -> vc <= vsl)
  )
} /* end problem */
```