Safe Intersections: At the Crossing of Hybrid Systems and Verification

Sarah M. Loos and André Platzer

Abstract— **Intelligent vehicle systems have interesting prospects for solving inefficiencies and risks in ground transportation, e.g., by making cars aware of their environment and regulating speed intelligently. If the computer control technology reacts fast enough, intelligent control can be used to increase the density of cars on the streets. The technology may also help prevent crashes at intersections, which cost the US $97 Billion in the year 2000. The crucial prerequisite for intelligent vehicle control, however, is that it must be correct, for it may otherwise do more harm than good. Formal verification techniques provide the best reliability guarantees but have had difficulties in the past with scaling to such complex systems. We report our successes with a logical approach to hybrid systems verification, which can capture discrete control decisions and continuous driving dynamics. We present a model for the interaction of two cars and a traffic light at a two lane intersection and verify with a formal proof that our system always ensures collision freedom and that our controller always prevents cars from running red lights.**

I. INTRODUCTION

The status quo in individual ground transportation is neither efficient nor particularly safe. In various places, crowded roads can hardly meet the demand, which causes long delays. Even more troublesome than delayed traffic are automobile accidents. Per year, $500 Billion total cost result from car crashes in the world, a large part of which is caused by crashes at intersections, e.g., $97 Billion in the year 2000 just in the US [1]. To fight both accidents and inefficiencies, intelligent vehicle systems have been studied for a while now [1–18]. Major initiatives include the California PATH project, SAFESPOT, PReVENT, the CICAS-V system, and many others. Chang et al. [1], for instance, propose CICAS-V specifically to prevent violations at intersections.

One big concern for all systems where computers with physical capabilities (like vehicle control) interface with humans is how their reliability can be ensured. Even if the designer plans the system with the best and most careful intentions, an intelligent transportation system could still cause a lot of damage if it has subtle safety bugs.

Testing is invaluable as a sanity check for system designs. But even exhaustive testing still will not find more subtle bugs, a problem that is confounded by the increasingly tricky interaction of multiple features in system designs.

Formal verification techniques have been very successful in many other domains and have found subtle bugs and verified the correctness of difficult systems, e.g., computer chips and software. Formal system modeling and verification has also been used for some aspects of ground transportation [2, 12, 17, 19, 20]. What makes the verification of ground transportation systems more challenging, though, is the fact that the software running on the computers cannot be considered in isolation. Instead, their effect on the physical behavior of the cars they are controlling has to be taken into account.

We model intelligent ground vehicles as hybrid systems, i.e., systems with interacting discrete dynamics (for control decisions) and continuous dynamics (for movement). Intelligent transportation scenarios may be distributed hybrid systems [21] when there is a multi-agent situation with multiple cars driving according to their hybrid system dynamics. We have developed logical verification techniques for these hybrid systems [22–24] and distributed hybrid systems [21] and have successfully used them to verify collision freedom of multi-agent highway control [19].

In this paper, we consider the problem of controlling cars and traffic lights to ensure collision freedom at intersections. Naïvely, this may appear to be a simple problem where all we have to do is to ensure that at most one of the traffic lights at an intersection is red. But this does not work, because the overall system would still be unsafe if the car controllers disobey the red lights or if the traffic lights switch in a way that the car controllers have no way of complying with. We develop a controller for a stoplight intersection, identify the crucial safety constraints, and formally verify that collision freedom is guaranteed.

II. RELATED WORK

Major initiatives have been devoted to developing safe next-generation automated car control systems, including the California PATH project, the SAFESPOT and PReVENT initiatives, the CICAS-V system, and many others. With the exception of [19], safety verification for car control systems has been for specific maneuvers or systems with a small number of cars.

Stursberg et al. [12] applied counterexample-guided verification to a cruise control system with two cars on one lane, and Althoff et al. [17] used reachability analysis to prove the safety of evasive maneuvers of autonomous vehicles. Damm et al. [2] give a verification rule that is specialized to collision freedom of traffic agents. To show that two cars do not collide, they must manually prove eighteen verification conditions. Lygeros and Lynch [20] prove safety only for one deceleration strategy for a string of vehicles: the leading vehicle applies maximum deceleration until it stops, while simultaneously all cars following it in the string decelerate to a stop. The previously mentioned projects verify safety

This material is based upon work supported by the National Science Foundation under NSF CAREER Award CNS-1054246 and NSF Grant No. CNS-0926181, CNS-0931985, CNS-1035800, and CNS-1035813. The first author was also supported by an NSF Graduate Research Fellowship.

Computer Science Department, Carnegie Mellon University, Pittsburgh, PA 15213, USA {sloos|aplatzer}@cs.cmu.edu

for specific maneuvers on the lane which do not scale up to a global model. Our work in car control on lanes scales to an arbitrarily large number of cars in the system. Additionally, our work on intersections is created with the primary goal of being combined with other verification results to be applied to larger and more complex systems. Our logic-based approach supports such combinations.

Research has also been done on the best action a car can take at a yellow light [25]. While this research does not provide a formal proof of safety, it does suggest a more efficient vehicle controller for non-sensing stoplights. Other projects have considered more general systems using simulation and other non-formal methods [4, 6, 9, 15]. Our techniques follow a formal, mechanized proof calculus, which verifies safety completely, rather than partially via a finite number of simulations. Our use of differential equations to represent the continuous dynamics of the vehicles enables a more realistic model than finite state models which cover only discrete dynamics.

III. THE INTERSECTION PROBLEM

In [19], we verified safety for a model of a multi-lane highway, with an arbitrary number of cars, each of which can change lanes, exit and enter the system. The research presented in this paper is the first step toward verifying urban roadways with large and complex intersections.

In order to verify a large, urban, roadway system, we first must tackle intersections. Difficult intersections would be nearly impossible to verify without breaking them down into smaller pieces. Even if you did manage to verify the safety of an automated control or emergency braking system for one complex intersection without decomposing it into smaller pieces, it would be difficult to generalize the method, so another intersection will require the same amount of work. In this paper, we prove safety for the basic principles, which is a major milestone in making the verification of complex intersections feasible. Our first insight into these difficult systems is that most intersections, including many that are large and complex, are just a compilation of two basic lane layouts: merges and crosses, shown in Fig. 1.

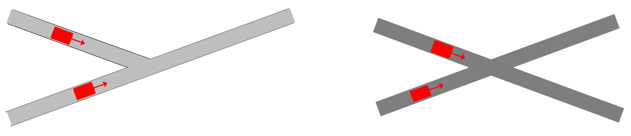
(a) T-intersection/merge. (b) Two lane intersection.

Fig. 1: Basic components of most intersections (*Note: The angle of intersection is irrelevant*)

In [19], we verified that merging from one lane into another is safe. The T-intersection shown in Fig. 1a is a special case of this merging scenario. We begin with two lanes, and have one of the lanes end at the intersection, so that all cars on that lane must switch into the other lane. This means that one lane is identified as the primary lane, and all cars merging from the secondary lane may only do so when the primary lane is clear. Thus, for a proof of safety, a traffic signal is not required, but may still be used.

While we ultimately want to verify a generic model for distributed, automated car control in a large urban environment with complex traffic patterns, the scope of this paper is smaller. In this paper, we present the complexities and challenges of modeling and verifying an intersection of two lanes, with one car driving on each lane. In future work, we intend to combine the results in this paper with the results in [19] to verify an intersection of two lanes with an arbitrary number of cars on each lane, each of which makes distributed control decisions based on local Vehicle to Vehicle (V2V) and Vehicle to Infrastructure (V2I) data. Combining this result with the safe merge maneuver will then allow us to verify large and complex intersections.

We begin in Sect. IV with a comprehensive review of the multi-lane highway problem for distributed car control. Then, in Sect. V, we simplify the two lane intersection into a single lane with just one car and one light. From the initial conditions and invariants used to verify this model, we discover a method for proving the more complex model presented in Sect. VI. We present in Sect. VI the safety of a two lane intersection, with one car driving on each lane and a stoplight operating at the crossing.

IV. HIGHWAY CONTROL

In [19] we present a formal verification for a system with an arbitrary number of cars driving on an arbitrary number of lanes, where cars can enter the highway, change lanes, and exit. In this section, we will summarize this work to give a solid background for research on intersection control. We will also discuss how this result can be used to verify T-intersections as shown in Fig. 1a.

A. Modeling

In our model of a highway system, the control for the cars is very intuitive. Cars may brake at any time, but they can only accelerate if the next car is a safe distance ahead. For lane change maneuvers, cars may only change into a lane if they leave enough space in front and behind.

The discrete control is simple enough to explain, but the difficulty comes in the verification, which relies on identifying strong enough invariants to prove safety properties which hold under all physical movements of the car.

It is the physical driving of the car down the lane that produces the continuous components in this hybrid system. We use state variables to represent the car's position, velocity, and acceleration. The continuous dynamics for the kinematic motion of the car are described by the following differential equation system: $x' = v$, $v' = a$. These are ideal-world dynamics and they are adequate for longitudinal lane maneuvers. Because sensor readings are not available continuously, we assume that the controller also does not have continuous control over acceleration. For simplicity, we still assume that, once set, the acceleration a takes instant effect. We assume a limit for the maximum acceleration and we denote it by $A \geq 0$. We assume that the car has an emergency brake with braking power $-B$, where $B > 0$. If B were not strictly positive, the car would not be able to brake. The car is also limited in its velocity: we assume a maximum speed limit, denoted by V, where $V > 0$.

We model a T-intersection as a specific case of this system: two lanes, where one lane ends immediately after the intersection, forcing all cars from that lane to merge into the continuing, primary lane. This T-intersection model inherits the safety verification from the highway model. It also requires that a primary lane be identified, since one lane will have traffic moving freely, and the other will end and require cars to merge into traffic on the main lane. This is similar to a T-intersection when taken as a component to a more complex intersection, since in most cases, cars from one lane will be turning into traffic on the other lane; however, it does not rely on a light control to mediate the traffic flow, but instead uses local sensors on individual cars.

B. *Verification*

In order to show that a system of cars driving on the highway is safe, we must prove that, under any circumstances, no car collides with another car. The verification of this safety requirement is separated into four, hierarchical pieces, starting with a proof of safety for two cars on a lane and working up to the full model. For a complete discussion of the highway model and its formal verification, see [19].

V. SINGLE LANE STOPLIGHT CONTROL

In this section, we study a model of a single car on a lane with a stoplight. In actuality, there would be a crossing lane at the light, with cars moving along it, but initially we will look at a single lane, and later examine the intersection in Sect. VI. The continuous dynamics of the physical movement of the car will be the same as for cars driving on a highway, as in Sect. IV; however, our safety requirements have changed. Instead of proving that the car will not hit another car in its lane, we prove that the car will not, under any circumstances, occupy an intersection while the light is red. This is an important building block for subsequent systems.

A. *Modeling*

In this model, we represent the lanes as lines, where cars and intersections appear as points on those lines. Alternatively, we could modify the equations to include a constant buffer to account for the size of the car, or the width of a lane. This is interesting future work, and will be necessary for verifying large-scale urban systems, but is beyond the scope of this paper. By ignoring the size of cars, we reduce the number of parameters in the system while retaining the essential characteristics of intersections: mutual exclusion in the spatial region of traffic lights.

The stoplight will need to coordinate with the car driving on the lane. For instance, if the car was driving at high speed, and the light quickly switched from green, through a short yellow, to red, it is possible that the car would be unable to avoid running the red light. We cannot assume continuous sensing and communication of the car's position to the traffic light, because that would be impossible to implement. Pseudo-periodic information about positions and velocities is more realistic. In this model, we, thus, assume that the stoplight is receiving updates about the car's position and velocity at least every ε time units (possibly more often but never less often). Similarly, the car receives updates about the color of the light at least every ε time units.

In our model, we allow the light to change from green to yellow under any circumstances. The light may also change from red to green at any time (although when we have cars traveling on an intersecting lane, this will only be allowed if the stoplight controlling the intersecting lane is also red). However, the light may only change from yellow to red under certain properties, since we must ensure that the car is always physically capable of adhering to red stoplight signals. It must always hold that when the light turns yellow, it stays yellow long enough for the car to detect the yellow light and then either come to a stop before the intersection, or pass through the intersection. This requirement is equivalent to the formula:

$$xI < x \vee xI > x + \frac{v^2}{2B} + \left(\frac{A}{B} + 1\right)\left(\frac{A}{2}\varepsilon^2 + v\varepsilon\right).$$

We denote the position of the stoplight by xI, time delay as ε, and remaining variables are as described in Sect. IV. This formula characterizes that the light may only turn red if either the car has passed the light ($xI < x$), or the car would be able to stop before the intersection. This should hold even in the worst case, where the car does not recognize the yellow light for a maximum of ε time, and is applying maximum acceleration A for that duration.

Just as the stoplight receives communication updates on the position and velocity of the car, similarly the car receives updates on the color of the light. These updates may not come regularly, but we assume all data is accurate upon delivery. Uncertainty in the sensor data could be handled by assuming upper and lower bounds on accuracy, or by using Stochastic Differential Dynamic Logic [26].

Since the maximum velocity of the car is V, we could replace the use of state variables with maximum values and require only that the stoplight activate a constant-length timer for a yellow light. However, while this would make the system easier to implement, it would reduce the efficiency of the system, since all yellow lights would necessarily stay yellow for their maximum length. By allowing the stoplight to receive updates from surrounding cars, it can make more informed decisions. For example, if there are no nearby cars on a lane with a green light, the light could change almost immediately through yellow to red, making a faster transition to a green light for intersecting traffic. As well as being a less efficient system, the proof complexity for this model is greater, since the antecedent would contain only parameters and we would be unable to use state variables directly. It would be interesting future work to use models proved in this paper as lemmas to aid verification of a system in which stoplights do not receive any data about the cars on their lane.

In addition to giving a non-deterministic model which encompasses all possible control decisions for the stoplight, the model also defines the control choices of the vehicle. The physical movement of the car follows the kinematic equations described in Sect. IV.

The discrete control choices of the car's acceleration, however, must now depend on information received from the stoplight. If the light is green, or if the car has already passed the light, then the car is free to accelerate. If the car is stopped, it may remain stopped as long as it is not in the intersection. We allow the car to brake at any time, so if there is an emergency, a human may always override the system by engaging the brakes. For the full model, see Model 1 in Appendix B. In this model, any time the light is ahead and either red or yellow, the car must be stopped or applying the brakes. Of course, this is not an efficient model when we look at systems with very long lanes or multiple intersections, since it will require a car to start braking for a red light when it is still miles away. Cars which will make it through the yellow also slow down before passing the light. It is an interesting extension of our model to add another choice of accelerating when the light is red, but the car is still a safe distance away. The controller described in this section is presented formally as "`lane`" in Appendix B.

B. Verification

Now that we have a suitable model for a single lane with one car and a stoplight, we identify a suitable set of safety requirements and prove that our model never violates them. For this model, we are not yet concerned with how this lane interacts with other lanes in the system. We only want to know that the local behavior on this lane is safe. Therefore, in order to prove safety, we show that the car never violates the stoplight control by running a red light.

Proposition 1 (Safety of single lane control): For a car driving on a lane with a single stoplight on the lane, where both the car and the light began in a controllable state and follow the control described in this Appendix B, the car will never be in the intersection while the light is red. In other words, if the light starts out red, and the car is in a position to stop for it (or has already passed it), then at no point in the future will the car be in the intersection while the light is red. This is expressed in *quantified differential dynamic logic* Qd$\mathcal{L}$ (see Appendix A for details) by the following provable formula:

$$I = red \land \left(xI < x \lor xI > x + \tfrac{v^2}{2B}\right)$$
$$\rightarrow [\mathtt{lane}](I = red \rightarrow xI \neq x)$$

Proposition 1 is easy to prove semi-automatically in our verification tool KeYmaera [27] after we identify a property which is stronger than the safety requirement, and always holds between sensor updates. We use this property formally as a *loop invariant* [22, 24]. Since the car may get updates at any time less than ε, the loop invariant must hold at all times less than ε. For Proposition 1, we use the invariant:

$$v \geq 0 \land v \leq V \land \left(I = red \rightarrow \left(xI < x \lor xI > x + \frac{v^2}{2B}\right)\right)$$

This formula means that, at all times, if the light is *red*, then either the car has passed the light, or it has position and velocity such that if it applies braking force $-B$, it will stop before it reaches the intersection. In addition, we are guaranteed that the velocity is always between 0 and V.

With this loop invariant, the proof required 9 human interactive steps out of a total of 4,142 nodes.

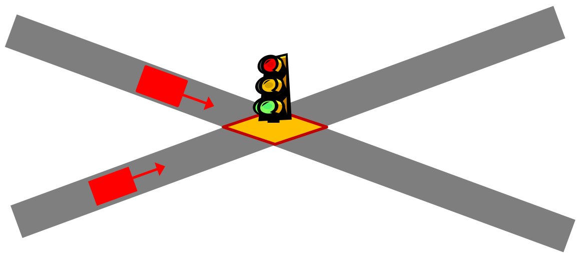
Fig. 2: Intersection

VI. INTERSECTION CONTROL

In Sect. V, we verified an automated control system for a single lane with one car and one light. Now, we create a comprehensive model of a two-lane intersection, where there is one car on each lane and a stoplight at their intersection, as illustrated in Fig. 2. It is important to notice that it is not necessary to model a situation where a car turns into the other lane at the intersection. Such instances correspond to a merge rather than an intersection (see Fig. 1a).

A. Modeling

Most of the modeling challenges have already been addressed in Sect. V. Increasing from a one lane view to verifying a two-lane intersection is very intuitive. Unfortunately, the introduction of additional lights and cars, each of which necessarily have multiple states, greatly increases the computational difficulty of the problem. The discrete and continuous control for the cars on each lane, for instance, is identical to the control presented in Sect. V, except for the change from primitive variables like x and v to first-order variables $x(1)$, $x(2)$, $v(i)$, etc. Additionally, the stoplights, on which the cars depend, will now depend on one another. So, in order for a light to turn green, the light must check that the other face of the stoplight is red. The primary difference is that we now have control choices for two cars and two stoplights instead of just one of each, as well as continuous dynamics for both cars. The controller described in this section is presented formally as "`ic`" in Appendix C.

B. Verification

Now that we have a model for a two lane intersection, we have to describe a set of requirements that we want the model to satisfy in order to ensure safety. These requirements will build upon the safety requirements proved in Sect. V. We want to show both that the cars do not enter the intersection when the light is red and that the light will always be red for at least one of the two lanes. By guaranteeing both of these requirements simultaneously, we have assured that the two cars will not collide with each other.

Proposition 2 (Safety of intersection control): If the stoplight and two cars start in a controllable state (i.e. each car is a safe distance from a red light), and the light and cars behave according to our model, then no car will ever enter the intersection when its light is red, and the stoplight will always have a red light in at least one of the two lanes. In other words, the two cars will not collide under the intersection protocol, `ic`. This is expressed in Qd$\mathcal{L}$ by the following provable formula:

$$\begin{aligned}&\Big(I(1) = red \wedge \big(xI(1) < x(1) \vee xI(1) > x(1) + \tfrac{v(1)^2}{2B}\big)\\ &\wedge\ I(2) = red \wedge \big(xI(2) < x(2) \vee xI(2) > x(2) + \tfrac{v(2)^2}{2B}\big)\Big)\\ &\quad \rightarrow [\mathtt{ic}]\Big((I(1) = red \rightarrow xI(1) \neq x(1))\\ &\qquad \wedge (I(2) = red \rightarrow xI(2) \neq x(2))\\ &\qquad \wedge (I(1) = red \vee I(2) = red)\Big)\end{aligned}$$

The proof of Proposition 2 is completed in KeYmaera. To simplify the computational complexity of the proof, we break the original requirement into the three simpler requirements,

$$(I(1) = red \rightarrow xI(1) \neq x(1)), \tag{1}$$

$$(I(2) = red \rightarrow xI(2) \neq x(2)),\ \textit{and} \tag{2}$$

$$(I(1) = red \vee I(2) = red), \tag{3}$$

and verify them separately. This is necessary because each of these requirements needs a different loop invariant.

Each of the three requirements has been proved in KeYmaera. The only difference, besides naming, between proving the red lights are not violated under this model and proving safety for the model in Sect. V is an increased branching factor due to the added choices in discrete and continuous dynamics for both cars and the stoplight. The branching factor can be lessened somewhat by manually eliminating unnecessary assumptions in the proof tree. In KeYmaera, the proof of the safety requirement in equation (1) took 443 human interactive steps out of 178,539 nodes, requirement (2) needed 378 interactive steps out of 170,210 nodes, while requirement (3) needed no human interactive steps, and was proved fully automatically with 91,112 nodes.

VII. Conclusions and Future Work

Distributed car control has been proposed repeatedly as a solution to safety and efficiency problems in ground transportation. Yet, a move to this next generation technology, however promising it may be, is only wise when its reliability has been ensured. We have presented formal verification results guaranteeing collision freedom in a series of increasingly complex intersection settings, culminating in a safety proof for an intersection of two single-lane roads.

Formal verification of intelligent vehicle systems with intersections as hybrid systems is essentially unstudied. Because of this, there are many rich topics for future research, including time synchronization of lights and cars, sensor inaccuracy, non-zero length cars, liveness properties, and combinations of merge and cross intersection protocols.

Acknowledgements. The authors would like to thank the reviewers for their constructive and thorough feedback.

References

[1] J. Chang, D. Cohen, L. Blincoe, R. Subramanian, and L. Lombardo, "CICAS-V research on comprehensive costs of intersection crashes," NHTSA, Tech. Rep. 07-0016, 2007.

[2] W. Damm, H. Hungar, and E.-R. Olderog, "Verification of cooperating traffic agents," *International Journal of Control*, vol. 79, no. 5, pp. 395–421, May 2006.

[3] T.-S. Dao, C. M. Clark, and J. P. Huissoon, "Distributed platoon assignment and lane selection for traffic flow optimization," in *IEEE IV'08*, 2008, pp. 739–744.

[4] ——, "Optimized lane assignment using inter-vehicle communication," in *IEEE IV'07*, 2007, pp. 1217–1222.

[5] R. Hall, C. Chin, and N. Gadgil, "The automated highway system / street interface: Final report," Institute of Transportation Studies, UC Berkeley, PATH Research Report UCB-ITS-PRR-2003-06, 2003.

[6] R. Hall and C. Chin, "Vehicle sorting for platoon formation: Impacts on highway entry and troughput," Institute of Transportation Studies, UC Berkeley, PATH Research Report UCB-ITS-PRR-2002-07, 2002.

[7] A. Hsu, F. Eskafi, S. Sachs, and P. Varaiya, "Design of platoon maneuver protocols for IVHS," Institute of Transportation Studies, UC Berkeley, PATH Research Report UCB-ITS-PRR-91-6, 1991.

[8] P. A. Ioannou, *Automated Highway Systems*. Springer, 1997.

[9] H. Jula, E. B. Kosmatopoulos, and P. A. Ioannou, "Collision avoidance analysis for lane changing and merging," Institute of Transportation Studies, UC Berkeley, PATH Research Report UCB-ITS-PRR-99-13, 1999.

[10] R. Horowitz, C.-W. Tan, and X. Sun, "An efficient lane change maneuver for platoons of vehicles in an automated highway system," Institute of Transportation Studies, UC Berkeley, PATH Research Report UCB-ITS-PRR-2004-16, 2004.

[11] S. E. Shladover, "Effects of traffic density on communication requirements for Cooperative Intersection Collision Avoidance Systems (CICAS)," Institute of Transportation Studies, UC Berkeley, PATH Working Paper UCB-ITS-PWP-2005-1, 2004.

[12] O. Stursberg, A. Fehnker, Z. Han, and B. H. Krogh, "Verification of a cruise control system using counterexample-guided search," *Control Engineering Practice*, 2004.

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

[14] T. Wongpiromsarn, S. Mitra, R. M. Murray, and A. G. Lamperski, "Periodically controlled hybrid systems," in *HSCC*, ser. LNCS, R. Majumdar and P. Tabuada, Eds., vol. 5469. Springer, 2009, pp. 396–410.

[15] W. Chee and M. Tomizuka, "Vehicle lane change maneuver in automated highway systems," Institute of Transportation Studies, UC Berkeley, PATH Research Report UCB-ITS-PRR-94-22, 1994.

[16] R. Johansson and A. Rantzer, Eds., *Nonlinear and Hybrid Systems in Automotive Control*. Society of Automotive Engineers Inc., 2003.

[17] M. Althoff, D. Althoff, D. Wollherr, and M. Buss, "Safety verification of autonomous vehicles for coordinated evasive maneuvers," in *IEEE IV'10*, 2010, pp. 1078 – 1083.

[18] L. Berardi, E. Santis, M. Benedetto, and G. Pola, "Approximations of maximal controlled safe sets for hybrid systems," in *Nonlinear and Hybrid Systems in Automotive Control*, 2002.

[19] S. M. Loos, A. Platzer, and L. Nistor, "Adaptive cruise control: Hybrid, distributed, and now formally verified," in *FM*, ser. LNCS, M. Butler and W. Schulte, Eds., vol. 6664. Springer, 2011, pp. 42–56.

[20] J. Lygeros and N. Lynch, "Strings of vehicles: Modeling safety conditions," in *HSCC*, 1998, pp. 273–288.

[21] A. Platzer, "Quantified differential dynamic logic for distributed hybrid systems," in *CSL*, ser. LNCS, A. Dawar and H. Veith, Eds., vol. 6247. Springer, 2010, pp. 469–483.

[22] ——, "Differential dynamic logic for hybrid systems." *J. Autom. Reas.*, vol. 41, no. 2, pp. 143–189, 2008.

[23] ——, "Differential-algebraic dynamic logic for differential-algebraic programs," *J. Log. Comput.*, vol. 20, no. 1, pp. 309–352, 2010.

[24] ——, *Logical Analysis of Hybrid Systems: Proving Theorems for Complex Dynamics*. Heidelberg: Springer, 2010.

[25] M. Oishi, I. Mitchell, A. Bayen, and C. Tomlin, "Invariance-preserving abstractions of hybrid systems: Application to user interface design," in *IEEE Transactions on Control Systems Technology*, vol. 16, 2008, pp. 229–244.

[26] A. Platzer, "Stochastic differential dynamic logic for stochastic hybrid programs," in *CADE*, ser. LNCS, N. Bjørner and V. Sofronie-Stokkermans, Eds., vol. 6803. Springer, 2011, pp. 431–445.

[27] A. Platzer and J.-D. Quesel, "KeYmaera: A hybrid theorem prover for hybrid systems." in *IJCAR*, ser. LNCS, A. Armando, P. Baumgartner, and G. Dowek, Eds., vol. 5195. Springer, 2008, pp. 171–178.

Appendix

A. Quantified Differential Dynamic Logic

Distributed car control systems are distributed hybrid systems, which we model by *quantified hybrid programs* (QHPs) [21]. QHPs are defined by the grammar (α,β are QHPs, θ a term, i a variable, f a function symbol, and H a formula of first-order logic):

$$\alpha,\beta \ ::= \ \forall i\!:\!C\ f(i) := \theta \mid \forall i\!:\!C\ f(i)' = \theta \,\&\, H \mid f(i) := * \\ \mid ?H \mid \alpha \cup \beta \mid \alpha;\beta \mid \alpha^*$$

The effect of *quantified assignment* $\forall i\!:\!C\ f(i) := \theta$ is an instantaneous discrete jump assigning θ to $f(i)$ simultaneously for all objects i of type C. Usually i occurs in θ. The effect of *quantified differential equation* $\forall i\!:\!C\ f(i)' = \theta \,\&\, H$ is a continuous evolution where, for all objects i of type C, all differential equations $f(i)' = \theta$ hold *and* (written & for clarity) formula H holds throughout the evolution (the state remains in the region described by H). Usually, i occurs in θ. Here $f(i)'$ is intended to denote the derivative of the interpretation of the term $f(i)$ over time during continuous evolution, not the derivative of $f(i)$ by its argument i. For $f(i)'$ to be defined, we assume f is an $\mathbb{R}$-valued function symbol. The effect of the random assignment $f(i) := *$ is to non-deterministically pick an arbitrary object (of type the type of $f(i)$) as the value of $f(i)$.

The effect of *test* $?H$ is a *skip* (i.e., no change) if formula H is true in the current state and *abort* (blocking the system run by a failed assertion), otherwise. *Non-deterministic choice* $\alpha \cup \beta$ is for alternatives in the behavior of the distributed hybrid system. In the *sequential composition* $\alpha;\beta$, QHP β starts after α finishes (β never starts if α continues indefinitely). *Non-deterministic repetition* α^* repeats α an arbitrary number of times $\geq$0.

For stating and proving properties of QHPs, we use *quantified differential dynamic logic* Qd$\mathcal{L}$ [21] with the grammar:

$$\phi,\psi \ ::= \ \theta_1 = \theta_2 \mid \theta_1 \geq \theta_2 \mid \neg\phi \mid \phi \wedge \psi \mid \phi \vee \psi \mid \phi \rightarrow \psi \\ \mid \forall i\!:\!C\ \phi \mid \exists i\!:\!C\ \phi \mid [\alpha]\phi \mid \langle\alpha\rangle\phi$$

In addition to all formulas of first-order real arithmetic, Qd$\mathcal{L}$ allows formulas of the form $[\alpha]\phi$ with a QHP α and a formula ϕ. Formula $[\alpha]\phi$ is true in a state ν iff ϕ is true in all states that are reachable from ν by following the transitions of α; see [21] for details.

B. Model of Single Lane Stoplight Control

In Model 1 we present formally in Qd$\mathcal{L}$ the model described in Sect. V. We denote the discrete control of the stoplight as *ICtrl* and the discrete control of the car as *CCtrl*. The continuous dynamics and evolution domain are called *dyn*. In the evolution domain, we restrict the car to driving forward on the lane ($v \geq 0$) and within the speed limit ($v \leq V$). We also require that the updates to sensor readings about the color of the light come within time ε ($t \leq \varepsilon$).

Model 1 Single Lane Control

$$\begin{aligned}
\texttt{lane} &\equiv (ICtrl; CCtrl; dyn)^* \\
ICtrl &\equiv (?(I = green);\ I := yellow \\
&\quad \cup\ ?\Big(I = yellow \\
&\qquad \wedge \left(xI{<}x \vee xI{>}x+\tfrac{v^2}{2B}+\left(\tfrac{A}{B}+1\right)\left(\tfrac{A}{2}\varepsilon^2+v\varepsilon\right)\right)\Big); \\
&\qquad I := red \\
&\quad \cup\ ?(I = red);\ I := green \\
&\quad \cup\ ?true) \\
CCtrl &\equiv (?(I = green \vee xI = x);\ a := A \\
&\quad \cup\ ?(v = 0 \wedge xI \neq x);\ a := 0 \\
&\quad \cup\ ?(v = V \wedge (I = green \vee xI = x));\ a := 0 \\
&\quad \cup\ a := -B) \\
dyn &\equiv (t := 0; x' = v, v' = a \ \&\ v \geq 0 \wedge v \leq V \wedge t \leq \varepsilon)
\end{aligned}$$

C. Model of Intersection Control

In Model 2 we give a formal model of the intersection control described in Sect. VI. This time, we denote the discrete control of intersections and cars relative to the lane on which they operate: $ICtrl(i)$ and $CCtrl(i)$, respectively for lane i. In order for a light to change from yellow to red, one of two properties must hold to ensure safety. The car must either be past the position of the light, or far enough away from the light that if it were to apply the brakes after a delay of at most ε time units, it would be able to come to a complete stop before the light.

Model 2 Intersection Control

$$\begin{aligned}
\texttt{ic} &\equiv (ICtrl(1); ICtrl(2); CCtrl(1); CCtrl(2); dyn)^* \\
ICtrl(i) &\equiv (?(I(i) = green);\ I(i) := yellow \\
&\quad \cup\ ?\Big(I(i) = yellow \\
&\qquad \wedge \Big(xI(i) < x \\
&\qquad\quad \vee \left(xI(i){>}x+\tfrac{v^2}{2B}+\left(\tfrac{A}{B}+1\right)\left(\tfrac{A}{2}\varepsilon^2+v\varepsilon\right)\right)\Big)\Big);\ I(i){:=}red \\
&\quad \cup\ ?\left(\bigwedge_j I(j) = red\right);\ I(i) := green \\
&\quad \cup\ ?true) \\
CCtrl(i) &\equiv (?(I(i) = green \vee xI(i) = x(i));\ a(i) := A \\
&\quad \cup\ ?(v(i) = 0 \wedge xI(i) \neq x(i));\ a(i) := 0 \\
&\quad \cup\ ?(v(i) = V \wedge \\
&\qquad (I(i) = green \vee xI(i) = x(i)));\ a(i) := 0 \\
&\quad \cup\ a(i) := -B) \\
dyn &\equiv (t := 0; x'(1) = v(1), v'(1) = a(1), \\
&\qquad x'(2) = v(2), v'(2) = a(2) \\
&\qquad \&\ v(1) \geq 0 \wedge v(2) \geq 0 \\
&\qquad \wedge v(1) \leq V \wedge v(2) \leq V \wedge t \leq \varepsilon)
\end{aligned}$$