# Differential Dynamic Logics

## Automated Theorem Proving for Hybrid Systems

André Platzer

**Designing and analyzing hybrid systems, which are models for complex physical systems, is expensive and error-prone. The dissertation presented in this article introduces a verification logic that is suitable for analyzing the behavior of hybrid systems. It presents a proof calculus and a new deductive verification tool for hybrid systems that has been used successfully to verify aircraft and train control.**

## 1 Hybrid Systems

Designing and analyzing complex physical systems like aircraft or train control or embedded systems in cars is expensive and complicated, but very important for ensuring safety. *Hybrid systems* [4, 2] are models for these systems and are characterized by interacting discrete and continuous dynamics. The continuous dynamics originates from the interaction with the physical environment and often comes from physical movement in space. The discrete dynamics typically results from discrete control decisions like when to brake or accelerate or in which direction to move, which is controlled by a small embedded controller or a full computer system. Correct behavior of these systems depends on correct interaction of the discrete dynamics with the continuous dynamics, viz. of the hybrid system. Safety properties like collision freedom, for instance, depend on the combination of continuous physical movement and discrete control decisions.

The thesis [7] presented in this article introduces logic-based analysis and automated theorem proving techniques for hybrid systems. It introduces the first compositional analysis technique for hybrid systems, which proves properties of hybrid systems by analyzing properties of parts of the system, recursively. This compositional approach, which is expressed naturally in the logic presented in the thesis, has scalability advantages and has been used successfully in aircraft and train control systems.

Figure 1 shows a typical example of a complex physical system with a natural hybrid system model: the *European Train Control System* (ETCS) [3], which has also been studied in the work presented here. In the simple case, the continuous dynamics of ETCS can be described by the differential equation $z'' = a$, which expresses that the train position $z$ is evolving over time and its second time-derivative $z''$ is the current acceleration $a$.

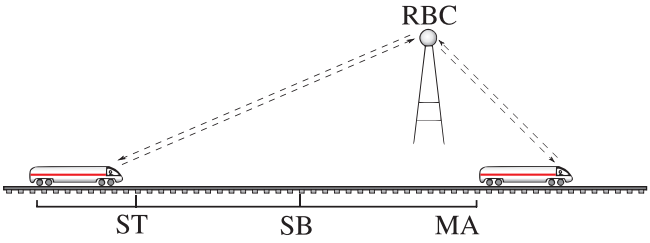


Figure 1: European Train Control System

The discrete dynamics of ETCS comes from the fact that the controllers change the acceleration $a$ depending on the current track situation. They repeatedly increase or decrease $a$ when the current speed is too high or too low for the current segment of the track profile. Likewise, computer controllers switch to braking mode (set $a := -b$ for a braking constant $b$), which happens at the *start braking* point *SB* at the latest, to ensure they always have a safe distance to other trains; see Fig. 1. For this, ETCS uses *movement authorities* (MA), which change dynamically using communication with the Radio Block Center (RBC), and represent where a train is allowed to move to on the track. At some point *ST* at the latest, the train will *start talking* with the RBC to negotiate for MA-extension, depending on other trains.

## 2 Differential Dynamic Logic

Analyzing complex physical systems is difficult, because their behavior depends on the interaction of physical dynamics with control and needs to take both the differential equations and discrete control switching behavior into account.

As a natural specification and verification language for safety, controllability, liveness and more general correctness properties of hybrid systems, we have introduced the *differential dynamic logic* d$\mathcal{L}$ [6, 7]. Unlike other logics [2], d$\mathcal{L}$ directly allows to state and prove properties of the behavior of hybrid systems.

Assuming *ETCS* denotes the hybrid system for the ETCS control protocol, the d$\mathcal{L}$ formula

$$b > 0 \wedge v^2 \leq 2b(MA - z) \;\rightarrow\; [ETCS]\, z \leq MA \qquad (1)$$

expresses that the train position $z$ will always be within the movement authority *MA* during the train run, if the braking power $b$ is positive and constraint $v^2 \leq 2b(MA - z)$ on velocity $v$ is satisfied initially. Under the assumptions on the left-hand side of the implication in (1), the modal subformula $[ETCS]\, z \leq MA$ on the right-hand side expresses that all states reachable by hybrid system *ETCS* satisfy $z \leq MA$, i.e., are inside the movement authority.

## 3 Automated Theorem Proving

For proving that a property like the safety property (1) about the train dynamics is indeed true, we have introduced a verification calculus [6, 7] that works by recursive symbolic decomposition.

That is, we prove property (1) by reducing it to properties of subsystems of *ETCS*, for instance, of the automatic train protection controller, the speed supervision, and of the RBC. In fact, we have introduced the very first compositional verification logic for hybrid systems. Because decompositions can exploit modularity in system designs, this compositional approach can scale to more complicated and larger systems than previous approaches, which were based on exhaustive state space exploration [4].

As part of this decomposition, we handle the effect of discrete state changes like $a := -b$ on the dynamics by symbolic execution of the hybrid system dynamics. Likewise, we analyze choices in mode switching and mode switch conditions by reasoning by cases. Simple differential equations can be handled immediately using their solution. The differential equation $z'' = a$, for instance, has solution $z(t) = \frac{a}{2}t^2 + v_0 t + z_0$ where $z_0$ and $v_0$ are the initial position and speed of the train. For systems with more complicated differential equations, so-called differential invariants [5] can be used to verify properties of differential equations when the solution is unknown or undecidable.

Another source of complexity are quantifiers for real variables. When analyzing properties of hybrid systems, several statements need to be proven for all ($\forall x$) or some ($\exists x$) values of a real variable $x$. For proving simple quantified (first-order) formulas, quantifier elimination in real-closed fields [1] can be used directly. Quantifiers in front of modal formulas cannot be handled like that, however, because they express quantified reachability statements. The following d$\mathcal{L}$ formula, for example, expresses that, under certain assumptions, there is a choice of the start braking point *SB* such that the *ETCS* controller always remains safe:

$$z \leq MA \wedge v = 0 \wedge b > 0 \rightarrow \exists SB\, [ETCS]\, z \leq MA \qquad (2)$$

To handle advanced quantified properties like (2), we introduce an approach that postpones quantifier elimination until the hybrid system has been decomposed sufficiently. To achieve this formally, we generalize theorem proving techniques for quantifiers to the presence of real arithmetic and combine them with quantifier elimination correspondingly, leading to an approach for deduction modulo theories [6].

Along with suitable algorithmic refinements, automated theorem proving techniques and verification algorithms [8], this approach has been implemented in the new verification tool KeYmaera[1] for hybrid systems; see Fig. 2.

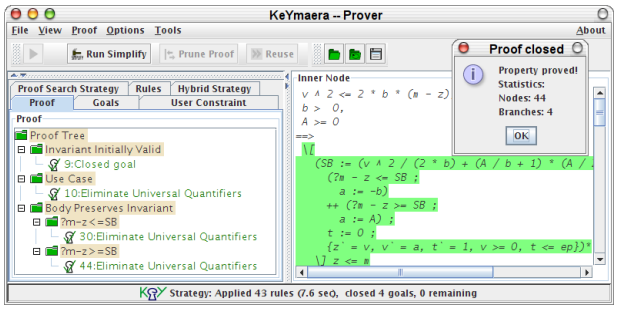


Figure 2: Verification tool KeYmaera for hybrid systems

[1] http://symbolaris.com/info/KeYmaera.html

# 4 Conclusions

The most important conceptual contribution of the thesis [7] presented in this article is the coherent logical analysis approach for hybrid systems, which can be understood naturally in terms of the specification and verification logic d$\mathcal{L}$ (differential dynamic logic). The most fundamental theoretical contribution is our completeness result showing that all true d$\mathcal{L}$ properties can be proven from elementary properties of differential equations, which is the first result of this kind. Practical contributions are the compositional verification calculus and verification algorithms that are implemented in the new verification tool KeYmaera. Important applications can be found in verification results for collision avoidance in train control and air traffic control protocols. This work has been conducted in the DFG SFB "Automatic Verification and Analysis of Complex Systems".

# References


[1] G. E. Collins and H. Hong. Partial cylindrical algebraic decomposition for quantifier elimination. *J. Symb. Comput.*, 12(3):299–328, 1991.

[2] J. M. Davoren and A. Nerode. Logics for hybrid systems. *IEEE*, 88(7):985–1010, July 2000.

[3] ERTMS User Group. ERTMS/ETCS System requirements specification, 2002. Version 2.2.2.

[4] T. A. Henzinger. The theory of hybrid automata. In *LICS*, pages 278–292, Los Alamitos, 1996. IEEE Computer Society.

[5] A. Platzer. Differential-algebraic dynamic logic for differential-algebraic programs. *Journal of Logic and Computation*, 2008. DOI 10.1093/logcom/exn070.

[6] A. Platzer. Differential dynamic logic for hybrid systems. *Journal of Automated Reasoning*, 41(2):143–189, 2008.

[7] A. Platzer. *Differential Dynamic Logics: Automated Theorem Proving for Hybrid Systems*. PhD thesis, Department of Computing Science, University of Oldenburg, 2008.

[8] A. Platzer and E. M. Clarke. Computing differential invariants of hybrid systems as fixedpoints. In A. Gupta and S. Malik, editors, *CAV*, volume 5123 of *LNCS*, pages 176–189. Springer, 2008.


## Contact


André Platzer
Carnegie Mellon University
Pittsburgh, PA-15213, USA
Tel.: +1 (412)-268-1558
Email: aplatzer@cs.cmu.edu
http://symbolaris.com/


Bild

**André Platzer** received a Diploma (M.Sc.) in computer science from the University of Karlsruhe (TH) in 2004 and a Ph.D. in computer science from the University of Oldenburg in 2008, both *summa cum laude*. Immediately after completing his Ph.D., he started as an Assistant Professor in Computer Science at the Carnegie Mellon University, Pittsburgh, PA, USA. His research includes verification of hybrid systems, logic, automated theorem proving and model checking, as well as symbolic-numerical algorithms.