15-424: Foundations of Cyber-Physical Systems

Lecture Notes on Ghosts & Differential Ghosts

André Platzer

Carnegie Mellon University
Lecture 15

1 Introduction

Lecture 10 on Differential Equations & Differential Invariants and Lecture 11 on Differential Equations & Proofs equipped us with powerful tools for proving properties of differential equations without having to solve them. *Differential invariants* (DI) [Pla10a] prove properties of differential equations by induction based on the right-hand side of the differential equation, rather than its much more complicated global solution. *Differential cuts* (DC) [Pla10a] made it possible to prove another property C of a differential equation and then change the dynamics of the system around so that it can never leave region C. Lecture 14 on Differential Invariants & Proof Theory studied some part of the proof theory of differential equations and proved the differential invariance chart that compares the deductive power of classes of differential invariants; see Fig. 1.

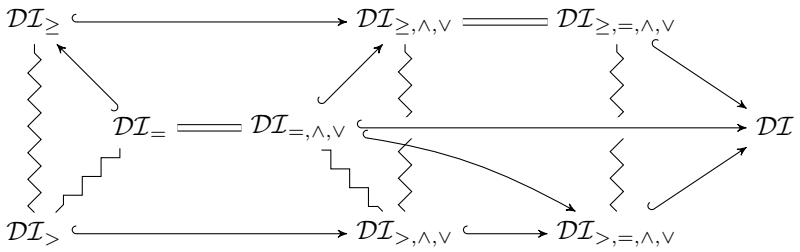

Figure 1: Differential invariance chart

It can be shown that differential cuts are a fundamental proof principle for differential equations [Pla12], because some properties can only be proved with differential cuts.

Yet, it can also be shown that there are properties where even differential cuts are not enough, but differential ghosts become necessary [Pla12]. Differential ghosts [Pla12],

spooky as they may sound, turn out to be a useful proof technique for differential equations.

This lecture is based on [Pla12, Pla10b].

2 Recap

Recall the following proof rules for differential equations from Lecture 11 on Differential Equations & Proofs:

Note 1 (Proof rules for differential equations).

$$\text{(DI)}\ \frac{H \vdash F'^{\theta}_{x'}}{F \vdash [x' = \theta \,\&\, H]F} \qquad \text{(DW)}\ \frac{H \vdash F}{\Gamma \vdash [x' = \theta \,\&\, H]F, \Delta}$$

$$\text{(DC)}\ \frac{\Gamma \vdash [x' = \theta \,\&\, H]C, \Delta \qquad \Gamma \vdash [x' = \theta \,\&\, (H \wedge C)]F, \Delta}{\Gamma \vdash [x' = \theta \,\&\, H]F, \Delta}$$

With cuts and generalizations, earlier lectures have also shown that the following can be proved:

$$\frac{A \vdash F \quad F \vdash [x' = \theta \,\&\, H]F \quad F \vdash B}{A \vdash [x' = \theta \,\&\, H]B} \tag{1}$$

3 Arithmetic Ghosts

$$q := \frac{b}{c} \quad \rightsquigarrow \quad q := *;\ ?qc = b \quad \rightsquigarrow \quad q := *;\ ?qc = b \wedge c \neq 0$$

where $q := *$ is the nondeterministic assignment that assigns an arbitrary real number to q.

$$x := 2 + \frac{b}{c} + e \quad \rightsquigarrow \quad q := *;\ ?qc = b;\ x := 2 + q + e \quad \rightsquigarrow \quad q := *;\ ?qc = b \wedge c \neq 0;\ x := 2 + q + e$$

Here q is called an arithmetic ghost, because q is an auxiliary variable that is only in the hybrid program for the sake of defining the quotient $\frac{b}{c}$.

4 Nondeterministic Assignments & Ghosts of Choice

The HP statement $x := *$ is a nondeterministic assignment that assigns an arbitrary real number to x. Comparing with the syntax of hybrid programs from Lecture 3 on Choice & Control, however, it turns out that such a statement is not in the official language of hybrid programs.

$$\alpha, \beta \ ::= \ x := \theta \mid ?H \mid x' = \theta \,\&\, H \mid \alpha \cup \beta \mid \alpha; \beta \mid \alpha^* \tag{2}$$

What now?

One possible solution, which is the one taken in the implementation of KeYmaera [PQ08], is to add the nondeterministic assignment $x := *$ as a statement to the syntax of hybrid programs.

$$\alpha, \beta \ ::= \ x := \theta \mid \ldots \mid x := *$$

Consequently, nondeterministic assignments need a semantics to become meaningful.

7. $\rho(x := *) = \{(\nu, \omega) \ : \ \omega = \nu$ except for the value of x, which can be any real number$\}$

And nondeterministic assignments finally need proof rules so that they can be handled in proofs.

$$(\langle:*\rangle) \ \frac{\exists x \, \phi}{\langle x := * \rangle \phi} \qquad ([:*]) \ \frac{\forall x \, \phi}{[x := *]\phi}$$

Another approach for adding nondeterministic assignments $x := *$ to hybrid programs is to consider whether we even have to do that. That is, to understand whether $x := *$ is truly a new program construct or whether it can be defined in terms of the other hybrid program statements from (2). Is $x := *$ definable by a hybrid program?

Before you read on, see if you can find the answer for yourself.

Nondeterministic assignment $x := *$ assigns any real number to x. One hybrid program that has the same effect of giving x any arbitrary real value [Pla10b, Chapter 3] is:

$$x := * \quad \overset{\text{def}}{\equiv} \quad x' = 1 \cup x' = -1 \tag{3}$$

That is not the only definition of $x := *$, though. An equivalent definition is [Pla13]:

$$x := * \quad \overset{\text{def}}{\equiv} \quad x' = 1; x' = -1$$

When working through the intended semantics of the left-hand side $x := *$ shown in case 7 above and the actual semantics of the right-hand side of (3) according to Lecture 3, it becomes clear that both sides of (3) mean the same. Hence, the above definition (3) capture the right concept. And, in particular, just like if-then-else, nondeterministic assignments do not really have to be added to the language of hybrid programs, because they can already be defined. Likewise, no proof rules would have to be added for nondeterministic assignments, because there are already proof rules for the constructs used in the right-hand side of the definition of $x := *$ in (3). Since the above proof rules for $x := *$ are particularly easy, though, it is usually more efficient to include them directly, which is what KeYmaera does.

What may, at first sight, appear slightly spooky about (3), however, is that the left-hand side $x := *$ is clearly an instant change in time where x changes its value instantaneously to some arbitrary new real number. That is less so for the right-hand side of (3), which involves two differential equations, which take time to follow.

The clue is that this passage of time is not observable in the state of the system. Consequently, the left-hand side of (3) really means the same as the right-hand side of (3). Remember from earlier lectures that time is not special. If a CPS wants to refer to time, it would have a clock variable t with the differential equation $t' = 1$. With such an addition, however, the passage of time t becomes observable in the value of variable t and, hence, a corresponding variation of the right-hand side of (3) would not be equivalent to $x := *$ (indicated by $\not\equiv$):

$$x := * \quad \not\equiv \quad x' = 1, t' = 1 \cup x' = -1, t' = 1$$

5 Differential-algebraic Ghosts

$$q' = \frac{b}{c} \quad \rightsquigarrow \quad q' = * \,\&\, qc = b \quad \rightsquigarrow \quad q' = * \,\&\, qc = b \wedge c \neq 0$$

See [Pla10b, Chapter 3] for the meaning of the nondeterministic differential equation $q' = *$.[1]

$$x' = 2 + \frac{b}{c} + e \quad \rightsquigarrow \quad x' = 2 + q + e, q' = * \,\&\, qc = b \quad \rightsquigarrow \quad x' = 2 + q + e, q' = * \,\&\, qc = b \wedge c \neq 0$$

[1] It is the same as the differential-algebraic constraint $\exists d\, q' = d$, but differential-algebraic constraints have not been introduced in this course so far.

Variable q is a differential-algebraic ghost in the sense of being an auxiliary variable in the differential-algebraic equation for the sake of defining the quotient $\frac{b}{c}$.

Together with the reduction of divisions in discrete assignments from Sect. 3, plus the inside that divisions in tests and evolution domain constraints can always be rewritten to division-free form, is a (sketchy) proof showing that hybrid programs and differential dynamic logic do not need divisions [Pla10b]. The advantage of eliminating divisions this way is that differential dynamic logic does not need special precautions for divisions and that the handling of zero divisors is made explicit in the way the divisions are eliminated from the formulas. In practice, however, it is still useful to use divisions, yet great care has to be exercised to make sure that no inadvertent divisions by zero could ever cause singularities.

6 Discrete Ghosts

Lemma 1 (Discrete ghosts). *The following is a sound proof rule for introducing auxiliary variables or (discrete)* ghosts*:*

$$\text{(IA)}\ \frac{\Gamma \vdash [y := \theta]\phi, \Delta}{\Gamma \vdash \phi, \Delta}$$

where y is a new program variable.

That proof rule IA is sound can be argued based on the soundness of the substitution axiom [:=] from Lecture 5 on Dynamical Systems & Dynamic Axioms. The assignment axiom [:=] proves validity of

$$\phi \leftrightarrow [y := \theta]\phi$$

because the fresh variable y does not occur in ϕ.

7 Remember the Bouncing Ball

Recall the following sequent for the bouncing ball from Lecture 7 on Control Loops & Invariants, which was based on an argument in Lecture 4 on Safety & Contracts.

$$2gh = 2gH - v^2 \wedge h \geq 0 \rightarrow [h' = v, v' = -g \,\&\, h \geq 0](2gh = 2gH - v^2 \wedge h \geq 0) \tag{4}$$

The dL formula (4) can be proved using the solutions of the differential equation with proof rule [′]. dL formula (4) can also be proved using differential invariants, with a differential cut and a use of differential weakening:

$$\text{DC}\left(\text{DI}\,\frac{\mathbb{R}\,\frac{*}{h \geq 0 \vdash 2gv = -2v(-g)}}{h \geq 0 \vdash (2gh' = -2vv')^{v}_{h'}{}^{-g}_{v'}}}{2gh = 2gH - v^2 \vdash [h'' = -g \,\&\, h \geq 0]2gh = 2gH - v^2}\ \text{DW}\,\frac{ax\,\frac{*}{h \geq 0 \wedge 2gh = 2gH - v^2 \vdash 2gh = 2gH - v^2 \wedge h \geq 0}}{2gh = 2gH - v^2 \vdash [h'' = -g \,\&\, h \geq 0 \wedge 2gh = 2gH - v^2](2gh = 2gH - v^2 \wedge h \geq 0)}\right)$$
$$2gh = 2gH - v^2 \vdash [h'' = -g \,\&\, h \geq 0](2gh = 2gH - v^2 \wedge h \geq 0)$$

Note that differential weakening (DW) works for proving the postcondition $h \geq 0$, but DI would not work, because the derivative of $h \geq 0$ is $v \geq 0$, which is not an invariant

of the bouncing ball since its velocity ultimately becomes negative when it is falling according to gravity. Note that this proofs is very elegant and has notably easier arithmetic than the arithmetic we ran into when working with solutions of the bouncing ball in earlier lectures.

The reason why this proof worked so elegantly is that the invariant $2gh = 2gH - v^2 \wedge h \geq 0$ was a very good choice that we came up with in a clever way in Lecture 4. Is there a way to prove (4) without such a distinctively clever invariant that works as a differential invariant right away? Yes, of course, because (4) can even be proved using solutions [′]. But it turns out that interesting things happen when we systematically try to understand how to make a proof happen that does not use the solution rule [′] and, yet, still uses solution-based arguments. Can you conceive a way to do use solutions for differential equations without invoking rule [′]?

Before you read on, see if you can find the answer for yourself.

8 Differential Ghosts

$$2gh = 2gH - v^2 \vdash [h'' = -g \,\&\, h \geq 0](2gh = 2gH - v^2 \wedge h \geq 0)$$

Use the usual abbreviations:

$$A_{h,v} \overset{\text{def}}{\equiv} 2gh = 2gH - v^2$$

$$B_{h,v} \overset{\text{def}}{\equiv} 2gh = 2gH - v^2 \wedge h \geq 0$$

$$(h'' = -g) \overset{\text{def}}{\equiv} (h' = v, v' = -g)$$

$$\text{DA}\,\dfrac{\text{DC}\,\dfrac{\text{DI}\,\dfrac{\mathbb{R}\,\dfrac{*}{h \geq 0 \vdash -g = -1g}}{h \geq 0 \vdash (v' = -t'g)^{-g}_{v'}\,{}^{1}_{t'}} \qquad A_{h,v} \vdash \{v_0 := v\}[h'' = -g, t' = 1 \,\&\, h \geq 0 \wedge v = v_0 - tg]B_{h,v}}{A_{h,v} \vdash \{v_0 := v\}[h'' = -g, t' = 1 \,\&\, h \geq 0]B_{h,v}}}{A_{h,v} \vdash \{v_0 := v\}[h'' = -g \,\&\, h \geq 0]B_{h,v}}}{A_{h,v} \vdash [h'' = -g \,\&\, h \geq 0]B_{h,v}}\,\text{IA}$$

where the proof step marked DA omits the (here trivial) left premise of rule DA, which proves because $B_{h,v} \leftrightarrow \exists t\, B_{h,v}$ is trivially valid in first-order logic, as the fresh t does even occur in $B_{h,v}$ here.

The right premise in the above proof proves as follows

$$\text{IA}\,\dfrac{\text{DC}\,\dfrac{\text{DI}\,\dfrac{\dfrac{*}{h \geq 0 \wedge v = v_0 - tg \vdash v = v_0 - 2\frac{g}{2}t}}{h \geq 0 \wedge v = v_0 - tg \vdash (h' = v_0t' - 2\frac{g}{2}tt')^{v}_{h'}\,{}^{1}_{t'}}}{A_{h,v} \vdash \{h_0 := h, v_0 := v\}[h'' = -g, t' = 1 \,\&\, h \geq 0 \wedge v = v_0 - tg]h = h_0 + v_0t - \frac{g}{2}t^2}\,\triangleright}{A_{h,v} \vdash \{h_0 := h, v_0 := v\}[h'' = -g, t' = 1 \,\&\, h \geq 0 \wedge v = v_0 - tg]B_{h,v}}}{A_{h,v} \vdash \{v_0 := v\}[h'' = -g, t' = 1 \,\&\, h \geq 0 \wedge v = v_0 - tg]B_{h,v}}$$

The proof step marked DC has a second premise which is elided (marked by $\triangleright$) and proves as follows:

$$\text{DW}\,\dfrac{\mathbb{R}\,\dfrac{*}{h \geq 0 \wedge v = v_0 - tg \wedge h = h_0 + v_0t - \frac{g}{2}t^2 \vdash B_{h,v}}}{A_{h,v} \vdash \{h_0 := h, v_0 := v\}[h'' = -g, t' = 1 \,\&\, h \geq 0 \wedge v = v_0 - tg \wedge h = h_0 + v_0t - \frac{g}{2}t^2]B_{h,v}}$$

The arithmetic (marked $\mathbb{R}$) can be proved with enough care, but it has a twist! First of all, the arithmetic can be simplified substantially using the equality substitution rule =r and subsequent weakening.

$$\text{=r}\,\dfrac{\wedge\text{l,Wr}\,\dfrac{\wedge\text{r}\,\dfrac{\text{Wl}\,\dfrac{\vdash 2g(h_0 + v_0t - \frac{g}{2}t^2) = 2gH - (v_0 - tg)^2}{h \geq 0 \vdash 2g(h_0 + v_0t - \frac{g}{2}t^2) = 2gH - (v_0 - tg)^2} \qquad \text{ax}\,\dfrac{*}{h \geq 0 \vdash h \geq 0}}{h \geq 0 \vdash 2g(h_0 + v_0t - \frac{g}{2}t^2) = 2gH - (v_0 - tg)^2 \wedge h \geq 0}}{h \geq 0 \wedge v = v_0 - tg \wedge h = h_0 + v_0t - \frac{g}{2}t^2 \vdash 2g(h_0 + v_0t - \frac{g}{2}t^2) = 2gH - (v_0 - tg)^2 \wedge h \geq 0}}{h \geq 0 \wedge v = v_0 - tg \wedge h = h_0 + v_0t - \frac{g}{2}t^2 \vdash 2gh = 2gH - v^2 \wedge h \geq 0}$$

Observe how this use of equality substitution and weakening helped simplify the arithmetic complexity of the formula substantially and even helped to eliminate a variable (v) right away. This can be useful to simplify arithmetic in many other cases as well. The arithmetic in the left branch

$$2g(h_0 + v_0t - \frac{g}{2}t^2) = 2gH - (v_0 - tg)^2$$

expands by polynomial arithmetic and cancels as follows

$$2g(h_0 + \cancel{v_0t} - \cancel{\tfrac{g}{2}t^2}) = 2gH - v_0^2 + \cancel{2v_0tg} + \cancel{t^2g^2}$$

That leaves the remaining condition

$$2gh_0 = 2gH - v_0^2$$

Indeed, this relation characterizes exactly how H, which turns out to have been the maximal height, relates to the initial height h_0 and initial velocity v_0. In the case of initial velocity $v_0 = 0$, this relation collapses to $h_0 = H$.

For the case of the bouncing ball, this proof was unnecessarily complicated, because the solution rule [′] could have been used instead. But the same proof technique can be useful in more complicated systems that do not have computable solutions, but in which other relations between initial (or intermediate) and final state can be proved.

Lemma 2 (Differential ghosts)**.** *The following is a sound proof rule* differential auxiliaries *(DA) for introducing auxiliary differential variables or* differential ghosts *[Pla12]:*

$$\text{(DA)}\ \frac{\Gamma \vdash \phi \leftrightarrow \exists y\, \psi, \Delta \quad \Gamma, \psi \vdash [x' = \theta, y' = \eta \,\&\, H]\psi, \Delta}{\Gamma, \phi \vdash [x' = \theta \,\&\, H]\phi, \Delta}$$

This proves

where y new and $y' = \eta, y(0) = y_0$ has a solution $y : [0, \infty) \to \mathbb{R}^n$ for each y_0.

Rule DA is applicable if y is a new variable and the new differential equation $y' = \eta$ has global solutions on H (e.g., because term η satisfies a Lipschitz condition [Wal98, Proposition 10.VII], which is definable in first-order real arithmetic and thus decidable). Without that condition, adding $y' = \eta$ could limit the duration of system evolutions incorrectly. In fact, it would be sufficient for the domains of definition of the solutions of $y' = \eta$ to be no shorter than those of x. Soundness is easy to see, because precondition ϕ implies ψ for some choice of y (left premise). Yet, for any y, ψ is an invariant of the extended dynamics (right premise). Thus, ψ always holds after the evolution for some y (its value can be different than in the initial state), which still implies ϕ (left premise). Since y is fresh and its differential equation does not limit the duration of solutions of x on H, this implies the conclusion. Since y is fresh, y does not occur in H, and, thus, its solution does not leave H, which would incorrectly restrict the duration of the evolution as well.

Intuitively, rule DA can help proving properties, because it may be easier to characterize how x changes in relation to an auxiliary variable y with a suitable differential equation ($y' = \eta$).

$$\text{DA}\dfrac{\mathbb{R}\dfrac{*}{\vdash x>0 \leftrightarrow \exists y\, xy^2=1} \qquad \text{DI}\dfrac{\mathbb{R}\dfrac{*}{\vdash -xy^2+2xy\frac{y}{2}=0}}{\vdash (x'y^2+x2yy'=0)^{-x\ \frac{y}{2}}_{x'\ y'}}}{xy^2=1 \vdash [x'=-x,y'=\frac{y}{2}]xy^2=1}}{x>0 \vdash [x'=-x]x>0}$$

It can be shown [Pla12] that there are properties such as this one that need differential ghosts (or differential auxiliaries) to prove.

9 Axiomatic Ghosts

When neglecting wind, gravitation, and so on, which is appropriate for analysing cooperation in air traffic control [TPS98], the in-flight dynamics of an aircraft at x can be described by the following differential equation system; see [TPS98] for details:

$$x_1' = v\cos\vartheta \qquad x_2' = v\sin\vartheta \qquad \vartheta' = \omega. \tag{5}$$

That is, the linear velocity v of the aircraft changes both positions x_1 and x_2 in the (planar) direction corresponding to the orientation ϑ the aircraft is currently heading toward. Further, the angular velocity ω of the aircraft changes the orientation ϑ of the aircraft.

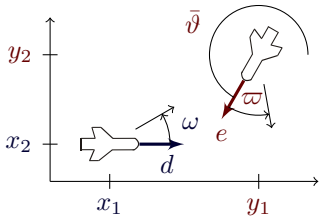

Figure 2: Aircraft dynamics

Unlike for straight-line flight ($\omega = 0$), the nonlinear dynamics in (5) is difficult to analyse [TPS98] for curved flight ($\omega \neq 0$), especially due to the trigonometric expressions which are generally undecidable. Solving (5) requires the Floquet theory of differential equations with periodic coefficients [Wal98, Theorem 18.X] and yields mixed polynomial expressions with multiple trigonometric functions. A true challenge, however, is the need to verify properties of the states that the aircraft reach by following these solutions, which requires proving that complicated formulas with mixed polynomial arithmetic and trigonometric functions hold true for all values of state variables and all possible evolution durations. However, quantified arithmetic with trigonometric functions is undecidable by Gödel's incompleteness theorem [Göd31].

To obtain polynomial dynamics, we axiomatize the trigonometric functions in the dynamics differentially and reparametrize the state correspondingly. Instead of angular orientation ϑ and linear velocity v, we use the linear speed vector

$$d = (d_1, d_2) := (v\cos\vartheta, v\sin\vartheta) \in \mathbb{R}^2$$

which describes both the linear speed $\|d\| := \sqrt{d_1^2 + d_2^2} = v$ and the orientation of the aircraft in space; see Figs. 2 and 3. Substituting this coordinate change into differential

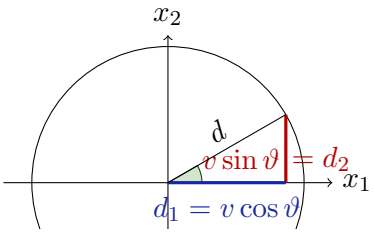

Figure 3: Reparametrize for differential axiomatization

equations (5), we immediately have $x_1' = d_1$ and $x_2' = d_2$. With the coordinate change, we further obtain differential equations for d_1, d_2 from differential equation system (5) by simple symbolic differentiation:

$$\begin{aligned} d_1' &= (v\cos\vartheta)' = v'\cos\vartheta + v(-\sin\vartheta)\vartheta' = -(v\sin\vartheta)\omega = -\omega d_2, \\ d_2' &= (v\sin\vartheta)' = v'\sin\vartheta + v(\cos\vartheta)\vartheta' \quad = \quad (v\cos\vartheta)\omega = \quad \omega d_1. \end{aligned}$$

The middle equality holds for constant linear velocity ($v' = 0$), which we assume, because only limited variations in linear speed are possible and cost-effective during the flight [TPS98, LLL00] so that angular velocity ω is the primary control parameter in air traffic control. Hence, equations (5) can be restated as the following differential equation $\mathcal{F}(\omega)$:

$$x_1' = d_1\,,\ x_2' = d_2\,,\ d_1' = -\omega d_2\ ,\ d_2' = \omega d_1 \qquad (\mathcal{F}(\omega))$$

$$y_1' = e_1\ ,\ y_2' = e_2\ ,\ e_1' = -\varpi e_2\ ,\ e_2' = \varpi e_1 \qquad (\mathcal{G}(\varpi))$$

Differential equation $\mathcal{F}(\omega)$ expresses that position $x = (x_1, x_2)$ changes according to the linear speed vector $d = (d_1, d_2)$, which in turn rotates according to ω. Simultaneous movement together with a second aircraft at $y \in \mathbb{R}^2$ having linear speed $e \in \mathbb{R}^2$ (also indicated with angle $\bar\vartheta$ in Fig. 2) and angular velocity ϖ corresponds to the differential equation $\mathcal{F}(\omega), \mathcal{G}(\varpi)$. Differential equations capture simultaneous dynamics of multiple traffic agents succinctly using conjunction.

By this *differential axiomatization*, we thus obtain polynomial differential equations. Note, however, that their solutions still involve the same complicated nonlinear trigonometric expressions so that solutions still give undecidable arithmetic [Pla10b, Appendix B]. Our proof calculus in this chapter works with the differential equations themselves and not with their solutions, so that differential axiomatization helps.

The same technique helps when handling other special functions in other cases by differential axiomatization.

10 Summary

The major lesson from today's lecture is that it can sometimes be easier to relate a variable to its initial value or to other quantities. Ghosts, in their various forms, let us achieve that by adding auxiliary variables into the system dynamics. Sometimes such ghosts are even necessary to prove properties. Although, as a workaround, it is also sometimes possible to rewrite the original model so that it already includes the ghost variables. The phenomenon that relations between state and ghost variables are sometimes easier to prove than just properties of state variables applies in either case. A secondary goal of today's lecture is, again, developing more intuition and deeper understandings of differential invariants and differential cuts.

References

[Göd31] Kurt Gödel. Über formal unentscheidbare Sätze der Principia Mathematica und verwandter Systeme I. *Mon. hefte Math. Phys.*, 38:173–198, 1931.

[LLL00] Carolos Livadas, John Lygeros, and Nancy A. Lynch. High-level modeling and analysis of TCAS. *Proc. IEEE - Special Issue on Hybrid Systems: Theory & Applications*, 88(7):926–947, 2000.

[Pla10a] André Platzer. Differential-algebraic dynamic logic for differential-algebraic programs. *J. Log. Comput.*, 20(1):309–352, 2010. doi:10.1093/logcom/exn070.

[Pla10b] André Platzer. *Logical Analysis of Hybrid Systems: Proving Theorems for Complex Dynamics*. Springer, Heidelberg, 2010. doi:10.1007/978-3-642-14509-4.

[Pla12] André Platzer. The structure of differential invariants and differential cut elimination. *Logical Methods in Computer Science*, 8(4):1–38, 2012. doi:10.2168/LMCS-8(4:16)2012.

[Pla13] André Platzer. A complete axiomatization of differential game logic for hybrid games. Technical Report CMU-CS-13-100R, School of Computer Science, Carnegie Mellon University, Pittsburgh, PA, January, Revised and extended in July 2013.

[PQ08] André Platzer and Jan-David Quesel. KeYmaera: A hybrid theorem prover for hybrid systems. In Alessandro Armando, Peter Baumgartner, and Gilles Dowek, editors, *IJCAR*, volume 5195 of *LNCS*, pages 171–178. Springer, 2008. doi:10.1007/978-3-540-71070-7_15.

[TPS98] Claire Tomlin, George J. Pappas, and Shankar Sastry. Conflict resolution for air traffic management: a study in multi-agent hybrid systems. *IEEE T. Automat. Contr.*, 43(4):509–521, 1998.

[Wal98] Wolfgang Walter. *Ordinary Differential Equations*. Springer, 1998.