

# Safe Temperature Regulation: Formally Verified and Real-World Validated

Carlos Isasa<sup>1</sup> , Noah Abou El Wafa<sup>2</sup> , Claudio Gomes<sup>1</sup> , Peter Gorm  
Larsen<sup>1</sup> , and André Platzer<sup>2</sup> 

<sup>1</sup> {cisasa, claudio.gomes, pgl}@ece.au.dk Aarhus University, Denmark

<sup>2</sup> {noah.abouelwafa, platzer}@kit.edu Karlsruhe Institute of Technology,  
Germany

**Abstract.** This paper presents a case study in the design and formal verification of a safe controller for the generic two-element lumped-capacitance model of temperature regulation, using formal cyber-physical system (CPS) theorem proving. The coupled dynamics and the absence of a fallback state reflect the complexities of real-world control systems and make it a representative challenge for theorem proving. A controller is developed by a design-by-invariant methodology. Verification using the axiomatic theorem prover KeYmaera X revealed critical assumptions for safety and pushes the frontier of CPS theorem proving. The parametric, general and provably safe controller can be applied to a wide range of temperature regulation tasks and is validated by deploying an instance of the verified controller on a physical system, demonstrating its robustness under model inaccuracies and confirming its real-world usability.

## 1 Introduction

From nuclear power plants to organ preservation, unsafe temperature changes lead to life-threatening outcomes. Despite the diverse domains, these systems share a common challenge: the need for safe and ideally verified controllers to maintain a safe temperature. Recent advances in formal theorem proving for cyber-physical systems (CPS) [20] offer a promising approach to finding a *generic* solution to this challenge by enabling formal verification through (computer-checkable) safety proofs for parametric controller specifications, that can be used practically in a wide array of applications and provide strong safety guarantees.

This paper presents a *generic temperature regulation controller*, that is *formally verified* through an integration of CPS theorem proving into the controller design process. Moreover, the controller design is validated experimentally on a *real-world instance* of the temperature regulation system, demonstrating how theorem proving as an approach to CPS verification can be *integrated end-to-end*: from the theoretical model and controller design via the formal safety proof to a real implementation on a physical system. The temperature changes of a generic heat-exchange system are modeled by linear ordinary differential equations, which provide a practical trade-off between accuracy and complexity. While sufficiently accurate for many practical and industrial applications

[4,7,9,15], designing a provably safe controller for this model is challenging. The proven controller is at the cutting edge of deductive verification, overcoming challenges that have not been addressed in this context previously, and provides valuable general insights for verifying more complex systems.

Differential dynamic logic (dL) [18] is used to model the controller and verify correctness in the axiomatic CPS theorem prover KeYmaera X [10]. The deductive approach enables formal verification for *unbounded time* and *generic parameters*, making it possible to find a versatile controller with very strong safety guarantees. Safety is fully addressed at design time, so that online safety checks or reachability analysis are no longer strictly needed. Nonetheless these can still increase the safety margin.

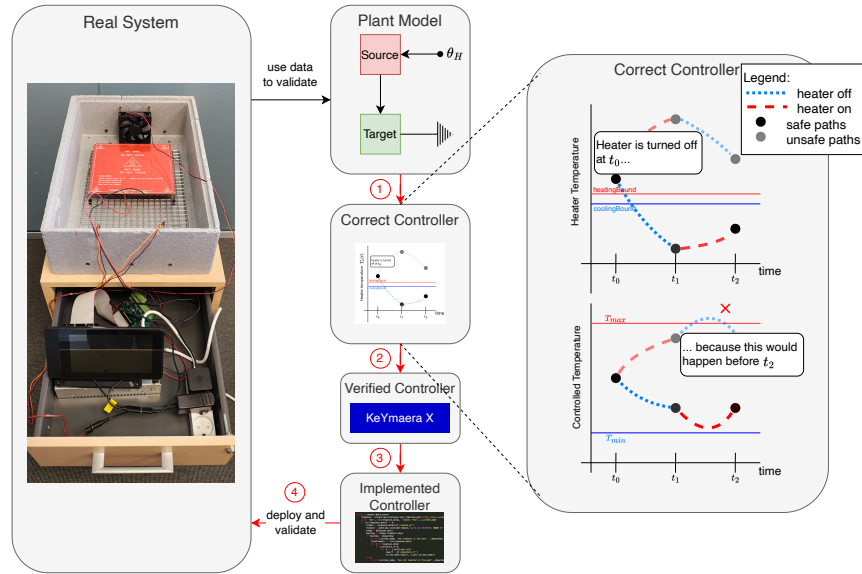
A new major difficulty of the considered system is the lack of a safe fallback action. The controller must always either choose to heat or not to heat. In the first case, it runs the risk that the temperature drops below a safe threshold and in the latter, the temperature may exceed the safe maximum. Thus, every action potentially leads to a state, from which a safety violation is inevitable. To tackle this, *directional invariants* are introduced as a new technique, which can be an ingredient in the verification of any system without a safe fallback action.

The successful validation of the verified controller on a real-world instance of the studied model confirms its correctness. By completing the circle (Figure 1) from the physical system, through modeling, control design, formal safety proof, and implementation, back to the real-world system, this paper provides a first-of-its-kind, end-to-end case study paving the way for the verification of even more complex CPSs. Furthermore, the experiments enable an empirical analysis of how safety depends on system parameters, revealing a surprising robustness of the verified controller to both calibration and state estimation errors.

*Summary of Contributions.* The red arrows in Figure 1 illustrate the contributions of this paper. A *generic provably safe controller* for temperature regulation is presented (①) and *formally verified* by a deductive proof in differential dynamic logic (②). Finally, an instance of the safe controller (③) is *validated on a real instance* of the model (④), which demonstrates both the practical feasibility of the deductive approach and the robustness of the generic approach to calibration and state estimation errors.

*Related Work.* The analyzed model has been used extensively in other scientific areas. Different techniques for the modeling of heat flow with lumped elements have been compared [1]. A similar model, including the estimation of its parameters and validation from data, was derived in detail [9], with the main difference that different types of heat transfer are considered (radiation). Model-based predictive control (MPC) approaches for heating in buildings are reviewed by Drgoña et al. [6]. MPC is complementary to this work, since the control law (from Section 3.1) can be inserted as an additional constraint in deriving an MPC, to ensure safety (see [6, Section 2.3]).

The lumped parameter model of temperature has been verified in specific



**Fig. 1.** Summary of the work. Contributions are marked by the red arrows.

instances via reachability analysis [24,15]. However, due to the need of instantiated parameters in order to carry a reachability analysis, no generic temperature regulation controller has been verified previously. Moreover, the safety guarantees presented here hold for an unbounded time horizon.

Other case studies for deductive CPS verification have been modeled and verified successfully in KeYmaera X [14,13,19,11] and using Hybrid CSP [25] and hybrid Hoare logic [16]. While end-to-end verification of CPSs has been considered [3], this work is the most complex model without fallback options using theorem proving, which is also evaluated on a real system.

## 2 Background

### 2.1 Deductive CPS Verification in dL

The formalization and verification in this paper is carried out in differential dynamic logic (dL), a language for describing models of hybrid systems with a proof calculus for the formal verification of safety properties. A brief introduction into the main ideas of dL is given here. For details see [18,10].

The logic dL can describe and reason about *discrete dynamics*, in the form of programs, combined with *continuous dynamics*, in the form of differential equations. The shape of a typical dL formula asserting the safety of a hybrid system model is

$$\text{assumptions} \rightarrow [(\text{ctrl}; \text{plant})^*] \text{ safe} \quad (1)$$

This formula consists of assertions and hybrid programs. The system  $(\text{ctrl}; \text{plant})^*$  is a *hybrid program* and serves as a nondeterministic over-approximation of a hybrid system. Hybrid programs are models, which do not deterministically describe one behavior, but rather describe a set of possible executions. In this way the model abstracts from minor details and focuses on the important aspects. And where deterministic models must consider modeling errors, a nondeterministic overapproximation should include the actual behaviour of the real system.

The condition **safe** and the **assumptions** are formulas in first-order logic (over  $\mathbb{R}$ ) defining the safe region which the system should not leave (**safe**) and the initial **assumptions** under which this should be guaranteed. In this case study the formula **safe** will be  $T_{\min} \leq T_c \wedge T_c \leq T_{\max}$  asserting that the controlled temperature  $T_c$  is in the range  $[T_{\min}, T_{\max}]$ .

Formula (1) combines these assertions and a hybrid program to formally state that under the **assumptions** every execution of the system  $(\text{ctrl}; \text{plant})^*$  remains **safe**. (This is indicated by the square brackets  $[ ]$  of the box-modality.)

*Hybrid Program Syntax* The hybrid programs in (1) and at the heart of **dL** are described here briefly. Their syntax is given by the following grammar

$$\alpha ::= x := e \mid ?\varphi \mid x' = f(x) \ \& \ \varphi \mid \alpha \cup \beta \mid \alpha; \beta \mid \alpha^*$$

where  $x$  is a variable,  $e$  a term in the variables and  $\varphi$  is a first-order formula. Assignments are represented as a hybrid program in **dL** by  $x := e$ , which discretely updates the value of the variable  $x$  to the value of the term  $e$ . The test program  $?\varphi$  restricts the nondeterminism by removing all possible execution traces which do not satisfy the formula  $\varphi$ . Most important are the *continuous programs*  $x' = f(x) \ \& \ \varphi$  of **dL**. These consist of an ordinary differential equation (ODE), which describes the continuous physical behavior of a CPS and are annotated with an *evolution domain constraint* formula  $\varphi$  to restrict the evolution of the dynamics to the set described by  $\varphi$ . The remaining hybrid programs are analogous to regular expressions. The choice program  $\alpha \cup \beta$  introduces a nondeterministic choice between  $\alpha$  and  $\beta$  and  $\alpha; \beta$  is a composition indicating that  $\beta$  is run after  $\alpha$ . The nondeterministic repetition program  $\alpha^*$  means that  $\alpha$  may be run repeatedly an arbitrary finite number of times.

The hybrid program model  $(\text{ctrl}; \text{plant})^*$  describes a typical control cycle consisting of a discrete controller **ctrl** followed by the evolution of a continuous process described by **plant** in a loop of arbitrary length indicated by  $*$ . The discrete controller **ctrl** models all possible control decisions. It relies on information provided by the system as input, performs computations and makes choices for the future execution. The communication between the controller and the plant is handled through shared variables. The **plant** describes the physical evolution of the continuous system. Typically, this takes the form of a continuous program.

*KeYmaera X* Differential dynamic logic is equipped with a proof calculus that enables *formal* (mechanized) safety proofs of formulas such as (1). The proof calculus for **dL** is implemented in the KeYmaera X theorem prover [10] for hybrid

systems. It facilitates the *interactive* theorem proving of theorems formalized in dL and is equipped with powerful *automation* techniques for proving properties of continuous dynamics [20] and finding differential invariants [22]. Proofs developed with KeYmaera X provide a witness of correctness, that can be easily checked by the small soundness-critical core of KeYmaera X.

## 2.2 Lumped-Capacitance Model

The lumped-capacitance model (LCM) describes the transmission of heat between lumped elements, ignoring the inner heat dynamics of each element. Physically, the model describes the heat transfer akin to Newton’s law of cooling [21]. By lumping elements together, the dynamics become significantly easier to handle through simulations and are simpler to study than the partial differential equations that can be used to model continuous heat distribution.

Here a controller for the two-element lumped-capacitance model (2ELCM) consisting of a heat source and a target whose temperature needs to stay between two bounds is verified. The heat source can either be powered on and thereby heat itself or be powered off. The target, under the assumption that the outside temperature is lower, continuously loses heat to the environment and heat is always transmitted from the heat source to the target.

In this model, a temperature controller cannot rely on a safe fallback action: turning off the heating is not a safe choice when the temperature is too low. Conversely, turning on the heating is not a safe choice when the temperature is too high (see Section 3.1).

The following lumped parameter heat transfer model is derived from the principles of thermodynamics [5]:

$$\begin{cases} T'_h = \frac{1}{C_h}(V \cdot I - G_h(T_h - T_c)) \\ T'_c = \frac{1}{C_c}(G_h(T_h - T_c) - G_c(T_c - T_s)) \end{cases} \quad (2)$$

where  $T_h$  is the heat source (e.g. a heater) temperature,  $T_c$  is the controlled temperature of the target,  $T_s$  is the (constant) temperature of a heat sink which models the environment,  $V, I$  are the voltage and current respectively and  $C_h, C_c, G_h, G_c$  are constant model parameters. Modeling the environment as a heat sink is justified when the environment is such that its temperature remains unaffected by temperature changes in the system. For example, the outside temperature of a building does not change (significantly) when the inside is heated up. The parameters  $C_h$  and  $C_c$  represent the heat capacitance (i.e. how fast energy is absorbed into temperature) of the heat source and the target, respectively, while the parameters  $G_h$  and  $G_c$  represent the heat transfer coefficients between the heat source and the target and between the target and the sink. Note that the rate of change of the controlled temperature  $T_c$  is proportional to the difference of the incoming heat, which itself is proportional to the temperature difference  $T_h - T_c$ , and the outgoing heat, which is itself proportional to the temperature difference  $T_c - T_s$ . The heat source temperature

behaves similarly, where the incoming heat comes from the electric power  $V \cdot I$  delivered. The parameters  $V, I, C_h, C_c, G_h, G_c$  are assumed to be positive and  $T_s \leq T_c \leq T_h$  is assumed initially, so the heat is always transferred in the same direction. This model has been created and validated on a real-world prototype [7] and will be used to validate the safe controller in Section 4.

### 3 Verified Safe Controller

Section 3.2 will formally define the full hybrid systems model of the temperature regulation system consisting of a discrete controller defined in Section 3.1 running in a control loop (similar to eq. (1)), in which `plant` follows the continuous dynamics eq. (2) and the controller is executed at least every  $\tau$  seconds. The verification of the model is discussed in Section 3.3.

#### 3.1 Verifiably Safe Controller

**Controller Requirements.** A safe controller must ensure that the temperature  $T_c$  remains within the safe temperature range  $T_{\min} \leq T_c \leq T_{\max}$ . It receives the current heat source temperature  $T_h$  and the current temperature of the heated object  $T_c$  and decides whether the heat source should be powered on or off for the next cycle (up to  $\tau$  seconds). We define the hybrid program `on`  $\equiv I := I_{on}$ , which sets the current  $I$  to  $I_{on}$ . This models turning on the heat source. Similarly, the hybrid program `off`  $\equiv I := 0$  sets the electric current  $I$  to 0 to model turning off the heat source.

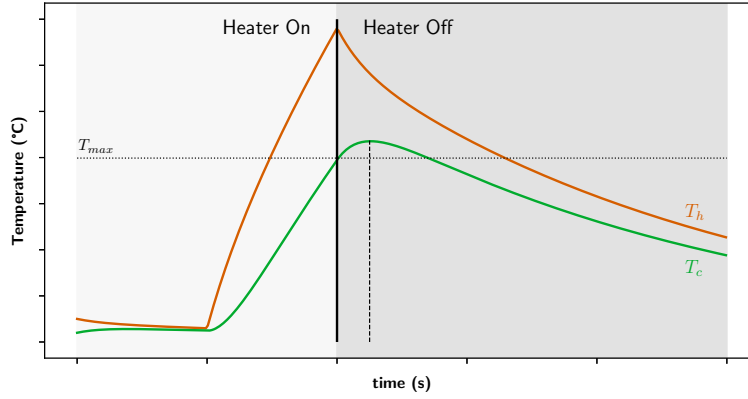
In order to obtain *provable* safety guarantees, the controller must make the choice whether to run `on` or `off` in a way that is safe and facilitates formal deductive reasoning. Abstractly, the controller (corresponding to the hybrid program `ctrl` in eq. (1)) is of the form

$$\text{ctrl} \equiv \{\{?(onSafe(T_h)); on\} \cup \{?(offSafe(T_h)); off\}\} \quad (3)$$

This hybrid program `ctrl` describes a *nondeterministic* controller, which can be read as follows: the controller *may* turn the heat source `on` whenever the constraint `onSafe( $T_h$ )` is satisfied and it *may* turn the heat source `off` if `offSafe( $T_h$ )` is true. Theoretically the conditions can be arbitrarily complex and a significant part of the contribution is to obtain conditions `onSafe` and `offSafe`, which depend only (polynomially) on the heat source (not the regulated target) temperature. This ensures the controller is practical (can be checked quickly at runtime), verifiably safe, and always has at least one choice available (can not get stuck).

**System Invariant.** A provably safe controller can be found through *design-by-invariant* [18], where the invariant properties that should remain true for as long as the system is running are derived first and then guide the design of the controller. Proving safety formally requires an *inductive* invariant of the system, which is strong enough to ensure its own invariance. For example, it must prevent

the temperature from reaching the upper limit of the safe range  $T_c = T_{\max}$  with too much energy in the heat source, which would make it impossible to cool down the system before safety is violated.



**Fig. 2.**  $T_c$  maximum reached after the heater is turned off ( $T_h$  maximum).

Clearly then, any invariant needs to ensure not only that  $T_c$  is in the safe region, but that it *will never* leave. Considering the upper and the lower limit of the safe region separately, this can be decomposed into two directional criteria that make up the following *directional invariant*:

- (I)  $T_{\min} \leq T_c \leq T_{\max}$
- (II) if  $T'_c \geq 0$  then the temperature will *never* exceed  $T_{\max}$
- (III) if  $T'_c \leq 0$  then the temperature will *never* drop below  $T_{\min}$

Combining (II)-(III) as a conjunction ensures that  $T_c$  never breaches the safety bounds, while crucially allowing reasoning about the bounds independently.

**Directional Invariant Refinement.** To describe an invariant formally, so that it is amenable to deductive verification, it is first necessary to capture ‘*never exceeding the bounds*’ (as used in (II) and (III)) symbolically in the form of bounds on  $T_c$ . This is done by conservatively over/underestimating the maximum and minimum values of  $T_c$  that can be reached in the next control cycle, given the current state of the system. The controller temperature  $T_c$  is locally minimal or maximal only at times  $t_e$  where  $T'_c(t_e) = 0$ . By solving the equilibrium condition  $T'_c = 0$  (recall eq. (2)) for  $T_c$ , the extremal values of  $T_c$  are seen to be of the form

$$\mathcal{E}(T_h(t_e)) \quad \text{with} \quad \mathcal{E}(T) := \frac{G_h T + G_c T_s}{G_c + G_h} \quad (4)$$

for some time  $t_e$  when the extremal value is attained. Because the exact expression for  $T_h(t_e)$  involves exponentials, it cannot be used for a dL verifiable

controller. However in the following a polynomial approximation is presented, that turns out to be practical and efficient as it is only used for short time (Section 4). Note that in case the heater is turned off ( $I = 0$ ),  $T_h$  is decreasing ( $T'_h = -G_h(T_h - T_c) \leq 0$  by eq. (2)). Since  $\mathcal{E}(T)$  is monotone, assuming that the controller can intervene at any time to turn off the heater, the maximal value of  $T_c$  is bounded above by  $\mathcal{E}(T_h)$ . As the invariant is evaluated at every point in time, the implication  $T'_c \geq 0 \rightarrow \mathcal{E}(T_h) \leq T_{\max}$  ensures (II). However, the controller can only act at specific times, and its decisions will be restricted (`onSafe` and `offSafe` defined appropriately) to take into account reaction delay to make  $\mathcal{E}(T_h) \leq T_{\max}$  invariant.

Similarly, (III) can be approximated by the implication  $T'_c \leq 0 \rightarrow \mathcal{E}(T_h) \geq T_{\min}$ . However, this implication will ensure (III) only in case the temperature  $T_h$  always rises, whenever the heater is turned on. This assumption is discussed below, since it depends in a surprising way on the controller constraint `onSafe`. In summary, the conjunction of the above conditions forms the invariant `inv`, presented in Invariant 1.

---

**Invariant 1** Inductive Directional System Invariant.

---

$$\text{inv} \quad \left| \begin{array}{l} 1 \quad T_{\min} \leq T_c \leq T_{\max} \\ 2 \quad T'_c \geq 0 \rightarrow \mathcal{E}(T_h) \leq T_{\max} \\ 3 \quad T'_c \leq 0 \rightarrow \mathcal{E}(T_h) \geq T_{\min} \end{array} \right.$$


---

**Safe Heating Condition.** To verify that a condition `onSafe` maintains Invariant 1 inductively, the directionality of the invariant means that attention can be restricted to the upper bound  $T_{\max}$ . Specifically, Line 3 from Invariant 1 ensures that the lower bound is not violated and since  $T_h$  is rising, Line 3 remains true. Because every control cycle is of duration  $\leq \tau$ , a control decision at  $t_0$  maintains the invariant, if the invariant holds at times  $t \in [t_0, t_0 + \tau]$ . Because  $T'_h \leq \frac{VI}{C_h}$  (see eq. (2)), the heat source temperature in this scenario can be bounded above:

$$\sup_{t \in [t_0, t_0 + \tau]} T_h(t) \leq \frac{VI_{on}}{C_h} \tau + T_h(t_0) = T_h^{\max}(T_h(t_0)) \quad (5)$$

where  $T_h^{\max}(T) := \frac{VI_{on}}{C_h} \tau + T$ . Consequently, by eq. (4) it is safe to turn on the heater if  $\mathcal{E}(T_h^{\max}(T_h(t_0))) \leq T_{\max}$ . By solving this expression for  $T_h(t_0)$ , a *decision bound*  $B_{\text{heat}}$  (heating bound) is obtained such that  $T_h(t_0) \leq B_{\text{heat}}$  guarantees that the controlled temperature will not overshoot  $T_{\max}$ . Defining

$$\text{onSafe}(T_h(t_0)) \equiv T_h(t_0) \leq B_{\text{heat}} \quad (6)$$

where

$$B_{\text{heat}} := \frac{T_{\max}(G_h + G_c) - G_c T_s}{G_h} - \frac{VI_{on} \tau}{C_h}$$



ensures that for the controller (3) line 3 of Invariant 1 is an inductive invariant. Note that the heating bound conveniently depends only on the system parameters  $G_h$ ,  $G_c$ ,  $C_h$ ,  $V$ ,  $I_{on}$ ,  $T_{max}$ ,  $T_s$ ,  $\tau$  and *not* on the system state  $T_c$ ,  $T_h$ .

**Safe Cooling Condition.** The derivation of the condition `offSafe` is more subtle as it critically relies on the assumption that the temperature  $T_c$  rises immediately once the heater is turned on. Using the lower bound

$$\inf_{t \in [t_0, t_0 + \tau]} T_h(t) \geq T_h(t_0) + \frac{G_h T_h(t_0)}{C_h} \cdot \tau = T_h^{\min}(T_h(t_0)) \quad (7)$$

where  $T_h^{\min}(T) := T + \frac{G_h T}{C_h} \cdot \tau$ , it is safe to turn off the heater if  $\mathcal{E}(T_h^{\min}(T_h(t_0))) = T_{\min}$ . Solving for  $T_h(t_0)$  assuming  $C_h > G_h \tau$  (see below), yields the (cooling) bound  $B_{cool}$

$$B_{cool} := \frac{T_{\min}(G_h + G_c) - G_c T_s}{G_h} \frac{C_h}{C_h - G_h \tau}$$

such that  $\mathcal{E}(T_h^{\min}(T_h(t_0))) \geq T_{\min}$  is true exactly if the decision guard

$$\text{offSafe}(s) \equiv T_h(t_0) \geq B_{cool} \quad (8)$$

holds. This ensures that the temperature does not drop below the lower limit of the safe range. Like the heating bound, the cooling bound  $B_{cool}$  also depends only on the system parameters  $G_h$ ,  $G_c$ ,  $C_h$ ,  $T_{\min}$ ,  $T_s$ ,  $\tau$ .

The two decision guards, `onSafe` and `offSafe`, completely define the non-deterministic controller `ctrl`. Invariance of `inv` and suitability of the controller are discussed in Section 3.3. The controller is simple to instantiate, as it only adds polynomial constraints (the decision bounds  $B_{heat}$  and  $B_{cool}$ ) on turning the heat source on and off, which are easy to implement and can be verified via theorem proving.

**Heater Power Assumption.** For Line 2 of Invariant 1 it was assumed that the heater immediately starts heating up after being turned on. In order to formally verify the controller, this assumption needs to be made symbolically. Interestingly, this assumption can depend on the definition of the controller, since the heater temperature  $T_h$  does not need to rise immediately whenever the heater is turned on, but  $T_h$  must rise when the heater has been turned on *by the controller* and the invariant `inv` holds. In other words it suffices to show that in case the invariant `inv` holds ( $T_c \geq T_{\min}$ ), the heating condition is satisfied (`onSafe`( $T_h(t_0)$ ) is true) and the heater is turned on ( $I = I_{on}$ ), the heater temperature rises immediately ( $T_h'(t_0) \geq 0$ ). This implication is guaranteed by the following assumption on the heater power:

$$VI_{on} \geq T_{\max}(G_h + G_c) - G_c T_s - G_h T_{\min} \quad (9)$$

*Proof.* It is shown that eq. (9) implies that the heater has sufficient power to immediately heat the box once it is turned on. Observe first that

$$VI_{on}(1 + \frac{G_h}{C_h} \tau) > VI_{on} \geq T_{\max}(G_h + G_c) - G_c T_s - G_h T_{\min}$$

Rearranging gives

$$\begin{aligned} VI_{on} &> T_{\max}(G_h + G_c) - G_c T_s - G_h T_{\min} - VI_{on} \frac{G_h}{C_h} \tau \\ &= G_h \left( \frac{T_{\max}(G_h + G_c) - G_c T_s}{G_h} - T_{\min} - \frac{VI}{C_h} \tau \right) \\ &= G_h (B_{\text{heat}} - T_{\min}) \end{aligned}$$

So since  $\text{onSafe}(T_h(t_0))$  is true,  $VI_{on} > G_h(T_h(t_0) - T_{\min})$ . By definition of the dynamics (eq. (2)):

$$T'_h(t_0) = VI_{on} - G_h(T_h(t_0) - T_{\min}) > 0$$

Thus the heater temperature rises immediately when the heater is turned on.  $\square$

The fact that the assumptions shown in eq. (9) are parametric in the controller variables  $V, I_{on}$ , allows for flexibility in engineering the system. Depending on the system parameters the power of the heat source can be chosen to satisfy eq. (9).

The derivation of the minimum heater power illustrates the power of the fully symbolic approach to verification. The increased complexity of the manual proof work is redeemed by its easy formal *interpretability*.

**Liveness.** For the verified model to be meaningful, the controller should always have a choice available. If  $B_{\text{heat}} \geq B_{\text{cool}}$  is true, the controller can always safely turn the heater on or off. Since both constants depend only on the parameters, liveness of the controller does *not* depend on the state of the system and so can be established at design time. And conversely, the parametric expressions of  $B_{\text{heat}}$  and  $B_{\text{cool}}$  help with making appropriate choices of for the heater power to ensure liveness. This is the reason the safety conditions were expressed in the form of heating and cooling bounds For the real-system parameters used in the validation (see Table 1), it is the case that  $B_{\text{heat}} \geq B_{\text{cool}}$ .

**Summary.** The conditions  $\text{onSafe}(T_h)$  and  $\text{offSafe}(T_h)$  were defined in terms of the decision bounds,  $B_{\text{heat}}$  and  $B_{\text{cool}}$  (Figure 1) and ensure that  $T_c$  will remain in the safe region. It is always possible to either cool down or heat up the system before  $T_c$  leaves the safe region. Which choice is available depends only on the temperature of the heat source,  $T_h$ . Perhaps surprisingly, restricting the controller decision guards,  $\text{onSafe}$  and  $\text{offSafe}$  to the heat source temperature makes the safety proof significantly easier, while still yielding a functional and safe controller, as will be demonstrated in Section 4.

### 3.2 Formal dL Model

The formal dL hybrid systems model of the 2ELCM is a control loop as in eq. (1). The postcondition **safe** is the safety condition, which says that the temperature of the target remains within the safe range  $T_{\min} \leq T_c \leq T_{\max}$ , where  $T_c$  denotes the

temperature of the heated object. The controller `ctrl` has been defined in eq. (3) and the heat dynamics describes the physical evolution of the system according to eq. (2). The precondition `assumptions` is the conjunction of the conditions listed in Listing 1, as obtained in the design process of the controller in Section 3.1 and the assumption that the system is in a safe starting state. (Line 3 captures the parametric liveness assumption as used in the derivation of  $B_{\text{cool}}$  and the heater power assumption from eq. (9).) The precondition `assumptions` is a conjunction of the conditions in Listing 1.

---

**Listing 1** Pre- and postconditions for safe controller.

---

assumptions	1	$G_h, G_c, C_c, C_h, I_{on}, V > 0 \wedge B_{\text{cool}} \leq T_h \leq B_{\text{heat}}$
	2	$T_{\min} \leq T_c \leq T_{\max} \wedge 0 \leq T_s \leq T_c \leq T_h 0$
	3	$C_h > G_h \tau \wedge VI_{on} > T_{\max}(G_h + G_c) - G_c T_s - G_h \cdot T_{\min}$
safe	4	$T_{\min} \leq T_c \leq T_{\max}$

---

The formal model of the continuous heat dynamics side of the hybrid system model is shown in Listing 2. In order to include the time-triggered nature of the controller, an explicit timer is introduced ( $t'=1$ ), which is reset ( $t:=0$  in Line 5 of Listing 2) at the beginning of every control cycle. The timer is used inside the evolution domain constraint ( $t \leq \tau$ ) to ensure that the controller is executed at least  $\tau$  seconds after the last controller execution. As the ODE has no explicit dependence on time, the timer does not affect the behaviour of the dynamics. The continuous plant dynamics are separated into two phases in order to make case-distinction reasoning possible in the verification process. It splits the phase `tempRise` in which the temperature is rising  $T'_c \geq 0$  and the phase `tempFall` where it is falling  $T'_c \leq 0$ . Switching between these phases (*ghost switching* [23]) is realized through the use of a nondeterministic choice facilitating the phase-transition and a loop allowing any execution to make an arbitrary (finite) number of phase transitions.

---

**Listing 2** Continuous dynamics model with ghost switching.

---

ODE	1	$T'_h = 1/C_h(VI - G_h(T_h - T_c)),$
	2	$T'_c = 1/C_c(G_h(T_h - T_c) - G_c(T_c - T_s)),$
tempFall	3	$\{\text{ODE}, t' = 1 \wedge t \leq \tau \wedge T'_c \leq 0\}$
tempRise	4	$\{\text{ODE}, t' = 1 \wedge t \leq \tau \wedge T'_c \geq 0\}$
plant	5	$\{t := 0; \{\text{tempFall} \cup \text{tempRise}\}^*\}$

---

### 3.3 Formal Verification of Safety

Any minor oversight in the design of the controller could lead to an unsafe controller leading to potentially dangerous outcomes. This possibility is excluded by a formal proof of safety carried out in KeYmaera X. The formal proofs are available in the repeatability package [12] and the proof tactic serves as an easily checkable witness of correctness. As such the proof also certifies the correctness of the controller to a third party, which would only need to trust the small proof-checking core of KeYmaera X [2] and the accuracy of the model itself.

This section describes some of the key challenges and insights required for the formal proof of safety and in particular their *general* lessons for modeling and verifying cyber-physical systems formally.

*Verification Process* The correct controller was obtained through several design and proof iterations. Many critical assumptions were made implicitly in the design of the controller and were only revealed through the process of constructing a formal proof. (See for example the assumption on the minimum power of the heater on Section 3.1.)

*Case Distinctions and Local Invariants* At the heart of the formal proof of safety lies the verification that Invariant 1 indeed is an *inductive invariant* of the system. Note that the invariant mentions  $T_{h,\text{init}}$  as the initial temperature of the heat source at the beginning of the control cycle (which can be introduced as a constant using a discrete ghost argument [18]). Ghost switching enables a formal proof by case distinction into four distinct cases along two dimensions: The heat source may be powered on or off and the temperature may be rising or falling. The *directional invariants* are a new ingredient that fully leverage the case distinction to reduce a complex property to its core and discharge many proof obligations using the advanced automation of KeYmaera X. The idea to make the invariant conditional on the direction of the evolution (cf. Lines 4 and 5 in Invariant 2) constitutes a novel approach in deductive CPS verification and is promising for many other cases. In particular, it is a powerful technique for treating the challenges posed by the lack of a safe fallback controller action.

The effect of the case distinction and the directional invariant is that the temperature  $T_c$  and the heat source temperature  $T_h$  can be considered (temporarily) monotone, which enables formalization of symbolic estimates of the temperatures, as used in the design of `onSafe` and `offSafe` in Section 3.1. To exploit the monotonicity syntactically, a key ingredient in the proof are the two local invariants, which remain true of the physical system only for the duration of one control cycle, one for each of the two modes of the controller. These local invariants are essentially the invariants for the continuous dynamics (differential invariants), with the difference that, thanks to the ghost switching they only need to be verified until the temperature reaches a critical point. The local invariant for the heating cases (Invariant 2) captures additionally that the heater temperature will be hotter than it is initially throughout the control cycle and is based on the over-approximation argument of the temperature. The directional

nature of the invariant determines whether the next extremal temperature is maximal or minimal. The local invariant for cooling is similar.

---

**Invariant 2** Local Invariant for Heating Cycle.

---

heatingInv	1	$T_{\min} \leq T_c \leq T_{\max} \wedge 0 \leq t \leq \tau$
	2	$0 \leq T_s \leq T_c \leq T_h \wedge T_{h,\text{init}} \leq T_h$
	3	$T_h^{\min}(T_{h,\text{init}}) \leq T_h \leq T_h^{\max}(T_{h,\text{init}})$
	4	$T_c' \leq 0 \rightarrow (T_c \geq \mathcal{E}(T_{h,\text{init}}) \wedge \mathcal{E}(T_{h,\text{init}}) \geq T_{\min})$
	5	$T_c' \geq 0 \rightarrow (T_c \leq \mathcal{E}(T_h^{\max}(T_{h,\text{init}})) \wedge \mathcal{E}(T_h^{\max}(T_{h,\text{init}})) \leq T_{\max})$

---

*Proof Technology.* The marked improvements in the theorem-proving technology for hybrid systems are critical for the successful formal verification. Of particular importance are recent advances in the automatic *complete* verification of differential invariants [20]. This obviates the need for complex reasoning about the continuous evolution. As long as a valid differential invariant can be found, it can be verified automatically. KeYmaera X implements invariance verification in the tactic `odeInvC`, which automatically proves any valid differential invariant for a system. However, for computational reasons, it is still necessary to simplify the invariance problem to avoid time outs. This requires controlling exactly which assumptions are required for invariance. In the `artificat` package [12] a part of the proof is explained in depth to illustrate the subtleties of reducing a complex problem to one that can be solved *efficiently* and *automatically*.

Another valuable tool in the verification process was the counterexample finding tool of KeYmaera X. In the process of designing and verifying the safe controller refinement, counterexamples made clear which assumptions were missing and highlighted possible scenarios that were overlooked.

## 4 Control Deployment and Validation

The controller was validated on the incubator described by Feng et al. [7] consisting of an insulated box containing a heater (the heat source in the 2ELCM) and a fan to distribute the heat (so that it can be modeled as a single lump). The heater can be turned on or off and the inside air temperature ( $T_c$ ) is viewed as the average of two sensor readings.

An important feature of this case study is that the heatbed temperature  $T_h$  is not measured directly. Instead, it is estimated through a Kalman filter [8] that uses the same model used for the proof of the controller. We evaluate in practice the impact of the uncertainty in this estimation and the controller performance.

To test the limits of the bounds, the physical controller is a refinement of the (nondeterministic) dL specification, which heats for as long as possible and then cools for as long as possible, by activating the heater exactly if:

1. either the heater was previously on and it is safe to leave it on; or
2. the heater was previously off and it is unsafe to leave it off.

This example illustrates the flexibility of the generic, verified controller, which allows many alternative implementations, which optimize for other measures and can be designed manually. A correct-by-construction controller monitor that checks at runtime that any choice made by the controller is allowed by the specification can be obtained through Modelplex [17].

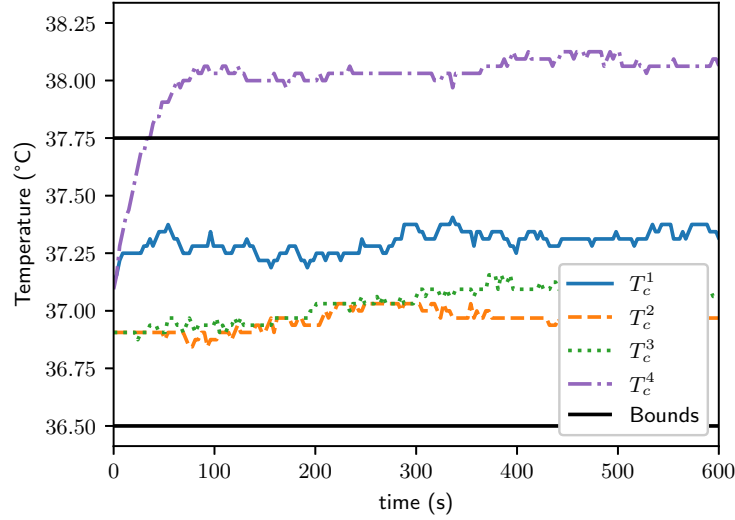
**Table 1.** Parameter sets used in the experiments. (The parameters are approximate and exact values can be found in the repeatability package [12].)

	Parameters				Power		Bounds		Source
	$C_c$	$G_c$	$C_h$	$G_h$	V	$I_{on}$	$B_{heat}$	$B_{cool}$	
$\Gamma_1$	24.59	0.16	47.26	0.22	12.17	1.55	49.47	49.17	Calibration Process
$\Gamma_2$	17.63	0.16	40.01	0.13	12.17	1.55	57.56	56.75	Calibration Process
$\Gamma_3$	34.63	0.16	40.01	0.13	12.17	1.55	58.02	57.22	$\Gamma_2$ with $C_c$ Disturbed
$\Gamma_4$	24.59	0.25	47.26	0.22	12.17	1.55	55.92	55.24	$\Gamma_1$ with $G_c$ Disturbed

*Experimental Setup.* Four distinct experiments with parameter sets  $\Gamma_1, \Gamma_2, \Gamma_3$  and  $\Gamma_4$ , shown in Table 1, were conducted to validate the controller on a real system with tight safety margins (parameter sets  $\Gamma_1$  and  $\Gamma_2$ ), and to investigate the impact of modeling errors and parameter variations (parameter sets  $\Gamma_3$  and  $\Gamma_4$ ). The parameters are used in the controller decision process and in the Kalman filter for state estimation. For the experiments the temperature bounds  $T_{min} = 3.5^\circ\text{C}$  and  $T_{max} = 37.75^\circ\text{C}$  were used for a range of  $1.25^\circ\text{C}$ . The first two parameter sets,  $\Gamma_1$  and  $\Gamma_2$  were obtained through a calibration process [7], which fits the parameters to an experimental run of the physical system. This process involves an underconstrained optimization problem, so that two different sets of parameters ( $\Gamma_1$  and  $\Gamma_2$ ), representing different, well-fitting models, are obtained. To study the effect of perturbations the most and least sensitive parameters were determined by a sensitivity analysis over  $B_{heat}$  and  $B_{cool}$  (computing partial derivatives of eqs. (6) and (8)). Parameter set  $\Gamma_3$ , was obtained by adding 17 to the value of the least sensitive parameter  $C_c$  in  $\Gamma_2$  (almost doubling it) and  $\Gamma_4$  is  $\Gamma_1$  with the value of the most sensitive parameter  $G_c$  multiplied by 1.5 in  $\Gamma_1$ .

The measured (average) air temperatures of the real-world experiments are overlaid in Figure 3 from the moment both sensor readings first exceed  $T_{min}$ .

*Discussion.* As verified deductively, fig. 3 shows that the controlled temperature ( $T_c^1$  and  $T_c^2$ ) remains within the bounds in both experiments ( $\Gamma_1, \Gamma_2$ ). Interestingly the verification result applies to very different values for the decision bounds  $B_{heat}$  and  $B_{cool}$  (see Table 1). These experiments provide evidence that our controller remains safe, despite typical modeling, calibration, and estimation uncertainties.



**Fig. 3.** Average incubator temperature  $T_c^i$  for each experiment with parameters  $I_i$ .

Interestingly, the temperature  $T_c^3$  also stays within the safety bounds. This is surprising, as a perturbation to one of the parameters (capacitance  $C_c$ ) might intuitively make the system unsafe. However, the capacitance  $C_c$  does not affect the heating and cooling bounds at all, so that, according to the model and the experiment, safety is not affected by changes to the capacitance  $C_c$ , such as placing objects inside the air volume. This is particularly relevant for applications where it is unknown what objects will be placed inside the temperature controlled environment.

Finally, the temperature  $T_c^4$  fails to stay within the safety bounds. The perturbation of the coefficient  $G_c$  (by 50%) makes the system unsafe, by virtue of the larger significance of the affected parameter. While the decision bound  $B_{\text{heat}}$  (see Table 1) is similar to the one in  $I_2$  the reliance of the Kalman filter on the same parameters means that the estimate of the temperature  $T_h$  does not fit the model anymore.

## 5 Conclusion

This paper introduced a formally verified controller for temperature regulation in the two-element lumped-capacitance model. The requirements of formal verification guided the design of the controller and revealed all critical assumptions on the heating system for the system to be provably safe. The controller was designed using several correct-by-construction over-/under-approximations of the safety bounds in order to obtain verifiably correct decision bounds. These approximations are correct for the dynamical system that models the real-world dynamics and thus correct for the real world dynamics assuming that the approximation is good enough. This process ensures safety for unbounded time

and presents a representative verification case study of a controller that does not rely on a safe fallback action. Safety is formalized in the hybrid systems theorem prover KeYmaera X, which delivers easily checkable correctness guarantees. The resulting controller is generic and allows for the straightforward implementation of a safe temperature-regulation unit in many application domains, as demonstrated by the real-world validation of the controller. The validation provides an end-to-end case study for CPS verification of a hybrid system with dynamics of general interest.

Furthermore, the validation on a real system shows the strength and applicability of this approach. The complexities of the formal approach are vindicated in the implementation and validation phase, where the verified bounds reveal which parameters are the most relevant to the inner workings of the system. This is validated by applying a large disturbance to the least relevant parameter ( $G_c$ ) and showing that it does not influence the safety of the system. All in all, this process minimizes calibration problems and thus allows for more robust implementations. Initially, the potential of modeling discrepancies and the fact that the controller requires a Kalman filter, are concerning, since the correctness hinges on the 2ELCM being a good fit for the system. However, it is shown that controller safety is quite robust with respect to modeling inaccuracies.

*Future Work.* This work opens up several exciting areas of research. As a real-world case study, the 2ELCM model provides a valuable aid to the development of advanced hybrid systems verification techniques. The surprising robustness of the presented controller with respect to parameter errors suggest an investigation of the impact that calibration errors and state estimation (the Kalman filter) can have on the safety of a formally verified CPSs beyond 2ELCM.

The conditions on the safe controller use an overapproximation of the maximal heater temperature. Alternative controller bounds that are derived similarly could be explored, using alternative assumptions and better bounds. This could achieve practical improvements with more efficient control envelopes at the expense of increased proof complexity.

The presented controller solves many challenges at the heart of the heat-transmission problem. Although still challenging, increasing the complexity of the model to more elements is now possible on the basis of the verified two-element controller. This makes it an interesting object for further research into the composability of deductive hybrid systems verification.

The approach used in the deductive verification has many generalizable lessons and critical insights that are useful in other cases, in which safety can not be maintained by simply retaining a safe fallback option at all times. This is potentially useful for other verification tasks and could benefit from being studied more generally.

**Acknowledgments** This research was supported by the RoboSAPIENS Project financed by the European Commission’s Horizon Europe programme under Grant 101133807 and the Alexander von Humboldt Professorship program.



## References

1. Afroz, Z., Shafiullah, G., Urmee, T., Higgins, G.: Modeling techniques used in building hvac control systems: A review. *Renewable and Sustainable Energy Reviews* **83**, 64–84 (2018). doi: [10.1016/j.rser.2017.10.044](https://doi.org/10.1016/j.rser.2017.10.044)
2. Bohrer, R., Rahli, V., Vukotic, I., Völz, M., Platzer, A.: Formally verified differential dynamic logic. In: Bertot, Y., Vafeiadis, V. (eds.) *Certified Programs and Proofs - 6th ACM SIGPLAN Conference, CPP 2017, Paris, France, January 16–17, 2017*. pp. 208–221. ACM (2017). doi: [10.1145/3018610.3018616](https://doi.org/10.1145/3018610.3018616)
3. Bohrer, R., Tan, Y.K., Mitsch, S., Myreen, M.O., Platzer, A.: VeriPhy: Verified controller executables from verified cyber-physical system models. In: Grossman, D. (ed.) *PLDI*. pp. 617–630. ACM, Philadelphia (2018). doi: [10.1145/3192366.3192406](https://doi.org/10.1145/3192366.3192406)
4. Buhagiar, A.J., Freitas, L., III, W.E.S., Larsen, P.G.: Digital twins for organ preservation devices. In: Margaria, T., Steffen, B. (eds.) *ISoLA 2022, Rhodes, Greece, October 22–30, 2022, Proceedings, Part IV. Lecture Notes in Computer Science*, vol. 13704, pp. 22–36. Springer (2022). doi: [10.1007/978-3-031-19762-8\\_3](https://doi.org/10.1007/978-3-031-19762-8_3), [https://doi.org/10.1007/978-3-031-19762-8\\_3](https://doi.org/10.1007/978-3-031-19762-8_3)
5. Cengel, Y.A., Boles, M.A., Kanoğlu, M.: *Thermodynamics: an engineering approach*, vol. 5. McGraw-hill New York, New York, USA (2011)
6. Drgoña, J., Arroyo, J., Cupeiro Figueroa, I., Blum, D., Arendt, K., Kim, D., Ollé, E.P., Oravec, J., Wetter, M., Vrabie, D.L., Helsen, L.: All you need to know about model predictive control for buildings. *Annual Reviews in Control* **50**, 190–232 (2020). doi: [10.1016/j.arcontrol.2020.09.001](https://doi.org/10.1016/j.arcontrol.2020.09.001)
7. Feng, H., Gomes, C., Thule, C., Lausdahl, K., Sandberg, M., Larsen, P.G.: The incubator case study for digital twin engineering (2021), <https://arxiv.org/abs/2102.10390>
8. Feng, H., Gomes, C., Larsen, P.G.: Model-based monitoring and state estimation for digital twins: The Kalman filter (2023), <https://arxiv.org/abs/2305.00252>
9. Frahm, M., Langner, F., Zwickel, P., Matthes, J., Mikut, R., Hagenmeyer, V.: How to derive and implement a minimalistic RC model from thermodynamics for the control of thermal parameters for assuring thermal comfort in buildings. In: *2022 Open Source Modelling and Simulation of Energy Systems (OSMES)*. pp. 1–6. IEEE, Aachen, Germany (Apr 2022). doi: [10.1109/osmses54027.2022.9769134](https://doi.org/10.1109/osmses54027.2022.9769134)
10. Fulton, N., Mitsch, S., Quesel, J.D., Völz, M., Platzer, A.: KeYmaera X: An axiomatic tactical theorem prover for hybrid systems. In: Felty, A.P., Middeldorp, A. (eds.) *CADE. LNCS*, vol. 9195, pp. 527–538. Springer, Berlin, Germany (2015). doi: [10.1007/978-3-319-21401-6\\_36](https://doi.org/10.1007/978-3-319-21401-6_36)
11. Garcia, L., Mitsch, S., Platzer, A.: HyPLC: Hybrid programmable logic controller program translation for verification. In: Bushnell, L., Pajic, M. (eds.) *ICCPs*. pp. 47–56 (2019). doi: [10.1145/3302509.3311036](https://doi.org/10.1145/3302509.3311036)
12. Isasa, C., Abou El Wafa, N., Platzer, A., Larsen, P.G., Gomes, C.: Artifact for Safe Temperature Regulation: Formally Verified and Real-World Validated. doi: [10.6084/m9.figshare.28869218](https://doi.org/10.6084/m9.figshare.28869218)
13. Jeannin, J., Ghorbal, K., Kouskoulas, Y., Gardner, R., Schmidt, A., Zawadzki, E., Platzer, A.: Formal verification of ACAS X, an industrial airborne collision avoidance system. In: Girault, A., Guan, N. (eds.) *EMSOFT*. pp. 127–136. IEEE Press, Amsterdam, Netherlands (2015). doi: [10.1109/EMSOFT.2015.7318268](https://doi.org/10.1109/EMSOFT.2015.7318268)
14. Kabra, A., Mitsch, S., Platzer, A.: Verified train controllers for the Federal Railroad Administration train kinematics model: Balancing competing brake and track

- forces. *IEEE Trans. Comput. Aided Des. Integr. Circuits Syst.* **41**(11), 4409–4420 (2022). doi: [10.1109/TCAD.2022.3197690](https://doi.org/10.1109/TCAD.2022.3197690)
15. Kapuria, A., Cole, D.G.: Formal verification of a nuclear plant thermal dispatch operation using system decomposition. In: 2024 IEEE 6th International Conference on Trust, Privacy and Security in Intelligent Systems, and Applications (TPS-ISA). pp. 543–548 (2024). doi: [10.1109/TPS-ISA62245.2024.00074](https://doi.org/10.1109/TPS-ISA62245.2024.00074)
  16. Liu, J., Lv, J., Quan, Z., Zhan, N., Zhao, H., Zhou, C., Zou, L.: A calculus for hybrid csp. In: APLAS. pp. 1–15 (2010), [https://doi.org/10.1007/978-3-642-17164-2\\_1](https://doi.org/10.1007/978-3-642-17164-2_1)
  17. Mitsch, S., Platzer, A.: ModelPlex: verified runtime validation of verified cyber-physical system models. *Formal Methods in System Design* **49**(1), 33–74 (Oct 2016). doi: [10.1007/s10703-016-0241-z](https://doi.org/10.1007/s10703-016-0241-z)
  18. Platzer, A.: *Logical Foundations of Cyber-Physical Systems*. Springer, Cham (2018). doi: [10.1007/978-3-319-63588-0](https://doi.org/10.1007/978-3-319-63588-0)
  19. Platzer, A., Quesel, J.D.: European Train Control System: A case study in formal verification. In: Breitman, K., Cavalcanti, A. (eds.) *ICFEM*. LNCS, vol. 5885, pp. 246–265. Springer, Rio de Janeiro (2009). doi: [10.1007/978-3-642-10373-5\\_13](https://doi.org/10.1007/978-3-642-10373-5_13)
  20. Platzer, A., Tan, Y.K.: Differential equation invariance axiomatization. *J. ACM* **67**(1), 6:1–6:66 (2020). doi: [10.1145/3380825](https://doi.org/10.1145/3380825), <https://doi.org/10.1145/3380825>
  21. Serway, R.A., Faughn, J.S.: *College Physics*. Saunders college publishing, Philadelphia, 5th ed. edn. (1999)
  22. Sogokon, A., Mitsch, S., Tan, Y.K., Cordwell, K., Platzer, A.: Pegasus: A framework for sound continuous invariant generation. In: ter Beek, M., McIver, A., Oliviera, J.N. (eds.) *FM*. LNCS, vol. 11800, pp. 138–157. Springer (2019). doi: [10.1007/978-3-030-30942-8\\_10](https://doi.org/10.1007/978-3-030-30942-8_10)
  23. Tan, Y.K., Mitsch, S., Platzer, A.: Verifying switched system stability with logic. In: *Proceedings of the 25th ACM International Conference on Hybrid Systems: Computation and Control. HSCC '22*, Association for Computing Machinery, New York, NY, USA (2022). doi: [10.1145/3501710.3519541](https://doi.org/10.1145/3501710.3519541)
  24. Wright, T., Gomes, C., Woodcock, J.: Formally verified self-adaptation of an incubator digital twin. In: Margaria, T., Steffen, B. (eds.) *ISoLA 2022*, Rhodes, Greece, October 22–30, 2022, *Proceedings, Part IV*. LNCS, vol. 13704, pp. 89–109. Springer, Rhodes, Greece (2022). doi: [10.1007/978-3-031-19762-8\\_7](https://doi.org/10.1007/978-3-031-19762-8_7)
  25. Zou, L., Lv, J., Wang, S., Zhan, N., Tang, T., Yuan, L., Liu, Y.: Verifying chinese train control system under a combined scenario by theorem proving. p. 262–280. *VSTTE 2013*, Springer-Verlag, Berlin, Heidelberg (2013). doi: [10.1007/978-3-642-54108-7\\_14](https://doi.org/10.1007/978-3-642-54108-7_14)