Experience Report: Verifying Properties of an Electro-Mechanical Braking System [†]

Thomas Strathmann[1] and Jens Oehlerking[2]

[1]OFFIS e. V., Escherweg 2, Oldenburg, Germany,
thomas.strathmann@offis.de
[2]Robert Bosch GmbH, Stuttgart, Germany,
jens.oehlerking@de.bosch.com

April 9, 2015

Abstract

In this experience report, we apply the hybrid verification tools iSAT-ODE, Flow*, and S-TaLiRo to a case study consisting of an experimental electro-mechanical braking system. Starting from a Simulink closed-loop model, we describe the derivation of hybrid automaton models for plant and controller and give verifcation results for the different tools.

1 Introduction

In recent years, several verification tools of increasing maturity have been developed within the hybrid systems community. In this paper, we report on our experience with applying some of these tools to a case study. These tools are iSAT-ODE [ERNF12], Flow* [CÁS13], and S-TaLiRo [ALFS11].

The case study we used for this purpose is an experimental electro-mechanical braking system consisting of a plant model and a controller comprising both feedback and feedforward control. While the model itself is not used for the development of actual products, it is representative of some challenges in the development of automotive systems. Based on this example, we show model simplifications under which we were able to obtain useful results from the verification tools. A special focus here are verification results including parameter variations within the model to achieve a quantifiable form of robustness for the property in question.

Related work includes [FHQW15], where the tool SpaceEx [FLGD$^+$11] has been applied to a variant of this same case study, with special focus on timing variations of the control software. Also, in [ZYZ$^+$14] the application of iSAT-ODE and Flow* for the verification of a closed-loop control system subject to parameter variations is described.

The paper is strucured as follows. We will first describe the case study in Section 2 and then derive a simplified model from the equations of the original model in Section 3, for use with the tools iSAT-ODE and Flow*. In Section 4 we then describe modifications to this base model required for the different tools and give the verification results.

[†]This work has been partially funded by the German Ministry for Education and Research (BMBF) under the funding ID 01IS12005M (SPES_XT Project)

2 The Experimental Electro-Mechanical Braking System

The system under analysis consists of an experimental electro-mechanical braking system, together with its controller, implemented in software.

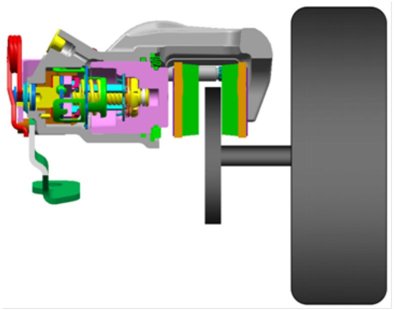


Figure 1: Schematic of the electro-mechanical brake with electrical engine on left hand side, brake disc connected to wheel on right hand side and brake caliper around the brake disc

Figure 1 shows an illustration of the the braking system. Its left hand side consists of an electrical engine, which is used to push the left side of the brake caliper against the brake disk. The brake disc is connected to the wheel, so that contact between caliper and disc will result in vehicle deceleration. Even when contact between caliper and disc has been established, the electrical engine can be used to exert additional braking force, allowing also for fine-grained control of the vehicle deceleration.

The original control software is a hybrid controller with modes for the idle state (caliper in the leftmost position), positioning modes (caliper left to right and right to left) and a force control mode with respect to an externally supplied set point. The individual control strategies per mode consist of a model-based feedforward controller and feedback through a PI-controller to account for disturbances and modeling errors. For some of the tools, we will only examine the feedback component of the controller, with others we were also able to deal with the feedforward control. Also, we assume that the braking force and the position of the caliper are readily available, while in reality they are estimated from the motor current. In general, the controller parameters are chosen with respect to two conflicting goals: a) ensuring a quick reaction to a brake request and b) minimizing the jerk of the vehicle (i.e., avoiding sudden changes of acceleration) by making the contact between caliper and disc as smooth as possible.

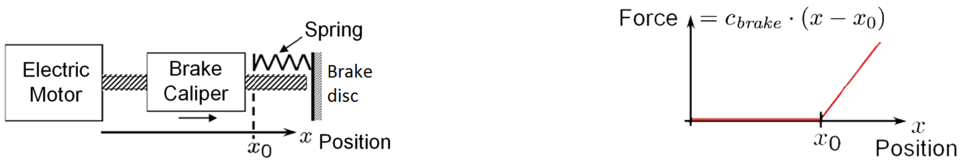
In its simplest version, the closed-loop system is numerically stiff and piecewise affine. The stiffness makes the system difficult to analyze with flowpipe-based methods. For example, in the context of [FHQW15], SpaceEx had difficulties with the dynamics of the braking mode. Furthermore, the model contains a number of parameters: the physical parameters of the brake hardware and the parameters of the PI-controller. For both these sets of parameters, parametric verification is useful in practice. Since the physical parameters of the brake are subject to wear and tear, as well as production tolerances, useful verification results are only obtained if they are robust with respect to these factors. The controller parameters are also often modified post-deployment, for example to take into account not formally modelled requirements such as vibration or noise, so that properties should ideally be verified for entire parameter ranges.

The two verification requirements taken into consideration for this experience report are:

- As soon as braking is requested, the contact between caliper and disk should occur within 23 ms.
- The brake caliper velocity upon contact should be less than 2 mm/s to limit jerk.

3 Modelling

Figure 2(a) shows a simplified model of the electro-mechanical brake. A DC-motor moves the brake caliper towards the brake disc. Once the caliper passes the position x_0, a braking force is exerted on the brake disc (cf. figure 2(b)). As a simplifying assumption the brake disc is modelled as a stiff spring.



(a) Schematic drawing of the simplified plant model

(b) Mode dependency of the brake force

Figure 2: Simplified model of the plant

The system is originally given as a Simulink model that needs to be translated into a form that is suitable for analysis with iSAT-ODE and Flow*. As a first step we identify the ordinary differential equations encoded as Simulink subsystem blocks.

$$\begin{aligned}
\dot{I} &= \frac{1}{L} \cdot (V - \tanh(100 \cdot I) \cdot V_{brush} - R \cdot I - K \cdot \omega) && (1)\\
\dot{\omega} &= \frac{1}{J} \cdot (K \cdot I - c_{gear} \cdot (\varphi - i \cdot x) - d_{rot} \cdot \omega) && (2)\\
\dot{\varphi} &= \omega && (3)\\
\dot{v} &= \frac{1}{m} \cdot (c_{gear} \cdot (\varphi - i \cdot x) \cdot i - c_{brake} \cdot x_1 - d_{trans} \cdot v) && (4)\\
\dot{x} &= v && (5)
\end{aligned}$$

There are five continuous state variables in total: The voltage V applied to the motor by the controller, the motor current I, the angular velocity ω and angle φ of the motor shaft, as well as the velocity v and position x of the brake caliper. The auxiliary variable x_1 depends on whether there is contact between the brake caliper and disc.

$$x_1 = \begin{cases} x - x_0 & \text{if } x \geq x_0 \\ 0 & \text{otherwise} \end{cases} \qquad (6)$$

Because the properties we are interested in in this report only deal with the mode where there is no contact, we will disregard the situation where $x > x_0$ and consequently assume $x_1 = 0$ in the following.

As preliminary experiments using both Flow* and iSAT-ODE with these equations posed significant problems that caused the numerical computation of the flowpipes to get stuck in several instances (sometimes aborted due to memory exhaustion), the plant model subsequently used in the closed-loop system model is based on a simplified version of equations (1) through (6). Although the behaviour of the simplified system quantitatively differs from the full model, it poses the same numerical challenges due to the stiffness of the ODEs. By letting $\varphi = i \cdot x$ (where i is the transmission ratio of the gear box that translates the rotational movement of the motor into the linear movement of the brake caliper) and assuming $\dot{\omega} = 0$ (because $\dot{\omega}$ is negligible compared to $K \cdot I$) we derive $\omega = \frac{K}{d_{rot}} \cdot I$. And because $V_{brush} = 0$ in the original model[1] we arrive at

$$\dot{I} \quad = \quad \frac{1}{L} \cdot (V - R \cdot I - \frac{K^2}{d_{rot}} \cdot I). \tag{7}$$

Finally, we combine equations (2) through (5) using the physical identity $\frac{\omega}{i} = v$ to derive the second of the simplified ODEs:

$$\dot{x} \quad = \quad \frac{K}{d_{rot} \cdot i} \cdot I \tag{8}$$

Given equations (7) and (8) we can define a hybrid automaton for the closed-loop system where the voltage is set by a PI-controller that takes as input the difference between the reference position x_0 and the actual position x of the brake caliper.

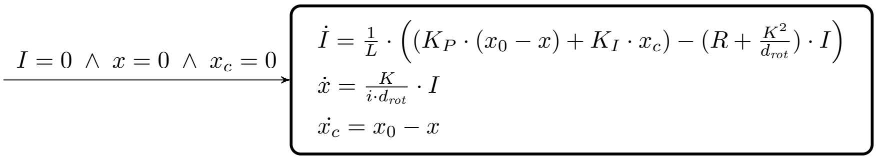

$I = 0 \wedge x = 0 \wedge x_c = 0$
$\dot{I} = \frac{1}{L} \cdot \Big((K_P \cdot (x_0 - x) + K_I \cdot x_c) - (R + \frac{K^2}{d_{rot}}) \cdot I\Big)$
$\dot{x} = \frac{K}{i \cdot d_{rot}} \cdot I$
$\dot{x_c} = x_0 - x$

Figure 3: ODEs of the closed-loop model of the plant with a continuous-time PI-controller depicted as a hybrid automaton

As the controller is actually implemented in software, we also define a hybrid automaton (cf. Figure 4) that incorporates a simple discrete-time version of the PI-controller which works by uniform sampling of the control error and calculation of the integral part by explicit Euler integration. The result is a sampled-data closed loop system. Note that, for fixed sampling, the resulting system could also be represented by a discrete-time linear system, which is generally easier to analyze. The reason the model is in the form of a hybrid automaton is that, by changing the update of the clock variable T to a non-deterministic assignment $T' :\in [-\zeta, \zeta]$ for a small constant $\zeta \ll T_{sample}$ we can also arrive at a simplistic model of sampling jitter.

4 Verification

In this section we describe in more detail some of the tool-specific modelling choices as well as results from selected experiments we performed with these tools including performance figures. All experiments with Flow* and iSAT-ODE were performed on a computer with 2.3 GHz AMD Opteron 6376 processor and 512 GiB RAM

[1] V_{brush} represents the losses due to friction in the engine, which were seen as negligible and parameterised with 0 in the Simulink model.

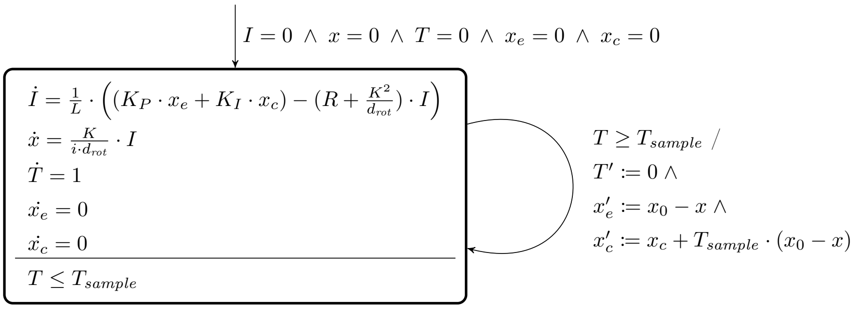

$I = 0 \wedge x = 0 \wedge T = 0 \wedge x_e = 0 \wedge x_c = 0$
$\dot{I} = \frac{1}{L} \cdot \Big((K_P \cdot x_e + K_I \cdot x_c) - (R + \frac{K^2}{d_{rot}}) \cdot I \Big)$
$\dot{x} = \frac{K}{i \cdot d_{rot}} \cdot I$
$\dot{T} = 1$
$\dot{x_e} = 0$
$\dot{x_c} = 0$
$T \leq T_{sample}$
$T \geq T_{sample}$ /
$T' := 0 \wedge$
$x_e' := x_0 - x \wedge$
$x_c' := x_c + T_{sample} \cdot (x_0 - x)$

Figure 4: Hybrid automaton of the plant with discrete-time PI-controller

running GNU/Linux. The experiments with S-TaLiRo were performed under Matlab/Simulink R2014a 64-Bit on a computer with 3.4 GHz Intel Core i5-3570 processor and 8 GiB RAM running 64-Bit Windows 7 Professional.

4.1 iSAT-ODE

iSAT-ODE [ERNF12] is a tool for bounded model checking of non-linear hybrid systems based on methods from SAT and constraint solving as well as numerical methods for ordinary differential equations. The hybrid system must be given in a predicative encoding, i.e. the initial state, final state, and transition relation are defined by first-order logic formulae. The ODE extension supports the safe enclosure of solutions of ODEs expressed as derivatives with respect to the variable `time`. In our encoding of the discrete-time automaton in Figure 4 the existence of this global time variable obviates the need for an additional timer variable T to model the sampling and discrete computations.

Because iSAT-ODE works by splitting intervals, the initial ranges of the continuous state variables of the model should be bounded as tightly as possible in order to reduce the size of the reachable state space. We used a Simulink simulation of the simplified model used in both the iSAT-ODE and Flow* experiments to derive reasonably tight yet conservative bounds for the state variables. A plot of this simulation is shown in Figure 5. From this plot we can also deduce the time point $t_0 \approx 0.146$ where the caliper makes contact with the disc. We can use this as a reference value to examine the different behaviours of the system that arise under variations of the parameters.

Requirement 1 can be expressed as the formula `abs(x - 0.05) <= 0.002 and time <` t_0 that characterises the state whose reachability iSAT-ODE checks. On the continuous-time model without parameter variations this property can be verified in 3 seconds. When the resistance R can vary between its nominal value of 0.5 Ω and 0.7 Ω, iSAT-ODE can verify in 312 seconds that the response time of the system is slower, i.e. that the state characterised by the formula `x < 0.048 and time >` t_0 is reachable. In general, finding the right target formula for these kinds of verification task involves starting from a known baseline behaviour and successively refining the formula and the ranges of the variables involved.

4.2 Flow*

Flow* [CÁS13] is a tool for safety verification of hybrid automata defined by non-linear differential equations with possibly uncertain initial conditions, and non-deterministic resets. It computes an overapproximation of the reachable state space in

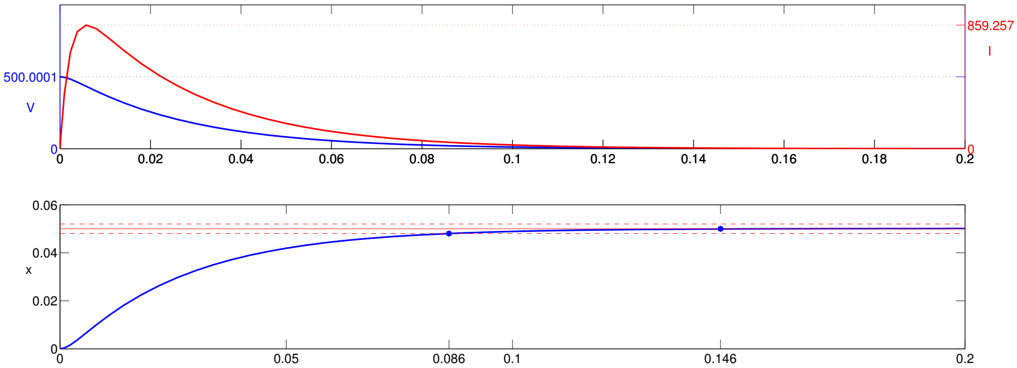


Figure 5: Plot of the behaviour of the simplified Simulink model with the points where the caliper reaches $x_0 - \varepsilon$ and x_0 highlighted

terms of Taylor models that represent the flowpipes up to a bounded time horizon and maximum number of discrete jumps.

As Flow* can deal with hybrid automata given in a textual format, it can operate directly on the automata presented in Section 3. Figure 6 shows the result of applying Flow* to the discrete-time hybrid automaton from where the physical parameter represented by the coefficient $\frac{1}{L} \cdot (R + \frac{K^2}{d_{\text{rot}}})$ varies by ±3 from its nominal value of 504, but stays constant during a run of the tool. This encoding of the uncertainty in the physical parameters is a compromise that takes into account the difficult numerical properties of the differential equations used. The sampling interval was set to $T_{sample} = 10^{-4}$. The simulation was run with time horizon 0.1 seconds and a maximum of 1001 jumps. The computation of the flowpipes took 43842 seconds using a minimum step size of 10^{-10} for Taylor models of order 3. This relatively small steps is due to the stiffness of the ODEs of the plant model. With larger step sizes the flowpipe computation is aborted because Flow* cannot ensure that Taylor models safely enclose the flowpipes. Due to the small step size, a small order of the Taylor models is sufficient. The base model with a clock jitter in the range $[-10^{-8}, 10^{7}]$ took 48100 seconds computing time with a similar result.

As the computed flowpipes represent an approximation of all possible trajectories of the system, the designer can quickly spot problematic deviations from the nominal behaviour due to parameter variations. This is in contrast to the single counter-example trace produced by tools like iSAT-ODE.

4.3 S-TaLiRo

S-TaLiRo [ALFS11] is a tool for falsifying requirements formalised in metric temporal logic (MTL) by finding a system trajectory that violates the requirement. This is a fundamentally different approach than the bounded model checking procedure employed by iSAT-ODE and the flowpipe construction of Flow*. The falsification procedure is based on minimisation of a robustness metric. In essence, this robustness value defines a tube around the trajectory that is invariant with respect to the property under consideration. This provides insight into how robustly a model satisfies a formal specification, which is especially useful for dealing with variable parameters or other uncertainties in the model. Because S-TaLiRo is integrated into

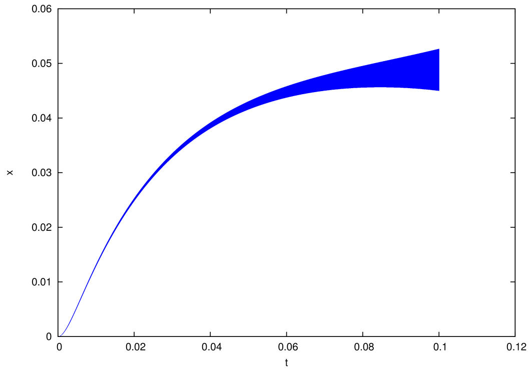
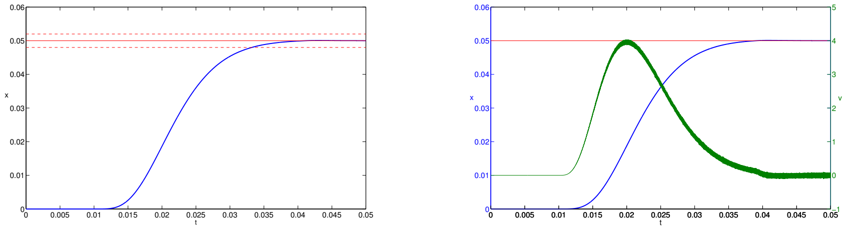


Figure 6: Plot of the result of applying Flow* to the discrete-time model with parameter variations

Matlab/Simulink we were able to use the original simulation model of the brake.

The requirement that the position of the brake disc reaches the set point x_0 with a tolerance ε and stays there can be formalised as the MTL formula $\varphi_1 \equiv \Diamond_{[0,\ t_0]} \Box\ (x \geq x_0 - \varepsilon \wedge x \leq x_0 + \varepsilon)$. For the experiments we fix the parameters of the formula to be $t_0 = 0.033$ seconds (because the brake request is issued at $t = 0.01$) and $\varepsilon = 0.002$. Uncertainty in the measurement of the position x of the brake caliper is modelled as an input x_{noise} that is added to the input of the controller. This disturbance was generated as a piecewise constant signal with 50 sample points over the time of 0.05 seconds which is a compromise between the number of parameters and the quality of the noise signal. The result of falsifying φ_1 for a model with $x_{noise} \in [-0.001,\ 0.001]$ is shown in Figure 7(a). S-TaLiRo finds a simulation run that violates the requirement in 130 seconds using 10 runs, where each run corresponds to a different initial value of the uncertain parameters that is used as a starting point for the optimisation procedure. Manual inspection of the plot reveals that at time $t = 0.033$ the caliper position is $x = 0.04788$.



(a) Plot of falsifying φ_1 for a model with sensor noise

(b) Plot of falsifying $\neg\varphi_2$ in S-TaLiRo without any parameter variations

Figure 7: Plots of falsifying trajectories found by S-TaLiRo

The set point x_0 is depicted by the horizontal solid red line and the tube around it defined by ε is depicted as the two dashed red lines. Letting the resistance R and inductance L of the motor vary with a tolerance of 5% yields a model that also violates the formula φ_1. Using 10 runs, S-TaLiRo finds the valuation $R = 0.52474$ and $L = 0.0010239$ that violates the requirement with robustness -0.0015 in 265 seconds.

The second requirement that the velocity of the caliper should stay below 2 mm/s upon contact with the brake disc is formalised as $\varphi_2 \equiv \Box\,((x \leq x_0 \wedge X(x \geq x_0)) \rightarrow v \leq 0.2)$. The model without any parameter variations or perturbations does not satisfy this requirement robustly. S-TaLiRo reports robustness value of $-7.1878 \cdot 10^{-7}$ for falsifying $\neg\varphi_2$, computed in 2.3 seconds. Figure 7(b) shows the corresponding plot with position x on the left (blue) and velocity v on the right (green) ordinate axis.

5 Conclusion

We examined an industrial case study with the help of three different state-of-the-art verification tools for hybrid systems, two of them based on computing the full reachable state space (Flow*) or a single trace (iSAT-ODE) up to a user-specified bound, and the third (S-TaLiRo) based on simulation. The tools iSAT-ODE and Flow* were used to verify properties on simplified continuous- and discrete-time models of the system under parameter variations. Due to the very stiff dynamics of the case study there are at present limits to what can be achieved with tools that explore the full state space. In a design process where simulation models are readily available, tools such as S-TaLiRo are a viable alternative. Although simulation-based tools do not provide the same kind of guarantees because they only sample a number of trajectories of the system, they can provide useful insights into the system behaviour, which are already very helpful in the design process.

References

[ALFS11] Y. S. R. Annapureddy, C. Liu, G. E. Fainekos, and S. Sankaranarayanan. S-TaLiRo: A Tool for Temporal Logic Falsification for Hybrid Systems. In *Tools and algorithms for the construction and analysis of systems*, volume 6605, pages 254–257. Springer, 2011.

[CÁS13] X. Chen, E. Ábrahám, and S. Sankaranarayanan. Flow*: An Analyzer for Non-Linear Hybrid Systems. In *Proc. of the 25th Int. Conf. on Computer Aided Verification (CAV'13)*, volume 8044 of *LNCS*, pages 258–263. Springer-Verlag, 2013.

[ERNF12] A. Eggers, N. Ramdani, N. S. Nedialkov, and M. Fränzle. Improving the SAT modulo ODE approach to hybrid systems analysis by combining different enclosure methods. In *Software & Systems Modeling*, pages 1–28, 2012.

[FHQW15] G. Frehse, A. Hamann, S. Quinton, and M. Woehrle. Formal Analysis of Timing Effects on Closed-loop Properties of Control Software. In *RTSS 2015*, 2015.

[FLGD+11] G. Frehse, C. Le Guernic, A. Donzé, S. Cotton, R. Ray, O. Lebeltel, R. Ripado, A. Girard, T. Dang, and O. Maler. SpaceEx: Scalable verification of hybrid systems. In *Computer Aided Verification*, pages 379–395. Springer Berlin/Heidelberg, 2011.

[ZYZ+14] H. Zhao, M. Yang, N. Zhan, B. Gu, L. Zou, and Y. Chen. Formal Verification of a Descent Guidance Control Program of a Lunar Lander. In *FM 2014: Formal Methods*, volume 8442 of *Lecture Notes in Computer Science*, pages 733–748. Springer International Publishing, 2014.