# Formal Verification of Nonlinear Simulink Models via Syntactic Hybridization

**Nikolaos Kekatos** [*] **Marcelo Forets** [**] **Goran Frehse** [***]

[*] *Bat. IMAG, 700 Av. Centrale, 38041, France. (e-mail: nikolaos.kekatos@imag.fr)*
[**] *Bat. IMAG, 700 Av. Centrale, 38041, France. (e-mail: marcelo.forets@imag.fr)*
[***] *Bat. IMAG, 700 Av. Centrale, 38041, France. (e-mail: goran.frehse@imag.fr).*

**Abstract:** In this paper we present a methodology that facilitates the integration of formal verification techniques into the model-based design paradigm. The focus is on set-based reachability analysis and on control systems that are described by hybrid dynamics and nonlinear components. Starting with a standard simulation model, e.g., in MATLAB/Simulink, we transform it into an equivalent verification model, formally a network of hybrid automata, in the SX format used by several reachability tools. A major obstacle here is that highly scalable reachability algorithms and tools exist for piecewise affine (PWA) dynamical models, but not for nonlinear ones. To obtain PWA over-approximations of nonlinear dynamics, we use an abstraction method known as hybridization. It partitions the state-space into a set of domains, and for each domain, it approximates the nonlinear dynamics by simpler ones with added nondeterministic inputs to account for the abstraction error. Existing hybridization procedures operate on the composed (flattened) system, so the number of partitions is a function of the abstraction error that is exponential in the number of variables. This quickly leads to intractably large models, even for small systems. To mitigate this problem, we decompose the original dynamics and carry out the state-space partitioning and PWA approximation on the components. The number of partitions in each PWA component is at most quadratic in the abstraction error so that an explosion in the number of partitions is largely avoided. Since the SX format can handle templates, several components may share the same abstraction. The result is a highly compact model that retains the modular structure of the original simulation model. If only a small subset of the partitions is reachable, the bottleneck of having excessively large PWA models can be avoided by composing the model on-the-fly during the reachability analysis. We illustrate the approach by verifying selected nonlinear MATLAB/Simulink models with the reachability tool SpaceEx.

*Keywords:* Reachability, Hybrid Systems, SpaceEx, Compositional Methods, Cruise Control, MATLAB

## 1. INTRODUCTION

In model-based design (MBD), the plant and its controller are designed based on a model, typically within a simulation environment like MATLAB/Simulink. Any kind of nondeterminism in the system, like disturbances, measurement noise, parameter uncertainties, user input, or operating conditions, may have adverse effects on the performance, which can be difficult to predict during the design step. Therefore the system is typically tested by simulating a large number of trajectories, each with a different choice for the nondeterministic quantities, and checking whether they satisfy the requirements. This process is generally incomplete since the number of different choices is prohibitively large or even infinite. Therefore, it can be hard to say with high confidence whether a requirement is truly satisfied under all circumstances. Formal verification attempts to guarantee that requirements are satisfied through a rigorous mathematical analysis of the system. One of the most widely used verification techniques is set-based reachability analysis, which exhaustively simulates families of trajectories using geometric operations on sets.

There are two main obstacles to applying reachability analysis in MBD. First, the simulation model needs to be converted to a suitable formal model, such as a hybrid automaton. Second, the model must be amenable to existing reachability algorithms, in particular in terms of scale. Highly scalable algorithms are known for piecewise-affine (PWA) dynamical systems, but not for more complex nonlinearities. While a large class of nonlinearities can be approximated arbitrarily well by a PWA system, the resulting models can be very large, again running into scalability problems. In this paper, we propose an approach to transform a simulation model into a compact, i.e., relatively small, verification model with PWA dynamics. To achieve this, we decompose the nonlinear system and perform the transformation component-wise. The resulting model can be fed to the verification tool SpaceEx, or translated into formats for other verification tools using the HyST tool [Bak et al. (2015)]. Since SpaceEx composes the model on-the-fly during the analysis, only the reachable partitions of the PWA approximations are instantiated.

⋆ This work was partially supported by the European Commission under grant 643921 (UnCoVerCPS) and by the Metro Grenoble through the project NANO2017.

Much work has been done towards the verification of Simulink models. A promising group of approaches can be categorized as *verification by simulation*. Donzé [Donzé (2010)] presented a MATLAB/Simulink based tool, *Breach*, which performs simulation-based verification (approximate reachability analysis) and conducts efficient signal monitoring of properties and requirements. Breach facilitates the computation and property investigation of large sets of trajectories, but it still cannot provide absolute confidence in the simulation results. Another MATLAB toolbox that is designed to be seamlessly integrated into the model based design process of MATLAB/Simulink is S-Taliro [Annpureddy et al. (2011)]. S-Taliro conducts fast and efficient simulations, but intrinsically relies on gridding, restricting the formal focus on falsification. The MATLAB/Simulink-based tool C2E2 [Fan et al. (2016)] generalizes simulation trajectories to families of trajectories by deriving a neighborhood around the simulated trajectory in which all trajectories have equivalent behavior. All the above verification by simulation approaches have in common that the set of initial states must be sampled. Since the number of required samples can increase exponentially with the number of state variables, this can limit the approach to systems with low-dimensional initial states. The tool HySon [Bouissou et al. (2012)], [Bouissou et al. (2014)] performs set-based simulation directly on a Simulink model, and allows to compute a good approximation of the set of all possible executions. However, the technical details suggest that it may have its drawbacks when analyzing hybrid systems for an unbounded switching horizon. In addition, the HySon is not publicly available and has not seen any more recent publications.

The translation of a Simulink/Stateflow model to a hybrid automaton is supported by the tools HyLink [Manamcheri et al. (2011)] and GreAT [Agrawal et al. (2004)]. However, urgency/must semantics and hierarchical modeling are not allowed. Recently, Minopoli and Frehse presented SL2SX, a semi-automated tool for translation of Simulink models into hybrid automata [Minopoli and Frehse (2016)]. The translator supports a large number of Simulink blocks, but is restricted by SpaceEx limitation to handle piecewise constant and affine dynamics. As a result, the user should analyze the missing blocks (unsupported or nonlinear) and decide how to replace or approximate them.

It is well-known that non-compositional methods for PWA approximations are not computationally efficient for complex systems, since an acceptable accuracy requires a very large number of states for the piecewise-affine approximation. Very recently, [Deshmukh et al. (2015)] presented an experimental comparison of a compositional approach, similar to that presented in this paper (called nested approximation there), against a simplex-partitioning PWA hybridization, showing that the former scales much better than the latter for increasing demands on precision. The compositional PWA-approximation is presented informally, and the paper does not discuss the implications in terms of the model size, nor preserving this compositionality in the generated model. The complexity of the approximation can further be reduced by focusing on a set of reference trajectories, as done in [Bak et al. (2016)].

The main contribution of this work is to introduce and implement a compositional syntactic hybridization method, which is suitable for Simulink models and takes advantage of the on-the-fly composition of hybrid systems that is supported by SpaceEx platform. Moreover, nonlinear models are automatically converted into a sound over-approximation with hybrid automata. The tool runs entirely from MATLAB and will be available at *http://spaceex.imag.fr.*

The rest of this paper is organized as follows. In Section 2 we review the state-of-the-art of existing tools that form the ground of our developments. In Section 3 we develop the strategy adopted in this paper for handling nonlinear subsystems. In Section 4 we describe the steps of our proposed methodology. An example is shown in Section 5, and finally, we draw the conclusions and perspectives in Section 6.

## 2. PRELIMINARIES

In this section, we present the software tools that we utilize in this paper and emphasize on their functionalities and limitations.

### 2.1 Model-based design with Simulink

Simulink is a widely used tool in industry to model and simulate physical systems. Formally, a Simulink model [Alur et al. (2008)] is the tuple $SL = \langle D, B, C \rangle$, where:

- $D = \{D_I, D_O\}$ is the set of input and output *variables* respectively;
- $B$ is the set of *blocks*, each block being defined by an interface (inputs, outputs, and parameters);
- $C \subseteq B \times B$ is the set of *connections* between blocks. This set enables the representation of data flows (which may or may not have physical meaning) between different blocks.

In Simulink, a block $b \in B$ is allowed to be a subsystem. This enables hierarchical modeling, keeping functionally related models together, and simplifying the overall design process by means of abstraction.

### 2.2 Monitoring with Breach

Breach [Donzé (2010)] relies on simulation-based techniques and can be applied to a large class of systems, described as Simulink, ODE or black-box models. The main purpose is to efficiently test a system against a large number of parameter choices and initial conditions, and conduct falsification analysis. For this purpose, Breach computes sensitivity analysis to measure the influence of a parameter change on each trajectory, and it can be used for approximate reachability and parameter synthesis.

### 2.3 Hybrid Automata and SpaceEx

Hybrid automata are a class of mathematical models of dynamical systems admitting both discrete-event (logical) and continuous (numerical) dynamics.

Formally, a SpaceEx model [Frehse et al. (2011)] is the tuple $SX = \langle Comp, Bind \rangle$, where $Comp$ represents the *components* (base or network), and $Bind$ is a relation that associates each network component with a set of components. A *base component* corresponds to a single hybrid automaton, whereas a *network component* corresponds to the parallel composition of several hybrid automata.

SpaceEx models respect the semantics of SX grammar; the format is similar to the standard hybrid automata, syntactically extended with hierarchy and templates.

Simulink relies on *must* semantics (also known as urgent or as-soon-as-possible (ASAP) semantics). That means that discrete events/transitions occur as soon as a given condition (guard) is satisfied. On the other hand, most formal tools for reachability analysis use *may* semantics, demonstarting a broader set of behaviors.

### *2.4 Semi-automatic translation with SL2SX*

SL2SX [Minopoli and Frehse (2016)] is a semi-automated tool that undertakes the translation of Simulink models to SpaceEx models. The translator accepts a Simulink model that is saved in XML format and generates as an output a network of hybrid automata in SX format, a format that is compatible with the SpaceEx platform.

The translation preserves all structural aspects of the Simulink diagram, such as the names, hierarchy and graphical positions. Moreover, having the mechanical aspects of the model transformation being carried out by a tool reduces errors.

## 3. COMPOSITIONAL SYNTACTIC HYBRIDIZATION

The objective of compositional syntactic hybridization is to approximate in a compositional manner the original model with a hybrid automaton with PWA dynamics. Three main steps are involved: *syntactic decomposition*, replacing the original system by an equivalent one with extra variables; *hybridization*, constructing a PWA approximation for each domain and providing a sound over-approximation of the original system by adding an error term; and finally *HA composition*, where the PWA model is transformed into a hybrid automaton in SX format.

In the following paragraphs we explain the technical details involved, considering an abstract nonlinear differential equation of the form

$$\frac{dx}{dt} = f(x), \qquad x \in \mathbb{R}^n. \tag{1}$$

We assume that $f$ is Lipschitz of constant $L > 0$ over the state space $\mathcal{X} \subset \mathbb{R}^n$.

### *3.1 Syntactic Decomposition*

The decomposition consists in constructing a new system where nonlinear terms are replaced by auxiliary variables,

$$\frac{dx}{dt} = g(x,y), \qquad y \in \mathbb{R}^m, \tag{2a}$$

$$y = h(x,y). \tag{2b}$$

Here $y$ is a vector of auxiliary variables, $g(x,y) \in \mathbb{R}^n$ is linear in both $x$ and $y$, and $h(x,y) \in \mathbb{R}^m$ includes all the nonlinear terms, $m$, of the original system, as explained in detail below. Notice that we have replaced the original system by a linear ODE in a higher-dimensional space, $\mathbb{R}^{n+m}$, coupled with a set of nonlinear algebraic constraints. Moreover, this step is exact.

Furthermore, let $V_i \subseteq \{x_1, \dots, x_n\}$ for $i \in \{1, \dots, m\}$ be the variables involved in the $i$-th nonlinearity, and let $p_i = |V_i|$ denote the number of variables in such expression. It should be noted that with a sufficient number of auxiliary variables, we can assume that $h_i(x)$ satisfies $1 \leq p_i \leq 2$ for all $i$.

*Example*. For $n = 4$, consider the polynomial vector field $f = [x_2 - x_3x_4, x_1x_2 - x_4, -x_3x_4, x_2 - x_3]$. Introducing the auxiliary variables $y_1 = x_3x_4$, $y_2 = x_1x_2$, then $\dot{x} = g(x,y) = [x_2 - y_1, y_2 - x_4, -y_1, x_2 - x_3]$ and $y = [y_1, y_2] = h(x,y) = [x_3x_4, x_1x_2]$ are linear and nonlinear respectively, and moreover $h$ satisfies the constraint $p_i \leq 2$ for each row. Consider a PWA approximation based on a rectangular partitioning of the state space, with partition elements of size $\ell$ in each dimension. Then the PWA approximation of $f$ leads to $O(1/\ell^4)$ elements and the PWA approximation of $h$ only to $O(1/\ell^2)$ elements.

### *3.2 Hybridization and error estimation*

We consider a set of non-overlapping *domains*, $R_{ij}$, which cover the operational range of the variables in $V_i$, where $j$ is a label for each individual domain. For each $R_{ij}$, we perform a PWA linearization of $h_i$. Hence, (2a)-(2b) is replaced by

$$\frac{dx}{dt} = g(x,y), \qquad y \in \mathbb{R}^m, \tag{3a}$$

$$y = \hat{h}(x,y), \tag{3b}$$

where $\hat{h}$ is a vector of PWA functions.

Let $op$ denote the operating point in the domain $R_{ij}$. Using Taylor's formula with Lagrange remainder, for each $1 \leq i \leq m$,

$$\hat{h}_i(x,y) = h_{i,op} + \frac{\partial h_i}{\partial x}\Big|_{op}(x - x_{op}) + \frac{\partial h_i}{\partial y}\Big|_{op}(y - y_{op}), \tag{4}$$

and

$$h_i(x,y) - \hat{h}_i(x,y) = \frac{1}{2}(x - x_{op})^{\mathrm{T}} \frac{\partial^2 h_i}{\partial x^2}\Big|_{\xi}(x - x_{op}) \tag{5}$$

$$\frac{1}{2}(y - y_{op})^{\mathrm{T}} \frac{\partial^2 h_i}{\partial y^2}\Big|_{\xi}(y - y_{op}) + (x - x_{op})^{\mathrm{T}} \frac{\partial^2 h_i}{\partial x \partial y}\Big|_{\xi}(y - y_{op}),$$

where $\xi = (\xi_x, \xi_y) \in \mathbb{R}^{n+m}$ is an intermediate point in the interval $\xi_x \in \{x_{op} + a(x - x_{op}), a \in [0,1]\}$, and similarly for $\xi_y$. The right-hand side of Eq. (5) is the Lagrange remainder, whose resulting values over the domain $R_i$ are used to estimate the approximation error [Berz and Hoffstätter (1998)].

The linearization errors $\varepsilon_h$ are computed by evaluation of the Lagrange remainder, and satisfy $y = h(x,y) \in \hat{h}(x,y) \oplus \varepsilon_h$. In this paper, we assume that $R_i$ are products of intervals (boxes). Several interesting alternatives exist, notably simplices [Asarin et al. (2007)]. For a box, it is known that the point which minimizes the absolute value of the Lagrange remainder is its center [Althoff et al. (2008)].

### *3.3 HA Composition*

The remaining step is to compose a hybrid automaton in the SX format, producing a model $SX = \langle Comp, Bind \rangle$, where the base components $Comp_b$ correspond to $h_i$ for all $i \in \{1, \dots, m\}$, and the network components $Comp_n$ are associated with the linear dynamics $g_i$. Moreover, the error is expressed by extra variables with range $\varepsilon_h$, and the error threshold $\mu > 0$ is an upper bound on the maximum value it can take (in some chosen norm $||\cdot||$).

To conclude this section, it should be mentioned that the outlined procedure enables us to approximate the reachable set of the original dynamics with arbitrary precision. Let $\Phi(t,x)$ denote the trajectory starting from $x$ evaluated at time $t$. The reachable set of the system from a set of initial points $X_0 \subseteq \mathcal{X}$ during the interval $[0,t]$ is defined as

$$Reach(T, X_0) = \{y = \Phi(\tau, x) : \tau \in [0,t],\ x \in X_0\}. \tag{6}$$

The approximate system converges to the original system, as the following theorem shows.

*Theorem 1.* (see [Asarin et al. (2003)]) The Hausdorff distance between the reachable set of (2a)-(2b) and the reachable set computed through hybridization, (3a)-(3b), from time 0 to a final time $T > 0$ satisfies

$$d_H\big(Reach_f(T,X_0), Reach_{\hat{f}}(T,X_0)\big) \leq \frac{2\mu}{L}\big(e^{LT} - 1\big). \qquad (7)$$

## 4. METHODOLOGY

In this section we present the steps of our approach and illustrate them through an example.

### *4.1 Modeling with Simulink*

The modeling and the control design is undertaken with MATLAB/Simulink, through the interconnection of blocks, signals, and systems. To clearly illustrate the methodology, the proposed steps are applied on a rotational pendulum model, shown in Fig. 1.

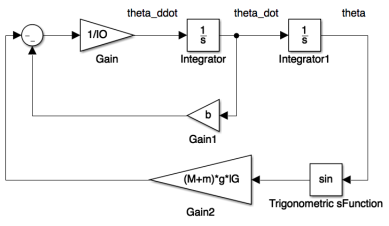


Fig. 1. Simulink model for the rotational pendulum. The system generates the evolution of the pendulum angle over time, when it is released from rest.

### *4.2 Estimation of the signal range*

To construct the abstraction of each Simulink block, it is vital to get the operating ranges of the corresponding signals. There are different ways to estimate them, such as simulations, interval analysis, or Monte Carlo methods. In a real setting, the user may ignore the precise dynamic range of each internal variable, but this would lead to an increase in the anticipated ranges of the variables and a potential increase in the number of hybridization domains. In this paper, we used Breach toolbox to get tight bounds on the behavior (min, max) of the input signals of the nonlinear Simulink blocks.

For the rotational pendulum, it is necessary to estimate the range of the signal that acts as an input in the nonlinear block. A set of simulations for uncertain initial conditions and a Sobol distribution (quasi-random number sequence) are shown in Fig. 2, and we plot the angle θ as a function of time.

### *4.3 SL2SX*

In the context of this work, we use SL2SX to handle the mechanical, but error-prone, aspects of deriving a hybrid automaton interpreted by SpaceEx from a Simulink model. The tool accepts several continuous-time, logical and arithmetical blocks, as well as blocks with discontinuous dynamics (e.g. switches).

As it can be seen in Fig. 3, SL2SX results in a model that can be easily compared with the Simulink diagram, due to the preservation of structure and names of blocks and variables. We have highlighted in red the nonlinear block, which is considered in the next subsection.

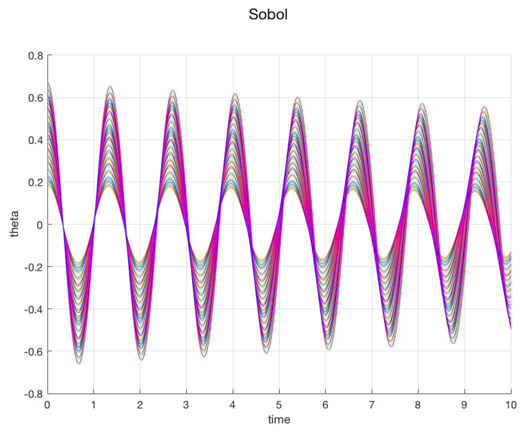


Fig. 2. State monitoring with Breach simulations.

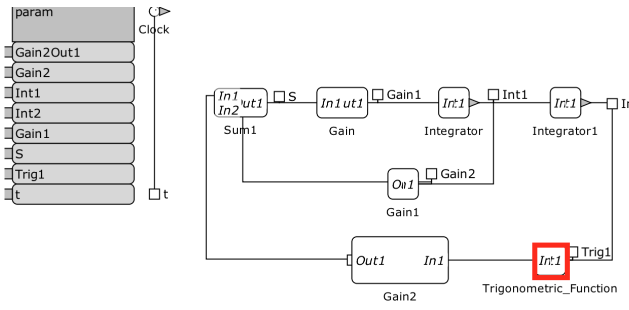


Fig. 3. SpaceEx model (SX) of the rotational pendulum constructed by the translator. This is the network component, and we highlight, in red, the block which corresponds to a trigonometric function.

### *4.4 Compositional Syntactic Hybridization*

Using the theory presented earlier in Section 3, the nonlinear function (sin) is over-approximated by a hybrid automaton. The decomposition step is implemented using the symbolic toolbox of MATLAB. The program identifies the mathematical formula associated with each nonlinear Simulink block, and automatically generates the decomposition $g(x,y)$ and $h(x,y)$.

As already mentioned, for the hybridization we consider the domains to be products of intervals. The threshold error $\mu$ is defined as a nondeterministic input which is represented by an interval, and is calculated for each domain, $\ell$. The final result is shown in Fig. 4.

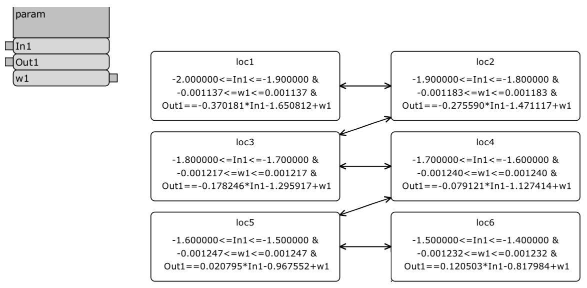

param
In1
Out1
w1
loc1
-2.000000<=In1<=-1.900000 &
-0.001137<=w1<=0.001137 &
Out1==-0.370181*In1-1.650812+w1
loc2
-1.900000<=In1<=-1.800000 &
-0.001183<=w1<=0.001183 &
Out1==-0.275590*In1-1.471117+w1
loc3
-1.800000<=In1<=-1.700000 &
-0.001217<=w1<=0.001217 &
Out1==-0.178246*In1-1.295917+w1
loc4
-1.700000<=In1<=-1.600000 &
-0.001240<=w1<=0.001240 &
Out1==-0.079121*In1-1.127414+w1
loc5
-1.600000<=In1<=-1.500000 &
-0.001247<=w1<=0.001247 &
Out1==0.020795*In1-0.967552+w1
loc6
-1.500000<=In1<=-1.400000 &
-0.001232<=w1<=0.001232 &
Out1==0.120503*In1-0.817984+w1

Fig. 4. PWA of a nonlinear component for the rotational pendulum. Only 6 locations are shown (out of 40). Here the nondeterministic input $w_1$ represents the approximation error.

### 4.5 *Reachability analysis with SpaceEx*

The necessary steps include the integration of the approximations into the original file. In this way, a complete SX model is constructed, combining the exactly translated blocks from SL2SX and the over-appoximated blocks from syntactic hybridization. The SX format is then processed by SpaceEx. An important point is that SpaceEx offers an efficient way to recompose the individual components, without having to partition the entire state space, done through the on-the-fly instantiation of the part of the model that is relevant.

The reachability analysis of the resulting hybrid system is done with one of several SpaceEx Analysis core algorithms (e.g. STC, LGG or PHAVer). In addition, there is the option to check more complicated properties or control specifications. Currently, SpaceEx supports safety verification problems, but there are several directions that enable the transformation of control objectives to reachability problems [Loukkas et al. (2016)].

Computing the reachable sets of the rotational pendulum, released from the most upward position, we get the phase portrait shown in Fig. 5.

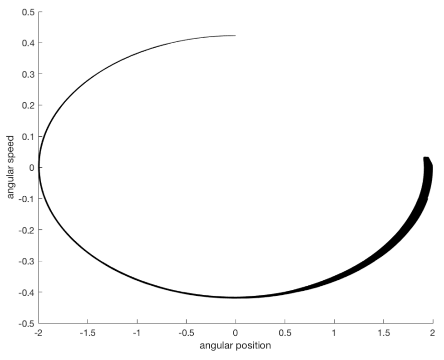


Fig. 5. Reachability results of the rotational pendulum for 1s.

## 5. CASE STUDY: CRUISE CONTROL SYSTEM

The number of driving assistance and safety systems of modern vehicles is constantly increasing. However, there still remain challenges in terms of ensuring safe integration of new modules to existing control functionalities. This challenge is even greater for automated vehicles, as no human intervention may be possible. In this respect, several automotive companies have complemented their MBD paradigm of simulation and testing with formal verification efforts [Althoff (2010)].

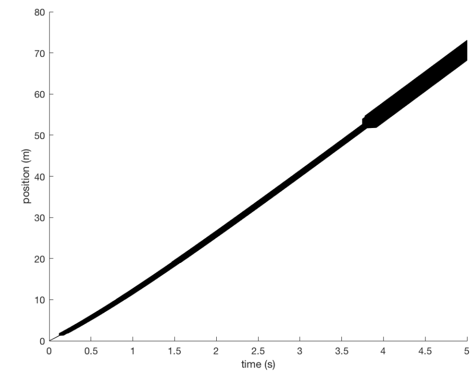
One of the most common driving assistance systems of modern vehicles is cruise control. The purpose of a cruise control system is to regulate the speed of the vehicle, despite external disturbances. Its basic operation is to measure the actual vehicle speed, compare it to the reference or desired speed and automatically accelerate or decelerate according to a control law [Ioannou and Chien (1993), Vahidi and Eskandarian (2003)].

Reachability analysis of a cooperative adaptive cruise controller has been considered in [Kianfar et al. (2012)]. The authors conduct safety analysis of two adjacent vehicles in a platoon, defined by a linear dynamical model and controller. For this case study, we consider a nonlinear physical model and a speed regulation objective [Corona and De Schutter (2006)].

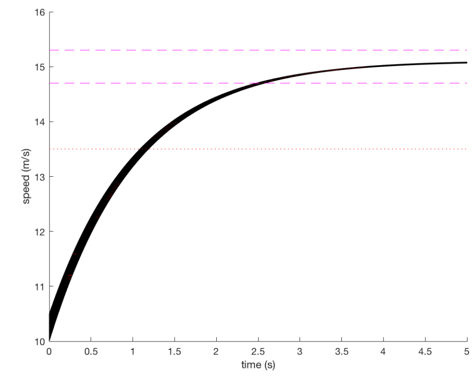
The closed-loop system, shown in Fig. 6, consists of the physical plant (ODEs) and a PID controller. The control objective is to regulate the vehicle speed while respecting the following design specifications:

- Rise-time $< 2$s
- Settling-time $< 5$s
- Steady-state error $< 2\%$
- Overshoot $< 10\%$

Considering a reference speed of 15 m/s and an uncertain initial speed in the interval $[10, 10.25]$ m/s, the reachable set results are shown in Fig. 6. The first figure illustrates the evolution of the vehicle position over time, whereas the second depicts the speed over time.



(a) Reachable sets computed with SpaceEx (Position over time).



(b) Reachable sets computed with SpaceEx (Speed over time). The value corresponding to 90% of the desired speed is described by the red dotted line. The values corresponding to 98% and 102% are in magenta.

Fig. 6. Reachable sets of cruise control system.

It can be readily observed that all the design requirements are met. The controller does not introduce any overshoot, the steady state error is below $2\%$ and the temporal specifications are satisfied.

## 6. CONCLUSION AND PERSPECTIVES

In this work we have proposed a methodology to perform set-based reachability analysis on a class of Simulink models with nonlinear functions. To approximate nonlinearities with piecewise affine functions, we use a compositional syntactic hybridization that relates to the structure of the dynamic equations, and includes the approximation error as an additional variable. The implementation, based on SpaceEx for reachability computations, uses on-the-fly composition of hybrid automata, so that only the reachable parts of the approximation are actually instantiated. We obtain a significant reduction in terms of the model and the analysis time, when comparing with the standard state-space discretization over the fully composed (flattened) model. Moreover, our compositional hybridization can be applied not only to the dynamics, but also to algebraic and initial constraints.

Our future work is targeted towards improvements in all three steps of the syntactic approach. As for the decomposition, we plan to broaden the syntactic method for a larger class of nonlinear functions and we would like to find out what is the best method to break down nonlinearities that includes a large amount of variables. As far as the PWA approximation is concerned, we intend to perform refinement of the discretization domains in order to reduce their total number and stay in the most interesting regions. Finally, we would like to explore the importance of adding non-uniform, multidimensional grid methods.

## ACKNOWLEDGEMENTS

The authors would like to thank Stefano Minopoli and Willem Hagemann for valuable discussions.

## REFERENCES

Agrawal, A., Simon, G., and Karsai, G. (2004). Semantic translation of simulink/stateflow models to hybrid automata using graph transformations. *Electronic Notes in Theoretical Computer Science*, 109, 43–56.

Althoff, M. (2010). Reachability analysis and its application to the safety assessment of autonomous cars. *Technische Universitt Mnchen*.

Althoff, M., Stursberg, O., and Buss, M. (2008). Reachability analysis of nonlinear systems with uncertain parameters using conservative linearization. In *Decision and Control, 2008. CDC 2008. 47th IEEE Conference on*, 4042–4048. IEEE.

Alur, R., Kanade, A., Ramesh, S., and Shashidhar, K. (2008). Symbolic analysis for improving simulation coverage of Simulink/Stateflow models. In *Proceedings of the 8th ACM international conference on Embedded software*, 89–98. ACM.

Annpureddy, Y., Liu, C., Fainekos, G., and Sankaranarayanan, S. (2011). S-taliro: A tool for temporal logic falsification for hybrid systems. In *International Conference on Tools and Algorithms for the Construction and Analysis of Systems*, 254–257. Springer.

Asarin, E., Dang, T., and Girard, A. (2003). Reachability analysis of nonlinear systems using conservative approximation. In *International Workshop on Hybrid Systems: Computation and Control*, 20–35. Springer.

Asarin, E., Dang, T., and Girard, A. (2007). Hybridization methods for the analysis of nonlinear systems. *Acta Informatica*, 43(7), 451–476.

Bak, S., Bogomolov, S., Henzinger, T.A., Johnson, T.T., and Prakash, P. (2016). Scalable static hybridization methods for analysis of nonlinear systems. In *HSCC'16*, 155–164.

Bak, S., Bogomolov, S., and Johnson, T.T. (2015). HYST: a source transformation and translation tool for hybrid automaton models. In *[HSCC'15]*, 128–133.

Berz, M. and Hoffstätter, G. (1998). Computation and application of taylor polynomials with interval remainder bounds. *Reliable Computing*, 4(1), 83–97.

Bouissou, O., Mimram, S., and Chapoutot, A. (2012). Hyson: Set-based simulation of hybrid systems. In *2012 23rd IEEE International Symposium on Rapid System Prototyping (RSP)*, 79–85. IEEE.

Bouissou, O., Mimram, S., Strazzulla, B., and Chapoutot, A. (2014). Set-based simulation for design and verification of Simulink models. In *Embedded Real Time Software and Systems (ERTS2)*.

Corona, D. and De Schutter, B. (2006). Adaptive cruise controller design: A comparative assessment for pwa systems. *IFAC Proceedings Volumes*, 39(5), 253–258.

Deshmukh, J.V., Ito, H., Jin, X., Kapinski, J., Butts, K., Gerhard, J., Samadi, B., Walker, K., and Xie, Y. (2015). Piecewise-affine approximations for a powertrain control verification benchmark. In *Workshop on Applied Verification for Continuous and Hybrid Systems*.

Donzé, A. (2010). Breach, a toolbox for verification and parameter synthesis of hybrid systems. In *International Conference on Computer Aided Verification*, 167–170. Springer.

Fan, C., Qi, B., Mitra, S., Viswanathan, M., and Duggirala, P.S. (2016). Automatic reachability analysis for nonlinear hybrid models with C2E2. In *CAV'16*, 531–538.

Frehse, G., Le Guernic, C., Donzé, A., Cotton, S., Ray, R., Lebeltel, O., Ripado, R., Girard, A., Dang, T., and Maler, O. (2011). SpaceEx: Scalable verification of hybrid systems. In *International Conference on Computer Aided Verification*, 379–395. Springer.

Ioannou, P.A. and Chien, C.C. (1993). Autonomous intelligent cruise control. *IEEE Transactions on Vehicular technology*, 42(4), 657–672.

Kianfar, R., Falcone, P., and Fredriksson, J. (2012). Reachability analysis of cooperative adaptive cruise controller. In *2012 15th International IEEE Conference on Intelligent Transportation Systems*, 1537–1542. IEEE.

Loukkas, N., Meslem, N., and Martinez-Molina, J.J. (2016). An interval technique to check the performance of control laws applied to wind turbines. *SWIM 2016*.

Manamcheri, K., Mitra, S., Bak, S., and Caccamo, M. (2011). A step towards verification and synthesis from simulink/stateflow models. In *Proceedings of the 14th international conference on Hybrid systems: computation and control*, 317–318. ACM.

Minopoli, S. and Frehse, G. (2016). SL2SX translator: From Simulink to SpaceEx models. In *Proceedings of the 19th International Conference on Hybrid Systems: Computation and Control*, 93–98. ACM.

Vahidi, A. and Eskandarian, A. (2003). Research advances in intelligent collision avoidance and adaptive cruise control. *IEEE transactions on intelligent transportation systems*, 4(3), 143–153.