Using HyTech to Verify an Automotive Control System

Thomas Stauner and Olaf Müller*
Department of Computer Science
Munich University of Technology
80290 Munich
{stauner,mueller}@informatik.tu-muenchen.de

Max Fuchs
BMW AG
FIZ EG-K-3
80788 Munich
maximilian.fuchs@bmw.de

Abstract. This paper shows how HyTech, a symbolic model checker for linear hybrid systems, can be used to verify a part of an abstracted automotive control system. The system controls the height of an automobile by a pneumatic suspension system and has been proposed by BMW AG as a case study taken from a current industrial development. For a system which controls one wheel we verify safety properties, such as that the height of the car maintains within desired bounds or that the height is not changed in curves, by reachability analysis. Furthermore, a property related to stability in the sense of control theory is verified. We believe that the case study can serve as a real-life benchmark problem for the formal analysis of embedded reactive systems.

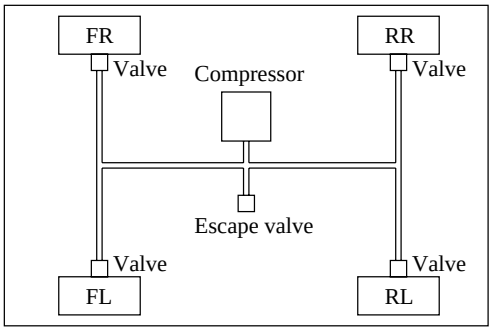
1 Introduction

Our research partner BMW AG proposed a case study taken from a current industrial development describing a system that controls the height of a chassis by pneumatic suspension. The goal of the paper is to evaluate a representative of the existing tools for the analysis of embedded reactive systems w.r.t. to the degree of automation, scalability and practicability by applying it to this case study. We choose the model of hybrid automata [ACH+95] because a state based model is more familiar to a design engineer than a logical approach. Among the existing hybrid model checkers we select HyTech [HHWT95], as it provides the most general input language, *linear hybrid automata*. Nevertheless, the case study incorporates continuous activities which cannot be described directly using linear hybrid automata. Therefore we have to employ approximation techniques [HWT96a, HH95].

The Electronic Height Control System. The main aims of the electronic height control (EHC) are to increase driving comfort, to keep the headlight load-independent and to adjust the chassis level to off-road and on-road conditions by providing a high and a low chassis level, selectable by the driver.

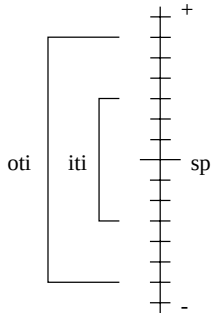
This is achieved by a pneumatic suspension at each of the four wheels. The chassis level can be increased by pumping air into the suspension of the wheels

* Research supported by "KorSys", BMBF.



FR: front right wheel
RR: rear right wheel
FL: front left wheel
RL: rear left wheel

Fig. 1.



sp: set point
iti: inner tolerance interval
oti: outer tolerance interval

Fig. 2.

and decreased by blowing air off. As there is only one compressor and one escape valve for all four wheels, to e. g. increase the level at the front right wheel, the compressor has to be turned on, the escape valve must be closed and the valve connected to the suspension of the front right wheel must be opened (Fig. 1).

Sensors measure the deviation of the actual chassis level from a defined zero level at each of the wheels. Disturbances of high frequency, caused by road holes or block pavement for example, are filtered out by low-pass filters before the values are passed on to the controller. The controller reads them with a given sampling rate and decides how valves and compressor must be operated during the next sampling interval. Escape valve or compressor respectively are turned on if the filtered chassis height is outside an outer tolerance interval for one of the wheels (Fig. 2). For example, if the height exceeds the outer tolerance interval for one wheel, the valve connected to this wheel's suspension and the escape valve are opened. The valves are closed again if the measured chassis levels at all four wheels are below the upper limit of an inner tolerance interval, in addition the filters are reset to the set point, i. e. the desired chassis level. To increase the chassis level the compressor and the valves connected to the suspensions are operated in a similar way.

As compressor and escape valve must not be used simultaneously, priority is given to the compressor in those cases where both would have to be used.

The actual values of the tolerance intervals are determined by the mode the car is in. The modes we model in this case study are "car is stopped, engine off" and "car is driving, engine on". In section 4.3 we will regard a further mode which provides that control is suspended when the car is driving through a bend. The original study of BMW contains some more modes which mainly differ in the actual values for the inner and outer tolerance intervals.

Results. We analyze a restricted model with only one wheel and one chassis level. This seems to be a natural abstraction and allows for the verification of several interesting properties without exceeding the efficiency limits of HyTech.

There are two factors which blow up the complexity of the case study. Firstly approximation techniques have to be applied to the filters with nonlinear[1] behavior. We argue that the complexity increase caused by such approximations are a major disadvantage w.r.t. to the practicability of the existing model checkers that are all restricted to linear hybrid systems. Secondly the environment of the EHC, namely the road conditions, are difficult to model, as they are of an inherently complex and unpredictable nature. Therefore we restrict ourselves to a road model that allows only such disturbances the compressor is able to compensate immediately.

Within this model we prove that our system keeps the chassis level within certain bounds and identify these bounds. We verify that compressor and escape valve are never used at the same time. Furthermore we analyze the property that the EHC does not try to change the chassis level when the car is driving through a bend. Inspired by the notion of stability in control theory we also examine the system response to a step-like disturbance in order to guarantee that the EHC cannot continue to use compressor and escape valve endlessly. For a particular disturbance we prove that usage of valves and compressor terminates within certain time bounds and identify these bounds.

A major problem of the existing hybrid model checkers seems to be their insufficient scalability. Our real-world example reached the efficiency limits of HyTech, where arithmetic overflows seem to be more serious in practice than nontermination. On the other hand, we appreciated the automation of the analysis and the ease of use of HyTech.

Related Work. Several case studies have been performed using HyTech. But only the steam boiler [HWT96b] and the audio control protocol [HWT95] case studies aimed at applying automated analysis techniques to real-life applications. Our case study seems to be distinctly more complex than previous ones because of the nonlinear filter and the complex environment. As far as we know, this paper is the first to describe a case study with HyTech not performed by the developers of the tool themselves. Other hybrid model checkers, such as KRONOS or UPPAAL, have been developed and applied to several examples, as well as non-algorithmic approaches. For a survey, see [AHS96].

2 Hybrid Automata

2.1 Hybrid Automata

A hybrid automaton A is a 10-tuple $(X, V, inv, init, flow, E, upd, jump, L, sync)$ with the following meaning (for a detailed introduction see [ACH+95]):

X is a finite ordered set $\{x_1, ..., x_n\}$ of variables over $\mathbb{R}$. A valuation s is a point in $\mathbb{R}^n$, the value of variable x_i in valuation s is the i-th component of s, s_i.

[1] *Nonlinear* in the literature on hybrid automata means that the system cannot be described by a linear hybrid automaton without approximation. This is different from the notion of (non)linear differential equations.

When Φ is a predicate over the variables in X, we write $[\![\Phi]\!]$ to denote the set of valuations s for which $\Phi[X := s]$ is true.

V is a finite set of locations. A state of automaton A is a tuple (v, s) consisting of a location $v \in V$ and a valuation s. A set of states is called a region.

inv is a mapping from the set of locations V to predicates over the variables in X. $inv(v)$ is the invariant of location v. When control is in v, only valuations $s \in [\![inv(v)]\!]$ are allowed. In our graphical representation of hybrid automata invariants *True* are omitted.

init is a mapping from the locations in V to predicates over X. $init(v)$ is called the initial condition of v. (v, s) is an initial state if $init(v)$ and $inv(v)$ are true for s. Initial conditions are expressed as incoming arrows marked with the condition in our automata diagrams. If the initial condition is *False*, the arrow is omitted.

flow is a mapping from the locations in V to predicates over $X \cup \dot{X}$, where $\dot{X} = \{\dot{x}_1, ..., \dot{x}_n\}$ and $\dot{x}_i$ denotes the first time derivative of x_i, dx_i/dt. When control is in location v the variables evolve according to differentiable functions which satisfy the flow condition $flow(v)$.

Formally the δ time-step relation $\xrightarrow{\delta}$ is defined as $(v, s) \xrightarrow{\delta} (v, s')$ iff there is a differentiable function $\rho : [0, \delta] \to \mathbb{R}^n$ with

- $\rho(0) = s$ and $\rho(\delta) = s'$
- the invariant of v is satisfied: $\rho(t) \in [\![inv(v)]\!]$ for $t \in [0, \delta]$
- the flow condition is satisfied: $flow(v)[X, \dot{X} := \rho(t), \dot{\rho}(t)]$ is true for $t \in [0, \delta]$, where $\dot{\rho}(t) = (\dot{\rho}_1(t), ..., \dot{\rho}_n(t))$.

The filter automaton of Fig. 4 for example has flow condition $\dot{f} = \frac{1}{T}(h - f)$.

$E \subseteq V \times V$ is a finite multiset of edges, called transitions. A transition $(v, v') \in E$ has source location v and target location v'. In our automata diagrams transitions are represented by arrows between the corresponding locations.

upd is a mapping from E to the power set of X. $upd(e)$ is called the update set of e. Variables in $upd(e)$ can change their value when transition e is taken.

jump is a mapping from E to jump conditions. The jump condition $jump(e)$ of transition e is a predicate over $X \cup Y'$, where $Y = upd(e) = \{y_1, ..., y_k\}$ and $Y' = \{y_1', ..., y_k'\}$. y_i' stands for the value of variable y_i after the transition is taken, while y_i refers to the value before the transition.

The transition-step relation $\xrightarrow{e}$ is defined as $(v, s) \xrightarrow{e} (v', s')$ iff

- $e = (v, v')$
- the invariants are satisfied: $s \in [\![inv(v)]\!]$ and $s' \in [\![inv(v')]\!]$
- the variables in $upd(e)$ change according to the jump condition: $jump(e)[X, Y' = s, s'[Y]]$ is true, where $s'[Y]$ is the restriction of s' to the variables in Y.
- the variables not in $upd(e)$ remain constant: $s_i = s_i'$ for $x_i \in X \setminus Y$.

Transition steps are discrete, no time passes while they are taken.

In our automata diagrams we write guarded assignments $\Phi \to y_i := c$ for update set $\{y_i\}$ and jump condition $\Phi \wedge y_i' = c$. Guards *True* are omitted.

L is a finite set of synchronization labels. Parallel automata which have a common synchronization label must take transitions with this label simultane-

ously.

sync is a mapping from E to $L \cup \{\tau_A\}$. $sync(e)$ is the synchronization label of transition e. The label τ_A may only be used in automaton A and marks transitions of A which are not synchronized with transitions in parallel automata. In automata diagrams each transition e is labeled with $sync(e)$ and $jump(e)$, the label τ_A is omitted.

All of the above mappings are total.

Parallel Composition. Parallel composition of hybrid automata can be conveniently used for specifying larger systems. A hybrid automaton is given for each part of the system, communication between the components may occur via shared variables and synchronization labels. The parallel composition of hybrid automata is obtained by a product construction.

2.2 Hybrid Automata and the Tool HyTech

HyTech can automatically analyze a subclass of hybrid automata, namely linear hybrid automata. A hybrid automaton A is linear if all its invariants and initial conditions are convex linear predicates over X, all flow conditions are convex linear predicates over $\dot{X}$ and all jump conditions are convex linear predicates over $X \cup X'$. A predicate over variables in a set Y is linear if it is an (in)equality between linear terms over Y. A linear term over Y is a linear combination over Y with rational coefficients. A predicate is a convex linear predicate if it is a finite conjunction of linear predicates.

HyTech can compute the set of states of a parallel composition of linear hybrid automata which is reachable from a set of initial states by repeatedly applying time- and transition-steps. It can also perform backward reachability analysis in which the region is computed from which a given final region can be reached by iterating time- and transition-steps. Monitor automata may be used to prove complex properties of hybrid systems. Sequences of time- and transition-steps leading from one region to another can be generated, which is very useful for finding out why given predicates are violated. Furthermore HyTech allows the analysis of models containing parameters and can thereby be used to synthesize constraints on these parameters which are necessary for correctness of the system [HHWT95].

Approximation techniques can be used to analyze nonlinear hybrid systems [HH95]. Such techniques ensure that properties which are verified for the approximating system also hold for the real system. In section 3.2 we will apply an approximation technique to the filter of the EHC.

3 System Description

3.1 The Environment

The measured chassis level is influenced by disturbances from the outside world, e. g. the road, and by escape valve and compressor.

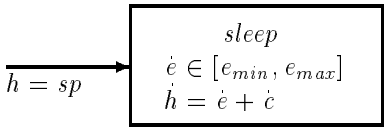


Fig. 3. Hybrid automaton for environment.

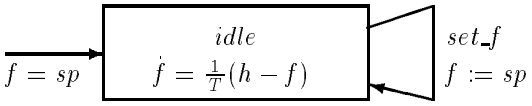


Fig. 4. Hybrid automaton for the filter.

As we cannot make any predictions on when a disturbance will occur and how strong it will be, we say that disturbances of limited strength may occur anytime. There is no sense in allowing disturbances of unlimited strength, since the real system, i. e. the stability of the car, is limited.

The exact effect escape valve and compressor have on the chassis level depends on many factors which we do not want to model explicitly. We only state that usage of the compressor increases the chassis level while usage of the escape valve decreases it. In our system the compressor can lift the chassis with a rate between $cp_{min} = 1\frac{mm}{s}$ and $cp_{max} = 2\frac{mm}{s}$, the escape valve can lower it with a rate between $ev_{min} = -2\frac{mm}{s}$ and $ev_{max} = -1\frac{mm}{s}$.

The rate of change of the chassis level at one wheel $\dot{h}$ now is the sum of the changes due to disturbances, denoted by $\dot{e}$, and the changes due to compressor and escape valve, denoted by $\dot{c}$ (Fig. 3). sp denotes the set point, i. e. the desired chassis level. In order to be able to prove upper and lower bounds for the chassis level in the presence of disturbances it is necessary to demand that disturbances cannot be stronger than compressor and escape valve respectively, i. e. $e_{min} \geq ev_{max}$ and $e_{max} \leq cp_{min}$. If the disturbances were stronger, the controller could not ensure bounds for the chassis level even if it operated correctly. In our system we set $e_{min} = ev_{max}$ and $e_{max} = cp_{min}$.

The influence of the environment is of course seriously limited this way. In reality disturbances stronger than compressor and escape valve are certainly possible. To get a more realistic environment it would therefore be necessary to model it in a way which allows strong disturbances, but provides that their influence on the system is not stronger than that of the controller over a longer period of time. We believe that the limits of the expressiveness of hybrid automata are reached with statements of this kind.

3.2 A Filter as Linear Hybrid Automaton

A filter is described by the hybrid automaton of Fig. 4. h is the chassis level as it comes from the sensor of a wheel, f is the filtered level. T is the time constant of the filter, i. e. the time it takes for f to reach $0.63\,h$ starting from $f = 0$, where

Location	Invariant	Flow condition
A	$h - f \in (-\infty, -10]$	$\dot{f} \in (-\infty, -5]$
B	$h - f \in [-10, -6]$	$\dot{f} \in [-5, -3]$
C	$h - f \in [-6, 0]$	$\dot{f} \in [-3, 0]$
D	$h - f \in [0, 6]$	$\dot{f} \in [0, 3]$
E	$h - f \in [6, 20]$	$\dot{f} \in [6, 10]$
F	$h - f \in [20, \infty)$	$\dot{f} \in [10, \infty)$

Table 1. Partitioning of location *idle*.

$h \neq f$ is constant [PH88]. We use a time constant of $2s$. The synchronization label *set_f* is used by the control logic to reset the filter to the set point *sp*.

Linear Phase-Portrait Approximation for a Filter. Unfortunately the filter of Fig. 4 is not a linear hybrid automaton, because its flow condition is not a convex linear predicate over $\{\dot{f}, \dot{h}\}$. To get a linear hybrid automaton we apply linear phase-portrait approximation, defined in [HWT96a], to the filter.
To do so we have to choose a partitioning of $[\![inv(idle)]\!] = \mathbb{R}^2$. The approximating automaton has one location for each partition then. The flow condition of such a new location is obtained by computing the upper and lower limits of the original flow condition in the respective partition. Every new location has a *set_f* transition to every other new location which corresponds to *idle* and to itself. Finally transitions have to be added which allow control to pass freely between the locations of the approximating filter which correspond to *idle*. The partitioning we use for location *idle* is shown in Table 1. $h - f$ has unit millimeters, $\dot{f}$ has unit millimeters per second.

In Table 1 locations C and D with $f = sp$ are initial states. There is a *set_f* transition with jump condition $f' = sp$ from every location in $\{A, ..., F\}$ to every location in $\{A, ..., F\}$. Furthermore there are τ_{filter} transitions with jump condition *True* from every location of the approximating filter to every other location of it to allow control to pass freely between them.

The choice of this partitioning is guided by the desire to keep the state space small and to make locations A and F unreachable which is achieved by locations B and E. The invariants of C and D are chosen in a way to avoid arithmetic overflows during HyTech's analysis of the EHC. The flow conditions of the locations are determined by the invariants and the original filter's flow condition: e.g. for $h - f \in [0, 6]$, $\dot{f} = \frac{1}{T}(h - f)$ is in $[0, 3]$.

A and F should be unreachable for two reasons. First it is not desirable that the distance between h and f is unbounded. Secondly in composition with the other parts of our model of the EHC the flow conditions of A and F lead to flow conditions which are not permitted by HyTech for efficiency reasons. We therefore use $+/-100$ as upper and lower bounds for $\dot{f}$ in A and F instead of $+/-\infty$ in Table 1. Unless the contrary is mentioned, reachability analysis proves that A and F are unreachable from the initial states in all models presented in this paper. Therefore these bounds do not affect system behavior.

Linear Phase-Portrait Approximation and Parallel Composition. Although phase-portrait approximation is not defined for parallel compositions of automata in [HWT96a], our approximation of the filter is correct. To see this we have to examine what can go wrong when the approximation technique is only applied to one automaton of a parallel composition, the filter in our case.

An automaton running in parallel to the approximation of the filter might want to take a transition which would falsify the invariant of the current location of the approximating filter automaton. It is therefore not allowed to take this transition. However in the unpartitioned filter the same transition would be possible, because $inv(idle) = True$ can never be violated. So the partitioned system would not be a correct approximation, because the reachable region of the original system would contain states which would not have corresponding states in the reachable region of the approximating system. This problem can be solved by forcing all transitions which change the value of variables appearing in the invariants of filter locations, i. e. all transitions which change f or h, to occur simultaneously with new transitions in the filter which allow control to pass between the locations $A, ..., F$. To make the transitions occur simultaneously a synchronization label must be used. We will encounter this situation in section 4.4 and apply the presented method.

Time-steps cannot affect the correctness of the approximating system. Due to the definition of time-steps which demands all variables to follow differentiable trajectories, $h - f$ must also follow a differentiable trajectory. When $h - f$ moves from one interval of the partitioning of *idle* into another in a time-step, it can therefore only do so by passing a value which is in both intervals. As soon as this value is reached, the approximating filter automaton can take a τ_{filter} transition which lets control pass freely between the locations of the corresponding intervals. Time can progress in the newly reached location then.

Linear phase-portrait approximation in the context of parallel composition is examined in detail in [Sta97].

3.3 The Controller

To focus on the main points, we restrict ourselves to model only one wheel. The principal architecture of the system is the same with four wheels. The only real simplification is that situations in which escape valve and compressor would have to be used at the same time, because the chassis level is too high at one wheel and too low at another, can no longer occur. If more than one wheel was regarded, in theory a situation is possible in which h diverges to $+\infty$ at one wheel due to the compressor being permanently busy with stabilizing the chassis level at another wheel. Our model of the environment would have to be modified to exclude this unrealistic case of a permanent disturbance in one direction.

Furthermore we only model the two basic modes *stopped* (with engine off) and *driving* (with engine on). In section 5.3 we add a further mode in which control is suspended when the car is in a bend. In addition, we abstract from the possibility to select different chassis levels and use only one level.

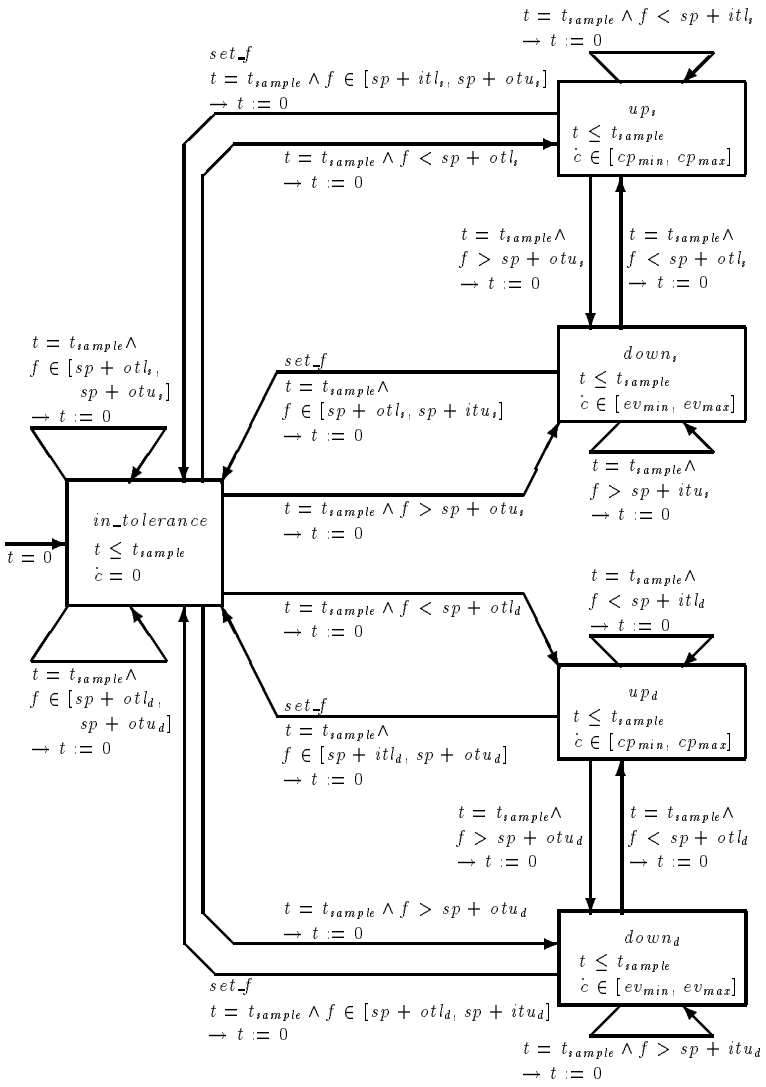


Fig. 5. Hybrid automaton for controller. $\dot{t} = 1$ in all locations.

In Fig. 5 we give the hybrid automaton for the simplified controller. It consists of two symmetric parts corresponding to the two modes, *stopped* and *driving*, denoted by indices s and d. In locations up_s and up_d the compressor is on and the valve connected to the suspension of the wheel is open, the chassis level is therefore increased by $\dot{c} \in [cp_{min}, cp_{max}]$. In locations $down_s$ and $down_d$ the escape valve and the valve to the wheel's suspension are open, the chassis level is decreased by $\dot{c} \in [ev_{min}, ev_{max}]$. Locations up_s and $down_s$ only differ from up_d and $down_d$ in the constants which are used in their incoming and outgoing transitions. In *in_tolerance* the compressor is off and both valves are closed,

the controller does not influence the chassis level, $\dot{c} = 0$. t is the controller's clock. As its flow condition is $\dot{t} = 1$ in all locations, we omitted it in Fig 5. t_{sample} determines the controller's sampling rate $\frac{1}{t_{sample}}$. $[sp + otl, sp + otu]$ is the outer tolerance interval, $[sp + itl, sp + itu]$ is the inner tolerance interval. These constants have the following values provided by BMW AG: $sp = 0mm$, $otl_s = -40mm$, $otu_s = 20mm$, $otl_d = -10mm$, $otu_d = 10mm$, $itl_s = -6mm$, $itu_s = 16mm$, $itl_d = -6mm$, $itu_d = 6mm$.

Starting from the initial state, the controller has to wait in *in_tolerance* until t_{sample} expired. Then it reads the filtered chassis height f and goes to location *up* or *down* if f is outside the outer tolerance interval, otherwise it reenters *in_tolerance*. Whenever *in_tolerance* is left the controller makes a non-deterministic choice between one transition with constants from mode *stopped* and one with constants from mode *driving*. This non-determinism models that the mode may change anytime. In the real system the controller reads a sensor to get the present mode.

The locations up_s and up_d are left after t_{sample} time units if f is no longer smaller than the lower limit of the inner tolerance interval $sp + itl$, otherwise they are reentered. If f is greater than the outer upper tolerance limit $sp + otu$ now, $down_s$ or $down_d$, respectively, has to be entered. Otherwise the filter is reset to the set point by taking the *set_f* transition from up_s or up_d to *in_tolerance*. Locations $down_s$ and $down_d$ work in a similar way. In all the *up* and *down* locations a mode change is ignored until *in_tolerance* is entered.

4 Verification

In the following we will show that our system keeps the chassis level within certain bounds and identify these bounds (Section 4.1). Apart from that we prove that compressor and escape valve are never used at the same time (Section 4.2). Furthermore it is safety critical that the EHC does not try to change the chassis level when the car is driving through a bend. To treat this property we have to model bends and modify the controller appropriately (Section 4.3). Finally we are interested in stability of the EHC. Stability in a very general sense means that the system responds in some controlled manner to applied inputs [PH88]. In our case we want to know whether the controller reaches *in_tolerance* again some time after a disturbance or whether it can continue to use compressor and escape valve infinitely by e.g. operating them alternately . To examine this we regard the system response to a step-like disturbance in Section 4.4.

4.1 Bounds for the Chassis Level

Using the model of section 3, HyTech computes that the chassis level h always is in $[-47mm, 27mm]$ for a sampling rate of once every second, $t_{sample} = 1s$. The computation takes 62 minutes[2] of CPU time. This means that the outer

[2] All performance figures were obtained on a Sun Sparcstation 20 with 128 MB RAM and 350 MB swap space.

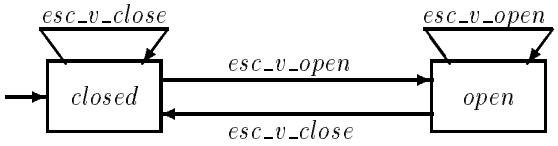


Fig. 6. Hybrid automaton for escape valve.

tolerance limits of mode *stopped*, $-40mm$ and $20mm$, are never exceeded by more that $7mm$. We believe that this result could be significantly improved by using a finer partitioning for the filter and a smaller t_{sample}.

Unfortunately the values of the constants in a model can lead to arithmetic overflows during the analysis with HyTech. When repeatedly computing successor regions in the model checking procedure increasing values can occur in the equations which determine the range of the variables of a system. These values can cause arithmetic overflow in the libraries HyTech uses for computing successors and can thereby abort analysis of a system. In our case the presented partitioning of the filter and the specific choice of the sampling rate turned out to be crucial for the avoidance of overflows. Therefore we are using this slow sampling rate in all our models although reading the sensor values more frequently would be more realistic.

We also tried to use HyTech to compute the slowest sampling rate which guarantees that the chassis level always stays within given bounds, but we did not succeed due to overflows. As soon as the filter is omitted in our model, this question can easily be solved using HyTech's feature to perform parametric analysis, but it is rather trivial then.

4.2 Escape Valve and Compressor

To prove that escape valve and compressor are never used at the same time it is first of all necessary to include them in our model of the EHC. Fig. 6 shows the hybrid automaton for the escape valve. It is opened by a *esc_v_open* transition and closed by a *esc_v_close* transition. The automata for the compressor and the valve connected to the suspension of the wheel are similar. The compressor is operated via the synchronization labels *comp_on* and *comp_off*, the valve to the suspension via *v_open* and *v_close*.

Whenever control changes from one of the locations of the controller of Fig. 5 to another one, the controller must open or close the valves and turn on or off the compressor correspondingly. For example, when control passes from up_d to *in_tolerance*, the compressor must be switched off (label *comp_off*) and the valve connected to the wheel's suspension must be closed (label *v_close*). As a transition cannot have two synchronization labels, we insert transient locations, i. e. locations in which no time passes, in the controller. Thereby one transition can be split into two, with a transient location in between them.

HyTech can analyze the resulting model in 58 minutes. It proves that all states with escape valve open and compressor on at the same time are unreachable from the initial region. In roughly the same CPU time it is possible to

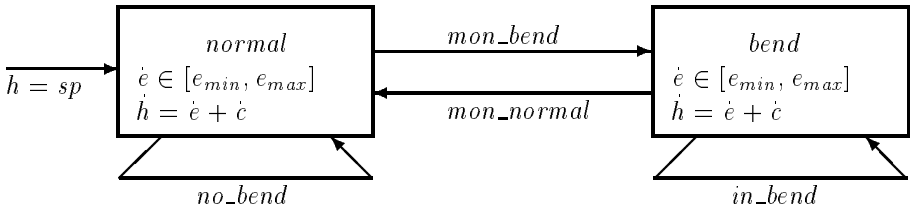


Fig. 7. Hybrid automaton for environment with curves.

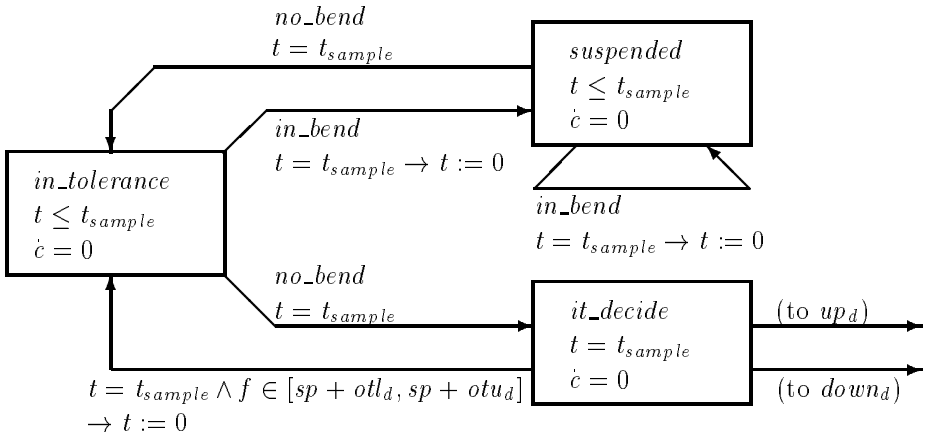


Fig. 8. Priority of the $*$_*bend* transitions. $\dot{t} = 1$ in all locations.

treat the property of the previous section together with verifying correct usage of escape valve and compressor in one analysis of the presented model.

4.3 Suspending Control in Curves

In this section we extend the environment by modeling curves and modify the controller of Fig. 5 to prevent it from changing the chassis level when the car is in a bend. Valves and compressor are omitted again.

The new model for the environment is presented in Fig. 7. The transitions with synchronization labels *in_bend* and *no_bend* allow the controller to determine whether the car is in a bend. The transitions between locations *normal* and *bend* have guard *True* and model that a bend can occur anytime. The synchronization labels *mon_bend* and *mon_normal* are matched by a monitor automaton which we use to determine whether the system works correctly. The locations *normal* and *bend* are derived from location *sleep* of Fig. 3.

As far as the controller is concerned, we need a new location, *suspended*, in which the controller does not change the chassis level and which is entered whenever the car is in a curve and left when the curve is over. It is important that *suspended* is entered even if the chassis level is outside the outer tolerance interval, since the controller may not use compressor or escape valve in a curve.

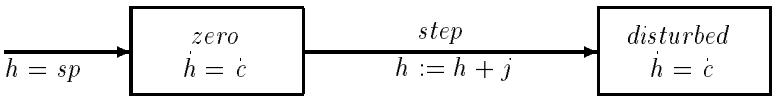


Fig. 9. Hybrid automaton for disturbance of step shape.

We must therefore find a way of modeling priority of transitions. Fig. 8 shows how this is achieved for *in_tolerance*. A new transient location, *it_decide*, is inserted. It can only be entered by a *no_bend* transition. The outgoing transitions of *in_tolerance* in the automaton of Fig. 5 now emerge from *it_decide*. The decision whether the chassis level has to be changed during the next time period is thereby postponed after ensuring that the car is not in a curve. Locations up_d and $down_d$ also get an *in_bend* transition to *suspended* and locations up_d*_decide* and $down_d$*_decide* are added in the same way as *it_decide* in Fig. 8. Mode *stopped* is omitted in this model of the controller since bends cannot occur when the car is not driving. Transitions can only be taken for $t = t_{sample}$, i. e. according to the sampling rate. The clock t is not reset to 0 when *suspended* is left, *in_tolreance* therefore becomes transient in this case and the controller is forced to make a control decision based on the chassis height right after *suspended* is left.

Our model of the system is now augmented by a monitor automaton which measures the time between a bend being entered (label *mon_bend*) and control being suspended (label *in_bend*) or the bend being left again (label *mon_normal*), whatever occurs first. As we also want to know whether the controller continues its normal operation after a bend, we furthermore measure the time between a *mon_normal* transition and the corresponding *no_bend* or *mon_bend* transition. If the elapsed time in one of these cases is greater or equal t_{sample}, the monitor enters an *error* location.

HyTech can analyze the resulting model in 73 seconds and verifies that *error* with a measured time greater than t_{sample} is unreachable from the system's initial states. This means that *suspended* is always entered at most t_{sample} time units after a bend which lasts longer than t_{sample} started. It is left again after at most t_{sample} time units after the bend ended, if no other bend started within t_{sample} time units after the first. As we have used backward reachability analysis starting from location *error* in this verification, locations A and F of the filter may be visited during the analysis. Their flow conditions which are not justified by the nonlinear filter automaton can influence the analysis by giving incorrect values for f. We argue that the actual values of f are not crucial in this verification task since they do not affect entry and exit of *suspended*.

4.4 Step Response of the EHC

In this section we examine the response of the EHC to a disturbance of step shape. Disturbances of this kind are typical test functions in control systems engineering, they can be used to examine stability of control systems. We use this specific environment scenario to analyze whether the activation of compressor and escape valve terminates after a step-like disturbance.

Fig. 9 shows the hybrid automaton for the environment we use to model such a disturbance. This automaton replaces the environment model we used in the previous sections. Starting with a chassis level $h = sp$, h may make a jump by j units anytime. No other disturbances are possible besides this single jump, i. e. h can only be manipulated by the controller in locations *zero* and *disturbed*, denoted by flow condition $\dot{h} = \dot{c}$ in these locations. As the *step* transition changes the value of h, the approximation technique we used for the filter in section 3.2 demands that it is taken simultaneously with a filter transition which allows control to pass to another filter location. This is achieved by adding new transitions with jump condition *True* and synchronization label *step* form every filter location in $\{A, ..., F\}$ to every location in $\{A, ..., F\}$.

To do a complete analysis of the system response to a step-like disturbance, we would like to regard steps of arbitrary height j, or at least all step heights which are allowed by the mechanics of the suspension system. Unfortunately the disturbances we can analyze with HyTech are restricted, as the values we use in our system can cause an arithmetic overflow thereby aborting analysis. Therefore we only analyze the controllers response in mode *driving* to steps with $j \in (16\,mm, 18\,mm]$ as an example. Values of j in this interval are interesting, as we expect the controller to use the escape valve after disturbances of this size.

Apart from that we modify the controller of Fig. 5 by not giving intervals for the performance of compressor and escape valve in up_d and $down_d$. Instead we use the conservative assumption that they operate at their minimum rates, $\dot{c} = cp_{min}$ in up_d and $\dot{c} = ev_{max}$ in $down_d$. This restriction was necessary to avoid arithmetic overflows using HyTech.

By adding a monitor automaton to our model we use HyTech's parametric analysis facility to compute that the controller leaves location *in_tolerance* at most $4.3s$ after the disturbance and reenters it after at most $22.3s$. The chassis level then lies in $[-1\,mm, 6\,mm]$ and *in_tolerance* is not left again. The analysis takes 370 seconds of CPU time.

5 Conclusion

We have shown that HyTech can successfully analyze several interesting properties of an abstract version of the electronic height control system. Nevertheless, we reached the efficiency limits of the tool. We believe that the nonlinear behavior of the filter and the necessarily applied approximation technique is the main reason for the obtained complexity. The major efficiency problem of HyTech are overflows caused by the arithmetic libraries which are very sensitive to the specific values of the constants of the model. As the user does not get any feedback on which constants have caused the overflow, trial and error must be used to identify them. In our models the use of fractions turned out to be disadvantageous as far as overflows are concerned. Therefore we only use integer constants in the current models. The problem of overflows could perhaps be weakened by developing routines which are optimized for the use with HyTech instead of

using standard libraries. Running out of memory is less problematic, because the reasons for this can be found easier than those for overflows.

Modeling the influences of the road, i.e. the environment of the system, was a major challenge of the case study. On the one hand the disturbances of the environment can be stronger than the controller for a short time, on the other hand they should be weaker than the controller when the average influence over a longer time period is regarded. We believe that the limits of expressiveness of hybrid automata are reached with complex environment models of this kind. Furthermore the high degree of non-determinism in such a model makes automated analysis extremely difficult if not impossible.

We investigated the interference of parallel composition and linear phase-portrait approximation, identified criteria for an approximation of a single component of a composition and applied them to the filter in our case study.

This case study may be a first step towards more industrial interest in the modeling and analysis of hybrid systems. We are interested in how other methods and tools can handle this case study and believe that it can serve as a real-life benchmark for the analysis of embedded systems.

Acknowledgement. We thank Howard Wong-Toi for his numerous hints on the use of HyTech.

References

[ACH+95] R. Alur, C. Courcoubetis, N. Halbwachs, T. A. Henzinger, P.-H. Ho, X. Nicollin, A. Olivero, J. Sifakis, and S. Yovine. The algorithmic analysis of hybrid systems. *Theoretical Computer Science*, 138:3–34, 1995.

[AHS96] R. Alur, T.A. Henzinger, and E.D. Sontag. *Hybrid Systems III.* Lecture Notes in Computer Science 1066. Springer-Verlag, 1996.

[HH95] T.A. Henzinger and P.-H. Ho. Algorithmic analysis of nonlinear hybrid systems. In P. Wolper, editor, *CAV 95: Computer-aided Verification*, Lecture Notes in Computer Science 939, pages 225–238. Springer-Verlag, 1995.

[HHWT95] T.A. Henzinger, P.-H. Ho, and H. Wong-Toi. A user guide to HYTECH. In E. Brinksma et al., editors, *Tools and Algorithms for the Construction and Analysis of Systems*, LNCS 1019, pages 41–71. Springer-Verlag, 1995.

[HWT95] P.-H. Ho and H. Wong-Toi. Automated analysis of an audio control protocol. In P. Wolper, editor, *CAV 95: Computer-aided Verification*, Lecture Notes in Computer Science 939, pages 381–394. Springer-Verlag, 1995.

[HWT96a] T.A. Henzinger and H. Wong-Toi. Linear phase-portrait approximations for nonlinear hybrid systems. In R. Alur, T.A. Henzinger, and E.D. Sontag, editors, *Hybrid Systems III*, LNCS 1066. Springer-Verlag, 1996.

[HWT96b] Thomas A. Henzinger and H. Wong-Toi. Using HYTECH to synthesize control parameters for a steam boiler. In J.-R. Abrial et al., editors, *Formal Methods for Industrial Applications: Specifying and Programming the Steam Boiler Control*, LNCS 1165. Springer-Verlag, 1996.

[PH88] Charles L. Phillips and Royce D. Harbor. *Feedback Control Systems.* Prentice-Hall, 1988.

[Sta97] Thomas Stauner. *Specification and Verification of an Electronic Height Control System using Hybrid Automata.* Master thesis, Munich University of Technology, 1997.