EPiC Series in Computing
Volume 43, 2017, Pages 68–79
ARCH16. 3rd International Workshop on Applied Verification for Continuous and Hybrid Systems

HyReach: A Verification Tool for Linear Hybrid Systems Based on Support Functions (Tool Presentation)

Ibtissem Ben Makhlouf[1], Norman Hansen[1], and Stefan Kowalewski[1]

Informatik 11 - Embedded Software RWTH Aachen University
Ahornstrasse 55, 52074 Aachen, Germany
makhlouf,hansen,kowalewski@embedded.rwth-aachen.de

Abstract

We present HyReach, a MATLAB-based toolbox for reachability analysis of linear hybrid systems based on support functions. The main goal of HyReach is to provide a single graphical user interface for easily configurable reachability analysis on different systems. HyReach offers a number of known algorithms for the computation of reachable sets within modes. These are combined with various approaches of handling mode invariants, guard intersection computations and transition strategies. Furthermore, plug-ins like the MPT and CVX toolboxes complement the existing MATLAB optimization toolbox to increase the variety of optimization algorithms for the computation of support functions. HyReach supports both textual and graphical inputs and outputs, allowing for flexibility and seamlessness in workflow and processing of data. We illustrate these features with examples in this paper.

1 Introduction

Reachability analysis of hybrid systems has been a topic of extensive research over the past years. New approaches and innovative ideas have contributed to significant progress in this field. Several tools have been developed and have been made available. A range of methods and procedures have been recently proposed that improve upon the performance of existing approaches. The question hereby is whether it is worthwhile to implement these methods because of their complexity in comparison with their small contribution. The evaluation of this contribution requires a practical comparison of different methods in the same environment and under the same conditions. In practice many implementations are available but in different development environments. Integrating them within a single platform demands time and effort. The platform SpaceEx [7] can be seen as a first step in this direction. Besides the PHAVer scenario [5], SpaceEx includes implementations of the Le Guernic-Girard LGG-algorithm [15] and its recent enhancement, the STC-algorithm [6]. The first and the second toolboxes are stand alone implementations. They receive their inputs and setting parameters via the SpaceEx GUI, which then visualizes their results. Otherwise they do not share any other components.

Besides the algorithmic core, many approaches have been proposed over the last couple of years for computing an over-approximation of the input contribution and the initial set.
We aim with HyReach to offer an analysis framework incorporating a number of known algorithms for the computation of reachable sets within modes and various approaches for handling invariants, guard intersection and transition selection strategies. Our main goal, however, is to allow a fully user configurable reachability analysis for linear hybrid systems. Similar to existing reachability analysis tools, e.g. [1, 19, 17, 18], we chose Matlab as the underlying platform for HyReach.
The paper is structured as follows. Section 2 briefly reviews the definition and some properties of support functions relevant for our implementation. A description of the structure and the features of the tool is given in Sections 3 and 4. Experimentation and testing results are then presented and shortly discussed in Section 5. Section 6 concludes this work by listing some remarks and directions for future work.

2 Support Functions

In HyReach, reachable sets are represented by their support functions. A support function is an alternative representation of convex geometric sets. Let $S \subseteq \mathbb{R}^n$ be a convex set and $l \in \mathbb{R}^n$ a direction. The support function of S in the direction l is defined as follows

$$\begin{aligned} \rho_S : \quad & \mathbb{R}^n \longrightarrow \mathbb{R} \cup \{\pm\infty\} \\ & l \longmapsto \rho_S(l) := sup_{x \in S} \langle l, x \rangle \end{aligned} \tag{1}$$

where $\langle l, x \rangle$ is the dot product. Furthermore support functions obey the following properties: Let S, S_1, $S_2 \subseteq \mathbb{R}^n$ be nonempty sets, l, l_1, $l_2 \in \mathbb{R}^n$, $A \in \mathbb{R}^{n\times m}$ and $\lambda \geq 0$ [9, 16],

1. $\rho_{\lambda S}(.) = \lambda \rho_S(.)$
2. $\rho_S(\lambda l) = \lambda \rho_S(l)$
3. $\rho_S(l_1 + l_2) = \rho_S(l_1) + \rho_S(l_2)$
4. $\rho_{AS}(l) = \rho_S\left(A^T l\right)$
5. $\rho_{S_1 \oplus S_2}(.) = \rho_{S_1}(.) + \rho_{S_2}(.)$, with $\oplus$ the Minkowski sum
6. $\rho_{ConvexHull(S_1 \cup S_2)}(.) = max\left(\rho_{S_1}(.), \rho_{S_2}(.)\right)$
7. $\rho_{S_1 \cap S_2}(l) \leq min\left(\rho_{S_1}(l), \rho_{S_2}(l)\right)$

These properties allow for the transformation of hard geometric operations of n-dimensional sets into ordinary algebraic operations. Consequently these transformations are used to derive an efficient computation of reachable sets.

3 Tool Architecture

The architecture of HyReach is illustrated in Fig.1. It consists of an interactive graphical user interface (GUI) and a computation core (CC). HyReach provides a GUI shown in Fig.2 for selection of parameters and to provide an overview of the overall analysis. Various options are provided from which the user can select to configure the reachability analysis. The GUI can be viewed as being composed of four components:
The input component: The user chooses the input file of the hybrid automaton. The input file can be a `.m` file where the hybrid automaton is described textually or a `.mdl` file with a graphical representation. The state and the input variables are then displayed in the

corresponding fields.

The user parameter component: We consider time step, time horizon, maximum number of iterations, input and the initial sets as user-provided parameters. If the input is textual, the `.m` file will include the values of these parameters, which can be directly imported and entered into the corresponding GUI fields if this option is chosen. The user is free to change these values thereafter. In the case a graphical representation of the hybrid automaton is used, those parameters must then be provided directly via the GUI.

The analysis setting component: The user chooses the general algorithm, as well as possible optimization algorithms for the computation of the reachable sets within continuous modes. The method for computing the intersection of the reachable set with the guard condition can also be selected between an equality and inequality guard condition. Clustering methods for handling transitions are also among the possible options made available to the user. The user can also choose between triggering the transition after a user-defined time or after reaching a fixed point with a user-defined tolerance. Several options have been also incorporated to allow the user to define how nondeterminism of transitions should be handled. The choice of directions for the evaluation of the support function is also left to the user.

The output component: The GUI captures the results from the CC and displays them in a textual and graphical form. The state and the input variables are displayed in the corresponding fields shown in Fig.2 as soon as the graphical input file is loaded. If more than one transition can be triggered within the chosen time horizon T, the user has the possibility to decide which transition will be followed when the option *Manual choice* is selected for *Transition handling*. If the *Output interval choice* is selected by the plot options, the interval range of all state variables are shown on the right side of the GUI. Upon completion of reachability analysis, the reachable sets are plotted in the form of 2D-projections along with its computation and plot times. The plot options field allows for the choice of two dimensions to be projected, as well as the export and saving of different plots. The time progress of the computation is visualized by a color gradient, beginning with green sets and ending with red ones. The enclosure of the sets

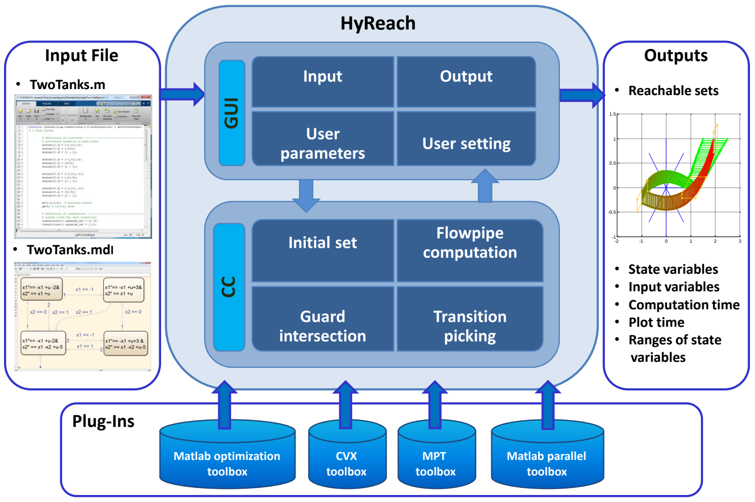


Figure 1: HyReach architecture.

intersecting the guards are plotted in yellow.
Concerning the computation core (CC), principle components are detailed in the next section.

4 Options to Configure the Reachability Analysis

We briefly describe the different options and scenarios, which the user can select to configure the reachability analysis. These options represent a wide range of algorithms from the MATLAB optimization toolbox, the CVX [10] and the MPT [13] toolboxes .

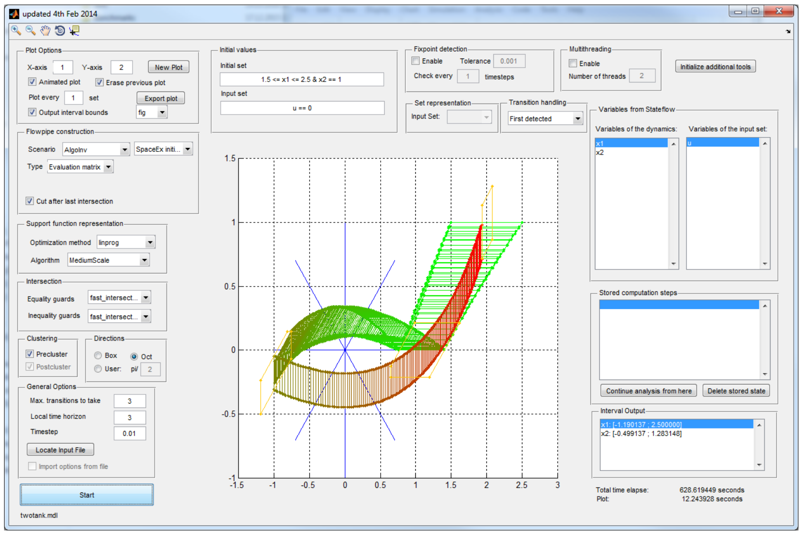
For the computation of the reachable sets in continuous modes, only dynamics of the form $\dot{x} = Ax + Bu + b$, where x is the state vector, u the input vector in a convex set V, b a constant vector, A and B the constant matrices, is currently allowed. For a given time step r, time horizon T, the over-approximations of the initial set Ω_0 and the input contribution $\mathcal{V}_r$, the reachable set Ω_k at time $t = kr$ can be recursively obtained by making use of the set equation $\Omega_k = \Phi\Omega_{k-1} \oplus \mathcal{V}_r$ where $\Phi = e^{rA}$. For the case of support functions, the formal equation is as follows:

$$\begin{aligned}\rho_{\Omega_k}(l) &= \rho_{\Phi\Omega_{k-1}}(l) + \rho_{\mathcal{V}_r}(l)\\ &= \rho_{\Omega_{k-1}}(\Phi^T l) + \rho_{\mathcal{V}_r}(l)\\ &= \rho_{\Omega_0}({\Phi^T}^k l) + \sum_{j=0}^{k-1} \rho_{\mathcal{V}_r}\left({\Phi^T}^j l\right).\end{aligned} \tag{2}$$

This equation represents the core of the recursive scheme for the computation of the flowpipe within continuous modes. Depending on the implementation and the over-approximation of the initial set and the input contribution, five scenarios are made available:

1. The *NoScale* scenario is an implementation of the approach detailed in [15] with the corresponding input and initial set over-approximations.

2. The *ConstU* scenario is based on the assumption that the input remains constant within small time steps. The basic algorithm is given in [2].



X-axis 1 Y-axis 2
[✓] Animated plot [✓] Erase previous plot
Plot every 1 set
[✓] Output interval bounds fig
Initial set: 1.5 <= x1 <= 2.5 & x2 == 1
Input set: u == 0
Fixpoint detection: [ ] Enable Tolerance 0.001 Check every 1 timesteps
Multithreading: [ ] Enable Number of threads 2
Set representation: Input Set:
Transition handling: First detected
Variables of the dynamics: x1, x2
Variables of the input set: u
Scenario: AlgoInv SpaceEx initi...
Type: Evaluation matrix
[✓] Cut after last intersection
Optimization method: linprog
Algorithm: MediumScale
Equality guards: fast_intersect...
Inequality guards: fast_intersect...
Clustering: [✓] Precluster [✓] Postcluster
Directions: Box [✓ Oct] User: pi/ 2
Max. transitions to take 3
Local time horizon 3
Timestep 0.01
[ ] Import options from file
twotank.mdl
x1: [-1.190137 ; 2.500000]
x2: [-0.499137 ; 1.283148]
Total time elapse: 628.619449 seconds
Plot: 12.243928 seconds

Figure 2: Screenshot of the GUI.

3. The *SpaceEx* scenario is an implementation of the approach described in [7] with a constant time step.
4. The *AlgoInv* scenario is an improved implementation of the algorithm suggested in [14]. Contrary to previous scenarios, this scenario is able to handle invariants within continuous modes. The check and the computation of the intersection with an invariant described as hyperplanes or halfspaces is done using property 7 of support functions.
5. The *AlgoInv2* scenario is an alternative implementation of *AlgoInv* allowing a parallelization of the computing process.

For the initialization of the *ConstU* scenario, the user can choose between the *ConstU*, *PreciseOmega0* and *SpaceEx* over-approximation methods for the initial set. These methods are described in [2], [14] and in [7] respectively. However, if *SpaceEx*, *AlgoInv* and *AlgoInv2* scenarios are chosen, only the *PreciseOmega0* and *SpaceEx* can be used for initialization.

We mentioned that the handling of intersections with guards is based on property 7 of support functions. However, owing to the inequality, this property can only result in an over-approximation of the intersection. A possible improvement of this approximation can be obtained by recalling the following property

$$\rho_{S_1 \cap S_2}(l) = \inf_{w \in \mathbb{R}^n} \left(\rho_{S_1}(w) + \rho_{S_2}(l - w) \right), \tag{3}$$

where $S_1, S_2 \subseteq \mathbb{R}^n$ are two convex sets and $l \in \mathbb{R}^n$ [8].

For the case of guards defined as hyperplanes $H_p = \{x \in \mathbb{R}^n : \langle d, x \rangle = e\}$ with $d \in \mathbb{R}^n$ and $e \in \mathbb{R}$, the intersection according to (3) can be reduced to the minimization of the following univariate linear piecewise function

$$\rho_{S \cap H_p}(l) = \inf_{\lambda \in \mathbb{R}} \left(\rho_S(l - \lambda d) + \lambda e \right). \tag{4}$$

For guards defined as halfspaces $H_s = \{x \in \mathbb{R}^n : \langle d, x \rangle \leq e\}$, the intersection is obtained by solving the following minimization problem

$$\rho_{S \cap H_s}(l) = \inf_{\lambda \in \mathbb{R}^+} \left(\rho_S(l - \lambda d) + \lambda e \right). \tag{5}$$

We implemented the following five methods for the intersection of reachable sets with a hyperplane.

- *fminsearch* is a direct Nelder-Mead sequential simplex algorithm, part of the MATLAB optimization toolbox.
- *fminunc* provides an alternative iterative method for solving unconstrained nonlinear optimization problems based on the Broyden Fletcher Goldfarb Shanno (BFGS) approach.
- *dichotomicSearch* is a reimplementation of the dichotomic search method proposed in [14].
- *RayAlgo* is an enhancement of the sandwich approach suggested in [8].
- *fast intersection* is an application of property 7 of support functions.

For the intersection with halfspaces, the following options are made available:

- *fminbnd* is a combination of the golden section search and the parabolic interpolation methods and is a part of the MATLAB optimization toolbox.
- *fmicon* is a Sequential Quadratic Programming-based (SQP) algorithm provided by the MATLAB optimization toolbox.
- *RayAlgo* as above.
- *fast intersection* as above.

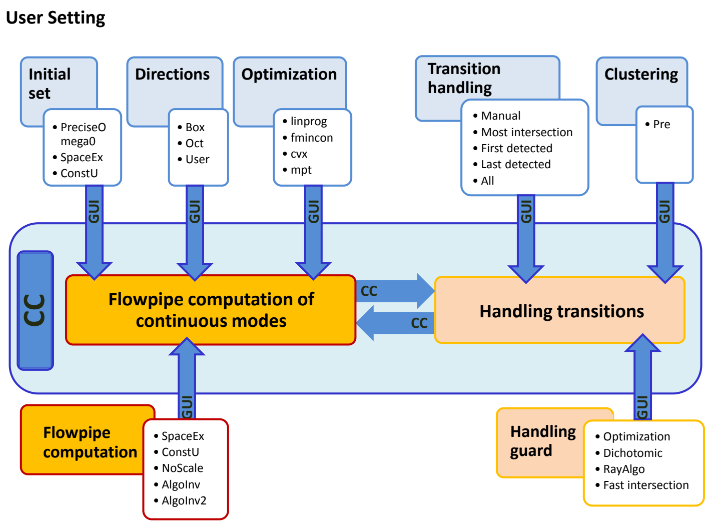


Figure 3: User setting possibilities for the configuration of the reachability analysis.

5 Experimentation and Testing

Our prototyping implementation is evaluated using the suite of the linear benchmarks described in [11]. The toolbox and the input files of the selected benchmarks are available as open-source [12]. Essential characteristics of the tested benchmarks are summarized in Table 1. The benchmarks differ not only in the description of their linear dynamics but considerably in their complexity. In fact, the complexity increases on one hand with the dimension of the system and on the other hand with the number of locations and the transitions between them. The nature and the number of conditions describing the guards has also a crucial impact on the complexity. In addition, the presence of invariants inside the locations adds to the complexity of the proposed benchmarks. We tested the scalability of the tool with respect to the number of variables, locations and transitions. Different parameter settings and scenarios for the flowpipe constructions were furthermore investigated. Table 2 shows for each benchmark of Table 1 the corresponding parameter setting and some selected results obtained using HyReach. For this evaluation, the *MPT-CDD-Criss-Cross* algorithm for the support function computation and the *fast-intersection* for both equality/inequality-guard intersections have been selected. We used an Intel Core i5-2520M @ 2.50GHz laptop with 8GB of RAM and Matlab R2014a. As results, we note in Table 2 the time complexity and the resulting interval enclosures of the reachable sets. The reachable sets are given as 2D-projection plots. We present for brevity just a few of them. We choose as illustrative examples the cruise control, the transient in flower, the two tanks and all navigation benchmarks proposed in Table 1. Their corresponding reachable sets are given in Fig.4. The controlled platoon of three trucks (3V-P) described in [2, 3] has been considered separately for testing our spontaneous transition implementation with the fixed point option. The graphical description of the benchmark is illustrated in Fig.5(a). In the corresponding textual description, this particular kind of transition is coded as follows:
$transitions\{location\ nb.\}.spontaneous = true;$
$transitions\{location\ nb.\}.timeelapse = -1;$
We chose via GUI the tolerance $\epsilon = 0.001$ as equality tolerance for checking the fixed point. The

Benchmark	continuous dynamics	dimension	modes	invariants	transitions	guards	guard condition	reset
1. BB: Bouncing ball	Ax+Bu+b	4	1	yes	loop	multiple	1 eq.+ 1 ineq.	yes
2. CM: Colliding masses	Ax+Bu	4	1	yes	loop	1	1 eq.	yes
3. CC: Cruise control	Ax+b	3	6	yes	11	multiple	1 eq.+ 4 ineq.	yes
4. Flower: transient in flower	Ax	2	4	yes	2	1	1 eq.	no
5. TT: 2-tanks	Ax+Bu+b	2	4	yes	7	1	1 eq.	no
6. Nav3x3	Ax+Bu	4	7	yes	16	multiple	1 eq.+ 2 ineq.	no
7. Nav4x4	Ax+Bu	4	14	yes	34	multiple	1 eq.+ 2 ineq.	no
8. Nav5x5	Ax+Bu	4	23	yes	58	multiple	1 eq.+ 2 ineq.	no
9. 3R-2MH: 3 rooms+ 2 movable heaters	Ax+b	3	7	yes	22	multiple	2 ineq.	no
10. 5D-LSS	Ax+Bu	5	5	no	5	1	1 eq.	no
11. 3V-P: 3-vehicle-platoon	Ax+Bu	9	2	no	2	spontaneous	-	no
12. 5V-P: 5-vehicle-platoon	Ax+Bu	15	1	no	-	-	-	no

Table 1: Essential characteristics of the benchmarks used for the tool assessment.

test of equality between reachable set using tolerance ϵ was set via the GUI to take place every time step. Different 2D-projections of the computed reachable sets are plotted in Fig.5(b-f). The blue sets there correspond to different fixed points found in each location. For the test of scalability, we used the scalable LQR-based platoon proposed in [4] as benchmark. The last row in Table 1 corresponds to this benchmark with a number of vehicles $n = 5$. Each vehicle extends the state vector of the platoon with three new states, the gap to the vehicle ahead, its derivative and the acceleration. The dimension of a platoon of n vehicles is therefore equal to $3n$. We performed further tests for $n = 10, 15, 20$, with the same parameter setting as in Table 2. We noted as expected a considerable increase in the computation time with the number of vehicles. But the computation ended with the intended results, namely all 2D-plots and interval enclosures. We were even able to obtain similar results with a time horizon $T = 30s$ and a time step $r = 0.001$.
Furthermore, we performed a comparative evaluation of the available methods and their possible combinations using the same suite of benchmarks. Details and results of this comparative study will be the topic of a future publication.

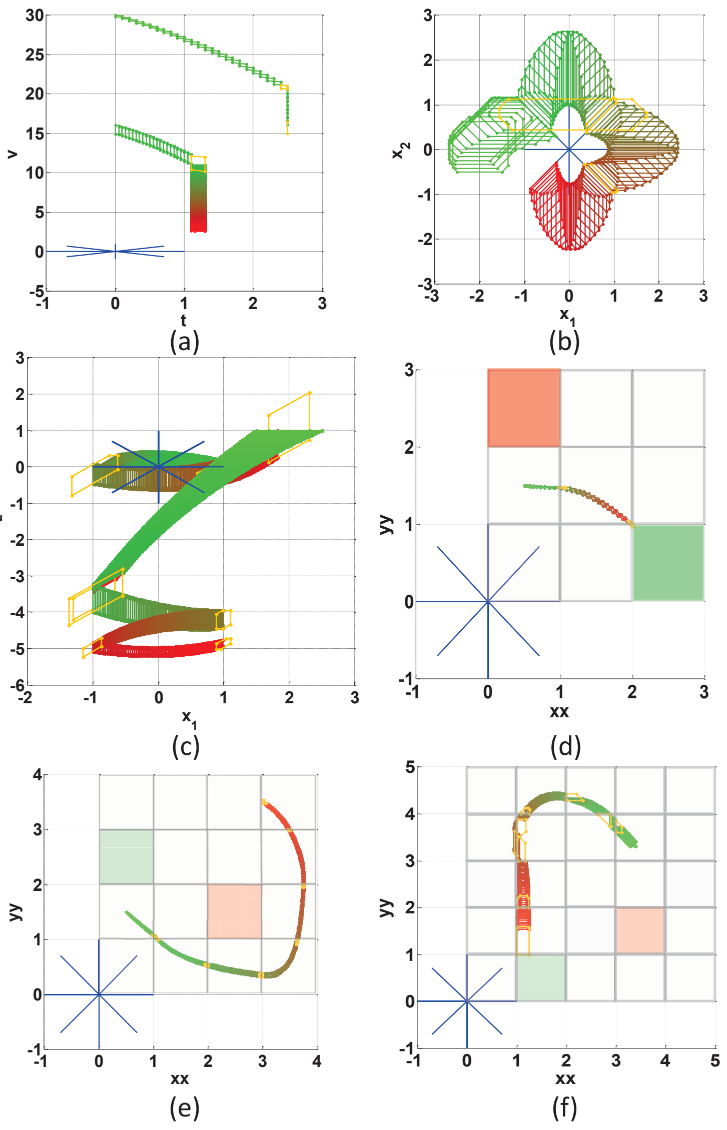
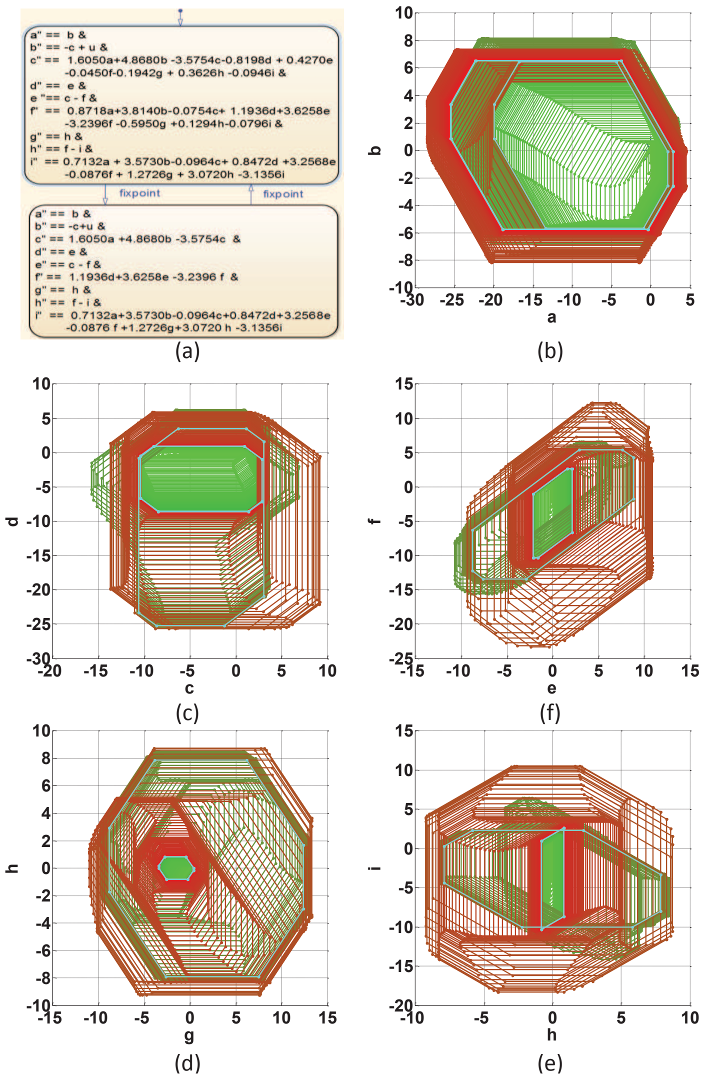


Figure 4: The resulting flowpipes of some benchmarks of Table 2. (a) The flowpipe of the cruise control as projected on the plane (t,v). (b) The flowpipe of the trajectory in flower benchmark. (c) The flowpipe of the two-tank benchmark. (d) The flowpipe of the navigation 3x3 benchmark as projected on the plane (xx,yy). (e) The flowpipe of the navigation 4x4 benchmark as projected on the plane (xx,yy). (d) The flowpipe of the navigation 5x5 benchmark as projected on the plane (xx,yy).

6 Conclusion

We present the toolbox HyReach for the computation of reachable sets of linear hybrid systems with uncertain inputs. It allows for different scenarios for the computation of the flowpipe inside continuous modes with and without invariants and also for different initial set options. It offers



```
a" ==  b &
b" == -c + u &
c" ==  1.6050a+4.8680b -3.5754c-0.8198d + 0.4270e
       -0.0450f-0.1942g + 0.3626h -0.0946i &
d" ==  e &
e "== c - f &
f' ==  0.8718a+3.8140b-0.0754c+ 1.1936d+3.6258e
       -3.2396f -0.5950g +0.1294h-0.0796i &
g" == h &
h" == f - i &
i" == 0.7132a + 3.5730b-0.0964c+ 0.8472d +3.2568e
       -0.0876f + 1.2726g + 3.0720h -3.1356i
```

```
a" ==  b &
b" == -c+u &
c" == 1.6050a +4.8680b -3.5754c  &
d" == e &
e" == c - f &
f' ==  1.1936d+3.6258e -3.2396 f  &
g" ==  h &
h" ==  f - i &
i" ==  0.7132a+3.5730b-0.0964c+0.8472d+3.2568e
       -0.0876 f +1.2726g+3.0720 h -3.1356i
```

Figure 5: The graphical input file of the platoon with three vehicles (a) and the 2D-projections of the reachable sets obtained with the fixed point triggered transition option and the parameter setting in Table 2 (a) The hybrid model (b-f) Different flowpipe projections.

the possibility to choose between different optimization algorithms for the computation of the support function using the Matlab optimization toolbox or the MPT and the CVX toolboxes as plug-ins. Furthermore, the direction template is chosen by the user. For handling transitions with equality and also inequality guards, many methods are made available in HyReach. We present some experiences and comparative evaluations of the available methods and algorithms using a suite of benchmarks. We note in general that the performances of these different methods and algorithms are similar for small dimensional benchmarks with small number of transitions and small number of directions. Once, one of these threeproperties becomes large, some methods and algorithms are shown to be more efficient than the others.

Within the HyPro-project [11], we are actually implementing a library for different representations of state sets for hybrid systems and their different geometric operations. This library includes orthogonal rectangles, zonotopes, polytopes, support functions and Taylor models. We are working towards the goal of a user-configurable reachability analysis.

References

[1] Matthias Althoff. An introduction to cora 2015. In *Proc. of the Workshop on Applied Verification for Continuous and Hybrid Systems*, 2015.

[2] I. Ben Makhlouf, H. Diab, and S. Kowalewski. Safety verification of a controlled cooperative platoon under loss of communication using zonotopes. In *ADHS12, Eindhoven, NL*, pages 333–338. In proceeding of the 4th IFAC Conference on Analysis and Design of Hybrid Systems, 2012.

[3] I. Ben Makhlouf and S. Kowalewski. Networked cooperative platoon of vehicles for testing methods and verification tools. In Goran Frehse and Matthias Althoff, editors, *ARCH14-15. 1st and 2nd International Workshop on Applied veRification for Continuous and Hybrid Systems*, volume 34 of *EPiC Series in Computing*, pages 37–42. EasyChair, 2015.

[4] I. Ben Makhlouf and S. Kowalewski. Optimizing safe control of a networked platoon of trucks using reachability. In Goran Frehse and Matthias Althoff, editors, *ARCH14-15. 1st and 2nd International Workshop on Applied veRification for Continuous and Hybrid Systems*, volume 34 of *EPiC Series in Computing*, pages 169–179. EasyChair, 2015.

[5] G. Frehse. PHAVer: Algorithmic verification of hybrid systems past hytech. In *HSCC'05, LNCS 3414*, pages 258–273. Springer-Verlag, 2005.

[6] G. Frehse, R. Kateja, and C. Le Guernic. Flowpipe approximation and clustering in space-time. In *Proc. Hybrid Systems: Computation and Control (HSCC'13)*, pages 203–212. ACM, 2013.

[7] 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 Shaz Qadeer Ganesh Gopalakrishnan, editor, *Proc. 23rd International Conference on Computer Aided Verification (CAV)*, LNCS. Springer, 2011.

[8] G. Frehse and R. Ray. Flowpipe-guard intersection for reachability computations with support functions. In *IFAC Conf. Analysis and Design of Hybrid Systems (ADHS)*, pages 94–101, 2012.

[9] P. K. Ghosh and K. V. Kumar. Support function representation of convex bodies, its application in geometric computing, and some related representations. *Computer Vision and Image Understanding*, 72(3):379–403, 1998.

[10] M. Grant and S. Boyd. CVX: Matlab software for disciplined convex programming, version 2.1. http://cvxr.com/cvx, mar 2014.

[11] HyPro Project. Benchmarks of continuous and hybrid systems. https://ths.rwth-aachen.de/research/projects/hypro/benchmarks-of-continuous-and-hybrid-systems/, 2014.

[12] HyReach. Software package and examples. https://embedded.rwth-aachen.de/doku.php?id=en:tools:hyreach, 2015.

[13] M. Kvasnica, P. Grieder, and M. Baotić. Multi-Parametric Toolbox (MPT), 2004.

[14] C. Le Guernic. *Reachability Analysis of Hybrid Systems with Linear Continuous Dynamics*. PhD thesis, VERIMAG, Oct. 2009.

[15] C. Le Guernic and A. Girard. Reachability analysis of hybrid systems using support functions. In *Proceedings of the 21st International Conference on Computer Aided Verification*, CAV '09, pages 540–554, Berlin, Heidelberg, 2009. Springer-Verlag.

[16] C. Le Guernic and A. Girard. Reachability analysis of hybrid systems using support functions. In *Computer Aided Verification, 21st International Conference, CAV 2009, Grenoble, France, June 26 - July 2, 2009. Proceedings*, pages 540–554, 2009.

[17] Hai Lin, X. D. Koutsoukos, and P. J. Antsaklis. Hystar: a toolbox for hierarchical control of piecewise linear hybrid dynamical systems. In *American Control Conference, 2002. Proceedings of the 2002*, volume 1, pages 686–691 vol.1, 2002.

[18] Ian M. Mitchell. The flexible, extensible and efficient toolbox ofÂ levelÂ set methods. *Journal of Scientific Computing*, 35(2):300–329, 2007.

[19] Ricardo Sanfelice, David Copp, and Pablo Nanez. A toolbox for simulation of hybrid systems in matlab/simulink: Hybrid equations (hyeq) toolbox. In *Proceedings of the 16th International Conference on Hybrid Systems: Computation and Control*, HSCC '13, pages 101–106, New York, NY, USA, 2013. ACM.

Ben. nb.	**T(s)**	**r (s)**	**Max. trans.**	**Flowp. constr.**	**Init. approx.**	**Dir.**	**Init./Inp. set**	**Trans. picker**	**Time (s)**	**Intervals**
1.	1	0.01	4	AlgoInv	SpaceEx	Oct	x1=2,x2=0	First	33.499	x1:[0;2] x2:[-6.289;3.762]
2.	1	0.01	4	AlgoInv	SpaceEx	Oct	a=0,b=3, c=2,d=-1	First	64.417	a:[0;2], b:[2;3] c:[-2;2], d:[-1;1]
3.	20	0.1	10	AlgoInv	SpaceEx	Oct	v=30,x=0, t=0	First	116.419	v:[2.587;30] x:[0;127.988] t:[0;2.5]
4.	20	0.05	10	AlgoInv2	SpaceEx	Oct	-2.5<=x1 <=-1.5, x2=0	Most	19.606	x1:[-2.694;2.422] x2:[-2.224;2.637]
5.	2	0.01	4	SpaceEx	SpaceEx	Oct	1.5<=x1 <=2.5, x2=1/ -0.1<=u <=0.1	All	119.214	x1:[-1.366;2.5] x2:[-5.229;2.036]
6.	20	0.05	4	AlgoInv2	SpaceEx	Oct	xx=0.5, yy=1.5 -0.01<=vx, vy<=0.01	Last	179.783	xx:[0.5;2.031] yy:[0.942;1.5] vx:[-0.010;0.749] vy:[-0.527;0.010]
7.	20	0.05	8	AlgoInv	SpaceEx	Oct	xx=0.5, yy=1.5 -0.01<=vx, vy<=0.01	Last	629.664	xx:[0.5;3.771] yy:[0.318;2.048] vx:[-0.010;0.919] vy:[-0.526;0.948]
8.	20	0.05	10	AlgoInv	SpaceEx	Oct	3.3<=xx, yy<=3.4 -0.01<=vx, vy<=0.01	Last	557.346	xx:[0.945;3.4] yy:[1.0;4.446] vx:[-0.882;0.299] vy:[-0.969;0.569]
9.	2	0.01	8	AlgoInv2	SpaceEx	Oct	x1=x2= x3=20	First	446.036	x1:[12.275;20] x2:[13.997;20.536] x3:[12.664;21.270]
10.	2	0.01	10	ConstU	ConstU	Oct	a=3,b=4, c=d=e=0 / -0.01<=u <=0.01	First	87.160	a:[-4.185;4.538] b:[-1.163;6.824] c:[-4.018;1.129] d:[-0.681;8.308] e:[-0.791;5.421]
11.	22	0.1	3	ConstU	ConstU	Oct	a=b= c=d= e=f= g=h= i=0	First (Fixpoint Tol.: 0.001 check every r)	378.133	a:[-28.540;4.479] b:[-8.171;8.184] c:[-15.803;9.163] d:[-25.634;6.274] e:[-10.740;10.919] f:[-23.285;12.267] g:[-11.061;13.316] h:[-9.252;8.729] i:[-18.304;10.517]
12.	15	0.1	1	ConstU	ConstU	Oct	d1=d2= d3=d4= d5=d6= d7=d8= d9=d10= d11=d12= d13=d14= d15=0	First (irre- levant)	52.911	d1:[-31.438;3.958] d2:[-7.369;8.219] d3:[-10.450;2.830] d4:[-15.228;2.109] d5:[-3.516;3.602] d6:[-10.994;3.262] d7:[-9.690;1.411] d8:[-2.141;2.206] d9:[-11.288;3.533] d10:[-5.927;0.890] d11:[-1.275;1.321] d12:[-11.459;3.700] d13:[-2.836;0.433] d14:[-0.600;0.625] d15:[-11.539;3.779]

Table 2: Some obtained results of the benchmark list of Table 1.