Computing Bounded Reach Sets from Sampled Simulation Traces

Zhenqi Huang & Sayan Mitra

Department of Electrical and Computer Engineering, University of Illinois at Urbana-Champaign

Introduction

- Bounded Safety Verification
- Two Existing Approaches (a) **Static analysis** and (b) **Dynamic analysis---testing and simulation**
- This work combines (a) and (b)
- Compute the set of states can be reached by any execution which can generate a given trace.

Assumptions

Model

- The differential equations in each location should be *Lipchitz continuous*.
- There is a *minimum dwell time* (no Zeno executions).
- The transitions of the system is *deterministic*.

Simulation Trace

- The (k + 1)th sample point in the trace is at most *c* distance away from the execution starting from the kth sampled point.
- Each sampled point records the true location of the real execution.

Algorithm

Our overapproximation algorithm involves two steps.

1. Estimating Accumulated Error

- We bound the error between *each sample point* ($\mathbf{v}_k$,t_k) and the *true execution α*. Illustrated in Fig. 1a.

- **Lemma 1:** If $\forall t \in [t_k, t_{k+1}], \alpha(\mathbf{v}_0, t).loc = i$ for some location i then,
$$\varepsilon_{k+1} = \varepsilon_k e^{L_i(t_{k+1}-t_k)} + c.$$

- **Corollary 1:** In addition, if the continuous states evolve follows $\dot{X} = A_i X + B_i$ then,
$$\varepsilon_{k+1} = \varepsilon_k ||e^{A_i(t_{k+1}-t_k)}|| + c.$$

- **Lemma 2**: If $\exists \eta \in [t_k, t_{k+1}]$ such that a single transition from location i to j occurs in $\alpha(\mathbf{v}_0, t)$, then there exists a constant $M > 0$ such that
$$\varepsilon_{k+1} = \left(\varepsilon_k + \frac{M}{L_j}\right) e^{L\delta} - \frac{M}{L_j} + c.$$
where $L = \max\{L_i, L_j\}$.

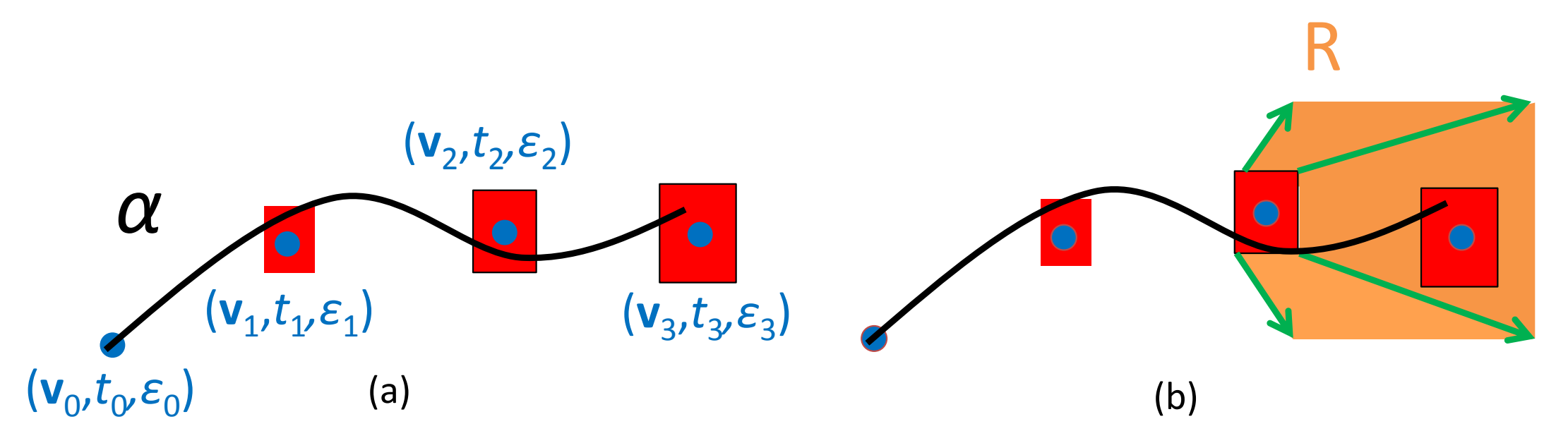

R
α
($\mathbf{v}_2$,t_2,ε_2)
($\mathbf{v}_1$,t_1,ε_1)
($\mathbf{v}_3$,t_3,ε_3)
($\mathbf{v}_0$,t_0,ε_0)
(a)
(b)

Fig. 1 Overapproximation of reach set
We compute an overapproximation of the true execution α using its sampled trace $\{(\mathbf{v}_k, t_k)\}_{k=0}^{l}$. Our algorithm evolves two steps: (a) compute the error bound ε_k of each sampled point, and (b) compute a tube segment $R \supseteq \alpha(\mathbf{v}_0, t)$ for all $t \in [t_k, t_{k+1}]$ by propagating the error ball $B(\mathbf{v}_k, \varepsilon_k)$.

2. Propagation between Sampled Points

- With a given sampled trace with error bounds we compute a tube containing all reachable states of any execution that may generate the sampled trace.
- The tube is the union of tube segments between two consecutive sample points. We execute Algorithm 1 to compute a tube segment. Illustrated in Fig. 1b.
- **Algorithm 1:**

```
1  $\sigma \leftarrow \epsilon_k$;
2  do
3      $\sigma \leftarrow b\sigma$    \\$b > 1$ is a constant;
4      $B \leftarrow Ball(\mathbf{v}_k.X, \sigma)$;
5      $m \leftarrow \sup_{X \in B} ||f(X)||$;
6  while  $\sigma - m\delta < \epsilon_k$ ;
7  $B_0 \leftarrow Ball(\mathbf{v}_k.X, \epsilon_k)$;
8  $f_{min} \leftarrow \delta \inf_{X \in B} f(X)$; $f_{max} \leftarrow \delta \sup_{X \in B} f(X)$;
9  $R \leftarrow Post(B_0, \delta, f_{min}, f_{max})$;
10 return $R$;
```

Implementation

- ***Hybrid Trace Verifier (HTV)*** implements our algorithm.
- HTV takes two inputs: (1) a hybrid automaton, and (2) a sampled trace.
- The hybrid automaton is translated from a Simulink/ Stateflow model by HyLink [2]. The sampled trace is simulated by MATLAB ODE solver.

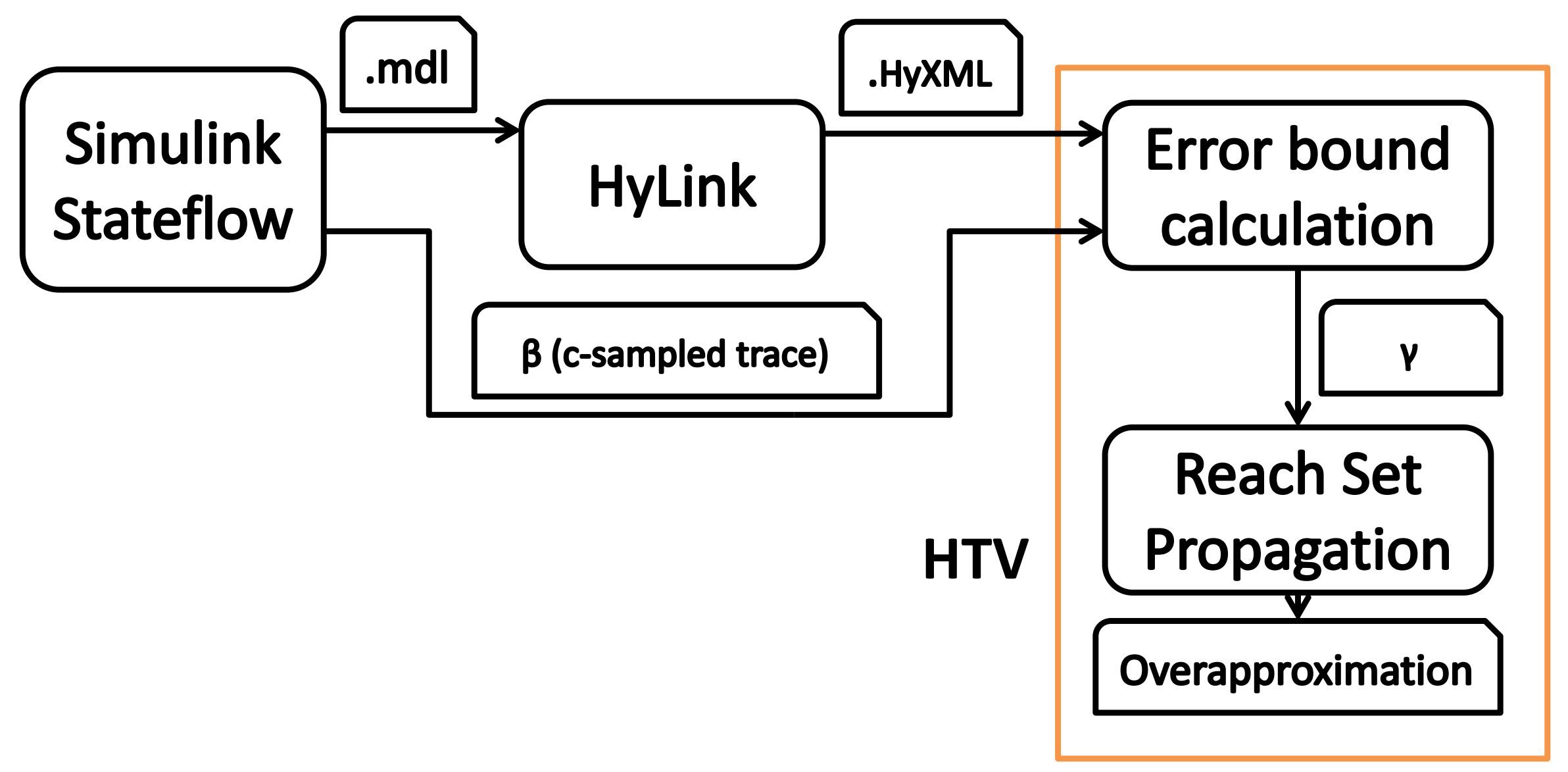


Fig. 2 Work flow of HTV
HTV is connected to MATLAB Simulink/Stateflow environment which is widely used in real-time system design. HyLink translates a Simulink/Stateflow model in to hybrid automaton. HTV takes the model as well as a sampled trace generated by MATLAB simulation engine to compute the overapproximation of reach set.

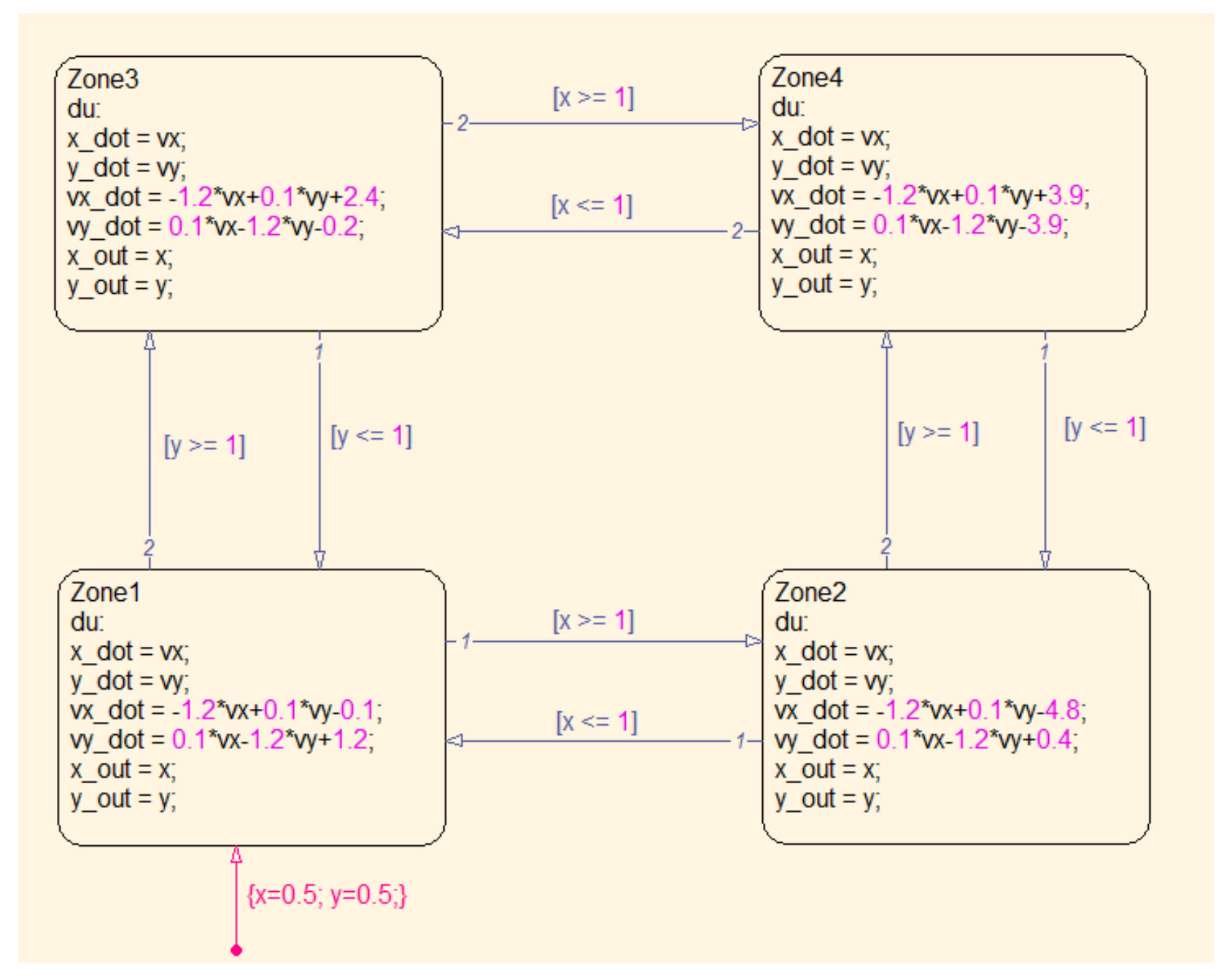


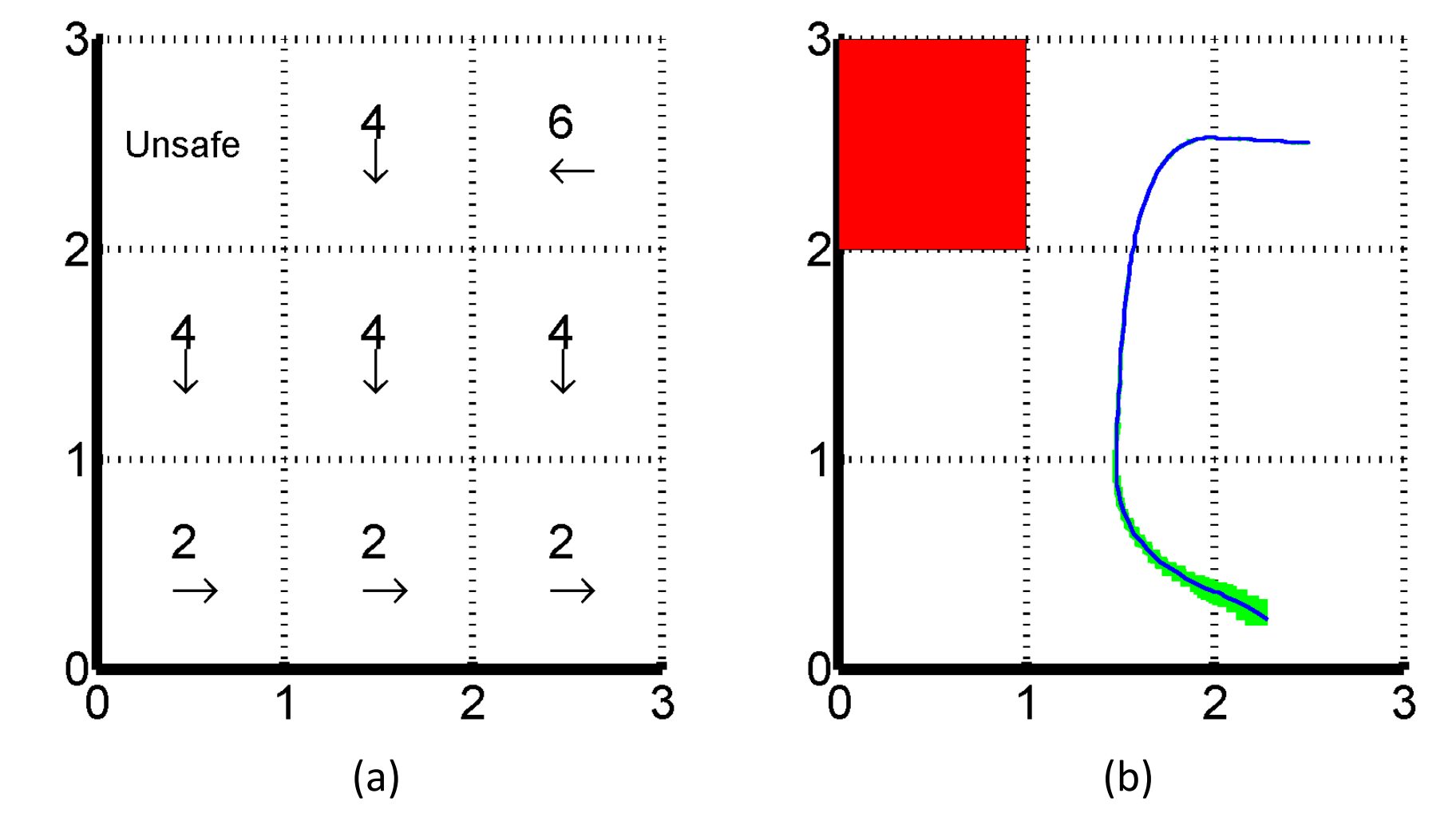
Fig. 3. A Model of Hybrid System built in Stateflow
This instance has 4 continuous variables (x,y,vx,vy) and 4 locations (Zone1-4).

Benchmarks	# vars, locs	Real time (s)	# samples	Reach time (s)	# transitions	Max error	Memory (Kb)
Navigation I	4,4	4	192	9.6	3	0.04	36
Navigation II	4,9	5	232	14	4	0.06	43
Navigation III	4,16	7	316	17	5	0.12	58
Navigation IV	8,16	7	321	36	5	0.13	110
Heating I	3,3	2	168	6.6	4	0.34	24.4
Heating II	10,10	2	168	23	4	0.67	70
Satellites	2,2	3.5	564	14	1	0.15	58
Engine ctrl	4,2	4	200	11	1	2.5	37

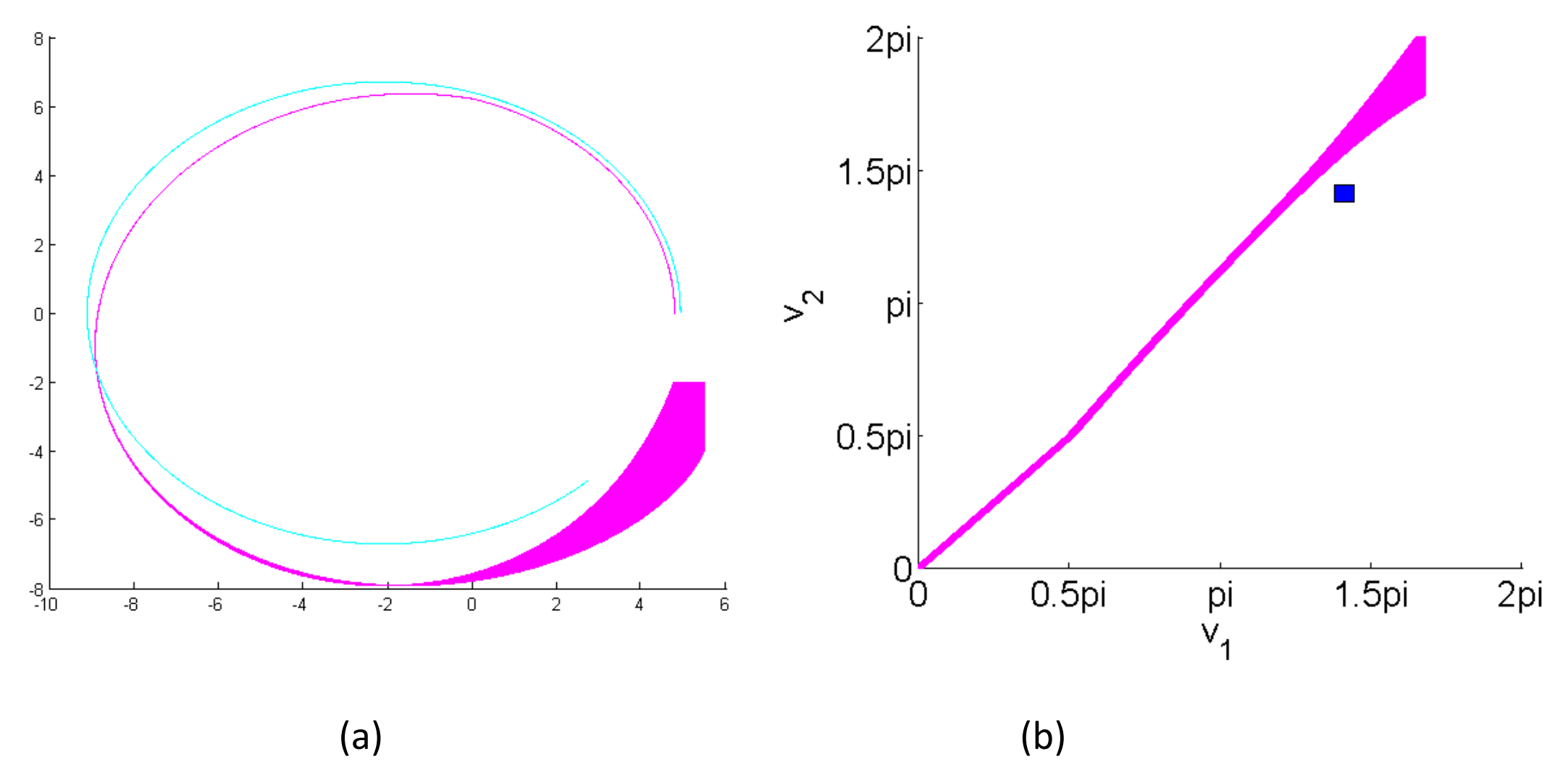
Experiments Evaluation

We applied HTV to verify safety properties of

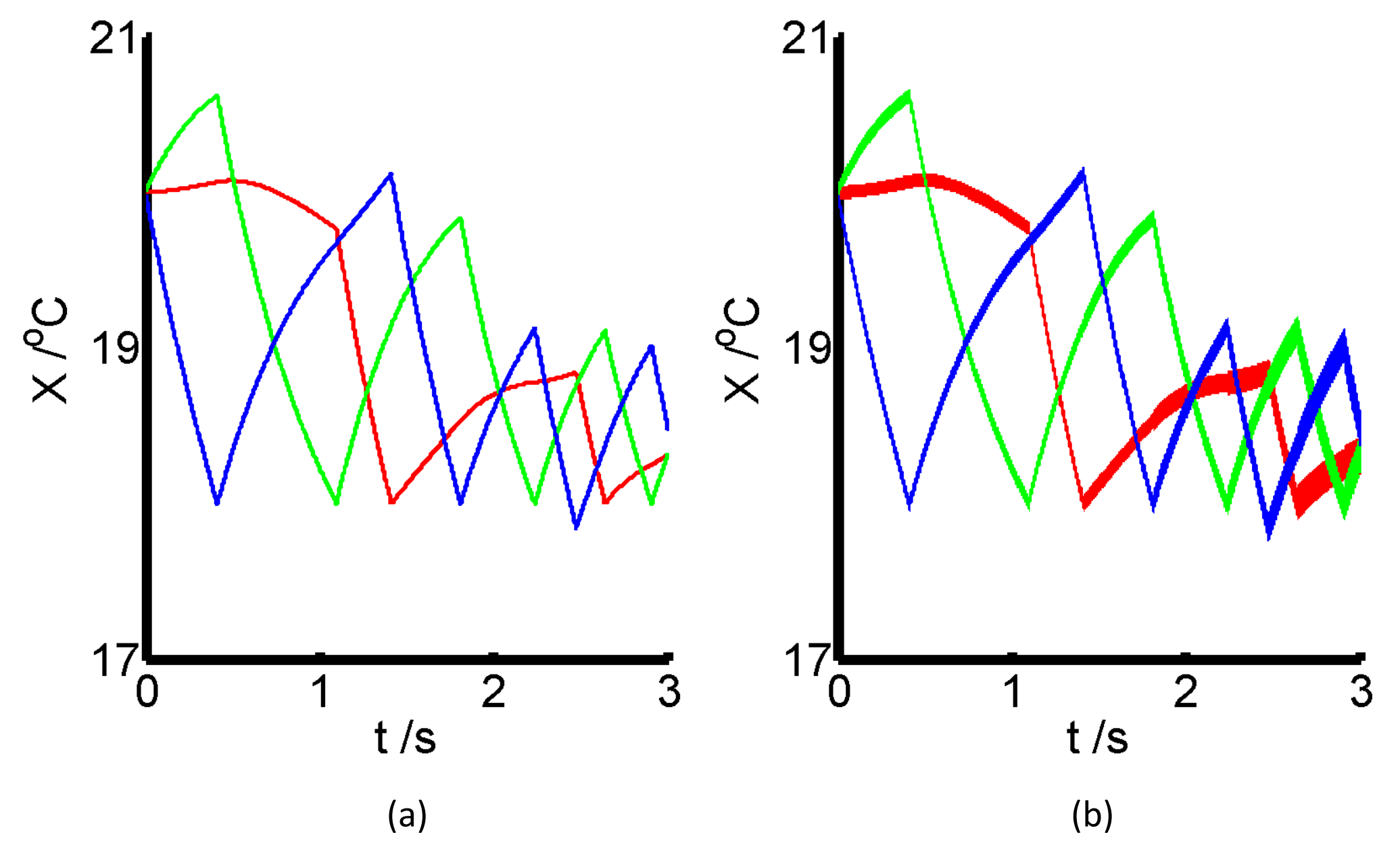
1. A navigation benchmark [3].
2. A satellite maneuver system.
3. A version of room heating benchmark.
4. An engine control system.



The navigation benchmark is a linear hybrid automaton which models positions of one or more particles in a partitioned plane. Each partition is associated with a constant external force. The safety property of interest is that the particles never reach some partition simultaneously. The figure illustrates:
(a) a partition of a plan and the external force associated to each partition;
(b) unsafe set (in red), an execution (in blue), and the overapproximation of the reach set (in green).
Safety of this instance is verified because the reach set does not intersect the unsafe set.



The satellite system models the angular position of a pair of satellites with nonlinear differential equations. The discrete states model the orbit that the satellites are following that may change through thrusting. The safety property of interest is that the satellites never come closer than some threshold value. The figure illustrates:
(a) reach set of two satellites in Cartesian coordinate;
(b) reach set in phase portrait (in purple) and the unsafe set (in blue).
Safety of this instance is verified because the reach set does not intersect the unsafe set.



A Room Heating Benchmark
The Room heating benchmark models several rooms that are heated by heaters that can be turned off and on. The continuous variables capture the temperature of all rooms. The discrete transitions capture heater control strategies. Figure (a) plots a real execution; and figure (b) visualizes the overapproaximation of reach set.
A
The safety property of interest is that the temperature of all rooms stay above a threshold, say 18C. The safety property of this instance is verified.

Conclusions and Discussion

- HTV's running time grows roughly linearly with the number of continuous variables of the system.
- The memory requirement of HTV increases linearly with the number of sample points and the dimension of the system.
- This approach may scale to larger problems.

References

[1] S. Mitra. A Verification Framework for Hybrid Systems. PhD thesis, Massachusetts Institute of Technology, Cambridge, MA 02139, September 2007.
[2] K. Manamcheri, S. Mitra, S. Bak, and M. Caccamo. A step towards verification and synthesis from simulink/stateflow models. In HSCC, 2011.
[3] A. Fehnker and F. Ivaneie. Benchmarks for hybrid systems verification. HSCC, 2993, 2004.

Acknowledgement

This research is supported by a research grant from John Deere Co. and NSF (CNS-1016791).

ILLINOIS