Learning Neural Hybrid Surrogates for Gradient-Based Falsification

Lasse Kötz and Knut Åkesson

***Abstract*— Falsification of hybrid dynamical systems remains challenging due to mode-dependent dynamics and discrete transitions. In this work, we propose a surrogate-based falsification approach enabled by learning a differentiable hybrid automaton model from data. This extends previous surrogate-based falsification methods, which were limited to modeling purely continuous dynamics. Specifically, we employ neural hybrid automata to learn both a latent mode encoder and the corresponding mode-conditioned vector fields. Subsequently, transition guards are inferred from the same trajectory data. We embed the resulting surrogate in a gradient-based optimal control formulation that minimizes a smooth robustness approximation of the safety specification to search for violations, and validate the solution on the original system to ensure soundness. The proposed method consistently uncovers counterexamples in 4 out of 6 of the ARCH-COMP 2024 benchmark specifications; in these cases, it achieves sample efficiency that is competitive with, or higher than, other state-of-the-art tools from the benchmark.**

I. Introduction

Recent developments in Cyber-Physical Systems (CPS) have increasingly deployed complex systems in safety-critical domains such as autonomous driving and medical devices, where failures can have severe consequences. These systems therefore require rigorous testing before deployment.

Hybrid systems are a class of CPS characterized by a combination of continuous dynamics and discrete modes. The continuous state evolutions are conditioned on the current active mode, while the mode itself switches discretely according to a transition set and guard functions. This additional property significantly complicates the dynamics of hybrid systems, making them piecewise-continuous.

Traditional verification methods, such as model checking and reachability analysis, are effective for verifying the correctness of small-scale systems under test (SuT). However, the underlying reachability problem is generally undecidable, making reachability analysis intractable [1].

In contrast, falsification aims to find a *counterexample* that violates a given safety specification — commonly expressed in signal temporal logic (STL). A counterexample consists of an input signal, along with the corresponding output signal. Falsification is particularly suitable when the system is treated as a black box, where internal models are either too complex for formal analysis or completely unavailable. Although falsification naturally avoids many of the issues that arise in formal verification, existing methods commonly neglect leveraging explicit dynamics information, available from SuT trajectories, in their falsification strategies.

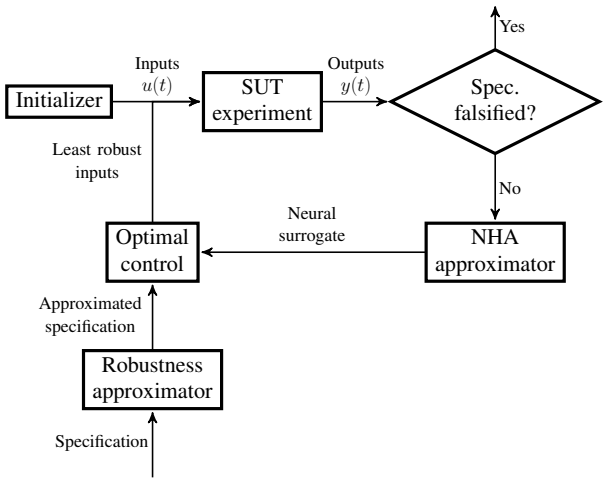


Fig. 1: New trajectories are acquired by experiments on the SuT using a least-robust input signal, resulting from an optimal control solution. The authenticity of the counterexample in the surrogate is always verified on the original SuT to ensure soundness.

In this paper, we build on the work of FalConN [2], extending it to hybrid systems using neural hybrid automata (NHA) and falsification by optimal control, highlighted in Figure 1. This is done by simultaneously training a discrete latent mode encoder network and the corresponding dynamics neural ODEs. Prior to training, trajectories are segmented into subtrajectories around switching events, using switch observers or, when unavailable, using temporal point process methods. This enables unsupervised encoder training, by minimizing reconstruction error on each segment. The trained encoder enables guard condition learning from regressing on data at transition times. Our method employs a hierarchical model selection strategy to learn guard functions. This is done by starting from a linear ansatz, only resorting to nonlinear models when the linear fit is insufficient.

The resulting surrogate NHA serves as a basis for computing candidate counterexamples, which are subsequently verified on the original SuT. To address the inherent non-smoothness of STL robustness semantics, we utilize a smooth log-sum-exponential smoothing, yielding a globally differentiable robustness objective. Doing so enables a gradient-based optimal control formulation by differentiating the learned surrogate simulator and robustness value. Although hybrid automata introduce discontinuities, the surrogate re-

This work was supported in part by VINNOVA under the VerSACE project (no. 2025-04143), as well as by MAXBATT (no. MRU2024-00381).

Lasse Kötz and Knut Åkesson are with the Department of Electrical Engineering, Chalmers University of Technology, 412 96, Göteborg, Sweden {kotz, knut}@chalmers.se

mains locally differentiable in a sequence-invariant region of the input space.

We evaluate the framework on benchmarks from the ARCH-COMP 2024 falsification competition [3] and compare against state-of-the-art tools. On the evaluated benchmarks, our method requires no more — and often fewer — SuT executions to find counterexamples on 4 out of the 6 evaluated specifications. We summarize the main contributions of this paper:

- We extend neural surrogate-based falsification to hybrid systems via neural hybrid automata.
- We show that gradient-based optimal control over the learned surrogates reduces the number of expensive SuT executions required for falsification.
- We evaluate the performance on the ARCH-COMP 2024 falsification competition benchmarks and compare against other state-of-the-art methods.

II. Related works

Recent advances in falsification include optimization-based falsification methods, treating the SuT as an opaque simulator and rely on gradient-free optimization to violate a given safety specification. Methods include, simulated annealing and line-search falsification. Widely used tools such as S-TaLiRo [4] and Breach [5] follow this paradigm, repeatedly simulating the SuT while optimizing input parameters through stochastic search. ForeSee [6] addresses the scale sensitivity of STL robustness by transforming the specification syntax tree and uses Monte Carlo tree search to explore the resulting structure. ATheNA [7] combines manually designed and learned fitness functions, enabling domain experts to bias the exploration toward critical regions.

To guide falsification more strategically and reduce execution cost, recent techniques approximate a robustness function from data. Bayesian optimization [8] learns a probabilistic mapping of the robustness as a function of the input signal parameters with an acquisition function to select new experiments. NNFal [9] trains a static depth neural network surrogate for the input-output behavior. The falsification is done using adversarial attack algorithms but is limited in its temporal operator support. Furthermore, it does not represent the dynamics as continuous time models. Online diffusion (OD) [10] uses online test generation, by denoising test vectors as parameterizations of piecewise constant input signals, based on generative adversarial networks. Moonlight [11] combines runtime monitoring with falsification and, in the ARCH-COMP 2024 competition, employs Bayesian optimization with trust regions to guide the search for violating inputs. EXAM-Net adopts an adversarial learning approach in which a discriminator network learns to predict robustness values from inputs, guiding the search toward likely violations.

Lately, more explicit formulations learn the system dynamics themselves, instead of an abstract robustness function mapping. This enables cheap simulations compared to the expensive SuT experiments. The surrogate dynamics can then be falsified, rather than the expensive SuT. Violating counterexamples are verified on the original SuT, making the algorithm sound with respect to the original system. ARIsTEO [12] estimates a set of models using system identification, and subjects them to black-box falsification. FreaK [13] is another tool which uses Koopman operator theory and dynamic mode decomposition to estimate a linear function of observables. Although this does not provide an explicit dynamics model, it yields a linear surrogate model in a lifted observable space, enabling MILP solvers to compute the least robust trajectory. Lastly, FalConN uses neural ODEs to learn the differential dynamics [2]. The framework supports grey-box scenarios in which prior knowledge about dynamics is available or assumed. The falsification step is carried out by identifying a symbolic representation and solving a nonlinear optimal control problem to synthesize falsifying trajectories, given the specification. However, it is inherently single-modal and does not learn the discrete modes nor guard functions and transitions.

Finally, some data-driven tools focus on reachability or abstraction rather than optimization-based falsification. DryVR [14] learns probabilistic discrepancy functions to construct over-approximated reachability tubes. FalCAuN [15] models the system under test as a black-box transition system and applies active automata learning to construct a Mealy machine abstraction, which is then analyzed using model checking techniques.

In conclusion, existing falsification frameworks either (i) treat the system as a black box and rely on gradient-free search, (ii) learn static input–output surrogates without modeling system dynamics, or (iii) approximate dynamics but do not expose differentiable structure for gradient-based optimization. Moreover, prior surrogate-based approaches primarily target purely continuous systems and do not address the challenges of learning and exploiting hybrid mode structure. This gap motivates the presented work.

III. Preliminaries

A. Hybrid Automata

We consider hybrid automata [16] as a tuple

$$\mathcal{H} \triangleq \langle Q, X, F, \text{Inv}, R, E, G, \text{Init} \rangle$$

with finite set of discrete modes $Q = \{q_1, \ldots, q_M\}$, $n-$dimensional continuous state space $X \subseteq \mathbb{R}^n$, transition edges $E \subseteq Q \times Q$, initial hybrid state Init $\subseteq X \times Q$, flow map $F = \{f_q : X \times U \times \mathbb{R}_{\geq 0} \to \mathbb{R}^n \mid q \in Q\}$, invariant Inv : $Q \to 2^X$ defining the region in which dynamics associated with mode q are valid, reset map $R : E \times X \to X$ and guard mapping $G : E \to 2^X$ describing the transition conditions.

The continuous state $x \in X$ evolves according to the vector field $f_q \in F$ conditioned on the discrete mode $q \in Q$ under invariant $\text{Inv}(q) \subseteq X$ and driven by a continuous input signal $u(t) \in U$

$$\dot{x} = f_q(x, u, t), \quad x(t) \in \text{Inv}(q). \tag{1}$$

This state evolution is continuous per mode and piecewise continuous globally. Each transition edge $e \in E$ is a tuple

(q, g, r, q') consisting of the modes pre and post transition q, q', along with the guard condition g and reset map r :

$$x \in G(e) \implies (q, x) \to (q', R(e, x)). \tag{2}$$

We denote a trajectory τ from $\mathcal{H}$ to be obtained by executing experiments on a hybrid system, either real-world or in simulation $\tau \leftarrow \text{Exec}_{\mathcal{H}}(x_0, q_0, u(t))$.

B. Signal Temporal Logic

Signal temporal logic (STL) is a logic formalism which extends linear temporal logic with real-time and real-valued constraints for signals. Its grammar - given a set of atomic predicates μ - is recursively defined as

$$\varphi ::= \mu \mid \neg\mu \mid \varphi \wedge \psi \mid \varphi \vee \psi \mid \Box_{[a,b]}\psi \mid \Diamond_{[a,b]}\psi \mid \varphi \mathcal{U}_{[a,b]}\psi.$$

Here, $\Box$, $\Diamond$ and $\mathcal{U}$ denote the globally, finally and until operators, respectively, and $[a, b]$ for $a, b \in \mathbb{R}_{0^+}, b \geq a$ define the closed interval time bounds.

Definition 1 (STL semantics): The STL semantics of a signal y at time t are defined as

$$\begin{aligned}
(y,t) &\vDash \mu && \iff \mu(y(t)) > 0 \\
(y,t) &\vDash \neg\mu && \iff \neg((y,t) \vDash \mu) \\
(y,t) &\vDash \varphi \wedge \psi && \iff (y,t) \vDash \varphi \wedge (y,t) \vDash \psi \\
(y,t) &\vDash \varphi \vee \psi && \iff (y,t) \vDash \varphi \vee (y,t) \vDash \psi \\
(y,t) &\vDash \Box_{[a,b]}\varphi && \iff \forall t' \in [t+a, t+b], (y,t') \vDash \varphi \\
(y,t) &\vDash \Diamond_{[a,b]}\varphi && \iff \exists t' \in [t+a, t+b], (y,t') \vDash \varphi \\
(y,t) &\vDash \varphi \mathcal{U}_{[a,b]}\psi && \iff \exists t' \in [t+a, t+b], (y,t') \vDash \psi \\
& && \quad \wedge \forall t'' \in [t, t'], (y,t'') \vDash \varphi.
\end{aligned}$$

Definition 2 (Robustness semantics): For a scalar real-valued function $\rho : \mathbb{R}^{n_{signals}} \mapsto \mathbb{R}$ on a specification φ, the robustness value is a metric of how strongly φ is satisfied or violated [17]. The max semantics are given by

$$\begin{aligned}
(y,t) \vDash \varphi &\iff \rho^{\varphi}(y,t) > 0 \\
\rho^{\neg\varphi}(y,t) &= -\rho^{\varphi}(y,t) \\
\rho^{\varphi_1 \wedge \varphi_2}(y,t) &= \min(\rho^{\varphi_1}(y,t), \rho^{\varphi_2}(y,t)) \\
\rho^{\varphi_1 \vee \varphi_2}(y,t) &= \max(\rho^{\varphi_1}(y,t), \rho^{\varphi_2}(y,t)) \\
\rho^{\Diamond_{[a,b]}\varphi}(y,t) &= \max_{t' \in [t+a,t+b]} (\rho^{\varphi}(y,t')) \\
\rho^{\Box_{[a,b]}\varphi}(y,t) &= \min_{t' \in [t+a,t+b]} (\rho^{\varphi}(y,t')) \\
\rho^{\varphi_1 \mathcal{U}_{[a,b]} \varphi_2}(y,t) &= \max_{t' \in [t+a,t+b]} (\min([\rho^{\varphi_2}(y,t'), \\
& \qquad \min_{t'' \in [t+a,t']} (\rho^{\varphi_1}(y,t''))])).
\end{aligned}$$

IV. Falsification using NHAs and Optimal Control

We expand the work in [2] – which previously assumed purely continuous dynamics – to incorporate hybrid automata by employing Neural Hybrid Automata (NHA) for learning [18]. This approach requires the user to make an assumption of the number of modes M in the system, although empirical results suggest that redundant modes collapse into one. Furthermore, approximate transition times or heuristics to determine them from trajectories are needed. In ideal cases, an observer can indicate when transitions occur, but in general we rely on change-point detection methods. Source and target mode information of transitions is not required.

The learned NHA is then subjected to a gradient-based optimal control formulation, in which robustness is smoothly approximated and thus fully differentiable with respect to input signals. If the objective value of the optimization problem is below zero, the surrogate is falsified and the counterexample is subsequently verified on the original SuT to avoid spurious results. The algorithm is not complete, but remains sound, as candidate counterexamples are always evaluated on the original SuT.

A. Neural Hybrid Automata learning

The NHA consists of a mode encoder network ε parameterized by ω, trained to classify latent mode probabilities from trajectory segments, and a piecewise-continuous dynamics model. For a segment τ_k active over the time interval $[t_k, t_{k+1}]$, the dynamics are represented as a neural ODE [19] with the possibility of incorporating a priori dynamics:

$$\dot{x} = f_{\theta_{q_k}}(x, u, t) + f_{q_k}^{\text{known}}(x, u, t), \quad t \in [t_k, t_{k+1}], \tag{3}$$

where q_k is the discrete mode for the segment. The unknown dynamics $f_{\theta_{q_k}}$ are parameterized by a universal approximator that is Lipschitz continuous in x and continuous in t, such as an MLP with tanh activations, guaranteeing existence and uniqueness of solutions by the Picard-Lindelöf theorem.

During training, we utilize a straight-through estimator, executing a hard mode selection during the forward pass, while gradients are computed using a differentiable sum of vector fields weighted by the softmax-derived mode probabilities p^k:

$$\begin{aligned}
p^k &= \text{softmax}(\varepsilon(\tau_k, \omega)), \\
\hat{q} &= \arg\max_{q \in Q} p_q^k, \\
\dot{x} &= f_{\theta_{\hat{q}}}(x, u, t) + f_{\hat{q}}^{\text{known}}(x, u, t), \\
\frac{\partial \dot{x}}{\partial \theta} &\approx \sum_{q \in Q} p_q^k \frac{\partial}{\partial \theta} \left[f_{\theta_q}(x, u, t) + f_q^{\text{known}}(x, u, t) \right],
\end{aligned} \tag{4}$$

where p_q^k is the estimated probability of segment k being attributed to latent mode q such that

$$p^k \in \Delta^Q, \quad \Delta^Q = \left\{ p \in \mathbb{R}^Q \mid p_q \geq 0, \sum_{q \in Q} p_q = 1 \right\}. \tag{5}$$

We train the encoder and neural ODEs jointly to cluster segments into latent modes. Neural ODE gradients can be computed by solving the adjoint sensitivity equation [20] backwards in time after having computed a scalar trajectory loss.

To mitigate mode collapse in segment boundaries, a phenomenon in which the encoder learns to only predict a subset of modes that exist in the available data, we add a penalty term for mode separation. The total loss function of trajectory τ becomes

$$\mathcal{L}^{\tau} = \sum_{k=1}^{K} \mathcal{L}_{\text{rec}}^{(\tau_k)} + \lambda_{\text{sep}} \sum_{k=1}^{K-1} \langle p^k, p^{k+1} \rangle, \tag{6}$$

where λ_{sep} is a regularization factor, $\langle p^k, p^{k+1} \rangle$ is the inner product of the subsequent mode probability distributions, while $\mathcal{L}_{\text{rec}}$ is the MSE reconstruction error. This additional penalty term shrinks as the encoder associates subsequent segments to different modes.

The guard functions are learned using a hierarchical model selection strategy, starting from a linear inductive structure and only resorting to nonlinearity if the linear hypothesis proves insufficient. Determining linear guards for each observed transition corresponds to solving the hyperplane $a_1 x_1 + \ldots + a_n x_n + b = 0$ that defines the state x at the respective transition. The parameters $w = (a_1, \ldots, a_n, b)$ are estimated as a solution to $w = \text{null}([X, 1])$. Determining an $n-$dimensional hyperplane requires making n linearly independent observations per unique transition.

B. Gradient-based optimal control

With a hybrid surrogate dynamical system $f_{\mathcal{H}}$ available, we formulate the falsification problem as an optimal control problem, minimizing the robustness. The robustness objective gradient with respect to the input signal is obtained by differentiating through the surrogate simulator which can be done efficiently using automatic differentiation.

The robustness semantics of STL are in general non-differentiable due to $\max$- and $\min$-operators. However, because the robustness gradient $\nabla_u \rho^{\neg\varphi}$ is required, the required operators are smoothly approximated using a log-sum exponential as introduced in [21]

$$\begin{aligned} \max([a_1, \ldots, a_m]) &\approx \frac{1}{s} \log \left(\sum_{i=1}^{m} e^{s a_i} \right) \\ \min([a_1, \ldots, a_m]) &\approx -\frac{1}{s} \log \left(\sum_{i=1}^{m} e^{-s a_i} \right), \end{aligned} \tag{7}$$

where s is a smoothing parameter. As $s \to \infty$, the operators become exact, offering asymptotic completeness of the robustness semantics.

A least-robust input signal is acquired by solving the optimal control formulation subject to the system dynamics and input constraints:

$$\begin{aligned} &\min_{u(\cdot)} \ \rho^{\varphi}(x(\cdot), u(\cdot)) \\ &\text{s.t. } \dot{x} = f_{\mathcal{H}}(x, u, t) \\ &u_{\min} \leq u(t) \leq u_{\max}, \quad \forall t \in [0, T]. \end{aligned} \tag{8}$$

This can be done in a multitude of ways, but given that the objective function is nonconvex w.r.t. inputs u, we employ a single-shooting quasi-Newton formulation to search for a falsifying trajectory using the L-BFGS algorithm. This enables approximate Hessian information, improving convergence near optima. The resulting optimal control problem induced by the smoothed STL specification is highly non-convex and therefore sensitive to initialization. We adopt the initialization strategy of [2], in which a batch of input parameters is sampled randomly and evaluated on the surrogate model. Each sample is scored using its robustness value normalized by its Euclidean distance to the training input parameters ϕ_{data}. The initialization is then selected as

$$\phi_o = \arg \min_{\phi \in \{\phi_1, \ldots, \phi_P\}} \frac{\rho^{\varphi}(\text{Exec}_{\mathcal{H}}(x_0, q_0, u_\phi(t)))}{\|\phi - \phi_{\text{data}}\|_2}. \tag{9}$$

This heuristic favors input signals that are both likely to violate the specification on the surrogate and sufficiently distinct from the training data, improving exploration of the optimization landscape.

V. Evaluation Setup

We evaluate our method on the chasing cars system in the ARCH-COMP 2024 falsification benchmark [3]. The chasing cars system [22] is set up with five cars driving along a straight road. The input signals are the throttle and brake to the leading car, which is described by a second order differential equation. The remaining four cars are described by a 3-mode hybrid automaton with the "keeping", "chasing" and "braking" modes used to keep distance while not lagging behind, as shown in Figure 2. The associated STL-specifications for the system are provided in Table I and specify both collision-free and proximity behaviors. We use the same benchmark-model as the tools in [3], which differs in some details from the original formulation in [22].[1]

TABLE I: The STL safety specifications for chasing cars constitutes a mix of collision-avoidance and proximity specifications between the positions of chasing car y_{i+1} and leading car y_i.

Spec.	STL
CC1	$\square_{[0,100]} y_5 - y_4 \leq 40$
CC2	$\square_{[0,70]} \lozenge_{[0,30]} y_5 - y_4 \geq 15$
CC3	$\square_{[0,80]} ((\square_{[0,20]} y_2 - y_1 \leq 20) \vee (\lozenge_{[0,20]} y_5 - y_4 \geq 40))$
CC4	$\square_{[0,65]} \lozenge_{[0,30]} \square_{[0,5]} y_5 - y_4 \geq 8$
CC5	$\square_{[0,72]} \lozenge_{[0,8]} ((\square_{[0,5]} y_2 - y_1 \geq 9) \rightarrow (\square_{[5,20]} y_5 - y_4 \geq 9))$
CCx	$\bigwedge_{i=1..4} \square_{[0,50]} (y_{i+1} - y_i > 7.5)$

The competition benchmarks feature two different instances; arbitrary piecewise-continuous as chosen by the participant, and piecewise-constant parameterization of the inputs. To ensure consistent and fair comparison with other tools, we evaluate our method on the fixed parameterization of the inputs, with switching interval durations of 5 seconds.

We initialize our data, consisting of a single trajectory, by uniform-random sampling. For the model structure of the follower cars, we assume NHAs with 3 latent modes, which is more than those reachable in the underlying system, whereas the leading car is parameterized by a single mode. The surrogate is not pre-trained prior to falsification. To limit training overhead and avoid converging to local optima, we run the experiments with a memory of the last five SuT trajectories, discarding prior ones. Lastly, the smoothing

[1] Evaluations on a corrected version were also made, which resulted in non-challenging instances in which all specifications are almost immediately falsified.

coefficient of the robustness semantics is set to $s = 2$, balancing smoothness and exactness.

We evaluate our method over 10 repeated runs to mitigate the influence of randomness. We set a significantly lower simulation budget (100) than the benchmarks (1500) to limit runtime.

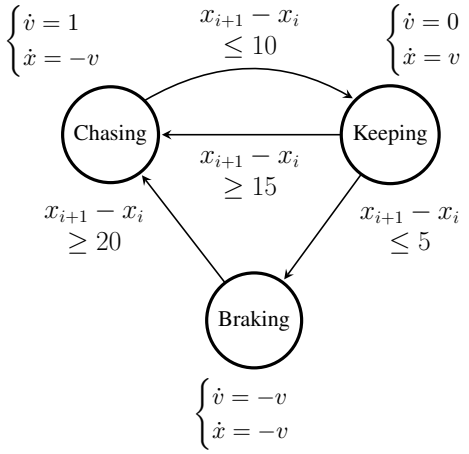


Fig. 2: Hybrid automaton for cruise policy including guard conditions as functions of relative position $x_{i+1} - x_i$. The initial state is keeping.

VI. Results

We evaluate the proposed NHA-based falsification method on the ARCH-COMP 2024 chasing cars benchmark, comparing it to all participating tools in Instance 2 (Table II). Performance is measured in terms of the number of successful falsifications (FR) and the number of SuT executions required to find a counterexample (mean $\bar{S}$ and median $\tilde{S}$).

For specifications CC_1, CC_2, CC_3, and CC_5, our method attains comparable or lower numbers of SuT evaluations than the baseline methods across repeated runs. Notably, for CC_5, our method achieves the lowest median and mean number of SuT executions among all other benchmark tools, while maintaining a high falsification rate. However, our method fails to consistently falsify CC_4 and CC_x specifications. Several factors may contribute to this limitation, including an overly restrictive simulation budget or a lack of exploration during optimal control and between falsification iterations.

Upon inspection of the learned NHA's vector fields (Figure 3), we find that they successfully prune the redundant and unreachable mode, indicating a robustness to over-approximation. Beyond interpretability, the resulting NHA provides a reusable representation of the SuT. When falsifying new specifications, it can serve as a prior for further searches, an advantage over approaches that do not explicitly model the system dynamics.

Overall, gradient-based falsification on learned hybrid surrogates successfully falsifies 4 of the 6 specifications, matching or reducing the number of required SuT evaluations relative to the benchmark tools. can reduce the number of required SuT evaluations in several benchmark instances, while remaining competitive with state-of-the-art falsification tools. The remaining two, CC_4 and CC_x, are not consistently falsified under the current budget, possibly due to the exploration limitations discussed above

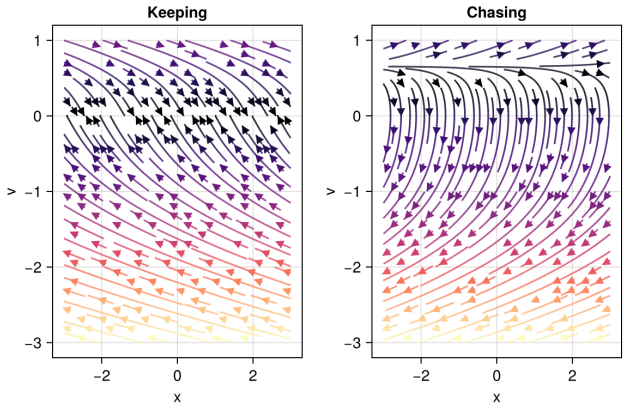


Fig. 3: The learned NHA vector fields used to successfully falsify the SuT. Note that the unreachable braking mode has been left out.

TABLE II: Performance of our approach and the participating tools in the ARCH-COMP 2024 competition, instance 2. We measure the number of successful falsifications FR, mean $\bar{S}$, and median $\tilde{S}$ number of SuT executions to find a counterexample, and evaluate our tool with a simulation budget of 100, contrary to the benchmark evaluation.

Tool	FR	$\bar{S}$	$\tilde{S}$	FR	$\bar{S}$	$\tilde{S}$
		CC_1			**CC_4**	
Our Approach	10/10	**2.9**	3.0	0/10	-	-
UR	10/10	16.4	9.5	0/10	-	-
ARIsTEO	10/10	11.3	11.5	0/10	-	-
ATheNA	10/10	60.5	30.0	2/10	514.0	514.0
EXAM-Net	10/10	36.1	36.0	0/10	-	-
FORESEE	10/10	21.8	21.5	7/10	680.6	732.0
FReaK	10/10	3.0	**2.0**	10/10	**1349.9**	**1201.0**
NNFal	9/10	1.22	1			
OD	10/10	61.2	53.0	0/10	-	-
ψ–TaLiRo	10/10	18.0	10.5	1/10	1387.3	1500.0
		CC_2			**CC_5**	
Our Approach	10/10	3.7	**3.0**	10/10	**22.8**	**17.0**
UR	10/10	12.4	13.0	10/10	37.4	22.0
ARIsTEO	10/10	10.5	10.0	10/10	29.1	22.5
ATheNA	10/10	94.4	106.5	9/10	112.0	91.0
EXAM-Net	9/10	398.6	364	10/10	78.2	106
FORESEE	8/10	224.3	19.5	10/10	88.7	25.5
FReaK	10/10	**3.3**	**3.0**	10/10	47.5	34.0
NNFal						
OD	10/10	109.3	44.0	10/10	55.7	53.0
ψ–TaLiRo	10/10	14.8	9.0	10/10	32.2	36.5
		CC_3			**CC_x**	
Our Approach	10/10	3.2	3.0	2/10	54.0	54.0
UR	10/10	19.6	21.0	6/10	396.7	284.5
ARIsTEO	10/10	18.8	12.5	9/10	610.4	465.0
ATheNA	10/10	119.2	95.5	5/10	86.4	84.0
EXAM-Net	10/10	14.9	10/10	10/10	448.1	390.5
FORESEE	10/10	36.1	11.5	10/10	**228.0**	**189.5**
FReaK	10/10	**2.6**	**2.0**	7/10	1723.6	938.0
NNFal						
OD	10/10	35.2	31.0	2/10	261.5	261.0
ψ–TaLiRo	10/10	11.4	10.5	10/10	244.2	225.0

VII. Conclusions

In this work, we have extended the FalConN surrogate-based falsification method [2] to incorporate hybrid systems by using NHAs with gradient-based optimal control. In contrast to prior work, the approach relies on learning a differentiable piecewise-continuous hybrid automaton with a vectorfield per latent mode. Within fixed mode-sequences, the dynamics remain locally differentiable, enabling gradient-based optimal control, using smoothed STL robustness.

Experimental results on the ARCH-COMP 2024 chasing cars benchmark example demonstrate that our method not only correctly discovers a reduced effective automaton structure of the SuT, but also achieves competitive or improved sample-efficiency relative to existing tools on many of the specifications.

Future work includes extending the method to more expressive trajectory optimization schemes, such as multiple-shooting methods for each branch of transitions as well as integrating hybrid gradient methods to search more efficiently for least-robust inputs. Lastly, evaluating the approach on a broader range of benchmark systems would provide further insight into its generalizability.

References

[1] T. A. Henzinger, P. W. Kopke, A. Puri, and P. Varaiya, "What's decidable about hybrid automata?" in *Proceedings of the twenty-seventh annual ACM symposium on Theory of computing*, 1995, pp. 373–382.

[2] L. Kötz, J. Sjöberg, and K. Åkesson, "Optimal control-based falsification of learnt dynamics via neural odes and symbolic regression," Saint Malo, France, May in press, accepted for publication, to appear in HSCC 2026 proceedings.

[3] T. Khandait, F. Formica, P. Arcaini, S. Chotaliya, G. Fainekos, A. Hekal, A. Kundu, E. Lew, M. Loreti, C. Menghi *et al.*, "Arch-comp 2024 category report: Falsification," in *International Workshop on Applied Verification for Continuous and Hybrid Systems*. EasyChair, 2024, pp. 122–144.

[4] Y. Annpureddy, C. Liu, G. Fainekos, and S. Sankaranarayanan, "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*. Springer, 2011, pp. 254–257.

[5] A. Donzé, "Breach, a toolbox for verification and parameter synthesis of hybrid systems," in *International Conference on Computer Aided Verification*. Springer, 2010, pp. 167–170.

[6] Z. Zhang, D. Lyu, P. Arcaini, L. Ma, I. Hasuo, and J. Zhao, "Effective hybrid system falsification using monte carlo tree search guided by QB-robustness," in *International Conference on Computer Aided Verification*. Springer, 2021, pp. 595–618.

[7] F. Formica, T. Fan, and C. Menghi, "Search-based software testing driven by automatically generated and manually defined fitness functions," *ACM Transactions on Software Engineering and Methodology*, vol. 33, no. 2, pp. 1–37, 2023.

[8] Z. Ramezani, K. Šehić, L. Nardi, and K. Åkesson, "Falsification of cyber-physical systems using bayesian optimization," *ACM Transactions on Embedded Computing Systems*, vol. 24, no. 3, pp. 1–23, 2025.

[9] A. Kundu, S. Gon, and R. Ray, "Data-driven falsification of cyber-physical systems," in *Proceedings of the 17th Innovations in Software Engineering Conference*, 2024, pp. 1–5.

[10] J. Peltomäki, F. Spencer, and I. Porres, "Wasserstein generative adversarial networks for online test generation for cyber physical systems," in *Proceedings of the 15th Workshop on Search-Based Software Testing*, 2022, pp. 1–5.

[11] L. Nenzi, E. Bartocci, L. Bortolussi, S. Silvetti, and M. Loreti, "Moonlight: a lightweight tool for monitoring spatio-temporal properties," *International Journal on Software Tools for Technology Transfer*, vol. 25, no. 4, pp. 503–517, 2023.

[12] C. Menghi, S. Nejati, L. Briand, and Y. I. Parache, "Approximation-refinement testing of compute-intensive cyber-physical models: An approach based on system identification," in *Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering*, 2020, pp. 372–384.

[13] S. Bak, S. Bogomolov, A. Hekal, N. Kochdumper, E. Lew, A. Mata, and A. Rahmati, "Falsification using reachability of surrogate koopman models," in *Proceedings of the 27th ACM International Conference on Hybrid Systems: Computation and Control*, 2024, pp. 1–13.

[14] C. Fan, B. Qi, S. Mitra, and M. Viswanathan, "Dryvr: Data-driven verification and compositional reasoning for automotive systems," in *International Conference on Computer Aided Verification*. Springer, 2017, pp. 441–461.

[15] M. Waga, "Falsification of cyber-physical systems with robustness-guided black-box checking," in *Proceedings of the 23rd International Conference on Hybrid Systems: Computation and Control*, 2020, pp. 1–13.

[16] J. Lygeros, K. H. Johansson, S. N. Simic, J. Zhang, and S. S. Sastry, "Dynamical properties of hybrid automata," *IEEE Transactions on automatic control*, vol. 48, no. 1, pp. 2–17, 2003.

[17] G. E. Fainekos and G. J. Pappas, "Robustness of temporal logic specifications for continuous-time signals," *Theoretical Computer Science*, vol. 410, no. 42, pp. 4262–4291, 2009.

[18] M. Poli, S. Massaroli, L. Scimeca, S. Chun, S. J. Oh, A. Yamashita, H. Asama, J. Park, and A. Garg, "Neural hybrid automata: Learning dynamics with multiple modes and stochastic transitions," *Advances in Neural Information Processing Systems*, vol. 34, pp. 9977–9989, 2021.

[19] T. Q. Chen, Y. Rubanova, J. Bettencourt, and D. Duvenaud, "Neural ordinary differential equations," *Advances in neural information processing systems*, vol. 31, 2018.

[20] L. S. Pontryagin, *Mathematical theory of optimal processes*. Routledge, 2018.

[21] Y. V. Pant, H. Abbas, and R. Mangharam, "Smooth operator: Control using the smooth robustness of temporal logic," in *2017 IEEE Conference on Control Technology and Applications (CCTA)*. IEEE, 2017, pp. 1235–1240.

[22] J. Hu, J. Lygeros, and S. Sastry, "Towards a theory of stochastic hybrid systems," in *International Workshop on Hybrid Systems: Computation and Control*. Springer, 2000, pp. 160–173.