Using Control Synthesis to Generate Corner Cases: A Case Study on Autonomous Driving

Glen Chou, Yunus Emre Sahin, *Student Member, IEEE*, Liren Yang, *Student Member, IEEE*, Kwesi J. Rutledge, Petter Nilsson, *Member, IEEE*, and Necmiye Ozay, *Member, IEEE*

***Abstract*—This paper employs correct-by-construction control synthesis, in particular controlled invariant set computations, for falsification. Our hypothesis is that if it is possible to compute a "large enough" controlled invariant set either for the actual system model or some simplification of the system model, interesting corner cases for other control designs can be generated by sampling initial conditions from the boundary of this controlled invariant set. Moreover, if falsifying trajectories for a given control design can be found through such sampling, then the controlled invariant set can be used as a supervisor to ensure safe operation of the control design under consideration. In addition to interesting initial conditions, which are mostly related to safety violations in transients, we use solutions from a dual game, a reachability game for the safety specification, to find falsifying inputs. We also propose optimization-based heuristics for input generation for cases when the state is outside the winning set of the dual game. To demonstrate the proposed ideas, we consider case studies from basic autonomous driving functionality, in particular, adaptive cruise control and lane keeping. We show how the proposed technique can be used to find interesting falsifying trajectories for classical control designs like proportional controllers, proportional integral controllers and model predictive controllers, as well as an open source real-world autonomous driving package.**

***Index Terms*—Formal verification, system verification, vehicle safety.**

I. Introduction

FORMAL verification, the process of algorithmically generating correctness certificates for a design, and falsification, the process of algorithmically finding trajectories and inputs that lead to a violation of specifications are important steps before a safety-critical control system can be deployed [2], [7], [19]. An alternative to these approaches, when a control design is not available but a plant model and specifications are available, is to synthesize a controller that, by construction, guarantees that the specifications are satisfied by the closed-loop system [14]. The key insight of this paper is to combine ideas from falsification and control synthesis to evaluate control designs for safety.

Consider the problem of evaluating a control design for an autonomous vehicle for safety. What would be a meaningful specification to run a falsification engine against in this case? The hard safety constraint—"do not crash!"—is easy to specify but can be trivially falsified. For instance, if a lead car, with very low speed, cuts in front of the autonomous car traveling with a relatively high speed, a crash is unavoidable. To get "interesting" corner cases, one might constrain the distance at which the lead car cuts in or the speed the lead car is traveling at when it cuts in. But can we systematically generate such constraints/assumptions? If a falsifying trajectory is found, can we say anything about existence of a controller that would be able to steer the vehicle to safety, or is safety simply an impossible task in this situation?

Motivated by these questions, in this paper we propose to use controlled invariant sets [4] to generate interesting corner cases for falsification. By an interesting corner case, we mean initial conditions from which ensuring safety is hard but not necessarily impossible. We restrict our attention to piecewise affine control systems subject to external disturbances (e.g., behavior of the other cars on the road and road profile) and safety constraints given as unions of polyhedra. We propose a scheme to sample initial conditions from the boundary of the invariant set. We also consider the problem of searching for falsifying disturbances (in addition to initial conditions). To this effect, we compute the winning set of a dual game, where control inputs are treated as disturbances and disturbances are treated as control, and where the goal is to reach the unsafe set. The dual strategy obtained by solving the dual game can be used to generate falsifying inputs when the state is within the winning set of the dual game. Greedy heuristics that aim to push the states to the dual game winning set are also proposed.

As an additional advantage, in case a control design is found unsafe using the proposed method, we can supervise this unsafe controller with the controlled invariant set in order to guarantee safety while still using the unsafe controller, which may have favorable performance related properties [13]. This supervision idea is similar to the simplex architecture [3], [20], where a performance controller is used together with a simpler

Manuscript received April 3, 2018; revised June 8, 2018; accepted July 2, 2018. Date of current version October 18, 2018. This work was supported by the Toyota Research Institute ("TRI"). TRI provided funds to assist the authors with their research but this article solely reflects the opinions and conclusions of its authors and not TRI or any other Toyota entity. *(Glen Chou, Yunus Emre Sahin, and Liren Yang contributed equally to this work.) (Corresponding author: Glen Chou.)*

G. Chou, Y. E. Sahin, L. Yang, K. J. Rutledge, and N. Ozay are with the Department of the Department of Electrical Engineering and Computer Science, University of Michigan, Ann Arbor, MI 48109 USA (e-mail: gchou@umich.edu; ysahin@umich.edu; yliren@umich.edu; krutledg@umich.edu; necmiye@umich.edu).

P. Nilsson is with the Mechanical and Civil Engineering Department, California Institute of Technology, Pasadena, CA 91125 USA (e-mail: pettni@caltech.edu).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TCAD.2018.2858464

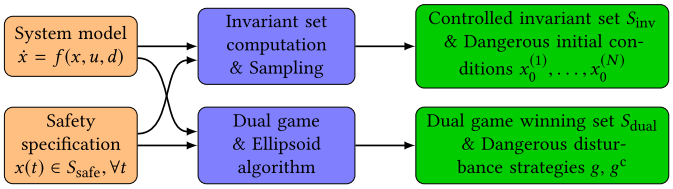


Fig. 1. Main workflow. Given a system model and a safety specification we synthesize a controlled invariant set contained inside the safe set and a winning set for the dual game. Based on these two objects we extract interesting initial conditions and disturbance strategies that are used to evaluate the safety of arbitrary (black-box) controllers.

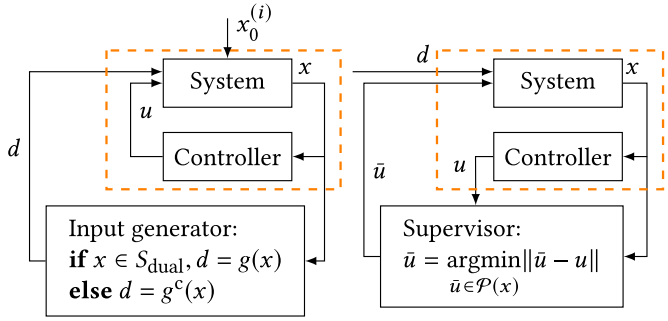


Fig. 2. In the evaluation phase, a (known) system model is controlled by a (black-box) controller. We discuss two settings, i.e., falsification (left) and supervision (right), for analyzing and enforcing safety of the closed-loop system, respectively. In falsification, the outputs of the framework in Fig. 1 are used to guide exploration of initial conditions ($x_0^{(i)}$) and disturbances (d) that lead to safety violations. As a by-product, a supervisor architecture that enforces invariance by rejecting potentially unsafe inputs (u) can be added around a controller that is found unsafe in the falsification step.

controller that has a certified safety envelope and that overwrites the performance controller only when its actions risk safety.

We demonstrate the proposed approach using two autonomous driving functions: 1) adaptive cruise control (ACC) and 2) lane keeping (LK). ACC aims to regulate the longitudinal dynamics of a vehicle either to a desired speed or a desired headway to a lead vehicle. LK controls the lateral dynamics of a vehicle to track the center line of the lane. We present safety specifications for both functions. We then apply the proposed approach to a set of controllers, including `Comma AI` software, an open source autonomous driving package, to reveal potential corner cases leading to specification violation.

II. Main Ingredients

Our goal in this paper is to search for interesting corner cases for falsification of closed-loop control systems. By interesting corner case, we mean a pair of initial condition and external (disturbance) input signal that leads to a trajectory violating a given safety specification, together with a certificate that it is possible to satisfy the specification for this initial condition and external input; therefore violation is indeed avoidable. We summarize the proposed framework in Figs. 1 and 2 before detailing the different components.

A. Controlled Invariant Sets

Invariance properties are the most basic safety properties, where the goal is to avoid an unsafe set at all times, and has been widely studied in [4]. The maximal (robust) controlled invariant set is the set of all states inside the safe set from which there exists a controller that can guarantee safety for all future times (under all possible realizations of uncertainty and disturbances).

Formally, we define a controlled invariant set for a continuous-time system using a tangent cone. Let S be a set in $\mathbb{R}^n$; a vector $y \in \mathbb{R}^n$ is called a feasible direction of set S at $x \in S$ if there exists $\varepsilon > 0$ such that $x + \delta y \in S$ for all $\delta \leq \varepsilon$. The tangent cone of a set S at x is then defined to be $T_S(x) := \text{closure}(\{y | y \text{ is feasible direction of } S \text{ at } x\})$. Consider a dynamical system described by the following differential equation:

$$\frac{d}{dt}x = f(x, u, d) \tag{1}$$

where $x \in X$ is the state, $u \in U$ is the control, and $d \in D$ is the disturbance. Here, X, U, and D represent the set of possible states, controls, and disturbances, respectively. Set $S_{\text{inv}} \subseteq X$ is called controlled invariant under the dynamics in (1) if [4]

$$\forall x \in \partial S_{\text{inv}} : \exists u \in U : \forall d \in D : f(x, u, d) \in T_{S_{\text{inv}}}(x) \tag{2}$$

where ∂S_{inv} represents the boundary of set S_{inv}. Set invariance can be defined similarly for discrete-time control systems of the form

$$x(t + 1) = F(x(t), u(t), d(t)). \tag{3}$$

A set S_{inv} is controlled invariant under dynamics in (3) if

$$\forall x \in S_{\text{inv}} : \exists u \in U : \forall d \in D : F(x, u, d) \in S_{\text{inv}}. \tag{4}$$

For simple linear system dynamics subject to additive disturbance or polytopic uncertainty, it is possible to approximate the maximal invariant set to an arbitrary precision [6], [18]. In this paper, we used polytopic invariant sets, as is done in [12], [13], and [22]. It is also possible to compute controlled invariant sets represented via barrier functions using sum-of-squares optimization [15] or approximate them via abstraction-based techniques [17]. Invariant set computation can be seen as a safety game between the control input u and the disturbance d, where the maximal controlled invariant set corresponds to the winning set (i.e., the set of all the initial states from which u can enforce safety irrespective of the values of d) in the game for the control input.

1) Supervision: The controlled invariant set can be used to supervise a legacy controller to avoid violation of the safety constraint [13], even when a controller for which a safety violation is found in the falsification step is used. The idea is to provide a recursive guarantee on safety by enforcing the trajectory to stay within a controlled invariant set S_{inv} contained by the safe set. The supervisor is a set-valued map $\mathcal{P}$ that maps the current state x_c to a set of control inputs $\mathcal{P}(x_c) \subseteq U$. Under any control $u \in \mathcal{P}(x_c)$, the next state stays within set S_{inv} under all disturbance. The supervisor overrides the legacy controller in a minimally intrusive way. That is, the supervisor is active and provides a control input in set $\mathcal{P}(x_c)$, whenever

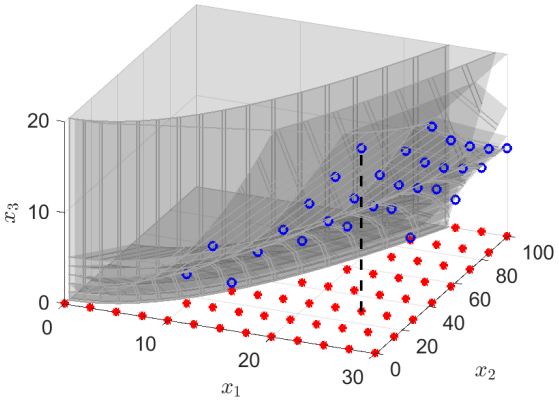


Fig. 3. Sampling the boundary of a union of polyhedra.

the legacy controller gives a control input outside $\mathcal{P}(x_c)$ at state x_c. As shown in Fig. 2, when the legacy controller's input u is in $\mathcal{P}(x_c)$, we have the supervisor output $\bar{u} = u$.

To be specific, the set $\mathcal{P}(x_c)$ can be constructed in the following way. Let $x(t+1) = F\big(x(t), u(t), d(t)\big)$ be the discrete-time dynamics. We define the set

$$\mathcal{P} := \{(x,u) \mid F(x,u,d) \in S_{\text{inv}},\ \forall d \in D\}. \tag{5}$$

Given the current state x_c, set $\mathcal{P}(x_c)$ is obtained by fixing the x component of the points in $\mathcal{P}$ to be x_c, i.e., $\mathcal{P}(x_c) := \{(x,u) \in \mathcal{P} \mid x = x_c\}$. In particular, under the assumption that S_{inv} is a polyhedron (or a union of polyhedra, resp.), F is linear in x, u, d, and D is a polyhedron, then $\mathcal{P}$ can also be represented as a polyhedron (or a union of polyhedra, resp.).

B. Sampling of the Boundary

We sample the boundary of the controlled invariant set to obtain potentially interesting initial conditions. As mentioned before, the focus in this paper is on controlled invariant sets that can be represented as a finite union of polyhedra. Fig. 3 illustrates the boundary sampling scheme of a polyhedron-union set. The gray shaded area is the union of the polyhedra. We assume the union set is contained within a hyper-rectangular domain, and sample along the first $n-1$ dimensions of the domain. These samples correspond to the red dots in the figure. Then the red dots are projected onto the boundary of the invariant set, which are marked by the blue circles in the figure. In particular, this projection can be done by the following procedure.

1) We first project each red dot y onto the boundary of each polyhedron $P = \{x \in \mathbb{R}^n \mid Ax \le b\}$ in the collection. To be specific, we fix the first $n-1$ coordinates of points x inside polyhedron P to be the same as the red dot y. This results in the 1-D polyhedron

$$P_y := \big\{x \in \mathbb{R}^n \mid Ax \le b,\ x_i = y_i, i = 1, \ldots, n-1\big\}. \tag{6}$$

   We then compute the vertex representation of the set P_y using MPT3 [9].
2) The vertices of P_y are admitted if they are not in the interior of other polyhedra.

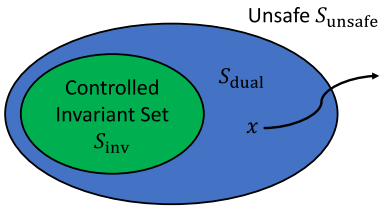


Fig. 4. Illustration: objective of the dual game.

Remark 1: The proposed sampling scheme can be extended to the case where the controlled invariant set is represented as the union of convex sets in form of $C = \{x \in \mathbb{R}^n \mid f_j(x) \le 0, j = 1, \ldots, m\}$. Similar to step 1) in the above procedure, set C_y is created as

$$C_y := \big\{x \in \mathbb{R}^n \mid f_j(x) \le 0,\ j = 1, \ldots, m \\ x_i = y_i,\ i = 1 \ldots, n-1\big\}. \tag{7}$$

Set C_y is a 1-D interval whose bounds can be computed by solving 1-D convex optimization problems $\min\{x_n \mid x \in C_y\}$ and $\min\{-x_n \mid x \in C_y\}$. Also note that a union of convex sets is not necessarily convex and may contain holes. Our proposed approach is able to sample the boundary of the holes as well.

For other types of sets, there is a brief survey in [8] on the existing approaches to sample the surface of nonconvex polyhedra. Other methods for generating (asymptotically) uniform samples on a polytope's boundary include the shake-and-bake method [5], [21], and sweep plane method [11], and these can be used as alternatives to the approach described above.

C. Computing the Falsifying Inputs

1) Dual Game: A falsifying scenario consists of two parts: 1) an initial condition and 2) a disturbance input profile. In this part, we show how to compute a falsifying input profile, through solving the so called dual game, given that the initial condition is outside the maximal invariant set. Theoretically, if the initial state is already outside the maximal invariant set, there exists a disturbance input profile that steers the trajectory outside the safe set. However, if the disturbance profile is not selected carefully, it does not necessarily lead to falsification.

We first define some terminology. Let the system dynamics be given by (1), and let S_{safe} be the safe set we want to stay inside for all time. The invariance game aims at finding the largest controlled invariant set $S_{\text{inv}} \subseteq S_{\text{safe}}$. Fig. 4 shows the objective of its dual game: we want to find set S_{dual}, and a dual strategy $g : S_{\text{dual}} \to D$, under which the states starting from S_{dual} are steered into unsafe set $S_{\text{unsafe}} := (S_{\text{safe}})^C$ in finite time, as long as $u \in U$.

We solve the dual game by computing the backwards reachable set of unsafe set S_{unsafe}. For linear discrete-time dynamics, assuming that unsafe set S_{unsafe} is a polytope, the backwards reachable set can be computed as a collection of polytopes using the same approach in [13]. The only difference is that we are now "controlling" the disturbance d and trying to be robust to the real control action $u \in U$. To be specific, let the dynamics be

$$x(t+1) = Ax(t) + Bu(t) + Ed(t) + K \tag{8}$$

where $x \in X$, $u \in U$, $d \in D$ are polytopes. We first compute a sequence of polytopes, starting with $P_0 = S_{\text{unsafe}}$, as follows:

$$P_{i+1} = \{(x, d) \in X \times D \mid \forall u \in U : Ax + Bu + Ed + K \in P_i\}. \quad (9)$$

We then project each polytope P_i onto X space to obtain $\{\overline{P}_i\}$, and the winning set of the dual game is given by $\bigcup_i \overline{P}_i$. To determine the dual game strategy g at the current state x, we locate x in one of the projected polytopes $\overline{P}_i$, and the dual strategy can be generated by picking $g(x)$ such that $(x, g(x)) \in P_i$.

When S_{unsafe} is nonconvex but can be expressed as a union of polytopes, we compute the backwards reachable set for each polytope and take the union of the obtained backwards reachable sets. This gives a conservative, yet sound, winning set for the dual game. Note that, when the invariant set or the winning set for the dual game is computed via such a conservative approach, there will be a gap between S_{inv} and S_{dual} in Fig. 4, corresponding to a set of initial conditions for which concluding whether they are interesting or not is not possible with the computed sets.

2) Ellipsoid Method: Note that the dual strategy g is defined on S_{dual} and is not applicable everywhere on S_{safe}. Thus, we need a *complementary strategy* g^c to generate falsifying inputs for states $x \in S_{\text{safe}} \setminus S_{\text{dual}}$. Next, we propose some heuristics for computing a complementary strategy. Assume that the safe set is given as a union of polyhedra $S_{\text{safe}} = \cup_i S_{\text{safe},i}$ and C_{safe} denotes the convex-hull of S_{safe}. It is shown in [10] that, for any compact set C, there exists a unique minimum volume ellipsoid (called LJ-ellipsoid of C) covering it. Denote the LJ-ellipsoid of C_{safe} with E_{safe}, which is defined as

$$E_{\text{safe}} = \left\{ x \in X \mid \left[x^\top, 1\right] Q \left[x^\top, 1\right]^\top \leq 1, Q > 0 \right\} \quad (10)$$

where the positive definite matrix Q parametrizing the ellipsoid can be computed using [16]. Define the *level* of $x \in X$ as

$$l(E_{\text{safe}}, x) = \left[x^\top, 1\right] Q \left[x^\top, 1\right]^\top. \quad (11)$$

It is reasonable to assume that points lying on the higher levels are closer to the unsafe set; hence, driving the system to higher levels would force it either to the unsafe set S_{unsafe} or to the winning set of the dual game S_{dual}. With this intuition, complementary strategy $g^c : S_{\text{safe}} \setminus S_{\text{dual}} \to D$ is defined such that it steers the system to the highest possible level set at each step

$$g^c(x) \doteq \underset{d}{\text{argmax}} \{ l(E_{\text{safe}}, x') | x' = Ax + Bu + Ed + K \} \quad (12)$$

where we assume that control input u is known.[1]

[1]When the invariant set is unbounded, the LJ-ellipsoid does not exist. In this case g^c can be computed by computing inputs that steer the state closer to S_{dual} by directly minimizing the distance to S_{dual} though the corresponding optimization problem can be more complex. Alternatively, if the rays corresponding to unbounded directions are known, an ellipsoid that is significantly elongated along those directions can be chosen by bounding those rays at a large enough level.

TABLE I
PARAMETER VALUES FOR THE ACC MODEL

Param.	Description	Value
m	car+cargo mass	1462 (kg)
f_0	friction/drag term	51 (N)
f_1	friction/drag term	1.2567 (Ns/m)
f_2	friction/drag term	0.4342 (Ns^2/m^2)
$v^{\min}, v_L^{\min}$	minimal car velocity	0 (m/s)
$v^{\max}, v_L^{\max}$	maximal car velocity	25 (m/s)
v^{des}	desired car velocity	20 (m/s)
$F_{w,c}^{\min}$	minimal force, comfort	-4305.9 (N)
$F_{w,c}^{\max}$	maximal force, comfort	2870.6 (N)
$F_{w,p}^{\min}$	minimal force, physical	-11482.5 (N)
$F_{w,p}^{\min}$	minimal force, physical	7176.6 (N)
$a_L^{\min}$	minimal acceleration	-0.97 (m/s^2)
$a_L^{\max}$	maximal acceleration	0.65 (m/s^2)
$\omega^{\min}$	minimal time headway	1.7 (s)
$h^{\min}$	minimal headway	4 (m)

Falsifying inputs are computed using g if the current state $x \in S_{\text{dual}}$, and g^c is used otherwise (see Fig. 2). Additionally, we develop some simple input-generation heuristics tailored for the ACC and LK functions of autonomous driving. These tailored heuristics will be presented on the fly in Section IV.

III. SYSTEM MODEL AND SPECIFICATIONS

For the case studies we consider two autonomous driving subsystems: 1) ACC and 2) LK. ACC controls the speed of the vehicle to follow a desired speed if there is no car in front, and to follow the lead vehicle within some safe following distance (headway) if there is a relatively slower lead vehicle in front. An LK controller controls the steering of the vehicle to avoid lane departures. Therefore, ACC controls the longitudinal dynamics and LK control deals with the lateral dynamics. In the rest of this section, we provide dynamical models used in our examples and formalize safety specifications for both systems.

A. Longitudinal and Lateral Dynamics

We use the following model from [13] to describe the longitudinal dynamics of the vehicle:

$$\frac{\mathrm{d}}{\mathrm{d}t} \begin{bmatrix} v \\ h \\ v_L \end{bmatrix} = \begin{bmatrix} \frac{1}{m}(F_w - f_0 - f_1 v - f_2 v^2) \\ v_L - v \\ a_L \end{bmatrix}. \quad (13)$$

The system states consist of the following car velocity v, lead car velocity v_L, and the headway h (i.e., the relative distance between the lead and following car). Control input F_w represents the net force acting on the mass of the following car. The lead car acceleration a_L can be viewed as a disturbance to the system. Finally, constants m, f_0, f_1, and f_2 are parameters of the model. The values of these parameters and the bounds of the variables can be found in Table I. In particular, the domain the dynamics are defined on is $X_{\text{ACC}} \coloneqq [v^{\min}, v^{\max}] \times [h^{\min}, \infty) \times [v_L^{\min}, v_L^{\max}]$.

TABLE II
PARAMETER VALUES FOR THE LK MODEL

Param.	Description	Value
v_N	nominal velocity	20 (m/s)
m	car+cargo mass	1462 (kg)
I_z	car moment of inertia	2500 (kgm^2)
a	vehicle geometry parameter	1.08 (m)
b	vehicle geometry parameter	1.62 (m)
$C_{\alpha f}$	tire parameter	85400 (N/rad)
$C_{\alpha r}$	tire parameter	90000 (N/rad)
$y^{\max}$	maximum lateral deviation	0.9 (m)
$\nu^{\max}$	maximum lateral velocity	1 (m/s)
$\Delta\Psi^{\max}$	maximum yaw-angle deviation	0.15 (rad)
$r^{\max}$	maximum yaw rate	0.27 (rad/s)
$\theta_s^{\min}$	minimum steering angle	-0.26 (rad)
$\theta_s^{\max}$	maximum steering angle	0.26 (rad)

The lateral dynamics are described by

$$\underbrace{\frac{\mathrm{d}}{\mathrm{d}t}\begin{bmatrix} y \\ \nu \\ \Delta\Psi \\ r \end{bmatrix}}_{x_{\mathrm{LK}}} = \underbrace{\begin{bmatrix} 0 & 1 & v_N & 0 \\ 0 & -\frac{C_{\alpha f}+C_{\alpha r}}{mv_N} & 0 & \frac{bC_{\alpha r}-aC_{\alpha f}}{mv_N} - v_N \\ 0 & 0 & 0 & 1 \\ 0 & \frac{bC_{\alpha r}-aC_{\alpha f}}{I_z v_N} & 0 & -\frac{a^2C_{\alpha f}+b^2C_{\alpha r}}{I_z v_N} \end{bmatrix}}_{A_{\mathrm{LK}}} \times \begin{bmatrix} y \\ \nu \\ \Delta\Psi \\ r \end{bmatrix} + \underbrace{\begin{bmatrix} 0 \\ \frac{C_{\alpha f}}{m} \\ 0 \\ a\frac{C_{\alpha f}}{I_z} \end{bmatrix}}_{B_{\mathrm{LK}}} \delta_f + \begin{bmatrix} 0 \\ 0 \\ -1 \\ 0 \end{bmatrix} r_d \tag{14}$$

where the states are: lateral deviation from the center of the lane (y), the lateral velocity (ν), the yaw-angle deviation in road-fixed coordinates ($\Delta\Psi$), and the yaw rate (r), respectively. The input δ_f is the steering angle of the front wheels, which is limited to lie within $\theta_s^{\min}$ and $\theta_s^{\max}$; and r_d is the desired yaw rate, which we interpret as a time-varying external disturbance and computed from road curvature by $r_d = v/R_0$ where R_0 is the (signed) radius of the road curvature and v is the vehicle's longitudinal velocity. Other parameters include m, the total mass of the vehicle, and a, b, $C_{\alpha f}$, and $C_{\alpha r}$, which are vehicle geometry and tire parameters. All values can be found in Table II. Accordingly, the domain the dynamics are defined on is $X_{\mathrm{LK}} \coloneqq [-y^{\max}, y^{\max}] \times [-\nu^{\max}, \nu^{\max}] \times [-\Delta\Psi^{\max}, \Delta\Psi^{\max}] \times [-r^{\max}, r^{\max}]$.

B. Formal Specifications for ACC and LK

For ACC, we focus on the safety aspect of requirement in this paper. The (safety part of) ISO Standard requirements for ACC systems [1] state:

1) The control input should stay within specified bounds all the times.
2) Whenever the lead car is close in the sense that the headway $h < v^{\mathrm{des}}\omega^{\mathrm{des}}$, the time headway ω needs to satisfy $\omega \geq \omega^{\min}$ at all times.

We extract the safety part of the above ISO requirement and express it formally in logic. Define sets

$$\begin{aligned} M &\coloneqq \left\{(v, h, v_L) \mid v^{\mathrm{des}} > h/\omega^{\mathrm{des}}\right\} \\ S &\coloneqq \left\{(v, h, v_L) \mid v \leq h/\omega^{\min}, h \geq h^{\min}\right\} \\ S_U &\coloneqq \left\{F_w \mid F_{w,c}^{\min} \leq F_w \leq F_{w,c}^{\max}\right\}. \end{aligned} \tag{15}$$

Set M is the set of states where the lead car is close, set S is the safe set of states, and set S_U contains the allowable control inputs. Adding the speed limits encoded by the domain X_{ACC}, the overall specification can be expressed as

$$\begin{aligned} (\forall t : F_w(t) \in S_U) \wedge (\forall t : ((v(t), h(t), v_L(t)) \in M) \\ \rightarrow ((v(t), h(t), v_L(t)) \in S \cap X_{\mathrm{ACC}})). \end{aligned} \tag{16}$$

To check safety in the presence of a close enough lead car, we assume the states are in M and consider the following safety specification, denoted by φ_{ACC}:

$$(\forall t : F_w(t) \in S_U) \wedge (\forall t : ((v(t), h(t), v_L(t)) \in S \cap X_{\mathrm{ACC}})). \tag{17}$$

In the later falsification experiments, we will consider violations of different aspects of the specification φ_{ACC}, that is,

$$\begin{aligned} \varphi_{\mathrm{ACC}}^1 &\coloneqq \forall t : v(t) \leq h/\omega^{\min} \\ \varphi_{\mathrm{ACC}}^2 &\coloneqq \forall t : h(t) \geq h^{\min} \\ \varphi_{\mathrm{ACC}}^3 &\coloneqq \forall t : h(t) \geq 0. \end{aligned} \tag{18}$$

These three safety specifications correspond to small time headway, small distance headway, and crash, respectively. Note that specification φ_{ACC}^2 implies φ_{ACC}^3 as $h^{\min} > 0$. Here, we distinguish specification φ_{ACC}^3 from φ_{ACC}^2 because violating φ_{ACC}^3 is considered to be more severe.

For LK, as mandated by the width of roads in the United States (approx. 3.8 m) and typical car widths (approx. 2 m), the specification states that the car must stay within $y^{\max}$ meters of the center of the lane, i.e., $|y(t)| \leq y^{\max}$. We also require the other states to remain in the domain X_{LK} as larger values of these states are either physically less meaningful (e.g., can correspond to the vehicle navigating in the reverse direction) or violate passenger comfort requirements. Moreover, the lateral dynamics model we use is valid for relatively smaller ranges of yaw rate, yaw angle, and lateral velocity. With these requirements, the overall specification for LK, denoted by φ_{LK}, is formally stated as

$$\forall t : (y(t), \nu(t), \Delta\Psi(t), r(t)) \in X_{\mathrm{LK}}. \tag{19}$$

Note that state y being out of bound should be considered to be a significant safety violation, while the other three states in x_{LK} being out of bounds leads to a less comfortable ride. Therefore, we will independently count the violations of the specification below in the falsification experiments

$$\varphi_{\mathrm{LK}}^1 \coloneqq \forall t : |y(t)| \leq y^{\max}. \tag{20}$$

TABLE III
CONTROLLERS USED IN OUR TESTS

	Controller (parameters)	Notation	Parameter
ACC	Proportional controller (P gain)	$\mathrm{P}_{\mathrm{ACC}}$ #1	$k_P = 600$
		$\mathrm{P}_{\mathrm{ACC}}$ #2	$k_P = 1800$
		$\mathrm{P}_{\mathrm{ACC}}$ #3	$k_P = 4000$
	PI controller (P/I gains)	$\mathrm{PI}_{\mathrm{ACC}}$ #1	$k_P = 600,\ k_I = 200$
		$\mathrm{PI}_{\mathrm{ACC}}$ #2	$k_P = 1800,\ k_I = 400$
		$\mathrm{PI}_{\mathrm{ACC}}$ #3	$k_P = 4000,\ k_I = 2000$
	MPC (horizon)	$\mathrm{MPC}_{\mathrm{ACC}}$ #1	2
		$\mathrm{MPC}_{\mathrm{ACC}}$ #2	8
		$\mathrm{MPC}_{\mathrm{ACC}}$ #3	20
LK	State feedback (poles)	P_{LK} #1	[-0.93; 0.92; 0.9; 0.8]
		P_{LK} #2	[-0.6±0.1i; 0.65±0.2i]
		P_{LK} #3	[0.003; 0.66±0.34i; 0.4]
	State feedback w/ integral action (poles)	$\mathrm{PI}_{\mathrm{LK}}$ #1	[-0.93; 0.92; 0.9; 0.8; 0.7]
		$\mathrm{PI}_{\mathrm{LK}}$ #2	[-0.6±0.1i; 0.65±0.2i; 0.7]
		$\mathrm{PI}_{\mathrm{LK}}$ #3	[0.002; 0.6±0.4i; 0.4; 0.7]
	MPC (horizon)	$\mathrm{MPC}_{\mathrm{LK}}$ #1	2
		$\mathrm{MPC}_{\mathrm{LK}}$ #2	5
		$\mathrm{MPC}_{\mathrm{LK}}$ #3	20

IV. CASE STUDIES

In this section, we evaluate the proposed approach on case studies with different controllers for ACC and LK. Details about the evaluated controllers can be found in Table III.

In what follows, both the wheel force F_w and the steering angle δ_f are bounded quantities. Thus, for controllers that cannot handle such input constraints, we use a saturation function before feeding their output to the system. The saturation function sat is defined as follows:

$$\mathrm{sat}_{\underline{x}}^{\overline{x}}(x) = \begin{cases} \underline{x} & \text{if } x \le \underline{x} \\ x & \text{if } \underline{x} < x < \overline{x} \\ \overline{x} & \text{if } x \ge \overline{x}. \end{cases} \tag{21}$$

In addition to sampling at the boundary of the invariant set, we also sample in the interior of the invariant set to generate less "tricky" initial conditions. These interior points are obtained by shifting (for ACC) or scaling (for LK) the boundary samples.

A. Adaptive Cruise Control Results

We computed a controlled invariant set S_{ACC} for the longitudinal dynamics in (13), and sampled the boundary of this set with the proposed approach to find falsifying initial conditions. The disturbance profile is computed by: 1) solving the dual game; 2) a simple heuristic that corresponds to the lead car doing a maximum braking; or 3) the lead car trying to achieve v^{des}. We explored the following three classes of controllers for meeting the ACC requirement.

1) For the first controller, we performed feedback linearization followed by pole placement with a hybrid proportional (P) controller, defined as

$$u = f_0 + f_2 v^2 - k_P\Big(v - \min\Big(v^{\mathrm{des}}, h/\omega^{\mathrm{des}}\Big)\Big) \tag{22}$$

where k_P is the proportional gain, the min part takes care of the two different ACC modes and $F_w = \mathrm{sat}_{F_{w,c}^{\min}}^{F_{w,c}^{\max}}(u)$ given the input saturations.

2) We also consider hybrid proportional-integral (PI) controllers with the following dynamics:

$$u(t) = f_0 + f_2 v^2 - k_P\Big(v(t) - \min\Big(v^{\mathrm{des}}, h(t)/\omega^{\mathrm{des}}\Big)\Big) - k_I e(t), \tag{23}$$

$$e(t) = \sum_{\tau=0}^{t}\Big(v(\tau) - \min\Big(v^{\mathrm{des}}, h(\tau)/\omega^{\mathrm{des}}\Big)\Big) \tag{24}$$

where $e(t)$ is the error state and k_I is the integral coefficient. Similarly, the control input u needs to be saturated to obtain practical F_w.

3) We also designed an MPC controller with a linearized discrete-time model with a sampling period of 0.1 s using the following formulation:

$$\begin{aligned} \min\ & \sum_{t=0}^{T} \|v(t) - \min(v^{\mathrm{des}}, h(t)/\omega^{\mathrm{des}})\| \\ \text{s.t. } & \text{Linearized, time-discretized dynamics of ACC} \\ & F_{w,c}^{\min} \le F_w(t) \le F_{w,c}^{\max}, \quad t = 0, \ldots, T-1 \\ & v_L^{\min} \le v_L(t) \le v_L^{\max}, \quad t = 0, \ldots, T \\ & v^{\min} \le v(t) \le v^{\max}, \quad t = 0, \ldots, T \\ & 0 \le h(t), \quad t = 0, \ldots, T \\ & v(0) = v_0, h(t) = h_0, v_L(t) = v_{L,0} \end{aligned} \tag{25}$$

where v_0, h_0, $v_{L,0}$ are the initial conditions and T is the length of the prediction horizon. Since the objective contains term $\min(v^{\mathrm{des}}, h(t)/\omega^{\mathrm{des}})$, the MPC is hybrid in its nature. To simplify the computation load, we replace the target velocity throughout the predicting horizon by $\min(v^{\mathrm{des}}, h(0)/\omega^{\mathrm{des}})$, so that the MPC problem can be solved by a QP solver.

Tables IV–VII summarize the falsification rates (FRs) for the samples both from the interior and the boundary of the controlled invariant set S_{ACC}, with disturbance a_L profile generated by multiple methods. It should be noted that the same controlled invariant set S_{ACC} is used to generate initial states to investigate the three different aspects of the safety specification in (18). This is because set S_{ACC} is synthesized against the overall safety specification in (17). Overall, the MPC controller seems better than the naïvely designed P controller in terms of safety.

Another key observation is that the falsification rates (FRs) of the test cases with interior initial conditions can be higher or lower than those on the boundary of set S_{ACC}, depending on how the disturbance (i.e., a_L) profile is generated. In particular, the test cases with interior initial conditions have higher FR than the boundary cases in Tables IV–VI, and usually have lower FR in Table VII. The key difference between Tables IV–VI and Table VII is that the leading car is usually decelerating (or maintaining constant speed) in the test cases from Table IV–VI, while it is accelerating under the test cases from Table VII. In what follows we briefly discuss how this difference affects the FRs by the interior initial conditions and by the boundary ones.

TABLE IV
ACC FRs, WITH NO INPUT GENERATION, FOR SPECIFICATIONS IN (17) AND (18)

Sampling location	Interior				Boundary			
Specification / Controller	φ^1_{ACC}	φ^2_{ACC}	φ^3_{ACC}	φ_{ACC}	φ^1_{ACC}	φ^2_{ACC}	φ^3_{ACC}	φ_{ACC}
P_{ACC} #1	0.41	0.20	0.15	0.41	0.42	0.19	0.14	0.43
P_{ACC} #2	0.25	0.14	0.09	0.25	0.26	0.13	0.09	0.26
P_{ACC} #3	0.15	0.11	0.08	0.17	0.14	0.11	0.07	0.16
PI_{ACC} #1	0.60	0.25	0.19	0.60	0.63	0.21	0.15	0.63
PI_{ACC} #2	0.49	0.22	0.16	0.49	0.41	0.17	0.11	0.42
PI_{ACC} #3	0.42	0.21	0.14	0.42	0.30	0.14	0.10	0.31
MPC_{ACC} #1	0.17	0.13	0.09	0.19	0.26	0.13	0.09	0.28
MPC_{ACC} #2	0.15	0.09	0.08	0.15	0.14	0.09	0.07	0.14
MPC_{ACC} #3	0.15	0.09	0.08	0.15	0.14	0.09	0.07	0.14

TABLE V
ACC FRs, WITH DUAL GAME, FOR SPECIFICATIONS IN (17) AND (18)

Sampling location	Interior				Boundary			
Specification / Controller	φ^1_{ACC}	φ^2_{ACC}	φ^3_{ACC}	φ_{ACC}	φ^1_{ACC}	φ^2_{ACC}	φ^3_{ACC}	φ_{ACC}
P_{ACC} #1	1.00	0.88	0.62	1.00	1.00	0.90	0.64	1.00
P_{ACC} #2	1.00	1.00	0.97	1.00	1.00	0.99	0.98	1.00
P_{ACC} #3	0.21	1.00	0.09	1.00	0.16	1.00	0.08	1.00
PI_{ACC} #1	0.95	0.98	0.50	1.00	0.95	0.97	0.55	1.00
PI_{ACC} #2	0.66	0.99	0.30	1.00	0.54	0.99	0.23	1.00
PI_{ACC} #3	0.55	0.99	0.30	1.00	0.39	0.99	0.21	1.00
MPC_{ACC} #1	0.32	0.99	0.10	1.00	0.39	0.99	0.09	1.00
MPC_{ACC} #2	0.20	0.12	0.09	0.20	0.15	0.09	0.08	0.15
MPC_{ACC} #3	0.20	0.12	0.09	0.20	0.15	0.09	0.08	0.15

TABLE VI
ACC FRs, WITH MAX BRAKING, FOR SPECIFICATIONS IN (17) AND (18)

Sampling location	Interior				Boundary			
Specification / Controller	φ^1_{ACC}	φ^2_{ACC}	φ^3_{ACC}	φ_{ACC}	φ^1_{ACC}	φ^2_{ACC}	φ^3_{ACC}	φ_{ACC}
P_{ACC} #1	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
P_{ACC} #2	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
P_{ACC} #3	0.25	1.00	0.15	1.00	0.20	1.00	0.13	1.00
PI_{ACC} #1	0.95	1.00	0.68	1.00	0.95	1.00	0.72	1.00
PI_{ACC} #2	0.66	1.00	0.43	1.00	0.57	1.00	0.32	1.00
PI_{ACC} #3	0.58	1.00	0.40	1.00	0.42	1.00	0.29	1.00
MPC_{ACC} #1	0.32	1.00	0.22	1.00	0.39	1.00	0.17	1.00
MPC_{ACC} #2	0.25	0.19	0.15	0.25	0.23	0.15	0.13	0.23
MPC_{ACC} #3	0.25	0.19	0.14	0.25	0.28	0.15	0.13	0.29

When the lead car is decelerating, the dynamics tend to have a "steady state" outside the controlled invariant set S_{ACC}. This is true because the lead car's deceleration shortens the headway h and hence pushes the state toward the boundary of S_{ACC}. In this case, the falsifications are due to the long term behavior of the dynamics as a trajectory may eventually leave S_{ACC} Since such undesired behaviors occur in a longer term, starting from the interior of set S_{ACC} may not prevent ultimate falsification.

Moreover, the trajectories initiating from the interior tend to move to the trickier parts of the boundary, where safe actions are limited, which increases the FR. We next explain why this is so. First note that a point in the interior of S_{ACC} usually has larger relative headway h and larger lead car velocity v. Consequently, the target velocity defined by $\min(v^{\text{des}}, h/\omega^{\text{des}})$ has a higher chance to be equal to v^{des} when starting from the interior. The controller hence accelerates to achieve v^{des} and maintains the velocity there. Now since the lead car velocity v_L is low (due to deceleration or small initial value), such acceleration will eventually lead to small headway h, which will change the target velocity from v^{des} to h/ω^{des}. At that moment, however, the following car velocity may be already relatively high. This hence leads to a harder scenario and increases the chance of falsification, which explains the result in Tables IV–VI.

On the contrary, when lead car's steady state speed is v^{des}, i.e., it is mostly accelerating, the dynamics tend to have a steady state inside the set S_{ACC}. This is true because the lead car's acceleration enlarges headway h and pushes the state toward inside of S_{ACC}. In this case, the falsifications are mainly due to the transient state of the dynamics because the state will eventually converge to that steady state inside S_{ACC}. By our conjecture, the initial conditions on the boundary of S_{ACC} have higher chances for capturing the falsifications due to transient state. This explains why the FR in Table VII agrees with our conjecture.

TABLE VII
ACC FRs, WITH LEAD CAR CONVERGING TO v^{DES}, FOR SPECIFICATIONS IN (17) AND (18)

Sampling location	Interior				Boundary			
Controller \ Specification	φ^1_{ACC}	φ^2_{ACC}	φ^3_{ACC}	φ_{ACC}	φ^1_{ACC}	φ^2_{ACC}	φ^3_{ACC}	φ_{ACC}
$\mathrm{P}_{\mathrm{ACC}}$ #1	0.29	0.09	0.06	0.29	0.41	0.11	0.06	0.41
$\mathrm{P}_{\mathrm{ACC}}$ #2	0.19	0.05	0.03	0.19	0.24	0.06	0.04	0.24
$\mathrm{P}_{\mathrm{ACC}}$ #3	0.12	0.04	0.03	0.12	0.12	0.05	0.03	0.12
$\mathrm{PI}_{\mathrm{ACC}}$ #1	0.58	0.08	0.05	0.58	0.66	0.09	0.05	0.67
$\mathrm{PI}_{\mathrm{ACC}}$ #2	0.52	0.07	0.04	0.52	0.48	0.06	0.03	0.48
$\mathrm{PI}_{\mathrm{ACC}}$ #3	0.45	0.06	0.03	0.45	0.35	0.05	0.03	0.36
$\mathrm{MPC}_{\mathrm{ACC}}$ #1	0.13	0.05	0.03	0.13	0.19	0.06	0.04	0.20
$\mathrm{MPC}_{\mathrm{ACC}}$ #2	0.12	0.04	0.03	0.12	0.11	0.04	0.03	0.11
$\mathrm{MPC}_{\mathrm{ACC}}$ #3	0.12	0.04	0.03	0.12	0.11	0.04	0.02	0.11

TABLE VIII
LK FRs, WITH INPUT GENERATION, FOR SPECIFICATIONS IN (19) AND (20)

	No input generation				Heuristics by Eq. (30)				Ellipsoid method + dual game			
Sampling location	Interior		Boundary		Interior		Boundary		Interior		Boundary	
Controller \ Specification	φ^1_{LK}	φ_{LK}	φ^1_{LK}	φ_{LK}	φ^1_{LK}	φ_{LK}	φ^1_{LK}	φ_{LK}	φ^1_{LK}	φ_{LK}	φ^1_{LK}	φ_{LK}
P_{LK} #1	0.22	0.45	0.58	0.81	0.99	1.00	1.00	1.00	1.00	1.00	1.00	1.00
P_{LK} #2	0.00	0.95	0.00	0.99	0.00	1.00	0.00	1.00	0.00	1.00	0.00	1.00
P_{LK} #3	0.00	0.77	0.00	0.94	0.00	1.00	0.00	1.00	0.00	1.00	0.00	1.00
$\mathrm{PI}_{\mathrm{LK}}$ #1	0.00	0.33	0.00	0.71	1.00	1.00	1.00	1.00	0.38	0.65	0.63	0.91
$\mathrm{PI}_{\mathrm{LK}}$ #2	0.00	0.99	0.03	1.00	0.14	1.00	0.26	1.00	0.08	1.00	0.15	1.00
$\mathrm{PI}_{\mathrm{LK}}$ #3	0.00	0.95	0.00	0.99	0.00	1.00	0.01	1.00	0.00	0.99	0.00	1.00
$\mathrm{MPC}_{\mathrm{LK}}$ #1	0.00	0.00	0.00	0.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
$\mathrm{MPC}_{\mathrm{LK}}$ #2	0.00	0.00	0.00	0.00	0.04	0.04	0.09	0.09	0.04	0.04	0.08	0.08
$\mathrm{MPC}_{\mathrm{LK}}$ #3	0.00	0.00	0.00	0.00	0.002	0.002	0.01	0.01	0.002	0.002	0.01	0.01

To summarize, initial conditions on the boundary help identify safety violations in the transient behavior, whereas input generation techniques tend to capture safety violations due to persistent disturbances (i.e., steady state).

B. Lane Keeping Results

Let us denote the state vector $[y, v, \Delta\Psi, r]^\top$ in (14) as x_{LK}. We computed a controlled invariant set S_{LK} for the LK model in (14), sampled the boundary of this set with the proposed approach to find falsifying initial conditions, and use various input generation methods to generate road profiles. We explored three classes of controllers for meeting the LK requirement.

1) A proportional (P) state feedback controller, defined as
$$u = K_P^\top x_{\mathrm{LK}}. \tag{26}$$
Several P controllers are designed by placing the poles in different locations. The control input δ_f is obtained by saturating u to account for the practical limit of the actuator, i.e., $\delta_f = \mathrm{sat}^{\theta_s^{\max}}_{\theta_s^{\min}}(u)$.
2) A PI controller, defined as
$$u = K_I^\top \widetilde{x}_{\mathrm{LK}} \tag{27}$$
where $\widetilde{x}_{\mathrm{LK}}$ expands x_{LK} to include an error state e
$$\frac{d}{dt}\begin{bmatrix} x_{\mathrm{LK}} \\ e \end{bmatrix} = \begin{bmatrix} A_{\mathrm{LK}} & \mathbf{0}_{4\times 1} \\ [1\ 0\ 0\ 0] & 0 \end{bmatrix}\begin{bmatrix} x_{\mathrm{LK}} \\ e \end{bmatrix} + \begin{bmatrix} B_{\mathrm{LK}} \\ 0 \end{bmatrix} u. \tag{28}$$
Several PI controllers are designed by choosing different pole locations. Similarly, the control input δ_f is obtained by saturating u accordingly.
3) An MPC controller with the following formulation:
$$\begin{aligned} \min\ & \sum_{t=0}^{T} x_{\mathrm{LK}}(t)^\top Q x_{\mathrm{LK}}(t) + u^2(t) \\ \text{s.t. } & \text{Time-discretized dynamics of LK} \\ & \theta_s^{\min} \le u(t) \le \theta_s^{\max},\ t = 0, \ldots, T-1 \\ & x_{\mathrm{LK}}(0) = x_0. \end{aligned} \tag{29}$$
where x_0 is the initial condition, $Q = \mathrm{diag}([1, 0, 0, 0])$. Since the input saturation is accounted for by the MPC constraints in the MPC formulation, the control input $\delta_f = u$.

Table VIII summarizes the FRs for the above three controllers. The initial conditions are generated by sampling the interior and the boundary of set S_{LK}, and the disturbance profiles are generated using ellipsoid method plus dual game and using a heuristic described as follows:
$$r_d = \begin{cases} r_d^{\min} & \text{if } y(t+\tau) \ge t(t) \\ r_d^{\max} & \text{if } y(t+\tau) < t(t) \end{cases} \tag{30}$$
where τ is the sampling time of the discrete-time system.

Overall, our MPC design seems safer than the PI design, which is safer than the P controller. Note that none of these designs are tuned properly, the goal is just to demonstrate how controlled invariant sets can be used to evaluate different designs.

C. `Comma AI`

Our framework is flexible enough to evaluate any type of controller as long as their inputs and outputs match the inputs

```
def get_steer_from_curvature(self, curv, u):
  # this function is the exact inverse of calc_curvature, returning steer angle given curvature
  sf = calc_slip_factor(self.CP)
  return self.CP.l * self.CP.sR * (1. - sf * u**2) / (1. - self.CP.chi) * curv
```

```
def get_steer_from_curvature(self, curv, u):
  # this function is the exact inverse of calc_curvature, returning steer angle given curvature
  sf = calc_slip_factor(self.CP)
  return self.CP.l * self.CP.sR * (1. - sf * u**2) / ((1. - self.CP.chi)* u) * curv
```

Fig. 5. Output of a diff utility showing the modification in Comma AI. Left: Comma AI⋆ and right: Comma AI.

TABLE IX
ACC FRs: COMMA AI, FOR SPECIFICATIONS IN (17) AND (18)

Sampling location	Interior				Boundary			
Specification / Input generation	φ^1_{ACC}	φ^2_{ACC}	φ^3_{ACC}	φ_{ACC}	φ^1_{ACC}	φ^2_{ACC}	φ^3_{ACC}	φ_{ACC}
No input generation	0.15	0.00	0.00	0.15	0.42	0.04	0.00	0.44
Heuristic (max brake)	0.15	0.29	0.00	0.29	0.39	0.43	0.00	0.48
Ellipsoid method + dual game	0.15	0.17	0.00	0.29	0.36	0.19	0.00	0.48
aLead = K(vLead -v_des)	0.43	0.00	0.00	0.43	0.56	0.01	0.00	0.56

and outputs of the system models used. We can also directly use the source code of a controller after developing a proper interface. In this section, we demonstrate our framework on an open source real-world autonomous driving package developed by Comma AI, a start-up working on self-driving car technologies (see https://comma.ai/). However, since we are just using a simplified model for the vehicle dynamics, the interface might not accurately reflect the performance of the software on an actual car. This can be improved by improving the models and interfaces, but the goal in this section is to simply show the applicability of the framework on realistic control software rather than accurately mimicking the performance.

We describe how we interfaced the Comma AI code (commit 5524dc8[2] at https://github.com/commaai) with our ACC and LK framework. The Comma AI code is written in Python. We call the Python code directly from within MATLAB by developing appropriate wrappers for input/output matching as described next.

The ACC module of Comma AI outputs two values: 1) gas and 2) brake commands, normalized to $[0, 1]$ and $[-1, 0]$, respectively. We scale these gas and brake commands by the physical gas and brake limits of average mid-sized sedans, $F^{\max}_{w,p}$ and $F^{\min}_{w,p}$, respectively. We then clip the scaled gas and brake commands to the comfort bounds $[0, F^{\max}_w]$ and $[-F^{\max}_w, 0]$, respectively. The sum of the scaled gas and brake commands is used as the control input to our system.

The LK module of Comma AI requires extra interfacing with our simulations. First, Comma AI outputs a control between $u \in [-1, 1]$, and we assume that these bounds map linearly onto the range of steering angles $[\theta^{\min}_s, \theta^{\max}_s]$. Second, Comma AI takes as input the road profile, at dR_c discretization, for the upcoming 50 m, which is measured along the tangent line of the current car configuration. To provide this at time step T.

1) We compute a sequence of future road curvature disturbances $r_d^{1:n}(T) \doteq \{d_t\}_{t=1}^n(T)$, where $n \geq 50$ using one of the input generation methods. To be consistent with prior road profiles, we fix $r_d^{1:n-1}(T) = r_d^{2:n}(T-1)$ and only compute $r_d^n(T)$ from scratch. In principle, we can compute $r_d^{1:n}(T)$ entirely from scratch; this setting can be interpreted as driving with Comma AI vision sensor failure/noise, leading to inconsistent roads from prior time steps. However, by ensuring that we provide consistent roads, we give Comma AI the advantage here by assuming that the vision data is exact.
2) Assuming that the $r_d^{1:n}(T)$ trajectory was obtained from measurements taken at a rate of dT_c of the vehicle traveling at v_N m/s on the center line, we can estimate the center line in road-fixed coordinates, $R_d^{1:n}(T)$, by approximating that the road traces out arcs of angle $r_d^i dT_c$ at every time step i. That is,

$$R_d^i(T) = R_d^{i-1}(T) + \frac{v_N}{r_d^i(T)} \begin{bmatrix} \cos\left(\frac{r_d^i(T)}{v_N} - \frac{\pi}{2}\right) \\ \sin\left(\frac{r_d^i(T)}{v_N} - \frac{\pi}{2}\right) + 1 \end{bmatrix}.$$

3) Since R_d is relative to a road-fixed coordinate system, we rotate and translate R_d into the car frame by rotating each waypoint by $-\Delta\Psi$ and translating by $-y$ (denoted $R'_d(T)$).
4) We evaluate $R'_d(T)$ values at a discretization of 1 m along the tangent line using linear interpolation.

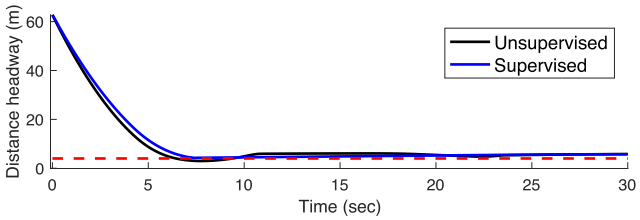


Fig. 6. Un/supervised ACC trajectories using Comma AI.

Additionally, we modified one of the vehicle model equations in the original Comma AI code (Fig. 5). We will refer to the original Comma AI code as Comma AI⋆, and to our modified code as Comma AI.

Figs. 7 and 8 show a trajectory generated by Comma AI and Comma AI⋆, respectively, that leaves the lane boundaries, overlaid by the trajectory generated by Comma AI and Comma AI⋆ when they are used as the legacy controller when the invariant set-based supervisor is active.

We see that for ACC, Comma AI manages to stay out of crashes for all initial conditions and input generation method

[2]After we settled on a version of Comma AI to use for this project, newer versions of the Comma AI code have changed the lane-keeping module from using a PI to an MPC-based controller. Testing this new controller within our framework is the subject of future work.

TABLE X
LK FRS: COMMA AI & COMMA AI* (WITHOUT MODIFICATION), FOR SPECIFICATIONS IN (19) AND (20)

	Comma AI				Comma AI* (without modification)			
Sampling location	**Interior**		**Boundary**		**Interior**		**Boundary**	
Specifcation / **Input generation**	φ^1_{LK}	φ_{LK}	φ^1_{LK}	φ_{LK}	φ^1_{LK}	φ_{LK}	φ^1_{LK}	φ_{LK}
No input generation	0.002	0.868	0.034	0.968	0	1	0.014	1
Heuristic by Eq. (30)	0.000	0.880	0.014	0.982	0.272	0.956	0.264	0.996
Ellipsoid method + dual game	0.006	0.952	0.038	0.994	1	1	1	1

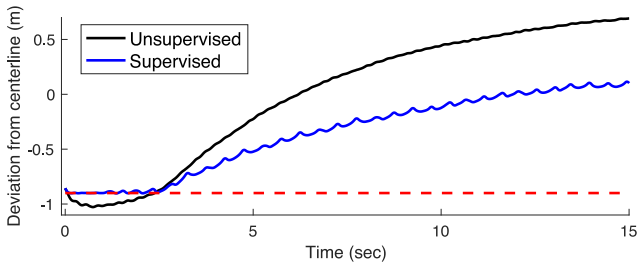


Fig. 7. Un/supervised LK trajectories using Comma AI.

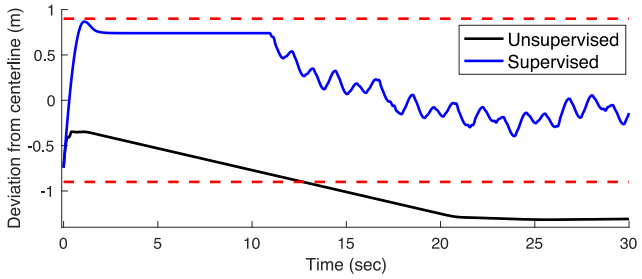


Fig. 8. Un/supervised LK trajectories using Comma AI*.

(see Fig. 6); however, it violates time and distance headways many times, as Table IX suggests. These violations are undesirable since the passengers might feel uncomfortable when Comma AI follows the lead car too closely. Furthermore, the Comma AI code itself sets a soft constraint for the desired distance headway being greater than 4 m, which is frequently violated.

The LK statistics in the left half of Table X indicate that while Comma AI stays within the lane boundaries for the most part when starting from nonzero initial conditions, in the process of stabilization, it tends to violate comfort bounds. Falsification is more likely when starting from initial conditions in the boundary than in the interior. Comma AI* (see the right half of Table X) drives slightly better on straight roads but performs much worse with input generation compared to Comma AI, which is consistent with the decent performance of Comma AI* on simple roads. The ellipsoid + dual game method of input generation seems to falsify Comma AI and Comma AI* more than the heuristic method; in fact, a straight road tends to falsify Comma AI and Comma AI* more than the heuristic method. This happens because the heuristic method tends to smooth out the natural overshoot that Comma AI and Comma AI* exhibit in their responses.

D. S-TaLiRo Results

For comparison and benchmarking purposes, we use S-TaLiRo, a falsification tool that is proposed in [2], to find falsifying initial conditions and disturbance trajectories. Although S-TaLiRo and our approach are somewhat complementary, we try to demonstrate some of the differences. First, note that S-TaLiRo does not provide any information about whether a falsifying initial condition disturbance pair is interesting, so it is not known if the violation is due to poor performance of the controller or it is unavoidable. To demonstrate to what extent S-TaLiRo can find interesting falsifying trajectories, we use the ACC example. We restrict the initial conditions that S-TaLiRo can choose by upper-bounding the headway $h \leq 200$ m. Let, $X_0 := [v^{\min}, v^{\max}] \times [h^{\min}, 200] \times [v_L^{\min}, v_L^{\max}]$. Then, the specification used in S-TaLiRo for falsification is

$$\left(\left(\left(v(0), h(0), v_L(0)\right) \in S \cap X_0\right) \wedge \left(\forall t : v_L(t) \geq v_L^{\min}\right)\right) \rightarrow \varphi^1_{\text{ACC}} \wedge \varphi^2_{\text{ACC}}. \tag{31}$$

In addition, we impose bounds on the external inputs, i.e., $a_L(t) \in [a_L^{\min}, a_L^{\max}]$ for all t; and the domain of the dynamics is accounted for by the simulation model. Note that by the assumptions on the initial conditions and v_L in (31), we avoid some of the trivially unsafe falsifications. To falsify P and PI controllers and Comma AI, we limit the number of samples S-TaLiRo can try to find a falsifying trajectory to 100; this acts as a timeout condition. We then run S-TaLiRo 100 times with the default option of simulated annealing, a random search method. Table XI summarizes the results, showing not all falsifying trajectories found by S-TaLiRo are interesting. Furthermore, S-TaLiRo sometimes fails to falsify P and PI controllers before our timeout condition, whereas Table V shows that the dual game approach always finds inputs that lead to falsification.

We also provide S-TaLiRo with initial conditions sampled from the invariant set boundary, therefore forcing it to find interesting initial conditions. These results are reported in the last column of Table XI. Although S-TaLiRo finds fewer falsifying trajectories for some controllers in this case, all of the falsifying trajectories found are interesting by definition. Comparing the last column in Table XI with that in Tables V and IX, we see that our input generation does better for most controllers except for Comma AI, for which S-TaLiRo has a slightly higher FR.

Finally, we give S-TaLiRo 111 initial conditions from the winning set of the dual game. S-TaLiRo takes 43 s to falsify all the points whereas dual game takes 170 s. This is partly

TABLE XI
ACC FALSIFICATION WITH S-TALIRO. THE FALSIFIED COLUMN SHOWS THE FRACTION OF TIMES S-TALIRO FINDS A FALSIFYING TRAJECTORY/INITIAL CONDITION PAIR FOR THE SPECIFICATION (31) BEFORE TIMING OUT. THE INVARIANT SET COLUMN SHOWS THE FRACTION OF TIMES A FALSIFYING PAIR WITH AN INTERESTING INITIAL CONDITION IS FOUND AMONG ALL RUNS. THE LAST COLUMN SHOWS THE FR WHEN S-TALIRO IS GIVEN INITIAL CONDITIONS SAMPLED FROM THE BOUNDARY OF THE INVARIANT SET

Controller	S-TaLiRo		S-TaLiRo with IC's on Boundary
	Falsified	In invariant set	
P_{ACC} #1	0.95	0.89	1.00
P_{ACC} #2	0.94	0.90	0.72
P_{ACC} #3	0.93	0.88	0.69
PI_{ACC} #1	0.97	0.91	0.96
PI_{ACC} #2	0.97	0.93	0.87
PI_{ACC} #3	0.96	0.91	0.89
Comma AI	0.68	0.51	0.58

due to the fact that the inputs selected by dual game input generation are arbitrary (within the winning inputs) but not necessarily aggressive, which can be mitigated by including an objective function in input generation phase. It is also worth mentioning that for these initial conditions, our approach is guaranteed to find a falsifying trajectory however, S-TaLiRo does not have such a guarantee due to its random nature.

V. CONCLUSION AND DISCUSSION

This paper proposed a simple idea on how to use controlled invariant sets and solutions from a dual game to generate interesting corner cases that can be used for falsification of safety specifications. We illustrated the effectiveness of this idea with an extensive case study on two autonomous driving functions, namely adaptive cruise control and lane keeping, with various types of controllers, including an open source autonomous driving package Comma AI. Our simulations show that we can identify corner cases with synthesis techniques and also supervise existing controllers to avoid failure in such corner cases.

The proposed approach should not be considered as an alternative to falsification techniques, as it is limited to safety specifications and to cases where approximating the maximal invariant set is possible. Therefore, it requires some knowledge of the system dynamics, although it is agnostic to the controller. In contrast, advanced falsification engines [2] can handle rich specifications given in signal temporal logic, and even black-box system models. On the other hand, we believe our approach can be used to seed falsification engines by applying it to the safety part of a specification - a direction for future research.

REFERENCES

[1] *Intelligent Transport Systems—Adaptive Cruise Control Systems—Performance Requirements and Test Procedures*, ISO Standard 15622:2010, 2010.

[2] Y. Annpureddy, C. Liu, G. E. Fainekos, and S. Sankaranarayanan, "S-TaLiRo: A tool for temporal logic falsification for hybrid systems," in *Proc. TACAS*, vol. 6605, 2011, pp. 254–257.

[3] S. Bak, K. Manamcheri, S. Mitra, and M. Caccamo, "Sandboxing controllers for cyber-physical systems," in *Proc. IEEE/ACM 2nd Int. Conf. Cyber Phys. Syst.*, 2011, pp. 3–12.

[4] F. Blanchini, "Survey paper: Set invariance in control," *Automatica*, vol. 35, no. 11, pp. 1747–1767, 1999.

[5] C. G. E. Boender *et al.*, "Shake-and-bake algorithms for generating uniform points on the boundary of bounded polyhedra," *Oper. Res.*, vol. 39, no. 6, pp. 945–954, 1991.

[6] E. De Santis, M. D. Di Benedetto, and L. Berardi, "Computation of maximal safe sets for switching systems," *IEEE Trans. Autom. Control*, vol. 49, no. 2, pp. 184–195, Feb. 2004.

[7] G. E. Fainekos, S. Sankaranarayanan, K. Ueda, and H. Yazarel, "Verification of automotive control applications using S-TaLiRo," in *Proc. Amer. Control Conf. (ACC)*, 2012, pp. 3567–3572.

[8] M. Ghosh, H.-Y. C. Yeh, S. Thomas, and N. M. Amato "Nearly uniform sampling on surfaces with applications to motion planning," Dept. Comput. Sci., Texas A&M Univ., College Station, TX, USA, Rep. TR13-005, 2013.

[9] M. Herceg, M. Kvasnica, C. N. Jones, and M. Morari, "Multi-parametric toolbox 3.0," in *Proc. Eur. Control Conf. (ECC)*, 2013, pp. 502–510.

[10] F. John, "Extremum problems with inequalities as subsidiary conditions," in *Traces and Emergence of Nonlinear Programming*. Basel, Switzerland: Springer, 2014, pp. 197–215.

[11] J. Leydold and W. Hörmann, "A sweep-plane algorithm for generating random tuples in simple polytopes," *Math. Comput.*, vol. 67, no. 224, pp. 1617–1635, 1998.

[12] L. P. Nilsson, "Correct-by-construction control synthesis for high-dimensional systems," Ph.D. dissertation, University of Michigan, Ann Arbor, MI, USA, 2017.

[13] P. Nilsson *et al.*, "Correct-by-construction adaptive cruise control: Two approaches," *IEEE Trans. Control Syst. Technol.*, vol. 24, no. 4, pp. 1294–1307, Jul. 2016.

[14] N. Ozay and P. Tabuada, "Guest editorial: Special issue on formal methods in control," *Discr. Event Dyn. Syst.*, vol. 27, no. 2, pp. 205–208, 2017.

[15] S. Prajna and A. Jadbabaie, "Safety verification of hybrid systems using barrier certificates," in *Proc. HSCC*, vol. 2993, 2004, pp. 477–492.

[16] E. Rimon and S. P. Boyd, "Efficient distance computation using best ellipsoid fit," in *Proc. IEEE Int. Symp. Intell. Control*, 1992, pp. 360–365.

[17] P. Roy, P. Tabuada, and R. Majumdar, "Pessoa 2.0: A controller synthesis tool for cyber-physical systems," in *Proc. HSCC*, 2011, pp. 315–316.

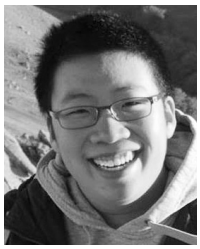
[18] M. Rungger and P. Tabuada, "Computing robust controlled invariant sets of linear systems," *IEEE Trans. Autom. Control*, vol. 62, no. 7, pp. 3665–3670, Jul. 2017.

[19] S. Sankaranarayanan and G. Fainekos, "Falsification of temporal properties of hybrid systems using the cross-entropy method," in *Proc. HSCC*, 2012, pp. 125–134.

[20] D. Seto, B. H. Krogh, L. Sha, and A. Chutinan, "Dynamic control system upgrade using the simplex architecture," *IEEE Control Syst.*, vol. 18, no. 4, pp. 72–80, Aug. 1998.

[21] R. L. Smith, "Efficient Monte Carlo procedures for generating points uniformly distributed over bounded regions," *Oper. Res.*, vol. 32, no. 6, pp. 1296–1308, 1984.

[22] S. W. Smith, P. Nilsson, and N. Ozay, "Interdependence quantification for compositional control synthesis with an application in vehicle safety systems," in *Proc. IEEE 55th Conf. Decis. Control (CDC)*, Las Vegas, NV, USA, 2016, pp. 5700–5707.

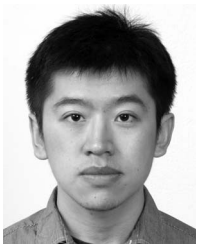


Glen Chou received dual B.S. degrees in electrical engineering and computer science and mechanical engineering from the University of California at Berkeley, Berkeley, CA, USA, in 2017. He is currently pursuing the Ph.D. degree in electrical and computer engineering with the University of Michigan, Ann Arbor, MI, USA.

His current research interests include machine learning and control, with particular emphasis on safe, adaptive learning, formal methods, and reinforcement learning.

Yunus Emre Sahin (S'16) received the B.S. degree in electrical and electronics engineering from Middle East Technical University, Ankara, Turkey, in 2014. He is currently pursuing the Ph.D. degree in electrical and computer engineering with the University of Michigan, Ann Arbor, MI, USA.

His current research interests include control theory, formal methods and optimization with applications in cyber-physical systems, multiagent coordination, and autonomous driving.



Liren Yang (S'17) received the B.S. degree in electrical and computer engineering from Shanghai Jiao Tong University, Shanghai, China, in 2013. He is currently pursuing the Ph.D. degree in electrical engineering with the University of Michigan, Ann Arbor, MI, USA.

His current research interests include control theory, hybrid systems, and formal methods.

Kwesi J. Rutledge received the B.S. degree in electrical engineering from the University of Michigan, Ann Arbor, MI, USA, in 2015 and the M.S. degree in electrical engineering from the University of California at San Diego, San Diego, CA, USA, in 2017. He is currently pursuing the Ph.D. degree with the Department of Electrical Engineering and Computer Science, University of Michigan.

His current research interests include estimation and formal methods as they relate to estimator synthesis and specification decomposition.

Petter Nilsson (M'15) received the B.S. degree in engineering physics and the M.S. degree in optimization and systems theory from the KTH Royal Institute of Technology, Stockholm, Sweden, in 2011 and 2013, respectively, the Ph.D. degree in electrical engineering from the University of Michigan, Ann Arbor, MI, USA, in 2017, and the B.S. degree in business and economics from the Stockholm School of Economics, Stockholm.

He is currently a Post-Doctoral Scholar with the California Institute of Technology, Pasadena, CA, USA, where he conducts research on specification-driven control and autonomy for safety-critical cyber-physical systems, with applications in autonomous driving, space exploration, and multiagent coordination.

Necmiye Ozay (M'04) received the B.S. degree in electrical engineering from Bogazici University, Istanbul, Turkey, in 2004, the M.S. degree in electrical engineering from Pennsylvania State University at University Park, University Park, PA, USA, in 2006, and the Ph.D. degree in electrical engineering from Northeastern University, Boston, MA, USA, in 2010.

She was a Post-Doctoral Scholar with the California Institute of Technology, Pasadena, CA, USA, between 2010–2013. She is currently an Assistant Professor of electrical engineering and computer science with the University of Michigan, Ann Arbor, MI, USA. Her research interests include control of dynamical systems, optimization, and formal methods with applications in cyber-physical systems, system identification, verification, and validation and autonomy.

Dr. Ozay was a recipient of several awards including the IEEE Control Systems Society Conference on Decision and Control Best Student Paper Award in 2008 and a prize paper award from the *Journal of Nonlinear Analysis: Hybrid Systems* for the years 2014–2016, the DARPA Young Faculty Award in 2014, the NSF CAREER Award, the NASA Early Career Faculty Award, the DARPA Director's Fellowship in 2016, and the ONR YIP Award in 2018. She has served on program committees of several international conferences and is currently an Associate Editor for the *Journal of Discrete Event Dynamic Systems*.