2019 IEEE 58th Conference on Decision and Control (CDC)
Palais des Congrès et des Expositions Nice Acropolis
Nice, France, December 11-13, 2019

Average-based Robustness for Continuous-Time Signal Temporal Logic

*Noushin Mehdipour[1], *Cristian-Ioan Vasile[2] and Calin Belta[1]

***Abstract*— We propose a new robustness score for continuous-time Signal Temporal Logic (STL) specifications. Instead of considering only the most severe point along the evolution of the signal, we use average scores to extract more information from the signal, emphasizing robust satisfaction of all the specifications' subformulae over their entire time interval domains. We demonstrate the advantages of this new score in falsification and control synthesis problems in systems with complex dynamics and multi-agent systems.**

I. Introduction

The increased adoption and deployment of cyber-physical systems in critical infrastructure in recent years have led to important questions about their correct functioning. These devices embedded in our cars, planes, and homes are becoming increasingly complex. Thus, automated tools are necessary to alleviate the need for manual design and proof of correct behavior. Formal methods have provided approaches to specify temporal requirements of systems, formally verify whether systems satisfy given specifications, and automatically synthesize control policies that are guaranteed to be correct by construction [1]. Temporal logics such as Linear Temporal Logics (LTL) [2], Metric Temporal Logic (MTL) [3], Time Window Temporal Logic (TWTL) [4] and Signal Temporal Logic (STL) [5] are popular specification languages due to their expressivity, similarity to natural language, and an amenable structure to symbolic reasoning.

STL defines properties over continuous-time signals in continuous spaces, and has been adopted for monitoring [5], falsification, and control problems such as path planning and multi-agent control with time constraints [6], [7], [8]. One of the major advantages of STL is that it admits quantitative semantics [9], known as robustness, which is interpreted as a measure of satisfaction or violation of a desired task or property. Thus, problems involving STL can be set up as optimization of the robustness, and powerful optimization algorithms can be leveraged.

The traditional robustness introduced in [9] uses max and min functions resulting in a non-differentiable function. It only takes into account the most critical part of the signal, and, thus, induces: 1) a masking effect, where the satisfaction of other parts of the formulae do not contribute to the score, and 2) locality, where only the value of the signal at only one time point determines the score. Both these properties have a negative impact when used in optimization problems. The masking effect hinders optimizers from obtaining gradient information to improve solutions, while locality results in solutions that are brittle to noise. The traditional score was used as the objective function in an optimization problem and maximized using heuristic optimization algorithms such as Particle Swarm Optimization, Simulated Annealing and Rapidly Exploring Random Trees (RRTs) in different synthesis, falsification and control problems [10], [11], [12]. Exact approaches in [13], [14] encoded the temporal and Boolean constraints as Mixed Integer Linear Programming (MILP) problems and used off-the-shelf MILP solvers to maximize robustness. Although MILP solved the issues of heuristic algorithms regarding guarantees on finding global optima, they were not scalable for large number of variables or complex temporal constraints, due to their NP-complete nature. Another drawback of MILP implementation is the necessity of having both constraints and system dynamics be linear or linearizable.

Recent efforts to improve STL robustness focus on smoothing the max and min functions to employ gradient-based optimization techniques [15], [16]. However, these approximations cause errors compared to the traditional robustness, and the *soundness* property is lost. Another effort is refining the robustness function to include more information of the signal, rather than only its most satisfying or violating part. In [17], *averageSTL* robustness was defined using time average for temporal operators in continuous-time signals and used to solve a falsification problem. This score did not tackle the problem with non-smooth min and max operations. [18] improved STL robustness for discrete signals by defining *Discrete Average Space Robustness (DASR)* for *Globally* and *Until* operators. The authors removed the non-smoothness by defining a simplified version called *Discrete Simplified Average Space Robustness (DSASR)*. However, a positive *DASR* or *DSASR* score did not correspond to satisfaction of the specification. Therefore, similar to approximation methods, additional constraints were imposed to guarantee correctness.

In [19], we defined *Arithmetic-Geometric Mean (AGM)* robustness for discrete signals and showed the superiority of *AGM* to the traditional robustness for control synthesis problems, and compared our gradient-based maximization to MILP implementations. In this paper, we extend the STL quantitative score from [19] to continuous-time signals by defining an Arithmetic-Geometric Integral Mean (AGIM) robustness. Rather than merely evaluating the most satisfying or violating points, we evaluate each subformulae and at every time, highlighting both the degree of satisfaction and how frequently a specification is satisfied. In contrast to previous works on average score where arithmetic mean was employed for some temporal operators, we refine robustness for all Boolean and temporal operators. We use arithmetic- and product-based means to capture the

*These authors contributed equally. This work was partially supported at Boston University by the National Science Foundation under grants IIS-1723995, CPS-1446151, and CMMI-1400167.[1]Noushin Mehdipour (noushinm@bu.edu), Calin Belta (cbelta@bu.edu) are with the Division of Systems Engineering at Boston University, Boston, and [2]Cristian-Ioan Vasile (cvasile@mit.edu) is with the Laboratory for Information and Decision Systems at Massachusetts Institute of Technology, Cambridge, MA, USA.

978-1-7281-1398-2/19/$31.00 ©2019 IEEE

importance of all outliers in signals based on the nature of operators. Moreover, in the proposed score, positive values correspond to satisfaction of the specification and negative values correspond to violation, showing the *soundness*. We use this score in falsification and multi-agent control synthesis problems for continuous-time systems.

II. Preliminaries

Let $f : \mathbb{R}^n \to \mathbb{R}$ be a real function. We define $[f]_+ = \begin{cases} f & f > 0 \\ 0 & \text{otherwise} \end{cases}$ and $[f]_- = -[-f]_+$, where $f = [f]_+ + [f]_-$.

A. Signal Temporal Logics (STL)

STL [5] is a logic designed to specify temporal properties of continuous-time signals. A *signal* $S : \mathbb{R}_{\geq 0} \to \mathbb{R}^n$ is a real-value function mapping each time $t \in \mathbb{R}_{\geq 0}$ to an n-dimensional vector $S(t)$. The STL syntax is defined as:

$$\varphi := \top \mid \mu \mid \neg\varphi \mid \varphi_1 \wedge \varphi_2 \mid \varphi_1 \mathbf{U}_{[a,b]} \varphi_2, \tag{1}$$

where $\top$ is the logical *True*, μ is a *predicate*, $\neg$ and $\wedge$ are the Boolean *negation* and *conjunction* operators, and $\mathbf{U}$ is the temporal *until* operator. Other Boolean and temporal operators are defined as $\varphi_1 \vee \varphi_2 := \neg(\neg\varphi_1 \wedge \neg\varphi_2)$ (disjunction), $\mathbf{F}_{[a,b]}\varphi := \top \mathbf{U}_{[a,b]}\varphi$ (*eventually*), and $\mathbf{G}_{[a,b]}\varphi := \neg\mathbf{F}_{[a,b]}\neg\varphi$ (*Globally*). In this paper, we focus on $\mathbf{F}$ and $\mathbf{G}$ operators. Temporal operator $\mathbf{F}_{[a,b]}\varphi$ requires the "specification φ to become *True* at some time in $[a,b]$". $\mathbf{G}_{[a,b]}\varphi$ requires "φ to be *True* at all times in $[a,b]$". A STL specification can have one or more predicates $\mu := l(S) \geq 0$ connected by Boolean and temporal operators and $l : \mathbb{R}^n \to \mathbb{R}$ is a real, linear or nonlinear continuous function defined over values of elements of S. STL is equipped with qualitative semantics which shows *whether* a signal S satisfies a given specification φ at time t ($S(t) \models \varphi$) or violates it ($S(t) \not\models \varphi$), and quantitative semantics, known as *robustness*, which measures *how much* the signal is satisfying or violating the specification. We denote the robustness for a specification φ with respect to signal S at time t as $\rho(\varphi, S, t)$ and refer to it as traditional robustness. For details on calculating traditional robustness, please refer to [9].

B. Geometric Product Integral

The geometric integral $\prod_a^b f(x)^{dx}$ is the continuous analog of the discrete product operator and is defined as [20]:

$$\prod_a^b f(x)^{dx} = \exp\left(\int_a^b \ln f(x)\, dx\right)$$

III. Problem Statement

Consider a continuous-time dynamical system as:

$$\dot{q}(t) = f(q, u),\ q(0) = q_0, \tag{2}$$

where $t \in \mathbb{R}_{\geq 0}$, $q(t) \in \mathbf{Q} \subseteq \mathbb{R}^n$ is the state, $u(t) \in \mathbf{U} \subseteq \mathbb{R}^m$ is the control input at time t, $q_0 \in \mathbf{Q}$ is the initial state, and $f : \mathbb{R}^n \to \mathbb{R}^n$ is locally Lipschitz. We denote the resulting system trajectory for the given control input $u(t)$ as $\langle q, u\rangle$. For system (2), we consider specifications given as STL formulae over predicates in its state. For example, the requirement that a vehicle maintains a maximum speed of 100 over 10 minutes can be written as $\phi = G_{[0,10]} Speed \leq 100$.

Problem 1: [Falsification] Given system (2) and a STL formula ϕ_f over predicates in state q, find input $u(t)$ such that the resulting trajectory violates the specification, i.e. $\langle q, u\rangle \not\models \phi_f$.

Problem 2: [Synthesis] Given system (2) and a STL formula ϕ_s over predicates in the state q, find input $u(t)$ such that the resulting trajectory satisfies the specification, i.e., $\langle q, u\rangle \models \phi_s$.

In other words, a falsification problem is interpreted as finding a counterexample for the given specification to predict possible faults that may occur in system, e.g., falsification of ϕ happens if "at some time between 0 and 10, speed goes beyond the 100 limit". However, in a control synthesis problem, we are interested in finding a control input such that the system trajectory meets the desired requirements, e.g., we want "the vehicle speed to be less than 100 for all times between 0 and 10".

Motivating Example: Assume we have an agent with the specification "*eventually* reach point B from point A and *always* avoid obstacle". Fig. 1 shows the discrete steps agent takes to reach B. Although these steps do not collide with obstacle and result in a positive discrete robustness, the trajectory connecting these steps passes through the obstacle. However, using a continuous-time score, we can correctly find a trajectory that does not collide with obstacle at any time. This example illustrates the need for a continuous-time score, as discretizing the system is not always preferable, especially when an appropriate discretization frequency is not known.

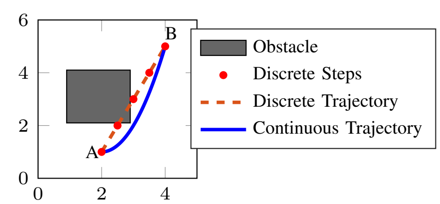

Fig. 1. Failure in collision avoidance in a discrete-time system

IV. Arithmetic-Geometric Integral Mean (AGIM) Robustness

We propose a new average-based robustness η for bounded continuous-time signals that captures more information about the signal relative to the traditional score. Our robustness definition returns a normalized score $\eta \in [-1, 1]$ with $\eta \in (0, 1]$ and $\eta \in [-1, 0)$ corresponding to satisfaction and violation of the specification, respectively; and $\eta = 0$ when satisfaction is inconclusive. Similar to traditional robustness, $|\eta|$ is a measure of how much the specification is satisfied or violated, while the normalization helps to have a meaningful comparison between signals of different scales. Throughout the definitions, we assume that we have bounded signals, all *Lebesgue integrable* in additive and multiplicative sense [20], and the components normalized to the interval $[-1, 1]$.

Definition 1 (AGIM Robustness): Let $S : \mathbb{R}_{\geq 0} \to [-1, 1]^n$ with s_i being its i^{th} component and $\pi \in [-1, 1]$. The normalized AGIM robustness $\eta(\varphi, S, t)$ for specification φ with respect to signal S at time t is recursively defined as:

- **logical** $True$ $\quad \eta(\top, S, t) := 1$
- $\varphi : s_i \geq \pi \quad \eta(\varphi, S, t) := \frac{1}{2}(s_i(t) - \pi)$
- **Negation** $\quad \eta(\neg\varphi, S, t) := -\eta(\varphi, S, t)$
- **Boolean and temporal operators** See (3)

$$\eta(\varphi_1 \wedge \varphi_2 \wedge ... \wedge \varphi_m, S, t) := \begin{cases} \sqrt[m]{\prod_{i=1,...,m} (1 + \eta(\varphi_i, S, t))} - 1 & \forall i \in [1, ..., m] \;.\; \eta(\varphi_i, S, t) > 0, \\ \frac{1}{m} \sum_{i=1,...,m} [\eta(\varphi_i, S, t)]_- & \text{otherwise} \end{cases}$$

$$\eta(\varphi_1 \vee \varphi_2 ... \vee \varphi_m, S, t) := \begin{cases} \frac{1}{m} \sum_{i=1,...,m} [\eta(\varphi_i, S, t)]_+ & \exists i \in [1, ..., m] \;.\; \eta(\varphi_i, S, t) > 0, \\ -\sqrt[m]{\prod_{i=1,...,m} (1 - \eta(\varphi_i, S, t))} + 1 & \text{otherwise} \end{cases}$$

$$\eta(\mathbf{G}_{[a,b]}\varphi, S, t) := \begin{cases} \sqrt[b-a]{\prod_a^b (1 + \eta(\varphi, S, \tau))^{d\tau}} - 1 & \forall \tau \in [t+a, t+b] \;.\; \eta(\varphi, S, \tau) > 0, \\ \frac{1}{b-a} \int_a^b [\eta(\varphi, S, t'_k)]_- d\tau & \text{otherwise} \end{cases} \tag{3}$$

$$\eta(\mathbf{F}_{[a,b]}\varphi, S, t) := \begin{cases} \frac{1}{b-a} \int_a^b [\eta(\varphi, S, t'_k)]_+ d\tau & \exists \tau \in [t+a, t+b] \;.\; \eta(\varphi, S, \tau) > 0, \\ -\sqrt[b-a]{\prod_a^b (1 - \eta(\varphi, S, \tau))^{d\tau}} + 1 & \text{otherwise} \end{cases}$$

Algorithm 1 describes the steps to determine satisfaction of specification ϕ and recursively calculate the AGIM robustness.

Theorem 1 (Soundness): The AGIM robustness is sound:

$$\begin{aligned} \eta(\varphi, S, t) > 0 \Leftrightarrow \rho(\varphi, S, t) > 0 \Rightarrow S \models \varphi, \\ \eta(\varphi, S, t) < 0 \Leftrightarrow \rho(\varphi, S, t) < 0 \Rightarrow S \not\models \varphi. \end{aligned} \tag{4}$$

Proofs are provided in the arXiv version[1].

A. Averaging Properties

The AGIM robustness finds satisfaction or violation of specification ϕ regarding all the subformulae φ of ϕ and at all appropriate times in the interval. In contrast to [17], [18] where only arithmetic mean was used, we argue for the need of both arithmetic and geometric integral means for different cases as follows. The arithmetic mean is affected by the total sum value of data and is usually used when no significant outliers are present. On the other hand, the geometric mean is sensitive to unevenness and is able to measure consistency in data. Consider the *eventually* operator, $F_{[a,b]}\varphi$, which is satisfied if φ is satisfied at least at one time. Taking the arithmetic mean, we have a score that takes into account the total sum of all satisfying times and is sensitive to the critical ones (outliers). On the other hand, for the *globally* operator, $G_{[a,b]}\varphi$, to be satisfied, we need φ to be satisfied at all times. Therefore, for this case we will use the geometric mean to not only regard all the times, but also emphasize the consistency in satisfaction. In other words, for $G_{[a,b]}\varphi$ to have a high score, we need all the times to have (even) high scores. Same argument holds for the robustness of the $\wedge$ and $\vee$ operators.

B. Smoothness Properties

The AGIM robustness $\eta(\phi, S, t)$ is smooth in S almost everywhere except on the satisfaction boundaries $\rho(\varphi, S, \tau) = 0$, where φ is a subformula of ϕ, and appropriate times τ as given in (3). Moreover, the gradient of η with respect to elements of S that are part of ϕ's predicates is non-zero wherever it is smooth. $\eta(\phi, S, t)$ is left-continuous in t for continuous signals, and differentiable in t almost everywhere if S is differentiable.

[1]http://arxiv.org/abs/1909.00898

V. Robustness Optimization

We formulate the falsification and control synthesis problems defined in Sec. III as optimization problems. Based on soundness of AGIM robustness, to find a violating trajectory for a specification ϕ_f, we can check if $\eta(\phi_f, \langle q, u\rangle) < 0$. Smaller η corresponds to a more violating behavior. Therefore, we can solve the falsification Problem 1 by minimizing the robustness of satisfaction of ϕ_f over all allowed control inputs:

$$\begin{aligned} u^* = \text{argmin}_u \eta(\phi_f, \langle q, u\rangle), \\ \text{s.t.} \quad \eta(\phi_f, \langle q, u\rangle) < 0, \\ \dot{q}(t) = f(q, u), \\ q(0) = q_0, \\ q(t) \in \mathbf{Q} \subseteq \mathbb{R}^n, \\ u(t) \in \mathbf{U} \subseteq \mathbb{R}^m. \end{aligned} \tag{5}$$

Similarly, soundness of AGIM robustness allows us to determine satisfaction of specification ϕ_s if $\eta(\phi_s, \langle q, u\rangle) > 0$. Larger η corresponds to a stronger satisfaction of the desired requirements. Therefore, we can solve the synthesis Problem 2 and find the trajectory which best satisfies the desired ϕ_s by maximizing robustness over all allowed control inputs:

$$\begin{aligned} u^* = \text{argmax}_u \eta(\phi_s, \langle q, u\rangle), \\ \text{s.t.} \quad \eta(\phi_s, \langle q, u\rangle) > 0, \\ \dot{q}(t) = f(q, u), \\ q(0) = q_0, \\ q(t) \in \mathbf{Q} \subseteq \mathbb{R}^n, \\ u(t) \in \mathbf{U} \subseteq \mathbb{R}^m. \end{aligned} \tag{6}$$

In [19], we assumed system dynamics is also smooth and used gradient ascent to optimize the robustness. In this work, we use the MATLAB Optimization Toolbox to deal with more complex and not necessarily differentiable dynamics as in [21]. We focus on finding piecewise constant inputs. For a given horizon T, we consider the continuous-time input to be:

$$u(t) = u_k, \qquad (k-1)T_s \le t \le kT_s \tag{7}$$

where T_s is the input sample time, $k \in \mathbb{N}, k \le \frac{T}{T_s}$. We hold each sample value u_k constant for one sample interval T_s to create a continuous-time input $u(t)$. We apply this continuous-time input to the system (2) to generate the continuous-time

Algorithm 1: STL SATISFACTION AND AGIM ROBUSTNESS RECURSIVE CALCULATION

Input: STL Formula ϕ; Signal S
Output: AGIM Robustness $\eta(\phi, S, t)$

Find $\eta(\varphi_i, S, \tau)$ for $i = \{1, 2, ..., m\}$ and $\tau \in [a, b]$;
CASE $\phi = \varphi_1 \wedge \varphi_2 \wedge ... \varphi_m$;
if $ANY(\eta(\varphi_i, S, t) \leq 0)$ *for* $i = \{1, 2, ..., m\}$ **then**
  $S \not\models \phi$ `Violation`;
  $\eta(\phi, S, t) := \frac{1}{m} \sum_i [\eta(\varphi_i, S, t)]_-$;
else
  $S \models \phi$ `Satisfaction`;
  $\eta(\phi, S, t) := \sqrt[m]{\prod_{i=1,...,m} (1 + \eta(\varphi_i, S, t))} - 1.$
end
CASE $\phi = \varphi_1 \vee \varphi_2 \vee ... \varphi_m$;
if $ANY(\eta(\varphi_i, S, t) > 0)$ *for* $i = \{1, 2, ..., m\}$ **then**
  $S \models \phi$ `Satisfaction`;
  $\eta(\phi, S, t) := \frac{1}{m} \sum_i [\eta(\varphi_i, S, t)]_+$;
else
  $S \not\models \phi$ `Violation`;
  $\eta(\phi, S, t) = -\sqrt[m]{\prod_{i=1,...,m} (1 - \eta(\varphi_i, S, t))} + 1.$
end
CASE $\phi = \mathbf{G}_{[a,b]}\varphi$;
if $ANY(\eta(\varphi, S, \tau) \leq 0)$ *for* $\tau \in [a, b]$ **then**
  $S \not\models \phi$ `Violation`;
  $\eta(\phi, S, t) := \frac{1}{b-a} \int_a^b [\eta(\varphi, S, \tau)]_- d\tau$;
else
  $S \models \phi$ `Satisfaction`;
  $\eta(\phi, S, t) := \sqrt[b-a]{\prod_a^b (1 + \eta(\varphi, S, \tau))^{d\tau}} - 1.$
end
CASE $\phi = \mathbf{F}_{[a,b]}\varphi$;
if $ANY(\eta(\varphi_i, S, \tau) > 0)$ *for* $\tau \in [a, b]$ **then**
  $S \models \phi$ `Satisfaction`;
  $\eta(\phi, S, t) := \frac{1}{b-a} \int_a^b [\eta(\varphi, S, \tau)]_+ d\tau$;
else
  $S \not\models \phi$ `Violation`;
  $\eta(\phi, S, t) = -\sqrt[b-a]{\prod_a^b (1 - \eta(\varphi, S, \tau))^{d\tau}} + 1.$
end

trajectory. The optimization process starts with generating a random sample sequence $u_s = \{u_1, u_2, ..., u_{T/T_s}\}$, converted to a continuous-time input $u(t)$ using (7) and finding system execution $\langle q, u \rangle$ starting from initial state q_0. We then use Matlab Constrained Parallel Optimization Toolbox to find an optimal control policy u_s^* under imposed constraints which optimizes the robustness η for the given STL constraints ϕ. All algorithms and simulations are implemented in Matlab running on an iMac with 3.3GHz Intel Core i5 CPU 32GB RAM.

VI. CASE STUDIES

In this section, we demonstrate the efficacy of the proposed robustness to solve falsification and control synthesis problems. We start with a simple verification problem, in which we compare the traditional and proposed robustness for a trajectory produced by the system under a given control input. Assume we want to study if the step response of a dynamical system takes values greater than a threshold, say 1.2. We can specify

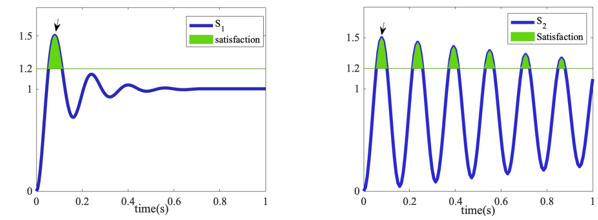

Fig. 2. Transient behavior of two dynamical systems with same ρ (points marked with arrow) and different η (areas colored in green).

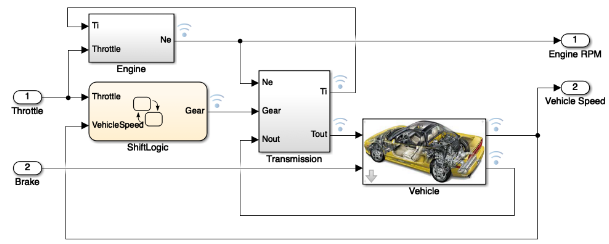

Fig. 3. The Simulink automatic transmission model diagram [21].

this behavior using STL as $\phi = F_{[0,T]}S > 1.2$, where S is the step response and T is the duration time. Fig. 2 shows the step responses of two different systems during the first second. The traditional robustness considers only the most satisfying part of the response, therefore, returns the same robustness for both systems determined by the point marked with arrow: $\rho(\phi, S_1) = \rho(\phi, S_2) = \max_{t \in [0,T]} (S_i(t) - 1.2) = 0.3$. However, the AGIM robustness takes the time average over the signal at all the satisfying time intervals determined by the colored area, and returns $\eta(\phi, S_1) \quad << \quad \eta(\phi, S_2)$ which helps to distinguish between the behaviors of the two systems. This example also illustrates the importance of having a continuous-time robustness rather than discretizing the dynamics and using a discrete-time score. For instance, if S_1 is discretized with a frequency smaller than $15Hz$, we miss the overshoot since the discrete robustness returns a non-positive score.

A. Falsification

We use the Automatic Transmission Model from Simulink [21] shown in Fig. 3, and compare falsifying the traditional robustness versus the proposed one both in computation time and performance. We show that, by using the new robustness, we can find not only a violating execution, but a more severe violating execution which indeed requires a higher priority to be managed. This is helpful especially in the design stage to figure out the worst performance of the system for a given temporal and space constraints and limits on inputs. In Fig. 3, the simulation time is $T = 30$ and *Throttle* is the input $u(t)$ with $\mathbf{U} = [0, 80]$ parameterized as a piecewise constant signal with $T_s = 5$ as in (7). The desired requirement of the system is: "RPM must always be less than 4000 and $Speed$ must always be less than 100 between time 0 and 30 seconds", specified as:

$$\phi_{Falsify} = G_{[0,30]}RPM \leq 4000 \wedge G_{[0,30]}Speed \leq 100 \quad (8)$$

The falsification for this specification happens if "RPM is greater than 4000 or $Speed$ greater than 100". Fig. 4 and Fig. 5 show $Speed$ and RPM traces found by minimizing the

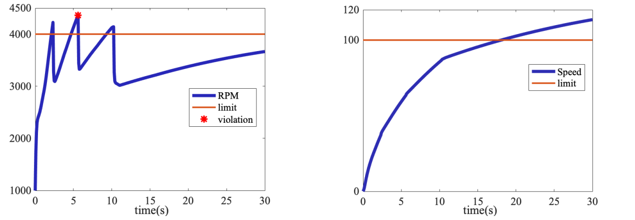

Fig. 4. Falsifying execution (RPM Left, $Speed$ Right) minimizing the traditional robustness ρ determined by the single point marked with *.

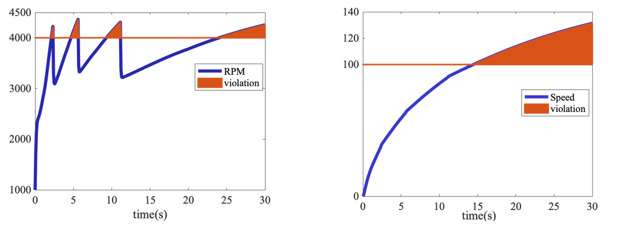

Fig. 5. Falsifying execution (RPM Left, $Speed$ Right) minimizing AGIM robustness η determined by the areas colored in red.

traditional and AGIM robustness, respectively. As illustrated in Fig. 4, for the traditional robustness, the unnormalized score is calculated considering only the most violating part of the signal, $\min\left(\min_{t\in[0,30]}(4000-RPM(t)), \min_{t\in[0,30]}(100-Speed(t))\right)$, marked with $*$. On the other hand, traces found by minimizing the AGIM robustness (3) evaluate all violating parts of both RPM and $Speed$ over the entire time and results in a more severe violating behavior, shown as the colored area in Fig. 5. Table I shows the average run time, number of optimization iterations and total number of robustness evaluations to find the first falsifying traces (first time robustness is negative) and when traces with minimum robustness are found.

B. Control Synthesis

We use the proposed robustness in a multi-agent system with time constraints. The agents' high level task is to achieve consensus or formation, and for certain time intervals, we impose additional temporal tasks for each agent.

Example 1: Consider 2 agents with dynamics (9) to achieve consensus and meanwhile satisfy some temporal requirements:

$$\dot{p}_i(t) = v_i(t), \quad \dot{v}_i(t) = u_{c_i}(t) + u_i(t), \quad i = 1, 2 \tag{9}$$

where p_i is position, v_i is velocity, u_{c_i} is the input to reach consensus and u_i is the input to be synthesized for agent i to satisfy the temporal task. The consensus input is [22]:

$$u_{c_i} = -\gamma_p \sum_{j\in N_i} a_{ij}(p_i - p_j) - \gamma_v \sum_{j\in N_i} a_{ij}(v_i - v_j) - \gamma_d v_i$$

where N_i is the set of neighboring agents for i, a_{ij} shows whether agent i is connected to agent j, $\gamma_p, \gamma_v, \gamma_d$ are constant coefficients for consensus on position, speed and dampening speed. The desired task is "*Eventually* $Agent1$ visits *Blue and* $Agent2$ visits *Green* within $[5, 15]$ *and eventually* $Agent1$ and $Agent2$ visit *Yellow* within $[15, 20]$ *and Always* within $[0, 20]$ $Agent1$ and $Agent2$ stay inside the boundary with speeds being in the allowed range", specified as STL formula:

$$\begin{aligned}\phi_1 = F_{[5,15]}\, p_1 \in Blue \;\wedge\; F_{[5,15]}\, p_2 \in Green \;\wedge\\ F_{[15,20]}\, p_1 \in Yellow \;\wedge\; F_{[15,20]}\, p_2 \in Yellow \;\wedge\\ G_{[0,20]}\, p_1, p_2 \in \mathbf{P} \;\wedge\; G_{[0,20]}\, v_1, v_2 \in \mathbf{V},\end{aligned} \tag{10}$$

TABLE I

TRADITIONAL AND AGIM ROBUSTNESS COMPARISON FOR $\phi_{Falsify}$

	Traditional			AGIM		
	Time	#Itr	#FuncEval	Time	#Itr	#FuncEval
First Falsifying Trace	23Sec.	4	35	27Sec.	7	56
Min. Falsifying Trace	49Sec.	14	107	54Sec.	20	138

where $p_i = [x_i, y_i]$ is position with $\mathbf{P} = [0, 10]^2$ and initial states $p_{1_0} = [0, 4]$, $p_{2_0} = [5, 2]$, $v_i = [vx_i, vy_i]$ is velocity with $\mathbf{V} = [-2, 2]^2$ and $u_i = [u_{x_i}, u_{y_i}]$ is the input vector with $\mathbf{U} = [-2, 2]^2$. Regions are represented as logical formulae, for instance, $p_2 \in Green := 6 \le x_2 \wedge x_2 \le 8 \wedge 5 \le y_2 \wedge y_2 \le 7$. The trajectory obtained by applying the optimal control input u^* to each agent found by maximizing the robustness η with $T_s = 0.1$ is shown in Fig. 6 (Left). Within $[0, 5]$, there is no individual temporal task for the agents except for staying inside the boundary. Therefore, the consensus input drives the agents to move towards each other. Starting at time $t = 5$, each agent is supposed to eventually visit a region within the next 10 seconds. The synthesized input u^* pushes the agents to visit their assigned regions *as fast as possible* and *stay* in each region (*center*) *as long as possible*, as it results in a higher score due to the averaging properties of η over time, and definition of space robustness, e.g., $\operatorname{argmax}_{x_2,y_2} \eta(p_2 \in Green, [x_2, y_2]) = [7, 6]$. Later, within $[15, 20]$, both agents visit region *Yellow* (*center*) *as fast as possible* and *stay* there until $t = 20$.
We next add an obstacle to the environment, and update the specification such that both agents avoid the obstacle:

$$\phi_2 = \phi_1 \wedge G_{[0,20]}\, p_1 \notin Black \wedge G_{[0,20]}\, p_2 \notin Black \tag{11}$$

Fig. 6 (Right) shows the agents' trajectories satisfying ϕ_2 avoiding the obstacle. Note that the trajectories are updated to avoid the obstacle, and due to the constraints on time and control input, the agents visit region $Yellow$ (robustness is positive) but do not reach its center. Fig. 7 shows the scores corresponding to each agent visiting the assigned regions for the specified time interval. In Fig.7 (Left), $Agent1$ reaches $Blue$ at $t = 7.3$ and stays until $t = 12.9$, and reaches $Yellow$ at $t = 16.3$. $Agent2$ reaches $Green$ at $t = 8.9$ and stays until $t = 14.2$, and reaches $Yellow$ at $t = 16.4$. In Fig.7 (Right), agents change their trajectories to avoid the obstacle. Therefore, it takes a longer time to get to $Yellow$ ($Agent1$ at $t = 18.2$ and $Agent2$ at $t = 17.1$). There is no temporal task in the first 5 seconds, and we illustrate trajectories up to $t = 20$ to show the satisfaction of the temporal tasks but consensus is achieved at later times.

Example 2: Consider a multi-agent system of 3 agents with dynamics (12) to form a triangle formation of length 2 and meanwhile satisfy some temporal requirements:

$$\begin{aligned}\dot{p}_i(t) &= u_{f_i}(t) + u_i(t), \quad i = 1, 2, 3\\ u_{f_i}(t) &= -\gamma_p \sum_{j\in N_i} a_{ij}(p_i(t) - p_j(t) - d_{ij}),\end{aligned} \tag{12}$$

where u_i is the input to agent i to be synthesized in order to satisfy the temporal logic requirements, u_{f_i} is the input to achieve the formation, and d_{ij} is the distance between agents i and j [23]. The desired task is specified as STL formula:

$$\begin{aligned}\phi_3 = F_{[5,15]}\, p_1 \in Blue \;\wedge\; F_{[15,25]}\, p_2 \in Green \;\wedge\\ F_{[25,35]}\, p_3 \in Red \;\wedge F_{[35,40]} G_{[0,5]}\, p_1 \in Yellow \;\wedge\\ G_{[0,45]}\, p_1, p_2, p_3 \in \mathbf{P}\end{aligned} \tag{13}$$

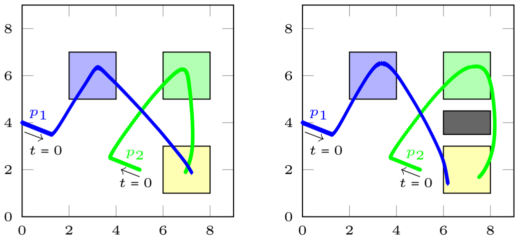

Fig. 6. Agents' trajectories satisfying ϕ_1 (Left) and ϕ_2 (Right)

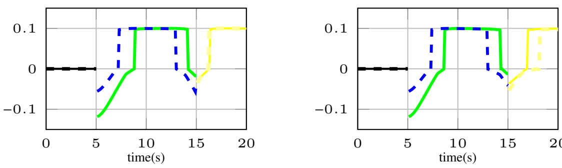

Fig. 7. Scores related to each agent visiting assigned regions for the specified time interval satisfying ϕ_1 (Left) and ϕ_2 (Right). Dashed and solid lines correspond to $Agent1$ and $Agent2$, respectively. Scores for each region are colored with the same color with positive score meaning that the agent is inside the region.

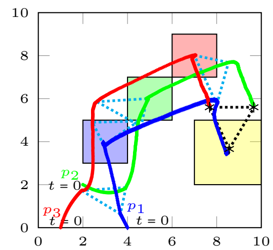

Fig. 8. Agents' trajectories satisfying ϕ_3. Triangles in cyan show formation at $t = 5, 13, 19, 28$ formed due to u_{f_i} while the agents meet their temporal requirements, and the black triangle shows final formation at $t = 45$.

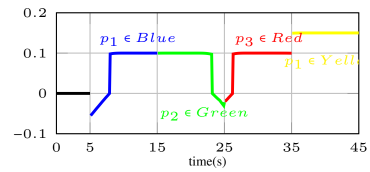

Fig. 9. Scores related to each agent visiting assigned regions satisfying ϕ_3. Positive score means the agent is inside the region.

with initial states $p_{1_0} = [4, 0]$, $p_{2_0} = [2, 2]$, $p_{3_0} = [1, 0]$ and $u_i = [u_{x_i}, u_{y_i}]$ is the input vector with $\mathbf{U} = [-3, 3]^3$.
The trajectory obtained by applying the optimal control input u^* to each agent found by maximizing the robustness η with $T_s = 0.1$ is shown in Fig. 8. Starting at time $t = 5$, $Agent1$ eventually visits its assigned region within the next 10 seconds (enters $Blue$ at $t = 8$, Fig. 9). Note that at this time, no temporal tasks are specified for the other agents. Therefore, only the formation input drives these agents to form a triangle. The same argument holds for $Agent2$ in $[15, 25]$ and $Agent3$ in $[25, 35]$. Within $[35, 40]$, $Agent1$ visits $Yellow$, and stays there for at least 5 seconds, and the desired formation is achieved.

VII. Conclusion

We presented a novel robustness score for continuous-time STL, which uses arithmetic and geometric integral means. We demonstrated that this score incorporates requirements of all the subformulae and all the times of the formula. This comes in contrast with traditional approaches that consider only critical ones. We showed that our definition provides a better violation or satisfaction score in falsification and control applications.

References

[1] C. Belta, B. Yordanov, and E. A. Gol, *Formal methods for discrete-time dynamical systems*. Springer, 2017, vol. 89.
[2] A. Pnueli, "The temporal logic of programs," in *18th Annual Symposium on Foundations of Computer Science*, Oct 1977, pp. 46–57.
[3] R. Koymans, "Specifying real-time properties with metric temporal logic," *Real-time systems*, vol. 2, no. 4, pp. 255–299, 1990.
[4] C.-I. Vasile, D. Aksaray, and C. Belta, "Time window temporal logic," *Theoretical Computer Science*, vol. 691, pp. 27–54, 2017.
[5] O. Maler and D. Nickovic, "Monitoring temporal properties of continuous signals," in *Formal Techniques, Modelling and Analysis of Timed and Fault-Tolerant Systems*. Springer, 2004, pp. 152–166.
[6] C. I. Vasile, V. Raman, and S. Karaman, "Sampling-based Synthesis of Maximally-Satisfying Controllers for Temporal Logic Specifications," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Vancouver, BC, Canada, 2017, pp. 3840–3847.
[7] L. Lindemann, C. K. Verginis, and D. V. Dimarogonas, "Prescribed performance control for signal temporal logic specifications," in *2017 IEEE 56th Annual Conference on Decision and Control (CDC)*, Dec 2017, pp. 2997–3002.
[8] A. Nikou, D. Boskos, J. Tumova, and D. V. Dimarogonas, "Cooperative planning for coupled multi-agent systems under timed temporal specifications," in *2017 American Control Conference (ACC)*. IEEE, 2017, pp. 1847–1852.
[9] A. Donzé and O. Maler, "Robust satisfaction of temporal logic over real-valued signals," in *International Conference on Formal Modeling and Analysis of Timed Systems*. Springer, 2010, pp. 92–106.
[10] N. Mehdipour, D. Briers, I. Haghighi, C. M. Glen, M. L. Kemp, and C. Belta, "Spatial-temporal pattern synthesis in a network of locally interacting cells," in *2018 IEEE Conference on Decision and Control (CDC)*. IEEE, 2018, pp. 3516–3521.
[11] H. Abbas and G. Fainekos, "Convergence proofs for Simulated Annealing falsification of safety properties," in *Allerton Conference on Communication, Control, and Computing*, 2012, pp. 1594–1601.
[12] S. M. LaValle and J. J. Kuffner Jr, "Randomized kinodynamic planning," *The International Journal of Robotics Research*, vol. 20, no. 5, pp. 378–400, 2001.
[13] V. Raman, A. Donzé, M. Maasoumy, R. M. Murray, A. Sangiovanni-Vincentelli, and S. A. Seshia, "Model predictive control with signal temporal logic specifications," in *53rd IEEE Conference on Decision and Control*, Dec 2014, pp. 81–87.
[14] S. Saha and A. A. Julius, "An MILP approach for real-time optimal controller synthesis with metric temporal logic specifications," in *American Control Conference (ACC)*. IEEE, 2016, pp. 1105–1110.
[15] Y. V. Pant, H. Abbas, and R. Mangharam, "Smooth operator: Control using the smooth robustness of temporal logic," in *IEEE Conference on Control Technology and Applications (CCTA)*, 2017, pp. 1235–1240.
[16] X. Li, Y. Ma, and C. Belta, "A policy search method for temporal logic specified reinforcement learning tasks," in *Annual American Control Conference (ACC)*. IEEE, 2018, pp. 240–245.
[17] T. Akazaki and I. Hasuo, "Time robustness in mtl and expressivity in hybrid system falsification," in *International Conference on Computer Aided Verification*. Springer, 2015, pp. 356–374.
[18] L. Lindemann and D. V. Dimarogonas, "Robust control for signal temporal logic specifications using discrete average space robustness," *Automatica*, vol. 101, pp. 377–387, 2019.
[19] N. Mehdipour, C.-I. Vasile, and C. Belta, "Arithmetic-Geometric Mean Robustness for Control from Signal Temporal Logic Specifications," *2019 American Control Conference(ACC)*, arXiv preprint at https://arxiv.org/abs/1903.05186.
[20] A. E. Bashirov, E. M. Kurpınar, and A. Özyapıcı, "Multiplicative calculus and its applications," *Journal of Mathematical Analysis and Applications*, vol. 337, no. 1, pp. 36–48, 2008.
[21] H. A. Bardh Hoxha and G. Fainekos, "Benchmarks for temporal logic requirements for automotive systems," *Proceedings of applied verification for continuous and hybrid systems*, 2014.
[22] N. Mehdipour, F. Abdollahi, and M. Mirzaei, "Consensus of multi-agent systems with double-integrator dynamics in the presence of moving obstacles," in *2015 IEEE Conference on Control Applications (CCA)*. IEEE, 2015, pp. 1817–1822.
[23] R. Olfati-Saber, "Flocking for multi-agent dynamic systems: Algorithms and theory," California Inst. of Tech. Pasadena Control and Dynamical System, Tech. Rep., 2004.