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 living 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], also 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. Moreover, both these works used arithmetic average to define robustness, which, as we show in this paper, is not a good measure for all temporal and Boolean operators.

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 and approximation robustness.

Discretizing a system is not always a good idea, especially in systems with fast transient behaviors or unknown frequency. Discretizing a system behavior with an incorrect frequency may result in missing an event or a violating behavior, e.g., a robot might collide with an obstacle in between the discrete-time instances even if it is collision free at those discrete-time moments.

In this paper, we extend the STL quantitative score from [19] to continuous-time signals by defining an Arithmetic-Geometric Integral Mean (AGIM) robustness. We discuss the advantages of this new definition compared to the traditional robustness and previous works on average 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 importance of all outliers in signals based on the nature of the operators. Moreover, in the proposed score, positive values correspond to satisfaction of the specification and negative values correspond to violation, showing the *soundness* of our definition. We use this score in falsification and multi-agent control synthesis problems for continuous-time systems, and compare the behavior and run-time complexity with traditional robustness.

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:

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

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*). Due to brevity, 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 $[a, b]$". $\varphi_1\mathbf{U}_{[a,b]}\varphi_2$ is used to specify that "φ_2 must become *True* at some time within $[a, b]$ and φ_1 must be always *True* prior to that". 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 ($S(t) \vDash \varphi$) or violates it ($S(t) \nvDash \varphi$), and quantitative semantics, also known as *robustness*, which measures *how much* the signal is satisfying or violating the specification.

Definition 1 (STL Robustness): The robustness $\rho(\varphi, S, t)$ for formula φ with respect to signal S at time t is recursively defined as [9]:

$$\begin{aligned}
\rho(\top, S, t) &:= \rho_\top, \\
\rho(\mu, S, t) &:= l(S(t)), \\
\rho(\neg\varphi, S, t) &:= -\rho(\varphi, S, t), \\
\rho(\varphi_1 \wedge \varphi_2, S, t) &:= \min(\rho(\varphi_1, S, t), \rho(\varphi_2, S, t)), \\
\rho(\varphi_1 \vee \varphi_2, S, t) &:= \max(\rho(\varphi_1, S, t), \rho(\varphi_2, S, t)), \\
\rho(\mathbf{G}_{[a,b]}\varphi, S, t) &:= \min_{\tau \in [t+a, t+b]} \rho(\varphi, S, \tau), \\
\rho(\mathbf{F}_{[a,b]}\varphi, S, t) &:= \max_{\tau \in [t+a, t+b]} \rho(\varphi, S, \tau),
\end{aligned} \quad (2)$$

where $\rho_\top \in \mathbb{R} \cup \{+\infty\}$ is the maximum robustness.

Theorem 1: Robustness ρ is sound, meaning that $\rho(\varphi, S, t) > 0$ implies that signal S satisfies φ at time t, and $\rho(\varphi, S, t) < 0$ implies that S violates φ at time t. We denote the robustness of specification φ at time 0 with respect to the signal S by $\rho(\varphi, S)$. We refer to this definition as traditional robustness.

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:

$$\dot{q}(t) = f(q, u), \ q(0) = q_0, \quad (3)$$

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 (3), we consider specifications given as STL formulae over predicates in its state q. 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 (3) and a STL formula ϕ_f over predicates in the state q, find a control input $u(t)$ such that the resulting trajectory $\langle q, u \rangle$ violates the specification, i.e. $\langle q, u \rangle \nvDash \phi_f$.

Problem 2: [Synthesis] Given system (3) and a STL formula ϕ_s over predicates in the state q, find a control input $u(t)$ such that the resulting trajectory $\langle q, u \rangle$ satisfies the specification, i.e., $\langle q, u \rangle \vDash \phi_s$.

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, the 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". Previous works use traditional and average-based robustness to find violating (falsification) and satisfying (synthesis) trajectories for the given specification. However, as described in Sec. I, traditional robustness considers only the most satisfying or violating subformula and time, and discard information of the other parts. Other average-based scores refine robustness only for the temporal operators using arithmetic means, which also have some limitations. We address these shortcomings by designing a new robustness for continuous-time signals.

Motivating Example: Assume we have an agent with the specification "*eventually* reach point B from point A and *always* avoid obstacle (colored in black)". The circle marks in Fig. 1 show the discrete steps the 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.
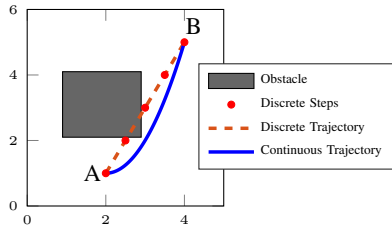

Fig. 1. Failure in collision avoidance in a discrete-time system

IV. ARITHMETIC-GEOMETRIC INTEGRAL MEAN (AGIM) ROBUSTNESS

We extend our work in [19] and 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 and proofs, 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 2 (AGIM Robustness): Let $S : \mathbb{R}_{\geq 0} \rightarrow [-1, 1]^n$ with s_i being its ith component and $\pi \in [-1, 1]$. The normalized AGIM robustness $\eta(\varphi, S, t)$ with respect to signal S at time t is recursively defined as:

- **logical** $True$ $\quad \eta(\top, S, t) \coloneqq 1$
- $\varphi : s_i \geq \pi$ $\quad \eta(\varphi, S, t) \coloneqq \frac{1}{2}(s_i(t) - \pi)$
- **Negation** $\quad \eta(\neg\varphi, S, t) \coloneqq -\eta(\varphi, S, t)$
- **Boolean and temporal operators** See (4)

Algorithm 1 describes the steps to determine satisfaction or violation of specification ϕ, and to recursively calculate the AGIM robustness with respect to signal S.

Algorithm 1: STL SATISFACTION AND AGIM ROBUSTNESS RECURSIVE CALCULATION

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

1   Find $\eta(\varphi_i, S, \tau)$ for $i = \{1, 2, ..., m\}$ and $\tau \in [a, b]$;
2   **CASE** $\phi = \varphi_1 \wedge \varphi_2 \wedge ...\varphi_m$;
3   **if** $ANY(\eta(\varphi_i, S, t) \leq 0)$ *for* $i = \{1, 2, ..., m\}$ **then**
4      $S \not\models \phi$ **Violation**;
5      $\eta(\phi, S, t) \coloneqq \frac{1}{m} \sum_i [\eta(\varphi_i, S, t)]_-$;
6   **else**
7      $S \models \phi$ **Satisfaction**;
8      $\eta(\phi, S, t) \coloneqq \sqrt[m]{\prod_{i=1,...,m} (1 + \eta(\varphi_i, S, t))} - 1.$
9   **end**
10   **CASE** $\phi = \varphi_1 \vee \varphi_2 \vee ...\varphi_m$;
11   **if** $ANY(\eta(\varphi_i, S, t) > 0)$ *for* $i = \{1, 2, ..., m\}$ **then**
12      $S \models \phi$ **Satisfaction**;
13      $\eta(\phi, S, t) \coloneqq \frac{1}{m} \sum_i [\eta(\varphi_i, S, t)]_+$;
14   **else**
15      $S \not\models \phi$ **Violation**;
16      $\eta(\phi, S, t) = -\sqrt[m]{\prod_{i=1,...,m} (1 - \eta(\varphi_i, S, t))} + 1.$
17   **end**
18   **CASE** $\phi = \mathbf{G}_{[a,b]}\varphi$;
19   **if** $ANY(\eta(\varphi, S, \tau) \leq 0)$ *for* $\tau \in [a, b]$ **then**
20      $S \not\models \phi$ **Violation**;
21      $\eta(\phi, S, t) \coloneqq \frac{1}{b-a} \int_a^b [\eta(\varphi, S, \tau)]_- d\tau$;
22   **else**
23      $S \models \phi$ **Satisfaction**;
24      $\eta(\phi, S, t) \coloneqq \sqrt[b-a]{\prod_a^b (1 + \eta(\varphi, S, \tau))^{d\tau}} - 1.$
25   **end**
26   **CASE** $\phi = \mathbf{F}_{[a,b]}\varphi$;
27   **if** $ANY(\eta(\varphi_i, S, \tau) > 0)$ *for* $\tau \in [a, b]$ **then**
28      $S \models \phi$ **Satisfaction**;
29      $\eta(\phi, S, t) \coloneqq \frac{1}{b-a} \int_a^b [\eta(\varphi, S, \tau)]_+ d\tau$;
30   **else**
31      $S \not\models \phi$ **Violation**;
32      $\eta(\phi, S, t) = -\sqrt[b-a]{\prod_a^b (1 - \eta(\varphi, S, \tau))^{d\tau}} + 1.$
33   **end**

Remark 1: The ANY method employed in Algorithm 1 is dependent on the representation of signals and the problem solved, i.e., falsification, synthesis, verification or moni-

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

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

$$\eta(\mathbf{G}_{[a,b]}\varphi, S, t) := \begin{cases} \sqrt[b-a]{\prod\limits_a^b \left(1 + \eta(\varphi, S, \tau)\right)^{d\tau}} - 1 & \forall \tau \in [t+a, t+b] \, . \, \eta(\varphi, S, \tau) > 0, \\ \frac{1}{b-a} \int\limits_a^b [\eta(\varphi, S, t'_k)]_- d\tau & \text{otherwise} \end{cases} \tag{4}$$

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

toring. As with traditional robustness, there exist efficient representations and data structures for these tasks.

Theorem 2 (Soundness): The AGIM robustness is sound, i.e., a trajectory with strictly positive robustness satisfies the given specification, and a trajectory with strictly negative robustness violates the given specification:

$$\eta(\varphi, S, t) > 0 \Leftrightarrow \rho(\varphi, S, t) > 0 \Rightarrow S \vDash \varphi,$$
$$\eta(\varphi, S, t) < 0 \Leftrightarrow \rho(\varphi, S, t) < 0 \Rightarrow S \nvDash \varphi. \tag{5}$$

Proof: [Sketch] We prove the soundness by structural induction over the formula ϕ. The *base case* corresponding to $\varphi \in \{\top, \mu\}$ is trivially true. For the *induction case*, we assume that the property holds for subformulae φ, and we need to show that it holds under Boolean and temporal operators. For brevity, we show only the "Globally" case. If $\eta(\mathbf{G}_{[a,b]}\varphi, S, t) > 0$, then $\eta(\varphi, S, \tau) > 0$ for all $\tau \in [t+a, t+b]$, which implies $\rho(\varphi, S, \tau) > 0$, $\forall \tau \in [t+a, t+b]$, and thus $\rho(\mathbf{G}_{[a,b]}\varphi, S, t) > 0$. Conversely, if $\eta(\mathbf{G}_{[a,b]}\varphi, S, t) < 0$, then there exists $\tau \in [t+a, t+b]$ such that $\eta(\varphi, S, \tau) < 0$, which implies $\rho(\varphi, S, \tau) < 0$ and $\rho(\mathbf{G}_{[a,b]}\varphi, S, t) < 0$. ∎

Proposition 1: Let S be a left (right) continuous signal and ϕ a STL formula. If $\eta(\phi, S, t) = 1$ (maximum normalized score), then $\eta(\varphi, S, \tau) = 1$ for all subformulae φ of ϕ and appropriate times τ as given by (4). Similarly, if $\eta(\phi, S, t) = -1$ (minimum normalized score), then $\eta(\varphi, S, \tau) = -1$ for all subformulae φ of ϕ and appropriate times τ in (4). Proof follows directly from Definition 2.

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 while the min and max functions in the traditional robustness result in a score which only considers the most critical time or subformula. 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*, $F_{[a,b]}\varphi$, 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 also sensitive to the critical ones (outliers). Therefore, if at some time we have a large score for subformula φ, the score for $F_{[a,b]}\varphi$ is highly affected by that. On the other hand, for the *globally*, $G_{[a,b]}\varphi$, to be satisfied, we need φ to be satisfied at all times. Even a single time near violation (with small positive score) has a significant impact on the satisfaction of the specification. Therefore, for this case we will use the geometric mean to not only regard all the satisfying 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. The same argument holds for the robustness of the $\wedge$ and $\vee$ operators.

B. Logical Properties

Theorem 3 (Boolean Property): The following hold:
1) *Idempotence:* $\eta(\varphi \wedge \varphi, S) = \eta(\varphi, S)$
2) *Commutativity:* $\eta(\varphi_1 \wedge \varphi_2, S) = \eta(\varphi_2 \wedge \varphi_1, S)$
3) *Monotonicity:* $\eta(\varphi_1 \wedge \varphi_2, S) \leq \eta(\varphi_3 \wedge \varphi_4, S)$, $\forall \varphi_i$ where $\eta(\varphi_1, S) \leq \eta(\varphi_3, S)$, $\eta(\varphi_2, S) \leq \eta(\varphi_4, S)$

The same properties hold for disjunction $\vee$.

Theorem 4 (Rules of Inference): The following hold:
1) *Law of non-contradiction:*
$$\eta(\varphi \wedge \neg\varphi, S) < 0 \, , \, \forall \varphi \, \text{ where } \, \eta(\varphi, S) \neq 0$$

2) *Law of excluded middle:*
$$\eta(\varphi \vee \neg\varphi, S) > 0 \, , \, \forall \varphi \, \text{ where } \, \eta(\varphi, S) \neq 0$$

3) *Double negation:*
$$\eta(\neg(\neg\varphi), S) = \eta(\varphi, S) \, , \, \forall \varphi$$

4) *DeMorgan's law:*
$$\eta(\varphi_1 \vee \varphi_2, S) = \eta(\neg(\neg\varphi_1 \wedge \neg\varphi_2), S)$$
$$\eta(\mathbf{G}_{[a,b]}\varphi, S) = \eta(\neg\mathbf{F}_{[a,b]}\neg\varphi, S)$$

All proofs follow directly from Definition 2.

C. Smoothness Properties

The AGIM robustness $\eta(\phi, S, t)$ is smooth in $S \in [-1, 1]^n$ 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 (4). Moreover, the gradient of η with respect to the elements of S that are part of ϕ's predicates is non-zero wherever it is smooth. The AGIM robustness $\eta(\phi, S, t)$ is left-continuous in t for continuous signals, and differentiable in t almost everywhere if S is differentiable.

Proof: [Sketch] All properties follow by structural induction. For continuity (differentiability) in t, we also need to show that the set of times where the function is not continuous (differentiable) is countable. This follows from the left-continuity, which implies only a countable set of times of discontinuity exists. ∎

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 the specification ϕ_f over all allowed control inputs:

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

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 specification ϕ_s by maximizing robustness over all allowed control inputs:

$$
\begin{aligned}
u^* = \operatorname{argmax}_u &\eta(\phi_s, \langle q, u \rangle), \\
\text{s.t.} \quad &\eta(\phi_s, \langle q, u \rangle) > 0, \\
&\dot{q}(t) = f(q, u), \\
&q(0) = q_0, \\
&q(t) \in \mathbf{Q} \subseteq \mathbb{R}^n, \\
&u(t) \in \mathbf{U} \subseteq \mathbb{R}^m.
\end{aligned}
\tag{7}
$$

As discussed earlier, our robustness definition is smooth and differentiable almost everywhere. 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 in the form of:

$$
u(t) = u_k, \qquad (k-1)T_s \le t \le kT_s
\tag{8}
$$

where T_s is 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 (3) to generate the continuous-time trajectory. The optimization processes for falsification and control synthesis start 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 (8) 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 and scalability of the proposed robustness to solve the falsification and control synthesis problems defined above. We compare our results with the traditional robustness in performance and computation time.

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 the step response of a dynamical system and consider the case where we want to find if the system response takes values greater than a threshold, say 1.2. We can specify 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 $\eta(\phi, S_1) \ll \eta(\phi, S_2)$ 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 was discretized with a frequency smaller than $15Hz$, we would have missed the overshoot since the discrete robustness would return a non-positive score and fail to provide correct information regarding the satisfaction of the specification.

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.

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

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

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

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

In Fig. 3, the simulation time is $T = 30$ seconds and *Throttle* is the input $u(t)$ with $\mathbf{U} = [0, 80]$ and parameterized as a piecewise constant signal with $T_s = 5$ as:

$$u(t) = \begin{cases} u_1 & 0 \le t < 5 \\ u_2 & 5 \le t < 10 \\ \vdots & \vdots \\ u_6 & 25 \le t < 30 \end{cases}$$

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:

$$\phi_{Falsify} = G_{[0,30]}RPM \le 4000 \wedge G_{[0,30]}Speed \le 100 \quad (9)$$

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 traditional and the 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, $\min\left(\min_{t\epsilon[0,30]}(4000 - RPM(t)), \min_{t\epsilon[0,30]}(100 - Speed(t))\right)$, marked with *. Note that, although the speed is violating the limit after $t = 20$, robustness is only affected by RPM (most violating subformula). On the other hand, the traces found by minimizing the AGIM robustness, calculated using (4), 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. The average time to evaluate the robustness at each evaluation

is 0.34 ms for the traditional robustness and 0.41 ms for the AGIM robustness. Note that the time in the first column includes the time of generating the input, running the simulink for the generated input at each evaluation, generating resulting trajectories and calculating robustness.

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: We consider 2 agents with double integrator dynamics. We would like the agents to achieve consensus and meanwhile satisfy some temporal requirements. The agents' dynamics are given as:

$$\dot{p}_i(t) = v_i(t), \quad \dot{v}_i(t) = u_{c_i}(t) + u_i(t), \quad i = 1, 2 \quad (10)$$

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 defined as [22]:

$$u_{c_i} = -\gamma_p \sum_{j\in N_i} a_{ij}(p_i - p_j) - \gamma_v \sum_{j\in N_i} a_{ij}(v_i - v_j) - \gamma_d v_i$$
$$(11)$$

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 the 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*",

	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

specified as STL formula:

$$\phi_1 = F_{[5,15]}\, p_1 \in Blue \;\wedge\; F_{[5,15]}\, p_2 \in Green \;\wedge$$
$$F_{[15,20]}\, p_1 \in Yellow \;\wedge\; F_{[15,20]}\, p_2 \in Yellow \;\wedge \quad (12)$$
$$G_{[0,20]}\, p_1, p_2 \in \mathbf{P} \;\wedge\; G_{[0,20]}\, v_1, v_2 \in \mathbf{V},$$

where $p_i = [x_i, y_i]$ is the position vector 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 the velocity with $\mathbf{V} = [-2,2]^2$ and $u_i = [u_{x_i}, u_{y_i}]$ is the input vector with $\mathbf{U} = [-2,2]^2$. The regions are represented as logical formulae, for instance, for $Agent2$ visiting the $Green$ region, we have $p_2 \in Green := 6 \leq x_2 \;\wedge\; x_2 \leq 8 \;\wedge\; 5 \leq y_2 \;\wedge\; y_2 \leq 7$.

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 (4), and the definition of space robustness, e.g., $\operatorname*{argmax}_{x_2,y_2} \eta\,(p_2 \in Green, [x_2, y_2]) = [7,6]$ (4). 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:

$$\phi_2 = \phi_1 \;\wedge\; G_{[0,20]}\, p_1 \notin Black \;\wedge\; G_{[0,20]}\, p_2 \notin Black \quad (13)$$

Fig. 6 (Right) shows the agents' trajectories satisfying ϕ_2 and 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 $t = 7.35$ and stays until $t = 12.9$, and reaches $Yellow$ at $t = 16.35$. $Agent2$ reaches $Green$ at $t = 8.9$ and stays until $t = 14.2$, and reaches $Yellow$ at $t = 16.4$. In Fig.7 (Right), where the obstacle is added, the agents change their trajectories to avoid the obstacle. Therefore, it takes a longer time to get to $Yellow$, $Agent1$ arrives at $t = 18.20$ and $Agent2$ at $t = 17.10$. Note that 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.
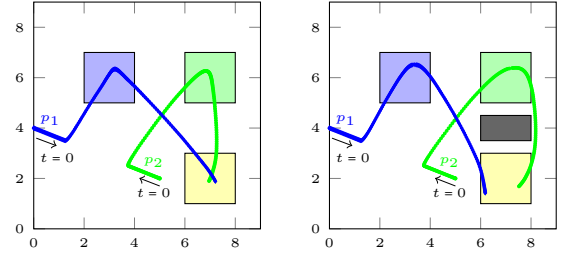

Fig. 6. Agents' trajectories satisfying ϕ_1 (Left) and ϕ_2 (Right)
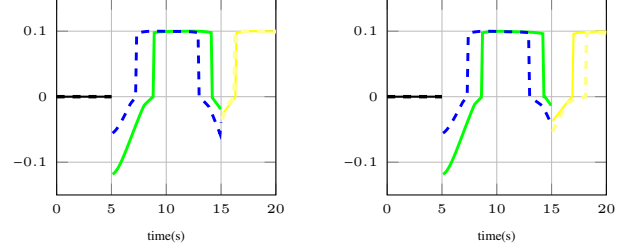

Fig. 7. Scores related to each agent visiting assigned regions for the specified time interval satisfying ϕ_1 (Left) and ϕ_2 (Right). Dashed and solid lines correspond to $Agent1$ and $Agent2$, respectively. Scores for each region are colored with the same color with positive score meaning that the agent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

Example 2: We consider a multi-agent system made of 3 agents with single integrator dynamics. We would like the agents to form a triangle formation of length 2 and meanwhile satisfy some temporal logic requirements. The agents' dynamics are given as:

$$\dot{p}_i(t) = u_{f_i}(t) + u_i(t), \quad i = 1,2,3$$
$$u_{f_i}(t) = -\gamma_p \sum_{j \in N_i} a_{ij}(p_i(t) - p_j(t) - d_{ij}), \quad (14)$$

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 "*Eventually Agent1 visits **Blue** within* $[5,15]$ *and Eventually Agent2 visits **Green** within* $[15,25]$ *and eventually Agent3 visits **Red** within* $[25,35]$ *and eventually Agent1 visits **Yellow** within* $[35,40]$ *and Always stay in **Yellow** for the next 5 seconds and Always all agents stay inside the boundary*", specified as STL formula:

$$\phi_3 = F_{[5,15]}\, p_1 \in Blue \;\wedge\; F_{[15,25]}\, p_2 \in Green \;\wedge$$
$$F_{[25,35]}\, p_3 \in Red \;\wedge F_{[35,40]} G_{[0,5]}\, p_1 \in Yellow \;\wedge \quad (15)$$
$$G_{[0,45]}\, p_1, p_2, p_3 \in \mathbf{P}$$

where $p_i = [x_i, y_i]$ is the position vector with $\mathbf{P} = [0,10]^3$ and 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. Within $[0,5]$, the formation input drives the agents to form a triangular formation. Starting at time $t = 5$, $Agent1$ eventually visits its assigned region within the next 10 seconds (enters $Blue$ at $t = 8.05$, see Fig. 9). Note that at this time, no temporal tasks are specified for the other agents. Therefore, only the formation input drives
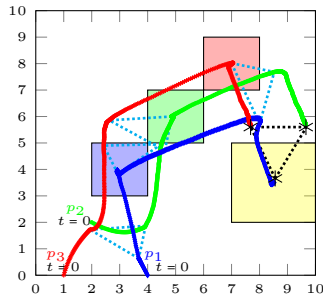
Fig. 8. Agents' trajectories satisfying ϕ_3 with agents staying in the center of assigned regions as long as possible. Triangles in cyan show the formation at $t = 5$, $t = 13$, $t = 19$ and $t = 28$ formed due to u_{f_i} while the agents meet their temporal requirements, and the black triangle shows the final formation at $t = 45$.
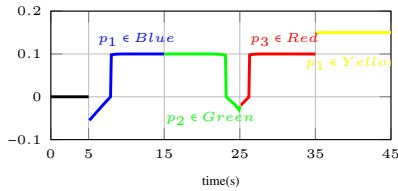

Fig. 9. Scores related to each agent visiting assigned regions for the specified time interval satisfying ϕ_3. Scores for each region are colored with its same color, and positive score means the agent is inside the region.

these agents to form a triangle. The same argument holds for $Agent2$ within $[15, 25]$ and $Agent3$ within $[25, 35]$. Within $[35, 40]$, $Agent1$ has to visit $Yellow$, and stay there for at least 5 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 (4). Fig. 9 shows the scores related to each agent visiting the assigned regions for the specified time interval. During the first 5 seconds, only formation input is applied (no temporal tasks assigned). At $t = 8.05$, $Agent1$ enters $Blue$ and stays in its center until $t = 15$, meanwhile, the formation input drives the other agents to move together to form a triangle. $Agent2$ is in the $Green$ region at $t = 15$ and stays there until $t = 23.1$, $Agent3$ enters Red at $t = 26.35$ and stays there until $t = 35$, and $Agent1$ visits $Yellow$ at $t = 35$ and stays there, and the desired formation is achieved.

VII. CONCLUSION AND FUTURE WORK

We presented a novel robustness score for continuous-time STL, which uses arithmetic and geometric integral means. We demonstrated that this score incorporates the 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. In future work, we will investigate how to combine our robustness score with Control Barrier Functions to find closed form control inputs.

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.