2021 60th IEEE Conference on Decision and Control (CDC)
December 13-15, 2021. Austin, Texas

Neural Network-based Control for Multi-Agent Systems from Spatio-Temporal Specifications

Suhail Alsalehi, Noushin Mehdipour, Ezio Bartocci and Calin Belta

2021 60th IEEE Conference on Decision and Control (CDC) | 978-1-6654-3659-5/21/$31.00 ©2021 IEEE | DOI: 10.1109/CDC45484.2021.9682921

***Abstract*— We propose a framework for solving control synthesis problems for multi-agent networked systems required to satisfy spatio-temporal specifications. We use Spatio-Temporal Reach and Escape Logic (STREL) as a specification language. For this logic, we define smooth quantitative semantics, which captures the degree of satisfaction of a formula by a multi-agent team. We use the novel quantitative semantics to map control synthesis problems with STREL specifications to optimization problems and propose a combination of heuristic and gradient-based methods to solve such problems. As this method might not meet the requirements of a real-time implementation, we develop a machine learning technique that uses the results of the off-line optimizations to train a neural network that gives the control inputs at current states. We illustrate the effectiveness of the proposed framework by applying it to a model of a robotic team required to satisfy a spatial-temporal specification under communication constraints.**

I. Introduction

Multi-agent systems are used as models in many applications, ranging from robotics to power networks, smart cities, and synthetic biology. Planning and controlling the motions of multi-agent systems are difficult problems, which have received a lot of attention in recent years. One of the main challenges is specifying their motions. Existing approaches include consensus algorithms [1], [2], in which the specification is to reach a desired global state (e.g., minimum / maximum inter-robot separation, specified centroid, heading alignment, etc.) and abstractions, in which a team is parameterized by a set of features (e.g., mean, variance, orientation), [3], [4].

Recently, spatio-temporal logics have emerged as formal ways to specify both spatial and temporal logic requirements for spatially distributed systems [5], [6]. Examples include Spatial Temporal Logic (SpaTeL) [7], Spatial Aggregation Signal Temporal Logic (SaSTL) [8], and Spatio-Temporal Reach and Escape Logic (STREL) [9], [10].

In this work, we employ STREL to specify complex spatio-temporal requirements for multi-agent teams. STREL extends STL with the spatial operators *reach*, *escape* and other derived spatial modalities; and it allows for specifying requirements for individual agents in teams. For example, the requirement "the agents must always surround the target in the time interval $[a, b]$" can be specified using the STREL formula $\varphi = G_{[a,b]}(agents \odot target)$ (reads *always* in time interval $[a, b]$ agents *surround* target). The original STREL semantics, as defined in [9], has no notion of spatial counting, which can be critical in some multi-agent robotic scenarios [11]. For instance, one might want to maximize the number of agents that surround a target. Furthermore, the original semantics does not account for the distance variability between agents, which can be critical to connectivity among agents in the team. To overcome these setbacks, we propose new counting quantitative STREL semantics that allows for spatial counting and depends on the distances among agents.

Similar to STL, STREL is equipped with quantitative semantics (robustness function) which quantify the degree of satisfaction of a formula by a (temporal and spatial) trajectory of a system, and allow for mapping control problems into optimization problems. Such optimization problems are often solved using Mixed Integer Linear Program (MILP) encodings [12], [5], [13] or gradient-based methods [14], [**?**], [15], [16], [17]. However, MILP encodings are complicated with unpredictable performance times and gradient-based methods are prone to premature convergence. We propose using a hybrid optimization method that combines heuristic and gradient-based algorithms [18], [19] where optimization is carried out in two stages: 1) find a good candidate solution using a heuristic algorithm and 2) improve the candidate solution using a gradient based algorithm.

Generating control inputs by solving optimization problems can be computationally expensive, and not amenable for real-time control. Thus, we propose an approach to real-time control by training Recurrent Neural Networks (RNN) [20], [21] from state-control trajectories generated off-line. Once trained, the RNN-based controller gives the control inputs based on the current state and the history states.

The main contributions of this work can be summarized as follows. First, we introduce novel, smooth counting quantitative semantics for STREL, which allows for optimizing spatial configurations. Second, we propose, for the first time to the best of our knowledge, a hybrid optimization approach for solving the control synthesis problem for a multi-agent networked system with spatio-temporal specifications. Third, we provide real-time RNN-based controllers for the control problem stated above.

II. Preliminaries

A. System Dynamics and Connectivity Conditions

Consider a team of N robotic agents labeled from the set $S = \{1, 2, \ldots, N\}$. Each agent $l \in S$ has a state $x_l[k] = (q_l[k], a_l)$ at (discrete) time k, where $q_l[k] \in \mathcal{Q} \subset \mathbb{R}^n$ is its

This work was partially supported by the National Science Foundation under grants IIS-2024606, IIS-1723995 and OIA-2020983 at Boston University and by the Austrian FFG-funded IoT4CPS project at TU Wien.

Suhail Alsalehi, Noushin Mehdipour and Calin Belta are with the Division of Systems Engineering, Boston University, Boston, MA 02215 USA (alsalehi, noushinm, cbelta)@bu.edu

Ezio Bartocci is with TU Wien, Vienna, Austria ezio.bartocci@tuwien.ac.at

dynamical state (e.g., position in space), and $a_l \in \mathcal{A}$ is an attribute that does not change over time (e.g., agent type), where $\mathcal{A}$ is a set of labels. The state of the team at time k is denoted by $x[k] = [x_1[k]^T, \ldots, x_N[k]^T]^T$. We assume that the dynamics of each agent $l \in S$ is given by:

$$q_l[k+1] = f_l(q_l[k], u_l[k]), \tag{1}$$

where $u_l[k] \in \mathcal{U} \subset \mathbb{R}^m$ is the control input for agent l at time k, $\mathcal{U}$ is the set of admissible controls, and $f : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^n$. The control inputs of the team at time k are denoted by $u[k] = [u_1[k]^T, \ldots, u_N[k]^T]^T$.

Example 1. Consider a team of $N = 8$ robotic agents, e.g. Mobile Ad-hoc sensor NETwork (MANET)[1], moving in a $2D$ Euclidean space (see Fig.1). The state of agent $l \in S = \{1, .., 8\}$ at time k is $x_l[k] = (q_l[k], a_l)$, where $q_l \in \mathbb{R}^2$ is the position of agent l, and $a_l \in \mathcal{A} = \{endDevice, coordinator, router\}$. Specifically, $a_3 = a_5 = a_6 = endDevice$, $a_1 = a_7 = a_8 = coordinator$ and $a_4 = a_2 = router$. □

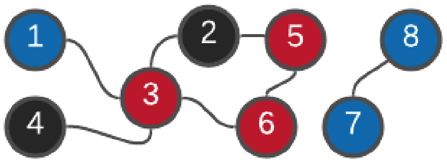


Fig. 1: A team of $N = 8$ robotic agents with types *endDevice* (red), *router* (black), and *coordinator* (blue). The lines are edges in the connection graph.

For $H \in \mathbb{N}$, we use $\mathbf{x}_l^H$ to denote the *state run* $x_l[0]x_l[1]\ldots x_l[H]$ of agent l, which is given by the dynamical state sequence $q_l[0]q_l[1]\ldots q_l[H]$ generated by the control sequence $u_l[0]\ldots u_l[H-1]$ starting at $q_l[0]$, along with the agent attribute a_l. We denote the state run of the team $x[0]x[1]\ldots x[H]$ by $\mathbf{x}^H$. Similarly, we use $\mathbf{u}^{H-1}$ to denote the control sequence of the team $u[0]\ldots u[H-1]$.

Two agents l and l' can communicate at time k if they are connected according to given conditions. For instance, agents may be connected if the Euclidean distance between them is less than a fixed communication range. We model the time-dependent inter-agent connectivity using an undirected graph (connection graph) $\lambda[k] = (S, E[k])$, where $E[k] \subseteq S \times S$. Specifically, $(l, l') \in E[k]$ means that l and l' are connected. We use $\boldsymbol{\lambda}^H$ to denote the sequence of connection graphs $\lambda[0]\lambda[1]\ldots\lambda[H]$. A *route* $\tau = \tau_1\tau_2\ldots$ is a path on a connection graph with τ_i representing the label of the agent at position $i \in \mathbb{Z}_{\geq 0}$ on route τ. The function $\boldsymbol{\tau}(l) : S \to \mathbb{Z}_{>0} \cup \varnothing$ returns the position of agent l on route τ if l is on τ and returns $\varnothing$ otherwise. Finally, the set of routes on the connection graph $\lambda[k]$ that start at agent l is denoted by $Routes(\lambda[k], l)$ where $\forall \tau \in Routes(\lambda[k], l), \tau_1 = l$.

For notational simplicity, we drop the superscript H from $\boldsymbol{\lambda}^H$, $\boldsymbol{x}^H$ and x_l^H when H is clear from context. The discrete time interval $[a, b] \cap \mathbb{Z}$ with $a, b \in \mathbb{Z}_{\geq 0}$ is denoted by $[a, b]$.

B. Spatio-Temporal Reach and Escape Logic (STREL)

STREL is a logic capable of describing complex behaviors in multi-agent teams. Formal definitions of the STREL syntax and semantics can be found in [9]. Here, we give informal definitions for the syntax and the qualitative semantics. We provide, however, a formal definition of the quantitative semantics because we will refer to it later in the paper. Informally, STREL formulas are formed using atomic propositions p, which in this paper are attributes from the set $\mathcal{A}$, or predicates $\mu_{g(q_l[k])\sim r}$ defined over the dynamical states of the agents, where $g : \mathcal{Q} \to \mathbb{R}$, $r \in \mathbb{R}$, and $\sim \in \{\leq, >\}$; logical operators $(\neg, \vee, \wedge)$; temporal operators: *eventually* $(F_{[a,b]})$ and *always* $(G_{[a,b]})$; and spatial operators: *reach* $(\mathcal{R}^f_{\leq d})$, *escape* $(\mathcal{E}^f_{>d})$ and *surround* $(\odot^f_{\leq d})$, where f is a distance function such as Euclidean distance $dist(l, l')$ or graph geodesic $hops(l, l')$ (number of edges in a shortest path between nodes l, l'), and d is a scalar.

The qualitative semantics define the satisfaction of a given STREL formula for $(\boldsymbol{\lambda}, x_l)$ at time k. For instance, $(\boldsymbol{\lambda}, x_l[k]) \models p$ ($\models$ reads "satisfies") if the attribute of agent l is $a_l = p$; $(\boldsymbol{\lambda}, x_l[k]) \models \mu_{g(q_l[k])\sim r}$ if $g(q_l[k]) \sim r$; $(\boldsymbol{\lambda}, x_l[k]) \models F_{[a,b]}\varphi$ if $\exists k' \in [k+a, k+b]$ such that $(\boldsymbol{\lambda}, x_l[k']) \models \varphi$; and $(\boldsymbol{\lambda}, x_l[k]) \models G_{[a,b]}\varphi$ if $\forall k' \in [k+a, k+b] : (\boldsymbol{\lambda}, x_l[k']) \models \varphi$. For the spatial operator *reach*, $(\boldsymbol{\lambda}, x_l[k]) \models \varphi_1 \mathcal{R}^f_{\leq d} \varphi_2$ if φ_2 is satisfied at agent l' reachable from l through a route τ with $\tau_1 = l$ such that $f(l, l') \leq d$, and φ_1 is satisfied at l and all the other agents between l, l' on τ and we call such a route τ a *satisfying route*. Similarly, $(\boldsymbol{\lambda}, x_l[k]) \models \mathcal{E}^f_{>d}\varphi$ if there exists a route τ with $\tau_1 = l$; and an agent $l' \in \tau$ such that $f(l, l') > d$, and φ is satisfied at all agents $\tau_1\tau_2\ldots\tau_{\boldsymbol{\tau}(l')-1}$. The operator *surround* expresses the notion of an agent with state that satisfies φ_1 being surrounded by agents with states satisfying φ_2 within a distance $f \leq d$. It can be derived from operators *reach* and *escape* as[2] $\varphi_1 \odot^f_{\leq d} \varphi_2 = \varphi_1 \wedge \neg\Big(\varphi_1 \mathcal{R}^f_{\leq d} \neg(\varphi_1 \vee \varphi_2)\Big) \wedge \mathcal{E}^f_{>d}\varphi_1 \wedge \varphi_1 \mathcal{R}^f_{\leq d}\varphi_2$. Additional operators can be derived from the operators above and are omitted here for brevity.

The *time horizon* $hrz(\varphi)$ of a STREL formula φ is the smallest time point in future for which the states of agents are needed to determine the satisfaction of the formula.

The quantitative valuation of a given STREL formula φ is defined by a real-valued *robustness function* ρ. The robustness ρ of a STREL formula φ with respect to $(\boldsymbol{\lambda}, x_l)$ at time k is calculated recursively by [9]:

$$\rho(\boldsymbol{\lambda}, x_l, p, k) = \iota(p, x_l[k]) \tag{2a}$$
$$\rho(\boldsymbol{\lambda}, x_l, \mu, k) = \iota(\mu, x_l[k]) \tag{2b}$$
$$\rho(\boldsymbol{\lambda}, x_l, \neg\varphi, k) = -\rho(\boldsymbol{\lambda}, x_l, \varphi, k) \tag{2c}$$

[1] MANET is a team of robotic agents of different types connected wirelessly and can move independently.

[2] We added the term $(\wedge\varphi_1\mathcal{R}^f_{\leq d}\varphi_2)$ to the definition provided in [9] to avoid false satisfaction in the case of isolated agents satisfying φ_1 while not being surrounded by agents satisfying φ_2.

$$\rho(\boldsymbol{\lambda}, x_l, \varphi_1 \vee \varphi_2, k) = \max(\rho(\boldsymbol{\lambda}, x_l, \varphi_1, k), \rho(\boldsymbol{\lambda}, x_l, \varphi_2, k)) \tag{2d}$$

$$\rho(\boldsymbol{\lambda}, x_l, F_{[a,b]}\varphi, k) = \max_{k' \in [k+a, k+b]} \rho(\boldsymbol{\lambda}, x_l, \varphi, k') \tag{2e}$$

$$\rho(\boldsymbol{\lambda}, x_l, G_{[a,b]}\varphi, k) = \min_{k' \in [k+a, k+b]} \rho(\boldsymbol{\lambda}, x_l, \varphi, k') \tag{2f}$$

$$\rho(\boldsymbol{\lambda}, x_l, \varphi_1 \mathcal{R}^f_{\leq d} \varphi_2, k) = \max_{\tau \in Routes(\lambda[k], l)} \max_{l' \in \tau : f(l,l') \leq d} \left[\min(\rho(\boldsymbol{\lambda}, x_{l'}, \varphi_2, k); \min_{j < \boldsymbol{\tau}(l')} \rho(\boldsymbol{\lambda}, x_{\tau_j}, \varphi_1, k)) \right] \tag{2g}$$

$$\rho(\boldsymbol{\lambda}, x_l, \mathcal{E}^f_{>d}\varphi, k) = \max_{\tau \in Routes(\lambda[k], l)} \max_{l' \in \tau : f(l,l') \leq d} \min_{j < \boldsymbol{\tau}(l')} \rho\left(\boldsymbol{\lambda}, x_{\tau_j}, \varphi, k\right) \tag{2h}$$

where ι is the *signal interpretation function* defined for atomic propositions and predicates by

$$\iota(p, x_l[k]) = \begin{cases} \rho_{max}, & \text{if } a_l = p \\ -\rho_{max}, & \text{otherwise} \end{cases} \tag{3}$$

$$\iota(\mu_{g(q_l[k]) \sim r}, x_l[k]) = \begin{cases} g(q_l[k]) - r, & \text{if } \sim = > \\ r - g(q_l[k]), & \text{if } \sim = \leq \end{cases} \tag{4}$$

Theorem 1. *[9] The STREL robustness defined by* (2) *is sound, i.e., positive robustness indicates satisfaction, and negative robustness indicates violation of the specification.*

C. Smooth Robustness Approximation

The $\max$ and $\min$ functions can be approximated by [14]:

$$\max(a_1, ..., a_n) \approx \widetilde{\max}(a_1, ..., a_n) = \tfrac{1}{\beta} \ln\left(\textstyle\sum_{i=1}^n e^{\beta a_i}\right),$$
$$\min(a_1, ..., a_n) \approx \widetilde{\min}(a_1, ..., a_n) = -\widetilde{\max}(-a_1, ..., -a_n)$$

and the approximation error is bounded by ϵ_β:

$$0 \leq \max(a_1, .., a_n) - \widetilde{\max}(a_1, .., a_n) \leq \tfrac{\ln(n)}{\beta} = \epsilon_\beta \tag{5}$$

III. Problem Formulation and Approach

Consider a team of N agents labeled from the set $S = \{1, \ldots, N\}$ with dynamics (1) and a cost function $J(u[k], x[k+1])$, which is the cost of ending in state $x[k+1]$ by applying the control inputs $u[k]$ at time k. Assume that spatio-temporal requirements are specified by a STREL formula φ interpreted over the state run of the team $\mathbf{x}^H$, and $H = hrz(\varphi)$. We aim to find a control sequences for agents to optimize the objective function.

Problem 1. *[Control Synthesis] Given a multi-agent team with dynamics (1), STREL formula φ, initial connection graph $\lambda[0]$, initial state of the system $x[0]$, planning horizon $H = hrz(\varphi)$, and cost function J; find an optimal control sequence $\mathbf{u}^{*H-1}$ that maximizes the robustness score and minimizes the cost, i.e.:*

$$\begin{aligned} \mathbf{u}^{*H-1} = \underset{\mathbf{u}^{H-1}}{\arg\max}\, & \rho_c(\boldsymbol{\lambda}, \boldsymbol{x}, \varphi, 0) - \gamma \sum_{k=0}^{H-1} J(u[k], x[k+1]) \\ s.t. \quad & \\ & q_l[k+1] = f(q_l[k], u_l[k]), \quad \forall l \in S, \forall k \in [0, H-1] \\ & u_l[k] \in \mathcal{U}, \qquad \forall l \in S, \forall k \in [0, H-1] \end{aligned} \tag{6}$$

where $\gamma > 0$ is a trade-off coefficient and $\rho_c(\boldsymbol{\lambda}, \boldsymbol{x}, \varphi, 0)$ is the robustness function for the team at time 0*.*

To solve Problem 1, we first introduce sound *counting STREL quantitative semantics* with robustness ρ_c, which has spatial counting and allows to optimize the spatial configuration for connectivity (Sec. IV). We use a smooth version of the robustness $\tilde{\rho}_c$ in the objective function, and employ a combination of heuristic and gradient-based optimization algorithms (Sec. V) to find a control sequence that maximizes the robustness of the STREL formula and minimizes the cost function. In addition, as the execution time for the optimization might not meet real-time control requirements, we propose training a RNN to learn controllers from state-control trajectories generated by solving the optimization problem with different initializations (Sec. VI). The trained RNN is then used to predict control inputs at each time.

IV. STREL Counting Quantitative Semantics

To motivate the new proposed STREL quantitative semantics, we start by discussing the limitations of the original semantics defined by the robustness function (2) in [9].

First, robustness function (2) does not depend on the distance between the agents. In practice, connectivity between robotic agents in a networked system depends on the spatial configuration and it is often the case that the smaller the distance between agents, the better the connectivity. Second, it does not have a notion of spatial counting, which can be beneficial to maximize the spatial satisfaction of a given formula. Third, it is defined only at the level of individual agents. We need a way to compute the robustness score for the team, which takes into account the number of agents that satisfy/violate a given formula.

Next, we introduce new STREL quantitative semantics *(counting robustness)* that addresses these limitations and differs from the original semantics defined in [9] in three ways 1) the proposed semantics depends on the distances among agents, 2) it performs spatial counting of satisfying/violating routes at individual agents, and 3) it captures the satisfaction degree of the team.

A. Optimizing the Spatial Configuration for Connectivity

To optimize the spatial configuration of a multi-agent team for connectivity, we require the robustness score to depend on the distance between agents. To this end, we introduce a function σ_{dist}, which depends on the distance between agents $f(l, l')$ and a scalar d. Specifically, σ_{dist} is a sigmoid function that take values between $[-1, 1]$ depending on the ratio $d_{norm} = \frac{f(l,l')}{d}$ and is defined for $f(l, l') \leq d$ and $f(l, l') > d$ as follows:

$$\sigma^{\leq}_{dist}(d_{norm}) = -\tanh(k_d(d_{norm} - 1)) \tag{7}$$

$$\sigma^{>}_{dist}(d_{norm}) = \tanh(k_d(d_{norm} - 1)) \tag{8}$$

where k_d is a hyperparameter.

When computing the robustness, we take $\min(\sigma_{dist}, \rho_c)$ (see Sec. IV). Notice that σ_{dist} allows the robustness score to vary beyond the distance constant d as defined by $f(l, l') \sim d$, $\sim = \{\leq, >\}$ as opposed to the original definition in (2).

B. Spatial Counting for Routes

Consider the robustness function of the spatial operator $escape$ defined by (2h) and define the robustness of a given route τ as $\rho_\tau := \max_{l' \in \tau : f(l,l') \leq d} \min_{j < \boldsymbol{\tau}(l')} \rho\left(\boldsymbol{\lambda}, x_{\tau_j}, \varphi, k\right)$. Notice that $\rho\left(\boldsymbol{\lambda}, x_l, \varphi, k\right) = \max_{\tau \in Routes(\lambda[k], l)} \rho_\tau,$, which means that it is enough to have one satisfying route $\tau \in Routes(\lambda[k], l)$ (with $\rho_\tau > 0$) to satisfy formula φ and the robustness score does not vary depending on the number of routes that satisfy/violate φ. We address this limitation by introducing an additional function σ_{routes}, which depends on the number of routes that satisfy/violate a given formula.

Formally, let $R^+, R^- \in \mathbb{N}$ be the number of satisfying/violating routes for a given spatial operator, respectively. We define the function σ_{routes}

$$\sigma_{routes} \quad = \quad \max\Big(\frac{1}{1+e^{k_R R^-}}, \frac{1}{1+e^{-k_R(R^+ - R^-)}}\Big) \tag{9}$$

where k_R is a hyperparameter.

C. Spatial Counting for Agents

As mentioned above, the STREL semantics is defined at the level of individual agents. A naïve method to compute the robustness of the team of N agents is to consider the minimum of the robustness of individual agents.

$$\rho\left(\boldsymbol{\lambda}, \boldsymbol{x}, \varphi, k\right) = \min_{l \in S} \rho\left(\boldsymbol{\lambda}, x_l, \varphi, k\right) \tag{10}$$

In this case, the robustness score for the team will reflect the worst robustness score among individual agents and will not depend on the number of agents that satisfy/violate the formula φ. Following a similar approach to route counting in Sec. IV-B, we introduce $\sigma_{ag}(Ag^+, Ag^-)$, which allows for varying the robustness score depending on the number of agents that satisfy/violate the specification.

Let Ag^+, Ag^- be the number of agents that satisfy and violate the specification, respectively. Then σ_{ag} is given by:

$$\sigma_{ag} \;=\; \max\Big(\frac{1}{1+e^{-k_{ag} Ag^-}}, \frac{1}{1+e^{k_{ag}(Ag^- - Ag^+)}}\Big) \tag{11}$$

D. Counting Robustness for STREL

We are now present the proposed counting robustness for the STREL spatial operators[3].

$$\begin{aligned}
\rho_c\left(\boldsymbol{\lambda}, x_l, \varphi_1 \mathcal{R}^f_{\leq d} \varphi_2, k\right) = \min \Bigg[& \sigma_{routes}(R^+, R^-) \\
& \max_{\tau \in Routes} \max_{l' \in \tau : dist(l') \leq d} \min(\rho_c\left(\boldsymbol{\lambda}, x_{l'}, \varphi_2, k\right); \\
& \min_{j < \boldsymbol{\tau}(l')} \rho_c\left(\boldsymbol{\lambda}, x_{\tau_j}, \varphi_1, k\right)); \sigma^{\leq}_{dist}\left(d_{norm}\right) \Bigg]
\end{aligned}$$

$$\begin{aligned}
\rho_c\left(\boldsymbol{\lambda}, x_l, \mathcal{E}^f_{>d} \varphi, k\right) = \min \Bigg[& \sigma_{routes}(R^+, R^-) \\
& \max_{\tau \in Routes} \max_{l' \in \tau : > d} \min_{j < \boldsymbol{\tau}(l')} \rho_c\left(\boldsymbol{\lambda}, x_{\tau_j}, \varphi, k\right)); \sigma^{>}_{dist}\left(d_{norm}\right) \Bigg]
\end{aligned} \tag{12}$$

The robustness for the team at time k is given by:

$$\rho_c(\boldsymbol{\lambda}, \boldsymbol{x}, \varphi, k) \quad = \quad \sigma_{ag} \min_{l \in \{1, \ldots, N\}} (\rho_c(\boldsymbol{\lambda}, x_l, \varphi, k)). \tag{13}$$

Theorem 2. *The counting robustness of STREL defined by* (12) *and* (13) *is sound.*

Proof. [sketch] A formal proof is omitted due to space constraints. Informally, soundness can be viewed as a sign consistency between the counting robustness ρ_c and the original robustness in [9] ρ. We show that the three functions σ_{dist}, σ_{routes}, σ_{ag} introduced to ρ do not affect the sign of the robustness, and thus show that ρ_c is sound.

First, σ_{routes} and σ_{ag} are positive and are multiplied by the robustness function provided by the original semantics and, thus, do not change the sign of the robustness score. Second, σ_{dist} changes in the range $[-1, 1]$ and it is negative only when the distance predicate $f(l, l') \sim d$ is violated. Since we take the minimum between the robustness function and σ_{dist}, the robustness function will still give positive values for satisfaction and negative values for violation as before. Thus, the soundness of the proposed counting robustness follows from the soundness of the original robustness. ■

V. Control Synthesis for STREL Specifications

Problem 1 is a constrained non-linear optimization problem. The max and min functions in the semantics render the objective function non-differentiable. We modify (6) by replacing the counting robustness ρ_c with its smooth version of proposed smooth robustness $\tilde{\rho_c}$ that we obtain by replacing the non-differentiable terms (min/max) with their smooth approximations described in Sec. II-C.

We solve the new problem by employing a two-stage optimization utilizing heuristic and gradient-based algorithms. In stage I, the search space is explored using a heuristic algorithm to find a good candidate solution. In stage II, a gradient-based algorithm is employed to improve the candidate solution from stage I. In this paper, we will use Particle Swarm Optimization (PSO) [22] and Sequential Quadratic Programming (SQP) [23] for stages I and II respectively.

VI. Learning RNN-Based Controllers

As already mentioned, solving the control synthesis problem by optimization can be expensive, and not feasible for real-time implementation. To address this, we propose to train a RNN using data obtained from off-line optimization, and then use it to generate control inputs at a given state.

Dataset generation. Given a multi-agent team as described in Sec. II-A, a STREL formula φ, a planning horizon $H \geq hrz(\varphi)$, a set of M state initializations $\{x[0]^{[1]}, \ldots, x[0]^{[M]}\}$ and their corresponding communication graphs $\{\lambda[0]^{[1]}, \ldots, \lambda[0]^{[M]}\}$, we generate a dataset D by solving the control synthesis problem described in Sec. V and choosing $m \leq M$ state-control trajectories with robustness above a given threshold ϵ_{min}, i.e

$$D = \{(\mathbf{x}^H_{[j]}, \mathbf{u}^{H[j]}); \tilde{\rho_c}^{[j]} \geq \epsilon_{min}\}$$

[3] Robustness for logical and temporal operators remains unchanged (2).

, where $\epsilon_{min} \geq \epsilon_\beta$ is the robustness margin used to account for the approximation error ϵ_β.

RNN implementation. The satisfaction of STREL formulas is history-dependent, due to the existence of temporal operators. In other words, the control at each time step is, in general, dependent on the current state and past states $u[k] = g(x[0], .., x[k])$. For this reason, we choose Recurrent Neural Networks (RNN), which are neural networks with memory. To implement the RNN, we use a Long Short Term Memory (LSTM) Network [20], [24]. LSTM has feedback channels and memory cells that can manage long-term dependencies by passing the history-dependence as hidden states. The function g can be approximated as follows:

$$\begin{aligned} h[k] &= \mathcal{R}(X[k], h[k-1], W_1) \\ \hat{u}[k] &= \mathcal{N}(h[k], W_2) \end{aligned} \tag{14}$$

where W_1, W_2 are the weight matrices of the RNN, $h[k]$ is the hidden state at time step k and $\hat{u}[k]$ is the predicted control at k. The network is trained to minimize the error between the predicted control and the optimized control given in the dataset:

$$\min_{W_1, W_2} \sum_{D} \sum_{k=0}^{H-1} \|u[k] - \hat{u}[k]\|^2 . \tag{15}$$

VII. Case Study: Networked Robotic Agents

In this section, we demonstrate the efficacy of our proposed framework with a case study. First, we solve the control synthesis problem for a multi-agent team and a given STREL formula off-line to generate state-control trajectories using the optimization approach described in Sec. V. Next, we use the generated state-control trajectories with robustness above a given threshold to train a RNN to predict control inputs for the team at each time step (Sec. VI). Finally, we discuss the obtained results for different solvers (PSO, SQP, and PSO+SQP) (Sec. V); and compare the performance under the original ((2), (10)) and our proposed semantics ((12), (13)).

System description Consider a team of $N = 7$ robotic agents (Fig.2) labeled from the set $S = \{1, 2, \ldots, 7\}$ in a 2D Euclidean space. The state of agent l at time step k is defined by $x_l[k] = (q_l[k], a_l)$ where $q_l[k] \in \mathbb{R}^2$ is the position of agent l at time k and $a_l \in \mathcal{A} = \{endDevice, coordinator, router\}$ is the type of the agent with $a_1 = a_2 = endDevice$, $a_4 = a_5 = a_7 = router$ and $a_3 = a_6 = coordinator$. Agents 1 and 2 are controllable with dynamics given by

$$q_l[k+1] = q_l[k] + u_l[k], \tag{16}$$

where $l \in \{1, 2\}$ and $u_l \in \mathcal{U} = [-0.2, 0.2]$.

Connectivity conditions Two agents l and l' are connected at time k if a) the Euclidean distance between agents l and l' is less than a fixed communication range $dist(l, l') \leq 2$, and b) in the corresponding Voronoi diagram, the cells corresponding to agents l and l' are adjacent at time k.

Snapshots of the team and the corresponding Voronoi and connection graphs at different times can be seen in Fig.2.

Spatio-temporal specifications Agents of type $endDevice$ must reach the center (circle with radius $= 0.4$) in time interval $[12, 13]$, while staying connected to at least one agent of type $router$ at all times in $[3, 13]$. To avoid collision, all agents must keep a distance of at least 0.15 from each other at all times. The STREL formula φ captures these requirements, with $hrz(\varphi) = 13$:

$$\begin{aligned} \varphi = G_{[0,13]}&(dist_{i \neq j}(q_i[k], q_j[k]) > 0.15) \\ &\bigwedge G_{[3,13]} endDevice \mathcal{R}^{dist}_{\leq 2} router \\ &\bigwedge F_{[12,13]}(dist(q_{i,endDevice}[k], origin) \leq 0.4). \end{aligned} \tag{17}$$

Control problem *Given* the team of $N = 7$ agents with dynamics (16), STREL formula φ ((17)), initial connection graph $\lambda[0]$, initial state of the system $x[0]$, planning horizon $H = hrz(\varphi) = 13$, *find* an optimal control sequence $\mathbf{u}^{*H-1}$ that solves Problem 1, where in (6), the cost function

$$J\left(u[k], x[k+1]\right) = \sum_{i=1}^{N} \sum_{k=0}^{H-1} \|u_i[k]\|^2 ,$$

$\gamma = 0.01$, and ρ_c is replaced by its smooth version $\tilde{\rho_c}$ (see Sec. V).

Dataset generation We generated over 1100 state-control trajectories with robustness score above a given threshold $\tilde{\rho_c}^* \geq \epsilon_{min} = 0.001 > \epsilon_\beta$ by solving the control synthesis problem off-line. We defined the *success rate* as the percentage of state-control trajectories with robustness $\tilde{\rho_c}^* \geq 0.001$. The success rate, average normalized robustness and computation times for solving the control synthesis problem with the proposed robustness using SQP, PSO and PSO+SQP are presented in Tab. I.

Algorithm	Success rate	Average robustness
SQP	44.7	0.0048
PSO	71.7	0.0037
PSO + SQP	93.8	0.0050

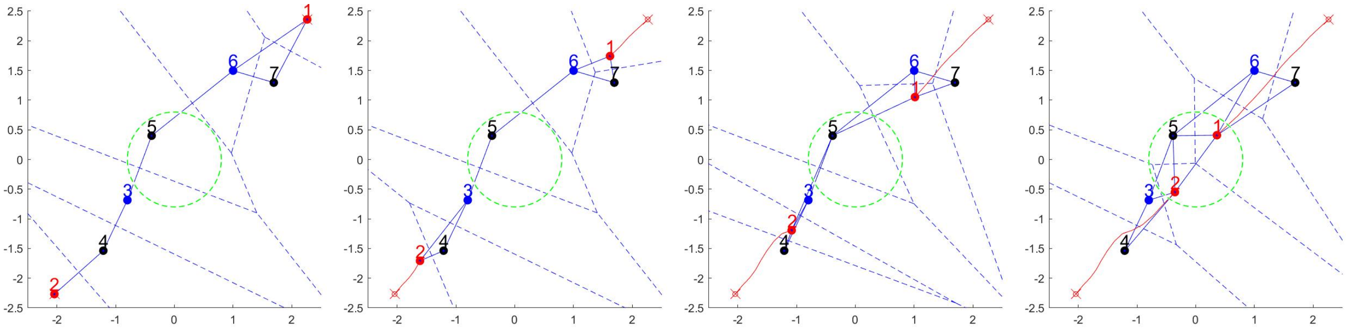
TABLE I: Performance of different optimization methods

Training the RNN We used an LSTM network with four hidden layers (each has 64 nodes) to learn the controllers. We used 850 trajectories for training and 275 trajectories for testing. We trained the network for 700 epochs for 6 minutes.

Results We ran 275 test simulations (from successful initializations[4]) and the average robustness for the trajectories generated using the RNN-based controller is 0.0037 compared to 0.0050 using PSO+SQP. The average execution times per trajectory using the RNN-based controller and PSO+SQP are 0.002 and 32.9 seconds, respectively. The success rate for trajectories generated from new initializations using the RNN-based controller is 93%.

In addition, we used PSO to solve the control synthesis problems with a) the naïve team robustness (10) based on the

[4] These are initializations for which PSO+SQP was able to generate trajectories that satisfy specifications



(a) $k = 0$ (b) $k = 4$ (c) $k = 8$ (d) $k = 12$

Fig. 2: Snapshots of the multi-agent team at different times. The agents are represented by labeled colored disks where labels are taken from the set $S = \{1, \ldots, 7\}$ and colors correspond to types of agents: *endDevice* (red), *router* (black), and *coordinator* (blue). The red solid lines represent the trajectories generated by agents 1,2 using control inputs from the RNN-based controller. The solid and dashed blue lines represent the connection and Voronoi graphs of the team, respectively.

original agent robustness (2), and b) the proposed robustness (12),(13) in the objective function. We solved each one with 100 initializations and obtained no satisfying state-control trajectories for the original robustness and 69 satisfying state-control trajectories for the proposed robustness. We explain the results by noting that the proposed robustness has a smooth search space (compared to the original robustness), which helps to avoid premature convergence. The obtained results demonstrate the efficacy of the trained RNN for real-time control.

VIII. Conclusions and Future Research

We proposed a framework for solving control synthesis problems for multi-agent networked systems from spatio-temporal specifications. We introduced new counting quantitative semantics for the Spatio-Temporal Reach Escape Logic (STREL) and used it to map the control problems to optimization problems. The proposed semantics are sound, smooth, and allow for spatial counting as well as optimizing the spatial configuration of the team for connectivity. To meet real-time control requirements, we learn Recurrent Neural Network (RNN) - based controllers from state-control trajectories generated by solving the optimization problem with different initializations. Future research directions include providing specifications satisfaction guarantees for the RNN-based controller; using reinforcement learning for online control; and scaling the framework to large teams of agents.

References

[1] F. Bullo, J. Cortés, and S. Martinez, *Distributed control of robotic networks: a mathematical approach to motion coordination algorithms*. Princeton University Press, 2009.

[2] M. Mesbahi and M. Egerstedt, *Graph theoretic methods in multiagent networks*. Princeton University Press, 2010, vol. 33.

[3] C. Belta and V. Kumar, "Abstraction and control for groups of robots," *IEEE Trans. on Robotics*, 2004.

[4] R. A. Freeman, Peng Yang, and K. M. Lynch, "Distributed estimation and control of swarm formation statistics," in *ACC*, 2006.

[5] I. Haghighi, S. Sadraddini, and C. Belta, "Robotic swarm control from spatio-temporal specifications," *CDC*, 2016.

[6] Z. Liu, B. Wu, J. Dai, and H. Lin, "Distributed communication-aware motion planning for networked mobile robots under formal specifications," *Trans. on Control of Network Systems*, 2020.

[7] I. Haghighi, A. Jones, Z. Kong, E. Bartocci, R. Grosu, and C. Belta, "Spatel: A novel spatial-temporal logic and its applications to networked systems," *HSCC*, 2015.

[8] M. Ma, E. Bartocci, E. Lifland, J. Stankovic, and L. Feng, "Sastl: spatial aggregation signal temporal logic for runtime monitoring in smart cities," in *ACM/IEEE 11th International Conference on Cyber-Physical Systems (ICCPS)*, 2020.

[9] E. Bartocci, L. Bortolussi, M. Loreti, and L. Nenzi, "Monitoring mobile and spatially distributed cyber-physical systems," *International Conference on Formal Methods and Models for System Design*, 2017.

[10] E. Bartocci, L. Bortolussi, M. Loreti, L. Nenzi, and S. Silvetti, "Moonlight: A lightweight tool for monitoring spatio-temporal properties," in *International Conference on Runtime Verification*, 2020.

[11] Y. E. Sahin, P. Nilsson, and N. Ozay, "Multirobot coordination with counting temporal logics," *IEEE Trans. Rob.*, 2020.

[12] V. Raman, M. Maasoumy, A. Donze, R. M. Murray, A. Sangiovanni-Vincentelli, and S. A. Seshia, "Model predictive control with signal temporal logic specifications," *CDC*, 2014.

[13] Z. Liu, B. Wu, J. Dai, and H. Lin, "Distributed communication-aware motion planning for multi-agent systems from stl and spatel specifications," *CDC*, 2018.

[14] Y. V. Pant, H. Abbas, and R. Mangharam, "Smooth operator: Control using the smooth robustness of temporal logic," *CCTA*, 2017.

[15] P. Varnai and D. V. Dimarogonas, "On robustness metrics for learning stl tasks," in *ACC*, 2020.

[16] N. Mehdipour, C. Vasile, and C. Belta, "Arithmetic-geometric mean robustness for control from signal temporal logic specifications," 2019.

[17] Y. Gilpin, V. Kurtz, and H. Lin, "A smooth robustness measure of signal temporal logic for symbolic control," *IEEE Control Systems Letters*, 2021.

[18] M. Mavrovouniotis, C. Li, and S. Yang, "A survey of swarm intelligence for dynamic optimization: Algorithms and applications," *Swarm and Evolutionary Computation*, 2017.

[19] T. A. Victoire and A. Jeyakumar, "Hybrid pso–sqp for economic dispatch with valve-point effect," *Electric Power Systems Research*, 2004.

[20] W. Liu, N. Mehdipour, and C. Belta, "Recurrent neural network controllers for signal temporal logic specifications subject to safety constraints," *IEEE Control Systems Letters*, 2021.

[21] S. Yaghoubi, G. Fainekos, and S. Sankaranarayanan, "Training neural network controllers using control barrier functions in the presence of disturbances," *ITSC*, 2020.

[22] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proceedings of ICNN'95-international conference on neural networks*. IEEE, 1995.

[23] E. Polak, *Optimization: algorithms and consistent approximations*. Springer Science & Business Media, 2012.

[24] S. Yaghoubi and G. Fainekos, "Worst-case satisfaction of stl specifications using feedforward neural network controllers: A lagrange multipliers approach," *ACM Trans. Embed. Comput. Syst.*, 2019.