Cooperative Decentralized Multi-agent Control under Local LTL Tasks and Connectivity Constraints

Meng Guo, Jana Tumova and Dimos V. Dimarogonas

Abstract— **We propose a framework for the decentralized control of a team of agents that are assigned local tasks expressed as Linear Temporal Logic (LTL) formulas. Each local LTL task specification captures both the requirements on the respective agent's behavior and the requests for the other agents' collaborations needed to accomplish the task. Furthermore, the agents are subject to communication constraints. The presented solution follows the automata-theoretic approach to LTL model checking, however, it avoids the computationally demanding construction of synchronized product system between the agents. A decentralized coordination scheme through a dynamic leader selection is proposed, to guarantee the low-level connectivity maintenance and a progress towards the satisfaction of each agent's task.**

I. Introduction

Cooperative control for multi-agent systems have been extensively studied for various purposes like consensus [19], formation [4], [5], and reference-tracking [12], where each agent either serves to accomplish a global objective or fulfil simple local goals such as reachability. In contrast, we focus on planning under complex tasks, such as periodic surveillance (repeatedly perform A), sequencing (perform A, then B, then C), or request-response (whenever A occurs, perform B). In particular, we consider a team of agents modeled as a dynamical system that are assigned a local task specification as Linear Temporal Logic (LTL) formulas. The agents might not be able to accomplish the tasks by themselves and hence requirements on the other agents' behaviors are also part of the LTL formulas.

The goal of this work is to find motion controllers and action plans for the agents that guarantee the satisfaction of all individual LTL tasks. We aim for a decentralized solution while taking into account the constraints that the agents can exchange messages only if they are close enough. Following the hierarchical approach to LTL planning, we first generate for each agent a sequence of actions as a high-level plan that, if followed, guarantees the accomplishment of the respective agent's LTL task. Second, we merge and implement the syntesized plans in real-time upon the run of the system. Namely, we introduce a distributed continuous controller for the leader-follower scheme, where the current leader guides itself and the followers towards the satisfaction of the leader's task. At the same time, the connectivity of the multi-agent system is maintained. By a systematic leader re-election, we ensure that each agent's task will be met in long term.

The authors are with the ACCESS Linnaeus Center, School of Electrical Engineering, KTH Royal Institute of Technology, SE-100 44, Stockholm, Sweden and with the KTH Centre for Autonomous Systems. mengg, tumova, dimos@kth.se. This work was supported by the EU STREP RECONFIG: FP7-ICT-2011-9-600825.

Multi-agent planning under temporal logic tasks has been studied in several recent papers [2], [10], [14], [16]–[18], [20]–[22]. Many of them build on top-down approach to planning, when a single LTL task is given to the whole team. For instance, in [2], [21], the authors propose decomposition of the specification into a conjunction of independent local LTL formulas. On the other hand, we focus on bottom-up planning from individual specification. Related work includes a decentralized control of a robotic team from local LTL specification with communication constraints proposed in [7]. However, the specifications there are local [11] and the agents do not impose any requirements on the other agents' behavior. In [10], the same bottom-up planning problem is considered and a partially decentralized solution is designed that takes into account only clusters of dependent agents instead of the whole group. This approach is later extended in [20], where a receding horizon approach to the problem is suggested. Both mentioned studies however assume that the agents are fully synchronized and the proposed solutions rely on the construction of the synchronized product system between the agents, or at least of its part. In contrast, in this work, we avoid the product construction completely.

The contribution of the paper can be summarized as the proposal of a decentralized motion and action control scheme for multi-agent systems which handles both connectivity constraints and collaborative tasks that are assigned locally. The features of the suggested solution are as follows: (1) the continuous controller is distributed and integrated with the leader election scheme; (2) the distributed leader election algorithm only requires local communications and guarantees sequential progresses towards individual desired tasks; and (3) the proposed coordination scheme operates in real-time, upon the run of the system as opposed to offline solutions that require fully synchronized motions of all agents.

The rest of the paper is organized as follows. In Section II we state the preliminaries. Section III formally introduces the considered problem. In Section IV we describe the proposed solution in details. Section V demonstrates the results in a simulated case study. Finally, we conclude in Section VI.

An extended version of this paper including detailed proofs can be found in [6].

II. Preliminaries

Given a set S, let 2^{S} and S^{ω} denote the set of all subsets of S and the set of all infinite sequences of elements of S.

Definition 1: An LTL formula ϕ over the set of atomic propositions Σ is defined inductively by:

$$\varphi ::= \sigma \mid \neg\varphi \mid \varphi \vee \varphi \mid \mathsf{X}\varphi \mid \varphi \mathsf{U} \varphi \mid \mathsf{F}\varphi \mid \mathsf{G}\varphi,$$

where $\sigma \in \Sigma$, $\neg$ (negation) and $\vee$ (disjunction) are standard Boolean connectives, and X (next), U (until), F (eventually), and G (always) are temporal operators. ■

The semantics of LTL is defined over infinite words over 2^Σ. Intuitively, an atomic proposition $\sigma \in \Sigma$ is satisfied on a word $w = w(1)w(2)\dots$ if it holds at $w(1)$, i.e. if $\sigma \in w(1)$. Formula $\mathsf{X}\,\phi$ holds true if ϕ is satisfied on the word suffix that begins in the next position $w(2)$, whereas $\phi_1 \,\mathsf{U}\, \phi_2$ states that ϕ_1 has to be true until ϕ_2 becomes true. $\mathsf{F}\,\phi$ and $\mathsf{G}\,\phi$ are true if ϕ holds on w eventually and always, respectively. For the formal definition see, e.g. [1].

Given an LTL formula φ over Σ, a word over 2^Σ that satisfies φ can be generated as follows [1]. First, the LTL formula is algorithmically translated into a corresponding Büchi automaton, e.g., by an off-the-shelf tool such as LTL2BA [9]. Second, by finding a finite path (prefix) followed by a cycle (suffix) containing an accepting state, we find a word that is accepted by the Büchi automaton $\mathcal{B}$ and hence satisfies φ.

In this particular work, we are interested only in subsets of 2^Σ that are singletons. Thus, we interpret LTL over words over Σ, i.e. over sequences of Σ instead of subsets of Σ.

III. Problem Formulation

A. Agent Dynamics and Network Structure

Let us consider a team of N agents, modeled by the single-integrator dynamics:

$$\dot{x}_i(t) = u_i(t), \qquad i \in \mathcal{N} = \{1, \cdots, N\}, \tag{1}$$

where $x_i(t)$, $u_i(t) \in \mathbb{R}^2$ are the state and control inputs of agent i at time $t > 0$, $x_i(0)$ is the given initial state, and $\mathbf{x}_i(t)$ is the *trajectory* of agent i from time 0 to $t \geq 0$. We assume that all agents start at time $t = 0$.

Each agent has a limited communication radius of $r > 0$. At time t, agent i can communicate, i.e. exchange messages with agent j if $\|x_i(t) - x_j(t)\| \leq r$. This constraint imposes certain challenges on the distributed coordination of multi-agent systems as the inter-agent communication or information exchange depends on their relative positions.

Agents i and j are *connected* at time t if and only if either $\|x_i(t) - x_j(t)\| \leq r$, or if there exists i', such that $\|x_i(t) - x_{i'}(t)\| \leq r$, where i' and j are connected. Hence, two connected agents can communicate indirectly. We assume that initially, all agents are connected. The particular message passing protocol is beyond the scope of this paper. For simplicity, we assume that message delivery is reliable, meaning that a message sent by agent i will be received by all connected agents j.

B. Task Specifications

Each agent $i \in \mathcal{N}$ is given of a set of M_i different *services* $\Sigma_i = \{\sigma_{ih}, h \in \{1, \cdots, M_i\}\}$ that it is responsible for, and a set of K_i regions, where subsets of these services can be *provided*, denoted by $\mathcal{R}_i = \{R_{ig},\, g \in \{1, \cdots, K_i\}\}$. For simplicity, R_{ig} is determined by a circular area:

$$R_{ig} = \{y \in \mathbb{R}^2 | \|y - c_{ig}\| \leq r_{ig}\} \tag{2}$$

where $c_{ig} \in \mathbb{R}^2$ and r_{ig} are the center and radius of the region, respectively, such that $r_{ig} \geq r_{min} > 0$, for a fixed minimal radius r_{min}. Furthermore, we assume that each region in $\mathcal{R}_{ih}$ is reachable for each agent. Labeling function $L_i : \mathcal{R}_i \to 2^{\Sigma_i}$ assigns to each region R_{ig} the set of services $L_i(R_{ig}) \subseteq \Sigma_i$ that can be provided in there.

Some of the services in Σ_i can be provided solely by the agent i, while others require cooperation with other agents. Formally, agent i is associated with a set of *actions* Π_i that it is capable of *executing*. The actions are of two types:

- action π_{ih} of *providing* the service $\sigma_{ih} \in \Sigma_i$;
- action $\varpi_{ii'h'}$ of *cooperating* with the agent i' in providing its service $\sigma_{i'h'} \in \Sigma_{i'}$.

A service σ_{ih} then takes the following form:

$$\sigma_{ih} = \pi_{ih} \wedge \Big(\bigwedge_{i' \in \mathcal{C}_{ih}} \varpi_{i'ih} \Big), \tag{3}$$

for the set of cooperating agents $\mathcal{C}_{ih}$, where $\emptyset \subseteq \mathcal{C}_{ih} \subseteq \mathcal{N} \setminus \{i\}$. Informally, a service σ_i is provided if the agent's relevant service-providing action and the corresponding cooperating agents' actions are executed at the same time. Furthermore, it is required that at the moment of service providing, the agent and the cooperating agents from $\mathcal{C}_{ih}$ occupy the same region R_{ig}, where $\sigma_{ih} \in L_i(R_{ig})$.

Definition 2 (Trace): A valid trace of agent i is a tuple $trace_i = (\mathbf{x}_i(t), \mathbb{T}_i^A, \mathbb{A}_i, \mathbb{T}_i^S, \mathbb{S}_i)$, where (1) $\mathbf{x}_i(t)$ is a trajectory of agent i; (2) $\mathbb{T}_i^A = t_1, t_2, t_3, \cdots$ is the sequence of time instances when agent i executes actions from Π_i; (3) $\mathbb{A}_i : \mathbb{T}_i^A \to \Pi_i$ represents the sequence of executed actions, both the service-providing and the cooperating ones; (4) $\mathbb{T}_i^S = \tau_1, \tau_2, \tau_3, \cdots$ is a sequence of time instances when services from Σ_i are provided. Note that $\mathbb{T}_i^S$ is a subsequence of $\mathbb{T}_i^A$ and it is equal to the time instances when service-providing actions are executed; (5) $\mathbb{S}_i : \mathbb{T}_i^S \to \Sigma_i$ represents the sequence of provided services satisfying that for all $l \geq 1$, there exists $g \in \{1, \cdots, K_i\}$ such that

(i) $\mathbf{x}_i(\tau_l) \in R_{ig}$, $\mathbb{S}_i(\tau_l) \in L_i(R_{ig})$, and $\mathbb{S}_i(\tau_l) = \sigma_{ih} \Rightarrow \mathbb{A}_i(\tau_l) = \pi_{ih}$;
(ii) for all $i' \in \mathcal{C}_{ih}$, $\mathbf{x}_{i'}(\tau_l) \in R_{ig}$ and $\mathbb{A}_{i'}(\tau_l) = \varpi_{i'ih}$. ■

In other words, the agent i can provide a service σ_{ih} only if (i) it is present in a region R_{ig}, where this service can be provided, and it executes the relevant service-providing action π_{ih} itself, and (ii) all its cooperating agents from $\mathcal{C}_{ih}$ are present in the same region R_{ig} as agent i and execute the respective cooperative actions needed. ■

Definition 3 (LTL Satisfaction): A valid trace $trace_i = (\mathbf{x}_i(t), \mathbb{T}_i^A, \mathbb{A}_i, \mathbb{T}_i^S = \tau_1, \tau_2, \tau_3, \cdots, \mathbb{S}_i : \mathbb{T}_i^S \to \Sigma_i)$, *satisfies* an LTL formula over φ_i, denoted by $trace_i \models \varphi_i$ if and only if $\mathbb{S}_i(\tau_1)\mathbb{S}_i(\tau_2)\mathbb{S}_i(\tau_3)\cdots \models \varphi_i$. ■

Remark 1: Traditionally, LTL is defined over the set of atomic propositions (APs) instead of services [1], where APs represent inherent properties of system states. The LTL formulas are then interpreted over trajectories of systems.In this work, we define LTL formulas over offered services rather than undetachable inherent properties of the system states. The agent is in our case given the option to decide whether a service $\sigma_{ih} \in L(R_{ig})$ is in state $x_i(t) \in R_{ig}$ satisfied or not. However, $\sigma_i \in \Sigma_i$ is never satisfied in state $x_i(t) \in R_{ig}$, such that $\sigma_i \notin L(R_{ig})$. The LTL f are thus interpreted over the sequences of provided services along the trajectories as opposed to the trajectories themselves. ■

C. Problem statement

Problem 1: Given a team of the agents $\mathcal{N}$ subject to dynamics in Eq. 1, synthesize for each agent $i \in \mathcal{N}$: (1) a control input u_i; (2) a time sequence $\mathbb{T}_i^A$; (3) an action sequence $\mathbb{A}_i$, such that the trace $trace_i = (\mathbf{x}_i(t), \mathbb{T}_i^A, \mathbb{T}_i^S, \mathbb{A}_i, \mathbb{S}_i)$ is valid and satisfies the given local LTL task specification φ_i over the set of services Σ_i. ■

IV. Problem Solution

Our approach to the problem involves two steps. In the offline step, we synthesize a high-level plan, i.e. a sequence of services for each of the agents. In the online step, we dynamically switch between the high-level plans through leader re-election. The team follows the leader towards providing its next scheduled service, during which the connectivity is maintained by the proposed continuous controller.

A. Connectivity Graph

Before proposing the continuous control scheme, let us introduce the notion of connectivity graph that will allow us to handle the communication constraints between the agents.

Recall that each agent has a limited communication radius $r > 0$ as defined in Section III-A. Moreover, let $\varepsilon \in (0, r)$ be a given constant, which plays an important role for the edge definition below. In particular, it introduces a hysteresis in the definition for edges in the connectivity graph.

Definition 4: Let $G(t) = (\mathcal{N}, E(t))$ denote the undirected time-varying connectivity graph formed by the agents, where $E(t) \subseteq \mathcal{N} \times \mathcal{N}$ is the set of edges for $t \geq 0$. At time $t = 0$, we set $E(0) = \{(i, j) | \|x_i(0) - x_j(0)\| < r\}$. At time $t > 0$, $(i, j) \in E(t)$ if and only if one of the following conditions hold: (1) $\|x_i(t) - x_j(t)\| \leq r - \varepsilon$; or (2) $r - \varepsilon < \|x_i(t) - x_j(t)\| \leq r$ and $(i, j) \in E(t^-)$, where $t^- < t$ and $|t - t^-| \to 0$. ■

Note that the condition (ii) in the above definition guarantees that a new edge will only be added when the distance between two unconnected agents decreases below $r - \varepsilon$. This property is crucial in proving the connectivity maintenance by Lemma 1 and the convergence by Lemma 2.

Consequently, by Def. 4 each agent $i \in \mathcal{N}$ has a time-varying set of neighbouring agents, denoted by $\mathcal{N}_i(t) = \{i' \in \mathcal{N} \,|\, (i, i') \in E(t)\}$. Note that if j is reachable from i in $G(t)$ then agents i and j are connected, i.e. they can communicate directly or indirectly. From the initial connectivity requirement, we have that $G(0)$ is connected. Hence, maintaining $G(t)$ connected for all $t \geq 0$ ensures that the agents are always connected, too.

B. Continuous Controller Design

In this section, let us firstly focus on the following problem: given a leader $\ell \in \mathcal{N}$ at time t and a goal region $R_{\ell g} \in \mathcal{R}_\ell$, propose a decentralized continuous controller that: (1) guarantees that all agents $i \in \mathcal{N}$ reach $R_{\ell g}$ at a finite time $\bar{t} < \infty$; (2) $G(t')$ remains connected for all $t' \in [t, \bar{t}]$. Both objectives are critical for the leader selection scheme introduced in Section IV-C, which ensures sequential satisfaction of φ_i for each $i \in \mathcal{N}$.

Denote by $x_{ij}(t) = x_i(t) - x_j(t)$ the pairwise relative position between neighbouring agents, $\forall (i, j) \in E(t)$. Thus $\|x_{ij}(t)\|^2 = \big(x_i(t) - x_j(t)\big)^T \big(x_i(t) - x_j(t)\big)$ denotes the corresponding distance. We propose the continuous controller with the following structure:

$$u_i(t) = -b_i\big(x_i - c_{ig}\big) - \sum_{j \in \mathcal{N}_i(t)} \nabla_{x_i} \phi\big(\|x_{ij}\|\big), \tag{4}$$

where $\nabla_{x_i}\phi(\cdot)$ is the gradient of the potential function $\phi\big(\|x_{ij}\|\big)$ with respect to x_i, which is to be defined below; $b_i \in \{0, 1\}$ indicates if agent i is the leader, i.e., $b_i = 1$ if agent i is the leader; $c_{ig} \in \mathbb{R}^2$ is the center of the next goal region for agent i; b_i and c_{ig} are derived from the leader selection scheme in Section IV-C later.

The potential function $\phi(\|x_{ij}\|)$ is defined as follows

$$\phi\big(\|x_{ij}\|\big) = \frac{\|x_{ij}\|^2}{r^2 - \|x_{ij}\|^2}, \qquad \|x_{ij}\| \in [0, r), \tag{5}$$

and has the following properties: (1) its partial derivative of $\phi(\cdot)$ over $\|x_{ij}\|$ is given by

$$\frac{\partial\, \phi\big(\|x_{ij}\|\big)}{\partial\, \|x_{ij}\|} = \frac{2r^2\, \|x_{ij}\|}{(r^2 - \|x_{ij}\|^2)^2} \geq 0, \tag{6}$$

for $\|x_{ij}(t)\| \in [0, r)$ and the equality holds when $\|x_{ij}\| = 0$; (2) $\phi\big(\|x_{ij}\|\big) \to 0$ when $\|x_{ij}\| \to 0$; (3) $\phi\big(\|x_{ij}\|\big) \to +\infty$ when $\|x_{ij}\| \to r$. As a result, controller (4) becomes

$$u_i(t) = -b_i\big(x_i - c_{ig}\big) - \sum_{j \in \mathcal{N}_i(t)} \frac{2r^2}{(r^2 - \|x_{ij}\|^2)^2}\, x_{ij}, \tag{7}$$

which only depends on x_i and x_j, $\forall j \in \mathcal{N}_i(t)$.

Lemma 1: Assume that $G(t)$ is connected at $t = T_1$ and agent $\ell \in \mathcal{N}$ is the fixed leader for all $t \geq T_1$. By applying the controller in Eq. (7), $G(t)$ remains connected and $E(T_1) \subseteq E(t)$ for $t \geq T_1$.

Proof: Assume that $G(t)$ remains *invariant* during $[t_1, t_2) \subseteq [T_1, \infty)$, i.e. no new edges are added to $G(t)$. Consider the following function:

$$V(t) = \frac{1}{2} \sum_{(i,\, j) \in E(t)} \phi(\|x_{ij}\|) + \frac{1}{2} \sum_{i=1}^{N} b_i (x_i - c_{ig})^T (x_i - c_{ig}), \tag{8}$$

which is positive semi-definite. The time derivative of (8) along system (1) is given by

$$\begin{aligned}\dot{V}(t) = &\sum_{i=1,\, i\neq\ell}^{N} \Big(\big(\sum_{j\in\mathcal{N}_i(t)} \nabla_{x_i}\phi(\|x_{ij}\|)\big)\, u_i \Big) \\ &+ \Big(\sum_{j\in\mathcal{N}_\ell(t)} \nabla_{x_\ell}\phi(\|x_{\ell j}\|) + (x_\ell - c_{\ell g}) \Big) u_\ell.\end{aligned} \tag{9}$$

By (7), the control input for each follower $i \neq \ell$ is given by $u_i = -\sum_{j\in\mathcal{N}_i(t)} \nabla_{x_i}\phi(\|x_{ij}\|)$, since $b_i = 0$ for all followers. The control input for the single leader ℓ is given by $u_\ell = -(x_\ell - c_{\ell g}) - \sum_{j\in\mathcal{N}_\ell(t)} \nabla_{x_\ell}\phi(\|x_{\ell j}\|)$, since $b_\ell = 1$. This implies that

$$\begin{aligned}\dot{V}(t) = &- \sum_{i=1,\, i\neq\ell}^{N} \| \sum_{j\in\mathcal{N}_i(t)} \nabla_{x_i}\phi(\|x_{ij}\|)\|^2 \\ &- \|(x_\ell - c_{\ell g}) + \sum_{j\in\mathcal{N}_\ell(t)} \nabla_{x_\ell}\phi(\|x_{\ell j}\|)\,\|^2 \leq 0.\end{aligned} \tag{10}$$

Thus $V(t) \leq V(0) < +\infty$ for $t \in [t_1,\, t_2)$, if no edges are added or removed during that period.

On the other hand, assume a *new* edge $(p,\, q)$ is added to $G(t)$ at $t = t_2$, where $p,\, q \in \mathcal{N}$. By Def. 4, a new edge can only be added if $\|x_{pq}(t_2)\| \leq r - \varepsilon$ and $\phi(\|x_{pq}(t_2)\|) = \frac{(r-\varepsilon)^2}{\varepsilon(2r-\varepsilon)} < +\infty$ since $0 < \varepsilon < r$. If more than one edges are added. Denote the set of newly-added edges at $t = t_2$ by $\widehat{E} \subset \mathcal{N} \times \mathcal{N}$. Let $V(t_2^+)$ and $V(t_2^-)$ be the value of function (8) before and after adding the set of new edges to $G(t)$ at $t = t_2$. We get

$$\begin{aligned}V(t_2^+) &= V(t_2^-) + \sum_{(p,\,q)\in\widehat{E}} \phi(\|x_{pq}(t_2)\|) \\ &\leq V(t_2^-) + |\widehat{E}|\,\frac{(r-\varepsilon)^2}{\varepsilon(2r-\varepsilon)} < +\infty.\end{aligned} \tag{11}$$

Thus $V(t) < \infty$ also holds when new edges are added to $G(t)$. As a result, $V(t) < +\infty$ for $t \in [T_1,\, \infty)$. By Def. 4, one existing edge $(i,\, j) \in E(t)$ will be lost only if $x_{ij}(t) = r$. It implies that $\phi(\|x_{ij}\|) \to +\infty$, i.e., $V(t) \to +\infty$ by (8). By contradiction, we can conclude that new edges might be added but no existing edges will be lost, namely $E(T_1) \subseteq E(t)$, $\forall t \geq T_1$. Thus given a connected $G(t)$ at $t = T_1$ and a fixed leader $\ell \in \mathcal{N}$ for $t \geq T_1$, it is guaranteed that $G(t)$ remains connected, $\forall t \geq T_1$. ■

Lemma 2: Given that $G(t)$ is connected at $t = T_1$ and the fixed leader $\ell \in \mathcal{N}$ for $t \geq T_1$, it is guaranteed that under controller (7) there exist $\bar{t} < +\infty$ that $x_i(\bar{t}) \in R_{\ell g}, \forall i \in \mathcal{N}$.

Proof: It is shown in Lemma 1 that $G(t)$ remains connected for $t \geq T_1$ if $G(T_1)$ is connected. Moreover $E(T_1) \subseteq E(t)$, $\forall t \geq T_1$, i.e. no existing edges will be lost. Then we show that all agents converge to the goal region of the leader in finite time. By (10), $\dot{V}(t) \leq 0$ for $t \geq T_1$ and $\dot{V}(t) = 0$ when the following conditions hold: (1) for $i \neq \ell$ and $i \in \mathcal{N}$, it holds that $\sum_{j\in\mathcal{N}_i(t)} h_{ij}(x_i - x_j) = 0$; and (2) for the leader $\ell \in \mathcal{N}$, it holds that $(x_\ell - c_{\ell g}) + \sum_{j\in\mathcal{N}_\ell(t)} h_{ij}(x_\ell - x_j) = 0$; where h_{ij} is defined as

$$h_{ij} = \frac{2r^2}{(r^2 - \|x_{ij}\|^2)^2}, \qquad \forall (i,\, j) \in E(t). \tag{12}$$

Clearly, $h_{ij} \in [0,\, 2/r^2)$ since $x_{ij} \in [0,\, r - \varepsilon)$, $\forall (i,\, j) \in E(t)$. We can construct a $N \times N$ matrix H satisfying $H(i, i) = \sum_{j\in\mathcal{N}_i} h_{ij}$ and $H(i, j) = -h_{ij}$, where $i \neq j \in \mathcal{N}$. As shown in [19], H is positive semidefinite with a single eigenvalue at the origin, of which the corresponding eigenvector is the unit column vector of length N, denoted by $\mathbf{1}_N$. By combining the above two conditions, we get

$$H \otimes I_2 \cdot \mathbf{x} + B \otimes I_2 \cdot (\mathbf{x} - \mathbf{c}) = 0 \tag{13}$$

where $\otimes$ denotes the Kronecker product [13]; $\mathbf{x}$ is the stack vector for x_i, $i \in \mathcal{N}$; I_2 is the 2×2 identity matrix; B is a $N \times N$ diagonal matrix, with zero on the diagonal except the $(l,\, l)$ element being one; $\mathbf{c} = \mathbf{1}_N \otimes c_{lg}$. Since $H \otimes I_2 \cdot \mathbf{c} = (H \otimes I_2) \cdot (\mathbf{1}_N \otimes c_{lg}) = (H \cdot \mathbf{1}_N) \otimes (I_2 \cdot c_{lg})$ and $H \cdot \mathbf{1}_N = \mathbf{0}_N$, it implies that $H \otimes I_2 \cdot \mathbf{c} = \mathbf{0}_{2N}$. By (13), it implies that $(H + B) \otimes I_2 \cdot (\mathbf{x} - \mathbf{c}) = 0$. Since H is positive semidefinite with one eigenvalue at the origin and B is diagonal with non-negative elements, $H + B$ is positive definite and (13) holds only when $\mathbf{x} = \mathbf{c}$, i.e., $x_i = c_{\ell g}$, $\forall i \in \mathcal{N}$.

By LaSalle's Invariance principle [15], the closed-loop system under controller (7) will converge to the largest invariant set inside $S = \{\mathbf{x} \in \mathbb{R}^{2N} \,|\, x_i = c_{\ell g}, \forall i \in \mathcal{N}\}$, as $t \to +\infty$. It means that all agents converge to the same point $c_{\ell g}$. Since clearly $c_{\ell g} \in R_{\ell g}$, by continuity all agents would enter $R_{\ell g}$ which has a minimal radius r_{min} by (2). Thus there exists $\bar{t} < +\infty$ that $x_i(\bar{t}) \in R_{\ell g}$, $\forall i \in \mathcal{N}$. ■

C. Progressive Goal and Leader Election

We now discuss the election of the leader and the choice of an associated goal region to ensure the overall progress. As the first offline and decentralized step, we generate for each agent i a *high-level plan*. Secondly, in a repetitive online procedure, each agent i is assigned a value that, intuitively, represents the agent's urge to provide the next service in its high-level plan. Using ideas from bully leader election algorithm [8], an agent with the strongest urge is always elected as a leader within the network. By changing the urge dynamically, we ensure that each of the agents is elected as a leader and hence progresses towards its goal infinitely often.

1) Offline high-level plan computation: Given an agent $i \in \mathcal{N}$, a set of services Σ_i, and an LTL formula φ_i over Σ_i, a high-level plan for i in the form of a service sequence $\Omega_i = \sigma_{i1} \cdots \sigma_{ip_i}(\sigma_{ip_i+1} \cdots \sigma_{is_i})^\omega$ can be computed via standard formal verification-based methods (see Sec. II).

2) Urge function: Let i be a fixed agent, t the current time and $\sigma_{i1} \cdots \sigma_{ik}$ a prefix of services of the high-level plan Ω_i that have been provided till t. Moreover, let $\tau_{i\lambda}$ denote the time, when the latest service, i.e., $\sigma_{i\lambda} = \sigma_{ik}$ was provided, or $\tau_{i\lambda} = 0$ in case no service prefix of Ω_i has been provided. Using $\tau_{i\lambda}$, we could define agent i's *urge* at time t as a tuple

$$\Upsilon_i(t) = (t - \tau_{i\lambda},\, i). \tag{14}$$

Algorithm 1 Complete Algorithm for each agent

```
Input: Agents' own ID $i$, the set of all agent IDs $\mathcal{N}$, formula $\varphi_i$
Output: $trace_i$
 1: compute plan $\Omega_i := \sigma_{i1} \cdots \sigma_{ip_i}(\sigma_{ip_i+1} \cdots \sigma_{is_i})^\omega$
 2: $\tau_{i\lambda} := 0$; $\sigma_{i\nu} := \sigma_{i1}$
 3: send $\mathsf{ready}(i)$ and wait to receive $\mathsf{ready}(j)$ for all $j \in \mathcal{N} \setminus \{i\}$
 4: send $\mathsf{init\_elect}(i, t_{cur})$ if $i = N$
 5: loop
 6:   wait to receive a message $m$
 7:   switch $m$
 8:     case $m = \mathsf{init\_elect}(i', t)$ for some $i' \in \mathcal{N}$ and time $t$
 9:       send $\mathsf{me}(\Upsilon_i(t))$ and receive $\mathsf{me}(\Upsilon_j(t))$
10:       elect the leader $\ell \in \mathcal{N}$ maximizing $\Upsilon_\ell(t)$
11:       send $\mathsf{finish\_elect}(i)$ and wait to receive $\mathsf{finish\_elect}(j)$
12:       if $\ell = i$ then
13:         $b_i := 1$
14:         pick $R_{\ell g} = R_{ig}$, such that $\sigma_{i\nu} \in L_i(R_{ig})$
15:         apply controller $u_i$ from (7) until $x_j(t) \in R_{\ell g}$ for
            all $j \in \{i\} \cup \mathcal{C}_{i\nu}$
16:         send $\mathsf{execute\_request}(\varpi_{ji\nu})$ for all $j \in \mathcal{C}_{i\nu}$
17:         execute $\pi_{i\nu}$
18:         $\tau_{i\lambda} := t_{cur}$; $\sigma_{i\nu} := \sigma_{i\nu+1}$
19:         update prefixes of $\mathbb{T}_i^A, \mathbb{A}_i, \mathbb{T}_i^S$, and $\mathbb{S}_i$
20:         send $\mathsf{init\_elect}(i, t_{cur})$
21:       else
22:         $b_i := 0$
23:         apply controller $u_i$ from (7) until a message $m$ is
            received; goto line 7
24:       end if
25:     case $m = \mathsf{execute\_request}(\varpi_{ii'h'})$
26:       execute $\varpi_{ii'h'}$
27:       update prefixes of $\mathbb{T}_i^A$, and $\mathbb{A}_i$; goto line 7
28:   end switch
29: end loop
```

To compare the agents' urges at time t, we use lexicographical ordering: $\Upsilon_i(t) > \Upsilon_j(t)$ iff (1) $t - \tau_{i\lambda} > t - \tau_{j\lambda}$; or (2) $t - \tau_{i\lambda} = t - \tau_{j\lambda}$, and $i > j$. Note that $i \neq j$ implies that $\Upsilon_i(t) \neq \Upsilon_j(t)$, $\forall t \geq 0$. Thus there exists exactly one agent with the maximal urge at any time t.

3) Overall algorithm: The solution for an agent $i \in \mathcal{N}$ is summarized in Alg. 1 and is run on each agent separately. The algorithm is initialized with the offline synthesis of high-level plans (line 1). Then, the agent broadcasts a message to acknowledge the others that it is ready to proceed and waits to receive analogous messages from the remaining agents (line 3). The first leader election is triggered by a message sent by the agent N (line 4) with the time stamp t_{cur}.

Several types of messages can be received. Message $\mathsf{init\_elect}(i', t)$ notifies that leader re-election is triggered (line 8). In such a case, the agent sends out the message $\mathsf{me}(\Upsilon_i(t))$ with its urge value $\Upsilon_i(t)$ and waits to receive analogous messages from the others (line 9). The agent with the maximal urge is elected as the leader (line 10) and the algorithm proceeds when each of the agents has set the new leader (line 11). The rest of the algorithm differs depending on whether the agent i is the leader (lines 12-21) or not (lines 21-24). Different controller scheme from (7) is applied to reach the leader's goal region. Then it provides service $\sigma_{i\nu}$ to finish its own service or help others (lines 16-17). Finally, the leader triggers a leader re-election (line 20). The algorithm naturally determines the trace $trace_i = (\mathbf{x}_i(t), \mathbb{T}_i^A, \mathbb{A}_i, \mathbb{T}_i^S, \mathbb{S}_i)$ by lines 19 and 27.

Lemma 3: Given an agent $i \in \mathcal{N}$ at time t, there exists $T \geq t$, such that $\Upsilon_i(T) > \Upsilon_j(T)$, for all $j \in \mathcal{N}$, and $t \geq 0$.

Proof: By contradiction. Assume that for all $t' \geq t$ there exists some $j \in \mathcal{N}$, such that $\Upsilon_i(t') < \Upsilon_j(t')$. Consider that $\ell \in \mathcal{N}$ is set as the leader at time t, and an agent $i' \in \mathcal{N}$ maximizes $\Upsilon_{i'}(t)$ among all agents in $\mathcal{N}$. From the construction of Alg. 1 and Lemmas 1 and 2, there exists $\tau_{\ell\nu} \geq t$ when the next leader's desired service $\sigma_{\ell\nu}$ has been provided and a leader re-election is triggered with the time stamp $\tau_{\ell\nu}$. Note that from (14), $\Upsilon_{i'}(\tau_{\ell\nu})$ is still maximal among the agents in $\mathcal{N}$, and hence i' becomes the next leader. Furthermore, there exists time $\tau_{i'\nu} \geq \tau_{\ell\nu}$ when the next desired service $\sigma_{i'\nu}$ of agent i' has been provided, and hence $\Upsilon_{i'}(\tau_{i'\nu}) < \Upsilon_j(\tau_{i'\nu})$, for all $j \in \mathcal{N}$, including the agent i. Since we assume that i does not become a leader for any $t' \geq t$, it holds that $\Upsilon_{i'}(t'') < \Upsilon_i(t'')$, $\forall t'' \geq \tau_{i'\nu}$. We can reason similarly about the remaining agents. As the number of agents is finite, after large enough $T \geq t$, we obtain that $\Upsilon_j(t') < \Upsilon_i(t')$ for all j and for all $t' \geq T$. This contradicts the assumption and the proof is complete. ■

From Alg. 1, each agent has its high-level plan Ω_i and waits for the first leader $\ell_1 \in \mathcal{N}$ to be elected. By Lemma 2 there exists a finite time $\bar{t}_1 > 0$, such that $x_{\ell_1}(\bar{t}_1) \in R_{\ell_1\nu}$, while at the same time by Lemma 1 the communication network $G(t)$ remains connected, $\forall t \in [0, \bar{t}_1]$. By induction, given a leader ℓ_t and a goal region $R_{\ell\nu}$ at time $t > 0$, there exists $\bar{t} \geq t$, such that $x_\ell(\bar{t}) \in R_{\ell\nu}$. Together with Lemma 3, we conclude that ϕ_i is satisfied, $\forall i \in \mathcal{N}$.

Corollary 1: Algorithm 1 solves Problem 1. ■

V. Example

In the following case study, we present an example of a team of four autonomous robots with heterogeneous functionalities. All simulations are carried out in MATLAB on a desktop computer (3.06 GHz Duo CPU and 8GB of RAM).

A. System Description

We consider agents $\mathfrak{R}_1$, $\mathfrak{R}_2$, $\mathfrak{R}_3$ and $\mathfrak{R}_4$ with dynamics as in Eq. (1). They all have the communication radius $1.5m$, while ε is $0.1m$. The workspace of size $4m \times 4m$ is given in Fig. 1, within which the regions of interest for $\mathfrak{R}_1$ are R_{11}, R_{12} (in red), for $\mathfrak{R}_2$ are R_{21}, R_{22} (in green), for $\mathfrak{R}_3$ are R_{31}, R_{32} (in blue) and for $\mathfrak{R}_4$ are R_{41}, R_{42} (in cyan). Each agent can provide various services as follows: agent $\mathfrak{R}_1$ can load (l_H, l_A), carry and unload (u_H, u_A) a heavy object H or a light object A. Besides, it can help $\mathfrak{R}_4$ to assemble (h_C) object C ; agent $\mathfrak{R}_2$ is capable of helping the agent $\mathfrak{R}_1$ to load the heavy object H (h_H), and to execute two tasks (t_1, t_2) without help; agent $\mathfrak{R}_3$ is capable of taking snapshots (s) when being present in its own or others' goal regions; agent $\mathfrak{R}_4$ can assemble (a_C) object C under the help of agent $\mathfrak{R}_1$.

B. Task Description

Each agent has been assigned a local complex task that requires collaboration: agent $\mathfrak{R}_1$ has to periodically load the

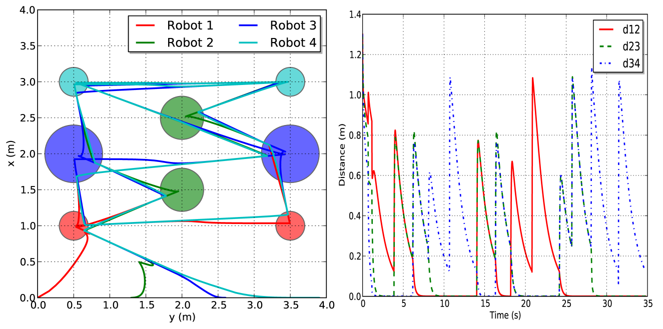

Fig. 1: Left: the agents' trajectory during time $[0, 34.8s]$; Right: the evolution of pair-wise distances $\|x_{12}\|$, $\|x_{23}\|$, $\|x_{34}\|$, which all stay below the radius $1.5m$ as required by the connectivity constraints.

heavy object H at region R_{11}, unload it at region R_{12}, load the light object A at region R_{12}, unload it at region R_{11}. In LTL formula, it is specified as $\phi_1 = \mathsf{GF}\big((l_H \wedge h_H \wedge r_{11}) \wedge \mathsf{X}(u_H \wedge r_{12})\big) \wedge \mathsf{GF}\big((l_A \wedge r_{12}) \wedge (u_A \wedge r_{11})\big)$; Agent $\mathfrak{R}_2$ has to service the simple task t_1 at region R_{21} and task t_2 at region R_{22} in sequence, but it requires $\mathfrak{R}_2$ to witness the execution of task t_2, by taking a snapshot at the moment of the execution. It is specified as $\phi_2 = \mathsf{F}\big((t_1 \wedge r_{21}) \wedge \mathsf{F}(t_2 \wedge s \wedge r_{22})\big)$; Agent $\mathfrak{R}_3$ has to surveil over both of its goal regions (R_{31}, R_{32}) and take snapshots there, which is $\phi_3 = \mathsf{GF}(s \wedge r_{31}) \wedge \mathsf{GF}(s \wedge r_{32})$; Agent $\mathfrak{R}_4$ has to assemble object C at its goal regions (R_{41}, R_{42}) infinitely often, which is $\phi_4 = \mathsf{GF}(a_C \wedge r_{41}) \wedge \mathsf{GF}(a_C \wedge r_{42})$. Note that ϕ_1, ϕ_3 and ϕ_4 require collaboration tasks be performed infinitely often.

C. Simulation Results

Initially, the agents start evenly from the x-axis, from $(0, 0)$, $(1.3, 0)$, $(2.6, 0)$, $(3.9, 0)$, respectively. By Def. 4, the initial edge set is $E(0) = \{(1, 2), (2, 3), (3, 4)\}$, yielding a connected $G(0)$. The system executing Alg. 1 is simulated for $35s$, of which the video demonstration can be viewed here [3]. Initially, agent $\mathfrak{R}_1$ is chosen as the leader. Controller (4) is applied for $\mathfrak{R}_1$ as the leader and the rest as followers, while the next goal region of $\mathfrak{R}_1$ is R_{11}. All agents belong to R_{11} after $t = 3.8s$. After that agent $\mathfrak{R}_2$ helps agent $\mathfrak{R}_1$ to load object H. Then agent $\mathfrak{R}_2$ is elected as the leader after the heavy object is loaded, where R_{21} is chosen as the next goal region. At $t = 6.1s$, all agents converge to R_{21}. Afterwards, the leader and goal region is switched in the following order: $\mathfrak{R}_3$ as leader to region R_{31} at $t = 6.1s$; $\mathfrak{R}_4$ as leader to R_{41} at $t = 8.1s$; $\mathfrak{R}_4$ as leader to R_{42} at $t = 10.6s$; $\mathfrak{R}_2$ as leader to R_{22} at $t = 14.2s$; $\mathfrak{R}_3$ as leader to R_{32} at $t = 16.3s$; $\mathfrak{R}_1$ as leader to R_{12} at $t = 18.2s$; $\mathfrak{R}_1$ as leader to R_{11} at $t = 20.1s$; $\mathfrak{R}_3$ as leader to R_{31} at $t = 24.2s$; $\mathfrak{R}_3$ as leader to R_{32} at $t = 25.7s$; $\mathfrak{R}_4$ as leader to R_{41} at $t = 28.1s$. $\mathfrak{R}_4$ as leader to R_{42} at $t = 31.4s$.

Figure 1 shows the trajectory of $\mathfrak{R}_1$, $\mathfrak{R}_2$, $\mathfrak{R}_3$, $\mathfrak{R}_4$ during time $[0, 34.7s]$, in red, green, blue, cyan respectively. Furthermore, the pairwise distance for neighbours within $E(0)$ is shown in Figure 1. It can be verified that they stay below the constrained radius $1.5m$ thus the agents remain connected.

VI. Conclusions and Future Work

We present a distributed motion and task control framework for multi-agent systems under complex local LTL tasks and connectivity constraints. It is guaranteed that all individual tasks are fulfilled, while at the same time connectivity constraints are satisfied. Further work includes inherently-coupled dynamics and time-varying network topology.

References

[1] C. Baier and J. P. Katoen. *Principles of Model Checking*. MIT Press, 2008.

[2] Y. Chen, X. C. Ding, A. Stefanescu, and C. Belta. Formal approach to the deployment of distributed robotic teams. *IEEE Transactions on Robotics*, 28(1): 158–171, 2012.

[3] Simulation. `https://www.dropbox.com/s/yhyueagokeihv7k/simulation.avi`.

[4] D. V. Dimarogonas and K. J. Kyriakopoulos. A connection between formation control and flocking behavior in nonholonomic multiagent systems. *IEEE International Conference on Robotics and Automation(ICRA)*, 940–945, 2006.

[5] M. B. Egerstedt and X. Hu. Formation constrained multi-agent control. *Georgia Institute of Technology*, 2001.

[6] Extended version. `http://arxiv.org/abs/1405.1836`.

[7] I. Filippidis, D. V. Dimarogonas, and K.J. Kyriakopoulos. Decentralized multi-agent control from local LTL specifications. *IEEE Conference on Decision and Control(CDC)*, 6235–6240, 2012.

[8] H. Garcia-Molina. Elections in a distributed computing system. *IEEE Transactions on Computers*, 31(1): 48–59, 1982.

[9] P. Gastin and D. Oddoux. LTL2BA tool, viewed September 2012. URL: http://www.lsv.ens-cachan.fr/ gastin/ltl2ba/.

[10] M. Guo and D. V. Dimarogonas. Reconfiguration in motion planning of single- and multi-agent systems under infeasible local LTL specifications. *IEEE Conference on Decision and Control(CDC)*, 2758–2763, 2013.

[11] M. Guo and D. V. Dimarogonas. Distributed plan reconfiguration via knowledge transfer in multi-agent systems under local LTL specifications. *IEEE International Conference on Robotics and Automation(ICRA)*, 2014.

[12] Y. Hong, J. Hu, and L. Gao. Tracking control for multi-agent consensus with an active leader and variable topology. *Automatica*, 42(7): 1177–1182, 2006.

[13] R. A. Horn and C. R. Johnson. *Matrix analysis*. Cambridge university press, 2012.

[14] S. Karaman and E. Frazzoli. Vehicle routing with temporal logic specifications: Applications to multi-UAV mission planning. *International Journal of Robust and Nonlinear Control*, 21:1372–1395, 2011.

[15] H. K. Khalil and J. W. Grizzle. *Nonlinear systems*, volume 3. Prentice hall Upper Saddle River, 2002.

[16] M. Kloetzer, X. C. Ding, and C. Belta. Multi-robot deployment from ltl specifications with reduced communication. *IEEE Conference on Decision and Control and European Control Conference(CDC-ECC)*, 4867–4872, 2011.

[17] S. G. Loizou and K. J. Kyriakopoulos. Automated planning of motion tasks for multi-robot systems. *IEEE Conference on Decision and Control (CDC)*, 78–83, 2005.

[18] M. M. Quottrup, T. Bak, and R. I. Zamanabadi. Multi-robot planning: a timed automata approach. *IEEE International Conference on Robotics and Automation (ICRA)*, 4417–4422, 2004.

[19] W. Ren. Multi-vehicle consensus with a time-varying reference state. *Systems & Control Letters*, 56(7): 474–483, 2007.

[20] J. Tumova and D. V. Dimarogonas. A receding horizon approach to multi-agent planning from LTL specifications. *American Control Conference (ACC)*, 1775–1780, 2014.

[21] A. Ulusoy, S. L. Smith, X. C. Ding, C. Belta, and D. Rus. Optimality and robustness in multi-robot path planning with temporal logic constraints. *International Journal of Robotics Research*, 32(8): 889–911, 2013.

[22] C. Wiltsche, F. A. Ramponi, and J. Lygeros. Synthesis of an asynchronous communication protocol for search and rescue robots. *European Control Conference (ECC)*, 1256–1261, 2013.