Automated Formation Control Synthesis from Temporal Logic Specifications

Shuhao Qi†, Zengjie Zhang†, Sofie Haesaert, Zhiyong Sun

Abstract— In many practical scenarios, multi-robot systems are envisioned to support humans in executing complicated tasks within structured environments, such as search-and-rescue tasks. We propose a framework for a multi-robot swarm to fulfill complex tasks represented by temporal logic specifications. Given temporal logic specifications on the swarm formation and navigation, we develop a controller with runtime safety and convergence guarantees that drive the swarm to formally satisfy the specification. In addition, the synthesized controller will autonomously switch formations as necessary and react to uncontrollable events from the environment. The efficacy of the proposed framework is validated with a simulation study on the navigation of multiple quadrotor robots.

I. INTRODUCTION

Multi-robot system control with complex tasks is challenging due to the high dimensions and the substantial amount of constraints [1]. Taking the search-and-rescue task as an example, the robots have to sequentially achieve a series of subtasks, including locating the survivors, navigating to the rescue spots, and transporting the survivors to a designated safe zone. In this type of complex task, robot swarms should autonomously change formations to cooperatively execute sensing and communication operations or to traverse narrow spaces. Such practical tasks involve complex navigational requirements imposed by the sequential subtasks in combination with formation control requirements on the robots. While the former navigational requirements have been successfully solved by first specifying them with Linear-time Temporal Logic (LTL) and using related tools [2], [3], there is not much work that explicitly incorporates the formation requirements. This paper studies the control synthesis of multi-robot systems for complex tasks represented by LTL specifications with automated formations.

For constrained control of multi-robot systems, problems such as obstacle avoidance and reachability in complex environments can be solved with recently developed control methods that also give runtime guarantees. These methods include control barrier functions (CBF) [4] and finite-time control Lyapunov function (CLF) [5]. CBFs can ensure the strict satisfaction of state-dependent constraints for dynamic systems by imposing the set invariance property [4]. In addition, fixed-time CLF [5] can ensure that the system converges to a given equilibrium point or a set within user-defined time. Also, in practice, it is convenient to incorporate CBF and CLF constraints in optimization-based controllers, like quadratic programming (QP), which can balance safety and convergence without massive computation [6], [5]. In this sense, CBF-CLF QP has been employed to ensure the safety and convergence of multi-robot systems [7], [8] and the satisfaction of the spatio-temporal constraints of temporal logic specifications [9], [10].

LTL specifications have been used to enable the collective behavior of multi-robot systems [2], [11], [12]. The allocation numbers of robots or systems to specific regions in space and time can be represented by counting temporal logic [13] and graph temporal logic [14]. Alternatively, when each agent in a system has been individually assigned a distinct temporal logic specification, local cooperation of agents has been developed in [15] where the least violating control is used in case of conflicting specifications. In contrast, in [12], the collective behavior of multi-robot systems is achieved, where the mean and variance features of the robot swarm are employed to control arbitrarily large swarm systems to satisfy temporal logic specifications. In some practical problems, multi-robot systems are expected to not only exhibit simple collective behaviors but also to achieve specific formations to traverse special terrain and to execute sensing and communication tasks. In contrast to [12], we aim at the explicit specification for automated formations in the navigation control of a robot swarm.

More precisely in this paper, we propose a formal framework for a multi-robot swarm system to solve a complicated autonomous navigation task with automated formations. A linear temporal logic (LTL) formula is used to specify this complicated navigation task and a symbolic model is abstracted to characterize the behavior of the swarm system. Based on this model, a symbolic controller is synthesized to generate the waypoints and desired formations subject to the LTL specification incorporating the influence of the environmental signals. Then, a QP-based control refinement method with CBFs and fixed-time CLFs is developed to ensure the satisfaction of the LTL specification with runtime guarantees. In such a way, the developed control method enables the swarm system to succeed in the navigation task with automated formations. Such a framework holds vast potential for real-world robotic swarm applications. The main contributions of this paper include 1) a framework for autonomous navigation and formation switching of multi-robot systems under LTL specifications, including finite abstractions, symbolic control synthesis, and control refinement;

2) an efficient QP-based control refinement with runtime guarantees on the task specification (autonomous navigation with automated formations) and collision avoidance.

The rest of the paper is organized as follows. Sec. II gives the preliminary knowledge and the main problem. The main results are given in Sec. III.In Sec. IV, we validate our framework and solutions with a simulation case on robot swarm navigation. Finally, Sec. V concludes the paper.

II. PRELIMINARIES AND PROBLEM STATEMENT

A. Multi-Robot System

Consider a multi-robot swarm system with r robots, where each robot is described by the following dynamic equation,

$$\mathcal{R}_i : \dot{x}_i(t) = g(x_i(t), u_i(t)), \quad i = 1, \ldots, r, \qquad (1)$$

where $x_i(t) \in \mathbb{X} \subset \mathbb{R}^n, u_i(t) \in \mathbb{U} \subset \mathbb{R}^m$ are the state and the control input of the i-th robot at time $t \in \mathbb{R}_{\geq 0}$, respectively, and $g : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^n$ is a smooth function that describes the dynamic model of the robot. For brevity, we use vectors $x(t) = [\, x_1^T(t), x_2^T(t), \cdots, x_r^T(t)\,]^T \in \mathbb{X}^r \subset \mathbb{R}^{rn}$ and $u(t) = [\, u_1^T(t), u_2^T(t), \cdots, u_r^T(t)\,]^T \in \mathbb{U}^r \subset \mathbb{R}^{rm}$ to denote the state and control input of all robots in the system. Especially, the initial system state is denoted by $x(0) = [\, x_1^T(0), x_2^T(0), \cdots, x_r^T(0)\,]^T$.

In this paper, the interaction among the robots is described by an undirected and completed graph, in the sense that all robots in the swarm are fully connected by local communication. We define $x_c = \frac{1}{r} \sum_{i=1}^{r} x_i$ as the *centroid*, or the geometric center of the swarm and $x_{ij} = x_i - x_j$ as the relative displacement between two robots $i \neq j$. A certain *formation* of the swarm system is described by all the relative displacements, i.e., $f = \{f_{ij}\}_{\frac{r(r-1)}{2}}$, where $f_{ij} \in \mathbb{R}^n$, for $i, j \in \{1, 2, \cdots, r\}$, $i \neq j$, denotes the desired displacement between two robots. We define $\mathbb{F}$ as the formation space that contains all possible formations f of the swarm system. Similarly, we define $\mathbb{W} \subset \mathbb{R}^n$ as the workspace of the swarm that contains all possible positions w of the centroid x_c.

B. LTL Specification for Navigation Task

We first introduce Linear Temporal Logic (LTL) [16], [17] as follows.

Syntax. The syntax of LTL is recursively defined as,

$$\psi ::= \top \mid p \mid \neg\psi \mid \psi_1 \wedge \psi_2 \mid \bigcirc\psi \mid \psi_1 \mathsf{U} \psi_2, \qquad (2)$$

where ψ_1, ψ_2 and ψ are LTL formulas, $p \in \mathsf{AP}$ is an atomic proposition, $\neg$ is the negation operator, $\wedge$ is the conjunction operator that connects two LTL formulas, and $\bigcirc$ and U represent the *next* and *until* temporal operators, respectively. Based on these essential operators, other logical and temporal operators, namely *disjunction* $\vee$, *implication* $\to$, *eventually* $\Diamond$, and *always* $\square$ can be defined as, $\psi_1 \vee \psi_2 := \neg(\psi_1 \wedge \psi_2)$, $\psi_1 \to \psi_2 := \neg\psi_1 \vee \psi_2$, $\Diamond\psi := \top \mathsf{U}\psi$, and $\square\psi := \neg\Diamond\neg\psi$.

Semantics. Consider a set of atomic propositions $\mathsf{AP} = \{p_1, \ldots, p_N\}$ which defines an alphabet 2^{AP}, where each letter $\omega \in 2^{\mathsf{AP}}$ contains the set of atomic propositions that are true. An infinite string of letters is a word $\boldsymbol{\omega} = \omega_0 \omega_1 \omega_2 \ldots$, where $\omega_i \in 2^{\mathsf{AP}}$, $i \in \mathbb{N}_{\geq 0}$, with a suffix

$\boldsymbol{\omega}_k = \omega_k \omega_{k+1} \omega_{k+2} \ldots, k \in \mathbb{N}_{\geq 0}$. For a given word $\boldsymbol{\omega}$, basic semantics of LTL are given as $\boldsymbol{\omega}_k \models p$, if $p \in \omega_k$; $\boldsymbol{\omega}_k \models \neg p$, if $p \notin \omega_k$; $\boldsymbol{\omega}_k \models \psi_1 \wedge \psi_2$, if $\boldsymbol{\omega}_k \models \psi_1$ and $\boldsymbol{\omega}_k \models \psi_2$; $\boldsymbol{\omega}_k \models \bigcirc\psi$, if $\boldsymbol{\omega}_{k+1} \models \psi$; $\boldsymbol{\omega}_k \models \psi_1 \mathsf{U} \psi_2$, if $\exists\, i \in \mathbb{N}$ such that $\boldsymbol{\omega}_{k+i} \models \psi_2$, and $\boldsymbol{\omega}_{k+j} \models \psi_1$ holds $\forall\, 0 \leq j < i$.

Reactive LTL formulas for robot swarm. In this paper, we use an LTL formula to specify whether the swarm system achieves the navigation task with automated formation. Motivated by [12], we select to use the centroid position and formation as essential features to respectively characterize the navigation and interaction behaviors of the swarm system. Therefore, we introduce atomic propositions for the swarm centroid in the workspace AP_w based on a labeling function $\mathcal{L}_w : \mathbb{W} \to 2^{\mathsf{AP}_w}$ and those for the formation AP_f with a labeling function $\mathcal{L}_f : \mathbb{F} \to 2^{\mathsf{AP}_f}$. Moreover, to specify how the swarm system should react to external signals from the environment, we use a finite set $\mathcal{E}$ to denote all possible states of the environmental signal and the set of atomic propositions AP_e to describe the property of the signal. Their correspondence is described by a labeling mapping $\mathcal{L}_e : \mathcal{E} \to 2^{\mathsf{AP}_e}$. In this sense, the overall set of atomic propositions of the swarm system for the navigation task is $\mathsf{AP} := \mathsf{AP}_e \cup \mathsf{AP}_w \cup \mathsf{AP}_f$. Then, we can use an LTL formula ψ defined on the output word of the swarm system $\boldsymbol{\omega} = \omega_0 \omega_1 \omega_2 \ldots$, where $\omega_k \in 2^{\mathsf{AP}}$, $k \in \mathbb{N}$, to specify the navigation task. We say that the robot achieves the navigation task if its output word satisfies the specification, i.e., $\boldsymbol{\omega} \models \psi$. In this sense, the overall labeling mapping of the swarm system is the combination of all mappings $\mathcal{L}_w$, $\mathcal{L}_f$, and $\mathcal{L}_e$, i.e., $\mathcal{L} : \mathbb{W} \times \mathbb{F} \times \mathcal{E} \to 2^{\mathsf{AP}}$. The subset LTL formulas known as Generalized Reactive(1) (GR(1)) [18], is specifically well fitted to deal with both external and internal variables. GR(1) formulas are of the form $\psi := \psi_e \to \psi_s$, where ψ_e constrains the allowed behavior of the uncontrolled propositions AP_e and ψ_s constrains the desired behaviors of the system with the controlled propositions $\mathsf{AP}_w \cup \mathsf{AP}_f$.

C. Problem Statement

Given the multi-robot system in (II-A) and the task specification ψ defined in (II-B), the navigation task studied in this paper can be formulated as a synthesis problem for the specification ψ. Before we give the formal problem statement, we explain how automated formation is addressed for the navigation task. The main objective of the navigation task is that the centroid of the swarm system should ultimately reach the navigation goal along a feasible path in the environment (*navigation*). Meanwhile, the environmental terrain requires that certain spots must be passed by certain formations. Therefore, the swarm system should also automatically switch to the feasible formations when passing the corresponding spots (*automated formation*). Also, the swarm system should always react to the environmental signals and perform the correct response (*reaction*). Besides, all robots in the swarm should avoid collisions with each other and the obstacles in the environment (*collision avoidance*). The formal problem statement is given as follows.

Main Problem 1 (Navigation with Automated Formation). *For a multi-robot swarm system $\mathcal{R} = \{\mathcal{R}_1, \mathcal{R}_2, \cdots, \mathcal{R}_r\}$ as defined in Eq. (1) with a state space $\mathbb{X}^r$, a workspace $\mathbb{W}$, a formation space $\mathbb{F}$, an environmental signal space $\mathcal{E}$, a labeling mapping $\mathcal{L} : \mathbb{W} \times \mathbb{F} \times \mathcal{E} \to \mathsf{AP}$, and a GR(1) formula $\psi := \psi_e \to \psi_s$ defined on the alphabet 2^{AP} specifying the requirements on navigation, automated formation, reaction, and collision avoidance, find a provably-correct reactive control policy to ensure that the output word $\boldsymbol{\omega}$ satisfies the specification ψ, i.e, $\boldsymbol{\omega} \models \psi$.* $\square$

III. Framework and Controller Design

A. The Control Design Framework

To capture behaviors of the dynamic system in Eq. 1, an abstract system can be defined in the following formulation,

Definition 1. *(Deterministic Finite Transition System, DFTS): A deterministic finite transition system is a tuple $\mathcal{T} = (S, s_0, A, \delta, \mathsf{AP}, \mathcal{L})$, where S and A are finite sets of states and actions, s_0 is the initial state, $\delta : S \times A \to S$ is a transition function that prescribes the state transition under a certain input,* AP *is a finite set of atomic propositions, and $\mathcal{L} : S \to 2^{\mathsf{AP}}$ is a labeling function.* $\square$

Given a sequence of actions $\mathbf{a} = a_0 a_1 a_2 \cdots$ with $a_i \in A$, a DFTS $\mathcal{T}$ initiated at $s_0 \in S$ generates a trajectory or a run $\boldsymbol{s} = s_0 s_1 s_2, \ldots$, where $s_{i+1} = \delta(s_i, a_i)$, $i \in \mathbb{N}$. Accordingly, the output word of the DFTS $\boldsymbol{\omega} = \omega_0 \omega_1 \omega_2 \cdots$ is uniquely defined for a given initial state $s_0 \in S$, where $\omega_i \in 2^{\mathsf{AP}}$, $i \in \mathbb{N}$. Let a history $h_k = s_0 a_0 s_1 a_1 s_2 a_2 ... s_k$ be given at time k with $h_k \in H$, we can then define a control strategy as a map μ from the set of histories to the action set, i.e., $\mu : H \to A$. A control strategy in the form $\mu : S \to A$ is a Markov strategy as it only depends on current state of $\mathcal{T}$.

In the next subsection, we will show that we can exactly develop a symbolic model as the abstraction of the swarm system in Eq. (1) using a DFTS with the same atomic proposition sets AP_w and AP_f. More specifically, let us denote two finite sets $\mathcal{W} := \{w_1, w_2, \ldots\} \subset \mathbb{W}$ and $\mathcal{F} := \{f_1, f_2, \ldots\} \subset \mathbb{F}$ with waypoints in the workspace $\mathbb{W}$ and the formation space $\mathbb{F}$ of the swarm system. The state set of the DFTS $S := \mathcal{W} \times \mathcal{F}$ is an abstraction of the feature space of the swarm system $\mathbb{W} \times \mathbb{F}$. As a consequence, the task specification ψ defined for the swarm system can now be specified for the abstract model with labeling map $\mathcal{L} : \mathcal{W} \times \mathcal{F} \times \mathcal{E} \to 2^{\mathsf{AP}}$. Consider the history signal $h_k^{\mathcal{E}} := s_0 e_0 a_0 s_1 e_1 a_1 s_2 e_1 a_2 ... s_k \in H^{\mathcal{E}}$ extended with the uncontrolled variables $e_k \in \mathcal{E}$. Then, we can decompose the main problem into two sub-problems with one being the control synthesis for the abstract model $\mathcal{T}$ and the other as control refinement to guarantee the soundness of the output words $\boldsymbol{\omega}$ for the concrete swarm system.

Sub-Problem 1 (Symbolic control synthesis). *Consider a DFTS $\mathcal{T}$ defined in Def. 1 as the abstraction of the swarm system given in Eq. (1). Synthesize a symbolic control strategy, $\mu : H^{\mathcal{E}} \to A$, such that for any feasible environmental signal from $\mathcal{E}$ and for the initial state s_0, the LTL specification ψ is satisfied.*

Given the control strategy μ for the abstract system $\mathcal{T}$, we should also make sure there is a refined control policy for the swarm system that drives the multi-robot system to satisfy the navigation specification ψ without collisions inside and outside the swarm.

Sub-Problem 2 (Control refinement). *For the swarm system given in Eq. (1) and its abstract model, a DFTS $\mathcal{T}$ defined in Def. 1, with a symbolic control strategy μ solved in Sub-Problem 1, design a controller that maps the symbolic state action pairs to a continous control map $\pi : \mathbb{X}^r \times S \times A \to \mathbb{U}^r$, such that the output word of the swarm system $\boldsymbol{\omega}$ assigned by the labeling mapping $\mathcal{L}$ satisfies the LTL specification defined in Sec. II-B, i.e., $\boldsymbol{\omega} \models \psi$.* $\square$

B. Symbolic Control Synthesis

This subsection discusses the synthesis of the symbolic control strategy in *Sub-Problem 1*. The synthesis process is performed in three steps:

1) Step 1: Determining Waypoint Sets: The first step is to determine the sets of waypoints in the workspace and the formation, namely $\mathcal{W}$ and $\mathcal{F}$ which are important to construct the abstraction model $\mathcal{T}$, as introduced in Sec. III. A waypoint is a desired position $w \in \mathcal{W}$ for the swarm centroid to reach or a desired formation $f \in \mathcal{F}$ for the swarm to achieve. In this work, the waypoints are selected by fully incorporating the physical property of the swarm, the practical conditions of the environment, and the specific requirements of the tasks. For example, there might exist some narrow spaces where the swarm can only pass with a certain formation. Also, the number of the waypoints is selected as possibly small while maintaining a dense distribution to make a balance between the scale of the abstraction model and the feasibility of the navigation task.

2) Step 2: Realization of the DFTS: After determining the finite sets of waypoints $\mathcal{W}$ and $\mathcal{F}$, the next step is to use them to construct a DFTS $\mathcal{T} = \{S, s_0, A, \delta, \mathsf{AP}, \mathcal{L}\}$ to realize an abstract model for the swarm system. Directly, the state set is $S = \mathcal{W} \times \mathcal{F}$. The action set A is also a finite symbolic set that contains all possible actions for state transitions. The atomic proposition set AP and labeling mapping $\mathcal{L}$ are the same as Sec. II-B. Thus, the most critical part of this step is to determine the transition function δ which describes to which waypoint the system transits given the current waypoint. We first assume a full transition and then eliminate the infeasible transitions by looking into the environmental conditions and the dynamic model of the swarm system. For example, we eliminate a transition from one waypoint to another if it forces the swarm to pass an area with an impractical formation. Moreover, the transition should also incorporate the feasibility of the control refinement process described by *Sub-Problem 2*, which will be discussed in the next subsection.

3) Step 3: Synthesis of Control Strategy: Given the abstract, symbolic model, we can use an off-the-shelf tool to solve the control synthesis problem. More precisely, we use the Omega solver [19] in Tulip [20], [21]. Internally, it constructs an enumerated transducer that ensures the satisfaction

of the GR(1) formula for any admissible behavior of the uncontrolled variables e_k. For the overall synthesis procedure can be referred to [18], [21]. Note that if the environment variable takes a value other than the imposed assumptions, no guarantees can be provided on the system behavior.

C. Control Refinement with Runtime Guarantees

In this subsection, we design a QP-based control refinement to solve *Sub-Problem 2*. Let the symbolic state $s = (w, f) \in S$ and the symbolic action $a \in A$ be given for which the next symbolic state should be $s_+ = \delta(s, a)$. The refined control inputs $u(t)$ should control the robot swarm such that the desired waypoints (w, f) defined by the symbolic control are reached. Additionally, different constraints are added concerning the runtime safety requirements.

1) Computing Robot Control Inputs Via Solving QP: Without losing generality, we consider a swarm with single integrator models, i.e., $\dot{x}_i(t) = u_i(t)$ for $i \in \{1, 2, \cdots, r\}$. For *Sub-Problem 2*, we need to solve the following QP problem formulated in Eq. (3), for all $i \neq j$, $i, j \in \{1, 2, \cdots, r\}$,

$$\min_z z^{\mathrm{T}} H z + Q^{\mathrm{T}} z \tag{3a}$$

$$\text{s.t.} \quad \|u_i\| \leq u_{\max}, \tag{3b}$$

$$\frac{\partial h_{\mathcal{W}}(x_c, w)}{\partial x_c} u_c \leq \delta_1 h_{\mathcal{W}}(x_c, w)$$
$$- \alpha_1 \max^{\gamma_1}\{0, h_{\mathcal{W}}(x_c, w)\} \tag{3c}$$
$$- \alpha_2 \max^{\gamma_2}\{0, h_{\mathcal{W}}(x_c, w)\},$$

$$\frac{\partial h_{\mathcal{F}}(x_{ij}, f_{ij})}{\partial x_{ij}} u_{ij} \leq \delta_1 h_{\mathcal{F}}(x_{ij}, f_{ij})$$
$$- \alpha_1 \max^{\gamma_1}\{0, h_{\mathcal{F}}(x_{ij}, f_{ij})\} \tag{3d}$$
$$- \alpha_2 \max^{\gamma_2}\{0, h_{\mathcal{F}}(x_{ij}, f_{ij})\},$$

$$\frac{\partial h_{\mathcal{D}}(x_{ij})}{\partial x_{ij}} u_{ij} \geq - \delta_2 h_{\mathcal{D}}(x_{ij}), \tag{3e}$$

$$\frac{\partial h_{\mathcal{O}}(x_i)}{\partial x_i} u_i \geq - \delta_2 h_{\mathcal{O}}(x_i), \tag{3f}$$

where $z = [u^T, \delta^T]^T \in \mathbb{R}^{2 \times r + 2}$ are decision variables, in which $\delta = [\delta_1, \delta_2] \in \mathbb{R}^2$ are slack variables, H is a diagonal matrix with positive constant elements, $Q = [\mathbf{0}_{2 \times r}, w_{\delta_1}, 0]$ where $w_{\delta_1} \in \mathbb{R}^+$ is a penalizing scalar of the slack variable δ_1, $u_{\max} \in \mathbb{R}^+$ defines the control limit of the system, and $u_{ij} = u_i - u_j$ for any $i, j \in \{1, 2, \cdots, r\}$, denote the control difference between two robots $i \neq j$. Sublevel sets of $h_{\mathcal{W}}(x, w) = \|x - w\|^2 - d_G^2$ and $h_{\mathcal{F}}(x_{ij}, f_{ij}) = \|x - f_{ij}\|^2 - d_F^2$ represent desired centroid position $w \in \mathbb{W}$ and swarm formation $f \in \mathbb{F}$ for the robot swarm respectively, where $d_G, d_F \in \mathbb{R}^+$ are tolerance thresholds. Superlevel sets of $h_{\mathcal{D}}(x) = \|x\|^2 - d_O^2$ and $h_{\mathcal{O}}(x) = 1 - (x - \eta)^T P(x - \eta)$ denote collision-free sets with other robots and local obstacle-free sets respectively, where $d_O \in \mathbb{R}^+$ is the minimal distance among agents and $\eta \in \mathbb{R}^n$ and $P \in \mathbb{R}^{n \times n}$ are constant parameters determined by the environment. Constant parameters α_1, α_2, γ_1, γ_1 are chosen as $\alpha_1 = \alpha_2 = \frac{\mu \pi}{2 T_{ud}}$, $\gamma_1 = 1 + \frac{1}{\mu}$, $\gamma_2 = 1 - \frac{1}{\mu}$ with user-defined constants $\mu > 1$ and T_{ud}.

To achieve the desired waypoints w and f, fixed-time CLF ensures the convergence to a given set within user-

defined time T_{ud}, and the concrete formulation can be found in [5]. Constraints (3c) and (3d) are formulated in the form of fixed-time CLF, which respectively drive the swarm system to reach a waypoint $w \in \mathbb{W}$ and achieve a formation $f = \{f_{ij}\}_{\frac{r(r-1)}{2}}$ that corresponds to the chosen next abstract state (w, f), with $w \in \mathcal{W}$, $f \in \mathcal{F}$. As for runtime safety, CBF is an effective tool to ensure the set invariance of a dynamic system in a given set, which is often used to ensure system safety of collision avoidance. In the form of control inputs condition for CBF [4], constraints (3f) and (3e) attempt to keep the robots within the safety sets $\mathcal{D}$ and $\mathcal{O}$ to avoid collisions with each other and with the obstacles in the environment. This corresponds to the *collision avoidance* objective in Sec. II-C. Note that the *reactivity* objective has been achieved already in the symbolic control in Sec. III-B.

2) Specification Satisfaction Analysis: If the QP in Eq. (3) has a feasible solution $u(t)$ and the slack variable δ_1 remains negative, the obtained control inputs $u(t)$ guarantee that the swarm reaches the given waypoint (w, f) within a maximum time T_{ud} [5]. This indicates that, for any waypoints and desired formations that satisfy the specification ψ, the swarm can track them within a finite predefined timing bound. Although the swarm may traverse other positions and formations between two waypoints, it ultimately reaches the next waypoint (w, f) within a strict timing. This means that, with the tolerance of the transient states of the swarm with a strict timing bound, the behavior of the swarm satisfies the given task specification ψ. Moreover, the constraints (3e) and (3f) guarantee that the system never traverses to dangerous regions. Therefore, we can claim that *Sub-Problem 2* is solved if the QP in Eq. (3) is feasible.

If the QP in Eq. (3) is infeasible, it might be caused by the constraints encoding the impractical waypoints or formations. In this case, the infeasibility can be resolved by pruning the transition function δ of the abstraction model $\mathcal{T}$ as developed in Sec. III-B. Specifically, the waypoints leading to the infeasible QP are recognized as infeasible transitions in the abstraction model $\mathcal{T}$ and are eliminated from δ. This is iteratively performed until all transitions in a synthesized strategy ensure the feasibility of the QP.

IV. CASE STUDY

In this section, we use a swarm navigation control case in simulation to validate the efficacy of our proposed framework and solution. Consider a homogeneous swarm system that contains $r = 3$ quadrotor robots moving in a two-dimensional planar environment $\mathbb{X} \subset \mathbb{R}^2$, where $\mathbb{X}$ is a $5\,\mathrm{m} \times 5\,\mathrm{m}$ square area. The dynamic model of the i-th robot of the swarm system is given as the following single integrator $\dot{x}_i(t) = u_i(t)$, $i = 1, 2, 3$, where $x_i(t), u_i(t) \in \mathbb{R}^2$ are the position and control input of robot i at time $t \in \mathbb{R}_{\geq 0}$. In this setting, the workspace containing the possible positions of the centroid is also $\mathbb{X}$, i.e., $\mathbb{W} := \mathbb{X}$.

The terrain of the environment is illustrated in Fig. 1. The environment is split into a 5×5 grid which generates 25 even square blocks, each with a size of $1\,\mathrm{m} \times 1\,\mathrm{m}$. The yellow block is the starting point of the robot swarm. The

(a) "battery": True   (b) "battery": True   (c) "battery": True   (d) "battery": False   (e) "battery": True
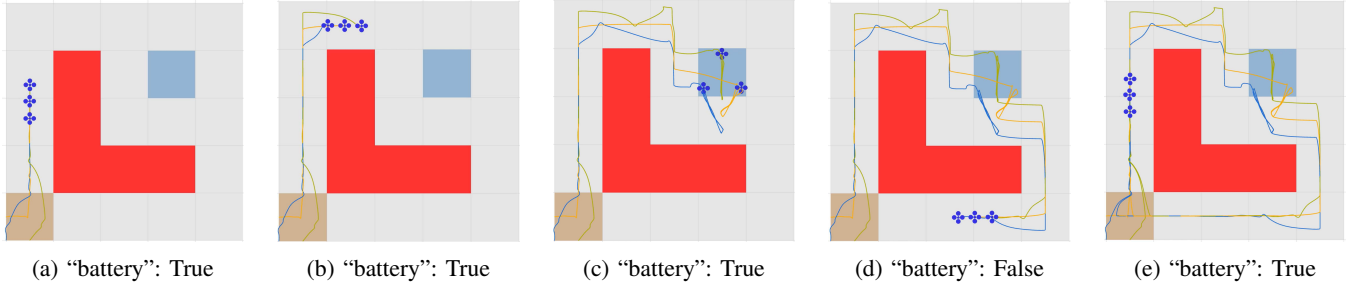
Fig. 1: The planar view of the environment and the robot trajectories in a simulation run, as time changes (left to right).

blue block is the navigation goal that the swarm needs to reach ultimately. The red blocks are the obstacles that the robots should avoid. All waypoints in $\mathcal{W}$ are assigned as the center of the accessible square partitions. Also, we determine an abstract formation set including three different formations $\mathcal{F} = \{f_1, f_2, f_3\}$ by partitioning $\mathbb{F}$, where f_1, f_2, f_3 represent a horizontal formation (as shown in Fig. 1b and Fig. 1d), a vertical formation (as shown in Fig. 1a and Fig. 1e), and a triangle shape formations (Fig. 1c), respectively. The definitions of $\mathcal{W}$ and $\mathcal{F}$ can be found in our online document on [22].

The DFTS $\mathcal{T} = \{S, s_0, A, \delta, \mathsf{AP}, \mathcal{L}\}$ as the abstract model of the quadrotor swarm is realized as follows. The state space $S = \mathcal{W} \times \mathcal{F}$, where $\mathcal{W}$, $\mathcal{F}$ are realized as finite sets with 25 and 3 elements, respectively. The transition relation δ is represented as a matrix and can also be found in our Github repository [22]. The main principles we use to construct the transition relation are as follows.

- When the swarm passes by a narrow corridor, it should switch to the thinnest formation to fit its direction.
- The swarm should not enter an obstacle (red) region.

The atomic proposition set is $\mathsf{AP} = \mathsf{AP}_w \cup \mathsf{AP}_f \cup \mathsf{AP}_e$, where $\mathsf{AP}_w = \{\text{freespace, home, goal, obstacle}\}$, $\mathsf{AP}_f = \{\text{horizon, vertical, triangle}\}$, and $\mathsf{AP}_e = \{\text{battery}\}$. The label mapping is $\mathcal{L} = \{\mathcal{L}_w, \mathcal{L}_f, \mathcal{L}_e\}$. Mapping $\mathcal{L}_w$ labels the yellow block in Fig. 1 as "home", the blue block as "goal", the red blocks as "obstacle", and all other blocks as "freespace". Mapping $\mathcal{L}_f$ is defined such that $\mathcal{L}_f(f_1) = $ "horizon", $\mathcal{L}_f(f_2) = $ "vertical", and $\mathcal{L}_f(f_3) = $ "triangle". Then, mapping $\mathcal{L}_e$ gives "battery" if all the batteries of the robots are charged. The abstract model is visualized in Fig. 2, where the x-y planes along the formation axis show the planar view of the environment for different formations. Thus, Fig. 2 clearly shows the transition between the 25×3 states of the DFTS.

The task is interpreted in the following natural language.

1) The swarm should infinitely visit "goal" in "triangle" formation, as long as "battery" is true.
2) All robots should avoid entering regions with obstacles.
3) The swarm should go back "home" to recharge once "battery" becomes false.

It can be specified as an LTL formula $\psi = \psi_e \rightarrow \psi_s$, where $\psi_e := \Box(\neg\text{battery} \wedge \text{home} \rightarrow \bigcirc\text{battery}) \wedge \Box(\neg\text{battery} \wedge \neg\text{home} \rightarrow \bigcirc\neg\text{battery})$ and $\psi_s := \Box\neg\text{obstacle} \wedge \Box\Diamond(\text{goal} \wedge \text{triangle}) \wedge \Box\Diamond\text{battery}$. The runtime safety requirements are
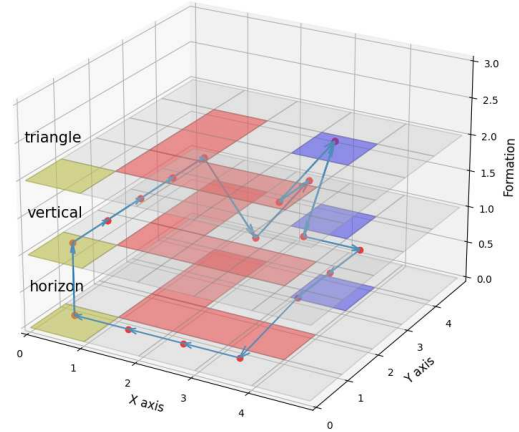

Fig. 2: The visualization of the states of the abstract model. The three planes distributed along the z-axis are the environment with waypoints corresponding to three formations. Each small square block is an abstract state. The blue line denotes the transition of the abstract states in a simulation run. The passing waypoints are marked as red dots.

formulated as bounded sets defined in Sec. III-C and encoded in the QP problem (3) for which the parameters can be found in our Github repository [22]. We give two important parameters $T_{ud} = 4\,\text{s}$ and $u_{\max} = 5\,\text{m/s}$. The synthesis of the symbolic control strategy is solved using an off-the-shelf LTL toolbox, TuLiP [21]. The QP problem is solved using the CasADi library [23] with the ipopt solver on a commercial laptop with CPU i7-10750H.

The trajectories of the robots in a simulation run are shown in Fig. 1. The swarm starts at "home" with the "vertical" formation. After leaving "home", the swarm goes along the horizontal corridor in the "horizon" formation, as shown in Fig. 1a, since the narrow space does not allow other formations. In Fig. 1b, the robot turns right and switches to the "horizon" formation to pass the short horizontal corridor. After reaching the "goal" in the open space, it switches to the "triangle" formation as specified by ψ, as shown in Fig. 1c. When any robots have low power ("battery" signal is false) as shown in Fig. 1d, the swarm goes back to "home" to recharge. Once it gets charged at "home" and the "battery" signal is true again, the robot resumes its previous task to navigate itself to the "goal" again, as shown in Fig. 1e. From Fig. 1, we can see that

the controlled swarm ensures an obstacle-free trajectory when approaching the desired task. Also, proper formations are automatically switched to traverse narrow areas. The resulting behavior of the robot swarm completely satisfies the LTL specifications. This is also reflected by Fig. 2 which visualizes the trajectory of the robot swarm in the abstract space. A video demonstration of this use case is accessible at `https://www.youtube.com/watch?v=r1aecBOeDq0`.

Fig. 3 indicates the satisfaction of runtime safety requirements, where the robot trajectories given a new waypoint and a desired formation are shown. In this case, a robot swarm from the initial position O is required to achieve a "triangle" formation and its center simultaneously reaches the green rounded region within $T_{ud} = 4\,\mathrm{s}$. During this period, all robots should avoid collision with the red rounded obstacle. In Fig. 3, the trajectories of the robots are drawn as solid lines and the formation of the swarm is in dotted lines. It is shown that the robots successfully avoid the obstacle and finally reach the waypoint at a tolerable range. The ultimate formation is "triangle" and the reaching time is within T_{ud}. This study shows that the robot controller solved from the QP problem strictly ensures not only the runtime safety requirements but also the fixed-time convergence condition.
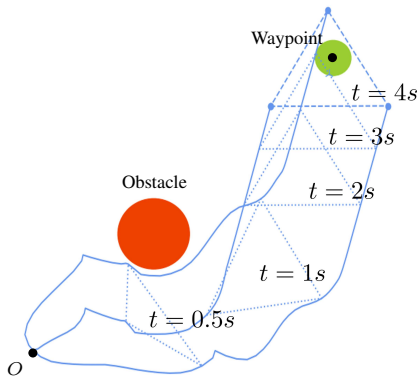

Fig. 3: The satisfaction of the runtime safety requirements.

V. CONCLUSION

In this paper, we develop a formal-method-based framework for multi-robot swarm systems to design a reactive controller for the autonomous navigation task with automated formations. Under the synthesized symbolic controller for the abstract model, the QP-based control refinement approach can ensure that the behavior of the swarm system satisfies the LTL specification with runtime guarantees. In the future, the framework will be expected to extend to more robotic applications with more complicated specifications.

REFERENCES

[1] L. Chen and Z. Sun, "Gradient-based bearing-only formation control: An elevation angle approach," *Automatica*, vol. 141, p. 110310, 2022.

[2] 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," *The International Journal of Robotics Research*, vol. 32, no. 8, pp. 889–911, 2013.

[3] H. Kress-Gazit, G. E. Fainekos, and G. J. Pappas, "Temporal-logic-based reactive mission and motion planning," *IEEE Transactions on Robotics*, vol. 25, no. 6, pp. 1370–1381, 2009.

[4] A. D. Ames, X. Xu, J. W. Grizzle, and P. Tabuada, "Control barrier function based quadratic programs for safety critical systems," *IEEE Transactions on Automatic Control*, vol. 62, no. 8, pp. 3861–3876, 2017.

[5] K. Garg, E. Arabi, and D. Panagou, "Fixed-time control under spatiotemporal and input constraints: A quadratic programming based approach," *Automatica*, vol. 141, p. 110314, 2022.

[6] A. D. Ames, X. Xu, J. W. Grizzle, and P. Tabuada, "Control barrier function based quadratic programs for safety critical systems," *IEEE Transactions on Automatic Control*, vol. 62, no. 8, pp. 3861–3876, 2017.

[7] L. Wang, A. D. Ames, and M. Egerstedt, "Safety barrier certificates for collisions-free multi-robot systems," *IEEE Transactions on Robotics*, vol. 33, no. 3, pp. 661–674, 2017.

[8] X. Tan and D. V. Dimarogonas, "Distributed implementation of control barrier functions for multi-agent systems," *IEEE Control Systems Letters*, vol. 6, pp. 1879–1884, 2021.

[9] L. Lindemann and D. V. Dimarogonas, "Control barrier functions for signal temporal logic tasks," *IEEE control systems letters*, vol. 3, no. 1, pp. 96–101, 2018.

[10] M. Srinivasan and S. Coogan, "Control of mobile robots using barrier functions under temporal logic specifications," *IEEE Transactions on Robotics*, vol. 37, no. 2, pp. 363–374, 2021.

[11] D. Sun, J. Chen, S. Mitra, and C. Fan, "Multi-agent motion planning from signal temporal logic specifications," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 3451–3458, 2022.

[12] M. Kloetzer and C. Belta, "Temporal logic planning and control of robotic swarms by hierarchical abstractions," *IEEE Transactions on Robotics*, vol. 23, no. 2, pp. 320–330, 2007.

[13] Y. E. Sahin, P. Nilsson, and N. Ozay, "Multirobot coordination with counting temporal logics," *IEEE Transactions on Robotics*, vol. 36, no. 4, pp. 1189–1206, 2020.

[14] F. Djeumou, Z. Xu, and U. Topcu, "Probabilistic swarm guidance subject to graph temporal logic specifications." in *Robotics: Science and Systems*, 2020.

[15] L. Lindemann and D. V. Dimarogonas, "Control barrier functions for multi-agent systems under conflicting local signal temporal logic tasks," *IEEE control systems letters*, vol. 3, no. 3, pp. 757–762, 2019.

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

[17] C. Belta, B. Yordanov, and E. Gol, *Formal Methods for Discrete-Time Dynamical Systems*. Springer, 01 2017, vol. 89.

[18] R. Bloem, B. Jobstmann, N. Piterman, A. Pnueli, and Y. Sa'ar, "Synthesis of reactive(1) designs," *Journal of Computer and System Sciences*, vol. 78, no. 3, pp. 911–938, 2012, in Commemoration of Amir Pnueli.

[19] I. Filippidis and R. M. Murray, "Symbolic construction of gr (1) contracts for systems with full information," in *2016 American Control Conference (ACC)*. IEEE, 2016, pp. 782–789.

[20] T. Wongpiromsarn, U. Topcu, N. Ozay, H. Xu, and R. M. Murray, "Tulip: a software toolbox for receding horizon temporal logic planning," in *Proceedings of the 14th international conference on Hybrid systems: computation and control*, 2011, pp. 313–314.

[21] I. Filippidis, S. Dathathri, S. C. Livingston, N. Ozay, and R. M. Murray, "Control design for hybrid systems with tulip: The temporal logic planning toolbox," in *2016 IEEE Conference on Control Applications (CCA)*, 2016, pp. 1030–1041.

[22] S. Qi, "Ltl-formation-control," *GitHub*, 2023, available at https://github.com/Miracle-qi/LTL-Formation-Control.

[23] J. A. E. Andersson, J. Gillis, G. Horn, J. B. Rawlings, and M. Diehl, "CasADi – A software framework for nonlinear optimization and optimal control," *Mathematical Programming Computation*, vol. 11, no. 1, pp. 1–36, 2019.