Sampling-Based Multi-Agent Path Planning Guided by Spatio-Temporal Logic Mission Objectives

Vidisha Kudalkar[1], Sujit Ponguluri[1], Anand Balakrishnan[2], Jyotirmoy V. Deshmukh[1]

[1]University of Southern California
[2]University of Texas at Austin
kudalkar@usc.edu, pongulur@usc.edu, anand.balakrishnan@austin.utexas.edu, jdeshmuk@usc.edu

Abstract

Multi-agent systems operate in shared environments where they must communicate and coordinate to accomplish complex missions. Designing motion plans that guarantee such cooperative behavior is challenging, particularly in continuous state spaces with tightly coupled spatial interactions. Sampling-based planners such as Probabilistic Roadmaps (PRM) scale well to high-dimensional domains but lack mechanisms to enforce mission-level behavioral requirements. Spatio-Temporal Reach and Escape Logic (STREL) enables reasoning about dynamic spatial relations among agents, making it well-suited for specifying coordinated multi-agent tasks. We construct dynamics-aware PRMs for each agent and use them to build a timeless abstraction of the joint multi-agent transition system. The mission objectives are represented using STREL and are symbolically encoded using an SMT solver to synthesize joint trajectories that are both dynamically consistent and logically compliant with the specification. We demonstrate the effectiveness of our approach in the *SwarmLab* drone swarm simulator, where the framework consistently generates coordinated and specification-compliant paths.

Introduction

Mission autonomy for multi-agent systems (MAS) requires specifying and enforcing coordinated tasks while preserving the safety of individual agents. In continuous environments, such as aerial swarms deployed for wildfire monitoring or search and rescue, these mission objectives extend beyond simple collision-free reachability. They require rigorous adherence to spatial goals and strict time bounds, all while accounting for dynamically evolving communication and sensing topologies. Consequently, agents must not only navigate from start to goal but often maintain formations, guarantee connectivity, or sequentially visit targets. As the dimensionality of the state space increases with the number of agents, designing motion plans that are both dynamically feasible and compliant with such high-level mission specifications becomes a computationally challenging problem.

One common approach for specifying complex spatial and temporal task objectives is to use logical formalisms such as Linear Temporal Logic (LTL) (Pnueli 1977) and Signal Temporal Logic (STL). However, these logics treat space as yet another predicate (through the use of region labels), and do not directly express spatial relations as required by MAS. Recent spatio-temporal logics such as Spatio-Temporal Reach and Escape Logic (STREL) (Bartocci et al. 2017; Nenzi et al. 2022) and STL with Graph Operators (STL-GO) (Zhao et al. 2025) allow modeling the spatial structure of the environment and the interaction between agents as first-class objects, enabling a more direct representation of mission autonomy requirements.

For example, consider a wildfire rescue mission where a fleet of drones must locate survivors and maintain a relay chain of at most two hops to a ground base station, and every drone along that chain must remain outside the fire perimeter to ensure the relay is operationally safe. This cooperative spatial requirement can be expressed compactly in STREL as: $\mathbf{G}_{[0,T]}\big[\text{Survivor} \Rightarrow (\neg\text{Fire } \mathbf{R}^{hops}_{\leq 2} \text{ Base})\big]$. The Reach operator simultaneously constrains the *length* of the relay chain and the *safety of every intermediate agent* along it – a requirement that standard temporal logics cannot express, since LTL and STL lack any notion of path length through a dynamic agent graph, and static spatial logics such as SSTL (Bortolussi and Nenzi 2014; Nenzi et al. 2015) and SpaTeL (Haghighi et al. 2015) cannot reason over topologies that change as agents move. The planning problem with respect to such spatio-temporal specifications, i.e., synthesizing trajectories that realize these inter-agent spatial requirements in continuous environments, remains largely unexplored.

Planning in continuous multi-agent environments is challenging because agent dynamics are often non-linear or non-holonomic, and MILP-based and SMT-based planning approaches for LTL or STL rely on linearity assumptions that do not hold for such dynamics. Sampling-based planners, such as Rapidly-Exploring Random Trees (RRT) and Probabilistic Roadmaps (PRM), are effective for planning in high-dimensional continuous spaces (Kavraki et al. 1996; LaValle 1998) and handle non-linear dynamics while constructing trees/graphs. Researchers have combined such planners with temporal logics (Vasile, Li, and Belta 2020; Karlsson, Barbosa, and Tumova 2020; Vasile, Raman, and Karaman 2017; Bhatia, Kavraki, and Vardi 2010); however, there is limited work on combining sampling-based planners with multi-agent specifications. Parallel efforts utilizing SMT solvers have bridged high-level planning with low-level control,

Copyright © 2026, Association for the Advancement of Artificial Intelligence (www.aaai.org). All rights reserved.

enabling dynamically feasible trajectories for LTL tasks (Shoukry et al. 2017; Saha et al. 2014). Yet, these methods are often confined to single-agent reach-avoid scenarios (Shoukry et al. 2016) or basic multi-robot collision avoidance. Crucially, current methods fail to capture the evolving inter-agent relations required for multi-robot coordination.

We propose a centralized planning framework bridging the gap between continuous dynamics and high-level logic. We first construct a dynamics-aware PRM for each agent, generating a "timeless" discrete abstraction that captures dynamic feasibility without fixed timing. These PRM-derived transition systems and the STREL specification are encoded into a query that is solved by an SMT solver (De Moura and Bjørner 2008) to synthesize a bounded sequence of joint actions. The resulting trajectories are *correct-by-construction*, guaranteed to be consistent with agent dynamics and satisfy given spatio-temporal requirements.

Our main contributions in this paper are as follows:
- We present a novel framework for sampling-based motion planning with STREL, enabling the formal synthesis of multi-agent behaviors in continuous environments.
- We develop a dynamics-aware PRM construction for agents that captures non-linear constraints, providing a structured abstraction of feasible motions.
- We encode the multi-agent PRM and a given STREL specification φ as an SMT query, which, if satisfiable, yields an open loop plan satisfying φ.
- We demonstrate the approach in the *SwarmLab* drone swarm simulator, showing that our framework consistently generates coordinated, specification-compliant plans.

Preliminaries

We consider a multi-agent system (MAS) comprising N mobile agents, $\mathcal{A} = \{A^1, \ldots, A^N\}$, operating within a compact metric space $\mathcal{L}$ equipped with a metric d (typically Euclidean $\mathbb{R}^2$ or $\mathbb{R}^3$). In such a system, we model two aspects: how an agent's state evolves over time, and how specific relations between pairs of agents evolve over time. The state of each agent encapsulates its spatial location alongside physical quantities (e.g., velocity, battery) and task-specific variables (e.g., current goals). State evolution is governed by agent-specific dynamical models. Relations such as spatial proximity or connectivity in a communication topology are symmetric binary relations, and we can model these using time-varying undirected graphs. We first define agent dynamics and joint transition systems, then introduce time-varying spatial graphs and the concepts used in STREL.

Multi-agent Dynamical System. Let S^i and U^i denote the state and action spaces of the i^{th} agent. Then, its motion model or dynamics is described by Eq. (1).

$$s^i_{t+1} = g^i(s^i_t, u^i_t). \tag{1}$$

Here, $s^i_t \in S^i$ is the state of the i^{th} agent at time t, $u^i_t \in U^i$ is its action, and s^i_{t+1} is the resulting state at time $t+1$. For any agent motion model, we can also equivalently describe the agent's set of behaviors using a transition system model for the agent. This is described as a tuple $(S^i, U^i, S^i_{\text{init}}, \delta^i, L)$, where, $S^i_{\text{init}} \subseteq S^i$ is a set of initial states of the agent,

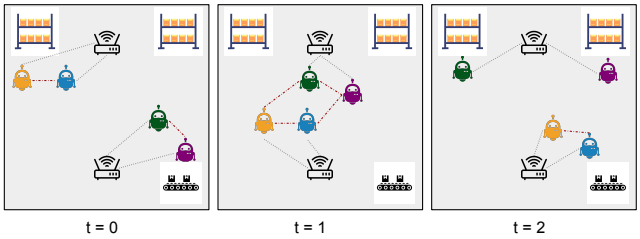


Figure 1: Example of agents navigating in a warehouse environment. The figure shows how agent relations evolve dynamically over time.

$\delta^i \subseteq S^i \times U^i \times S^i$ is a non-empty set of transitions, and $L : S^i \to 2^{AP}$ is a labeling function mapping each agent state to some subset of atomic propositions AP. Each atomic proposition represents a subset of agent states that satisfy this proposition.

We note that $(s^i_t, u^i_t, s^i_{t+1}) \in \delta^i$ iff $s^i_{t+1} = g^i(s^i_t, u^i_t)$, i.e., the transitions are consistent with the dynamics in Eq. (1). A state represents the instantaneous configuration of an agent, including aspects such as its physical quantities such as its location, orientation, velocity, amount of charge on its battery, etc. Both the state space S and action space U can be finite or infinite. We define the function $\ell : S \to \mathcal{L}$ that projects the state of the agent to its location.

From individual agent transition systems, we can define a multi-agent transition system as follows:

Definition 1 (Multi-Agent Transition System) *The multi-agent transition system (MATS) consisting of N agents is defined as a tuple $\mathcal{T} = (\mathbf{S}, \mathbf{U}, \mathbf{S_{init}}, \boldsymbol{\delta}, \mathbf{L})$, where: $\mathbf{S} = S^1 \times \ldots \times S^N$ is the set of joint states; $\mathbf{U} = U^1 \times \ldots \times U^N$ is the set of joint actions; $\mathbf{S_{init}} = S^1_{\text{init}} \times \ldots \times S^N_{\text{init}}$ is the set of joint initial states; $\boldsymbol{\delta} \subseteq \mathbf{S} \times \mathbf{U} \times \mathbf{S}$ is the set of joint transitions, where, $((s^1_t, \ldots, s^N_t), (u^1_t, \ldots, u^N_t), (s^1_{t+1}, \ldots, s^N_{t+1})) \in \boldsymbol{\delta}$ if $\forall i$: $(s^i_t, u^i_t, s^i_{t+1}) \in \delta^i$; $\mathbf{L} : \mathbf{S} \to \left(2^{AP}\right)^N$ is the labeling function mapping each joint state to a set of subsets of atomic propositions AP, where each subset contains the atomic propositions that hold for each agent in the joint state.*

Multi-Agent Spatial Graphs. A multi-agent dynamic spatial graph is a time-stamped sequence of graphs $G_0, \ldots, G_T$, for all $t \in [0, T]$. $G_t = (V, E_t)$ is an undirected graph expressing some symmetric, binary relation between agents. The vertex set V of this graph is the set of agents $\mathcal{A}$ i.e $V = \{A^1, \ldots, A^N\}$. We assume that there is a *graphing function* that at any given time t defines the edge set E_t for the dynamic graph G_t. Below we give some specific examples of such graphing functions that produce E_t. We note that the specific definitions of the graphing functions can play an important role in the problem we wish to solve.

ϵ-Proximity Graphs. Recall that ℓ is the function mapping an agent's state to its location, i.e, a point in the metric space $(\mathcal{L}, d)$. Then, the ϵ-proximity graph ϵ-$G^{\text{prox}}_t = (V, E^{\text{prox}}_t)$ is defined as: $(A^i, A^j) \in E^{\text{prox}}_t$ iff $d(\ell(s^i_t), \ell(s^j_t)) < \epsilon$.

ϵ-Connectivity Graphs. Assume that there is a fixed set of

beacons $\{B_1, \ldots, B_M\}$ (modeled as points in the metric space), the connectivity graph ϵ-$G^{\text{conn}} = (V, E_t^{\text{conn}})$ at time t is defined as: $(A^i, A^j) \in E_t^{\text{conn}}$ iff $\exists m : d(A^i, B_m) < \epsilon \wedge d(A^j, B_m) < \epsilon$ i.e., both agents can communicate only if they are close enough to the same beacon.

We assume that there is some graphing function M that takes as input the time t, the locations of the agents at t and relevant ancillary information at time t (such as obstacles, beacons, etc.) and for each pair of agents A^i, A^j, outputs the adjacency relation E_t at that time t.

Example 1 *Consider a MAS with four mobile agents in a warehouse environment as shown in Fig. 1. The dotted lines visualize active communication links, distinguishing between robot-to-network connections (black) and robot-to-robot connectivity (red). The figure depicts the temporal evolution of the fleet's trajectory: at $t = 0$, the agents begin in separated clusters; at $t = 1$, they converge centrally to establish a multi-hop communication relay and finally, at $t = 2$, the agents disperse to reach their assigned goal states at the shelves and conveyor belt. This demonstrates that the joint trajectories encode the synchronous evolution of all agents under their respective local dynamics.*

Definition 2 (Spatio-Temporal Multi-Agent Trajectory) *Let $\mathbf{s}_t = (s_t^1, \ldots, s_t^N)$ and $\mathbf{u}_t = (u_t^1, \ldots, u_t^N)$ be the shorthand for the joint state and joint action, respectively. A* spatio-temporal multi-agent trajectory *ρ over a finite horizon $T \in \mathbb{Z}_{\geq 0}$ is the sequence of states, actions, and spatial graphs denoted as $\rho = (\mathbf{s}_0, G_0), \mathbf{u}_0, \ldots, (\mathbf{s}_{T-1}, G_{T-1}), \mathbf{u}_{T-1}, (\mathbf{s}_T, G_T)$, where for each time t, G_t is produced by the graphing function M from $\mathbf{s}_t$ and other ancillary information, and for all $t \in [0, T-1]$, $(\mathbf{s}_t, \mathbf{u}_t, \mathbf{s}_{t+1}) \in \boldsymbol{\delta}$.*

Definition 3 (Spatial Route, Spatial Distance Function) *Recall that ℓ is the function that projects an agent's state to its location. A spatial route $route_t$ at time t from agent A^{i_1} to agent A^{i_k} on a graph $G_t = (V, E_t)$ is a finite spatial path of k hops, represented as the sequence $route_t(A^{i_1}, A^{i_k}) = \langle \ell(s_t^{i_1}), \ell(s_t^{i_2}), \ldots, \ell(s_t^{i_k}) \rangle$. Here, $(A^{i_j}, A^{i_{j+1}}) \in E_t, \forall j \in \{1, \ldots, k-1\}$. The set of routes along the graph (V, E_t) from agent A^{i_1} at time t is denoted as $Routes_t(E_t, A^{i_1})$. We assume a spatial distance function f that maps a given $route_t(A^{i_1}, A^{i_k})$ to its length.*

Remark 1 *In general, the multi-agent graph can be* weighted*, i.e., $G_t = (V, E_t)$ can be defined such that E_t maps each pair of nodes (i.e., agents) to some value from the semiring $(W, \oplus, \otimes)$. This allows us to compute more general spatial distance functions where the distance along a route is defined as the cumulative value of all edges using the $\otimes$ operator to combine values. Shortest routes can then be defined using the $\oplus$ operator. In this paper, we restrict ourselves to the Boolean $(\min, +)$ semi-ring, i.e., $W = (\{0, 1\}, \min, +)$.*

Spatio-temporal Reach and Escape Logic

Spatio-Temporal Reach and Escape Logic (STREL) (Bartocci et al. 2017; Nenzi et al. 2022) is a formal specification language designed to reason about both temporal and spatial properties of dynamic spatio-temporal systems, which are modeled as dynamic graphs where vertices exhibit temporal behavior and edge relations may change over time. We leverage this formalism to model our multi-agent system, where agents are represented as vertices and time-varying edges encode the dynamic multi-agent graph.

Definition 4 (Syntax of STREL) *The STREL syntax over discrete-time traces is defined recursively as:*

$$\varphi ::= \mu | \neg\varphi | \varphi_1 \wedge \varphi_2 | \varphi_1 \mathbf{U}_{[t_1,t_2]} \varphi_2 | \varphi_1 \mathbf{R}^f_{[d_1,d_2]} \varphi_2 | \mathbf{E}^f_{[d_1,d_2]} \varphi$$

where μ is an atomic predicate; $\neg$, $\wedge$ are standard Boolean connectives; $\mathbf{U}_{[t_1,t_2]}$ is the Until operator over the interval $[t_1, t_2]$, where $t_1, t_2 \in \mathbb{Z}_{\geq 0}, t_1 \leq t_2$; $\mathbf{R}^f_{[d_1,d_2]}$ and $\mathbf{E}^f_{[d_1,d_2]}$ are Reach and Escape spatial operators, where distances $d_1, d_2 \in \mathbb{R}_{\geq 0}, d_1 \leq d_2$ and f is a spatial distance function.

We can derive *disjunction* $\vee$ and *implication* $\Rightarrow$ operators, temporal operators *Eventually* $\mathbf{F}_{[t_1,t_2]}$ and *Globally* $\mathbf{G}_{[t_1,t_2]}$ and spatial operators *Somewhere* $\diamondsuit^f_{\leq d}\varphi$, *Everywhere* $\boxdot^f_{\leq d}\varphi$, and *Surround* $\varphi_1 \circledcirc^f_{\leq d} \varphi_2$. The *Reach* operator $\varphi_1 \mathbf{R}^f_{[d_1,d_2]} \varphi_2$ holds for an agent A^i if there exists a route $route(A^i, A^j)$ from A^i to A^j, such that A^j satisfies φ_2, $f(route(A^i, A^j)) \in [d_1, d_2]$, and all intermediate agent-vertices A^k along $route(A^i, A^j)$ satisfy φ_1. The *Escape* operator $\mathbf{E}^f_{[d_1,d_2]}\varphi$ holds for agent A^i if there exists a $route(A^i, A^j)$ escaping the current region containing agent-vertices that satisfy φ, and $f(route(A^i, A^j)) \in [d_1, d_2]$. The key distinction between the Reach and Escape operators is that the Reach operator measures distance along the traversed spatial path, whereas the Escape operator measures the shortest-path distance between the start and end locations. The *Somewhere* $\diamondsuit$ and *Everywhere* $\boxdot$ operators check satisfaction for some or all agents within distance d, respectively. The *Surround* $\circledcirc$ operator requires an agent in a φ_1-region to be enclosed by a φ_2-region.

Semantics of STREL. The Boolean semantics (Liu et al. 2025) for a discrete-time STREL specification φ is evaluated point-wise for each ego agent A^i and each time t. Given a multi-agent joint trajectory ρ and a multi-agent dynamic graph G_t, the semantics determines whether φ is satisfied or violated by agent A^i and is defined in Table 1: [1]

Definition 5 (Time Horizon of a STREL Formula) *Given a STREL specification φ, the* time horizon *T of φ is the minimum temporal bound needed to determine the satisfaction of φ for the joint trajectory of the MAS.*

Problem Definition

We consider a MAS comprising N agents, each agent modeled by a transition system model $(S^i, U^i, S^i_{\text{init}}, \delta^i, L)$, $i = 1, \ldots, N$. The corresponding multi-agent transition system is $\mathcal{T} = (\mathbf{S}, \mathbf{U}, \mathbf{S_{init}}, \boldsymbol{\delta}, \mathbf{L})$ as described in the Def. (1). Given the MATS for N agents and a bounded-horizon STREL specification φ with horizon T find a joint trajectory $\rho = (\mathbf{s}_0, \mathbf{s}_1, \ldots, \mathbf{s}_T)$ such that $\rho \models \varphi$.

[1] We use $\mathbf{E}^f_{\leq d}$ and $\mathbf{E}^f_{\geq d}$ as shorthand for the $\mathbf{E}^f_{[0,d]}$ and $\mathbf{E}^f_{[d,\infty]}$ respectively, analogous shorthand is used for all spatial operators.

Formula φ	$(G_t, \rho, A^i, t) \models \varphi$
μ	$\mu \in L(s_t^i)$
$\neg\varphi$	$\neg\big((G_t, \rho, A^i, t) \models \varphi\big)$
$\varphi_1 \wedge \varphi_2$	$(G_t, \rho, A^i, t) \models \varphi_1 \ \wedge \ (G_t, \rho, A^i, t) \models \varphi_2$
$\varphi_1 \mathbf{U}_{[t_1,t_2]} \varphi_2$	$\exists t' \in [t+t_1, t+t_2], (G_t, \rho, A^i, t') \models \varphi_2$ $\wedge\, [\forall t'' \in [t, t'),\ (G_t, \rho, A^i, t'') \models \varphi_1]$
$\varphi_1 \mathbf{R}^f_{\leq d} \varphi_2$	$\exists route_t(A^i, A^j) \in Routes_t(E_t, A^i)$, $[f(route_t(A^i, A^j)) \leq d] \wedge [(G_t, \rho, A^j, t) \models$ $\varphi_2] \ \wedge [\forall A^k \in route_t(A^i, A^j)$ s.t. $A^k \neq A^j$, $(G_t, \rho, A^k, t) \models \varphi_1]$
$\mathbf{E}^f_{\geq d} \varphi$	$\exists route_t(A^i, A^j) \in Routes_t(E_t, A^i)$, $[f(route_t(A^i, A^j)) \geq d] \wedge$ $[\forall A^k \in route_t(A^i, A^j), (G_t, \rho, A^k, t) \models \varphi]$

Table 1: Boolean Semantics for STREL

To address this, we propose a method that combines sampling-based motion planning with SMT solving. First, we utilize a sampling-based planner to efficiently generate a graph of feasible motions for each agent, thereby capturing diverse behaviors. These sampled structures are then encoded symbolically alongside the STREL specification φ into an SMT solver. Finally, we extract a valid joint trajectory that satisfies the spatio-temporal mission objectives.

Multi-agent Planning

As described in Eq. (1), we assume that agents have dense state and action spaces. Thus, the MATS defined in Def. 1 is infinite in size. This makes it intractable for planning with respect to discrete-time STREL specifications. In this section, we describe how we can create a *timeless and finite* under-approximation of the MATS which makes it amenable to use with constraint-solving based planning methods. The main idea is to use the sampling-based technique of *Probabilistic Roadmaps* to create this under-approximation. PRMs randomly sample states in the agent's state space, and then connect states s and s' with an edge if there is a steering function that allows the agent to move from s to s'.

We note that the state of agents can include physical attributes such as velocity, orientation, etc. However, for ease of exposition, we consider PRMs where the agent state consists of only the location in the space. The algorithms presented here can be extended to richer state spaces with minimal changes. With this assumption, we assume that the states of the PRM are points in compact metric space $(\mathcal{L}, d)$. Given a set of obstacles $\{O_1, \ldots, O_m\}$, where each $O_j \subseteq \mathcal{L}$ is a connected subset of $\mathcal{L}$, we define the obstacle space as $\mathcal{L}_{\text{obs}} = \bigcup_{i=1}^{m} O_i$, and free space as $\mathcal{L}_{\text{free}} = \mathcal{L} \setminus \mathcal{L}_{\text{obs}}$.

There are many specific instantiations of the PRM algorithm for single-agent settings; we describe a general algorithm for constructing PRMs in Algorithm 1. Initially, the algorithm samples P random states from $\mathcal{L}_{\text{free}}$ (Line 1) and stores them in the set S – this is the set of nodes of the PRM. Then, for each $s \in S$, we find the k-nearest neighbors (denoted as $\mathcal{N}(s)$) of s (Line 4); here nearness is defined based on the metric d. Then, for each neighbor in this set, we check if the convex hull of s and s', i.e. the line segment connecting these points intersects any obstacle. If yes, this neighbor is skipped. If not, then the STEER function attempts to find a $k+1$ step path $s_0, \ldots, s_k$, where $s_0 = s$ and $s_k \approx s'$, and for all $i \in [0, k-1]$, s_{i+1} is the state reached by taking action $u_i \in U$ in state s_i. Each step in this path is at the same discrete-time interval Δt. This is a crucial requirement that allows us to later *synchronize paths* across multiple agents. By $s_k \approx s'$, we mean that $d(s_k, s') < \varepsilon$ for some user-chosen threshold $0 < \varepsilon << 1$. Once such a path is obtained, each transition in this path is added to the set of transitions that comprise the PRM's edge relation (Line 9).

Algorithm 1: Single-agent PRM construction

Require: Number of states P; free space $\mathcal{L}_{\text{free}}$; obstacle regions $\{O_j\}_{j \in J}$; neighbor parameter k;
1: $S \leftarrow$ SAMPLERANDOMSTATES$(P, \mathcal{L}_{\text{free}})$
2: $\delta \leftarrow \emptyset$
3: **for** each $s \in S$ **do**
4: $\quad \mathcal{N}(s) \leftarrow$ K-NEARESTNEIGHBORS(s, S, k)
5: $\quad$ **for** each $s' \in \mathcal{N}(s)$ **do**
6: $\quad\quad$ **if** $\forall j \in J$: convex_hull$(\{s, s'\}) \cap O_j = \emptyset$ **then**
7: $\quad\quad\quad (s_0, u_0, \ldots, s_{p-1}, u_{p-1}, s_p) =$ STEER(s, s') ▷ $s_0 = s$ and $s_p = s'$
8: $\quad\quad\quad$ **for** $i = 0, 1, \ldots, p-1$ **do**
9: $\quad\quad\quad\quad \delta \leftarrow \delta \cup \{(s_i, u_i, s_{i+1})\}$
return S, δ

Multi-agent PRM. We assume that agents are homogeneous, and all agent actions are centrally planned. Under these assumptions, we can reuse a single-agent PRM to explore multi-agent spatio-temporal trajectories. As the initial states of agents may be defined to lie in specific subsets of the state space, the only modification to Algorithm 1 is to change how PRM nodes are initially sampled – in addition to random samples, for each agent, we sample its initial state set, i.e., S^i_{init} and add it to the set of initial states. The rest of the algorithm remains unchanged. We denote the PRM for agent A^i as $\text{PRM}^i = (S^i, \delta^i)$.

A multi-agent spatio-temporal trajectory is implicitly constructed. Recall that $\mathbf{s}_t = (s_t^1, \ldots, s_t^N)$ and $\mathbf{u}_t = (u_t^1, \ldots, u_t^N)$ are joint state and joint action at time t. A *discrete-time* path in the metric space $\mathcal{L}$ is defined as a function $\pi : \mathbb{T} \to \mathcal{L}$. We say that $\mathbf{s}_0 \xrightarrow{\mathbf{u}_0} \mathbf{s}_1 \to \ldots \xrightarrow{\mathbf{u}_{T-1}} \mathbf{s}_T$ is a feasible multi-agent trajectory iff $\forall i$: $s_0^i \xrightarrow{u_0^i} s_1^i \to \ldots \xrightarrow{u_{T-1}^i} s_T^i$ is a valid path in the single-agent PRM.

The multi-agent spatial graph at each timestep t can be obtained by applying the graphing function M to the agents' states and the environment configuration. We adopt the definition of *ϵ-proximity graphs* from the Preliminaries Section. Recall that the ϵ-proximity graphs $\epsilon\text{-}G_t^{\text{prox}} = (V, E_t^{\text{prox}})$ are originally defined over the set of agents $\mathcal{A}$ at a specific time step t, where an edge (A^i, A^j) exists between agents iff $d(\ell(s_t^i), \ell(s_t^j)) < \epsilon$. In our formulation, we leverage the PRM of the agent as a *timeless abstraction* of the transition system; i.e., the states in the PRM are decoupled from the specific time step t. This allows an agent to be in any valid state at any given time t. Therefore, we redefine ϵ-proximity

graph ϵ-$G^{\text{prox}} = (V, E^{\text{prox}})$ as $((A^i, \ell(s^i)), (A^j, \ell(s^j))) \in E^{\text{prox}}$ iff $d(\ell(s^i), \ell(s^j)) < \epsilon$ where s^i, s^j are states of agents A^i, A^j in their respective PRMs. The vertex set consists of all the possible locations in the PRM of each agent, denoted by $V = \{(A^i, \ell(s^i)) \mid \forall\ s^i \in S^i,\ \forall i \in \{1, \dots N\}\}$.

While the underlying PRM contains no explicit temporal information, we synthesize agent trajectories by unrolling the PRM over a discrete time horizon T. The validity of a path $s_0^i \xrightarrow{u_0^i} s_1^i \to \dots \xrightarrow{u_{T-1}^i} s_T^i$ is maintained by imposing transition constraints at each timestep t, ensuring that for any consecutive time steps t and $t+1$, the corresponding states are connected in the underlying PRM.

Symbolic Encoding of the Planning Problem

In this section, we formalize the multi-agent planning problem considered in this work. Our approach constructs two constraints: a *motion-feasibility constraint* Ψ_{motion} obtained by encoding the timeless PRM abstractions for each agent in MATS, and a *specification constraint* Ψ_{spec} obtained by encoding the STREL formula that describes the mission objective. The planning problem is to find a joint trajectory ρ over a bounded time horizon T that satisfies the planning constraint $\Psi_{\text{plan}} = \Psi_{\text{motion}} \wedge \Psi_{\text{spec}}$. Formally, we seek a satisfying assignment for the joint trajectory $\rho = (\mathbf{s}_0, \mathbf{s}_1, \dots, \mathbf{s}_T)$ such that $(\mathbf{s}_0, \mathbf{s}_1, \dots, \mathbf{s}_T) \models \Psi_{\text{plan}}$.

We utilize Z3 (De Moura and Bjørner 2008) as a SMT solver and use λ-*expressions* of the form $\lambda \texttt{Vars}.\psi(\texttt{Vars})$ to denote the symbolic sub-expressions over a tuple of variables `Vars`. In our encoding, we use symbolic variables s, t, and i for the agent's state, the discrete time step, and the agent identifier, respectively. These variables are arguments to the λ-expressions used to recursively construct the constraints. The resulting formulas are therefore *fully symbolic* and suitable for satisfiability solving using an SMT solver.

Encoding Timeless PRMs using SMT Constraints

Since the agent PRM serves as a time-independent abstraction, spatial relations are defined over candidate PRM states rather than a realized trajectory. We symbolically encode the trajectories in PRM $PRM^i = (S^i, \delta^i)$ of each agent A^i by defining the path constraints. We use symbolic variables $x_0^i, x_1^i, \dots, x_T^i$ to represent the states in the trajectory of agent A^i over horizon T. To ensure that this trajectory represents a feasible path, we impose the following constraints.

The *state constraint* ensures that each path variable x_t^i in the trajectory corresponds to a valid PRM state in S^i. The transitions $(s, s') \in \delta^i$ is captured by the *connectivity constraint* using a Boolean predicate $\mathsf{Trans} : S^i \times S^i \to \mathbb{B}$ where $\mathsf{Trans}(s, s') = \top \Leftrightarrow (s, s') \in \delta^i$. Finally, the *initial state constraint* enforces that the trajectory must start from a valid state in S^i_{init}. The trajectory $x_0^i, x_1^i, \dots, x_T^i$ encodes a valid path in PRM iff it satisfies the conjunction of these three constraints, formalized as:

$$\Psi_{\text{path}}(i) := \bigvee_{s \in S^i_{\text{init}}} (x_0^i = s) \wedge \bigwedge_{t=0}^{T} \bigvee_{s \in S^i} (x_t^i = s) \wedge \bigwedge_{t=0}^{T-1} \mathsf{Trans}(x_t^i, x_{t+1}^i)$$

This formulation ensures that the trajectory starts at a valid initial state, remains within the domain of the state space, and respects the connectivity of PRM^i. To enforce motion feasibility for the collective multi-agent system, we require that every agent simultaneously satisfy its respective path constraints. The motion constraint is defined as $\Psi_{\text{motion}} := \bigwedge_{i \in \mathcal{A}} \Psi_{\text{path}}(i)$. This ensures that the joint trajectory $\sigma = (x^1, \dots, x^N)$ is composed of valid paths consistent with each agent's roadmap.

Symbolic Encoding of STREL Specifications

In this section, we present the symbolic encoding of STREL specifications for path planning in MAS. Let s, t, and i be symbolic variables denoting the agent's state, the discrete time step, and the agent identifier, respectively. We define an encoding function $\mathcal{Z}(\varphi, i)$ that maps a STREL formula φ for ego agent i to a Z3 `lambda` term over the symbolic arguments s, t, i as $\mathcal{Z}(\varphi, i) = \lambda(s, t, i).\big[\dots\big]$. The body of the lambda $[\dots]$ is constructed recursively via structural induction on the syntax of φ. We define the encoding for the base cases (atomic propositions), followed by the inductive steps for Boolean, temporal, and spatial operators.

Atomic Proposition. For an agent A^i, the satisfaction set of predicate $\mu \in AP$ is the subset of states where μ holds and is defined as $\texttt{SAT}(\mu, i) = \{s^i \in S^i \mid \mu \in L(s^i)\}$ where s^i is a state of agent A^i in its PRM and L is the labeling function. We encode μ as a disjunction over these concrete states using a Z3 lambda expression as:

$$\mathcal{Z}(\mu, i) = \lambda(s, t, i).\Big[\bigvee_{q \in SAT(\mu, i)} (s = q)\Big]$$

which evaluates to `True` whenever the symbolic state s corresponds to a state labeled with μ. We note that while μ is time-invariant, the variable t is retained in the signature of lambda expression for consistency with other operators.

Negation. The *negation* operator evaluates to `True` iff the subformula is not satisfied at (s, t, i) and is encoded as:

$$\mathcal{Z}(\neg\varphi, i) = \lambda(s, t, i).\Big[\neg\mathcal{Z}(\varphi, i)(s, t, i)\Big]$$

Conjunction. The *conjunction* operator evaluates to `True` iff both subformulas φ_1 and φ_2 are satisfied at (s, t, i). The encoding is as follows:

$$\mathcal{Z}(\varphi_1 \wedge \varphi_2, i) = \lambda(s, t, i).\Big[\mathcal{Z}(\varphi_1, i)(s, t, i) \bigwedge \mathcal{Z}(\varphi_2, i)(s, t, i)\Big]$$

Eventually. The bounded *Eventually* operator $\mathbf{F}_{[a,b]}\varphi$ requires that φ holds at some timestep within the interval $[t+a, t+b]$. The Z3 encoding evaluates to `true` whenever there exists a symbolic time t' within the bounded horizon $[t+a, t+b]$ such that the subformula φ is satisfied by agent i at (s, t', i).

$$\mathcal{Z}(\mathbf{F}_{[a,b]}\varphi, i) = \lambda(s, t, i).\Big[\bigvee_{t'=t+a}^{t+b} \mathcal{Z}(\varphi, i)(s, t', i)\Big]$$

Until. The bounded *Until* operator $\varphi_1 \mathbf{U}_{[a,b]} \varphi_2$ captures a temporal dependency where φ_1 must persist continuously until φ_2 becomes true within the interval $[t+a, t+b]$. The

Algorithm 2: Bounded SMT-based Multi-agent Planning

Require: PRMs $\{PRM^1, \ldots, PRM^N\}$, STREL specification φ, ϵ-proximity graph ϵ-$G^{\text{prox}} = (V, E^{\text{prox}})$

1: $\mathcal{S} \leftarrow$ Z3SOLVER()
2: Create symbolic variables for trajectories $\{x_t^i \mid i \in \{1, \ldots, N\}, t \in \{0, \ldots, T\}\}$, state s, time t, agent i
3: $\Psi_{\text{motion}} \leftarrow \Psi_{\text{path}}(i), i \in \{1..N\}$ ▷ Encode PRMs
4: $\Psi_{\text{spec}} \leftarrow \mathcal{Z}(\varphi, i)(x_0^i, 0, i)$ ▷ Encode STREL formula
5: $\mathcal{S}$.add($\Psi_{\text{motion}} \wedge \Psi_{\text{spec}}$)
6: **if** $\mathcal{S}$.check() == SAT **then**
7: $\quad \mathcal{M} \leftarrow \mathcal{S}$.model()
8: $\quad$ **return** $\rho = \{(\mathcal{M}(x_0^i), \ldots, \mathcal{M}(x_T^i)) \mid i \in 1 \ldots N\}$
9: **else**
10: $\quad$ **return** UNSAT

encoding evaluates to `true` whenever there exists a symbolic time $t' \in [t+a, t+b]$ such that φ_2 holds at (s, t', i) and φ_1 holds continuously at all time steps $t'' \in [t, t')$.

$$\mathcal{Z}(\varphi_1 \mathbf{U}[a,b]\varphi_2, i) = \lambda(s,t,i).\Bigg[\bigvee_{t'=t+a}^{t+b}\Bigg(\mathcal{Z}(\varphi_2, i)(s, t', i) \wedge \Bigg[\bigwedge_{t''=t}^{t'} \mathcal{Z}(\varphi_1, i)(s, t'', i)\Bigg]\Bigg)\Bigg]$$

Reach. The *Reach* operator $\varphi_1 \mathbf{R}_{\leq d} \varphi_2$ requires a valid spatial path $a_0, \ldots, a_k$ starting from agent i ($a_0 = i$) of length k ($1 \leq k \leq d$). The predicate $Link(a_m, a_{m+1})$ enforces spatial connectivity, requiring that consecutive agents share an edge in the underlying ϵ-proximity graph. Semantically, φ_1 must hold for all intermediate agents $a_0, \ldots, a_{k-1}$, while φ_2 must be satisfied by the last agent a_k. The Distinct constraint ensures that all agents in the sequence are unique. The encoding is given as:

$$\mathcal{Z}(\varphi_1 \mathbf{R}_{\leq d} \varphi_2, i) = \lambda(s,t,i).\Bigg[\bigvee_{k=1}^{d}\Bigg(\bigwedge_{m=0}^{k-1} Link(a_m, a_{m+1}) \wedge \bigwedge_{m=0}^{k-1} \mathcal{Z}(\varphi_1, a_m)(s_m, t, a_m) \wedge \mathcal{Z}(\varphi_2, a_k)(s_k, t, a_k) \wedge (a_0 = i) \wedge \text{Distinct}(a_0, \ldots, a_k)\Bigg)\Bigg]$$

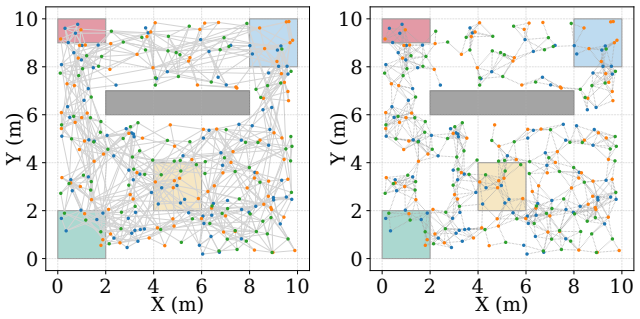
Escape. The *Escape* operator $\mathbf{E}_{\leq d}\varphi$ requires a spatial path of length k (where $0 \leq k \leq d$) originating from agent i, such that every agent on the path satisfies the property φ. We encode *Escape* operator as:

$$\mathcal{Z}(\mathbf{E}_{\leq d}\varphi, i) = \lambda(s,t,i).\Bigg[\bigvee_{k=0}^{d}\Bigg(\bigwedge_{m=0}^{k-1} Link(a_m, a_{m+1}) \wedge \bigwedge_{m=0}^{k} \mathcal{Z}(\varphi, a_m)(s_m, t, a_m) \wedge (a_0 = i) \wedge \text{Distinct}(a_0, \ldots, a_k)\Bigg)\Bigg]$$

Specification Constraint Instantiation. These recursive encodings construct a lambda function $\mathcal{Z}(\varphi, i)$ representing the logic of the specification. The specification constraint Ψ_{spec} is obtained by instantiating the encoding at the ego agent's initial state and time $t = 0$ given by

$$\Psi_{\text{spec}} := \mathcal{Z}(\varphi, i)(x_0^i, 0, i).$$

This operation effectively grounds the symbolic argument s in the lambda signature with the trajectory state variable x_0^i, and propagates $t = 0$ through the recursive operators.



(a) PRMs for 3 agents (b) Multi agent spatial graph

Figure 2: Experimental setup showing the environment configuration

Multi-Agent Planning Algorithm

Algorithm 2 outlines the planning pipeline, starting with initializing Z3 solver and creating the symbolic variables (Lines 1-2). The encoding proceeds by aggregating the PRM path constraints into Ψ_{motion} (Line 3) and constructing the specification constraint Ψ_{spec} (Line 4) by instantiating the recursive STREL encoding $\mathcal{Z}(\varphi, i)$ on the initial state variables x_0^i at $t = 0$, grounding the high-level logic onto the concrete motion plan. The problem is thus reduced to checking the satisfiability of $\Psi_{\text{plan}} = \Psi_{\text{motion}} \wedge \Psi_{\text{spec}}$ (Line 5). If satisfiable, the solver returns a model $\mathcal{M}$ that maps each symbolic state variable x_t^i to a concrete state in agent PRM, yeilding the trajectory for agent A^i, $\rho_i = (s_0^i, \ldots, s_T^i) = (\mathcal{M}(x_0^i), \ldots, \mathcal{M}(x_T^i))$ (Lines 6 - 10). The resulting joint trajectory ρ represents a time-synchronized plan that is guaranteed to be both kinematically feasible (via PRM connectivity) and logically compliant with the spatio-temporal specification φ.

Case Studies

We present a case study demonstrating our framework for multi-agent path planning under STREL specifications in the *SwarmLab* (Soria, Schiano, and Floreano 2020) drone simulation environment.

Experiment Setup. We conduct a multi-drone navigation case study in the *SwarmLab* simulator in a $10 \times 10\,\text{m}^2$ workspace containing predefined base station, threat or no-fly zone, target region (such as the goal and delivery areas), and static obstacles as shown in Fig. 3. The drones evolve in discrete time with a fixed timestep of $\Delta t = 1\text{s}$ under unicycle dynamics with in-place rotation: $x_{t+1}^i = x_t^i + v_t^i \cos\theta_t^i \Delta t$, $y_{t+1}^i = y_t^i + v_t^i \sin\theta_t^i \Delta t$, $\theta_{t+1}^i = \theta_t^i + \Delta\theta_t$ where agent state $s_t^i = (x_t^i, y_t^i, \theta_t^i)$, control input $u_t^i = v_t^i$, and velocity constraint $v \in [1.0, 2.0]\,\text{m/s}$. Agents are connected when their pairwise distance is within 1.0m. Examples of PRMs and resulting spatial graph are shown in Fig. 2.

Specifications. We evaluate our approach using four STREL specifications that capture mission-level requirements in multi-drone systems. We consider the atomic predicates such as `base`, `goal`, `delivered`, and `threat`, which

Agents	Spec	PRM Construction		Prox. Graph	Planning with SMT	
		Avg. Nodes	Time (s)	Construction (s)	Encoding (s)	Runtime (s)
3	φ_1 : Deliver while being connected	106	0.436	0.003	5.231	0.626
	φ_2 : Persistent connectivity while threatened				4.527	0.802
	φ_3 : Threat Detection and Escape				4.150	0.310
	φ_4 : Reach while persistently connected				3.891	0.246
5	φ_1 : Deliver while being connected	110	0.772	0.008	19.100	2.364
	φ_2 : Persistent connectivity while threatened				15.260	4.701
	φ_3 : Threat Detection and Escape				22.928	2.156
	φ_4 : Reach while persistently connected				21.012	0.278
8	φ_1 : Deliver while being connected	109	1.220	0.020	53.619	6.586
	φ_2 : Persistent connectivity while threatened				50.505	11.074
	φ_3 : Threat Detection and Escape				138.889	4.369
	φ_4 : Reach while persistently connected				110.524	0.836
10	φ_1 : Deliver while being connected	109	1.646	0.034	80.393	9.003
	φ_2 : Persistent connectivity while threatened				85.082	16.251
	φ_3 : Threat Detection and Escape				275.774	7.986
	φ_4 : Reach while persistently connected				237.111	1.267

Table 2: Quantitative results for the multi-agent STREL-guided planner. We report the computational time (in seconds) for the geometric pre-processing (PRM Construction) and the symbolic planning pipeline (Proximity Graph, Encoding, and Solving) across varying team sizes ($N = 3$ to 10) and specifications.

are determined by the agent's region in the environment. The predicate `drone` holds at locations occupied by the drone.

$$\varphi_1 = \mathbf{F}_{[0,10]}\big(\texttt{delivered} \wedge \diamondsuit^{hops}_{[0,2]}(\texttt{drone})\big)$$

$$\varphi_2 = \mathbf{F}_{[0,10]}\Big(\texttt{threat} \wedge \mathbf{G}_{[1,3]}\big(\diamondsuit^{hops}_{[0,2]}(\texttt{drone} \wedge \neg\texttt{threat})\big)\Big)$$

$$\varphi_3 = \mathbf{F}_{[0,10]}\Big(\texttt{threat} \wedge \mathbf{F}_{[1,3]}\big(\mathbf{E}^{hops}_{[0,2]}(\texttt{drone} \wedge \neg\texttt{threat})\big)\Big)$$

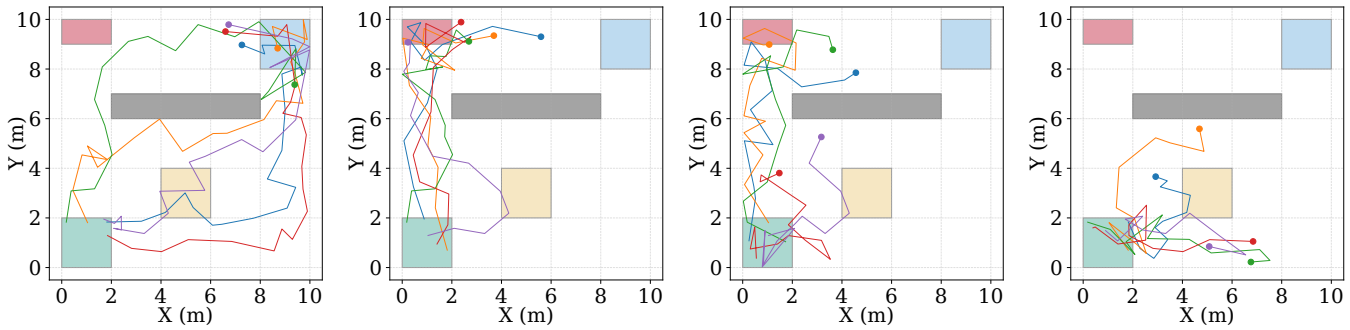
$$\varphi_4 = \big(\texttt{drone}\ \mathbf{R}^{hops}_{[0,2]}\ \texttt{base}\big)\ \mathbf{U}_{[0,10]}\ \texttt{goal}$$

(1) Delivery while maintaining connectivity (φ_1) requires that at any time t within the first 10 timesteps, an agent must reach the delivery zone and simultaneously be connected to another agent within 2 hops. (2) Threat response with persistent nearby safety (φ_2) means that at some time $t \in [0, 10]$ steps, an agent enters the threat region, and in the next $[1, 3]$ timesteps, a drone located safely outside the threat region remains available within two spatial hops. (3) Threat detection followed by guaranteed escape (φ_3) means that within the first 10 timesteps, an agent enters a threat region and an escape route of at most two spatial hops leading to a safe drone location becomes available within the subsequent interval $[1, 3]$ timesteps. (4) Goal achievement under base reachability (φ_4) means a drone needs to be in the goal region within 10 timesteps, and it must maintain a two-hop spatial link to the base till it reaches the goal region.

Scaling with number of agents. Table 2 reports the performance of the multi-agent STREL-guided planner for team sizes $N = 3$ to 10 agents under four specifications φ_1 - φ_4 (Fig. 3). PRM construction is efficient, increasing nearly linearly from 0.436s to 1.646s with team size. Spatial graph generation time is negligible (< 0.04s), while symbolic encoding dominates the total runtime and it increases with the number of agents, growing from roughly 4s for $N = 3$ to 275s for $N = 10$ under φ_3. In contrast, SMT solving times remain comparatively low across all scenarios.

Scaling with time horizon. For specification φ_1, we fixed a team of 10 agents and varied the horizon $T \in \{20, 40, 60, 80, 100\}$. The corresponding encoding times were $80, 99, 106, 122, 139$ seconds, respectively. The SMT solving time remained nearly constant at ≈ 9 seconds for all horizons, increasing slightly to 12s for $T = 100$. These results indicate that increasing the horizon primarily affects encoding due to the growth in symbolic variables and inter-agent constraints, whereas the solving time shows minimal sensitivity to the horizon.

Scaling with specification complexity. We evaluated scalability using 10 agents and horizons $T \in \{20, 40, 60, 80, 100\}$ for two specifications with increasing predicate complexity. The first specification encodes a single threat avoidance objective while reaching the base region $(\mathbf{F}_{[0,T]}(\texttt{drone}\ \mathbf{R}^{hops}_{[0,2]}\ \texttt{base})) \wedge (\mathbf{G}_{[0,T]}\ \neg\texttt{threat})$. The encoding times were $74, 86, 104, 148, 205$ seconds, while the solving times were $14, 24, 27, 54, 67$ seconds, respectively. The second specification introduces multiple threat avoidance objectives $(\mathbf{F}_{[0,T]}(\texttt{drone}\ \mathbf{R}^{hops}_{[0,2]}\ \texttt{base})) \wedge (\mathbf{G}_{[0,T]}\ \neg\texttt{threat}_1 \wedge \neg\texttt{threat}_2 \wedge \neg\texttt{threat}_3)$ where $\texttt{threat}_1, \texttt{threat}_2, \texttt{threat}_3$ are predicates. Under the same conditions, the encoding times were $99, 86, 103, 137, 130$ seconds, while the solving times increased to $10, 41, 70, 116, 234$ seconds, respectively. These results show that adding predicates has a limited impact on encoding time but significantly increases SMT solving time, particularly for larger horizons.



(a) φ_1: Deliver while being connected (b) φ_2: Persistent connectivity while threatened (c) φ_3: Threat Detection and Escape (d) φ_4: Reach while persistently connected

Figure 3: Multi-agent path planning results for specifications φ_1-φ_4 for $N = 5$ agents. The environment consists of five regions of interest: Base station, Goal region, Delivery Zone, Threat Zone, and Obstacle.

Related Work

Sampling-based algorithms such as PRM (Kavraki et al. 1996), RRT (LaValle 1998), and EST (Hsu, Latombe, and Motwani 1997) solve high-dimensional planning problems by constructing collision-free graphs or trees, avoiding explicit state-space discretization. Asymptotically optimal variants like RRT* and PRM* (Karaman and Frazzoli 2011) guarantee convergence to optimal solutions, while kinodynamic extensions integrate system dynamics and control constraints (Kindel et al. 2000; Perez et al. 2012). Despite their geometric and dynamic efficacy, these methods typically lack mechanisms to enforce complex specifications beyond collision avoidance.

Several works have addressed this gap by extending planners to incorporate temporal logic. RRT-based methods have been used to satisfy long-horizon LTL objectives with reactive behaviors (Vasile, Li, and Belta 2020), while spatio-temporal RRT* variants address subsets of STL operators (Karlsson, Barbosa, and Tumova 2020). Other work steers RRT* using biased space–time sampling to progressively grow along the direction of increasing STL satisfaction (Vasile, Raman, and Karaman 2017).

Alternatively, several works have explored optimization-based approaches using SMT or MILP for motion planning. In (Shoukry et al. 2016), a lazy SMT framework is presented that combines workspace abstractions with convex feasibility checks, although it is limited to single-agent reach–avoid tasks. Extensions to multi-robot LTL planning include composing motion primitives (Saha et al. 2014) and integrating SAT with convex optimization (Shoukry et al. 2017). Alternatively, MILP-based techniques using piecewise-linear waypoints enable the synthesis of multi-agent long-horizon trajectories over STL (Sun et al. 2022). However, these methods rely on traditional temporal logics (LTL/STL), which treat space as static regions and struggle to capture the time-varying spatial dependencies inherent to dynamic multi-agent formations.

Recent work introduces expressive spatio-temporal logics that explicitly account for spatial structure and coordinated MAS behaviors. SSTL (Bortolussi and Nenzi 2014; Nenzi et al. 2015), SaSTL (Ma et al. 2020), SpaTeL (Haghighi et al. 2015) have extended STL by introducing some spatial operators, but they operate under static spatial topologies. In contrast, STREL (Bartocci et al. 2017; Nenzi et al. 2022) augments STL with reach and escape operators to reason over a spatial graph, and STL-GO (Zhao et al. 2025) allows counting neighboring agents that satisfy a property along the graph's edges, enabling reasoning over dynamically evolving spatial configurations. Despite this expressivity, a unified framework that combines these dynamic spatio-temporal logics with sampling-based planning for continuous multi-agent systems remains unexplored.

Conclusion

This paper presents a novel framework for multi-agent motion planning that integrates dynamics-aware sampling-based abstractions with formal spatio-temporal mission specifications expressed in STREL. By encoding PRM for each agent together with the STREL specification into an SMT query, we synthesis coordinated multi-agent trajectories that are both dynamically feasible and compliant with the given logical specification. The correctness guarantees of our framework are relative to the sampled roadmap since PRMs are a *timeless under-approximation* of each agent's transition system, any trajectory returned is guaranteed to satisfy both the PRM transition constraints and the STREL specification, but the PRM may not capture all dynamically feasible behaviors of the underlying agent. Increasing the number of sampled nodes improves coverage, but completeness with respect to the true continuous dynamics is not guaranteed. Scalability of the current approach can be compromised with the increasing number of agents, planning horizons and complex specifications. Additionally, we focus on centralized planning with homogeneous agents and we will explore stochasticity, partial observability and heterogeneous fleet of agents in future work.

References

Bartocci, E.; Bortolussi, L.; Loreti, M.; and Nenzi, L. 2017. Monitoring mobile and spatially distributed cyber-physical

systems. In *Proceedings of the 15th ACM-IEEE International Conference on Formal Methods and Models for System Design*, 146–155.

Bhatia, A.; Kavraki, L. E.; and Vardi, M. Y. 2010. Sampling-based motion planning with temporal goals. In *2010 IEEE International Conference on Robotics and Automation*, 2689–2696. IEEE.

Bortolussi, L.; and Nenzi, L. 2014. Specifying and monitoring properties of stochastic spatio-temporal systems in signal temporal logic. In *Proceedings of the 8th International Conference on Performance Evaluation Methodologies and Tools*, 66–73.

De Moura, L.; and Bjørner, N. 2008. Z3: An efficient SMT solver. In *International conference on Tools and Algorithms for the Construction and Analysis of Systems*, 337–340. Springer.

Haghighi, I.; Jones, A.; Kong, Z.; Bartocci, E.; Gros, R.; and Belta, C. 2015. SpaTeL: a novel spatial-temporal logic and its applications to networked systems. In *Proceedings of the 18th International Conference on Hybrid Systems: Computation and Control*, 189–198.

Hsu, D.; Latombe, J.-C.; and Motwani, R. 1997. Path planning in expansive configuration spaces. In *Proceedings of international conference on robotics and automation*, volume 3, 2719–2726. IEEE.

Karaman, S.; and Frazzoli, E. 2011. Sampling-based algorithms for optimal motion planning. *The international journal of robotics research*, 30(7): 846–894.

Karlsson, J.; Barbosa, F. S.; and Tumova, J. 2020. Sampling-based motion planning with temporal logic missions and spatial preferences. *IFAC-PapersOnLine*, 53(2): 15537–15543.

Kavraki, L. E.; Svestka, P.; Latombe, J.; and Overmars, M. H. 1996. Probabilistic roadmaps for path planning in high-dimensional configuration spaces. *IEEE Trans. Robotics Autom.*, 12(4): 566–580.

Kindel, R.; Hsu, D.; Latombe, J.-C.; and Rock, S. 2000. Kinodynamic motion planning amidst moving obstacles. In *Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No. 00CH37065)*, volume 1, 537–543. IEEE.

LaValle, S. 1998. Rapidly-exploring random trees: A new tool for path planning. *Research Report 9811*.

Liu, W.; Alsalehi, S.; Mehdipour, N.; Bartocci, E.; and Belta, C. 2025. Quantifying the Satisfaction of Spatio-Temporal Logic Specifications for Multiagent Control. *IEEE Transactions on Automatic Control*, 70(8): 5098–5113.

Ma, M.; Bartocci, E.; Lifland, E.; Stankovic, J.; and Feng, L. 2020. SaSTL: Spatial aggregation signal temporal logic for runtime monitoring in smart cities. In *2020 ACM/IEEE 11th International Conference on Cyber-Physical Systems (ICCPS)*, 51–62. IEEE.

Nenzi, L.; Bartocci, E.; Bortolussi, L.; and Loreti, M. 2022. A logic for monitoring dynamic networks of spatially-distributed cyber-physical systems. *Logical Methods in Computer Science*, 18.

Nenzi, L.; Bortolussi, L.; Ciancia, V.; Loreti, M.; and Massink, M. 2015. Qualitative and quantitative monitoring of spatio-temporal properties. In *Runtime Verification: 6th International Conference, RV 2015, Vienna, Austria, September 22-25, 2015. Proceedings*, 21–37. Springer.

Perez, A.; Platt, R.; Konidaris, G.; Kaelbling, L.; and Lozano-Perez, T. 2012. LQR-RRT*: Optimal sampling-based motion planning with automatically derived extension heuristics. In *2012 IEEE International Conference on Robotics and Automation*, 2537–2542. IEEE.

Pnueli, A. 1977. The temporal logic of programs. In *18th annual symposium on foundations of computer science (sfcs 1977)*, 46–57. ieee.

Saha, I.; Ramaithitima, R.; Kumar, V.; Pappas, G. J.; and Seshia, S. A. 2014. Automated composition of motion primitives for multi-robot systems from safe LTL specifications. In *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 1525–1532. IEEE.

Shoukry, Y.; Nuzzo, P.; Balkan, A.; Saha, I.; Sangiovanni-Vincentelli, A. L.; Seshia, S. A.; Pappas, G. J.; and Tabuada, P. 2017. Linear temporal logic motion planning for teams of underactuated robots using satisfiability modulo convex programming. In *2017 IEEE 56th annual conference on decision and control (CDC)*, 1132–1137. IEEE.

Shoukry, Y.; Nuzzo, P.; Saha, I.; Sangiovanni-Vincentelli, A. L.; Seshia, S. A.; Pappas, G. J.; and Tabuada, P. 2016. Scalable lazy SMT-based motion planning. In *2016 IEEE 55th Conference on Decision and Control (CDC)*, 6683–6688. IEEE.

Soria, E.; Schiano, F.; and Floreano, D. 2020. SwarmLab: A MATLAB drone swarm simulator. In *2020 IEEE/RSJ international conference on intelligent robots and systems (IROS)*, 8005–8011. Ieee.

Sun, D.; Chen, J.; Mitra, S.; and Fan, C. 2022. Multi-agent motion planning from signal temporal logic specifications. *IEEE Robotics and Automation Letters*, 7(2): 3451–3458.

Vasile, C. I.; Li, X.; and Belta, C. 2020. Reactive sampling-based path planning with temporal logic specifications. *The International Journal of Robotics Research*, 39(8): 1002–1028.

Vasile, C.-I.; Raman, V.; and Karaman, S. 2017. Sampling-based synthesis of maximally-satisfying controllers for temporal logic specifications. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 3840–3847. IEEE.

Zhao, Y.; Yu, X.; Hoxha, B.; Fainekos, G.; Deshmukh, J.; and Lindemann, L. 2025. STL-GO: Spatio-Temporal Logic with Graph Operators for Distributed Systems with Multiple Network Topologies. *ACM Transactions on Embedded Computing Systems*.