Security-Aware Planning and Control of Multi-Agent Systems with LTL Tasks

Georgios Mitsos * **Dimos V. Dimarogonas** ** **Siyuan Liu** *

* *Department of Electrical Engineering, Eindhoven University of Technology, Eindhoven, the Netherlands (e-mails: {g.mitsos, s.liu5}@tue.nl)*
** *Division of Decision and Control Systems, KTH Royal Institute of Technology, Stockholm, Sweden (e-mail: dimos@kth.se)*

Abstract: This paper presents a secure-by-construction planning and control framework for multi-agent systems under linear temporal logic (LTL) specifications. The framework protects sensitive information from a passive intruder with partial observations of the agents' motion. Security in multi-agent coordination is captured by two notions that prevent the intruder from inferring whether a secret task has been executed and from identifying the agent responsible for its execution. The proposed framework incorporates the security constraints directly into the LTL synthesis procedure by constructing a secure finite transition system that removes all paths violating these constraints. Standard LTL synthesis is then applied to this secure abstraction to generate discrete plans, which are then refined into dynamically feasible continuous trajectories. This synthesis procedure provides formal guarantees that the resulting behavior of the multi-agent system satisfies both the global LTL specification and the security constraints. The effectiveness of the proposed framework is demonstrated through a two-drone case study.

Keywords: Secure-by-construction synthesis, multi-agent systems, linear temporal logic, opacity, trajectory planning

1. INTRODUCTION

Linear temporal logic (LTL) (Baier and Katoen, 2008) can specify complex tasks beyond point-to-point navigation (LaValle, 2006). By abstracting agent dynamics to finite-state models, model-checking techniques enable synthesis of fully automated, correct-by-design discrete plans (Smith et al., 2011), which are then refined into continuous trajectories via low-level controllers (Fainekos et al., 2009; Kloetzer and Belta, 2008).

In multi-agent systems, LTL enables high-level planning beyond classical cooperative tasks, e.g., collision avoidance, consensus, and formation. However, interdependent tasks and coordination requirements make LTL synthesis over the joint state space quickly intractable. To improve scalability, various decentralized methods have been proposed (Kloetzer and Belta, 2007; Loizou and Kyriakopoulos, 2004; Guo and Dimarogonas, 2014).

Despite these advances, information security during task execution has received comparatively less attention. Autonomous cyber-physical systems (CPS), such as drone swarms and intelligent transportation systems, often operate in settings where task correctness and confidentiality are equally critical. This motivates secure-by-construction strategies that incorporate security constraints directly into the control synthesis process (Liu et al., 2022; Yin et al., 2021). Recent works have incorporated information-flow security in robotic path planning via the notion of *opacity*; see, e.g., (Ma and Cai, 2021; Hadjicostis, 2018; Wang et al., 2020; Xie et al., 2021) for single-agent systems and (Yu et al., 2022) for multi-agent systems. However, these works do not address systems with continuous dynamics, where controllers must simultaneously ensure security and LTL satisfaction in the physical workspace.

In this paper, we propose a security-aware planning framework for multi-agent systems with continuous affine dynamics subject to LTL specifications. We consider the presence of an intruder modeled as a malicious observer with partial observations of the system. The main contributions of this work are summarized as follows:

(i) We introduce two complementary information-flow security notions: Type-A Security, which prevents the intruder from inferring whether any agent has visited a secret region, and Type-B Security, which prevents identification of the agent responsible for the secret visit. These notions can be enforced independently or jointly.

(ii) Based on these notions, we construct secure Weighted Transition Systems (WTSs) and apply standard LTL synthesis to generate secure discrete plans, which are then refined into dynamically feasible continuous trajectories that satisfy both the LTL tasks and the security constraints. Compared to existing LTL-based security-aware planning methods (Yang et al., 2020; Yu et al., 2022; Xie et al., 2021), where systems are modeled as finite WTSs, our framework is, to the best of our knowledge, the first directly applicable to multi-agent systems with continuous dynamics, input constraints, and complex LTL specifications. Moreover, unlike (Zhong et al., 2025), which focuses on opacity and safety, our framework addresses security-aware multi-agent planning under general LTL tasks.

[1] Corresponding author: Siyuan Liu (e-mail: *s.liu5*@tue.nl)

(iii) Under the proposed security notions, the resulting discrete planning algorithm achieves significantly lower complexity than that in (Yu et al., 2022) (cf. Sec. 4.3).

The paper is organized as follows. Sec. 2 introduces preliminaries and notation. Sec. 3 formalizes the security notions for multi-agent systems, and states the planning problem. Sec. 4 describes the planning and control framework, ensuring both LTL satisfaction and security in continuous workspaces. Sec. 5 demonstrates the proposed method through a two-drone case study, and Sec. 6 concludes the paper. Due to space limitations, the proofs are omitted and can be found in the arXiv version (Mitsos et al., 2026).

2. PRELIMINARIES

Notation. Let $\mathbb{R}$ denote the set of real numbers and $\mathbb{N}$ the set of positive integers. We denote $\mathbb{R}_{\geq 0}$ and $\mathbb{R}_{+}$ the sets of nonnegative and positive real numbers, respectively. For any $n, m \in \mathbb{N}$, $\mathbb{R}^n$ denotes the set of n-dimensional column vectors and $\mathbb{R}^{n\times m}$ the set of $n \times m$ real matrices. Vectors are denoted by bold lowercase letters and matrices by bold uppercase letters. A class-$\mathcal{K}$ function is a function $\alpha : \mathbb{R}_{\geq 0} \to \mathbb{R}_{\geq 0}$ that is strictly increasing and with $\alpha(0) = 0$. For any vector $\mathbf{x} \in \mathbb{R}^n$, $\|\mathbf{x}\|$ denotes the Euclidean norm, i.e., $\|\mathbf{x}\| = \sqrt{\mathbf{x}^\top \mathbf{x}}$.

2.1 System model

Consider a multi-agent system with M agents, indexed by $\mathcal{A} = \{1, \dots, M\}$, operating in a workspace modeled as a full-dimensional polytope $P \subset \mathbb{R}^n$.

A full-dimensional polytope is a compact convex set with non-empty interior in $\mathbb{R}^n$ and can be represented as the intersection of finitely many closed half-spaces, i.e., $P = \{\mathbf{x} \in \mathbb{R}^n \mid \mathbf{p}_i^\top \mathbf{x} + g_i \geq 0,\ i = 1, \dots, d\}$. It consists of lower-dimensional *faces*: a j-face has dimension $j < n$; facets are $(n-1)$-faces, edges are 1-faces, and vertices are 0-faces.

Each Agent $i \in \mathcal{A}$ follows the linear affine dynamics

$$\dot{\mathbf{x}}_i = \mathbf{A}_i \mathbf{x}_i + \mathbf{B}_i \mathbf{u}_i + \mathbf{b}_i, \tag{1}$$

where $\mathbf{x}_i \in P$ is Agent i's state vector, $\mathbf{u}_i \in U_i \subset \mathbb{R}^m$ is the input vector, $\mathbf{A}_i \in \mathbb{R}^{n\times n}$ and $\mathbf{B}_i \in \mathbb{R}^{n\times m}$ are constant matrices, and $\mathbf{b}_i \in \mathbb{R}^n$ is a constant vector. The set U_i is a convex polytope indicating bounded control inputs.

The continuous trajectory of Agent i is denoted by $\mathbf{x}_i(t) : \mathbb{R}_{\geq 0} \to P$. The global trajectory of the system collects all agents' trajectories, i.e., $\mathbf{x}(t) = \big[\mathbf{x}_1(t)^\top \ \dots \ \mathbf{x}_M(t)^\top\big]^\top$.

2.2 Weighted transition systems

The goal of this paper is to synthesize a continuous trajectory for a multi-agent system as in (1) subject to security constraints (cf. Sec. 3) and LTL specifications (cf. Sec. 2.4), defined over a partitioned abstraction of the agents' motion. The polytopic workspace is partitioned into non-overlapping sub-polytopes $q_1, \dots, q_p$ such that $P = \bigcup_{j=1}^{p} q_j$ and $q_j \cap q_k = \emptyset$, for $j \neq k$. Each Agent i's motion among regions is abstracted as a transition in a finite WTS $\mathcal{T}^i$, which may differ across agents.

Definition 1. A Weighted Transition System $\mathcal{T}^i$ is a tuple $\big(Q^i, Q_0^i, \longrightarrow_i, w_i, \mathcal{AP}_i, L_i\big)$, where $Q^i = \{q_1, \dots, q_p\}$ is the set of states, and $Q_0^i \subseteq Q^i$ the set of initial states. Each q_j, $j = 1, \dots, p$, corresponds to Agent i being in region q_j. The transition relation $\longrightarrow_i \subseteq Q^i \times Q^i$ collects all the *feasible transitions* for Agent i, and the cost function $w_i : Q^i \times Q^i \to \mathbb{R}_+$ assigns a weight $w_i(q, q')$ to each $(q, q') \in \longrightarrow_i$. The set of atomic propositions $\mathcal{AP}_i = \{\pi_j\}_{j\in\mathbb{N}}$ represents properties of interest, and the labeling function $L_i : Q^i \to 2^{\mathcal{AP}_i}$ specifies the properties that hold at each state.

The *dynamical feasibility* of the transitions in $\longrightarrow_i$ w.r.t. the continuous system (1) is characterized in Sec. 4.2. The abstracted motion of all M agents, each executing one transition per time step, is represented by a Global WTS (gWTS), defined as the synchronous product of the individual agents' WTSs.

Definition 2. A Global WTS $\mathcal{T}_g$ is a tuple $\big(Q_g, Q_{g,0}, \longrightarrow_g, w_g, \mathcal{AP}_g, L_g\big)$, where $Q_g = Q^1 \times \dots \times Q^M$ and $Q_{g,0} = Q_0^1 \times \dots \times Q_0^M$ are the sets of global and initial states, respectively. A transition (q, q'), $q, q' \in Q_g$, is included in the transition relation $\longrightarrow_g$ if $(q^i, q^{i\prime}) \in \longrightarrow_i$, for all $i \in \mathcal{A}$. The corresponding weight is defined as $w_g(q, q') = \sum_{i=1}^{M} w_i(q^i, q^{i\prime})$. The label of any $q \in Q_g$ is $L_g(q) = \big(L_1(q^1), \dots, L_M(q^M)\big) \subseteq \mathcal{AP}_g = 2^{\mathcal{AP}_1} \times \dots \times 2^{\mathcal{AP}_M}$.

Each global state $q = \big(q^1, \dots, q^M\big) \in Q_g$ collects the local states of all agents. A local path of Agent i is a sequence $\tau_i = \tau_i(1)\tau_i(2)\dots$ in $\mathcal{T}^i$, where $\tau_i(1) \in Q_0^i$ and $(\tau_i(j), \tau_i(j+1)) \in \longrightarrow_i$, for all $j \geq 1$. Its trace is $trace(\tau_i) = L_i(\tau_i(1))\, L_i(\tau_i(2)) \dots$. A global path $\tau = \tau(1)\tau(2)\dots$ in $\mathcal{T}_g$ satisfies $\tau(1) \in Q_{g,0}$ and $(\tau(j), \tau(j+1)) \in \longrightarrow_g$ for all $j \geq 1$. Its trace is $trace(\tau) = L_g(\tau(1))\, L_g(\tau(2)) \dots$. The set of all global paths in $\mathcal{T}_g$ is denoted $Path(\mathcal{T}_g)$. We also write a path $\tau \in Path(\mathcal{T}_g)$ as $\tau(1) \to_g \tau(2) \to_g \dots$.

A special class of infinite paths is the prefix-suffix form

$$\tau = \tau(1) \dots \tau(k)[\tau(k+1) \dots \tau(N)]^\omega, \tag{2}$$

where the prefix $\tau(1) \dots \tau(k)$ is executed once, and the suffix $\tau(k+1) \dots \tau(N)$ repeats indefinitely. It must hold that $(\tau(N), \tau(k+1)) \in \longrightarrow_g$.

2.3 Control barrier functions

Consider Agent i as in (1) and a set $\mathcal{C} \subseteq P \subset \mathbb{R}^n$ defined as the superlevel set of a continuously differentiable function $h : \mathbb{R}^n \to \mathbb{R}$, i.e., $\mathcal{C} = \{\mathbf{x} \in P \mid h(\mathbf{x}) \geq 0\}$.

Definition 3. Consider the set $\mathcal{C}$. The function $h(\mathbf{x})$ is a control barrier function (CBF) if there exists a locally Lipschitz class-$\mathcal{K}$ function α s.t. for each $\mathbf{x}_i \in P$:

$$\sup_{\mathbf{u}_i \in U_i} \left[\Big(\frac{\partial h}{\partial \mathbf{x}_i}(\mathbf{x}_i) \Big)^\top (\mathbf{A}_i \mathbf{x}_i + \mathbf{B}_i \mathbf{u}_i + \mathbf{b}_i) \right] \geq -\alpha\big(h(\mathbf{x}_i)\big). \tag{3}$$

CBFs (Ames et al., 2019) will be used in Sec. 4.2 to guarantee *forward invariance* of workspace partitions.

2.4 Linear temporal logic

LTL formulas are defined over a set of atomic propositions $\mathcal{AP}$ using Boolean and temporal operators. The syntax of LTL is given recursively (Baier and Katoen, 2008):

$$\phi ::= p \mid \neg\phi \mid \phi_1 \wedge \phi_2 \mid \mathcal{X}\phi \mid \phi_1 \,\mathcal{U}\, \phi_2,$$

where $p \in \mathcal{AP}$ and ϕ, ϕ_1, ϕ_2 are LTL formulas; $\mathcal{X}$ and $\mathcal{U}$ denote the *next* and *until* operators, respectively. The Boolean disjunction ($\vee$) and *true*, and the temporal operators *eventually* ($\Diamond$) and *always* ($\Box$) are defined as usual.

LTL formulas are interpreted over infinite traces. A trace $r \in \left(2^{\mathcal{AP}}\right)^{\omega}$ satisfies an LTL formula ϕ, written as $r \models \phi$, if ϕ holds over r according to the LTL semantics (Baier and Katoen, 2008), which are omitted here for brevity.

Definition 4. A Nondeterministic Büchi Automaton (NBA) is a tuple $\mathcal{B}_\phi = (S, S_0, F, \Sigma, \delta)$, where S is a finite set of states; $S_0 \subseteq S$ is the set of initial states; $F \subseteq S$ is the set of accepting states; $\Sigma = 2^{\mathcal{AP}}$ is the input alphabet; and $\delta : S \times \Sigma \to 2^S$ is the transition relation.

For any LTL formula ϕ over $\mathcal{AP}$, there exists a NBA $\mathcal{B}_\phi$ with input alphabet $\Sigma = 2^{\mathcal{AP}}$ such that $\mathcal{B}_\phi$ accepts exactly the infinite traces that satisfy ϕ (Wolper et al., 1983).

3. SECURITY CONSTRAINTS IN MULTI-AGENT SYSTEMS AND PROBLEM FORMULATION

In this section, we introduce two new security notions for multi-agent systems, capturing whether a malicious intruder modeled as an outside observer can infer certain secret information. The problem formulation follows.

3.1 Security constraints for multi-agent systems

Consider a multi-agent system whose motion is described by a gWTS as in Def. 2. We assume the passive intruder has knowledge of the individual agent models, i.e., $\mathcal{T}^i = \left(Q^i, Q^i_0, \longrightarrow_i, w_i, \mathcal{AP}_i, L_i\right)$, for all $i \in \mathcal{A}$, and it can observe each Agent i through an output function $H_i : Q^i \to Y_i$, where Y_i is the output set of possible observation symbols.

To formulate the security requirement, we aim to prevent the intruder from inferring secret information based on available observations; specifically, whether sensitive behaviors occur and, when applicable, which agents execute them. These sensitive behaviors are modeled as visits to designated secret states $Q^i_S \subseteq Q^i$ for each Agent i.

To incorporate these security-related features, the standard WTS in Def. 1 is extended as

$$\mathcal{T}^i = \left(Q^i, Q^i_0, \longrightarrow_i, w_i, \mathcal{AP}_i, L_i, Y_i, H_i, Q^i_S\right).$$

The corresponding gWTS is rewritten as

$$\mathcal{T}_g = \left(Q_g, Q_{g,0}, \longrightarrow_g, w_g, \mathcal{AP}_g, L_g, Y_g, H_g\right), \qquad (4)$$

where $Y_g = Y_1 \times \cdots \times Y_M$ and $H_g : Q_g \to Y_g$ with $H_g(q) = \left(H_1\left(q^1\right), \ldots, H_M\left(q^M\right)\right)$. An output sequence is denoted as $y_g = y_g(1)\, y_g(2) \ldots$, or equivalently, $y_g = y_g(1) \to_g y_g(2) \to_g \cdots$.

The two types of security notions are formalized as follows.

Definition 5. Consider the gWTS in (4). A global path $\tau \in Path(\mathcal{T}_g)$ is *Type-A secure* if there exists a global path $\tau' \in Path(\mathcal{T}_g)$ such that: (i) $H_g(\tau) = H_g(\tau')$, and (ii) for every $i \in \mathcal{A}$ and $j \geq 1$: $\tau_i(j) \in Q^i_S \Rightarrow \tau'_i(j) \notin Q^i_S$.

Intuitively, this property ensures plausible deniability: whenever an agent visits a secret state, there exists an alternative behavior of the system that is indistinguishable to the intruder, and in which the agent is not in a secret state at that time. In the sequel, we refer to τ as the "real path" and to τ' as its "copy path".

Remark 1. For Type-A security to be meaningful, each Agent $i \in \mathcal{A}$ must have at least one non-secret state, i.e. $Q_i \backslash Q^i_S \neq \emptyset$, and a non-secret initial state, i.e., $Q^i_0 \backslash Q^i_S \neq \emptyset$. Otherwise, the property is trivially violated.

Type-A security treats each agent independently; that is, there is no collaboration between agents to guarantee security. Satisfaction of Type-A security implies satisfaction of initial-state, current-state, and infinite-step opacity (Saboori and Hadjicostis, 2007) for each agent. Note that Type-A security does not prevent the intruder from identifying an agent as the only candidate for a secret task, e.g., if the intruder knows the secret task location and only one agent produces the corresponding observation along its path. Type-B security addresses such cases.

Definition 6. Consider the gWTS in (4). A global path $\tau \in Path(\mathcal{T}_g)$ is *Type-B secure* if, for every Agent $i \in \mathcal{A}$ and every time step $j \geq 1$:

$$\tau_i(j) \in Q^i_S \Rightarrow \exists k \neq i \text{ s.t. } H_k(\tau_k(j)) = H_i(\tau_i(j)).$$

Intuitively, Type-B security ensures that whenever an agent visits a secret state, at least one other agent produces the same observation at that time, preventing the intruder from identifying which agent is responsible for the secret visit. A path that satisfies both Type-A and Type-B security is called *Type-A/B secure*.

Similar security notions for multi-agent systems have been introduced in (Yu et al., 2022), which also capture whether an intruder can infer a secret state has been visited (Type-I) and by which agent (Type-II). However, unlike Type I, which requires the alternative (copy) path to never visit any secret state, our Type-A security notion imposes this restriction only at the time steps when an agent is in a secret state. Similarly, unlike Type II, which requires that some other agent visit a secret state, our Type-B leverages the intruder's limited knowledge of the system's evolution and only requires that another agent produces the same observation at those steps. These relaxations reduce the size of the problem while preserving similar security guarantees, making our approach considerably more scalable than the one in (Yu et al., 2022); a detailed complexity analysis is provided in Sec. 4.4.

3.2 Problem formulation

The desired LTL formulas and security properties were defined over discrete paths and their traces, however the goal is to design a control scheme for agents evolving in continuous time and space. Therefore, each discrete transition must correspond to a realizable continuous motion between representative points of the workspace.

Assumption 1. For each discrete transition, Agent i moves between representative points of the corresponding regions in $\mathcal{T}^i$. Without loss of generality, we select these points as the region centroids. Thus, a global transition (q, q') starts with all Agents i at the centroids of regions q^i, and ends at the centroids of regions $q^{i\prime}$.

This assumption may exclude some feasible trajectories but allows us to guarantee the existence of a continuous controller for every discrete transition. The next definition formalizes how a global path is induced as the sequence of visited sub-polytopes.

Definition 7. Let $\mathbf{x}_i(t)$ be the trajectory of Agent i, and define $f_i(t) = q_j$ when $\mathbf{x}_i(t) \in q_j$. The switching times of Agent i are defined recursively as $t_{i,1} = 0$, and $t_{i,k+1} = \inf\left\{t > t_{i,k} \,|\, f_i(t) \neq f_i(t_{i,k})\right\}$. This yields $\tau_i = \tau_i(1)\, \tau_i(2) \ldots$, with $\tau_i(j) = f_i(t_{i,j})$. Let $(s_\ell)_{\ell \in \mathbb{N}}$ be the

increasing ordering of all switching times across all agents. At each s_ℓ, the global state $\tau(\ell) = (\tau_1(\ell), \ldots, \tau_M(\ell))$ is defined, and all agents $i \in \mathcal{A}$ transition from $\tau_i(\ell-1)$ to $\tau_i(\ell)$ simultaneously. If the system eventually remains in a global state $\tau(k)$, then $\tau(k)$ repeats indefinitely, generating an infinite global path.

Remark 2. By Def. 7, each global trajectory $\mathbf{x}$ induces a unique global path τ, representing the sequence of visited sub-polytopes. Since labels, observations, and secret states are constant within each sub-polytope, $\mathbf{x}(t)$ and τ produce identical traces. Hence, we write $trace(\mathbf{x}) \models \phi$ to indicate that a continuous trajectory satisfies an LTL formula ϕ. We then have $trace(\mathbf{x}) \models \phi$ if and only if $trace(\tau) \models \phi$, and, Type-A/B security of τ implies the same for $\mathbf{x}$.

This paper first aims to compute a discrete plan satisfying the global LTL task and security constraints, and then synthesize a low-level controller that implements the motion of the system corresponding to each discrete transition.

Problem 1. Given a multi-agent system as in (1) operating in a partitioned workspace P and a global task specified by an LTL formula ϕ, synthesize a controller such that the generated global trajectory $\mathbf{x} : \mathbb{R}_{\geq 0} \to P$ satisfies (i) $trace(\mathbf{x}) \models \phi$, and (ii) $\mathbf{x}$ is Type-A/B secure.

4. SECURE-BY-CONSTRUCTION PLANNING AND CONTROL

This section presents the planning and control framework for solving Problem 1. First, secure discrete abstractions are constructed, retaining only global paths satisfying the security constraints. Next, transition feasibility w.r.t. system dynamics and input constraints is verified. LTL synthesis is then used to compute a prefix-suffix path that satisfies the global task. Finally, a continuous trajectory is generated from the discrete path, guaranteeing satisfaction of both the global LTL task and the security constraints.

4.1 Secure global transition systems

To enforce Type-A security, we construct the Twin-Global WTS (Twin-gWTS), which captures pairs of indistinguishable global states, and the Secure Twin-gWTS, which removes pairs where any agent's real and copy local states are both secret. The concept of a Twin-Weighted Transition System was introduced in (Yang et al., 2020) and is extended here to multi-agent systems.

Definition 8. Let $\mathcal{T}_g$ be a gWTS in (4). The corresponding Twin-gWTS is $\mathcal{V} = \big(X, X_0, \longrightarrow_V, w_V, \mathcal{AP}_g, L_V\big)$, where:

- $X \subseteq Q_g \times Q_g$ is the set of states, with $(q_1, q_2) \in X$ if $q_1, q_2 \in Q_g$ and $H_g(q_1) = H_g(q_2)$
- $X_0 = X \cap (Q_{g,0} \times Q_{g,0})$ is the set of initial states
- $\longrightarrow_V \subseteq X \times X$ is the transition relation, where, for any $(q_1, q_2), (q_1', q_2') \in X$, $((q_1, q_2), (q_1', q_2')) \in \longrightarrow_V$ if: $(q_1, q_1') \in \longrightarrow_g$, $(q_2, q_2') \in \longrightarrow_g$, and $H_g(q_1') = H_g(q_2')$
- $w_V : X \times X \to \mathbb{R}_+$ is the cost function, where, for any $q = ((q_1, q_2), (q_1', q_2')) \in \longrightarrow_V$, $w_V(q) = w_g(q_1, q_1')$
- $L_V : X \to \mathcal{AP}_g$ is the labeling function, where, for any $(q_1, q_2) \in X$, $L_V((q_1, q_2)) = L_g(q_1)$

Each twin state $q = (q_1, q_2) = \big((q_1^1, \ldots, q_1^M), (q_2^1, \ldots, q_2^M)\big)$ in the Twin-gWTS represents a pair of indistinguishable global states, i.e., $H_g(q_1) = H_g(q_2)$, where q_1 is the "real state" and q_2 the "copy state".

Definition 9. Let $\mathcal{V}$ be a Twin-gWTS in Def. 8. The corresponding Secure Twin-gWTS is $\mathcal{V}_s = \big(X_s, X_{s,0}, \longrightarrow_s, w_s, \mathcal{AP}_s, L_s\big)$, and is obtained by removing all twin states (q_1, q_2), in which there is an Agent $i \in \mathcal{A}$ such that $q_1^i, q_2^i \in Q_S^i$. All other components are inherited from $\mathcal{V}$.

For a twin state $x_s = (q_1, q_2) \in X_s$, denote $\Pi(x_s) = q_1$. For a twin path $\sigma = \sigma(1)\sigma(2)\cdots \in Path(\mathcal{V}_s)$, the projected path in $Path(\mathcal{T}_g)$ is $\Pi(\sigma) = \Pi(\sigma(1))\Pi(\sigma(2))\ldots$.

For Type-B security, we construct the Type-B gWTS.

Definition 10. Let $\mathcal{T}_g$ be a gWTS in (4). The corresponding Type-B gWTS $\mathcal{T}_g^B$ is $\mathcal{T}_g^B = \big(Q_g^B, Q_{g,0}^B, \longrightarrow_g^B, w_g^B, \mathcal{AP}_g^B, L_g^B\big)$, where the global states are restricted to

$$Q_g^B = \Big\{(q^1, \ldots, q^M) \in Q_g \,|\, q^i \in Q_S^i \Rightarrow \exists k \neq i : H_i(q^i) = H_k(q^k)\Big\}.$$

All other components are inherited from $\mathcal{T}_g$ in (4).

Type-A/B security. Type-A and Type-B security can be enforced simultaneously by first constructing the Type-B gWTS $\mathcal{T}_g^B$, and then building the Twin-gWTS $\mathcal{V}$ and Secure Twin-gWTS $\mathcal{V}_s$ on top of it. The following propositions are provided without proofs, as they hold trivially.

Proposition 1. By construction, every path $\sigma \in Path(\mathcal{V}_s)$ satisfies both Type-A and Type-B security.

Proposition 2. For every path $\tau \in Path(\mathcal{T}_g)$ that satisfies both Type-A and Type-B security, there exists at least one path $\sigma \in Path(\mathcal{V}_s)$ such that $\Pi(\sigma) = \tau$.

Proposition 2 shows that $\mathcal{V}_s$ includes all paths that satisfy both security types, yielding completeness guarantees.

4.2 Transition feasibility

A discrete transition $(q, q') \in \longrightarrow_s$, with $q \neq q'$, is said to be *dynamically feasible w.r.t. system* (1) if there exist controllers $\mathbf{u}_i$ for all real and copy Agents $i \in \mathcal{A}$ that drive the agents between the centroids ξ_i of the current region q^i at $t = 0$ and ξ_i' of $q^{i\prime}$ at $t = t_f$, satisfying: (i) the agent's dynamics in (1); (ii) input constraints; (iii) initial and final centroid positions; (iv) if $q^i \neq q^{i\prime}$, Agent i crosses the exit facet $F_{c,i} = q^i \cap q^{i\prime}$ at $t = t_c$ as in Def. 7; and (v) Agent i remains within q^i for $t \in T_1 = [0, t_c]$, and within $q^{i\prime}$ for $t \in T_2 = [t_c, t_f]$. The real and copy Agent i, for each $i \in \mathcal{A}$, share the same dynamics and input bounds.

Lemma 3. Let $q^i \subset \mathbb{R}^n$ be a convex polytope with facets $F_j \in q^i$, represented by $h_j(\mathbf{x}) = \mathbf{p}_j^\top \mathbf{x} + g_j = 0$. Suppose there exists a locally Lipschitz continuous controller $\mathbf{u}_i : T_1 \to U_i$ s.t. for all $F_j \in q^i$ and $t \in T_1$,

$$\mathbf{p}_j^\top (\mathbf{A}_i \mathbf{x}_i(t) + \mathbf{B}_i \mathbf{u}_i(t) + \mathbf{b}_i) \geq -\alpha(h_j(\mathbf{x}_i(t))), \quad (5)$$

where $\alpha(r) = \gamma r$ with $\gamma > 0$. Then, q^i is *forward invariant* for $t \in T_1$. If the same holds for all $F_j \in q^{i\prime}$ and $t \in T_2$, then $q^{i\prime}$ is forward invariant for $t \in T_2$.

The functions h_j are CBFs as in Def. 3. Any locally Lipschitz controller $\mathbf{u}_i(t) : [0, t_c] \to U_i$ that satisfies (5) on all facets F_j of q^i renders q^i *forward invariant* (Ames et al., 2019). The same reasoning applies to $q^{i\prime}$.

To determine if a transition $(q, q') \in \longrightarrow_s$ is feasible, we compute a trajectory $\mathbf{x}_i$ and a controller $\mathbf{u}_i$ that realizes (q, q') while satisfying conditions (i)–(v), by solving a centralized quadratic program (QP) for each (q, q').

$$\min_{\mathbf{x}_i, \mathbf{u}_i,\, i\in\mathcal{A}} \quad \int_0^{t_f} \sum_{i\in\mathcal{A}} ||\mathbf{u}_i(t)||^2 dt$$

$$\begin{aligned}
\text{s.t.} \quad & \dot{\mathbf{x}}_i(t) = \mathbf{A}_i\mathbf{x}_i(t) + \mathbf{B}_i\mathbf{u}_i(t) + \mathbf{b}_i,\ \forall i \in \mathcal{A} \\
& \mathbf{u}_i(t) \in \mathcal{U}_i,\ \forall i \in \mathcal{A} \\
& \mathbf{x}_i(0) = \xi_i, \quad \mathbf{x}_i(t_f) = \xi_i',\ \forall i \in \mathcal{A} \\
& h_{c,i}(\mathbf{x}_i(t_c)) = \mathbf{p}_{c,i}^\top \mathbf{x}_i(t_c) + g_{c,i} = 0,\ \forall i \in \mathcal{A} \\
& \mathbf{p}_j^\top(\mathbf{A}_i\,\mathbf{x}_i(t) + \mathbf{B}_i\,\mathbf{u}_i(t) + \mathbf{b}_i) \geq -\gamma\, h_j(\mathbf{x}_i(t)), \\
& \qquad \forall i \in \mathcal{A},\ F_j \in q^i,\ t \in [0, t_c] \\
& \mathbf{p}_j^\top(\mathbf{A}_i\,\mathbf{x}_i(t) + \mathbf{B}_i\,\mathbf{u}_i(t) + \mathbf{b}_i) \geq -\gamma\, h_j(\mathbf{x}_i(t)), \\
& \qquad \forall i \in \mathcal{A},\ F_j \in q^{i\,\prime},\ t \in [t_c, t_f]
\end{aligned}$$

The QP is solved numerically in a sampled-data fashion over a fixed horizon t_f. Despite employing centralized planning, the QP is convex and can be solved efficiently. If it admits a solution, then $(q, q') \in \longrightarrow_s$ is feasible: conditions (i)-(iii) are enforced by the first three constraints, while conditions (iv)-(v) follow from Lemma 3 and the last three constraints. Otherwise, (q, q') is considered infeasible and is discarded from the gWTS.

Remark 3. Since sub-polytopes are defined by non-strict inequalities, trajectories remaining on facets over finite time intervals may create ambiguities in labels, observations, and secret status. To avoid this, a small slack variable $\epsilon > 0$ is introduced in all CBF constraints:

$$\mathbf{p}_j^\top(\mathbf{A}_i\,\mathbf{x}_i(t) + \mathbf{B}_i\,\mathbf{u}_i(t) + \mathbf{b}_i) \geq -\gamma\,(h_j(\mathbf{x}_i(t)) - \epsilon).$$

ensuring trajectories remain strictly inside each sub-polytope except at facet-crossing instants $t = t_c$.

4.3 LTL task satisfaction

To incorporate a given LTL specification ϕ, we synchronize the resulting secure Twin gWTS $\mathcal{V}_s$ with the NBA $\mathcal{B}_\phi$ that accepts ϕ to form a Product Büchi Automaton (PBA).

Definition 11. A Product Büchi Automaton is $\mathcal{A}_P = \mathcal{V}_s \otimes \mathcal{B}_\phi = (S_P, S_{P,0}, F_P, \delta_P, w_P)$, where $S_P = X_s \times S$, $S_{P,0} = X_{s,0} \times S_0$ and $F_P = X_s \times F$ are the sets of states, initial states and accepting states, respectively. The transition relation is $\delta_P \subseteq S_P \times S_P$, where $((q,s),(q',s')) \in \delta_P$ iff $(q,q') \in \longrightarrow_s$ and $\exists \sigma \in L_s(q')$ such that $s' \in \delta(s,\sigma)$. The cost function is $w_P : \delta_P \to \mathbb{R}_+$, where, for any $((q,s),(q',s')) \in \delta_p$, $w_P((q,s),(q',s')) = w_s(q,q')$.

A path τ in $\mathcal{V}_s$ satisfies ϕ if and only if it corresponds to an accepting path in the PBA, i.e., a path that starts from an initial state in $S_{P,0}$ and visits accepting states in F_P infinitely often. The cost of τ is defined as $J(\tau) = \beta\, J_{\text{prefix}}(\tau) + (1-\beta)\, J_{\text{suffix}}(\tau)$, $\beta \in [0,1]$, with $J_{\text{prefix}}(\tau) = \sum_{i=1}^{k} w(\tau(i), \tau(i+1))$, and $J_{\text{suffix}}(\tau) = \sum_{i=k+1}^{N-1} w(\tau(i), \tau(i+1)) \ + \ w(\tau(N), \tau(k+1))$.

4.4 Security-aware multi-agent planning and control

The workspace is first partitioned and the motion of the system is abstracted as a gWTS $\mathcal{T}_g$ as described in Sec. 2.2. To enforce the security properties, a secure gWTS is then constructed as in Sec. 4.1: the Type-B gWTS $\mathcal{T}_g^B$ for Type-B security, and the Secure Twin-gWTS $\mathcal{V}_s$ for Type-A. The feasibility of each transition in the secure gWTS $\mathcal{W}$ (i.e., $\mathcal{T}_g^B$ or $\mathcal{V}_s$) is verified using the QP in Sec. 4.2.

Alg. 1 computes an optimal prefix-suffix plan satisfying the LTL task. The corresponding dynamically feasible trajectories and control inputs are then reconstructed by concatenating the motion segments $\mathbf{x}^{q,q'}$ associated with each discrete transition (q,q') in $\tau^\star$ and obtained from the QP in Sec. 4.2. The resulting trajectories satisfy both the LTL task and the required security properties.

Algorithm 1 Secure Trajectory Planning and Control

Input: LTL formula ϕ, secure gWTS $\mathcal{W}$, weight β
Output: secure trajectory $\mathbf{x}^\star$, control $\mathbf{u}^\star$, cost $J^\star$

```
Convert LTL formula φ to NBA B_φ
Build PBA A_P from W and B_φ
Initialize τ* ← ∅ and J* ← ∞
for each accepting state q_acc ∈ F_P do
    Compute shortest prefix path π_prefix from any initial state q_0 ∈ S_{P,0} to q_acc using Dijkstra's algorithm
    Compute shortest suffix path π_suffix starting and ending at q_acc using Dijkstra's algorithm
    Compute total cost J_total = β·J_prefix + (1−β)·J_suffix
    if J_total < J* then
        J* ← J_total,   τ* ← π_prefix ⊕ π_suffix
Reconstruct trajectory x*(t) and control u*(t) from τ*
return x*(t), u*(t), J*
```

Theorem 4. Consider a multi-agent system as in (1) subject to an LTL formula ϕ and Type-A/B security constraints. Alg. 1 computes a continuous trajectory $\mathbf{x}^\star(t)$ that: (i) satisfies ϕ, (ii) guarantees Type-A, and/or Type-B security, and (iii) is dynamically feasible.

Complexity analysis. Consider a system of M agents, where each agent's motion is represented by a WTS with at most $|Q|$ states, and let the NBA for the LTL formula have $|S|$ states. The gWTS $\mathcal{T}_g$ contains at most $|Q|^M$ states. For Type-A/B security, constructing the Type-B gWTS together with the Twin-gWTS and Secure Twin-gWTS yields at most $|Q|^{2M}|S|$ states for the PBA. Thus, Alg. 1 performs a shortest-path search over a graph with at most $|Q|^{2M}|S|$ states. Compared to (Yu et al., 2022), which results in a graph of $(2|Q|)^{2M^2+M}|S|$ states, our framework achieves significantly lower computational complexity.

5. CASE STUDY

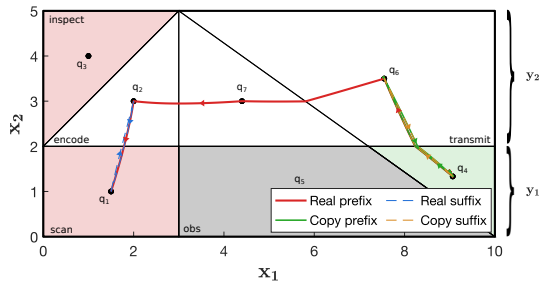
We demonstrate our framework in a two-drone scenario. Drone 1 scans air quality and encodes the collected data, while Drone 2 relays the data and visits a station for inspection. The workspace is partitioned into seven convex regions labeled with the atomic propositions $\mathcal{AP} = \{\texttt{scan}, \texttt{encode}, \texttt{transmit}, \texttt{inspect}, \texttt{obs}\}$, as shown in Fig. 1. Green, red, and gray areas denote initial, secret, and obstacle regions, respectively. An intruder observes the workspace through two observation regions. The goal is to synthesize a trajectory satisfying Type-A/B security and the LTL specification:

$$\begin{aligned}
\phi = \Box\Diamond\Big(\texttt{scan}^1 \wedge \Diamond\left(\texttt{encode}^1 \wedge \Diamond\texttt{transmit}^2\right)\Big) \wedge \\
\Diamond\texttt{inspect}^2 \wedge \Box\,\neg\,\texttt{obs},
\end{aligned}$$

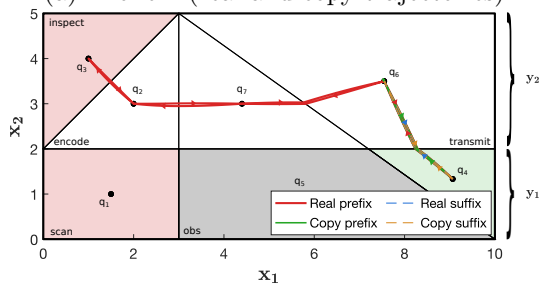
which requires Drone 1 to repeatedly scan and encode data, followed by Drone 2's transmission. Also, Drone 2 must perform an inspection, while both always avoid the obstacle. The superscript denotes the responsible agent.

Each Drone $i \in \{1,2\}$ follows linear affine dynamics

$$\dot{\mathbf{x}}_i = \begin{bmatrix} 0.1 & -0.15 \\ -0.1 & 0.2 \end{bmatrix} \mathbf{x}_i + \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \mathbf{u}_i + \begin{bmatrix} 0.2 \\ 0.3 \end{bmatrix},$$



(a) Drone 1 (real and copy trajectories)



(b) Drone 2 (real and copy trajectories)

Fig. 1. Workspace and synthesized trajectories.

where $\mathbf{x}_i = [x_{i,1}\ x_{i,2}]^\top \in P$, $\mathbf{u}_i = [u_{i,1}\ u_{i,2}]^\top \in [-3,3]^2$.

The implementation was carried out in Python 3.11, using `CDD` (Fukuda, 2025) for polytope operations and `LTL2BA` (Gastin and Oddoux, 2020) for LTL to NBA conversion. First, the discrete abstraction $\mathcal{T}_g$ (Sec. 2.2) and the corresponding secure abstractions were constructed (Sec. 4.1). Transition feasibility was verified by solving the QP, subject to the system dynamics, control bounds, facet-crossing conditions, and CBF constraints (Sec. 4.2), using $\gamma = 1$, $\epsilon = 0.01$. Next, Alg. 1 was applied to construct the PBA by synchronizing the NBA corresponding to ϕ with the secure abstraction (Sec. 4.3). The path minimizing $J = 0.5\, J_{\text{prefix}} + 0.5\, J_{\text{suffix}}$ was then computed, and the corresponding continuous trajectories were obtained (Sec. 4.4). The trajectories are shown in Fig. 1. The complete algorithm runtime is 70 sec on a MacBook Air M2. The results demonstrate that the proposed framework generates dynamically feasible trajectories satisfying both the LTL specification and the security properties.

6. CONCLUSION

This paper presented a centralized framework for security-aware multi-agent planning and control under LTL specifications. The proposed approach synthesizes dynamically feasible trajectories satisfying both the LTL task and security properties. Future work includes extensions to nonlinear dynamics and scalable hierarchical architectures.

REFERENCES

Ames, A., Coogan, S., Egerstedt, M., Notomista, G., Sreenath, K., and Tabuada, P. (2019). Control barrier functions: theory and applications. In *Proc. 18th IEEE Eur. Conf. Control*, 3420–3431.

Baier, C. and Katoen, J. (2008). *Principles of Model Checking*. MIT Press.

Fainekos, G.E., Girard, A., Kress-Gazit, H., and Pappas, G.J. (2009). Temporal logic motion planning for dynamic robots. *Automatica*, 45(2), 343–352.

Fukuda, K. (2025). cddlib: An efficient implementation of the double description method.

Gastin, P. and Oddoux, D. (2020). LTL2BA (version 1.3).

Guo, M. and Dimarogonas, D.V. (2014). Multi-agent plan reconfiguration under local LTL specifications. *Int. J. Robot. Res.*, 34(2), 218–235.

Hadjicostis, C.N. (2018). Trajectory planning under current-state opacity constraints. *IFAC-PapersOnLine*, 51, 337–342.

Kloetzer, M. and Belta, C. (2007). Temporal logic planning and control of robotic swarms by hierarchical abstractions. *IEEE Trans. Robot.*, 23(2), 320–330.

Kloetzer, M. and Belta, C. (2008). A fully automated framework for control of linear systems from temporal logic specifications. *IEEE Trans. Autom. Control*, 53(1), 287–297.

LaValle, S.M. (2006). *Planning Algorithms*. Cambridge Univ. Press.

Liu, S., Trivedi, A., Yin, X., and Zamani, M. (2022). Secure-by-construction synthesis of cyber-physical systems. *Annu. Rev. Control*, 53, 30–50.

Loizou, S.G. and Kyriakopoulos, K.J. (2004). Automatic synthesis of multi-agent motion tasks based on LTL specifications. In *Proc. 43rd IEEE Conf. Decis. Control*, volume 1, 153–158.

Ma, Z. and Cai, K. (2021). Optimal secret protections in discrete-event systems. *IEEE Trans. Autom. Control*, 67(6), 2816–2828.

Mitsos, G., Dimarogonas, D.V., and Liu, S. (2026). Security-aware planning and control of multi-agent systems with LTL tasks. *arXiv preprint arXiv:2605.13134*.

Saboori, A. and Hadjicostis, C.N. (2007). Notions of security and opacity in discrete event systems. In *Proc. 46th IEEE Conf. Decis. Control*, 5056–5061.

Smith, S.L., Tumova, J., Belta, C., and Rus, D. (2011). Optimal path planning for surveillance with temporal-logic constraints. *Int. J. Robot. Res.*, 30(14), 1695–1708.

Wang, Y., Nalluri, S., and Pajic, M. (2020). Hyperproperties for robotics: Planning via HyperLTL. In *IEEE Int. Conf. Robot. Autom.*, 8462–8468.

Wolper, P., Vardi, M.Y., and Sistla, A.P. (1983). Reasoning about infinite computation paths. In *Proc. 24th Annu. Symp. Found. Comput. Sci.*, 185–194.

Xie, Y., Yin, X., Li, S., and Zamani, M. (2021). Secure-by-construction controller synthesis for stochastic systems under linear temporal logic specifications. In *60th IEEE Conf. Decis. Control*, 7015–7021.

Yang, S., Yin, X., Li, S., and Zamani, M. (2020). Secure-by-construction optimal path planning for linear temporal logic tasks. In *Proc. 59th IEEE Conf. Decis. Control*, 4460–4466.

Yin, X., Zamani, M., and Liu, S. (2021). On approximate opacity of cyber-physical systems. *IEEE Trans. Autom. Control*, 66(4), 1630–1645.

Yu, X., Yin, X., Li, S., and Li, Z. (2022). Security-preserving multi-agent coordination for complex temporal logic tasks. *Control Eng. Pract.*, 123, 105130.

Zhong, B., Liu, S., Caccamo, M., and Zamani, M. (2025). Secure-by-construction synthesis for control systems. *IEEE Trans. Autom. Control*, 70(6), 4170–4177.