STLGame: Signal Temporal Logic Games in Adversarial Multi-Agent Systems

Shuo Yang[1] YANGS1@SEAS.UPENN.EDU
Hongrui Zheng[1] HONGRUIZ@SEAS.UPENN.EDU
Cristian-Ioan Vasile[2] CVASILE@LEHIGH.EDU
George J. Pappas[1] PAPPASG@SEAS.UPENN.EDU
Rahul Mangharam[1] RAHULM@SEAS.UPENN.EDU
[1] *University of Pennsylvania, Philadelphia, PA, USA*
[2] *Lehigh University, Bethlehem, PA, USA*

Editors: N. Ozay, L. Balzano, D. Panagou, A. Abate

Abstract

We study how to synthesize a robust and safe policy for autonomous systems under signal temporal logic (STL) tasks in adversarial settings against unknown dynamic agents. To ensure the worst-case STL satisfaction, we propose STLGame, a framework that models the multi-agent system as a two-player zero-sum game, where the ego agents try to maximize the STL satisfaction and other agents minimize it. STLGame aims to find a Nash equilibrium policy profile, which is the best case in terms of robustness against unseen opponent policies, by using the fictitious self-play (FSP) framework. FSP iteratively converges to a Nash profile, even in games set in continuous state-action spaces. We propose a gradient-based method with differentiable STL formulas, which is crucial in continuous settings to approximate the best responses at each iteration of FSP. We show this key aspect experimentally by comparing with reinforcement learning-based methods to find the best response. Experiments on two standard dynamical system benchmarks, Ackermann steering vehicles and autonomous drones, demonstrate that our converged policy is almost unexploitable and robust to various unseen opponents' policies. All code and additional experimental results can be found on our project website: https://sites.google.com/view/stlgame

Keywords: Signal temporal logic, game theory, multi-agent systems

1. Introduction

Safety-critical autonomous systems operate in dynamic environments to facilitate human life. These autonomous systems usually have complex tasks that are critical for safety and time and have logical dependency between tasks. For instance, Amazon drones[1] may need to deliver items to customers within a specified time and visit a particular warehouse every hour; in addition, safety, such as no collision with other drones or obstacles, is the top priority. These complex task and safety specifications can be expressed naturally as a temporal logic formula Fainekos et al. (2009); Smith et al. (2011), which combines temporal and logical operators to formally define these tasks in a rigorous way. Dynamic environments typically involve multiple active agents. Multi-agent systems control under temporal logic specifications has been studied in the past mainly assuming agents are

1. https://www.aboutamazon.com/news/transportation/amazon-drone-delivery-arizona

cooperative Lindemann and Dimarogonas (2019); Sun et al. (2022). However, this does not hold in many cases. Other agents can be independent or even adversarial to ego agents. For instance, drones from different companies can be sharing the same airspace, posing risks on the task completion and even safety of each other. Moreover, other agents may adopt non-stationary policies, which makes it even more challenging to synthesize a safe and robust policy.

In this work, we consider the problem where the ego agents'[2] task is specified by a signal temporal logic (STL) formula Maler and Nickovic (2004), and we desire to synthesize a control policy for ego agents such that the STL task can be satisfied. The challenging aspect here is that the ego agents share the environment with other a-priori unknown, independent, and potentially adversarial agents. To ensure that STL task is satisfied as much as possible even in the worst case, we formulate **STLGame** (Signal Temporal Logic Game) by modeling the system as a two-player zero-sum stochastic game where ego agents try to maximize STL satisfaction and opponents aim to minimize the same STL satisfaction. STLGame finally concerns finding a Nash Equilibrium (NE) policy for ego agents. The advantage of using an NE policy is that it provides the best-case expected returns against any possible unknown opponent agents. STLGame may be somewhat conservative in practice because other agents are not necessarily adversarial, but it provides the safest and most robust solution when no information about other agents is revealed beforehand.

The proposed STLGame essentially asks us to find a Nash equilibrium policy profile in continuous environments where the level of STL satisfaction/robustness is the utility function. However, converging to Nash equilibrium is generally extremely challenging, if not impossible, in continuous action spaces. We leverage the classic Fictitious Play (FP) Brown (1951) framework from the game theory community in this work, which provably converges to NE in two-player zero-sum games. In addition, another key aspect of enabling STLGame's success in the continuous space is leveraging differentiable STL formulas Pant et al. (2017); Leung et al. (2023), which significantly improves sample efficiency when optimizing a control policy. To summarize, our contributions are as follows:

1. We formulate STLGame, a robust STL synthesis framework that ensures worst-case STL satisfaction against dynamic opponent agents, and use the iterative fictitious play to converge to Nash equilibrium policies;
2. To compute the best response policy at each iteration of FP, we propose an STL gradient-based method that can synthesize an approximate best response against opponents' policies;
3. We compare the gradient-based method with model-free reinforcement learning for best responses and empirically demonstrate its effectiveness, convergence and sample efficiency;
4. We implement our framework on Ackermann steering vehicles and autonomous drones, and empirically show that the converged policy is almost unexploitable, and robust to unseen opponents. Our code is publicly available[3].

1.1. Related Work

Temporal Logic Games With (linear) temporal logic (LTL) specifications, games are usually studied in abstracted transition systems with finite discrete states, such as Büchi game and Rabin game Baier and Katoen (2008); Belta et al. (2017), where the non-determinism in the environment is modeled as the second player Kloetzer and Mahulea (2012). Cui et al. (2023) proposes a novel framework and algorithm for general non-zero-sum settings to enable both players to achieve

2. Hereafter, we use *ego agents* to denote those controllable agents, and use *other agents* and *opponents* interchangeably to denote those uncontrollable agents.
3. https://github.com/shuoyang2000/STLgame

their objectives. Our work instead considers games with STL specification which is evaluated on real-time and real-valued signals, and the system has continuous state-action spaces, so the game complexity is much higher.

Vande Kamp et al. (2023) formulates a potential game for multi-agent systems under STL specifications, assuming individual agent's utility function is aligned with the global objective function. Yu et al. (2023) also considers multi-agents control under STL tasks where opponents' policy is unknown, but assuming opponents' policy is stationary and some data reflecting the policy can be accessed. Unlike these two works, we consider the adversarial case and do not assume stationary policy or policy data access on opponents. The closest work to ours is Muniraj et al. (2018), in which the authors also consider STL zero-sum games and use minimax Q-learning Littman (1994) to converge to Nash policy. However, Muniraj et al. (2018) mainly focuses on discrete action spaces as it is not clear how to effectively apply minimax Q-learning to continuous spaces without significantly sacrificing control performance. Our work instead bridges this gap using fictitious play and differentiable STL formulas. To the best of our knowledge, this is the first work to study temporal logic-based adversarial games in continuous state-action spaces.

Fictitious Play For many discrete games such as poker, counterfactual regret minimization (CFR) method and its variants Zinkevich et al. (2007); Brown and Sandholm (2019) are state-of-the-art. In general, it is still not clear how to extend CFR into continuous games, although there is some domiain-specific results Zheng et al. (2024). Fictitious self-play Heinrich et al. (2015), on the other hand, can generalize to continuous games easily Goldstein and Brown (2022) where a best response policy in continuous space can be computed at each iteration.

2. Background

2.1. Signal Temporal Logic

In this work, we use signal temporal logic (STL) to describe the tasks that ego agents aim to achieve. STL was introduced in Maler and Nickovic (2004) and its syntax is defined as

$$\phi ::= \texttt{True} \mid \pi^{\mu} \mid \neg\phi \mid \phi_1 \wedge \phi_2 \mid \phi_1 \mathbf{U}_{[a,b]} \phi_2, \tag{1}$$

where $\pi^{\mu} : \mathbb{R}^n \to \{\texttt{True}, \texttt{False}\}$ is a predicate whose truth value is determined by the sign of the function $\mu : \mathbb{R}^n \to \mathbb{R}$. The symbols $\neg$ and $\wedge$ denote the Boolean operators "negation" and "conjunction" respectively, and a, b are non-negative scalars ($a \leq b$) used to denote time lower and upper bounds respectively. $\mathbf{U}_{[a,b]}$ denotes the temporal operator "*until*". Note that these basic operators can be combined to define some other commonly-used temporal and logical operators such as "eventually" $\Diamond_{[a,b]}\phi := \texttt{True}\mathbf{U}_{[a,b]}\phi$, "always" $\Box_{[a,b]}\phi := \neg\Diamond_{[a,b]}\neg\phi$, "disjunction" $\phi_1 \vee \phi_2 := \neg(\neg\phi_1 \wedge \neg\phi_2)$, and "implication" $\phi_1 \to \phi_2 := \neg\phi_1 \vee \phi_2$.

STL Qualitative Semantics. STL specifications are evaluated over signal/trajectory $\mathbf{s} = s_0 s_1 \cdots$. We use $(\mathbf{s}, k) \models \phi$ to denote that $\mathbf{s}$ satisfies the STL formula ϕ at time k. Formally, STL semantics are defined as follows:

$$(\mathbf{s}, k) \models \pi^{\mu} \iff \mu(s_k) \geq 0, \quad (\mathbf{s}, k) \models \neg\phi \iff (\mathbf{s}, k) \not\models \phi,$$
$$(\mathbf{s}, k) \models \phi_1 \wedge \phi_2 \iff (\mathbf{s}, k) \models \phi_1 \text{ and } (\mathbf{s}, k) \models \phi_2,$$
$$(\mathbf{s}, k) \models \phi_1 \mathbf{U}_{[a,b]} \phi_2 \iff \exists k' \in [k+a, k+b], \text{s.t. } (\mathbf{s}, k') \models \phi_2. \text{ and } (\mathbf{s}, k'') \models \phi_1, \forall k'' \in [k, k'].$$

The intuition of $\phi_1 \mathbf{U}_{[a,b]} \phi_2$ is that ϕ_2 can hold some time between $[k+a, k+b]$ in the future and ϕ_1 will always hold until then. For convenience, we write $\mathbf{s} \models \phi$ instead of $(\mathbf{s}, 0) \models \phi$.

STL Quantitative Semantics. In addition to the STL qualitative Boolean semantics, we can also define the STL *quantitative* semantics $\rho^{\phi}(\mathbf{s}, k) \in \mathbb{R}$, also called *robustness*, that indicates how much

ϕ is satisfied or violated Fainekos and Pappas (2009); Donzé and Maler (2010). The robustness degree $\rho^{\phi}(\mathbf{s}, k)$ is defined as follows:

$$\rho^{\texttt{True}}(\mathbf{s}, k) = \infty, \qquad \rho^{\pi^{\mu}}(\mathbf{s}, k) = \mu(\mathbf{s}_k),$$
$$\rho^{\neg\phi}(\mathbf{s}, k) = -\rho^{\phi}(\mathbf{s}, k), \qquad \rho^{\phi_1 \wedge \phi_2}(\mathbf{s}, k) = \min(\rho^{\phi_1}(\mathbf{s}, k), \rho^{\phi_2}(\mathbf{s}, k)),$$
$$\rho^{\phi_1 \mathbf{U}_{[a,b]} \phi_2}(\mathbf{s}, k) = \max_{k' \in [k+a, k+b]} \min(\rho^{\phi_2}(\mathbf{s}, k'), \min_{k'' \in [k, k']} \rho^{\phi_1}(\mathbf{s}, k'')).$$

For any trajectory $\mathbf{s}$ and STL ϕ, we have that $\mathbf{s} \models \phi$ if and only if $\rho^{\phi}(\mathbf{s}, 0) \geq 0$. Similarly, we write $\rho^{\phi}(\mathbf{s})$ instead of $\rho^{\phi}(\mathbf{s}, 0)$ for simplicity. In addition, we define the horizon length T_{ϕ} of an STL ϕ as the duration of time needed to verify whether a signal satisfies ϕ Dokhanchi et al. (2014); Sadraddini and Belta (2015). For instance, the horizon length of $\phi = \Diamond_{[1,10]} \Box_{[2,5]} \pi^{\mu}$ is 15.

2.2. Stochastic Game

We consider the stochastic game Shapley (1953) $G = (\mathcal{N}, S, A, f, r, \rho_0, T, \gamma)$ with N agents, where $\mathcal{N} = \{1, 2, \cdots, N\}$ denotes the set of N agents, $S := \times_{i \in \mathcal{N}} S^i \subseteq \mathbb{R}^n$ and $A := \times_{i \in \mathcal{N}} A^i$ are the set of states and joint actions respectively, $f : S \times A \to S$ is the deterministic transition function, $r^i : S \times A \times S \to \mathbb{R}$ is the reward function for agent i, ρ_0 represents the distribution of initial conditions, $T \in \mathbb{N}$ denotes the time horizon, $\gamma \in [0, 1]$ is the discount factor. At timestep t, each agent i picks an action a_t^i according to its (potentially stochastic) policy $\pi^i \in \Pi^i : \sum_{a^i \in A^i} \pi^i(a^i | s) = 1$ and the system state evolves according to the joint action $a_t = \times_{i \in \mathcal{N}} a_t^i$ by discrete-time dynamics

$$s_{t+1} = f(s_t, a_t), \tag{2}$$

where f is a continuous and nonlinear function in $S \times A$. A joint policy is $\pi = (\pi^1, \pi^2, \cdots, \pi^N)$.

Definition 1 *A* partially observable stochastic game (POSG) *Hansen et al. (2004) is defined by the same elements of a stochastic game and additionally defines for each agent $i \in \mathcal{N}$: 1) a set of observations O_i, and 2) observation function $\mathcal{O}_i : A \times S \times O_i \to [0, 1]$.*

Following Albrecht et al. (2024), in a POSG, we define the *history* by $\hat{h}_t = \{(s_\tau, o_\tau, a_\tau)_{\tau=0}^{t-1}, s^t, o^t\}$ up to time t, consisting of the joint states, observations, and actions of all agents in each time step before t, and the current state s_t and current observation o_t. We use $\sigma(\hat{h})$ to denote the history of observations from the full history $\hat{h}$. We denote $s(\hat{h}) = s_t$ the last state in $\hat{h}$ and denote $\langle\rangle$ the concatenation operator. For any joint policy profile $\pi = \langle \pi^i, \pi^{-i} \rangle$ where $-i$ denotes all other agents except i and any history $\hat{h}$, two interlocked functions V and Q are defined as

$$V_{\pi}^i(\hat{h}) = \sum_{a \in A} \pi(a \mid \sigma(\hat{h})) Q_{\pi}^i(\hat{h}, a), \tag{3}$$

$$Q_{\pi}^i(\hat{h}, a) = r^i(s(\hat{h}), a, s') + \gamma \sum_{o' \in O} \mathcal{O}(o' | a, s') V_{\pi}^i(\langle \hat{h}, a, s', o' \rangle), \text{where } s' = f(s(\hat{h}), a), \tag{4}$$

where $V_{\pi}^i(\hat{h})$ is the expected return (or value function) for agent i at the history $\hat{h}$ under the joint policy profile π, and $Q_{\pi}^i(\hat{h}, a)$ is the expected return (or Q function) for agent i when it executes the action a after the history $\hat{h}$ under the joint policy profile π. Note that the policy chooses actions based on observation history here instead of the current state like in vanilla stochastic games. We define the *expected return* for agent i from the initial state of the POSG as $U^i(\pi) = \mathbb{E}_{s_0 \sim \rho_0, o_0 \sim \mathcal{O}(\cdot | \emptyset, s_0)}[V_{\pi}^i(s_0, o_0)]$. The set of *best response* (BR) policies for agent i against π^{-i} is defined as

$$\mathcal{BR}(\pi^{-i}) := \arg\max_{\pi^i \in \Pi^i} U^i(\langle \pi^i, \pi^{-i} \rangle),$$

where $\langle \pi^i, \pi^{-i} \rangle$ is the joint policy profile consisting of π^i and π^{-i}. We write $U^i(\pi^i, \pi^{-i})$ instead of $U^i(\langle \pi^i, \pi^{-i} \rangle)$ for simplicity. Note that there may exist many BR policies w.r.t. to a π^{-i}. The set of ϵ-*approximate best response* (BR) policies for agent i against π^{-i} is defined as

$$\mathcal{BR}_\epsilon(\pi^{-i}) := \{\pi^i \in \Pi^i : U^i(\pi^i, \pi^{-i}) \geq U^i(\mathcal{BR}(\pi^{-i}), \pi^{-i}) - \epsilon\}.$$

A *Nash equilibrium* is a joint policy profile π such that $\pi^i \in \mathcal{BR}(\pi^{-i})$ for any $i \in \mathcal{N}$. For $\epsilon > 0$, an ϵ-Nash equilibrium is a joint policy profile π such that $\pi^i \in \mathcal{BR}_\epsilon(\pi^{-i})$ for any $i \in \mathcal{N}$. We then define the *exploitability* of any joint policy profile π as

$$\mathcal{E}(\pi) := \sum_{i \in \mathcal{N}} \Big(U^i(\mathcal{BR}(\pi^{-i}), \pi^{-i}) - U^i(\pi^i, \pi^{-i}) \Big). \tag{5}$$

Intuitively, exploitability can measure the distance between the current policy and a Nash equilibrium policy. Note that when the exploitability $\mathcal{E}(\pi)$ is 0, then π is a Nash equilibrium policy profile. And exploitability of ϵ yields at least an ϵ-Nash equilibrium. In addition, a stochastic game is a *zero-sum game* if $\sum_{i \in \mathcal{N}} r^i(s, a, s') = 0$ for any state-action pair (s, a) and next state $s' = f(s, a)$.

3. Problem Formulation

In this work, we consider the two-player zero-sum stochastic game. In other words, we have two teams of agents with one team of M agents (ego agents) trying to maximize a utility function and another team of $N - M$ agents (opponent agents) trying to minimize the utility function. We assume that all agents within a team are fully cooperative. Without loss of generality, we assume that $N = 2$ and $M = 1$ in this work, i.e., each team has only one agent.[4]

The mission that the ego agent is expected to achieve is expressed by an STL formula ϕ with bounded horizon $T_\phi = T > 0$. Since the quantitative satisfaction of STL is evaluated on the full trajectory $\mathbf{s} = s_0 s_1 \cdots s_T$, the reward function for ego agent under a general STL specification ϕ is formally defined as

$$r^1(s(\hat{h}_t), a, s') = \begin{cases} 0 & \text{if } t < T \\ \rho^\phi(\mathbf{s}) & \text{if } t = T \end{cases} \tag{6}$$

and the reward function for opponent agent is defined as $r^{-1}(s(\hat{h}_t), a, s') = -r^1(s(\hat{h}_t), a, s')$. One can define less sparser rewards as in Aksaray et al. (2016) for special specifications where $\phi = \Diamond_{[0,T]} \varphi$ or $\phi = \Box_{[0,T]} \varphi$. The main problem considered in this paper is formulated as follows.

Problem 1 *Given the POSG G with dynamical system (2), given an STL formula ϕ, synthesize policies $\pi^{1,*}$ and $\pi^{-1,*}$ for ego and opponent agents respectively such that:*

$$\pi^{1,*} = \arg\max_{\pi^1 \in \Pi^1} \min_{\pi^{-1} \in \Pi^{-1}} \mathop{\mathbb{E}}_{\mathbf{s} \sim \pi} \rho^\phi(\mathbf{s}), \quad \pi^{-1,*} = \arg\min_{\pi^{-1} \in \Pi^{-1}} \max_{\pi^1 \in \Pi^1} \mathop{\mathbb{E}}_{\mathbf{s} \sim \pi} \rho^\phi(\mathbf{s}), \tag{7}$$

where $\mathbf{s} \sim \pi$ means that the trajectory $\mathbf{s}$ is sampled under policy profile π.

In other words, the ego (opponent) agent is synthesizing a policy that can maximize (minimize) the STL satisfaction robustness under the worst case from the opponent (ego). It can also be interpreted as a robust policy that optimizes the worst-case return, and it is essentially the best choice when the agent is unaware of the uncertainty in the environment or what policy the other agent will use. Any policy solution profile $\pi^* = (\pi^{1,*}, \pi^{-1,*})$ from (7) is called a *minimax solution*. Interestingly, in two-player zero-sum games, the set of minimax solutions coincides with the set of Nash equilibrium Owen (2013). Therefore, to solve (7), we need to find a Nash policy profile.

4. The results developed in this work can be transferred to the case with more agents in each team. However, more agents lead to higher complexity to learn the cooperative policy inside the team. We leave it for future work.

4. Fictitious Play for STLGame

Fictitious Play Brown (1951) is an iterative algorithm that can provably converges to a Nash equilibrium in two-player zero-sum games. The players (ego and opponent agents) play the game repeatedly, and each player adopts a best response policy to the average policy of the other player at each iteration. As shown in Algorithm (1), the agents initialize their average policy using a random policy. At each iteration k, the agent computes a best response policy β^i_{k+1} to the other agent's current average policy π^{-i}_k and then update its average policy by

$$\pi^i_{k+1} = \frac{k}{k+1}\pi^i_k + \frac{1}{k+1}\beta^i_{k+1}. \tag{8}$$

The key challenge in fictitious play is the computation of best response policy, especially in environments with continuous action space. To mitigate this issue, a more general version of fictitious play, called *generalized weakened fictitious play* Leslie and Collins (2006), is proposed by tolerating approximate best responses and perturbed average policy while still converging to Nash equilibrium.

Definition 2 *A generalized weakened fictitious play is a process of mixed strategies* $\{\pi_k \in \Pi\}$*:*

$$\pi^i_{k+1} \in (1-\alpha_{k+1})\,\pi^i_k + \alpha_{k+1}\left(\mathcal{BR}_{\epsilon_k}(\pi^{-i}_k) + M^i_{k+1}\right), \forall i \in \{1,-1\},$$

with $\alpha_k \to 0$ *and* $\epsilon_k \to 0$ *when* $k \to \infty$, $\sum_{k=1}^{\infty}\alpha_k = \infty$, *and* M_k *is a sequence of perturbations that satisfies*

$$\lim_{k\to\infty}\sup_j\left\{\left\|\sum_{l=k}^{j-1}\alpha_{l+1}M_{l+1}\right\| \text{ s.t. } \sum_{l=k}^{j-1}\alpha_{l+1} \le J\right\} = 0, \forall J \ge 0.$$

One can notice that naive fictitious play is a special case of generalized weakened fictitious play with $\alpha_k = \frac{1}{k}$ (average step size), $\epsilon_k = 0$ (strict best response) and $M_k = 0$ (no perturbation).

Algorithm 1 Fictitious Play

Initialize π^1_0, π^{-1}_0 with random policies
for $k = 0, \dots, K$ **do**
  for $i = 1, -1$ **do**
    $\beta^i_{k+1} = \mathcal{BR}(\pi^{-i}_k)$
    $\pi^i_{k+1} = \frac{k}{k+1}\pi^i_k + \frac{1}{k+1}\beta^i_{k+1}$

4.1. Best Responses in STLGame

The main step of using the (generalized weakened) fictitious play is still the computation of (approximate) best responses. In the context of STLGames, the best response computation at each iteration is synthesizing the optimal control policy given an opponent agent adopting a fixed but unknown average policy (i.e., π^{-1}_k at iteration k) in the environment. Notice that the opponent policy π^{-1}_k is fixed, stochastic, and unknown to the ego agent, so the best response computation for the ego agent in STLGame is formulated as the following problem[5]:

Problem 2 *Given the POSG* G *with dynamical system (2) where the opponent agent is playing an unknown policy* π^{-1}_k*, given an STL formula* ϕ*, synthesize an optimal control policy that maximizes STL robustness:* $\pi^{1,*}_k = \arg\max_{\pi^1_k \in \Pi^1} \mathbb{E}_{\mathbf{s}\sim\langle\pi^1_k,\pi^{-1}_k\rangle}\,\rho^{\phi}(\mathbf{s})$.

Due to the fact that the opponent has an unknown policy, it is not clear how to use classic approaches such as mixed-integer linear program (MILP)-based method Raman et al. (2014) to synthesize an optimal policy, as they typically assume access to the precise model of the entire environment.

5. Opponent agent's BR synthesis problem can be formulated in a dual manner. We only analyze the case of an ego agent here for simplicity.

Regardless of the lack of knowledge on opponent's policy, we can still interact with it and gradually have better understanding on its policy, and then compute/learn a (near)-optimal policy. This motivates us to use reinforcement learning to (approximately) solve Problem (2).

Reinforcement Learning for BR. Some existing work tried to leverage reinforcement learning (RL) techniques to facilitate STL policy synthesis in both single agent system Li et al. (2017); Venkataraman et al. (2020); Wang et al. (2024) and multi agents system Wang et al. (2023). RL training is guided by a reward function, which is the robustness value of the rollout trajectory w.r.t. the STL specification in our case. For a general STL formula ϕ, the reward function is defined as (6). One can use any model-free RL methods such as Q-learning Watkins and Dayan (1992) for environments with discrete action space and Proximal Policy Optimization (PPO) Schulman et al. (2017) for continuous environments.

However, the reward function in (6) is sparse because the robustness value is not accessible until the end of the trajectory for a general specification. Learning with sparse reward signals is a challenging exploration problem for RL algorithms, especially for continuous environments. The sample efficiency for sparse reward setting may be concerning, which influences the (approximate) best response quality if the exploration is terminated once the computation budget is depleted. Finally, the iterated fictitious play process may be unstable due to the under-par response policy.

Gradient-based Method for BR. One drawback of using reinforcement learning for BR synthesis is that it is unaware of the STL information but only of the STL final robustness values. However, STL has rich information such as its gradients, has not been used by RL when computing a response policy. Recent work Pant et al. (2017); Leung et al. (2023) shows that STL robustness formulas can be represented as computation graphs, enabling robustness gradient computation with respect to the action. Thanks to the STL robustness gradients, we can directly use gradient descent algorithm to learn a control policy represented by a neural network (see Figure (1)). Specifically, at each training iteration, we rollout the system with the policy network, compute the STL robustness value over full trajectories, and update the policy network parameters to maximize robustness via backpropagation and gradient descent. This gradient-based method is expected to require significantly less computation budget than using RL. Thus, the trained policies under the same computation budget are expected to be much closer to an actual best response.

We implement and compare both the reinforcement learning and the gradient-based method to compute the best responses in the experiments. To calculate the average policy in fictitious play, one can represent the average policy using a single stochastic network, and train this network using a supervised learning approach to mimic the behavior of the ground truth average policy; see e.g., Heinrich et al. (2015). In the experiment, we observe that using the supervised learning-based average policy does not work well in STL games, so we follow the average rule (8) directly by sampling the set of found BR policies according to the sampling weight produced by (8) and it produces more stable and less exploitable results.

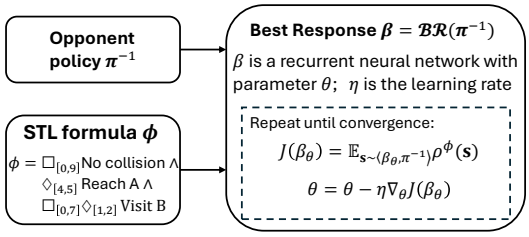

Figure 1: STL Gradient-based Best Response

BR Policy Representation Unlike traditional RL/control methods, which make decisions mainly based on the current observation or state, STL control policy need to also depend on the history observations/states sequence for two reasons: 1) STL formula is evaluated over the whole trajectory so the future trajectory should highly depend on the past trajectory; 2) ego/opponent agent's policy

information is implicitly encoded into the history observation sequence and single observation does not reflect the policy. Therefore, we use Recurrent Neural Networks (RNNs) Hochreiter (1997) instead of Multi-Layer Perceptrons (MLPs), as control policy networks. For more complicated tasks with high-dimensional observation spaces, one can use Transformers Vaswani et al. (2017) for better representation ability; for relatively simpler tasks, one can choose RNNs such as Long Short-Term Memory (LSTM) Hochreiter (1997) for its simplicity. In this work, we use LSTM as our policy backbone, where the input is the history observations $\sigma(\hat{h}_t)$ and the output is action a_t.

5. Experiments

In our experiments, our main objective is to answer the following three questions.

- **Convergence**: can we learn an almost unexploitable policy profile using our approach in STLGames? In other words, can we approximately obtain the Nash joint policy profile?
- **Efficiency**: is the policy convergence efficient in continuous environments for STLGame?
- **Robustness**: is an ego agent with the converged policy robust to unseen opponents?

Our approach is tested in two simulation benchmarks: Ackermann steering vehicles and autonomous drones. We implement and compare both reinforcement learning and the STL gradient-based method as the best response policy synthesis approaches. In our experiments, we focus on three metrics. First, the **exploitability** (5) of the policy profiles found. Exploitability signals how close we are to the Nash equilibrium. Second, the STL **robustness value**. Robustness signals how much the control policy (un)satisfies the STL specification. Lastly, the **satisfication rate** of STL specification in rollouts. STL specifications are satisfied when the robustness value is non-negative.

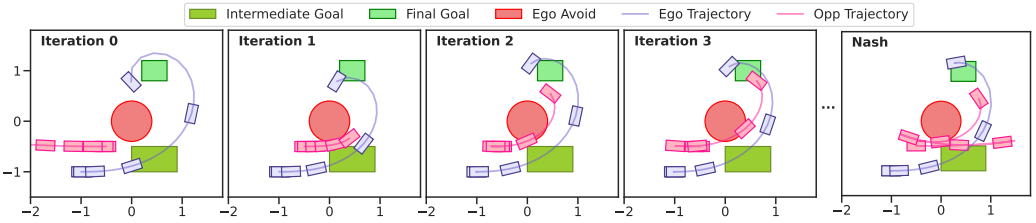

Figure 2: Vehicles trajectories sampled from some FSP iterations and finally from Nash profile. In the trajectories sampled from Nash profile, the opponent is trying its best to block the ego vehicle.

Ackermann Steering Vehicles. The Ackermann steering vehicles are modeled by the following kinematic single-track dynamics. We consider two vehicles with the same dynamics in the environment, as shown in Figure (2). The state space model consists of $\mathbf{x} = [s_x, s_y, \delta, v, \Psi]$, where s_x and s_y denote the position in x and y coordinate, δ is the steering angle, v is the velocity, Ψ is the heading angle, and l_{wb} is the wheelbase. The control inputs are $[u_1, u_2] = [v_\delta, a_{long}]$ where v_δ is the steering angle velocity and a_{long} is the longitude acceleration. The continuous-time system is:

$$\dot{x}_1 = x_4 \cos(x_5), \dot{x}_2 = x_4 \sin(x_5), \dot{x}_3 = f_{steer}(x_3, u_1), \dot{x}_4 = f_{acc}(x_4, u_2), \dot{x}_5 = \frac{x_4}{l_{wb}} \tan(x_3).$$

where f_{steer} and f_{acc} impose physical constraints on steering and acceleration. The system is discretized with sampling time $\Delta t = 0.1s$. The STL task is formally defined as (where $T = 50$):

$$\begin{aligned} \phi = &\Diamond_{[0,T-1]}(q^1 \in \text{Intermediate Goal}) \wedge \Diamond_{[0,T-1]}(q^1 \in \text{Final Goal}) \\ &\wedge \Box_{[0,T-1]} \neg (q^1 \in \text{Red Circle}) \wedge (\|q^1 - q^{-1}\|_2^2 \geq d_{\min}^2). \end{aligned} \tag{9}$$

Intuitively, the ego vehicle should eventually reach the intermediate goal and final goal, and always avoid the red dangerous region, and maintain a safe distance with the opponent vehicle. The ego vehicle aims to satisfy ϕ and the opponent vehicle aims to satisfy $\neg\phi$.

Autonomous Drones. Autonomous drones are nowadays widely deployed in various domains such as forestry, industry, and agriculture. These deployed drones may face some uncertainty or even other unknown agents as adversaries in the environment. Thus, it is critical to develop a robust control policy in the worst-case scenarios. This is formulated as a zero-sum game between two drone players. Specifically, we consider the 3D environment showed in Figure (3), where the ego drone should 1) eventually arrive at the goal area, 2) always avoid the unsafe column, 3) always maintain safe distance with the opponent drone, and 4) always obey the altitude rules in different Zones. They are formalized by the STL formula (similar to Pant et al. (2017)):

$$\phi = \Diamond_{[0,T-1]}(q^1 \in \text{Goal}) \wedge \Box_{[0,T-1]}\big(\neg(q^1 \in \text{Unsafe}) \wedge (\|q^1 - q^{-1}\|_2^2 \geq d_{\min}^2)\big)$$
$$\Box_{[0,T-1]}\big(q^1 \in \text{Zone}_1 \implies z^1 \in [1,5]\big) \wedge \Box_{[0,T-1]}\big(q^1 \in \text{Zone}_2 \implies z^1 \in [0,3]\big), \quad (10)$$

where $T = 50$, and q^1 is the position of ego drone in the x, y, z coordinate, and q^{-1} is the position of opponent drone. We use the same drone dynamics from Pant et al. (2015, 2017) which have been shown successful in real-time quad-rotors control.

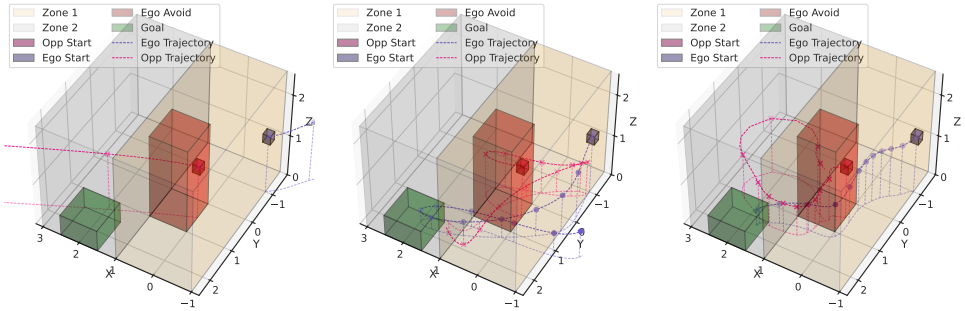

Figure 3: Drone trajectories from randomly initialized policy (Left) to FSP iteration1 (Middle) and finally Nash profile (Right). For better visualizations, please see our website.

Exploitability Convergence and Efficiency. Exploitability measures how much worse the joint policy profile is compared to the Nash profile. A policy is said to be less exploitable if it is closer to a Nash policy. However, a challenge of computing exploitability (see its definition (5)) is that it involves the computation of the best responses. In this work, we empirically consider the best policy learned among all training epochs in the gradient-based method as the best response. It takes around 10 minutes to train each FSP iteration on AMD Ryzen7 4800H.

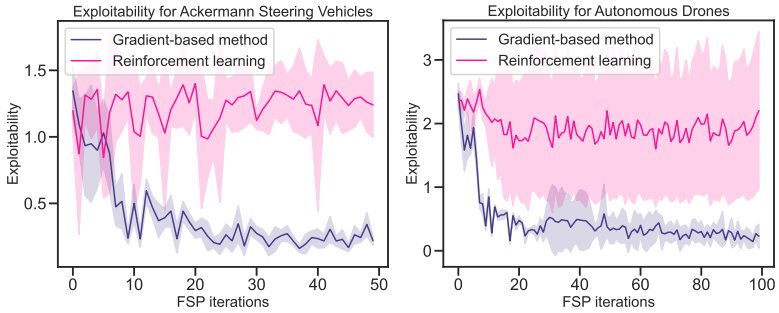

Figure 4: Exploitability for Ackermann Steering Vehicles (Left) and Autonomous Drones (Right). We run five different initial conditions for each environment.

The exploitability curves are plotted in Figure (4) for both benchmarks. Both plots are run over five random initial conditions, and the solid lines denote the mean value, and the shaded region denotes the standard deviation. We observe that FSP with gradient-based BR converges to almost

zero exploitability, signaling that the final average policy profile is approximately Nash. For RL-based BR, we use PPO with LSTM-based actor and critic under the same simulation budget, which fails to converge. PPO struggled to learn effective response policies at each iteration due to sparse reward signals. Interested readers can refer to our website for more experimental results.

Scenario	Policy	Seen Opponents		Unseen Opponents	
		STL Robustness	Satisfaction Rate	STL Robustness	Satisfaction Rate
Vehicles	Nash	-0.06 ± 0.12	18.7%	-0.12 ± 0.17	16.7%
	BR	-0.02 ± 0.02	20.0%	-0.21 ± 0.24	2.0%
Drones	Nash	0.10 ± 0.06	94.7%	0.01 ± 0.27	88.0%
	BR	0.09 ± 0.05	96.0%	-0.73 ± 0.16	0.0%

Table 1: We play Nash policy and best response policy against seen policies and unseen policies. The best response is trained only against the average of seen policies.

Robustness against Unseen Opponents. Nash policy is the best choice for the ego agent, in the sense of worst-case return, when the ego agent has no prior information on the opponents' policy. To test this, we collect elite opponent policies and split them into *seen policy set* and *unseen policy set*. We train a best response policy to only the average of all seen policies. The results are reported in Table (1). We can see that Nash policy is a more conservative in the sense that it is worse than the BR policy when playing against seen policies. However, when it comes to unseen opponents, the BR policy performs significantly worse in terms of both robustness value and STL satisfaction rate. This signals that the Nash policy can be deployed safely to unseen opponents while previously learned BR policy may fail drastically (e.g., in drones example). Some Ackermann vehicle trajectory demonstrations of Nash policy against unseen opponents can be found in Figure (5), and those for the drone experiment can be found on our website.

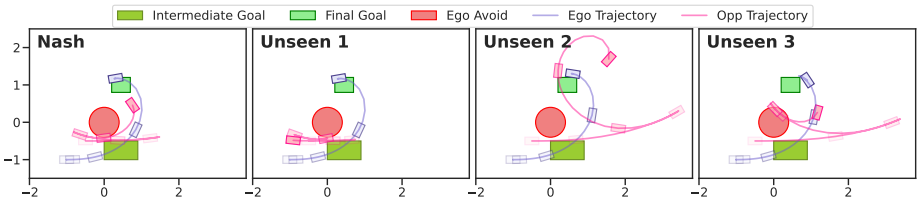

Figure 5: Sampled vehicles trajectories when the ego vehicle is playing a Nash policy, and the opponent is playing a Nash (Left), or some unseen policy (Unseen 1, Unseen 2, Unseen 3).

6. Conclusion

In this paper, we formulate STLGame for continuous environments, where ego agents aim to maximize STL satisfaction while opponent agents aim to minimize it. We propose an STL gradient-based method to learn a best response policy w.r.t. the opponents' average policy, under the classic fictitious play framework, and we show that it can converge to an almost unexploitable policy (Nash policy) effectively and efficiently. The STL gradient-based method significantly outperforms RL for best responses policy learning. The converged policy is empirically shown to be robust to unseen opponent policies, and achieve higher STL robustness values and satisfaction rate compared to the best response policy trained from seen opponents. In the future, we will consider more agents inside the ego and opponent teams and aim to scale the multi-agent control synthesis under STL tasks.

References

Derya Aksaray, Austin Jones, Zhaodan Kong, Mac Schwager, and Calin Belta. Q-learning for robust satisfaction of signal temporal logic specifications. In *2016 IEEE 55th Conference on Decision and Control (CDC)*, pages 6565–6570. IEEE, 2016.

Stefano V. Albrecht, Filippos Christianos, and Lukas Schäfer. *Multi-Agent Reinforcement Learning: Foundations and Modern Approaches*. MIT Press, 2024. URL https://www.marl-book.com.

Christel Baier and Joost-Pieter Katoen. *Principles of model checking*. MIT press, 2008.

Calin Belta, Boyan Yordanov, and Ebru Aydin Gol. *Formal methods for discrete-time dynamical systems*, volume 89. Springer, 2017.

George W Brown. Iterative solution of games by fictitious play. *Act. Anal. Prod Allocation*, 13(1): 374, 1951.

Noam Brown and Tuomas Sandholm. Superhuman ai for multiplayer poker. *Science*, 365(6456): 885–890, 2019.

Bohan Cui, Alessandro Giua, and Xiang Yin. Towards supervisory control theory in tactical environments: A stackelberg game approach. In *2023 62nd IEEE Conference on Decision and Control (CDC)*, pages 7937–7943. IEEE, 2023.

Adel Dokhanchi, Bardh Hoxha, and Georgios Fainekos. On-line monitoring for temporal logic robustness. In *International Conference on Runtime Verification*, pages 231–246. Springer, 2014.

Alexandre Donzé and Oded Maler. Robust satisfaction of temporal logic over real-valued signals. In *International Conference on Formal Modeling and Analysis of Timed Systems*, pages 92–106. Springer, 2010.

Georgios E Fainekos and George J Pappas. Robustness of temporal logic specifications for continuous-time signals. *Theoretical Computer Science*, 410(42):4262–4291, 2009.

Georgios E Fainekos, Antoine Girard, Hadas Kress-Gazit, and George J Pappas. Temporal logic motion planning for dynamic robots. *Automatica*, 45(2):343–352, 2009.

Maxwell Goldstein and Noam Brown. Converging to unexploitable policies in continuous control adversarial games. In *Deep Reinforcement Learning Workshop NeurIPS*, 2022.

Eric A Hansen, Daniel S Bernstein, and Shlomo Zilberstein. Dynamic programming for partially observable stochastic games. In *AAAI*, volume 4, pages 709–715, 2004.

Johannes Heinrich, Marc Lanctot, and David Silver. Fictitious self-play in extensive-form games. In *International conference on machine learning*, pages 805–813. PMLR, 2015.

S Hochreiter. Long short-term memory. *Neural Computation MIT-Press*, 1997.

Marius Kloetzer and Cristian Mahulea. Ltl planning in dynamic environments. *IFAC Proceedings Volumes*, 45(29):294–300, 2012.

David S Leslie and Edmund J Collins. Generalised weakened fictitious play. *Games and Economic Behavior*, 56(2):285–298, 2006.

Karen Leung, Nikos Aréchiga, and Marco Pavone. Backpropagation through signal temporal logic specifications: Infusing logical structure into gradient-based methods. *The International Journal of Robotics Research*, 42(6):356–370, 2023.

Xiao Li, Cristian-Ioan Vasile, and Calin Belta. Reinforcement learning with temporal logic rewards. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3834–3839. IEEE, 2017.

Lars Lindemann and Dimos V Dimarogonas. Control barrier functions for multi-agent systems under conflicting local signal temporal logic tasks. *IEEE control systems letters*, 3(3):757–762, 2019.

Michael L Littman. Markov games as a framework for multi-agent reinforcement learning. In *Machine learning proceedings 1994*, pages 157–163. Elsevier, 1994.

Oded Maler and Dejan Nickovic. Monitoring temporal properties of continuous signals. In *International symposium on formal techniques in real-time and fault-tolerant systems*, pages 152–166. Springer, 2004.

Devaprakash Muniraj, Kyriakos G Vamvoudakis, and Mazen Farhood. Enforcing signal temporal logic specifications in multi-agent adversarial environments: A deep q-learning approach. In *2018 IEEE Conference on Decision and Control (CDC)*, pages 4141–4146. IEEE, 2018.

Guillermo Owen. *Game theory*. Emerald Group Publishing, 2013.

Yash Vardhan Pant, Houssam Abbas, Kartik Mohta, Truong X Nghiem, Joseph Devietti, and Rahul Mangharam. Co-design of anytime computation and robust control. In *2015 IEEE real-time systems symposium*, pages 43–52. IEEE, 2015.

Yash Vardhan Pant, Houssam Abbas, and Rahul Mangharam. Smooth operator: Control using the smooth robustness of temporal logic. In *2017 IEEE Conference on Control Technology and Applications (CCTA)*, pages 1235–1240. IEEE, 2017.

Vasumathi Raman, Alexandre Donzé, Mehdi Maasoumy, Richard M. Murray, Alberto Sangiovanni-Vincentelli, and Sanjit A. Seshia. Model predictive control with signal temporal logic specifications. In *53rd IEEE Conference on Decision and Control*, pages 81–87, 2014. doi: 10.1109/CDC.2014.7039363.

Sadra Sadraddini and Calin Belta. Robust temporal logic model predictive control. In *2015 53rd Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, pages 772–779. IEEE, 2015.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

Lloyd S. Shapley. Stochastic games. *Proceedings of the national academy of sciences*, 39(10): 1095–1100, 1953.

Stephen L Smith, Jana Tumova, Calin Belta, and Daniela Rus. Optimal path planning for surveillance with temporal-logic constraints. *The International Journal of Robotics Research*, 30(14): 1695–1708, 2011.

Dawei Sun, Jingkai Chen, Sayan Mitra, and Chuchu Fan. Multi-agent motion planning from signal temporal logic specifications. *IEEE Robotics and Automation Letters*, 7(2):3451–3458, 2022.

Levi Vande Kamp, Abbasali Koochakzadeh, Yasin Yazicioglu, and Derya Aksaray. A game theoretic approach to distributed planning of multi-agent systems under temporal logic specifications. In *AIAA SCITECH 2023 Forum*, page 1657, 2023.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. Attention is all you need. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf.

Harish Venkataraman, Derya Aksaray, and Peter Seiler. Tractable reinforcement learning of signal temporal logic objectives. In *Learning for Dynamics and Control*, pages 308–317. PMLR, 2020.

Jiangwei Wang, Shuo Yang, Ziyan An, Songyang Han, Zhili Zhang, Rahul Mangharam, Meiyi Ma, and Fei Miao. Multi-agent reinforcement learning guided by signal temporal logic specifications. *arXiv preprint arXiv:2306.06808*, 2023.

Siqi Wang, Xunyuan Yin, Shaoyuan Li, and Xiang Yin. Tractable reinforcement learning for signal temporal logic tasks with counterfactual experience replay. *IEEE Control Systems Letters*, 2024.

Christopher JCH Watkins and Peter Dayan. Q-learning. *Machine learning*, 8:279–292, 1992.

Xinyi Yu, Yiqi Zhao, Xiang Yin, and Lars Lindemann. Signal temporal logic control synthesis among uncontrollable dynamic agents with conformal prediction. *arXiv preprint arXiv:2312.04242*, 2023.

Hongrui Zheng, Zhijun Zhuang, Stephanie Wu, Shuo Yang, and Rahul Mangharam. Bridging the gap between discrete agent strategies in game theory and continuous motion planning in dynamic environments. *arXiv preprint arXiv:2403.11334*, 2024.

Martin Zinkevich, Michael Johanson, Michael Bowling, and Carmelo Piccione. Regret minimization in games with incomplete information. *Advances in neural information processing systems*, 20, 2007.