A Framework for Black-Box Controller Design to Automatically Satisfy Specifications using Signal Temporal Logic

Kristy Sakano

Xu Lab, UMD College Park Department of Aerospace Engineering

International Conference on Unmanned Air Systems 2025

ML-based Control

Machine learning-enabled controllers can accomplish complex and difficult control tasks!

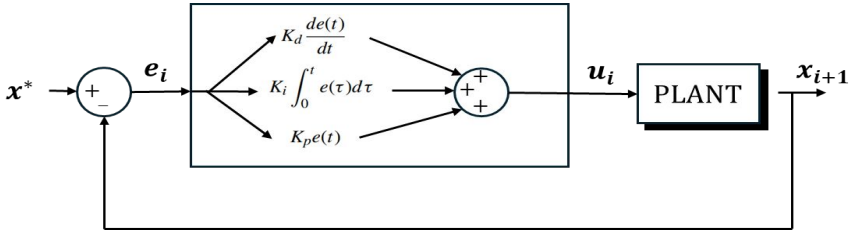
Google's Self-Driving Car (now Waymo)

Lockheed-Martin's Indago 4

Control Design Examples

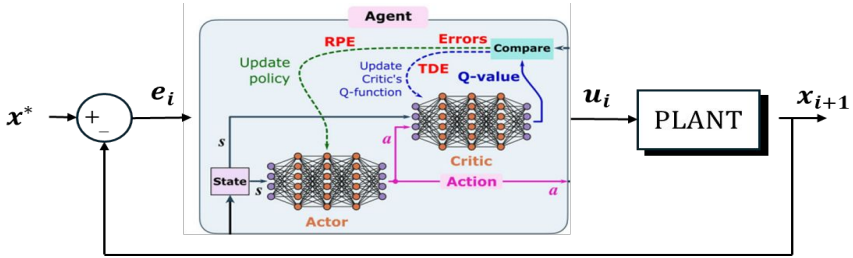
Classical controllers are built *deductively* from requirements, while ML control requirements are *inductively* determined *after* construction.

Classical PID Control System



Deductive: we pick PID coefficients construct a controller that *directly satisfy* a set of control and mission requirements

ML-based Control System

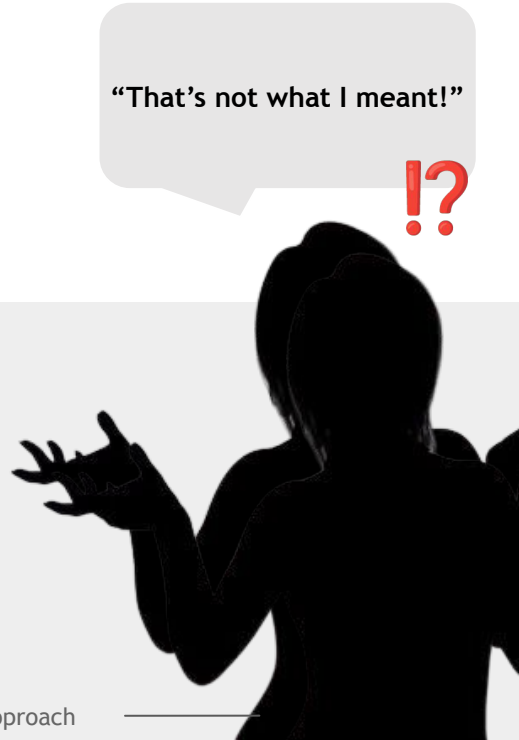


Inductive: exact requirements and performance are only determined *after* construction

Why Formalization Matters for RL Systems

RL Agents are trained to **maximize** rewards, but:

- Rewards **may not capture desired behaviors**
- Agents may **exploit reward loopholes**



Takeaway

- Agents may succeed at the task but fail at the mission.
- We need a **structured way to express requirements.**

What is Signal Temporal Logic (STL)?

Human Readable Requirements

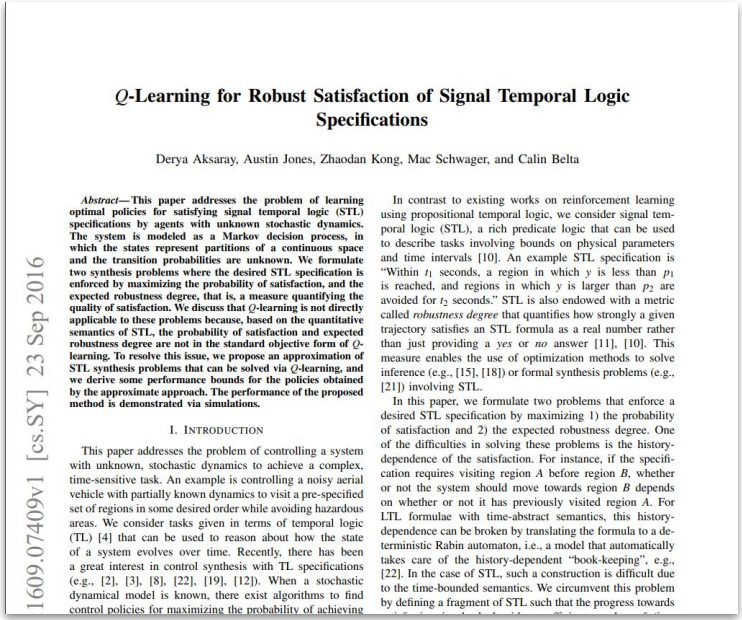
"When the agent is engaged in a turn, the maximum speed should be kept low."

Signal Temporal Logic Specifications

$$\phi_1 = G_{[0,t]}(|\dot{\psi}| > 1 \Rightarrow |v| < \nu_1)$$

STL & RL Related Works

Aksaray et al (2016)



Q-Learning for Robust Satisfaction of Signal Temporal Logic Specifications

Derya Aksaray, Austin Jones, Zhaodan Kong, Mac Schwager, and Calin Belta

***Abstract*— This paper addresses the problem of learning optimal policies for satisfying signal temporal logic (STL) specifications by agents with unknown stochastic dynamics. The system is modeled as a Markov decision process, in which the states represent partitions of a continuous space and the transition probabilities are unknown. We formulate two synthesis problems where the desired STL specification is enforced by maximizing the probability of satisfaction, and the expected robustness degree, that is, a measure quantifying the quality of satisfaction. We discuss that *Q*-learning is not directly applicable to these problems because, based on the quantitative semantics of STL, the probability of satisfaction and expected robustness degree are not in the standard objective form of *Q*-learning. To resolve this issue, we propose an approximation of STL synthesis problems that can be solved via *Q*-learning, and we derive some performance bounds for the policies obtained by the approximate approach. The performance of the proposed method is demonstrated via simulations.**

I. Introduction

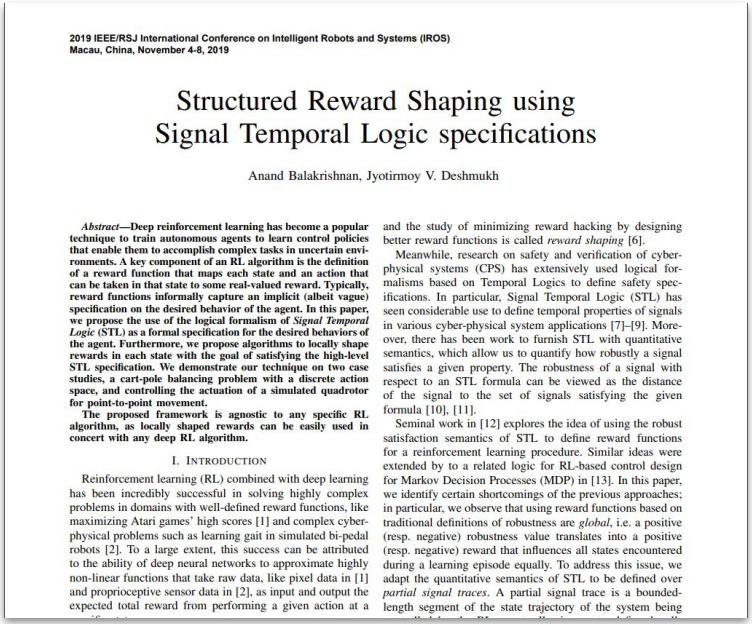
This paper addresses the problem of controlling a system with unknown, stochastic dynamics to achieve a complex, time-sensitive task. An example is controlling a noisy aerial vehicle with partially known dynamics to visit a pre-specified set of regions in some desired order while avoiding hazardous areas. We consider tasks given in terms of temporal logic (TL) [4] that can be used to reason about how the state of a system evolves over time. Recently, there has been a great interest in control synthesis with TL specifications (e.g., [2], [3], [8], [22], [19], [12]). When a stochastic dynamical model is known, there exist algorithms to find control policies for maximizing the probability of achieving

In contrast to existing works on reinforcement learning using propositional temporal logic, we consider signal temporal logic (STL), a rich predicate logic that can be used to describe tasks involving bounds on physical parameters and time intervals [10]. An example STL specification is "Within t_1 seconds, a region in which y is less than p_1 is reached, and regions in which y is larger than p_2 are avoided for t_2 seconds." STL is also endowed with a metric called *robustness degree* that quantifies how strongly a given trajectory satisfies an STL formula as a real number rather than just providing a *yes* or *no* answer [11], [10]. This measure enables the use of optimization methods to solve inference (e.g., [15], [18]) or formal synthesis problems (e.g., [21]) involving STL.

In this paper, we formulate two problems that enforce a desired STL specification by maximizing 1) the probability of satisfaction and 2) the expected robustness degree. One of the difficulties in solving these problems is the history-dependence of the satisfaction. For instance, if the specification requires visiting region A before region B, whether or not the system should move towards region B depends on whether or not it has previously visited region A. For LTL formulae with time-abstract semantics, this history-dependence can be broken by translating the formula to a deterministic Rabin automaton, i.e., a model that automatically takes care of the history-dependent "book-keeping", e.g., [22]. In the case of STL, such a construction is difficult due to the time-bounded semantics. We circumvent this problem by defining a fragment of STL such that the progress towards

STL Specifications → transform to Q-learning reward → maximize robustness

Balakrishnan and Deshmukh (2019)



2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
Macau, China, November 4-8, 2019

Structured Reward Shaping using Signal Temporal Logic specifications

Anand Balakrishnan, Jyotirmoy V. Deshmukh

***Abstract*—Deep reinforcement learning has become a popular technique to train autonomous agents to learn control policies that enable them to accomplish complex tasks in uncertain environments. A key component of an RL algorithm is the definition of a reward function that maps each state and an action that can be taken in that state to some real-valued reward. Typically, reward functions informally capture an implicit (albeit vague) specification on the desired behavior of the agent. In this paper, we propose the use of the logical formalism of *Signal Temporal Logic* (STL) as a formal specification for the desired behaviors of the agent. Furthermore, we propose algorithms to locally shape rewards in each state with the goal of satisfying the high-level STL specification. We demonstrate our technique on two case studies, a cart-pole balancing problem with a discrete action space, and controlling the actuation of a simulated quadrotor for point-to-point movement.**

The proposed framework is agnostic to any specific RL algorithm, as locally shaped rewards can be easily used in concert with any deep RL algorithm.

I. Introduction

Reinforcement learning (RL) combined with deep learning has been incredibly successful in solving highly complex problems in domains with well-defined reward functions, like maximizing Atari games' high scores [1] and complex cyber-physical problems such as learning gait in simulated bi-pedal robots [2]. To a large extent, this success can be attributed to the ability of deep neural networks to approximate highly non-linear functions that take raw data, like pixel data in [1] and proprioceptive sensor data in [2], as input and output the expected total reward from performing a given action at a

and the study of minimizing reward hacking by designing better reward functions is called *reward shaping* [6].

Meanwhile, research on safety and verification of cyber-physical systems (CPS) has extensively used logical formalisms based on Temporal Logics to define safety specifications. In particular, Signal Temporal Logic (STL) has seen considerable use to define temporal properties of signals in various cyber-physical system applications [7]–[9]. Moreover, there has been work to furnish STL with quantitative semantics, which allow us to quantify how robustly a signal satisfies a given property. The robustness of a signal with respect to an STL formula can be viewed as the distance of the signal to the set of signals satisfying the given formula [10], [11].

Seminal work in [12] explores the idea of using the robust satisfaction semantics of STL to define reward functions for a reinforcement learning procedure. Similar ideas were extended by to a related logic for RL-based control design for Markov Decision Processes (MDP) in [13]. In this paper, we identify certain shortcomings of the previous approaches; in particular, we observe that using reward functions based on traditional definitions of robustness are *global*, i.e. a positive (resp. negative) robustness value translates into a positive (resp. negative) reward that influences all states encountered during a learning episode equally. To address this issue, we adapt the quantitative semantics of STL to be defined over *partial signal traces*. A partial signal trace is a bounded-length segment of the state trajectory of the system being

STL Specifications → local robustness → hyper-local rewards → better RL training

What is unique about our approach?

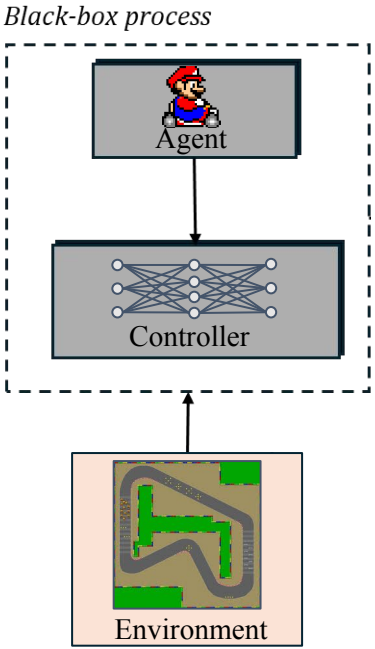
🤝 **Collaborative tuning:** Frequent check-ins with operators to refine acceptance tolerance

⚠️ **Avoid Overfitting:** No brittle, hyper-local reward functions

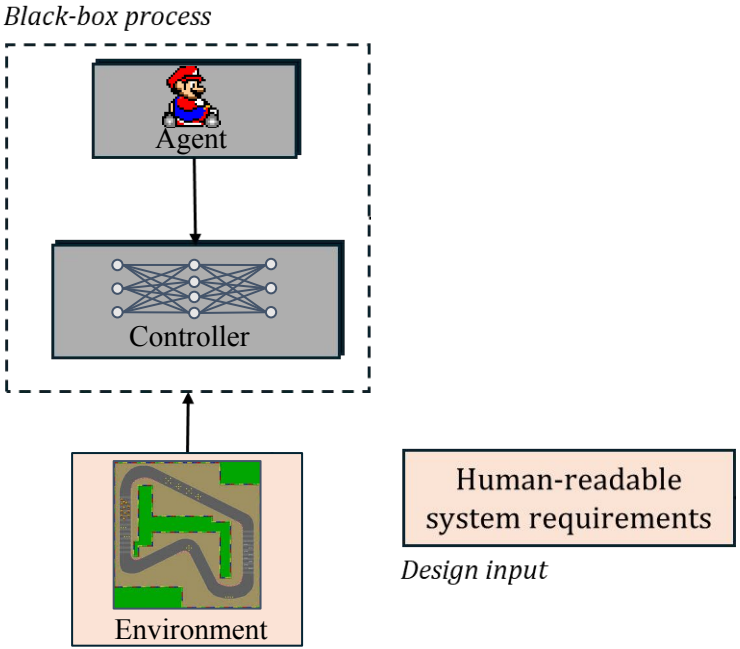
🚳 **No Demos? No problem:** Learn without expert trajectories or imitation learning

🔒 **Black-box controller compatible:** Can't always implement formal robustness guarantees during training

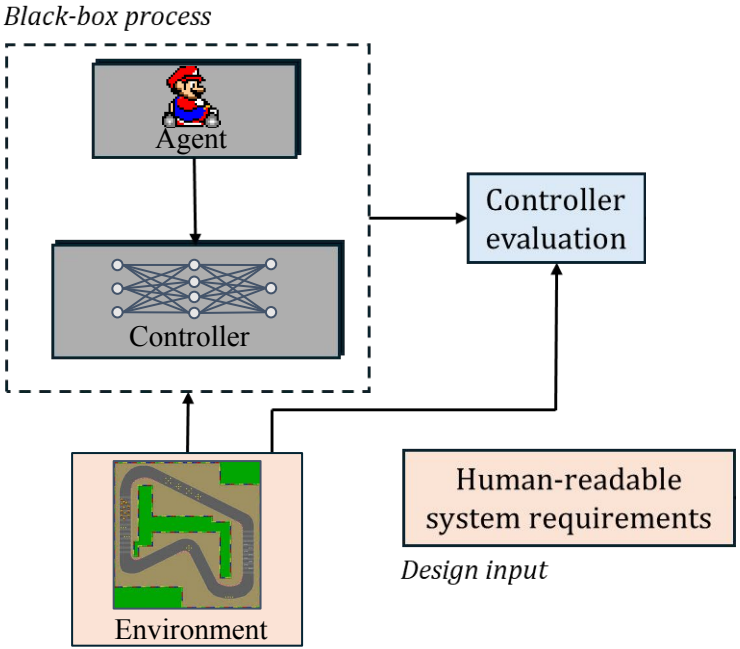
Our Approach to Specification-Driven Iterative Design



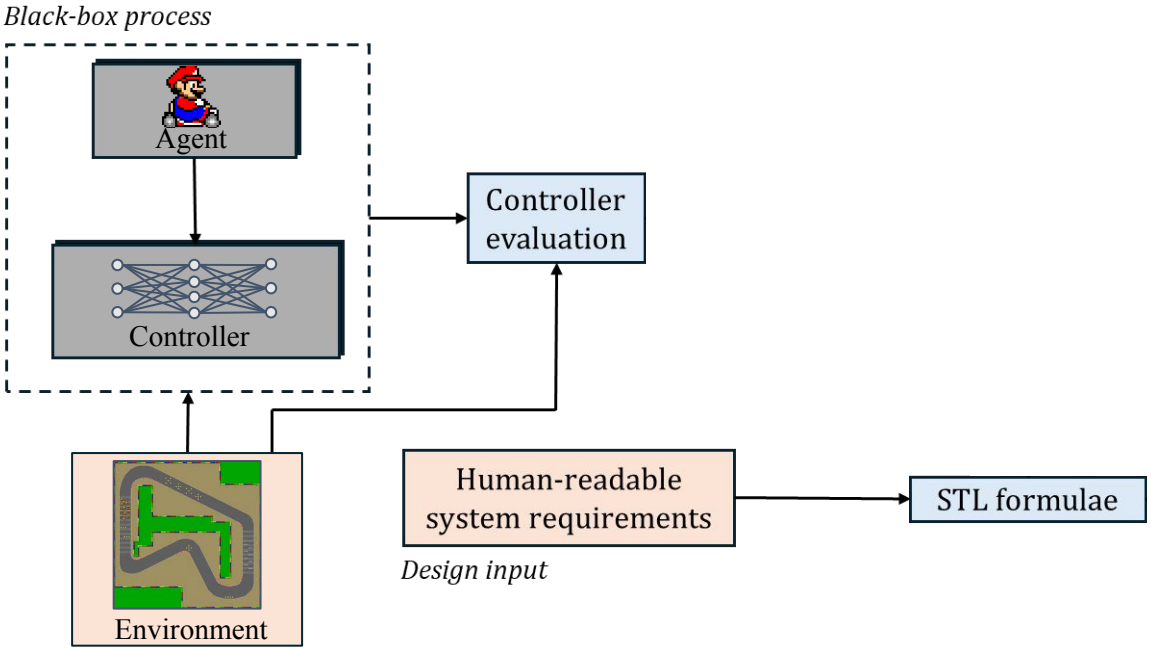
Our Approach to Specification-Driven Iterative Design



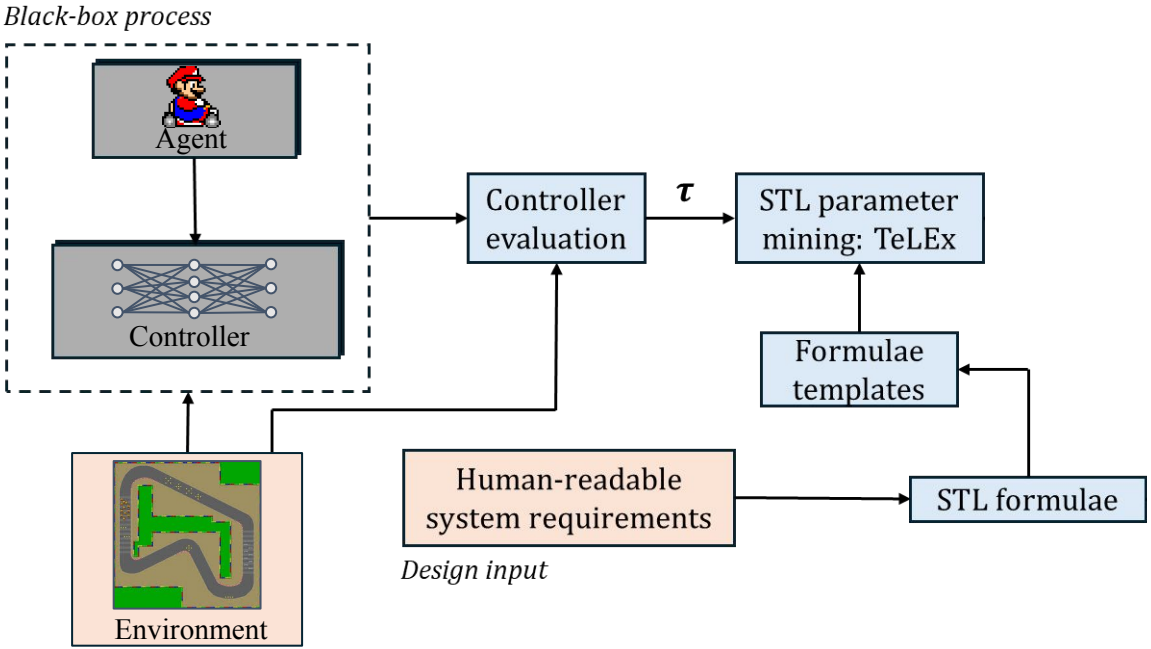
Our Approach to Specification-Driven Iterative Design



Our Approach to Specification-Driven Iterative Design

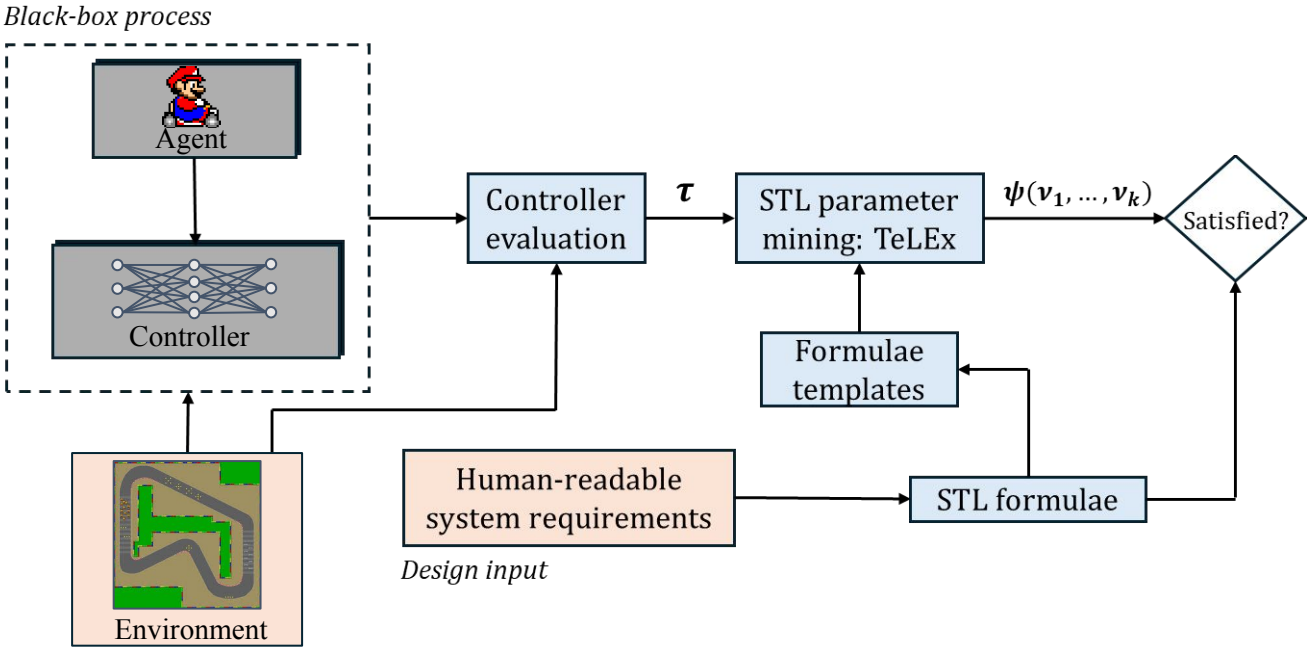


Our Approach to Specification-Driven Iterative Design



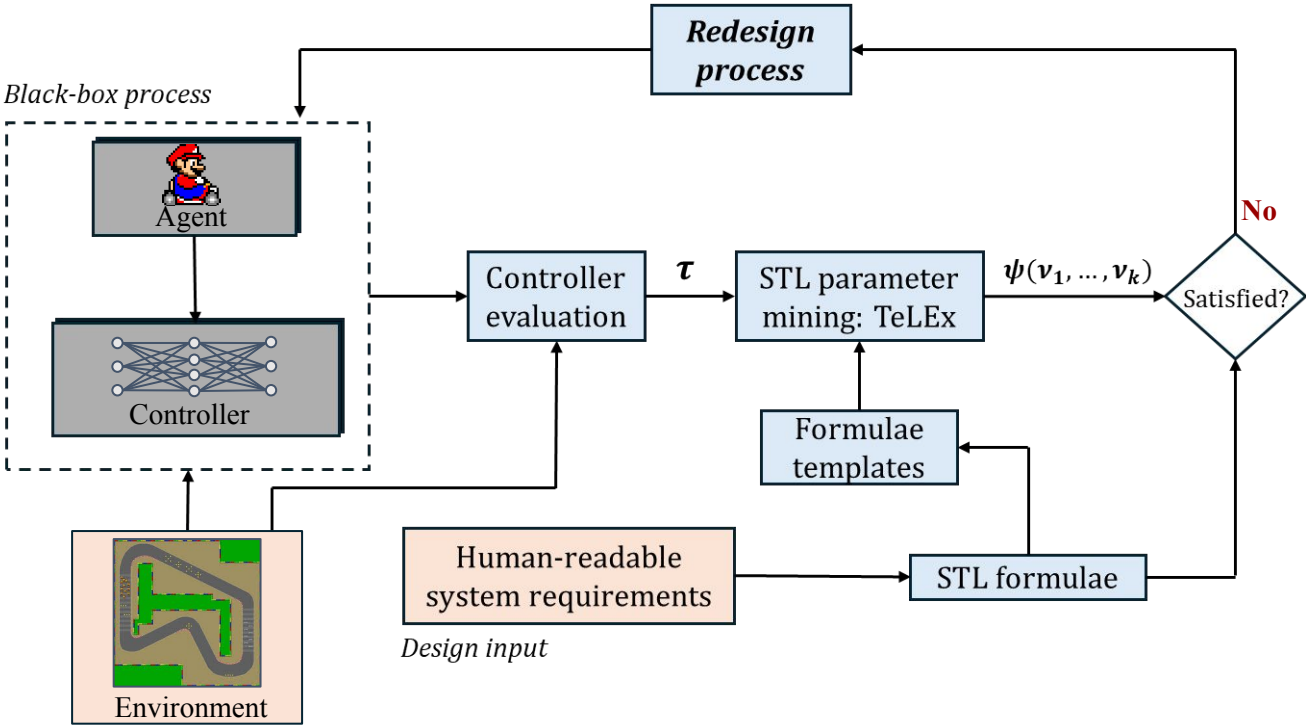
Telex: S. Jha et al. (2019)

Our Approach to Specification-Driven Iterative Design



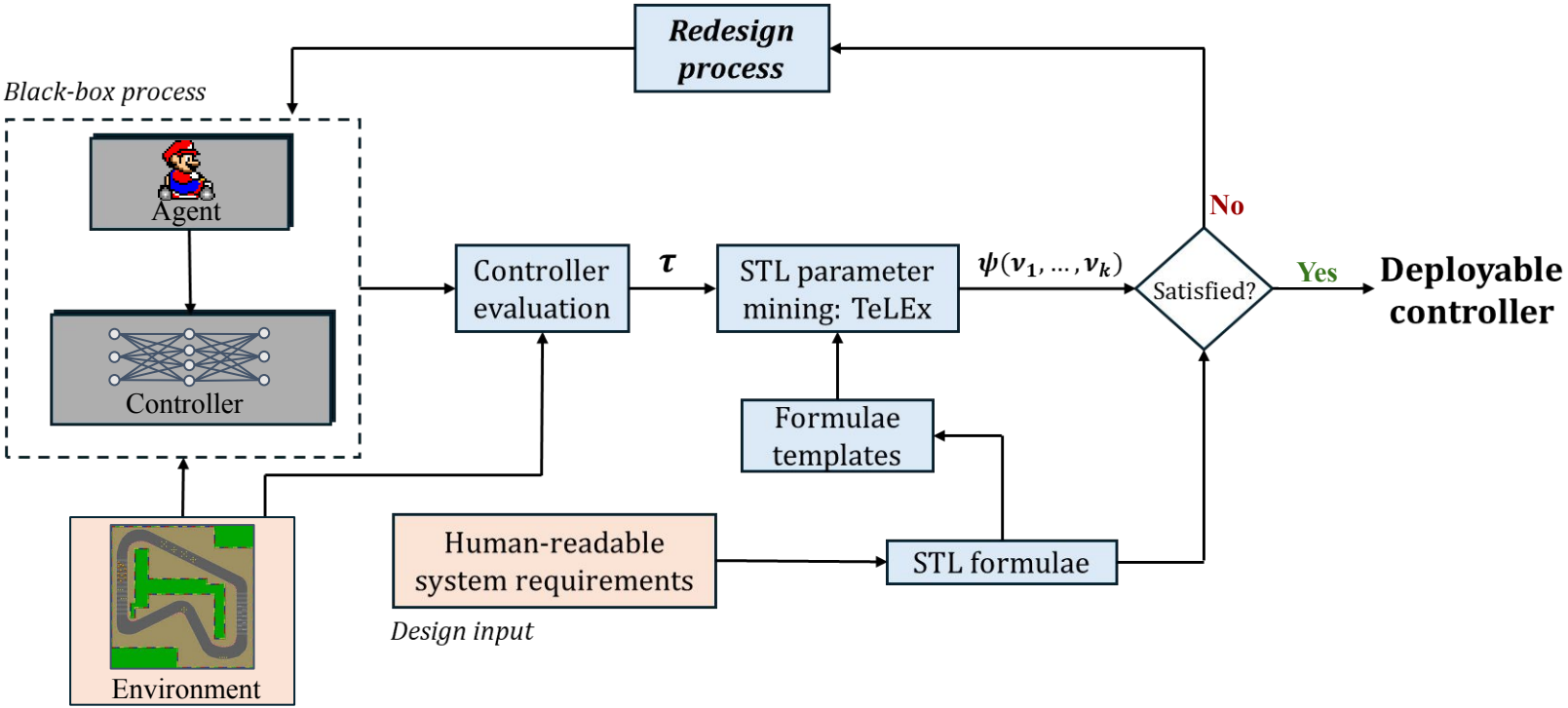
Telex: S. Jha et al. (2019)

Our Approach to Specification-Driven Iterative Design



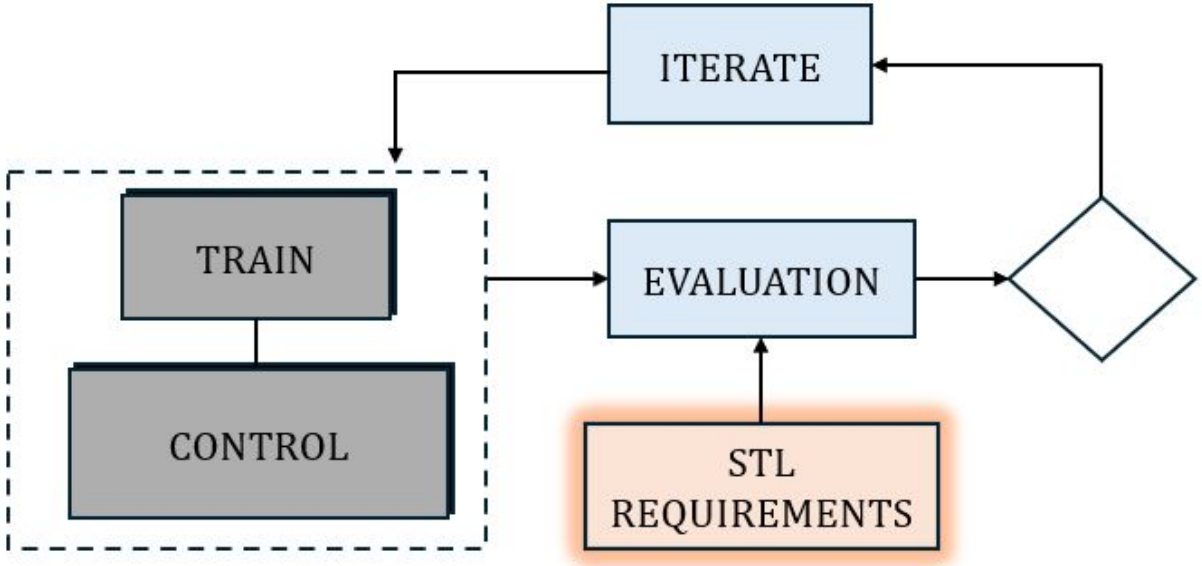
Telex: S. Jha et al. (2019)

Our Approach to Specification-Driven Iterative Design

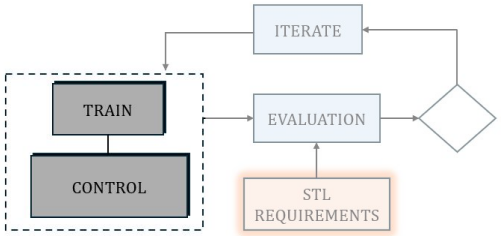


Telex: S. Jha et al. (2019)

Our Approach to Specification-Driven Iterative Design



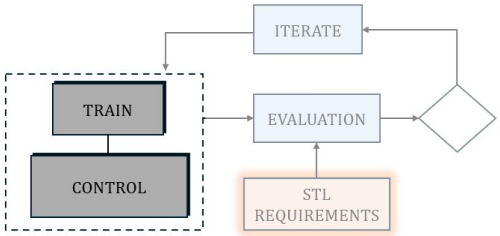
Training in Simulation

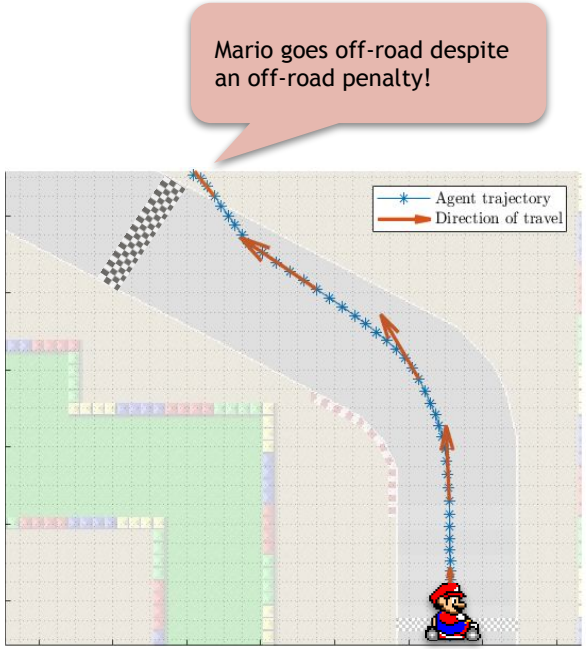


8 agents training simultaneously for 1,000,000 steps (roughly 3.5 hours)

Challenges With Our RL Training

Reward Term	Behavior encouraged
Checkpoint Reward	Forward progress around the track
Speed Reward	Fast driving
Road Reward	Staying on the track
Time Reward	Quick mission completion

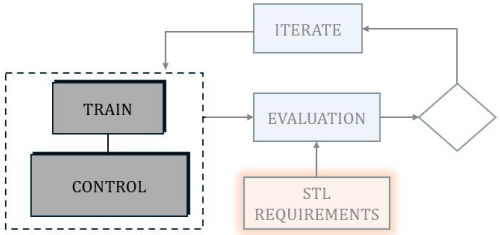


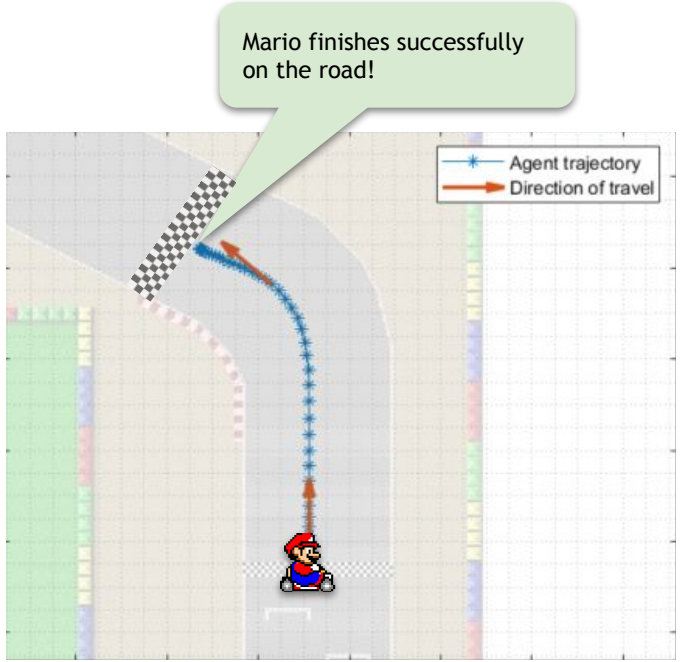


Evaluated trace of (bad) agent after training.

Agent Performance Evaluation

- Trained on the same mission and environment as evaluated
- Control is *stable* and *continuously advancing* towards the goal.



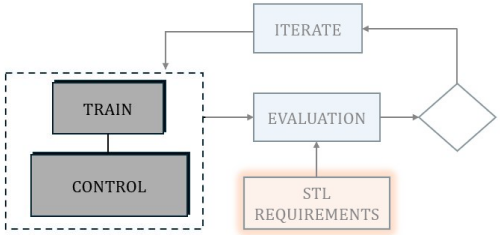


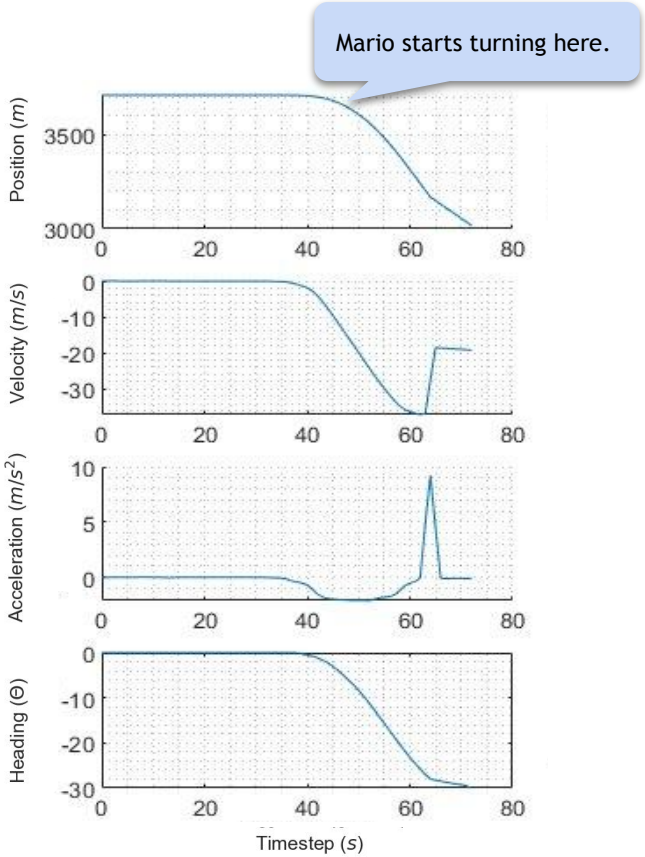
Evaluated trace of (good) agent after training.

Pre-Processing The Data

From raw position traces (x, y, t), we compute **velocity**, **acceleration**, and **heading.**

These **derived state quantities** are essential for controller logic & evaluating Signal Temporal Logic (STL) properties.





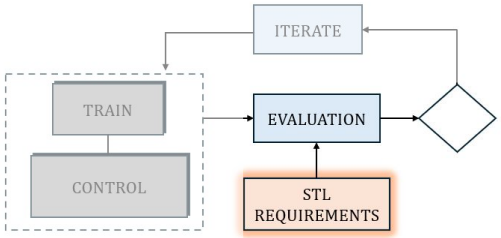
Mining Requirement-based Properties in TeLEx

"When the agent is engaged in a turn, the maximum speed should be kept low."

```
G[0,72] ( ((phidot > 0.5)|(phidot < -0.5)) → speed < a? 0;80)
  Synthesized STL Formula: G[0.0,72.0](((phidot > 0.5) | (phidot < -0.5)) → (  speed <
44.16)
  Theta Optimal Value: 0.027
  Optimization Time: 0.046

  Test result of synthesized STL on each trace: [True]
  Robustness Metric Value: [0.004]
```

Φ_1: what is the maximum speed when engaged in a turn?
→ **44.16 m/s**



TeLEx also provides robustness!

This tells us how well our specifications were satisfied.

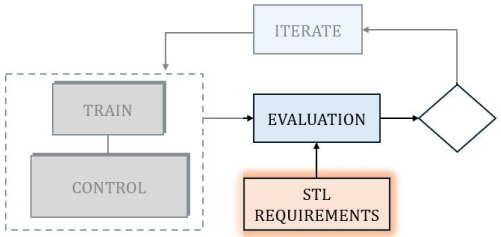
Mining Requirement-based Properties in TeLEx Part 2

```
F[0, a? 1;70](speed > 20)
 Synthesized STL formula: F[0.0,32.005](speed > 20.0)
 Theta Optimal Value: 0.2501
 Optimization time: 0.0022
```

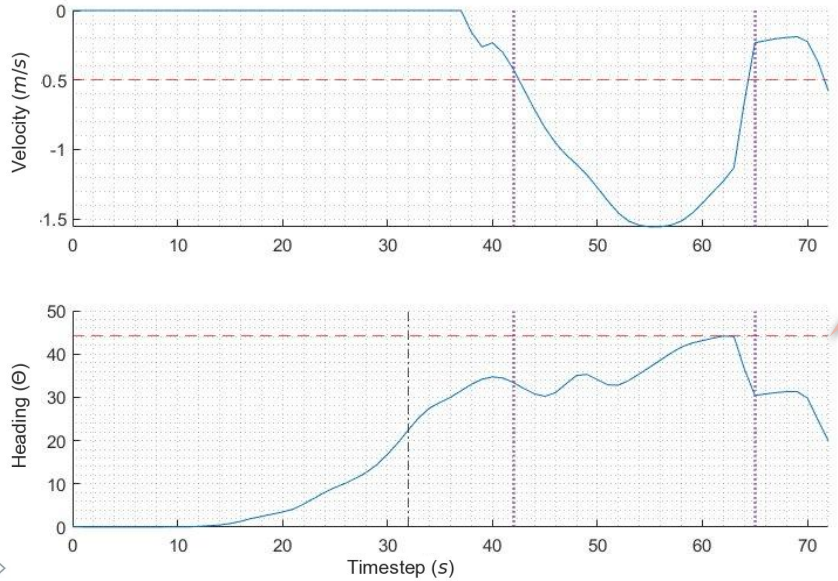
Φ_2: when does the vehicle reach necessary speed?
→ **at t = 32 sec**

```
G[a? 0;50, b? 50;72](phidot < -0.5)
 Synthesized STL formula: G[42.001, 64.999](phidot < -0.5)
 Theta Optimal Value: 0.3533
 Optimization time: 0.0200
```

Φ_3: how long is the vehicle engaged in the turn?
→ **23 sec**

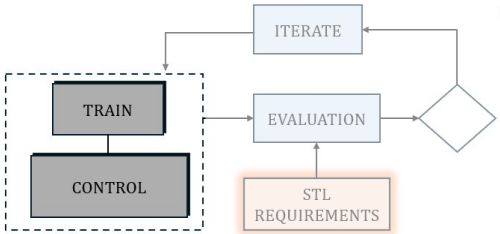


pSTL Mined Temporal Properties



Φ_1: what is the maximum speed when engaged in a turn?

Red line is the maximum speed when in a turn



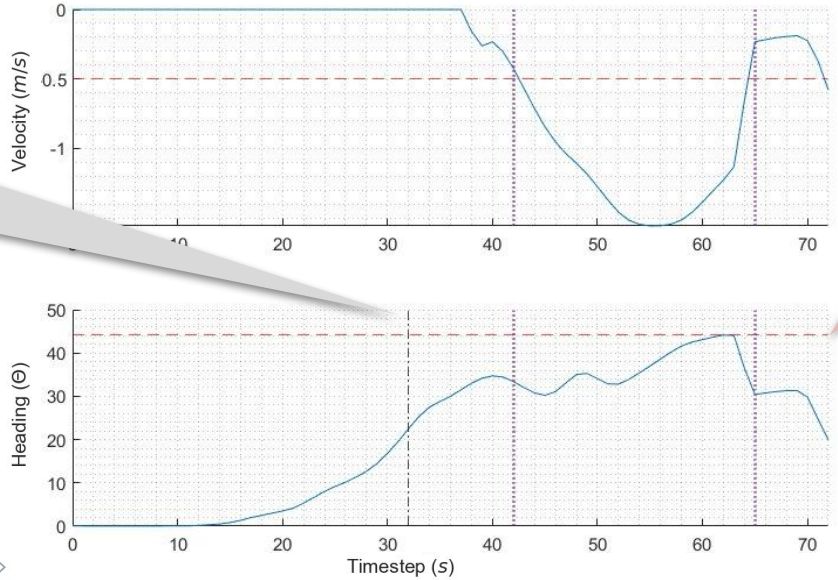
pSTL Mined Temporal Properties

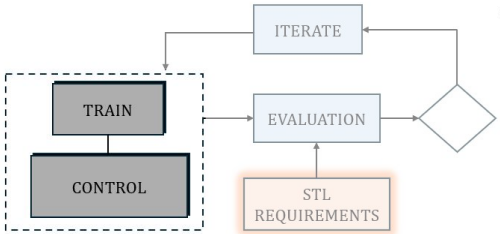
Φ_2: when does the vehicle reach necessary speed?

***Black** line identifies when the speed reaches 20 m/s*

Φ_1: what is the maximum speed when engaged in a turn?

***Red** line is the maximum speed when in a turn*





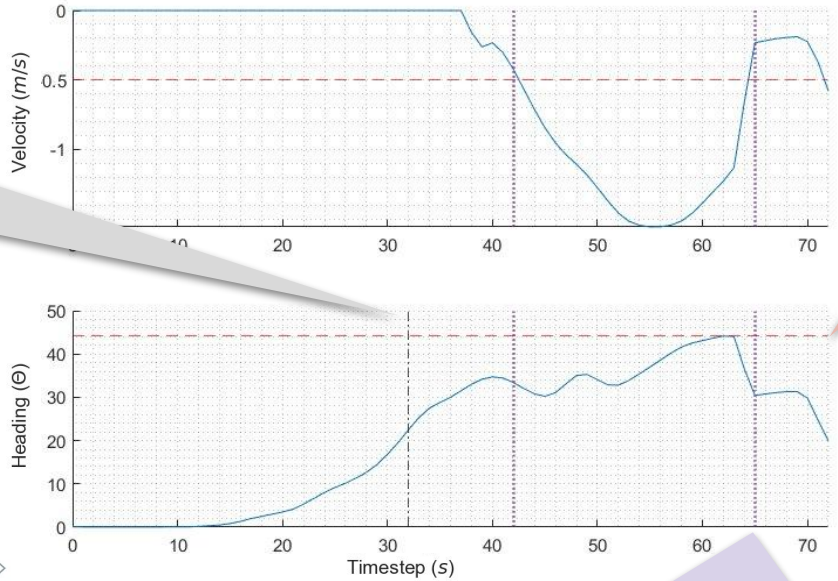
pSTL Mined Temporal Properties

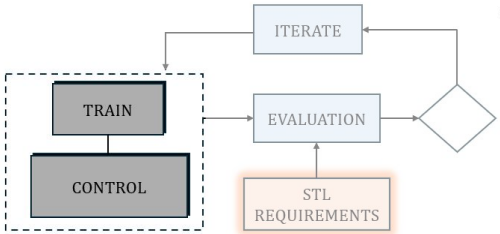
Φ_1: what is the maximum speed when engaged in a turn?

Red line is the maximum speed when in a turn

Φ_2: when does the vehicle reach necessary speed?

***Black** line identifies when the speed reaches 20 m/s*



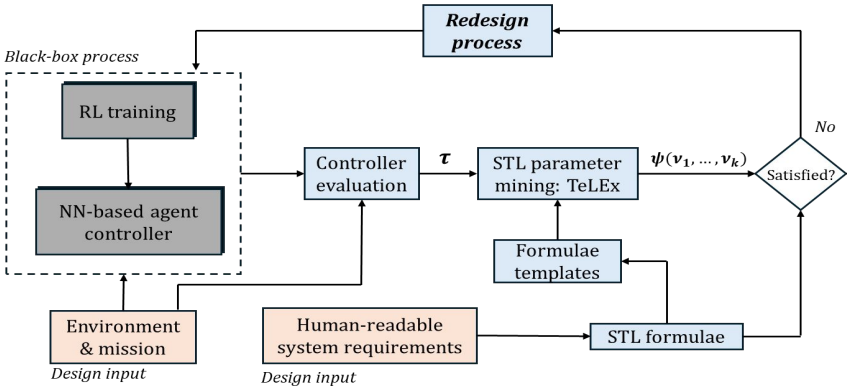


Φ_3: how long is the vehicle engaged in the turn?

***Purple** lines are the correctly identified turn*

Future Work: Closing The Loop

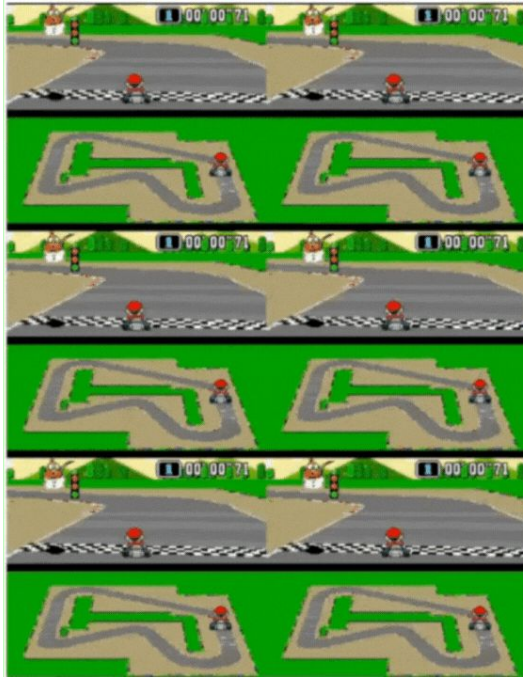
We showed you can train an ML-enabled controller and evaluate it's requirements with STL. *What's next?*



- How to retune the **reward structure**?
- What **ML system properties** must be true in the framework?
- How to reconcile **competing requirements**?

Key Takeaways

- We integrate **formal specifications** into our **iterative black-box design** process.
- We **executed 1 loop** of our iterative black-box design process.
- We will next **begin our re-design process**.



Citations

K. Sakano, J. Mockler, A. Chen, and H. Xu, “A Framework for Black-Box Controller Design to Automatically Satisfy Specifications using Signal Temporal Logic,” 2025 International Conference on Unmanned Systems, 2025, pp. 587-594.

A. Balakrishnan and J. V. Deshmukh, "Structured Reward Shaping using Signal Temporal Logic specifications," *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Macau, China, 2019, pp. 3481-3486, doi: 10.1109/IROS40897.2019.8968254.

D. Aksaray, A. Jones, Z. Kong, M. Schwager and C. Belta, "Q-Learning for robust satisfaction of signal temporal logic specifications," 2016 IEEE 55th Conference on Decision and Control (CDC), Las Vegas, NV, USA, 2016, pp. 6565-6570, doi: 10.1109/CDC.2016.7799279. keywords: {Robustness;Semantics;Trajectory;Learning (artificial intelligence);Markov processes;Standards},

M. Poliquin, “Stable retro, a maintained fork of openai’s gym-retro,”https://github.com/Farama-Foundation/stable-retro, 2025.

S. Jha, A. Tiwari, S. A. Seshia, T. Sahai, and N. Shankar, “Telex: learning signal temporal logic from positive examples using tightness metric,” Formal Methods in System Design, vol. 54, pp. 364–387, 2019.

Photo Credits

Waymo: https://waymo.com/blog/2019/01/automl-automating-design-of-machine

Indago 4: https://www.lockheedmartin.com/en-us/news/features/2021/5-small-unmanned-products-to-watch-this-year.html