Formal Methods in Robot Policy Learning: A Survey on Current Techniques and Future Directions

Anonymous authors
Paper under double-blind review

Abstract

As hardware and software systems have grown in complexity, Formal Methods (FMs) have been indispensable tools for (1) rigorously specifying acceptable behaviors, (2) synthesizing programs to meet these specifications, and (3) validating the correctness of existing programs. In the field of robotics, a similar trend of rising complexity has emerged, driven in large part by the adoption of deep learning. While this shift has enabled the development of highly performant robot policies, their implementation as deep Neural Networks (NNs) has posed challenges to traditional formal analysis, leading to models that are inflexible, fragile, and difficult to interpret. In response, the robotics community has introduced new formal and semi-formal methods to support the precise specification of complex objectives, guide the learning process to achieve them, and enable the verification of learned policies against them. In this survey, we provide a comprehensive overview of how FMs are integrated into robot policy learning. We organize our discussion around three key pillars: specification, synthesis, and verification of learned policies. For each, we highlight representative techniques, compare their scalability and expressiveness, and summarize how they contribute to meaningfully improving realistic robot safety and correctness. We conclude with a discussion of remaining obstacles for achieving that goal and promising directions for advancing FMs in robot learning.

1 Introduction

Both in natural and artificial settings, it has been repeatedly observed that surprising levels of complexity and capability can *emerge* from systems built by massively scaling and composing simple components or rules (Anderson, 1972). The "bitter lesson" learned by artificial intelligence researchers is one instance of this phenomenon: methods that scale effectively with data and compute tend to outperform those that rely on explicitly incorporating human knowledge (Sutton, 2019). The emergent complexity of such scalable systems is one of the most significant factors behind the success of Deep Learning (DL) (Kaplan et al., 2020). When combined with well-developed theoretical frameworks for general decision-making and control, DL techniques have scaled from achieving superhuman performance in game playing (Mnih et al., 2015), to enabling effective robotic manipulation (Levine et al., 2016; 2018), and now to powering increasingly general embodied agents and foundational models for robotics (Reed et al., 2022; Hu et al., 2024; Firoozi et al., 2024). With this flexibility, it is becoming viable to deploy robots alongside humans in unstructured environments and with open-ended goals—fulfilling roles such as household assistants (1X Technologies, 2025), autonomous vehicle controllers (Favaro et al., 2023), and medical assistants (Empleo et al., 2025).

However, these new use cases demand higher levels of confidence in deployed robot policies and, consequently, greater levels of policy interpretability, robustness to uncertainty, and compliance with behavioral and safety specifications. Alarmingly, achieving such confidence appears to be fundamentally constrained by the very increase in policy complexity that has enabled these systems' viability. Learned robot policies, typically parameterized by deep NNs, are well known for their lack of interpretability (Zhang et al., 2021), poor generalization to out-of-distribution settings (Fang et al., 2024), and vulnerability to adversarial inputs (Szegedy et al., 2014; Xiong & Jagannathan, 2024). These limitations are particularly concerning in safety-critical or high-stakes environments, where failures can lead to catastrophic outcomes, and conventional approaches to specifying robot behavior, such as reward functions or behavioral demonstrations, have so far failed to address these issues of policy trustworthiness. Due to their lack of guarantees, they can lead to unsafe, under-performing, or misaligned behaviors due to phenomena such as reward hacking (Skalse et al., 2022) or overfitting (Goodfellow et al., 2016). Even when safety is not the primary concern, these approaches also lack the expressive power to formally and concisely define complex, compositional, or temporally extended tasks.

In the world of software, similar safety concerns have emerged as increasingly complex programs were deployed in safety-critical settings (Baase, 2008), and these concerns have motivated the development of Formal Methods (FMs) to mitigate or eliminate these risks. Central to these and subsequent FM frameworks is the capability to rigorously **specify** system behavior, for which modal logics like Linear-time Temporal Logic (LTL) (Pnueli, 1977) have been used with great success. After formally specifying some behavior, FMs can allow a user to directly **synthesize** high-level, correct-by-construction programs from a Formal Specification (FS) (Church, 1963; Solar-Lezama, 2009; Srivastava et al., 2010). Closely related FMs, such as Model-Checking, provide ways to formally **verify** that complex programs satisfy their specifications (Baier et al., 2008; Kästner et al., 2018; Murray et al., 2013). Formal specification, synthesis, and verification now constitute a complete toolkit for developing highly dependable software systems.

In robotics, these three categories of FMs have likewise been applied and extended to ensure the correctness and safety of controllers *without* any learning components. For example, specification languages such as Signal Temporal Logic (STL) have been developed to express requirements over continuous-time for real-valued properties of dynamical systems (Maler & Nickovic, 2004), and probabilistic and hybrid-system model-checking techniques have been introduced as well (Alur et al., 1991). These advancements have enabled the synthesis and verification of robotic policies based on traditional planning algorithms (Kress-Gazit et al., 2009), as well as controllers derived from classical optimization (Sun et al., 2022a) and control theory (Abate et al., 2019). Such formal approaches offer guarantees not only for immediate control actions but also for long-term behavioral trajectories, directly addressing the predictability and robustness of robotic systems.

With the rise of *learning*-based methods in robotics, more recent research efforts have sought to use FMs to synthesize and verify learned policies and address the critical weaknesses of purely learning-based approaches mentioned above. These new FMs have the potential to systematically provide interpretability, behavioral guarantees, and safety assurances for neural-network-based robot policies. Moreover, FSs allow for richer and more concise behavior definitions than reward functions or demonstrations, easily capturing complex, compositional, and temporally extended requirements. This combination promises not only robust performance but also strict conformance to intended behaviors, substantially reducing susceptibility to value misalignment or adversarial exploitation. Consequently, FMs represent a promising approach for transforming learned robot policies from impressive but opaque models into transparent, reliable, and trustworthy solutions ready for real-world deployment. In this survey, we contribute a thorough overview of these recently developed methods.

Scope and Contribution This survey aims to provide for robotics researchers an accessible introduction to and comprehensive overview of recent work at the intersection of FMs and DL for robot control. Unlike prior surveys that focus on applying learning methods to FMs (e.g., Wang et al. (2020)), we instead focus on the application of FMs to learning-based methods. Specifically, this is a survey of FMs as applied to learning methods for robot policies, rather than to general Machine Learning (ML) systems (Larsen et al., 2022; Urban & Miné, 2021).

Several surveys have reviewed the use of FMs in robotics. For instance, Luckcuck et al. (2019) examine a range of FMs applications in autonomous robotic systems, with a focus on model checking for robotics software and specification languages. However, since that survey, DL-based systems have gained greater prominence in robotics and remain largely unexamined in that context. Similarly, Belta & Sadraddini (2019) survey formal synthesis methods for optimization-based robot controllers but does not address learned controllers. Some existing surveys have examined subfields of FMs relevant to ensuring the safety of learned controllers—such as Specification Mining (SM) (Bartocci et al., 2022), learning certificate functions (e.g., Lyapunov functions, barrier functions, and contraction metrics) (Dawson et al., 2023), set-propagation-based reachability analysis (Althoff et al., 2021), Hamilton-Jacobi (HJ) reachability analysis (Bansal et al., 2017),

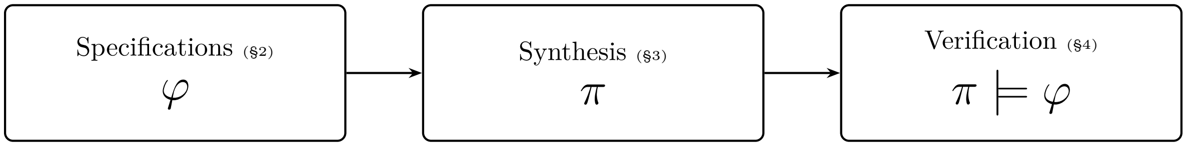


Figure 1: Our survey organizes work on FMs for robot policy learning in three categories: Work on learning-based-policy-amenable **specifications** for robot behavior, typically denoted by ϕ, FM-informed learning (**synthesis**) of robot policies, typically denoted by π, and **verification** that a learned robot policy satisfies a specification, typically denoted by $\pi \models \phi$.

cyber-physical system verification (Tran et al., 2022), and adjacent safe learning methods (Brunke et al., 2022). We also include the most relevant above subfields in this survey, provide an up-to-date overview of each, and refer readers to existing works for more in-depth details where appropriate. More importantly, however, we offer a comprehensive comparison across these methods and situate them within a unified organizational framework. To our knowledge, no prior survey has provided such an integrated perspective on how FMs have been developed and adapted specifically for DL-based robot policy learning. We expect this to be a valuable resource for guiding future research in this area.

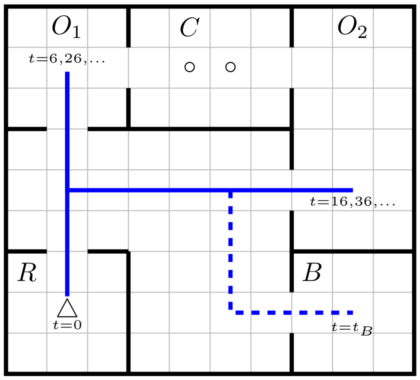
Outline We consider the use of FMs for robot policy development to be naturally summarized with the three stage pipeline shown in Figure 1. The pipeline begins with the acquisition of a formal behavioral specification, proceeds to the synthesis of a policy based on the specification, and concludes with the verification of the policy against that specification. We therefore have adopted the same structure to organize our survey. Section 2 surveys the first stage, which is further stratified into methods for acquiring specifications and representing specifications for downstream usage with NNs (e.g., as real-valued inputs, differentiable loss functions, etc.). Section 3 surveys policy synthesis methods and summarizes how policy learning techniques can be informed by behavioral specifications to enable more complex tasks, accelerate learning, or improve safety. Section 4 surveys methods focused on verifying that learned controllers conform to specifications. We clarify for each method the underlying verification approach, the specifications they support, and the assumptions they make on the system and controller. Finally, Section 5 discusses notable gaps in the existing work to clarify directions for future work, and Section 6 concludes the survey with a summary of the above content.

2 Formal Specifications for Learning-Based Policies

We begin our survey focusing on the foundational concept underlying all FMs: the Formal Specification (FS). No method for policy synthesis or verification can operate without a precise description of the behavioral requirements that a policy must fulfill. However, acquiring and representing such specifications is a significant challenge, especially for systems that are high dimensional, stochastic, or partially observable. To provide context, we first introduce the basic definitions and the most widely used representations of FSs as they appear throughout the literature on FMs. We then review key research on two core problems: how formal specifications are **obtained**, and how they are **represented** to support downstream synthesis and verification methods.

2.1 Preliminaries

Languages Classically, FMs aim to ensure that systems behave as desired (Baier et al., 2008), but behaving "as desired" is fraught with ambiguity. The first step to eliminating that ambiguity is considering the system under an appropriate formal model, and the standard choice is a discrete transition system. We provide an illustrative example of such a system and a FS of complex and temporally extended behavior for the system in Figure 2. In general, a discrete transition system is a tuple $(S, A, \longrightarrow, I, AP, L)$, where S is a set of states, A is a set of actions, $\longrightarrow \subseteq S \times A \times S$ is a transition relation (i.e., a set of possible transitions (s, a, s') between pairs of states $s, s' \in S$ given actions $a \in A$), $I \subseteq S$ is a set of initial states, AP is a set of atomic



(a) The office building grid-world environment with two office rooms (O_1 and O_2), a conference room (C), a storage room (R), and a break-room (B).

$$\underbrace{\Box\Diamond O_1 \wedge \Box\Diamond O_2}_{\text{Recurrence}} \wedge \underbrace{\Box C_{\text{busy}} \Rightarrow \neg C}_{\text{Safety}} \wedge \underbrace{\Diamond B}_{\text{Guarantee}}$$

(b) An LTL expression describing a language (i.e., a set of behaviors) that requires the two offices are visited infinitely often, the conference room is never visited when it is occupied, and the break-room is eventually visited at least once. According to the standard classification in Manna & Pnueli (1990), these are recurrence, safety, and guarantee sub-expressions.

$$\begin{array}{rccccccccccccc} t = & 0 & \dots & 6 & \dots & 16 & \dots & 26 & \dots & 36 & \cdots & t_B & \cdots \\ L(s_t) = & \begin{Bmatrix} R \\ C_{\text{busy}} \end{Bmatrix} & & \begin{Bmatrix} O_1 \\ C_{\text{busy}} \end{Bmatrix} & & \begin{Bmatrix} O_2 \\ C_{\text{busy}} \end{Bmatrix} & & \begin{Bmatrix} O_1 \\ C_{\text{busy}} \end{Bmatrix} & & \begin{Bmatrix} O_2 \\ C_{\text{busy}} \end{Bmatrix} & \cdots & \begin{Bmatrix} B \\ C_{\text{busy}} \end{Bmatrix} & \cdots \end{array}$$

(c) The labeling for each state in the example behavior, through which one can determine if the behavior satisfies the LTL expression.

Figure 2: A typical discrete domain modeling a single agent in a building with several rooms. The model is defined with atomic propositions corresponding to the agent's presence in each room and the occupancy status of the conference room ($AP = \{R, O_1, O_2, C, B, C_{\text{busy}}\}$ using the symbols in 2a). The example behavior in blue depicts the agent exiting the starting room, cycling infinitely between visiting the offices, and at some point visiting the break-room. An example LTL expression (2b) can be used to specify complex behaviors and is evaluated over sequences of labels (2c) obtained via the labeling function $L : S \to 2^{AP}$.

propositions, and $L : S \to 2^{AP}$ is a labeling function that maps each state to the set of true propositions for that state (Baier et al., 2008). Under this model, the behavior of a system can be formally defined as a (possibly infinite) sequence of states $(s_0, s_1, \dots) \in S^\infty$, where S^∞ denotes the set of all possible finite and infinite sequences over S. The possible sequences are further restricted to the sequences starting with states $s_0 \in I$, and those that can be induced by another sequence of actions $(a_0, a_1, \dots) \in A^\infty$, and the system's transition relation $\longrightarrow$.

Provided that that behaviors under this model (as well as others) are formally sequences of states, it is natural to use tools from formal languages to express specifications of desirable behavior. While this provides many formalisms with the expressiveness necessary to describe most behaviors relevant to robotics applications, we focus on two particularly useful and relevant formalisms: LTL (Pnueli, 1977) and Finite State Automata (FSAs).

Temporal Logic LTL is an extension of propositional logic. Logical expressions for any system, typically denoted by ϕ and belonging to a set of expressions Φ, are defined over the atomic propositions (AP) for that system. Atomic propositions can be thought of as the simplest boolean observations one can make from the state of the system. These can then be combined in more complex expressions using standard boolean operators such as "not" ($\neg$), "and" ($\wedge$), "or" ($\vee$), and "implies" ($\Rightarrow$). To describe behaviors over time, LTL introduces additional "temporal" operators referred to as "next", denoted by $\bigcirc$ or X, "eventually" (or "finally"), denoted by $\Diamond$ or F, "always" (or "globally"), denoted by $\Box$ or G, and "until" denoted by $\mathcal{U}$. The syntax of LTL is given by the following grammar:

$$\varphi ::= \texttt{true} \mid \texttt{false} \mid p \mid \neg\varphi \mid \varphi_1 \wedge \varphi_2 \mid \varphi_1 \vee \varphi_2 \mid \varphi_1 \Rightarrow \varphi_2 \mid \bigcirc\varphi \mid \Diamond\varphi \mid \Box\varphi \mid \varphi_1 \mathcal{U} \varphi_2.$$

A complete treatment of LTL is available in Baier et al. (2008), but we provide a brief summary of the temporal operator semantics here. The first three operators $\bigcirc$, $\Diamond$, and $\Box$ are unary, meaning they apply to only one sub-expression, and respectively express the requirement that the sub-expression is true at the immediate next moment in time, any moment in the future, or for all times in the future. The until operator

$\mathcal{U}$ is binary, meaning that it applies to two sub-expressions, and expresses the requirement that the first sub-expression is true at all future times until the second sub-expression is true at some future time. By composing and nesting these operators, one can quickly express highly complex and temporally extended behaviors with simple LTL specifications like the example in Figure 2.

Signal Temporal Logic (STL) (Maler & Nickovic, 2004) is a popular formalism that extends LTL in order to specify behaviors for real-valued signals defined over continuous time intervals. Using STL requires that each atomic proposition $p \in AP$ is associated with a real-valued function of the system state $\mu : S \to \mathbb{R}$, and the boolean value of the proposition in a state s is determined by the value of $\mu(s) \geq c$ for some threshold value $c \in \mathbb{R}$. Furthermore, the temporal operators are extended to consider the value of sub-expressions over a specific future time interval $[a, b]$ with $a \in \mathbb{R}, b \in \mathbb{R} \cup \infty$. The resulting grammar for STL expressions is

$$\psi ::= \ \mu(s) \geq c \mid \neg\psi \mid \psi_1 \wedge \psi_2 \mid \psi_1 \vee \psi_2 \mid \psi_1 \Rightarrow \varphi_2 \mid \Diamond_{[a,b]}\psi \mid \Box_{[a,b]}\psi \mid \psi_1 \mathcal{U}_{[a,b]}\psi_2,$$

with timing and threshold parameters a, b, c. Expressions in STL, in addition to supporting the true-or-false semantics of LTL expressions, support a quantitative semantics defined by a "robustness" function ρ that measures how much a sampled signal satisfies or violates the property specified by an STL formula $\psi \in \Psi$. We refer the reader to Maler & Nickovic (2004) and Dawson & Fan (2022) for the original definition and a more recent summary of STL semantics.

Automata A class of abstract machines termed FSAs provide a very useful alternative formalism for the same languages expressible in LTL. Within FSAs, there are automata that are meant to represent languages with finite elements (i.e., sets containing only finite behaviors) and ω-automata, which are capable of representing languages with infinite elements. More specifically, these are called regular and ω-regular languages. The former automata have a single, well-established representation, so we elaborate here on the more complex ω-automata.

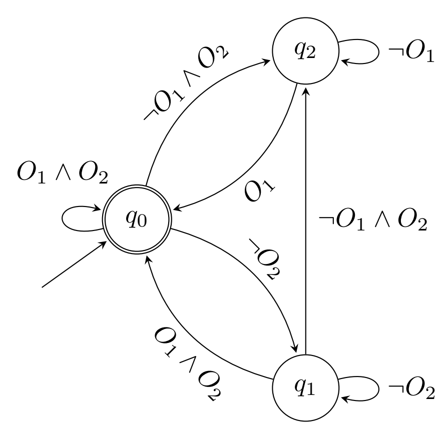


Figure 3: A Deterministic Büchi Automaton (DBA) for $\phi = \Box\Diamond O_1 \wedge \Box\Diamond O_2$. Intuitively, as the agent periodically encounters states s satisfying $O_1 \in L(s)$ and $O_2 \in L(s)$, the automaton moves from q_0, to q_1, to q_2, and back to q_0. As a result, the accepting state $q_0 \in F$ is visited infinitely often and the sequence is deemed to satisfy ϕ.

An ω-automaton is, like LTL, used to represent sets of sequences composed of elements in the set 2^{AP}. This is equivalent to representing sets of acceptable behaviors for a discrete transition system. An automaton does this by defining a procedure for reading one of these sequences element-by-element and subsequently determining whether the sequence is a member of the language. An example depicting the automaton for an LTL expression from the domain in Figure 2 is shown in Figure 3. The automaton is always in some state $q \in \{q_0, q_1, q_2\}$, and starts in the state q_0. As states are encountered in the system, the labeling function yields corresponding labels $L(s) \in 2^{AP}$ that then induce some change in the automaton's internal state as determined by the automaton's edges. The resulting sequence of internal automaton states then determines the membership of the input sequence in the set. Formally, an ω-automaton is defined by the tuple $A = (\mathcal{Q}, \Sigma, \delta, q_0, \alpha)$, consisting of a set of internal states $\mathcal{Q}$, an alphabet Σ, a transition relation $\delta : \mathcal{Q} \times \Sigma \to 2^{\mathcal{Q}}$, an initial state $q_0 \in \mathcal{Q}$, and some acceptance condition $\alpha \subset Q^\infty$. In our example and in the context of discrete transition systems, $\Sigma = 2^{AP}$. Each element $\sigma \in \Sigma$ of the input sequence causes some change in the automaton's internal state to a new state as permitted by the transition relation $q' \in \delta(q, \sigma)$. For a deterministic automaton, the transition function $\delta : Q \times \Sigma \to Q$ leads to a single automaton state $q' = \delta(q, \sigma)$ for all $q, \sigma \in Q \times \Sigma$. The resulting "run" of automaton states $q^\omega = q_0, q_1, q_2, \ldots$ is then said to be accepted if $q^\omega \in \alpha$. There are multiple different kinds of ω-Automata that differ based on the mechanism by which they define α (Hahn et al., 2022; Baier et al., 2008). For a brief overview, we note that the Rabin and Büchi acceptance conditions are prevalent.

A Deterministic Rabin Automaton (DRA) is a deterministic automaton with an acceptance condition defined by a set of pairs $\Omega = \{(B_i, G_i)\}$ where $B_i, G_i \subseteq Q$ are sets of states. The automaton accepts runs q^ω for which there exists a pair (B_i, G_i) in Ω such that q^ω visits B_i finitely many times and G_i infinitely many times. A DBA has a slightly simpler acceptance condition defined by a set of states $F \subseteq Q$, and accepts runs q^ω for which q^ω visits F infinitely many times. However, DBAs are unable to fully express the class of ω-regular languages, while Non-Deterministic Büchi Automata (NDBAs) are able to do so. We emphasize how these differences between automaton types have been relevant to research utilizing them in robot policy learning below in Subsection 2.2.2. We also note that in many cases, the methods we survey rely on standard algorithms for translating from LTL to automata (Gastin & Oddoux, 2001), so manual design of automata is almost never necessary.

Although there are many additional formalisms developed in FMs, we conclude our summary of background here as the majority of work seeking to bridge FMs with DL for robotic control has focused on the above fundamental specification representations. We refer the reader to Alur & Dill (1994) and Rheinboldt & Paz (2014) for examples of those other formalisms, and note that their continued exploration is a potential avenue for future work.

2.2 Current Techniques

We now survey the current research on FSs for robot policy behavior, which we first categorize according to the following two major research questions: (1) how can we **acquire** useful FSs despite the complexity of realistic robot systems, and (2) how should we **represent** FSs for our eventual goals of formal robot policy synthesis and verification? We further classify the works on FS representations as being relevant to policy-input or policy-feedback. A guide for these three categories and the following section is provided in Figure 4.

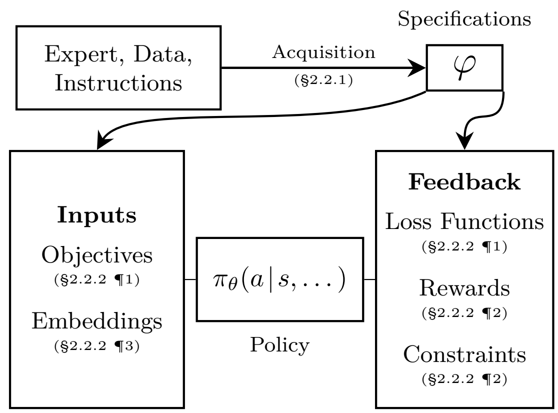


Figure 4: We organize the research on FSs for robot policy learning into two broad categories: methods for specification acquisition and methods for specification representation. Within the latter category, we further categorize works by their focus on representations for policy input or representations for policy feedback.

2.2.1 Obtaining Specifications

Specifications become harder to define as one loses the ability to effectively model the system, which is the case for the complex, real-time, continuous, stochastic, and partially-observable dynamical systems in robotics. The oldest strategy developed to address this difficulty is SM (Ammons et al., 2002). SM is the automatic generation of FSs from past experience or other informal descriptions of correctness. This problem was initially motivated by the difficulty of manually designing sufficiently comprehensive sets of program specifications and the fact that verification procedures are only as useful as the specifications against which they verify. Instead of manual design, a SM procedure could produce specifications given typical executions of a mostly-correct software system (Ammons et al., 2002), after which the system could be more rigorously verified in all execution contexts with respect to those specifications. This difficulty only worsens as systems become more complex, high-dimensional, continuous, real-time, stochastic, partially observable, and integrated with learned components (Seshia et al., 2018). Consequently, research into using SM in the context of robotics has attracted significant attention.

From Data with Templates Early work on SM focused on the more tractable problem of only estimating parameters within some parameterized FS provided by an expert. In many cases, specifications for continuous systems are expressed in STL, but selecting appropriate timings, thresholds, and other parameters can be difficult in practice. To address this, users can supply a partially specified STL formula with free parameters known as a “template” and apply SM techniques to infer valid parameter values from observed data.

For instance, consider a user interested in characterizing how long it takes a system to stabilize. They might start with the template $\psi = \Diamond_{[0,a]}\Box_{[0,\infty]}p_\theta(x)$, with $p_\theta(x)$ being a predicate corresponding to the system having stabilized parameterized by θ. This template formally specifies that at some point within the first a seconds, the system enters a stable state determined by p_θ and remains there indefinitely. SM can then be used to infer the range of values for a and θ such that the resulting specification is satisfied by a dataset of observed system behaviors.

Asarin et al. (2012) presented two of the earliest approaches for this problem. The first involves determining the boolean combination of linear inequalities that constitute the domain of valid parameter assignments and simplifying them to obtain several sets of satisfactory parameters (Lasaruk & Sturm, 2009). The second uses randomized search techniques to find parameter values that best match the available data by some metric. Many follow-up works in SMs have reused either of these fundamental techniques. For instance, if an accurate system model is available, one can go beyond fitting to a fixed set of example traces. In this case, a parameter estimation strategy can be interleaved with a specification falsification procedure (see Section 4.2.4), producing specifications that hold for rollouts across a range of initial conditions (Jin et al., 2015; Hoxha et al., 2018). However, a notably different strategy, and a particularly significant step for FMs in robot policy learning as a whole, came with the development of differentiable STL specifications in the Signal Temporal Logic Computation Graphs (STLCG) framework (Leung et al., 2019). STLCG enabled the computation of the gradient of a parameterized specification’s satisfaction with respect to its parameters, which allowed the application of gradient-based optimization techniques to SM. This framework has since enabled several advances in control and policy synthesis for learned robots, which are covered in the subsequent sections.

From Data without Templates More recent approaches in SM have moved beyond parameter estimation in specification templates and attempt to obtain the template as well. Techniques in this category typically first find a suitable parameterized STL template for the structure and subsequently estimate the parameters using one of the above techniques (Bartocci et al., 2022). Techniques for learning a full STL formula along with its structure include direct search over restricted subsets of parameterized STL (Kong et al., 2014; Jones et al., 2014), constructing the formula to fit discretizations of the example signals (Vaidyanathan et al., 2017), decision tree learning algorithms (Bombara & Belta, 2021), incrementally constructing, parameterizing, and filtering STL formulae guided by their robustness (Jha et al., 2019), alternating enumeration and parameter estimation of a subset of STL formulae (Mohammadinejad et al., 2020), and genetic algorithms (Nenzi et al., 2018; Aydin & Gol, 2020). Li et al. (2024) notably take an approach similar to STLCG by encoding an STL formula as an NN and performing SM by training the NN. The complexity of the specification is then reduced as coefficients in the network are driven to zero during training. Adjacent works also explore methods for generating specifications from natural language Liu et al. (2022), using Bayesian inference to sample specifications from distributions constructed with probabilistic programming languages Shah et al. (2018), and learning automata representing tasks (Topper et al., 2022; Dohmen et al., 2022; Gaon & Brafman, 2020; Rens et al., 2020; Xu et al., 2021) using L^* (Angluin, 1987) or Hidden Markov Models (Abate et al., 2023). For more details on these and other SM techniques specifically for STL specifications, we defer to the survey by Bartocci et al. (2022).

With Higher Levels of Abstraction An alternative approach to the challenge of obtaining specifications has been enabling fundamentally more expressive logical languages. When dealing with high-dimensional or partially observable systems, much that we want to specify can only be realistically defined in terms of high-level, geometrically-rich, or abstract concepts that are difficulty to represent formally. For instance, STL excels in tractable systems where predicates relevant to a user’s objective, such as reaching a goal, can simply be defined in terms of a basic function on the system state (e.g., the agent’s spatial coordinates). Defining such predicates becomes significantly harder when these properties of the system state are not directly measurable, or when concepts like “reaching a goal” can’t be simply computed even if they are.

Some research has addressed this challenge by developing languages to more easily express spatial qualities. Haghighi et al. (2015) relies on a quad-tree representation of the system state and then nests Tree Spatial Superposition Logic (TSSL) expressions (Gol et al., 2014), which can naturally express relationships between regions of the system state, within STL. Bortolussi & Nenzi (2014) similarly enable expression of relationships by adding a "somewhere" operator to STL that operates over a graph-based representation of the system state. Now, DL has yielded the most effective strategies for reliably perceiving concepts at high-levels of abstract, and several formalisms have been adapted to integrate with these perception systems. Timed Quality Temporal Logic (TQTL) (Dokhanchi et al., 2019) and its successor Spatio-Temporal Quality Logic (STQL) (Hekmatnejad, 2021; Balakrishnan et al., 2021) permit expressing properties over streams of data provided by an object-detection system. Most recently, Kapoor et al. (2025a) explores directly expressing properties using the latent space of a pretrained model, such that expressing requirements such as "eventually reach goal A" can be evaluated based on distance between latent representations of the current system state and the goal state. Although these methods compromise on the formal verifiability of such expressions, they gain significant expressiveness for use with synthesis methods.

2.2.2 Using Specifications

After obtaining a specification, a typical next step is using it to shape a robot policy such that the policy produces behavior satisfying the specification. While we thoroughly cover more details on the strategies using specifications in policy synthesis and verification in Sections 3 and 4, we introduce in this subsection the variety of fundamental innovations made just in representing the specification, which can be reused in many of those downstream methods.

As Loss Functions and Optimization Criteria Following the STLCG framework introduced in Leung et al. (2019), similar general-purpose frameworks (Leung et al., 2023; Kapoor et al., 2025b) have been developed for expressing differentiable STL specifications, where the gradient of the specification's satisfaction can be computed with respect to all STL parameters and inputs. Naturally, this has been extensively used in robotics applications involving gradient-based optimization, such as training policies with STL-informed loss functions (Liu et al., 2023; Xiong et al., 2024). In the case that the policy is itself performing an optimization procedure, STL specifications can be used online as objective functions (Raman et al., 2014; Meng & Fan, 2023), which has notably been used to compliment the iterative denoising process in diffusion models (Zhong et al., 2023b; Meng & Fan, 2024b).

As Reward Functions Another prominent usage of FSs, particularly in robot policy synthesis, has been in automata-theoretic Reinforcement Learning (RL) Hahn et al. (2019). In this subfield, FSs are expressed as automata which monitor trajectories of a Markov Decision Process (MDP) and dispense rewards to incentivize choices that lead to satisfying the automaton's acceptance condition. Works such as Camacho et al. (2019b) and Jothimurugan et al. (2019) propose methods for automatically deriving these automata from LTL and related temporal logics, and subsequently defining well-structured reward functions over these automata—either by performing value iteration over the automaton or by quantitatively evaluating the predicates labeling each edge. Reward Machines (Icarte et al., 2018; Camacho et al., 2019b; Toro Icarte et al., 2022) are a specific subclass of reward-dispensing FSAs. While they cannot express all ω-regular behaviors, their restricted structure has enabled the development of several specialized RL algorithms that more effectively learn policies for finite-length behaviors.

For ω-regular behaviors, the earliest approaches used DRAs, due to their ability to express all ω-regular languages. However, subsequent results found that these early approaches using DRAs could fail to obtain optimal policies for certain objectives (Hahn et al., 2019), and it was later shown that this limitation of DRAs was unavoidable (Hahn et al., 2022). Subsequent algorithms (Hasanbeig et al., 2019a;b) used the simpler Büchi acceptance condition but, to avoid the restrictions of DBAs, used Limit-Deterministic Büchi Automata (LDBAs) (Sickert et al., 2016). These automata can still express all ω-regular languages, but are restricted so that any non-determinism can be resolved based on the past sequence of automaton states, which is necessary in a sequential decision making context (Vardi, 1985). These automata that are suitable for RL have since been formally classified as "Good-for-MDPs" automata in Hahn et al. (2020), and provide a very general formalism for learning to satisfy arbitrary formally specified, temporally extended objectives.

As Embeddings Finally, we cover the research dedicated to produce generic embeddings of FS to provide them as input to learned policies. Several significant works in this area take the approach of encoding an STL specification as a latent vector of real numbers. Hashimoto et al. (2022) uses a Word2Vec-style skip-gram model (Mikolov et al., 2013) to train ana NN to encode similar STL specifications to similar latent vectors. Saveri et al. (2024) encodes STL specifications based on the principal components of a Gram matrix with elements defined by the robustness of a set of candidate behaviors with respect to a set of candidate specifications. Other approaches, building on uses of Graph Neural Networks (GNNs) for processing symbolic representations of logical languages (Lamb et al., 2021; Crouse et al., 2020), first translate specifications into their automata representation and then use a graph neural network to produce a finite-embedding summarizing the automaton (Xie et al., 2021; Yalcinkaya et al., 2025). Again, such methods sacrifice the formal soundness of their input FSs in order to gain from the versatility of conditional generative models.

3 Formal Methods in Robot Policy Synthesis

In this section, we survey the strategies developed for synthesizing policies, or in other words, learning policies, in order to satisfy FSs. Applying FMs to policy learning holds the promise of enabling policies that are not only more trustworthy and reliable, but also capable of learning and executing the rich, non-Markovian behaviors commonly expressed in typical FSs. This overall objective contains several constituent challenges that we describe, along with their associated bodies of research, in the following subsections. For background, we first provide a short summary of relevant techniques developed for classic formal synthesis methods, the existing applications of FMs in robot planning and control, and the fundamentals of robot policy learning. We then systematically cover the work in formal synthesis for robot policies.

3.1 Preliminaries

Program Synthesis Although the goal of automatically generating programs has a long history dating back to the early days of computer science—ranging from the development of high-level language compilers to today's LLM-based coding assistants—there exists a distinct class of "formal synthesis" methods. These seek to generate programs to provably obey FSs that jointly define syntactic structure and semantic correctness requirements for the program (Solar-Lezama, 2023). The roots of formal synthesis trace back to work by Alonzo Church on generating digital circuits from mathematical specifications of behavior Church (1963), and to classic approaches that synthesized functional programs using interactive theorem provers via the Curry–Howard correspondence Manna & Waldinger (1980). The field has since evolved to incorporate efficient search-based strategies for navigating program spaces defined by syntactic constraints (Solar-Lezama, 2009; Alur et al., 2013; 2018), often with guidance from verification tools (Srivastava et al., 2010) or input-output examples (Feser et al., 2015). Several paradigms developed in formal program synthesis—such as Counter-example Guided Inductive Synthesis (CEGIS) (Solar-Lezama et al., 2006; 2008)—have since been adapted to the synthesis of robotic controllers and learning-based policies. In CEGIS, for example, the synthesis process is iteratively refined through counterexamples produced by formal verification procedures, ensuring that the candidate program converges to one satisfying the full specification.

Reactive Synthesis Reactive synthesis methods developed as program synthesis was applied to concurrent programs, for which an appropriate model is a "reactive system". Unlike typical programs that produce a single output and terminate, reactive systems continuously receive inputs from their environment and produce corresponding outputs (Pnueli & Rosner, 1988). As a result, they naturally model non-terminating systems such as controllers, protocols, and embedded systems that must maintain correct behavior indefinitely. The reactive synthesis problem is often framed as a two-player game between the system and its environment, with the objective of synthesizing the strategy that ensures the specification is satisfied regardless of how the environment behaves. Foundational work by Pnueli & Rosner (1988) introduced this game-theoretic view of synthesis for LTL specifications, and subsequent efforts have focused on improving scalability, synthesizing systems from useful subsets of LTL (Bloem et al., 2012), and producing implementations in hardware or software (Ehlers, 2010; Meyer et al., 2018).

Formal Methods in Motion Planning and Optimal Control The aforementioned reactive synthesis techniques have played a foundational role in many FMs applied to robotics. Early work by Fainekos et al. (2005) began to introduce using FMs for robotics problems by exploiting model checking techniques (refer to Section 4) to support LTL as a specification language for motion planning. This line of research into motion planning with FSs has since extended to multi-agent systems and more expressive specification languages (Sun et al., 2022a; Verhagen et al., 2024). Kress-Gazit et al. (2009) subsequently utilized reactive synthesis methods to obtain automaton-based motion planning policies that guarantee satisfaction of the input task. This approach has since been generalized for arbitrary MDP optimal control under temporal logic constraints (Wongpiromsarn et al., 2010; Smith et al., 2011; Ding et al., 2011; Chu (Dennis) Ding et al., 2011). However, these methods rely on the assumption that the underlying system is discrete or can be faithfully modeled with a discrete abstraction. As a result, this approach to policy synthesis has been termed "abstraction-based" (Belta & Sadraddini, 2019).

An alternative group of methods circumvent this reliance on a discrete abstraction by directly formulating and solving optimal control problems derived from FS. Within this category, popular approaches include encoding temporal logic as sets of constraints in Model Predictive Control (MPC) schemes (Raman et al., 2014) or offline trajectory optimizations (Yang et al., 2020) and solving mixed continuous-discrete Hamilton-Jacobi-Bellman (HJB) equations via approximate dynamic programming (Horowitz et al., 2014; Papusha et al., 2016). These methods are increasingly capable, but still face computational challenges in real-time settings and often assume full state observability or perfect models—limitations which motivate integration with learning-based methods.

Robot Policy Learning The introduction of DL to robotics has made it possible to practically utilize very general frameworks, such as RL (Sutton, 2018) and Imitation Learning (IL) (Hussein et al., 2017), for solving robotic optimal control and planning problems in the real world. RL typically solve problems modeled in terms of MDPs, which are defined by a tuple $(\mathcal{S}, \mathcal{A}, \mathcal{T}, d_0, r, \gamma)$. This tuple consists of a state space $\mathcal{S}$, an action space $\mathcal{A}$, a transition function $\mathcal{T} : \mathcal{S} \times \mathcal{A} \to P(\mathcal{S})$ (where $P(\mathcal{S})$ represents the space of probability measures over $\mathcal{S}$), an initial state distribution $d_0 \in P(\mathcal{S})$, a reward function $r : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$, and a discount factor γ. Given an MDP, the objective is to learn a stationary policy $\pi : \mathcal{S} \to P(\mathcal{A})$ that maximizes the expected discounted return $\mathbb{E}_{\tau \sim \pi} \left[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t)\right]$ over trajectories $\tau = (s_0, a_0, s_1, a_1, \dots)$ generated by interacting with the environment. A variety of RL algorithms can be used to solve this objective, commonly categorized by their use of environment models (model-free vs. model-based), their data regime (on-policy vs. off-policy), and their policy representation (value-based vs. policy-based). Most of these approaches rely on online interaction to collect experience.

Alternatively, IL encompasses policy learning frameworks that leverage a dataset of expert demonstrations and do not necessarily require extensive online interaction. One prominent subclass is Inverse Reinforcement Learning (IRL), which aims to recover a reward function that explains the expert's behavior, allowing subsequent policy optimization via standard RL techniques (Abbeel & Ng, 2004). Another is Behavior Cloning, which frames learning as a supervised learning problem, directly training a policy to imitate expert actions—for example, by minimizing a loss function based on the negative log-likelihood of the expert action under the learned policy. More recently, offline RL methods have emerged, which seek to learn a reward-maximizing policy from a fixed dataset of state, action, and reward tuples (Levine et al., 2020). Unlike behavior cloning, which merely mimics the expert, offline RL seeks to optimize behavior with respect to the observed rewards, even when they are suboptimal or sparse.

Despite significant advances, all of these policy learning methods are still generally considered unreliable in safety-critical settings (Skalse et al., 2022; Goodfellow et al., 2016), and this has motivated research into using FMs to overcome the limitations of learning from manually designed rewards or expert demonstrations alone.

3.2 Current Techniques

We now survey the current research integrating FMs into policy learning, grouped by the major policy learning paradigm they belong to. This overall categorization and the constituent subcategories are depicted in Figure 5. We note that each category primarily builds upon a different body of research discussed in

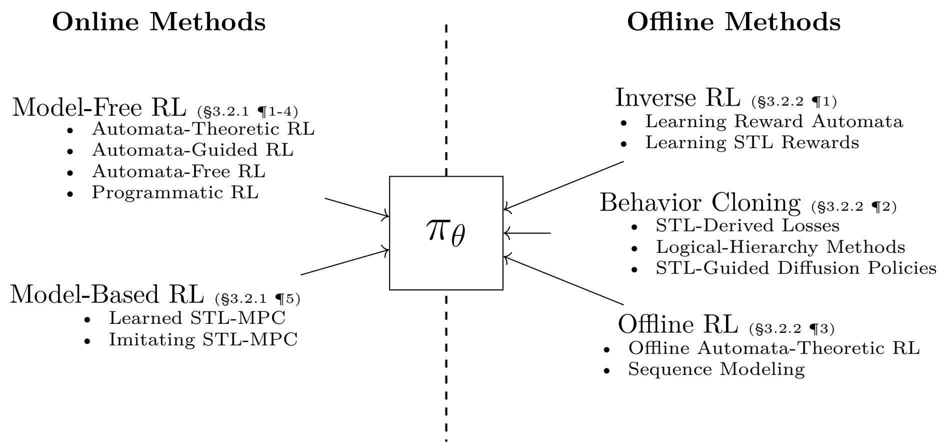


Figure 5: The research in formal synthesis for robot policies can be organized into categories corresponding to five major robot policy learning frameworks. The first two, model-free and model-based RL methods, are online methods that rely on environment interaction during learning. Within model-free RL, there are algorithms that learn from automata-theoretic rewards, algorithms that use automata for guidance without explicit rewards, algorithms that do not rely on automata at all, and algorithms based on programmatic policies. Model-based RL primarily extends optimal control techniques, such as MPC, with DL components to enhance performance or versatility. The remaining three categories, IRL, Behavior Cloning, and offline RL, are primarily offline methods that learn from pre-collected demonstration datasets. Their constituent subfields include learning FSs as reward functions for IRL, augmenting behavior cloning losses with FSs, and training sequence models to generate actions conditioned on high FS satisfaction.

the preliminaries and has different reasons motivating the introduction of either DL or FMs depending on their background. Online methods in particular introduce DL to generalize the aforementioned classical approaches in Section 3.1 to settings they are otherwise inapplicable. These include settings with unknown and complex system models, partial observability, and stochasticity, settings where standard optimization approaches are too slow, or settings where complex decisions must be made that consider longer, coarser horizons or partially satisfiable specifications. Offline methods, however, typically come from the robotics or DL for control communities and attempt to introduce FMs into existing-learning based methods to support more complex tasks and improve their safety and reliability.

3.2.1 Online Policy Learning

Automata-Theoretic Reinforcement Learning The first works to enable DL to produce policies satisfying high-level FSs established a connection between automata-theoretic verification techniques for probabilistic systems—developed in Courcoubetis & Yannakakis (1995); Vardi (1985)—and model-free RL. Sadigh et al. (2014) initiated this line of research by reusing the DRAs commonly employed by those verification techniques. This work trained the policy to act in a product MDP, so named because the state space of this new MDP was formed with the cartesian product of the plain MDP state space and the automaton state space. This automaton state augmentation can be seen as providing a minimal "memory" necessary for obtaining Markovian rewards and policies for otherwise non-Markovian objectives. The Rabin acceptance condition for a given task automaton could therefore be translated into a valid reward function whose maximization would imply maximizing the probability of satisfying the LTL formula. Although this performed well in practice, subsequent work highlighted their limitations. Specifically, Hahn et al. (2019) demonstrated that for certain MDPs and ω-regular objectives, it is not always possible to define a reward function based on the Rabin acceptance condition that yields an optimal strategy. This impossibility was later established in general by Hahn et al. (2022).

In place of DRAs, DBAs emerged as an attractive alternative due to their simpler and more intuitive acceptance condition. However, simple DBAs are not ideal due to lacking support for some ω-regular objectives. The immediate alternative of using NDBAs, with completely unrestricted non-determinism, is also infeasible because resolving non-deterministic choices for the automaton may require an unbounded look ahead into the future (Vardi, 1985), which is impossible in a sequential-decision-making context. The solution is to rely on a special class of automata with restricted non-determinism, in which every non-deterministic choice can be resolved based solely on the history of the execution. Hasanbeig et al. (2019a;a) and Hahn et al. (2019) were the first to use one of these automata by defining a reward from a LDBA (Sickert et al., 2016). As discussed in Section 2.2.2, this class of automata used in model-free RL has been further studied and formalized as "Good-for-MDPs" automata in Hahn et al. (2020). Independently, automata-theoretic methods for finite regular languages using plain FSAs were developed under the name Reward Machines, and a variety of reward shaping and accelerated RL algorithms were developed to use their particular formulation (Camacho et al., 2019a; Icarte et al., 2018; Toro Icarte et al., 2022). Reward-shaping techniques originally developed for Reward Machines in Camacho et al. (2019a) have also been extended to ω-automata in Bagatella et al. (2024).

Later work in this direction has relaxed assumptions about system observability implicit in the above methods. Hasanbeig et al. (2019b) notably relax the assumption that one has access to the exact atomic proposition labeling function L, and instead perform learning inside a probabilistically-labeled MDP. On top of this relaxation, Cai et al. (2021b;a; 2024) relax the assumption that the input FS is always satisfiable and ensure the resulting policy for unsatisfiable specifications are reasonable.

Automata-Guided Reinforcement Learning Instead of dispensing rewards, automata can also serve as high-level structures for guiding the execution of multiple low-level or goal-conditioned policies. For example, Jothimurugan et al. (2021) and Wang & Zhu (2025) (notably supporting ω-regular properties) both adopt an approach in which a separate policy for each edge of the automaton is first trained via RL. These edge-specific policies are then composed at execution time using a graph search procedure, such as Dijkstra's algorithm, to identify an optimal path through the automaton. Other approaches, such as those of Nangue Tasse et al. (2022) and Araki et al. (2021), similarly plan over automata to compose lower-level policies but instead rely on value iteration to compute an optimal composition.

One can trade some generality for computational efficiency by training a single goal-conditioned policy to replace the full library of edge-specific or task-specific policies used in the above methods (Qiu et al., 2023). Extending this idea even further, recent work has proposed to directly condition policies on embeddings (Section 2.2.2) created from automata (Yalcinkaya et al., 2025) and LTL formulae (Kuo et al., 2020; Vaezipoor et al., 2021), enabling trained agents to generalize to entire new tasks rather than singular goals. However, the diversity of these tasks is somewhat limited by assuming atomic propositions here are not arbitrary conditions and exclusively refer to reaching goals in the environment

Automata-Free Reinforcement Learning There are also RL methods that bypass the use of automata for rewards or guidance and instead rely directly on other FS formats. Several of these are concerned with continuous-time requirements written in STL, which cannot easily use automaton-based approaches. Aksaray et al. (2016) focus on STL formulae that can be evaluated over a fixed, finite horizon length n and define an MDP that augments the state space to record at all times the past n states, from which a Markovian reward function can be defined. Li et al. (2017) propose defining a non-Markovian reward based on STL robustness over entire episodes and searching for the optimal policy based on these complete policy rollouts. Other methods use similar reasoning to provide reward shaping based on an STL formula evaluated over a fixed-length history (Balakrishnan & Deshmukh, 2019). More recently, works have explored directly using the gradients provided by differentiable STL specifications to train a policy online (Xiong et al., 2024; Eappen et al., 2024), although this only supports STL specifications that express requirements of the direct policy outputs (actions) as opposed to the system state.

Programmatic RL For more formal policy synthesis, there is also growing interest in RL algorithms that use *programmatic* policy representations, which more closely align with traditional program synthesis techniques. Early approaches in this direction focused on approximating RL-trained NN-based oracle policies with more interpretable, program-like representations that offer benefits such as verifiable correctness (Bastani et al., 2019) and improved generalization (Inala et al., 2020). Zhu et al. (2019) in particular took advantage of this verifiability and adopted a CEGIS-style approach, first approximating a learned policy with a symbolic program and iteratively refining the program when counterexamples to correctness are found during verification. Subsequent work sought to bypass the need for an oracle policy altogether by developing program representations that could be trained end-to-end with policy gradient methods (Qiu & Zhu, 2022). More recently, Cui et al. (2024) address the limitations of these approaches in long-horizon, sparse-reward settings by using programmatic policies not just for execution, but also to aid in exploration and guide a tree-search over possible programs with encountered rewards.

Model-Based Reinforcement Learning Finally, research has also explored FMs for learning FS-satisfying policies with model-based RL. Although there are some automata-theoretic approaches in this community (Fu & Topcu, 2014), there are more that draw upon the latter category of optimization-based optimal control methods discussed in Paragraph 3.1. They then introduce learning techniques to remove dependence on knowledge of system dynamics, improve performance, or mimic decisions of experts in situations when specifications can only be partially satisfied. For instance, Cho & Oh (2018) formulate an MPC problem with constraints derived from a sequence of prioritized STL formulae, but learns from a dataset of expert behavior the degree to which those constraints should be satisfied. Meng & Fan (2023) bypass the typical sampling-based or gradient-based optimization used in MPC by directly learning a model that predicts optimal action trajectories satisfying STL specifications, enabling real-time performance.

Several methods simultaneously learn a model of the system dynamics along with the control policy. For example, Kapoor et al. (2020) learns a deterministic predictive model to enable MPC, using an objective derived from a given STL formula. Similarly, Liu et al. (2023) learns a simple deterministic model but additionally address the history dependence inherent in LTL-based objectives by implementing the policy as a Recurrent Neural Network (RNN). Learning the dynamics model can also facilitate formal guarantees for the resulting policy. For instance, Cohen & Belta (2021) propose to decompose a high-level temporal logic requirement into a sequence of optimal control problems, each solved individually using model-based RL. The overall policy then selects among subproblem solutions based on the current state of the task automaton. Moreover, the learned models and associated value functions support the construction of barrier functions (See Section 4.2.3) that formally certify the safety of the synthesized controller.

3.2.2 Offline Policy Learning

Incorporating logical specifications into offline policy learning is the most recent and shallowest development of the work we survey. RL methods benefit from the fact that the logical specification can be turned into a form of online feedback in a variety of ways, but using a specification in conjunction with a set of demonstrations is more complex. However, these approaches are not burdened by the issues of sample-inefficiency found with online methods and consequently more easily scale to realistic robot systems. Furthermore, the incorporation of highly informative, compact FSs holds the promise of dramatically reducing their initial data requirements.

Inverse Reinforcement Learning Early methods in offline policy learning extended the product MDP construction developed for online settings and applied IRL techniques to learn reward functions defined over the product MDP state space (Wen et al., 2017; Zhou & Li, 2018). Several of these approaches framed the problem as an instance of CEGIS, introducing the idea of generating adversarial counterexamples that highlight policy violations of ϕ in unrepresented regions of the training data, thereby improving robustness during learning (Zhou & Li, 2018; Puranic et al., 2021; Ghosh et al., 2021; Dang et al., 2023). More recent work has enhanced these IRL-based approaches by incorporating the STL inference techniques described in Section 2.2.1, to express the learned reward function in a more structured and interpretable representation (Liu et al., 2025).

Behavior Cloning Temporal logic specifications have also been leveraged in behavior cloning as additional differentiable terms in loss functions for training policies. Innes & Ramamoorthy (2020) used this approach to train a model for a dynamic motion primitive parameterized policy. More recent approaches blend offline and online learning: an offline phase first trains a policy to mimic overall movement patterns, followed by an online phase that refines the policy by learning boundaries between discrete logical modes (Wang et al., 2022; 2024). Other behavioral cloning methods employ generative models such as diffusion models (Meng & Fan, 2024a; Zhong et al., 2023b;a; Feng et al., 2024a) and flow matching techniques (Meng & Fan, 2025) whose inference process can be directly guided by robustness gradients from frameworks such as STLCG (Leung et al., 2023).

Offline Reinforcement Learning The most recent advances have focused on fully offline RL techniques (Levine et al., 2020). Due to the complexity of defining reward functions for LTL specifications, relatively few methods apply offline RL to product MDPs with the kinds of reward structures discussed in Section 3.2.1. One notable exception is Feng et al. (2024b), which appears to be the first to adopt this approach by learning from a hierarchical dataset and training a state-option value function where rewards reflect each action's alignment with an automaton derived from the input specification. Guo et al. (2024) takes a different direction by proposing an STL-conditioned decision transformer that conditions on the trajectory's robustness at each step. Overall, this direction of offline policy learning remains underexplored and presents significant opportunities to build upon the insights developed in the online setting.

4 Formal Methods in Robot Policy Verification

We conclude our survey by examining research aimed at the formal verification of learned robot policies with respect to given FSs. Compared to the synthesis methods discussed previously, these approaches place significantly greater emphasis on establishing formal guarantees of correct behavior, as opposed to just informally seeking to achieve requirements given by an FS. To support our discussion, we first review key background material on the established formal verification techniques for discrete, continuous, and hybrid systems and standalone NNs. To our knowledge, no existing work offers a comprehensive, high-level overview of every verification method, and we aim to provide such an overview here. We then present an organized account of the current research on verifying realistic robotic systems with learned policies.

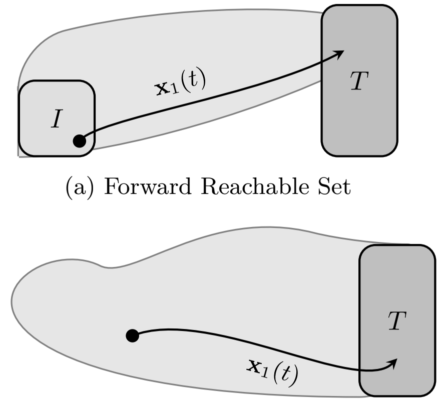
4.1 Preliminaries

Verification for Discrete Systems As mentioned already, verification methods were originally developed in the domain of software to determine whether a given program satisfies a formal specification. The goal is to establish, ideally through proof, that no execution of the program can violate the specification. A central class of these techniques is model checking, which operates on a discrete model of the program. By exhaustively exploring all possible executions of the model and checking them against an automaton encoding the specification, model checking can either confirm universal satisfaction or produce a counterexample trace that violates the specification (Baier et al., 2008). One often preforms reachability analysis as part of model checking, which focuses on determining if the system can ever reach an undesirable or target state from another state. For example, it can be used to verify that a concurrent program never enters a deadlock state.

In contrast to formal verification, other validation methods such as falsification provide weaker guarantees. Falsification, sometimes called semi-formal verification, attempts to find a counterexample input under which the system violates the specification. While effective at identifying bugs, falsification does not offer any guarantee of correctness in the absence of a counterexample, unlike formal verification, which aims to provide a proof of correctness across all executions.

Verification for Continuous and Hybrid Systems While exhaustively checking all possible system behaviors is infeasible for infinite-state systems, some specialized verification techniques have still been extended to apply to these settings. Reachability analysis is one of the most successful techniques that has been generalized from the discrete setting to both continuous (Bertsekas & Rhodes, 1971) and hybrid (i.e., mixed discrete-continuous) systems (Alur et al., 1995). In these contexts, reachability analysis yields an approximate continuous set of reachable states. Furthermore, the reachability computation can either be computed forward in time from some initial set of states I—resulting in a forward reachable set—or backward in time from some target set of states T—resulting in a backward reachable set. A visualization for these two types of reachability computation is given in Figure 6. These computations then can be used to verify a wide range of safety and liveness properties, often given as "Reach-Avoid specifications" (e.g., eventually reach region A and always avoid region B). For example, if one wishes to prove that a system will always avoid an unsafe set T, one can either show that the forward reachable set does not intersect with T or that the backward reachable set from T does not include the current system state. As a result, the forward reachable set only makes a useful verification of the overall system when considering a specific "worst-case" initial set to start from, whereas a backward reachable set lets the agent know in general from what states safety is guaranteed and from which states it is not (Mitchell, 2007).

Reach-avoid specifications are less expressive than the general FSs discussed in Section 2, but they still represent a highly challenging class of problems. For many systems, reachability queries are formally undecidable, meaning no algorithm can solve them in finite time for all instances (Fijalkow et al., 2019). Therefore, a range of approximate methods have been developed, aiming to balance scalability with minimal approximation error. These include set-propagation techniques (Althoff et al., 2021), numerical methods based on Hamilton-Jacobi equations (Mitchell et al., 2005; Bansal et al., 2017), and approaches based on simulation and sensitivity analysis (Girard & Pappas, 2006). Several verification toolkits have been built around these methods (Frehse et al., 2011; Annpureddy et al., 2011; Chen et al., 2013; Althoff, 2015; Duggirala et al., 2015; Kong et al., 2015; Fan et al., 2016). Nevertheless, these tools face practical limitations in terms of scalability and assumptions about model availability, which are central challenges examined in much of the literature surveyed here.



(a) Forward Reachable Set

(b) Backward Reachable Set

Figure 6: Reachability analysis for continuous and hybrid systems can verify all possible ways a system may evolve from a set of initial states I, or all possible ways a system might possibly evolve toward a set of target states T.

Another class of verification methods for continuous dynamical systems, known as certificate function methods, has been developed to verify specific system properties studied in classical dynamical systems theory. These methods rely on the construction of scalar functions, namely Lyapunov functions, Barrier functions, and Contraction metrics, whose existence implies desirable control properties. The existence of a Lyapunov function V certifies that a system will be eventually driven toward a stable equilibrium region since it guarantees that $V(x)$ always decreases along system trajectories to eventually reach zero, i.e., $V(x) \to 0$ as $t \to \infty$. In contrast, a Barrier function B certifies safety rather than goal-reaching: if the system starts within the zero sublevel set $\{x : B(x) < 0\}$, then the system remains in this safe set indefinitely, never crossing into the superlevel set $\{x : B(x) > 0\}$. Finally, contraction metrics generalize Lyapunov functions to time-varying or trajectory-tracking contexts. Instead of certifying convergence to a single point, they prove that a system can be controlled to converge exponentially toward any given reference trajectory, provided it is dynamically feasible. An informative introduction and survey of these techniques is provided in Dawson et al. (2023), so we do not include a full theoretical treatment here. Instead, we focus on how certificate function methods compare to other available verification approaches for dynamical systems, particularly in terms of their flexibility, scalability, and applicability to learned or black-box systems.

Verification for Neural Networks The aforementioned verification methods have been historically developed in the context of continuous and hybrid dynamical systems commonly modeled as linear systems, general nonlinear systems defined by Ordinary Differential Equations (ODEs), piecewise-affine systems, polynomial systems, or hybrid automata combining discrete modes with continuous dynamics. Verifying learned robot policies requires incorporating neural networks into system dynamics. Yet, the standalone verification of neural networks has already attracted substantial research attention, highlighting the difficulty of the problem. Methods for NN verification are sometimes said to address the "open-loop" verification problem, as they only address the verification of properties for a standalone control without considering the feedback loop between the controller and the plant. If the plant model is available in a compatible representation, many existing techniques can be viewed as verifying only single-step or non-temporal properties of the closed-loop system. The most significant techniques for this problem have reused satisfiability solvers (Pulina &

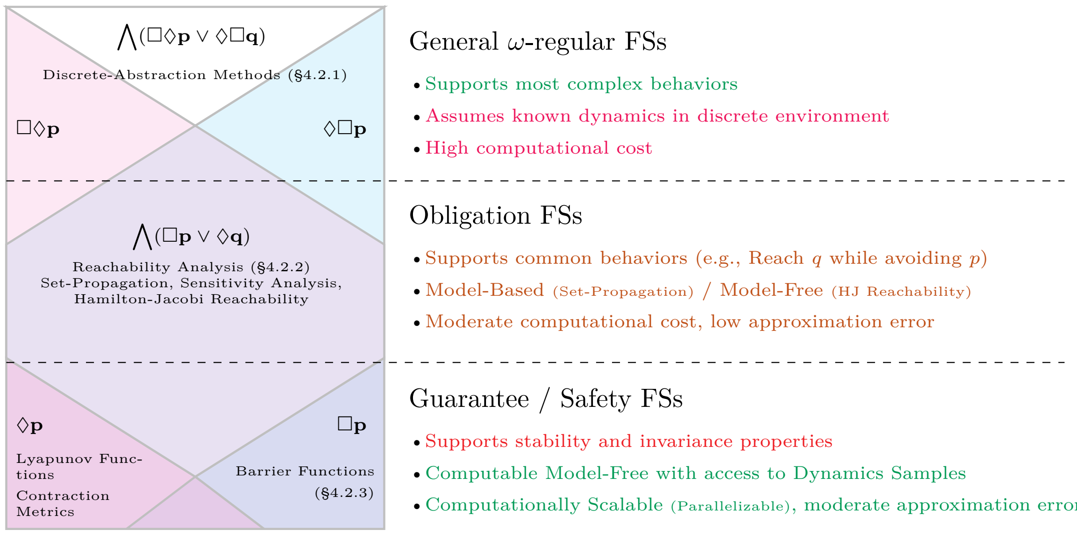


Figure 7: Our survey of robot policy verification methods is organized by three major categories of temporal FSs, visualized here using the temporal property hierarchy of Manna & Pnueli (1990). The first category (top), primarily represented by methods using discrete system abstractions, is defined by support for general ω-regular FSs. The second category (middle) supports the more limited "obligation" class of FSs and is primarily represented by reachability analysis methods. The final category (bottom) supports particular forms of safety (invariance) and guarantee (eventual stability) FSs and is represented by certificate function methods. The typical capabilities of these methods are included and approximately rated as either poor in red, moderate in orange, or good in green.

Tacchella, 2010; Katz et al., 2017), reachability analysis (primarily set-propagation methods) (Tran et al., 2019a; 2020a), optimization, and search. Reachability analysis for NN verification (as opposed to systems verification) treats the NN itself as a discrete time dynamical system with each transition induced by the i^{th} layer Li et al. (2019); Gehr et al. (2018); Singh et al. (2019a; 2018; 2019b;c); Lomuscio & Maganti (2017). Optimization and search approaches attempt to find a maximally or sufficiently specification-violating input Tjeng et al. (2019); Dvijotham et al. (2018); Wong & Kolter (2018); Bunel et al. (2020); Fazlyab et al. (2021); Bastani et al. (2017); Raghunathan et al. (2018). These techniques are also surveyed in Liu et al. (2021). While they are useful for verifying a restricted class of input-output properties for NN controllers, we focus only on the methods developed for verifying richer *temporal* properties of NN-controlled systems as this is most relevant to the domain of robot policies.

4.2 Current Techniques

We now survey recent methods developed to verify that learned robot policies satisfy FSs. For clarity of discussion, we uniformly frame these techniques as all aiming to prove that, for all initial states in some set I, a given NN controller π operating within a dynamical system f will generate a trajectory τ that satisfies a specification ϕ. Under this unified view, the surveyed approaches can be compared along several axes—for example, the assumptions made about the policy π, the system f, computational scalability, or the specification ϕ. We choose to primarily organize our survey around the last axis, as it naturally aligns with the others: supporting more complex specifications typically requires trade-offs in scalability and stronger assumptions on the system model and policy. Figure 7 summarizes the surveyed work across the three major categories along this axis of FS support. While this categorization provides a useful structure, many methods within each category blur these boundaries as they seek to relax the assumptions made by the most established approaches. We also describe these methods and the additional capabilities they provide in the following sections.

4.2.1 Discrete-Abstraction Methods

In Section 3.1, we discussed how discrete abstractions offer a significantly more tractable form of system model for controller synthesis. Consequently, many methods adopt the approach of constructing a finite-state abstraction of a continuous system and then applying reactive synthesis to design a discrete controller, which can subsequently be refined into a continuous one (Dimitrova & Majumdar, 2014). The same principle applies to verifying system properties, and some early approaches for verifying NN-controlled systems also rely on such abstractions in conjunction with verification methods for discrete systems. However, relatively little research has explored using discrete abstractions to simplify the overall environment and dynamics, likely due to the restrictive assumptions required to abstract the system in the first place, which limit their applicability. A prominent example is Sun et al. (2019), who propose a method that, under assumptions of linear system dynamics and a Rectified Linear Unit (ReLU)-based NN architecture, utilizes a discrete system abstraction to verify closed-loop properties of a NN controller equipped with a LiDAR perception system. Their approach formulates these guarantees as Satisfiability Modulo Convex Programming (SMC) queries (Shoukry et al., 2017), akin to well-established Satisfiability Modulo Theories (SMT) tools, using their custom system model. A similar strategy, with comparable assumptions but additional support for system stochasticity, is presented by Sun et al. (2022b).

While these abstraction-based methods offer considerable flexibility in the temporal specifications they can handle and provide strong correctness guarantees, the assumptions they impose are often seen as burdensome and unrealistic for many practical systems. Moreover, the discrete abstractions themselves are prone to combinatorial explosions in complexity, which further limits their practicality (Dimitrova & Majumdar, 2014; Lindemann & Dimarogonas, 2019). As a result, these approaches have attracted less attention compared to methods that offer greater computational scalability.

4.2.2 Reachability-Based Methods

One class of computationally scalable verification methods is based on reachability analysis. Just as reachability analysis serves as a more tractable alternative to full model checking in traditional FMs, it also offers a practical alternative to discrete-abstraction-based approaches for continuous systems. As discussed in Section 4.1, several effective techniques exist for applying reachability analysis to high-dimensional continuous dynamics. Due to its scalability and despite some trade-offs in generality, a substantial portion of recent work has focused on reachability analysis as a viable means of verifying learned policies.

Set-Propagation Set-propagation methods for systems with NN controllers naturally emerged by combining existing techniques developed independently for NNs and for dynamical systems without learned components. Several early works took this integrative approach, adapting methods from both communities to analyze closed-loop systems. One of the first such methods was proposed by Akintunde et al. (2018), who introduced a verification approach for closed-loop Linear Time-Invariant (LTI) systems with ReLU-based NN controllers using Mixed-Integer Linear Programming (MILP). Their method built on prior work in verifying standalone NNs by Lomuscio & Maganti (2017). Using MILP, one can encode constraints to ensure that an initial state x belongs to a given input set I, that it is transformed layer by layer according to the NN's structure, and that the resulting output lies within a target set O. If the resulting optimization problem is feasible, then some state in I can reach O. Because MILP solvers can only accommodate piecewise-linear activations like ReLU, each neuron requires a binary variable to handle its two linear regions. By constructing such a program with n layers of transformation, one can check whether the output set O is reachable in n steps from some state in I. Moreover, by interpreting O as a set to be avoided, the method can verify safety properties of the form $\phi = \Box x \notin O$.

Later work by Xiang et al. (2018); Xiang & Johnson (2018) extended the use of MILP by explicitly computing the bounds of a hyper-rectangle that over-approximates the output set of a ReLU-based NN. These bounds could then be passed to external tools for reachability analysis of systems modeled by piecewise-linear or ODE-based dynamics. The Verisig method proposed by Ivanov et al. (2019) similarly repurposed existing system-level reachability tools by translating sigmoid-based NNs into hybrid systems modeled with ODEs, leveraging the fact that the sigmoid activation function satisfies a quadratic differential equation. This translation enabled the composition of the neural network with the plant model, allowing the verification of the resulting closed-loop system using tools such as Flow* (Chen et al., 2013). However, these early approaches frequently suffered from significant over-approximation errors during reachability analysis. A major factor contributing to this issue was the reliance on hyper-rectangular set representations, which poorly capture the often non-convex and non-contiguous output sets produced by neural networks. More generally, directly composing the reachability computations of the NN and the dynamical system often led to rapidly compounding approximation errors due to the well-known wrapping effect (Neumaier, 1993). To address these issues, Dutta et al. (2019) proposed approximating the NN function itself, rather than just its output set, using a Taylor polynomial over a specified input range. Their tool, named Sherlock, uses this approximation to create a more accurate composite model with the plant, leading to improved precision in the computed reachable set. This Taylor-model approach was later adopted and refined in follow-up versions of Verisig, which introduced techniques such as Taylor-model preconditioning and shrink-wrapping (Ivanov et al., 2020; 2021). A related line of work replaces Taylor models with Bernstein polynomials to abstract the NN, as implemented in the ReachNN tool (Huang et al., 2019). This approach reduces dependence on specific activation functions and supports networks with heterogeneous activations.

NNV, a tool introduced by Tran et al. (2020b), integrates and generalizes many of these earlier ideas. It implements a wide range of NN reachability algorithms based on different set representations, including polytopes (Tran et al., 2019b), star sets (Tran et al., 2019a), zonotopes (Singh et al., 2018), and ImageStar sets (Tran et al., 2020a). When combined with system-level reachability components, NNV supports verification of both feedforward and convolutional NNs, across a variety of activation functions. The follow-up NNV 2.0 release significantly enhanced the tool's scalability and numerical accuracy, while extending support to neural ODEs, semantic segmentation networks, and RNNs (Lopez et al., 2023). The recent work by Hashemi et al. (2023) extends set-propagation methods beyond one assumption common in the work discussed so far: the restriction to handling reach-avoid specifications. Instead, they target more general specifications written in STL by constructing an NN that maps initial system states to the robustness of the resulting trajectories with respect to the STL formula. Using this network, they apply standard neural network verification techniques to determine whether any initial states lead to negative robustness values. This enables the verification of policies against a broader class of temporal specifications using only reachability-based methods.

Sensitivity Analysis Despite extensive development for set propagation methods, they are fundamentally limited by their reliance on specific formats of highly accurate and deterministic system models. A slightly more flexible approach is available through simulating many trajectories and analyzing their sensitivity to their starting conditions or properties of the system related to trajectory divergence (Girard & Pappas, 2006; Donzé & Maler, 2007; Ramdani et al., 2008; Duggirala et al., 2013). One use case for sensitivity analysis is significantly refining the reachable sets computed by set propagation methods (Ladner & Althoff, 2023). It can also be used on its own, as in one algorithm presented in Hashemi et al. (2023) that supports a more general class of models not necessarily using ReLU activations. Their method involves dense sampling of the initial state space and estimation of the Lipschitz constant of the system's robustness function, which is then used to certify the remaining, non-sampled points. In a similar spirit, Salamati et al. (2020) propose a Bayesian inference-based technique that operates with a partially unknown model and a dataset of trajectories, constructing a posterior distribution to characterize the satisfaction probability of a given STL specification.

Hamilton-Jacobi Reachability Analysis Finally, a significant body of recent research on verifying NN policies is based on HJ reachability analysis. At its core, HJ reachability models the interaction between a control system and an adversarial disturbance as a two-player zero-sum differential game (Tomlin et al., 2000). The goal is to determine the set of states from which the system can be guaranteed to reach (or avoid) a target set despite worst-case disturbances. This problem can be formulated as a HJ Partial Differential Equation (PDE), whose solution defines a value function over the state space. The zero-superlevel set of this value function then represents the set of states from which one player is able to win, which yields the backward reachable set from a given target region (Mitchell et al., 2005).

Unlike set-propagation or sensitivity-based methods, traditional HJ reachability solves a PDE using grid-based dynamic programming, whose computational cost scales exponentially with the dimension of the state space. Addressing this scalability issue has been a central focus in research efforts aiming to apply HJ reachability to the verification of realistic robot policies. One promising direction is the use of deep-RL-style approximate dynamic programming techniques, with a contractive operator derived from the reachability PDE, to obtain a NN-approximated reachability value function (Fisac et al., 2019; Hsu et al., 2021). These methods improve scalability and, crucially, eliminate the need for an explicit model of the system. However, they typically require large amounts of simulated experience and may carry safety risks during training, limiting their applicability in real-world deployments unless reliable simulators or offline data are available. A second major approach, introduced by Bansal & Tomlin (2021), draws inspiration from physics-informed NNs. The HJ reachability PDE is encoded directly into a loss function that provides guidance for training a sinusoidal-NN (Sitzmann et al., 2020) to approximate the PDE solution. While this method therefore assumes known system dynamics and boundary conditions, it benefits by being significantly more data-efficient than RL-like, sample-based approaches .

Extensions of HJ reachability methods have also been proposed to support several new features. Bansal et al. (2020) enable the computation of probabilistic reachable sets, a crucial capability for safe navigation around humans, where reasoning about uncertainty and intent is essential. Yu et al. (2022) integrate reachability-based safety methods with RL frameworks to jointly learn safe policies and their associated safety sets. Some work has even sought to bridge HJ reachability with STL (Chen et al., 2020) to support richer classes of safety specifications, similar to some efforts for extending set-propagation methods.

4.2.3 Certificate-Function Based Methods

The final category of verification methods we survey derives proofs of FS satisfaction through *certificate functions*. As noted above, these FSs are typically limited to properties concerning system invariance (e.g., remaining on one side of a barrier) or stability (e.g., eventually converging to an equilibrium point or trajectory). While such properties are not often expressed as LTL specifications, we include them as such in Figure 7 to facilitate comparison with other verification methods. In exchange for their restricted expressiveness, certificate-based methods offer the advantages of not requiring an explicit system model and achieving favorable computational scalability.

A certificate function is obtained by a search over a set of differentiable functions, with constraints determined by the particular certificate type (Dawson et al., 2023). Naturally, this function space can be represented using a family of parameterized NNs, enabling the search to be cast as an unconstrained training problem by incorporating the constraints as penalty terms. The objective is then evaluated empirically over a finite set of system trajectories. This forms the basic approach underlying most certificate-learning methods (Richards et al., 2018; Srinivasan et al., 2020; Zhao et al., 2020). Subsequent research has addressed both challenges and downstream applications of this approach, including joint learning of certificates and dynamics models (Kolter & Manek, 2019), rapid adaptation of nominal certificates to new systems (Taylor et al., 2019), and new NN architectures tailored for certificate functions (Gaby et al., 2022). One particularly important challenge is that learned certificate functions resulting from this procedure are not guaranteed to satisfy their required properties. This can arise for two reasons: the NN search space may be too limited to contain a valid certificate or the empirical loss may fail to enforce constraints across the entire state space. Therefore, new formal synthesis techniques for certificate functions, based on CEGIS, have been introduced which adaptively refine the NN search space (Peruffo et al., 2021) and verify certificate constraints using SMT-solvers (Abate et al., 2021b;a; Edwards et al., 2024).

Certificate-function-based verification also differs from other methods in that it naturally extends to the synthesis of formally verified controllers. Just as Lyapunov and Barrier functions certify the stability and invariance of a closed-loop control system, Control Lyapunov Functions (CLFs) and Control Barrier Functions (CBFs) certify the existence of a controller for an open-loop system that can yield a closed-loop system with the desired properties. This has motivated several lines of research into certificate-regularized imitation learning (Cosner et al., 2022) and reinforcement learning (Perkins & Barto, 2003; Chow et al., 2018; Chang et al., 2019; Xiong et al., 2022; Marvi & Kiumarsi, 2021; Emam et al., 2025; Ma et al., 2021). In this synthesis context, other work has also sought to generalize CLFs and CBFs to support a broader range of STL-specified properties, through constructions such as time-varying CBFs (Lindemann & Dimarogonas, 2019). For a more comprehensive overview of certificate-based synthesis methods and of certificate functions more broadly, we again refer the reader to Dawson et al. (2023).

4.2.4 Falsification

Finally, several notable methods have instead focused on the problem of falsification or "semi-formal" verification mentioned in Section 4.1. Falsification is the process of showing that there exists an initial state from which the system follows a trajectory τ that does *not* satisfy a specification ϕ. Although this is logically separated from the above verification problem by only a single negation operator, this form of the problem can be easier to solve and more easily used to relax assumptions on the system, controller, and specification. One benefit of this formulation is that one can find a solution with a non-exhaustive search through the set I. Broad approaches to falsification can involve some form of stochastic optimization (Das et al., 2021) or robustness-guided RL (Yamagata et al., 2021). More recent approaches have enhanced the falsification process by incorporating notions of NN "coverage," aiming to ensure that the network's activation space is thoroughly explored during falsification (Zhang et al., 2023). This idea draws inspiration from prior work on automated NN testing (Sun et al., 2018; Pei et al., 2019). Other studies such as Dreossi et al. (2019) address the falsification of multi-component systems that combine DL for perception with classical control. Their approach alternates between searching for specification violations in the classical control component—using inputs from various abstractions of the DL perception component—and identifying errors in the DL component itself, which then inform and refine those abstractions. These methods, although not providing the same guarantees as the above verification methods, anticipate the trade-off between guarantees and practicality. They still provide insight into the safety of a learned robot policy but scale better to real-world scenarios and require significantly fewer assumptions on the network architecture and model design.

5 Future Research Directions

Finally, we summarize in this section the most significant remaining questions in formally specifying robot behavior, synthesizing policies to follow those specifications, and verifying that extant learned policies also follow them. For each specific area of FMs covered in our survey, we describe nascent and hypothetical research directions that we believe are promising. Moreover, we motivate these new directions with links to the larger themes we have identified as already having significance in the work surveyed above. For instance, one theme is the struggle to optimally balance formality (i.e., the strength of guarantees) and versatility (i.e., the degree of supported specification, policy, and system complexity). Another is the increasing reliance on scalable numerical methods and decreasing favor for discrete-abstraction-based methods for synthesis and verification. We hope that these directions and the underlying trends they belong to will promote future research toward the complete resolution of safety, interpretability and trustworthiness concerns in the modern wave of robot learning research.

5.1 Specifications

Specification Mining Within SM, extracting specifications from the full, unrestricted STL grammar remains largely intractable due to the immense search space and the limited guidance provided by datasets of observed behaviors alone. A key direction for making this problem more tractable is to incorporate external domain knowledge to constrain and guide the search. This presents a natural opportunity to leverage foundational language models, which have already been shown to be highly effective complements to genetic algorithms by providing inductive biases or structural priors over languages (Novikov et al., 2025). Incorporating knowledge from a system model—either by simulating additional trajectories to enrich the dataset or by analytically reasoning about the model's behavior to rule out irrelevant specifications—is another underexplored and promising direction. Current SM methods typically focus on extracting a narrow class of specifications from specific types of datasets, most often learning unconditional formulas from complete system traces. Future work may expand this scope by targeting conditional, input-output specifications, especially from partial or noisy observations. This capability is particularly critical when learning specifications that express long-term or infinite-horizon objectives, such as those described by ω-regular languages, which cannot be validated over finite traces and instead require extrapolation or inductive generalization (Bartocci et al., 2022). Lastly, SM holds significant promise in its applicability to constructing interpretable, temporally structured representations of goals for artificially intelligent agents. Liu et al. (2025) demonstrates one use-case in IRL, but future work can also explore leveraging specifications for interpreting existing black-box AI behavior, guiding other policy learning methods with formal supervision, and enhancing inference-time control through formally grounded goal representations.

Partial Observability and Stochasticity Obtaining and representing specifications for systems featuring stochasticity and partial observability still faces significant challenges. There are a small number of works investigating stochasticity (Yoo & Belta, 2015; Sadigh & Kapoor, 2016) or partial observability (Kapoor et al., 2025a) individually, but seemingly no methods pursuing them simultaneously. Future work may hypothetically extend the approach taken by Kapoor et al. (2025a) and incorporate more sophisticated and contextual probabilistic models to build formal specifications whose semantics would seamlessly incorporate multi-modal distributions for its constituent atomic propositions. This would continue the trend of increasing FS flexibility at the expense of formal specificity, so additional work may be necessary to ensure adequate guarantees are obtained when applying synthesis and verification methods to these FSs. Furthermore, future work can extend FS representations and implementation frameworks to accommodate these more esoteric FS features. Differentiable STL frameworks can naturally be extended to support differentiable probabilistic models. Automaton-based representations supporting probabilistic propositions also already exist (Rheinboldt & Paz, 2014) but may need adaptation for application to automata-theoretic RL.

5.2 Synthesis

Handling more Complex Specifications A broad limitation in current research on formal synthesis lies in the temporal complexity of the supported FSs. We believe this gap presents an opportunity for both theoretical and practical advances. On the theoretical side, several fundamental obstacles have been identified in satisfying complex LTL specifications with RL. Addressing these challenges remains an open and compelling direction for future work. Notably, Yang et al. (2022) show that RL for almost all LTL objectives under their default semantics is intractable. That is, no algorithm can guarantee near-optimal performance for anything beyond the simplest LTL objectives given any finite number of environment interactions. Despite this, the work surveyed in Section 3.2.1 demonstrates progress by relying on reward structures derived from relaxed LTL semantics. Nonetheless, developing a unified and theoretically grounded solution to this intractability will be essential for establishing a more robust foundation for combining LTL and RL. One promising direction comes from Alur et al. (2023), who propose a discounted LTL semantics that avoids the intractability while still supporting expressive specifications. This framework offers a fertile starting point for future research. It is also the case that these more complex specifications quickly increase the sparsity of rewards, for which less-theoretical research efforts can investigate practical strategies such as incorporating additional model-information with differentiable simulators (Bozkurt et al., 2025), introducing additional rewards in the experience replay buffer (Voloshin et al., 2023), or improving exploration with evolutionary RL algorithms (Zhu et al., 2021).

Combining Formal and Informal Requirements A distinct avenue for future research could instead focus on combining LTL constraints with standard RL objectives, as opposed to the existing algorithms that solely focus on solving tasks completely specified with an LTL expression. This is a critical direction, as many FSs are more meaningful when the agent pursues a competing objective, requiring it to make optimal decisions that balance task performance with formal constraints. For instance, complex specifications imposing recurrence and fairness constraints can have trivial solutions in the absence of another objective. A concrete example would be the specification “always periodically return to the charging station” in combination with a normal reward for collecting and organizing objects. The agent must then decide when to pursue objects and when to return to the charging station. Without the competing objective, however, the optimal behavior would be to remain permanently at the charging station. A possible starting point for this new direction is generalizing standard constrained MDP constraints (Altman, 1999) to accommodate functions of LTL reward or STL robustness.

Better Offline Learning with FSs In the same vein, combining LTL constraints with objectives used in offline policy learning is also a fruitful and insufficiently explored direction. Offline policy learning is already considered important for safety critical settings due to circumventing the need for safe exploration, so their combination is natural. A possible broad strategy for this direction could apply innovations in automata-theoretic RL for handling non-Markovian objectives to the offline RL setting (Levine et al., 2020). Initially, research in this direction may examine enhancing methods for offline RL specifically for datasets with rewards and states augmented with an automaton known a priori. Later work can then return to using standard datasets and incorporate additional guidance from specifications mined from the available data (Section 2.2.1). The final and most ambitious goal for this line of research would be eventually generalizing the concept of learning optimal goal-reaching policies from suboptimal data to learning general specification-following policies from suboptimal data not necessarily following *any* specification.

5.3 Verification

The final major research direction we identify from the surveyed literature is in continuing to scale verification methods to support modern robot policies deployed in the real-world. While recent work has already made substantial progress toward verifying satisfaction of FSs under increasingly relaxed assumptions and for more complex, high-dimensional systems, many of these methods still face significant challenges. Among the most scalable and promising lines of research are sampling-based (Lew & Pavone, 2021), dynamic-programming-based (Hsu et al., 2021; Fisac et al., 2019), and general neural-network-approximation-based (Bansal & Tomlin, 2021; Edwards et al., 2024) verification methods. We anticipate these will provide a foundation for research into more practical verification pipelines, and that this research will proceed by carefully conceding the currently offered strong guarantees in exchange for computational scalability, fewer assumptions on the environment, and support for significantly more complex models.

Relaxing Guarantees for Scalability Several recent works already exemplify this trend through probabilistic extensions of reachability analysis and certificate-based methods, which allow for reasoning over stochastic system models (Lew & Pavone, 2021; Bansal et al., 2020; Castañeda et al., 2024). This probabilistic framing is particularly suitable for unstructured real-world deployment, where full certainty of future events is almost never achievable and models ought to be learned or refined online. Accounting for verification based on complex stochastic world models also paves the way for, assuming sufficient computational scalability, online runtime verification. In this paradigm, robots would assess policy safety on-the-fly using up-to-date environment observations and short-term predictions. While these guarantees may be local in time or hold only with high probability, they are often the most actionable and scalable approach for long-horizon tasks in uncertain environments.

Large Policy Models A significant amount of future work can also be devoted to scalability concerns for more modern, complex policy models, which remains a bottleneck for many formal verification pipelines. Many current verification tools operate under restrictive assumptions about policy architectures; they often only targeting shallow or moderately deep feedforward networks with piecewise-linear activations. However, state-of-the-art policies increasingly rely on much more expressive and compositional architectures such as transformer and diffusion-based generative models. These models are often opaque to current analyzers, but initial efforts toward verifying these models using reachability analysis have begun to address these challenges (Shi et al., 2020; Manzanas Lopez et al., 2022; Tobias Ladner, 2025). Still, future research will have to further extend these techniques or pioneer new ones to handle the complex networks used in real robot policies.

6 Conclusion

Throughout this survey, we have shown that FMs in robot policy learning have the potential to substantially increase our confidence in the safe, correct, and reliable behavior of autonomous systems. To make this case, we reviewed current research organized into three major categories. First, we surveyed techniques for specifying robot behavior through formal specifications (Section 2), which provide an unambiguous language for expressing complex task requirements and safety constraints. These include works on learning specifications from demonstrations and representing these specifications as loss functions, reward functions, and embeddings. Second, we described approaches for synthesizing policies that satisfy such specifications (Section 3). These include methods that incorporate automata-based rewards and other forms of FS guidance into model-free and model-based RL, techniques in IRL that produce interpretable reward functions in the form of FSs, and behavior cloning and offline RL approaches that integrate supervision or conditioning based on FSs. Finally, we examined a complementary set of verification (and falsification) techniques that aim to prove (or disprove) whether a learned policy satisfies a given specification (Section 4). The verification methods range from the most formal but least scalable methods based on formulating and checking discrete systems abstractions, to the more focused, moderately scalable reachability-analysis methods, and finally to the most specific and scalable certificate function methods. These three categories—specification, synthesis, and verification—and the hierarchy they impose cover the entirety of current research introducing FMs for robot policy development.

Beyond summarizing the state of the art, our survey provides a coherent organizational framework that can serve both new entrants and seasoned researchers in the field. For readers newly approaching formal methods in robot learning, we have included accessible illustrations, examples, and conceptual overviews that clarify foundational ideas and map out the landscape of existing work. For experts, our survey offers a concise yet comprehensive taxonomy of techniques, which also provides starting points for deeper inquiry into all the surveyed subfields and draws connections between them that are not often discussed. Most importantly, by summarizing current trends across specification, synthesis, and verification research, we identify a number of underexplored questions and promising future directions (Section 5). In particular, we emphasize the need to increase the expressiveness and scalability of current FMs to support learning and verification in real-world robotic systems. We believe that achieving this goal will be essential to ultimately deploying robots that are not just performant, but provably safe and trustworthy.

References

1X Technologies. NEO Gamma | 1X, 2025. URL `https://www.1x.tech/neo`.

Alessandro Abate, Iury Bessa, Lucas Cordeiro, Cristina David, Pascal Kesseli, Daniel Kroening, and Elizabeth Polgreen. Automated formal synthesis of provably safe digital controllers for continuous plants. *Acta Informatica*, Dec 2019. ISSN 1432-0525. doi: 10.1007/s00236-019-00359-1.

Alessandro Abate, Daniele Ahmed, Alec Edwards, Mirco Giacobbe, and Andrea Peruffo. Fossil: a software tool for the formal synthesis of lyapunov functions and barrier certificates using neural networks. In *Proceedings of the 24th International Conference on Hybrid Systems: Computation and Control*, HSCC '21, New York, NY, USA, 2021a. Association for Computing Machinery. ISBN 9781450383394. doi: 10.1145/3447928.3456646. URL `https://doi.org/10.1145/3447928.3456646`.

Alessandro Abate, Daniele Ahmed, Mirco Giacobbe, and Andrea Peruffo. Formal synthesis of lyapunov neural networks. *IEEE Control Systems Letters*, 5(3):773–778, July 2021b. ISSN 2475-1456. doi: 10.1109/lcsys.2020.3005328. URL `http://dx.doi.org/10.1109/LCSYS.2020.3005328`.

Alessandro Abate, Yousif Almulla, James Fox, David Hyland, and Michael Wooldridge. Learning Task Automata for Reinforcement Learning Using Hidden Markov Models. In *ECAI 2023*, pp. 3–10. IOS Press, 2023. doi: 10.3233/FAIA230247. URL `https://ebooks.iospress.nl/doi/10.3233/FAIA230247`.

Pieter Abbeel and Andrew Y. Ng. Apprenticeship learning via inverse reinforcement learning. In *Proceedings of the Twenty-First International Conference on Machine Learning*, ICML '04, pp. 1, New York, NY, USA, 2004. Association for Computing Machinery. ISBN 1581138385. doi: 10.1145/1015330.1015430. URL `https://doi.org/10.1145/1015330.1015430`.

Michael E. Akintunde, Alessio Lomuscio, Lalit Maganti, and Edoardo Pirovano. Reachability analysis for neural agent-environment systems. In *International Conference on Principles of Knowledge Representation and Reasoning*, 2018. URL `https://api.semanticscholar.org/CorpusID:53012905`.

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)*, pp. 6565–6570, 2016. doi: 10.1109/CDC.2016.7799279.

Matthias Althoff. An introduction to CORA 2015. In *Proc. of the 1st and 2nd Workshop on Applied Verification for Continuous and Hybrid Systems*, pp. 120–151. EasyChair, December 2015. doi: 10.29007/zbkv. URL `https://easychair.org/publications/paper/xMm`.

Matthias Althoff, Goran Frehse, and Antoine Girard. Set propagation techniques for reachability analysis. *Annual Review of Control, Robotics, and Autonomous Systems*, 4(Volume 4, 2021):369–395, 2021. ISSN 2573-5144. doi: https://doi.org/10.1146/annurev-control-071420-081941. URL `https://www.annualreviews.org/content/journals/10.1146/annurev-control-071420-081941`.

Eitan. Altman. *Constrained Markov Decision Processes*. Stochastic Modeling Series. Taylor & Francis, 1999. ISBN 978-0-8493-0382-1.

Rajeev Alur and David L. Dill. A theory of timed automata. *Theoretical Computer Science*, 126(2):183–235, 1994. ISSN 0304-3975. doi: https://doi.org/10.1016/0304-3975(94)90010-8. URL `https://www.sciencedirect.com/science/article/pii/0304397594900108`.

Rajeev Alur, Costas Courcoubetis, and David Dill. Model-checking for probabilistic real-time systems. In Javier Leach Albert, Burkhard Monien, and Mario Rodríguez Artalejo (eds.), *Automata, Languages and Programming*, pp. 115–126, Berlin, Heidelberg, 1991. Springer Berlin Heidelberg. ISBN 978-3-540-47516-3.

Rajeev Alur, Costas Courcoubetis, Nicolas Halbwachs, Thomas A. Henzinger, Peihsin H. Ho, Xavier Nicollin, Alfredo M. Olivero, Joseph Sifakis, and Sergio Yovine. The algorithmic analysis of hybrid systems. *Theoretical Computer Science*, 138(1):3–34, 1995. ISSN 0304-3975. doi: https://doi.org/10.1016/0304-3975(94)00202-T. URL `https://www.sciencedirect.com/science/article/pii/030439759400202T`. Hybrid Systems.

Rajeev Alur, Rastislav Bodik, Garvit Juniwal, Milo M. K. Martin, Mukund Raghothaman, Sanjit A. Seshia, Rishabh Singh, Armando Solar-Lezama, Emina Torlak, and Abhishek Udupa. Syntax-guided synthesis. In *2013 Formal Methods in Computer-Aided Design*, pp. 1–8, 2013. doi: 10.1109/FMCAD.2013.6679385.

Rajeev Alur, Rishabh Singh, Dana Fisman, and Armando Solar-Lezama. Search-based program synthesis. *Commun. ACM*, 61(12):84–93, November 2018. ISSN 0001-0782. doi: 10.1145/3208071. URL `https://doi.org/10.1145/3208071`.

Rajeev Alur, Osbert Bastani, Kishor Jothimurugan, Mateo Perez, Fabio Somenzi, and Ashutosh Trivedi. Policy Synthesis and Reinforcement Learning for Discounted LTL. In Constantin Enea and Akash Lal (eds.), *Computer Aided Verification*, pp. 415–435, Cham, 2023. Springer Nature Switzerland. ISBN 978-3-031-37706-8.

Glenn Ammons, Rastislav Bodík, and James R. Larus. Mining specifications. In *Proceedings of the 29th ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages*, POPL '02, pp. 4–16, New York, NY, USA, 2002. Association for Computing Machinery. ISBN 1581134509. doi: 10.1145/503272.503275. URL `https://doi.org/10.1145/503272.503275`.

Philip W. Anderson. More is different. *Science*, 177(4047):393–396, 1972. doi: 10.1126/science.177.4047.393. URL `https://www.science.org/doi/abs/10.1126/science.177.4047.393`.

Dana Angluin. Learning regular sets from queries and counterexamples. *Information and Computation*, 75(2):87–106, 1987. ISSN 0890-5401. doi: https://doi.org/10.1016/0890-5401(87)90052-6. URL `https://www.sciencedirect.com/science/article/pii/0890540187900526`.

Yashwanth Annpureddy, Che Liu, Georgios Fainekos, and Sriram Sankaranarayanan. S-taliro: A tool for temporal logic falsification for hybrid systems. In Parosh Aziz Abdulla and K. Rustan M. Leino (eds.), *Tools and Algorithms for the Construction and Analysis of Systems*, pp. 254–257, Berlin, Heidelberg, 2011. Springer Berlin Heidelberg. ISBN 978-3-642-19835-9.

Brandon Araki, Xiao Li, Kiran Vodrahalli, Jonathan Decastro, Micah Fry, and Daniela Rus. The logical options framework. In Marina Meila and Tong Zhang (eds.), *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 307–317. PMLR, 18–24 Jul 2021. URL `https://proceedings.mlr.press/v139/araki21a.html`.

Eugene Asarin, Alexandre Donzé, Oded Maler, and Dejan Nickovic. Parametric Identification of Temporal Properties. In Sarfraz Khurshid and Koushik Sen (eds.), *Runtime Verification*, pp. 147–160, Berlin, Heidelberg, 2012. Springer Berlin Heidelberg. ISBN 978-3-642-29860-8.

Sertac Kagan Aydin and Ebru Aydin Gol. Synthesis of monitoring rules with stl. *Journal of Circuits, Systems and Computers*, 29(11):2050177, 2020. doi: 10.1142/S0218126620501777. URL https://doi.org/10.1142/S0218126620501777.

Sara Baase. *A Gift of Fire: Social, Legal, and Ethical Issues for Computing and the Internet*. Alternative Etext Formats. Pearson Prentice Hall, 2008. ISBN 9780136008484. URL https://books.google.com/books?id=GcBaS87q74gC.

Marco Bagatella, Andreas Krause, and Georg Martius. Directed exploration in reinforcement learning from linear temporal logic, 2024. URL https://arxiv.org/abs/2408.09495.

Christel. Baier, Joost-Pieter. Katoen, and Kim Guldstrand. Larsen. *Principles of Model Checking*. MIT Press, 2008. ISBN 978-0-262-02649-9.

Anand Balakrishnan and Jyotirmoy V. Deshmukh. Structured reward shaping using signal temporal logic specifications. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 3481–3486, 2019. doi: 10.1109/IROS40897.2019.8968254.

Anand Balakrishnan, Jyotirmoy Deshmukh, Bardh Hoxha, Tomoya Yamaguchi, and Georgios Fainekos. Percemon: Online monitoring for perception systems. In *Runtime Verification: 21st International Conference, RV 2021, Virtual Event, October 11–14, 2021, Proceedings*, pp. 297–308, Berlin, Heidelberg, 2021. Springer-Verlag. ISBN 978-3-030-88493-2. doi: 10.1007/978-3-030-88494-9_18. URL https://doi.org/10.1007/978-3-030-88494-9_18.

Somil Bansal and Claire J. Tomlin. Deepreach: A deep learning approach to high-dimensional reachability. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1817–1824. IEEE Press, 2021. doi: 10.1109/ICRA48506.2021.9561949. URL https://doi.org/10.1109/ICRA48506.2021.9561949.

Somil Bansal, Mo Chen, Sylvia Herbert, and Claire J. Tomlin. Hamilton-jacobi reachability: A brief overview and recent advances. In *2017 IEEE 56th Annual Conference on Decision and Control (CDC)*, pp. 2242–2253, 2017. doi: 10.1109/CDC.2017.8263977.

Somil Bansal, Andrea Bajcsy, Ellis Ratner, Anca D. Dragan, and Claire J. Tomlin. A hamilton-jacobi reachability-based framework for predicting and analyzing human motion for safe planning. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 7149–7155, 2020. doi: 10.1109/ICRA40945.2020.9197257.

Ezio Bartocci, Cristinel Mateis, Eleonora Nesterini, and Dejan Nickovic. Survey on mining signal temporal logic specifications. *Information and Computation*, 289:104957, 2022.

Osbert Bastani, Yani Ioannou, Leonidas Lampropoulos, Dimitrios Vytiniotis, Aditya Nori, and Antonio Criminisi. Measuring neural net robustness with constraints, 2017. URL https://arxiv.org/abs/1605.07262.

Osbert Bastani, Yewen Pu, and Armando Solar-Lezama. Verifiable reinforcement learning via policy extraction, 2019. URL https://arxiv.org/abs/1805.08328.

Calin Belta and Sadra Sadraddini. Formal methods for control synthesis: An optimization perspective. *Annual Review of Control, Robotics, and Autonomous Systems*, 2(Volume 2, 2019):115–140, 2019. ISSN 2573-5144. doi: https://doi.org/10.1146/annurev-control-053018-023717. URL https://www.annualreviews.org/content/journals/10.1146/annurev-control-053018-023717.

Dimitri P. Bertsekas and Ian B. Rhodes. On the minimax reachability of target sets and target tubes. *Automatica*, 7(2):233–247, 1971. ISSN 0005-1098.

Roderick Bloem, Barbara Jobstmann, Nir Piterman, Amir Pnueli, and Yaniv Sa'ar. Synthesis of reactive(1) designs. *Journal of Computer and System Sciences*, 78(3):911–938, 2012. ISSN 0022-0000. doi: https://doi.org/10.1016/j.jcss.2011.08.007. URL https://www.sciencedirect.com/science/article/pii/S0022000011000869. In Commemoration of Amir Pnueli.

Giuseppe Bombara and Calin Belta. Offline and online learning of signal temporal logic formulae using decision trees. *ACM Trans. Cyber-Phys. Syst.*, 5(3), March 2021. ISSN 2378-962X. doi: 10.1145/3433994. URL https://doi.org/10.1145/3433994.

Luca Bortolussi and Laura Nenzi. Specifying and monitoring properties of stochastic spatio-temporal systems in signal temporal logic. In *Proceedings of the 8th International Conference on Performance Evaluation Methodologies and Tools*, VALUETOOLS '14, pp. 66–73, Brussels, BEL, 2014. ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering). ISBN 9781631900570. doi: 10.4108/icst.Valuetools.2014.258183. URL https://doi.org/10.4108/icst.Valuetools.2014.258183.

Alper Kamil Bozkurt, Calin Belta, and Ming C. Lin. Accelerated learning with linear temporal logic using differentiable simulation, 2025. URL https://arxiv.org/abs/2506.01167.

Lukas Brunke, Melissa Greeff, Adam W. Hall, Zhaocong Yuan, Siqi Zhou, Jacopo Panerati, and Angela P. Schoellig. Safe Learning in Robotics: From Learning-Based Control to Safe Reinforcement Learning. *Annual Review of Control, Robotics, and Autonomous Systems*, 5(Volume 5, 2022):411–444, 2022. ISSN 2573-5144. doi: https://doi.org/10.1146/annurev-control-042920-020211. URL https://www.annualreviews.org/content/journals/10.1146/annurev-control-042920-020211. Publisher: Annual Reviews Type: Journal Article.

Rudy Bunel, Alessandro De Palma, Alban Desmaison, Krishnamurthy Dvijotham, Pushmeet Kohli, Philip H. S. Torr, and M. Pawan Kumar. Lagrangian decomposition for neural network verification. *Conference on Uncertainty in Artificial Intelligence*, 2020.

Mingyu Cai, Shaoping Xiao, Baoluo Li, Zhiliang Li, and Zhen Kan. Reinforcement learning based temporal logic control with maximum probabilistic satisfaction. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 806–812, 2021a. doi: 10.1109/ICRA48506.2021.9561903.

Mingyu Cai, Shaoping Xiao, Zhijun Li, and Zhen Kan. Optimal probabilistic motion planning with potential infeasible ltl constraints. *IEEE Transactions on Automatic Control*, 68(1):301–316, 2021b.

Mingyu Cai, Zhangli Zhou, Lin Li, Shaoping Xiao, and Zhen Kan. Reinforcement learning with soft temporal logic constraints using limit-deterministic generalized büchi automaton. *Journal of Automation and Intelligence*, 2024. ISSN 2949-8554. doi: https://doi.org/10.1016/j.jai.2024.12.005. URL https://www.sciencedirect.com/science/article/pii/S2949855424000637.

Alberto Camacho, Rodrigo Toro Icarte, Toryn Q. Klassen, Richard Valenzano, and Sheila A. McIlraith. Ltl and beyond: Formal languages for reward function specification in reinforcement learning. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19*, pp. 6065–6073. International Joint Conferences on Artificial Intelligence Organization, 7 2019a. doi: 10.24963/ijcai.2019/840. URL https://doi.org/10.24963/ijcai.2019/840.

Alberto Camacho, Rodrigo Toro Icarte, Toryn Q. Klassen, Richard Valenzano, and Sheila A. McIlraith. Ltl and beyond: Formal languages for reward function specification in reinforcement learning. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19*, pp. 6065–6073. International Joint Conferences on Artificial Intelligence Organization, 7 2019b. doi: 10.24963/ijcai.2019/840. URL https://doi.org/10.24963/ijcai.2019/840.

Fernando Castañeda, Jason J. Choi, Wonsuhk Jung, Bike Zhang, Claire J. Tomlin, and Koushil Sreenath. Recursively feasible probabilistic safe online learning with control barrier functions, 2024. URL https://arxiv.org/abs/2208.10733.

Ya-Chien Chang, Nima Roohi, and Sicun Gao. Neural Lyapunov Control. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d' Alché-Buc, E. Fox, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper_files/paper/2019/file/2647c1dba23bc0e0f9cdf75339e120d2-Paper.pdf.

Mo Chen, Qizhan Tam, Scott C. Livingston, and Marco Pavone. Signal Temporal Logic Meets Reachability: Connections and Applications. In Marco Morales, Lydia Tapia, Gildardo Sánchez-Ante, and Seth Hutchinson (eds.), *Algorithmic Foundations of Robotics XIII*, pp. 581–601, Cham, 2020. Springer International Publishing. ISBN 978-3-030-44051-0.

Xin Chen, Erika Ábrahám, and Sriram Sankaranarayanan. Flow*: An analyzer for non-linear hybrid systems. In Natasha Sharygina and Helmut Veith (eds.), *Computer Aided Verification*, pp. 258–263, Berlin, Heidelberg, 2013. Springer Berlin Heidelberg. ISBN 978-3-642-39799-8.

Kyunghoon Cho and Songhwai Oh. Learning-based model predictive control under signal temporal logic specifications. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 7322–7329, 2018. doi: 10.1109/ICRA.2018.8460811.

Yinlam Chow, Ofir Nachum, Edgar Duenez-Guzman, and Mohammad Ghavamzadeh. A lyapunov-based approach to safe reinforcement learning. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018. URL https://proceedings.neurips.cc/paper_files/paper/2018/file/4fe5149039b52765bde64beb9f674940-Paper.pdf.

Xu Chu (Dennis) Ding, Stephen L. Smith, Calin Belta, and Daniela Rus. Ltl control in uncertain environments with probabilistic satisfaction guarantees*. *IFAC Proceedings Volumes*, 44(1):3515–3520, 2011. ISSN 1474-6670. doi: https://doi.org/10.3182/20110828-6-IT-1002.02287. URL https://www.sciencedirect.com/science/article/pii/S1474667016441558. 18th IFAC World Congress.

Alonzo Church. Application of Recursive Arithmetic to the Problem of Circuit Synthesis. *Journal of Symbolic Logic*, 28(4):289–290, 1963. Publisher: Association for Symbolic Logic.

Max H. Cohen and Calin Belta. Model-based reinforcement learning for approximate optimal control with temporal logic specifications. In *Proceedings of the 24th International Conference on Hybrid Systems: Computation and Control*, HSCC '21, New York, NY, USA, 2021. Association for Computing Machinery. ISBN 9781450383394. doi: 10.1145/3447928.3456639. URL https://doi.org/10.1145/3447928.3456639.

Ryan K. Cosner, Yisong Yue, and Aaron D. Ames. End-to-end imitation learning with safety guarantees using control barrier functions. In *2022 IEEE 61st Conference on Decision and Control (CDC)*, pp. 5316–5322, 2022. doi: 10.1109/CDC51059.2022.9993193.

Costas Courcoubetis and Mihalis Yannakakis. The complexity of probabilistic verification. *J. ACM*, 42(4): 857–907, July 1995. ISSN 0004-5411. doi: 10.1145/210332.210339. URL https://doi.org/10.1145/210332.210339.

Maxwell Crouse, Ibrahim Abdelaziz, Cristina Cornelio, Veronika Thost, Lingfei Wu, Kenneth Forbus, and Achille Fokoue. Improving graph neural network representations of logical formulae with subgraph pooling, 2020. URL https://arxiv.org/abs/1911.06904.

Guofeng Cui, Yuning Wang, Wenjie Qiu, and He Zhu. Reward-guided synthesis of intelligent agents with control structures. *Proc. ACM Program. Lang.*, 8(PLDI), June 2024. doi: 10.1145/3656447. URL https://doi.org/10.1145/3656447.

Thao Dang, Alexandre Donzé, Inzemamul Haque, Nikolaos Kekatos, and Indranil Saha. Counter-example guided imitation learning of feedback controllers from temporal logic specifications. In *2023 62nd IEEE Conference on Decision and Control (CDC)*, pp. 5339–5344, 2023. doi: 10.1109/CDC49753.2023.10383831.

Moumita Das, Rajarshi Ray, Swarup Kumar Mohalik, and Ansuman Banerjee. Fast falsification of neural networks using property directed testing, 2021. URL https://arxiv.org/abs/2104.12418.

Charles Dawson and Chuchu Fan. Robust counterexample-guided optimization for planning from differentiable temporal logic. *IROS*, 2022.

Charles Dawson, Sicun Gao, and Chuchu Fan. Safe control with learned certificates: A survey of neural lyapunov, barrier, and contraction methods for robotics and control. *IEEE Transactions on Robotics*, pp. 1–19, 2023. doi: 10.1109/TRO.2022.3232542.

Rayna Dimitrova and Rupak Majumdar. Deductive control synthesis for alternating-time logics. In *2014 International Conference on Embedded Software (EMSOFT)*, pp. 1–10, 2014. doi: 10.1145/2656045.2656054.

Xu Chu Ding, Stephen L Smith, Calin Belta, and Daniela Rus. Mdp optimal control under temporal logic constraints. In *2011 50th IEEE Conference on Decision and Control and European Control Conference*, pp. 532–538. IEEE, 2011.

Taylor Dohmen, Noah Topper, George Atia, Andre Beckus, Ashutosh Trivedi, and Alvaro Velasquez. Inferring Probabilistic Reward Machines from Non-Markovian Reward Signals for Reinforcement Learning. *Proceedings of the International Conference on Automated Planning and Scheduling*, 32(1):574–582, June 2022. doi: 10.1609/icaps.v32i1.19844. URL https://ojs.aaai.org/index.php/ICAPS/article/view/19844.

Adel Dokhanchi, Heni Ben Amor, Jyotirmoy V. Deshmukh, and Georgios Fainekos. Evaluating perception systems for autonomous vehicles using quality temporal logic. In Christian Colombo and Martin Leucker (eds.), *Runtime Verification- 18th International Conference, RV 2018, Proceedings*, Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), pp. 409–416. Springer Verlag, 2019. ISBN 9783030037680. doi: 10.1007/978-3-030-03769-7_23.

Alexandre Donzé and Oded Maler. Systematic simulation using sensitivity analysis. In Alberto Bemporad, Antonio Bicchi, and Giorgio Buttazzo (eds.), *Hybrid Systems: Computation and Control*, pp. 174–189, Berlin, Heidelberg, 2007. Springer Berlin Heidelberg. ISBN 978-3-540-71493-4.

Tommaso Dreossi, Alexandre Donzé, and Sanjit A. Seshia. Compositional Falsification of Cyber-Physical Systems with Machine Learning Components. *Journal of Automated Reasoning*, 63(4):1031–1053, December 2019. ISSN 1573-0670. doi: 10.1007/s10817-018-09509-5. URL https://doi.org/10.1007/s10817-018-09509-5.

Parasara Sridhar Duggirala, Sayan Mitra, and Mahesh Viswanathan. Verification of annotated models from executions. In *2013 Proceedings of the International Conference on Embedded Software (EMSOFT)*, pp. 1–10, 2013. doi: 10.1109/EMSOFT.2013.6658604.

Parasara Sridhar Duggirala, Sayan Mitra, Mahesh Viswanathan, and Matthew Potok. C2e2: A verification tool for stateflow models. In Christel Baier and Cesare Tinelli (eds.), *Tools and Algorithms for the Construction and Analysis of Systems*, pp. 68–82, Berlin, Heidelberg, 2015. Springer Berlin Heidelberg. ISBN 978-3-662-46681-0.

Souradeep Dutta, Xin Chen, and Sriram Sankaranarayanan. Reachability analysis for neural feedback systems using regressive polynomial rule inference. In *Proceedings of the 22nd ACM International Conference on Hybrid Systems: Computation and Control*, HSCC '19, pp. 157–168, New York, NY, USA, 2019. Association for Computing Machinery. ISBN 9781450362825. doi: 10.1145/3302504.3311807. URL https://doi.org/10.1145/3302504.3311807.

Krishnamurthy Dj Dvijotham, Robert Stanforth, Sven Gowal, Timothy A. Mann, and Pushmeet Kohli. A dual approach to scalable verification of deep networks. In *Conference on Uncertainty in Artificial Intelligence*, 2018. URL https://api.semanticscholar.org/CorpusID:3972365.

Joe Eappen, Zikang Xiong, Dipam Patel, Aniket Bera, and Suresh Jagannathan. Scaling safe multi-agent control for signal temporal logic specifications. In *8th Annual Conference on Robot Learning*, 2024. URL https://openreview.net/forum?id=N1K4B8N3n1.

Alec Edwards, Andrea Peruffo, and Alessandro Abate. Fossil 2.0: Formal certificate synthesis for the verification and control of dynamical models. In *Proceedings of the 27th ACM International Conference on Hybrid Systems: Computation and Control*, HSCC '24, New York, NY, USA, 2024. Association for Computing Machinery. ISBN 9798400705229. doi: 10.1145/3641513.3651398. URL https://doi.org/10.1145/3641513.3651398.

Rüdiger Ehlers. Symbolic bounded synthesis. In Tayssir Touili, Byron Cook, and Paul Jackson (eds.), *Computer Aided Verification*, pp. 365–379, Berlin, Heidelberg, 2010. Springer Berlin Heidelberg. ISBN 978-3-642-14295-6.

Yousef Emam, Gennaro Notomista, Paul Glotfelter, Zsolt Kira, and Magnus Egerstedt. Safe reinforcement learning using robust control barrier functions. *IEEE Robotics and Automation Letters*, 10(3):2886–2893, 2025. doi: 10.1109/LRA.2022.3216996.

Wilbert Peter Empleo, Yitaek Kim, Hansoul Kim, Thiusius Rajeeth Savarimuthu, and Iñigo Iturrate. Safe uncertainty-aware learning of robotic suturing, 2025. URL https://arxiv.org/abs/2505.16596.

Georgios E. Fainekos, Hadas. Kress-Gazit, and George J. Pappas. Temporal logic motion planning for mobile robots. In *Proceedings of the 2005 IEEE International Conference on Robotics and Automation*, pp. 2020–2025, 2005. doi: 10.1109/ROBOT.2005.1570410.

Chuchu Fan, Bolun Qi, Sayan Mitra, Mahesh Viswanathan, and Parasara Sridhar Duggirala. Automatic Reachability Analysis for Nonlinear Hybrid Models with C2E2. In Swarat Chaudhuri and Azadeh Farzan (eds.), *Computer Aided Verification*, pp. 531–538, Cham, 2016. Springer International Publishing. ISBN 978-3-319-41528-4.

Zhen Fang, Yixuan Li, Feng Liu, Bo Han, and Jie Lu. On the Learnability of Out-of-distribution Detection. *Journal of Machine Learning Research*, 25(84):1–83, 2024. URL http://jmlr.org/papers/v25/23-1257.html.

Francesca Favaro, Laura Fraade-Blanar, Scott Schnelle, Trent Victor, Mauricio Peña, Johan Engstrom, John Scanlon, Kris Kusano, and Dan Smith. Building a credible case for safety: Waymo's approach for the determination of absence of unreasonable risk, 2023. URL https://arxiv.org/abs/2306.01917.

Mahyar Fazlyab, Manfred Morari, and George J. Pappas. Safety verification and robustness analysis of neural networks via quadratic constraints and semidefinite programming, 2021. URL https://arxiv.org/abs/1903.01287.

Zeyu Feng, Hao Luan, Pranav Goyal, and Harold Soh. Ltldog: Satisfying temporally-extended symbolic constraints for safe diffusion-based planning. *IEEE Robotics and Automation Letters*, 9(10):8571–8578, 2024a. doi: 10.1109/LRA.2024.3443501.

Zeyu Feng, Hao Luan, Kevin Yuchen Ma, and Harold Soh. Diffusion meets options: Hierarchical generative skill composition for temporally-extended tasks. *arXiv preprint arXiv:2410.02389*, 2024b.

John K. Feser, Swarat Chaudhuri, and Isil Dillig. Synthesizing data structure transformations from input-output examples. *SIGPLAN Not.*, 50(6):229–239, June 2015. ISSN 0362-1340. doi: 10.1145/2813885.2737977. URL https://doi.org/10.1145/2813885.2737977.

Nathanaël Fijalkow, Joël Ouaknine, Amaury Pouly, João Sousa-Pinto, and James Worrell. On the decidability of reachability in linear time-invariant systems. In *Proceedings of the 22nd ACM International Conference on Hybrid Systems: Computation and Control*, HSCC '19, pp. 77–86, New York, NY, USA, 2019. Association for Computing Machinery. ISBN 9781450362825. doi: 10.1145/3302504.3311796. URL https://doi.org/10.1145/3302504.3311796.

Roya Firoozi, Johnathan Tucker, Stephen Tian, Anirudha Majumdar, Jiankai Sun, Weiyu Liu, Yuke Zhu, Shuran Song, Ashish Kapoor, Karol Hausman, et al. Foundation models in robotics: Applications, challenges, and the future. *The International Journal of Robotics Research*, 2024. doi: https://doi.org/10.1177/02783649241281508.

Jaime F. Fisac, Neil F. Lugovoy, Vicenç Rubies-Royo, Shromona Ghosh, and Claire J. Tomlin. Bridging hamilton-jacobi safety analysis and reinforcement learning. In *2019 International Conference on Robotics and Automation (ICRA)*, pp. 8550–8556, 2019. doi: 10.1109/ICRA.2019.8794107.

Goran Frehse, Colas Le Guernic, Alexandre Donzé, Scott Cotton, Rajarshi Ray, Olivier Lebeltel, Rodolfo Ripado, Antoine Girard, Thao Dang, and Oded Maler. Spaceex: Scalable verification of hybrid systems. In Ganesh Gopalakrishnan and Shaz Qadeer (eds.), *Computer Aided Verification*, pp. 379–395, Berlin, Heidelberg, 2011. Springer Berlin Heidelberg. ISBN 978-3-642-22110-1.

Jie Fu and Ufuk Topcu. Probably approximately correct mdp learning and control with temporal logic constraints. In *Proceedings of Robotics: Science and Systems*, Berkeley, USA, July 2014. doi: 10.15607/RSS.2014.X.039.

Nathan Gaby, Fumin Zhang, and Xiaojing Ye. Lyapunov-net: A deep neural network architecture for lyapunov function approximation. In *2022 IEEE 61st Conference on Decision and Control (CDC)*, pp. 2091–2096, 2022. doi: 10.1109/CDC51059.2022.9993006.

Maor Gaon and Ronen Brafman. Reinforcement Learning with Non-Markovian Rewards. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(04):3980–3987, April 2020. ISSN 2374-3468. doi: 10.1609/aaai.v34i04.5814. URL https://ojs.aaai.org/index.php/AAAI/article/view/5814. Number: 04.

Paul Gastin and Denis Oddoux. Fast LTL to büchi automata translation. In Gérard Berry, Hubert Comon, and Alain Finkel (eds.), *Computer Aided Verification*, pp. 53–65, Berlin, Heidelberg, 2001. Springer Berlin Heidelberg. ISBN 978-3-540-44585-2.

Timon Gehr, Matthew Mirman, Dana Drachsler-Cohen, Petar Tsankov, Swarat Chaudhuri, and Martin Vechev. Ai2: Safety and robustness certification of neural networks with abstract interpretation. In *2018 IEEE Symposium on Security and Privacy (SP)*, pp. 3–18, 2018. doi: 10.1109/SP.2018.00058.

Shromona Ghosh, Yash Vardhan Pant, Hadi Ravanbakhsh, and Sanjit A. Seshia. Counterexample-guided synthesis of perception models and control. In *2021 American Control Conference (ACC)*, pp. 3447–3454, 2021. doi: 10.23919/ACC50511.2021.9482896.

Antoine Girard and George J. Pappas. Verification using simulation. In João P. Hespanha and Ashish Tiwari (eds.), *Hybrid Systems: Computation and Control*, pp. 272–286, Berlin, Heidelberg, 2006. Springer Berlin Heidelberg. ISBN 978-3-540-33171-1.

Ebru Aydin Gol, Ezio Bartocci, and Calin Belta. A formal methods approach to pattern synthesis in reaction diffusion systems. In *53rd IEEE Conference on Decision and Control*, pp. 108–113, 2014. doi: 10.1109/CDC.2014.7039367.

Ian Goodfellow, Yoshua Bengio, Aaron Courville, and Yoshua Bengio. *Deep learning*, volume 1. MIT Press, 2016.

Zijian Guo, Weichao Zhou, and Wenchao Li. Temporal logic specification-conditioned decision transformer for offline safe reinforcement learning. In *ICML*, 2024. URL https://openreview.net/forum?id=7bg10Jj3bG.

Iman Haghighi, Austin Jones, Zhaodan Kong, Ezio Bartocci, Radu Gros, and Calin Belta. Spatel: a novel spatial-temporal logic and its applications to networked systems. In *Proceedings of the 18th International Conference on Hybrid Systems: Computation and Control*, HSCC '15, pp. 189–198, New York, NY, USA, 2015. Association for Computing Machinery. ISBN 9781450334334. doi: 10.1145/2728606.2728633. URL https://doi.org/10.1145/2728606.2728633.

Ernst Moritz Hahn, Mateo Perez, Sven Schewe, Fabio Somenzi, Ashutosh Trivedi, and Dominik Wojtczak. Omega-Regular Objectives in Model-Free Reinforcement Learning. In Tomáš Vojnar and Lijun Zhang (eds.), *Tools and Algorithms for the Construction and Analysis of Systems*, pp. 395–412, Cham, 2019. Springer International Publishing. ISBN 978-3-030-17462-0.

Ernst Moritz Hahn, Mateo Perez, Sven Schewe, Fabio Somenzi, Ashutosh Trivedi, and Dominik Wojtczak. Good-for-mdps automata for probabilistic analysis and reinforcement learning. In Armin Biere and David Parker (eds.), *Tools and Algorithms for the Construction and Analysis of Systems*, pp. 306–323, Cham, 2020. Springer International Publishing. ISBN 978-3-030-45190-5.

Ernst Moritz Hahn, Mateo Perez, Sven Schewe, Fabio Somenzi, Ashutosh Trivedi, and Dominik Wojtczak. An impossibility result in automata-theoretic reinforcement learning. In Ahmed Bouajjani, Lukáš Holík, and Zhilin Wu (eds.), *Automated Technology for Verification and Analysis*, pp. 42–57, Cham, 2022. Springer International Publishing. ISBN 978-3-031-19992-9.

Mohammadhosein Hasanbeig, Alessandro Abate, and Daniel Kroening. Logically-constrained reinforcement learning, 2019a. URL https://arxiv.org/abs/1801.08099.

Mohammadhosein. Hasanbeig, Yiannis. Kantaros, Alessandro. Abate, Daniel. Kroening, George J. Pappas, and Insup. Lee. Reinforcement learning for temporal logic control synthesis with probabilistic satisfaction guarantees. In *2019 IEEE 58th Conference on Decision and Control (CDC)*, pp. 5338–5343, 2019b. doi: 10.1109/CDC40024.2019.9028919.

Navid Hashemi, Bardh Hoxha, Tomoya Yamaguchi, Danil Prokhorov, Georgios Fainekos, and Jyotirmoy Deshmukh. A neurosymbolic approach to the verification of temporal logic properties of learning-enabled control systems. In *Proceedings of the ACM/IEEE 14th International Conference on Cyber-Physical Systems (with CPS-IoT Week 2023)*, ICCPS '23, pp. 98–109, New York, NY, USA, 2023. Association for Computing Machinery. ISBN 9798400700361. doi: 10.1145/3576841.3585928. URL https://doi.org/10.1145/3576841.3585928.

Wataru Hashimoto, Kazumune Hashimoto, and Shigemasa Takai. Stl2vec: Signal temporal logic embeddings for control synthesis with recurrent neural networks. *IEEE Robotics and Automation Letters*, 7(2):5246–5253, 2022. doi: 10.1109/LRA.2022.3155197.

Mohammad Hekmatnejad. *Formalizing Safety, Perception, and Mission Requirements for Testing and Planning in Autonomous Vehicles*. PhD thesis, Arizona State University, USA, 2021.

Matanya B. Horowitz, Eric M. Wolff, and Richard M. Murray. A compositional approach to stochastic optimal control with co-safe temporal logic specifications. In *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 1466–1473, 2014. doi: 10.1109/IROS.2014.6942750.

Bardh Hoxha, Adel Dokhanchi, and Georgios Fainekos. Mining parametric temporal logic properties in model-based design for cyber-physical systems. *International Journal on Software Tools for Technology Transfer*, 20(1):79–93, February 2018. ISSN 1433-2787. doi: 10.1007/s10009-017-0447-4. URL https://doi.org/10.1007/s10009-017-0447-4.

Kai-Chieh Hsu, Vicenç Rubies-Royo, Claire J. Tomlin, and Jaime F. Fisac. Safety and liveness guarantees through reach-avoid reinforcement learning. In *Proceedings of Robotics: Science and Systems*, Held Virtually, July 2021. doi: 10.15607/RSS.2021.XVII.077.

Yafei Hu, Quanting Xie, Vidhi Jain, Jonathan Francis, Jay Patrikar, Nikhil Keetha, Seungchan Kim, Yaqi Xie, Tianyi Zhang, Hao-Shu Fang, Shibo Zhao, Shayegan Omidshafiei, Dong-Ki Kim, Ali akbar Agha-mohammadi, Katia Sycara, Matthew Johnson-Roberson, Dhruv Batra, Xiaolong Wang, Sebastian Scherer, Chen Wang, Zsolt Kira, Fei Xia, and Yonatan Bisk. Toward general-purpose robots via foundation models: A survey and meta-analysis, 2024. URL https://arxiv.org/abs/2312.08782.

Chao Huang, Jiameng Fan, Wenchao Li, Xin Chen, and Qi Zhu. Reachnn: Reachability analysis of neural-network controlled systems. *ACM Trans. Embed. Comput. Syst.*, 18(5s), October 2019. ISSN 1539-9087. doi: 10.1145/3358228. URL https://doi.org/10.1145/3358228.

Ahmed Hussein, Mohamed Medhat Gaber, Eyad Elyan, and Chrisina Jayne. Imitation learning: A survey of learning methods. *ACM Comput. Surv.*, 50(2), April 2017. ISSN 0360-0300. doi: 10.1145/3054912. URL https://doi.org/10.1145/3054912.

Rodrigo Toro Icarte, Toryn Klassen, Richard Valenzano, and Sheila McIlraith. Using reward machines for high-level task specification and decomposition in reinforcement learning. In Jennifer Dy and Andreas Krause (eds.), *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 2107–2116. PMLR, 10–15 Jul 2018. URL `https://proceedings.mlr.press/v80/icarte18a.html`.

Jeevana Priya Inala, Osbert Bastani, Zenna Tavares, and Armando Solar-Lezama. Synthesizing programmatic policies that inductively generalize. In *International Conference on Learning Representations*, 2020. URL `https://openreview.net/forum?id=S1l8oANFDH`.

Craig Innes and Subramanian Ramamoorthy. Elaborating on Learned Demonstrations with Temporal Logic Specifications. In *Proceedings of Robotics: Science and Systems*, Corvalis, Oregon, USA, July 2020. doi: 10.15607/RSS.2020.XVI.004.

Radoslav Ivanov, James Weimer, Rajeev Alur, George J. Pappas, and Insup Lee. Verisig: verifying safety properties of hybrid systems with neural network controllers. In *Proceedings of the 22nd ACM International Conference on Hybrid Systems: Computation and Control*, HSCC '19, pp. 169–178, New York, NY, USA, 2019. Association for Computing Machinery. ISBN 9781450362825. doi: 10.1145/3302504.3311806. URL `https://doi.org/10.1145/3302504.3311806`.

Radoslav Ivanov, Taylor J. Carpenter, James Weimer, Rajeev Alur, George J. Pappas, and Insup Lee. Verifying the safety of autonomous systems with neural network controllers. *ACM Trans. Embed. Comput. Syst.*, 20(1), December 2020. ISSN 1539-9087. doi: 10.1145/3419742. URL `https://doi.org/10.1145/3419742`.

Radoslav Ivanov, Taylor Carpenter, James Weimer, Rajeev Alur, George Pappas, and Insup Lee. Verisig 2.0: Verification of neural network controllers using taylor model preconditioning. In Alexandra Silva and K. Rustan M. Leino (eds.), *Computer Aided Verification*, pp. 249–262, Cham, 2021. Springer International Publishing. ISBN 978-3-030-81685-8.

Susmit Jha, Ashish Tiwari, Sanjit A. Seshia, Tuhin Sahai, and Natarajan Shankar. TeLEx: learning signal temporal logic from positive examples using tightness metric. *Formal Methods in System Design*, 54(3): 364–387, November 2019. ISSN 1572-8102. doi: 10.1007/s10703-019-00332-1. URL `https://doi.org/10.1007/s10703-019-00332-1`.

Xiaoqing Jin, Alexandre Donzé, Jyotirmoy V. Deshmukh, and Sanjit A. Seshia. Mining requirements from closed-loop control models. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 34(11):1704–1717, 2015. doi: 10.1109/TCAD.2015.2421907.

Austin Jones, Zhaodan Kong, and Calin Belta. Anomaly detection in cyber-physical systems: A formal methods approach. In *53rd IEEE Conference on Decision and Control*, pp. 848–853, 2014. doi: 10.1109/CDC.2014.7039487.

Kishor Jothimurugan, Rajeev Alur, and Osbert Bastani. A Composable Specification Language for Reinforcement Learning Tasks. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d' Alché-Buc, E. Fox, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL `https://proceedings.neurips.cc/paper_files/paper/2019/file/f5aa4bd09c07d8b2f65bad6c7cd3358f-Paper.pdf`.

Kishor Jothimurugan, Suguman Bansal, Osbert Bastani, and Rajeev Alur. Compositional reinforcement learning from logical specifications. *Advances in Neural Information Processing Systems*, 34:10026–10039, 2021.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B. Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. Scaling laws for neural language models, 2020. URL `https://arxiv.org/abs/2001.08361`.

Parv Kapoor, Anand Balakrishnan, and Jyotirmoy V. Deshmukh. Model-based reinforcement learning from signal temporal logic specifications, 2020. URL `https://arxiv.org/abs/2011.04950`.

Parv Kapoor, Abigail Hammer, Ashish Kapoor, Karen Leung, and Eunsuk Kang. Pretrained embeddings as a behavior specification mechanism, 2025a. URL https://arxiv.org/abs/2503.02012.

Parv Kapoor, Kazuki Mizuta, Eunsuk Kang, and Karen Leung. Stlcg++: A masking approach for differentiable signal temporal logic specification, 2025b. URL https://arxiv.org/abs/2501.04194.

Guy Katz, Clark Barrett, David L. Dill, Kyle Julian, and Mykel J. Kochenderfer. Reluplex: An efficient smt solver for verifying deep neural networks. In Rupak Majumdar and Viktor Kunčak (eds.), *Computer Aided Verification*, pp. 97–117, Cham, 2017. Springer International Publishing. ISBN 978-3-319-63387-9.

J. Zico Kolter and Gaurav Manek. Learning Stable Deep Dynamics Models. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d' Alché-Buc, E. Fox, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper_files/paper/2019/file/0a4bbceda17a6253386bc9eb45240e25-Paper.pdf.

Soonho Kong, Sicun Gao, Wei Chen, and Edmund Clarke. dreach: δ-reachability analysis for hybrid systems. In Christel Baier and Cesare Tinelli (eds.), *Tools and Algorithms for the Construction and Analysis of Systems*, pp. 200–205, Berlin, Heidelberg, 2015. Springer Berlin Heidelberg. ISBN 978-3-662-46681-0.

Zhaodan Kong, Austin Jones, Ana Medina Ayala, Ebru Aydin Gol, and Calin Belta. Temporal logic inference for classification and prediction from data. In *Proceedings of the 17th International Conference on Hybrid Systems: Computation and Control*, HSCC '14, pp. 273–282, New York, NY, USA, 2014. Association for Computing Machinery. ISBN 9781450327329. doi: 10.1145/2562059.2562146. URL https://doi.org/10.1145/2562059.2562146.

Hadas Kress-Gazit, Georgios E. Fainekos, and George J. Pappas. Temporal-logic-based reactive mission and motion planning. *IEEE Transactions on Robotics*, 25(6):1370–1381, 2009. doi: 10.1109/TRO.2009.2030225.

Yen-Ling Kuo, Boris Katz, and Andrei Barbu. Encoding formulas as deep networks: Reinforcement learning for zero-shot execution of ltl formulas. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 5604–5610, 2020. doi: 10.1109/IROS45743.2020.9341325.

Daniel Kästner, Ulrich Wünsche, Jörg Barrho, Marc Schlickling, Bernhard Schommer, Michael Schmidt, Christian Ferdinand, Xavier Leroy, and Sandrine Blazy. CompCert: Practical experience on integrating and qualifying a formally verified optimizing compiler. In *ERTS 2018: Embedded Real Time Software and Systems*. SEE, January 2018. URL http://xavierleroy.org/publi/erts2018_compcert.pdf.

Tobias Ladner and Matthias Althoff. Automatic abstraction refinement in neural network verification using sensitivity analysis. In *Proceedings of the 26th ACM International Conference on Hybrid Systems: Computation and Control*, HSCC '23, New York, NY, USA, 2023. Association for Computing Machinery. ISBN 9798400700330. doi: 10.1145/3575870.3587129. URL https://doi.org/10.1145/3575870.3587129.

Luís C. Lamb, Artur d'Avila Garcez, Marco Gori, Marcelo O.R. Prates, Pedro H.C. Avelar, and Moshe Y. Vardi. Graph neural networks meet neural-symbolic computing: a survey and perspective. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence*, IJCAI'20, 2021. ISBN 9780999241165.

Kim Larsen, Axel Legay, Gerrit Nolte, Maximilian Schlüter, Marielle Stoelinga, and Bernhard Steffen. Formal methods meet machine learning (f3ml). In *Leveraging Applications of Formal Methods, Verification and Validation. Adaptation and Learning: 11th International Symposium, ISoLA 2022, Rhodes, Greece, October 22–30, 2022, Proceedings, Part III*, pp. 393–405, Berlin, Heidelberg, 2022. Springer-Verlag. ISBN 978-3-031-19758-1. doi: 10.1007/978-3-031-19759-8_24. URL https://doi.org/10.1007/978-3-031-19759-8_24.

Aless Lasaruk and Thomas Sturm. Effective quantifier elimination for presburger arithmetic with infinity. In Vladimir P. Gerdt, Ernst W. Mayr, and Evgenii V. Vorozhtsov (eds.), *Computer Algebra in Scientific Computing*, pp. 195–212, Berlin, Heidelberg, 2009. Springer Berlin Heidelberg. ISBN 978-3-642-04103-7.

Karen Leung, Nikos Aréchiga, and Marco Pavone. Backpropagation for parametric stl. In *2019 IEEE Intelligent Vehicles Symposium (IV)*, pp. 185–192, June 2019. doi: 10.1109/IVS.2019.8814167.

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. doi: 10.1177/02783649221082115. URL https://doi.org/10.1177/02783649221082115.

Sergey Levine, Chelsea Finn, Trevor Darrell, and Pieter Abbeel. End-to-End Training of Deep Visuomotor Policies. *Journal of Machine Learning Research*, 17(39):1–40, 2016. URL http://jmlr.org/papers/v17/15-522.html.

Sergey Levine, Peter Pastor, Alex Krizhevsky, Julian Ibarz, and Deirdre Quillen. Learning hand-eye coordination for robotic grasping with deep learning and large-scale data collection. *The International journal of robotics research*, 37(4-5):421–436, 2018.

Sergey Levine, Aviral Kumar, George Tucker, and Justin Fu. Offline reinforcement learning: Tutorial, review, and perspectives on open problems, 2020. URL https://arxiv.org/abs/2005.01643.

Thomas Lew and Marco Pavone. Sampling-based reachability analysis: A random set theory approach with adversarial sampling. In Jens Kober, Fabio Ramos, and Claire Tomlin (eds.), *Proceedings of the 2020 Conference on Robot Learning*, volume 155 of *Proceedings of Machine Learning Research*, pp. 2055–2070. PMLR, 16–18 Nov 2021. URL https://proceedings.mlr.press/v155/lew21a.html.

Danyang Li, Mingyu Cai, Cristian-Ioan Vasile, and Roberto Tron. Tlinet: Differentiable neural network temporal logic inference, 2024. URL https://arxiv.org/abs/2405.06670.

Jianlin Li, Jiangchao Liu, Pengfei Yang, Liqian Chen, Xiaowei Huang, and Lijun Zhang. *Analyzing Deep Neural Networks with Symbolic Propagation: Towards Higher Precision and Faster Verification*, pp. 296–319. Springer International Publishing, 2019. ISBN 9783030323042. doi: 10.1007/978-3-030-32304-2_15. URL http://dx.doi.org/10.1007/978-3-030-32304-2_15.

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)*, pp. 3834–3839, 2017. doi: 10.1109/IROS.2017.8206234.

Lars Lindemann and Dimos V. Dimarogonas. Control barrier functions for signal temporal logic tasks. *IEEE Control Systems Letters*, 3(1):96–101, 2019. doi: 10.1109/LCSYS.2018.2853182.

Changliu Liu, Tomer Arnon, Christopher Lazarus, Christopher Strong, Clark Barrett, and Mykel J. Kochenderfer. Algorithms for verifying deep neural networks. *Found. Trends Optim.*, 4(3–4):244–404, February 2021. ISSN 2167-3888. doi: 10.1561/2400000035. URL https://doi.org/10.1561/2400000035.

Jason Xinyu Liu, Ziyi Yang, Benjamin Schornstein, Sam Liang, Ifrah Idrees, Stefanie Tellex, and Ankit Shah. Lang2LTL: Translating natural language commands to temporal specification with large language models. In *Workshop on Language and Robotics at CoRL 2022*, 2022. URL https://openreview.net/forum?id=VxfjGZzrdn.

Wenliang Liu, Mirai Nishioka, and Calin Belta. Safe model-based control from signal temporal logic specifications using recurrent neural networks. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 12416–12422, 2023. doi: 10.1109/ICRA48891.2023.10161201.

Wenliang Liu, Danyang Li, Erfan Aasi, Roberto Tron, and Calin Belta. Interpretable imitation learning via generative adversarial stl inference and control. In *2nd International Conference on Neuro-symbolic Systems (NeuS)*, 2025.

Alessio Lomuscio and Lalit Maganti. An approach to reachability analysis for feed-forward relu neural networks, 2017. URL https://arxiv.org/abs/1706.07351.

Diego Manzanas Lopez, Sung Woo Choi, Hoang-Dung Tran, and Taylor T. Johnson. Nnv 2.0: The neural network verification tool. In Constantin Enea and Akash Lal (eds.), *Computer Aided Verification*, pp. 397–412, Cham, 2023. Springer Nature Switzerland. ISBN 978-3-031-37703-7.

Matt Luckcuck, Marie Farrell, Louise A. Dennis, Clare Dixon, and Michael Fisher. Formal specification and verification of autonomous robotic systems: A survey. *ACM Comput. Surv.*, 52(5), September 2019. ISSN 0360-0300. doi: 10.1145/3342355. URL https://doi.org/10.1145/3342355.

Haitong Ma, Jianyu Chen, Shengbo Eben, Ziyu Lin, Yang Guan, Yangang Ren, and Sifa Zheng. Model-based constrained reinforcement learning using generalized control barrier function. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 4552–4559, 2021. doi: 10.1109/IROS51168.2021.9636468.

Oded Maler and Dejan Nickovic. Monitoring temporal properties of continuous signals. In Yassine Lakhnech and Sergio Yovine (eds.), *Formal Techniques, Modelling and Analysis of Timed and Fault-Tolerant Systems*, pp. 152–166, Berlin, Heidelberg, 2004. Springer Berlin Heidelberg. ISBN 978-3-540-30206-3.

Zohar Manna and Amir Pnueli. A hierarchy of temporal properties (invited paper, 1989). In *Proceedings of the Ninth Annual ACM Symposium on Principles of Distributed Computing*, PODC '90, pp. 377–410, New York, NY, USA, 1990. Association for Computing Machinery. ISBN 089791404X. doi: 10.1145/93385.93442. URL https://doi.org/10.1145/93385.93442.

Zohar Manna and Richard Waldinger. A deductive approach to program synthesis. *ACM Trans. Program. Lang. Syst.*, 2(1):90–121, January 1980. ISSN 0164-0925. doi: 10.1145/357084.357090. URL https://doi.org/10.1145/357084.357090.

Diego Manzanas Lopez, Patrick Musau, Nathaniel P. Hamilton, and Taylor T. Johnson. Reachability analysis of a general class of neural ordinary differential equations. In Sergiy Bogomolov and David Parker (eds.), *Formal Modeling and Analysis of Timed Systems*, pp. 258–277, Cham, 2022. Springer International Publishing. ISBN 978-3-031-15839-1.

Zahra Marvi and Bahare Kiumarsi. Safe reinforcement learning: A control barrier function optimization approach. *International Journal of Robust and Nonlinear Control*, 31(6):1923–1940, 2021. doi: https://doi.org/10.1002/rnc.5132. URL https://onlinelibrary.wiley.com/doi/abs/10.1002/rnc.5132.

Yue Meng and Chuchu Fan. Signal temporal logic neural predictive control. *IEEE Robotics and Automation Letters*, 8(11):7719–7726, 2023. doi: 10.1109/LRA.2023.3315536.

Yue Meng and Chuchu Fan. Diverse controllable diffusion policy with signal temporal logic. *IEEE Robotics and Automation Letters*, 9(10):8354–8361, 2024a. doi: 10.1109/LRA.2024.3444668.

Yue Meng and Chuchu Fan. Diverse controllable diffusion policy with signal temporal logic. *IEEE Robotics and Automation Letters*, 9(10):8354–8361, 2024b. doi: 10.1109/LRA.2024.3444668.

Yue Meng and Chuchu Fan. Telograf: Temporal logic planning via graph-encoded flow matching. *International Conference on Machine Learning*, 2025.

Philipp J. Meyer, Salomon Sickert, and Michael Luttenberger. Strix: Explicit reactive synthesis strikes back! In Hana Chockler and Georg Weissenbacher (eds.), *Computer Aided Verification*, pp. 578–586, Cham, 2018. Springer International Publishing. ISBN 978-3-319-96145-3.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed Representations of Words and Phrases and their Compositionality. In C. J. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger (eds.), *Advances in Neural Information Processing Systems*, volume 26. Curran Associates, Inc., 2013. URL https://proceedings.neurips.cc/paper_files/paper/2013/file/9aa42b31882ec039965f3c4923ce901b-Paper.pdf.

Ian M. Mitchell. Comparing forward and backward reachability as tools for safety analysis. In Alberto Bemporad, Antonio Bicchi, and Giorgio Buttazzo (eds.), *Hybrid Systems: Computation and Control*, pp. 428–443, Berlin, Heidelberg, 2007. Springer Berlin Heidelberg. ISBN 978-3-540-71493-4.

I.M. Mitchell, A.M. Bayen, and C.J. Tomlin. A time-dependent hamilton-jacobi formulation of reachable sets for continuous dynamic games. *IEEE Transactions on Automatic Control*, 50(7):947–957, 2005. doi: 10.1109/TAC.2005.851439.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A. Rusu, Joel Veness, Marc G. Bellemare, Alex Graves, Martin Riedmiller, Andreas K. Fidjeland, Georg Ostrovski, Stig Petersen, Charles Beattie, Amir Sadik, Ioannis Antonoglou, Helen King, Dharshan Kumaran, Daan Wierstra, Shane Legg, and Demis Hassabis. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533, February 2015. ISSN 1476-4687. doi: 10.1038/nature14236. URL `https://doi.org/10.1038/nature14236`.

Sara Mohammadinejad, Jyotirmoy V. Deshmukh, Aniruddh G. Puranic, Marcell Vazquez-Chanlatte, and Alexandre Donzé. Interpretable classification of time-series data using efficient enumerative techniques. In *Proceedings of the 23rd International Conference on Hybrid Systems: Computation and Control*, HSCC '20, New York, NY, USA, 2020. Association for Computing Machinery. ISBN 9781450370189. doi: 10.1145/3365365.3382218. URL `https://doi.org/10.1145/3365365.3382218`.

Toby Murray, Daniel Matichuk, Matthew Brassil, Peter Gammie, Timothy Bourke, Sean Seefried, Corey Lewis, Xin Gao, and Gerwin Klein. sel4: From general purpose to a proof of information flow enforcement. In *2013 IEEE Symposium on Security and Privacy*, pp. 415–429, 2013. doi: 10.1109/SP.2013.35.

Geraud Nangue Tasse, Steven James, and Benjamin Rosman. Skill machines: Temporal logic composition in reinforcement learning. In *Deep Reinforcement Learning Workshop @ NeurIPS*, 2022.

Laura Nenzi, Simone Silvetti, Ezio Bartocci, and Luca Bortolussi. A robust genetic algorithm for learning temporal specifications from data. In Annabelle McIver and Andras Horvath (eds.), *Quantitative Evaluation of Systems*, pp. 323–338, Cham, 2018. Springer International Publishing. ISBN 978-3-319-99154-2.

Arnold Neumaier. *The Wrapping Effect, Ellipsoid Arithmetic, Stability and Confidence Regions*, pp. 175–190. Springer Vienna, Vienna, 1993. ISBN 978-3-7091-6918-6. doi: 10.1007/978-3-7091-6918-6_14. URL `https://doi.org/10.1007/978-3-7091-6918-6_14`.

Alexander Novikov, Ngân Vũ, Marvin Eisenberger, Emilien Dupont, Po-Sen Huang, Adam Zsolt Wagner, Sergey Shirobokov, Borislav Kozlovskii, Francisco J. R. Ruiz, Abbas Mehrabian, M. Pawan Kumar, Abigail See, Swarat Chaudhuri, George Holland, Alex Davies, Sebastian Nowozin, Pushmeet Kohli, and Matej Balog. Alphaevolve: A coding agent for scientific and algorithmic discovery, 2025. URL `https://arxiv.org/abs/2506.13131`.

Ivan Papusha, Jie Fu, Ufuk Topcu, and Richard M. Murray. Automata theory meets approximate dynamic programming: Optimal control with temporal logic constraints. In *2016 IEEE 55th Conference on Decision and Control (CDC)*, pp. 434–440, 2016. doi: 10.1109/CDC.2016.7798307.

Kexin Pei, Yinzhi Cao, Junfeng Yang, and Suman Jana. Deepxplore: automated whitebox testing of deep learning systems. *Commun. ACM*, 62(11):137–145, October 2019. ISSN 0001-0782. doi: 10.1145/3361566. URL `https://doi.org/10.1145/3361566`.

Theodore J. Perkins and Andrew G. Barto. Lyapunov design for safe reinforcement learning. *J. Mach. Learn. Res.*, 3(null):803–832, March 2003. ISSN 1532-4435.

Andrea Peruffo, Daniele Ahmed, and Alessandro Abate. Automated and Formal Synthesis of Neural Barrier Certificates for Dynamical Models. In Jan Friso Groote and Kim Guldstrand Larsen (eds.), *Tools and Algorithms for the Construction and Analysis of Systems*, pp. 370–388, Cham, 2021. Springer International Publishing. ISBN 978-3-030-72016-2.

Amir Pnueli. The temporal logic of programs. In *18th Annual Symposium on Foundations of Computer Science (sfcs 1977)*, pp. 46–57, 1977. doi: 10.1109/SFCS.1977.32.

Amir Pnueli and Roni Rosner. A framework for the synthesis of reactive modules. In Frederich H. Vogt (ed.), *CONCURRENCY 88*, pp. 4–17, Berlin, Heidelberg, 1988. Springer Berlin Heidelberg. ISBN 978-3-540-45999-6.

Luca Pulina and Armando Tacchella. An abstraction-refinement approach to verification of artificial neural networks. In *Computer Aided Verification: 22nd International Conference, CAV 2010, Edinburgh, UK, July 15-19, 2010. Proceedings 22*, pp. 243–257. Springer, 2010.

Aniruddh Puranic, Jyotirmoy Deshmukh, and Stefanos Nikolaidis. Learning from demonstrations using signal temporal logic. In Jens Kober, Fabio Ramos, and Claire Tomlin (eds.), *Proceedings of the 2020 Conference on Robot Learning*, volume 155 of *Proceedings of Machine Learning Research*, pp. 2228–2242. PMLR, 16–18 Nov 2021. URL https://proceedings.mlr.press/v155/puranic21a.html.

Wenjie Qiu and He Zhu. Programmatic reinforcement learning without oracles. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=6Tk2noBdvxt.

Wenjie Qiu, Wensen Mao, and He Zhu. Instructing goal-conditioned reinforcement learning agents with temporal logic objectives. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine (eds.), *Advances in Neural Information Processing Systems*, volume 36, pp. 39147–39175. Curran Associates, Inc., 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/7b35a69f434b5eb07ed1b1ef16ace52c-Paper-Conference.pdf.

Aditi Raghunathan, Jacob Steinhardt, and Percy Liang. Certified defenses against adversarial examples. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=Bys4ob-Rb.

Vasumathi Raman, Mehdi Maasoumy, and Alexandre Donzé. Model predictive control from signal temporal logic specifications: a case study. In *Proceedings of the 4th ACM SIGBED International Workshop on Design, Modeling, and Evaluation of Cyber-Physical Systems*, CyPhy '14, pp. 52–55, New York, NY, USA, 2014. Association for Computing Machinery. ISBN 9781450328715. doi: 10.1145/2593458.2593472. URL https://doi.org/10.1145/2593458.2593472.

Nacim Ramdani, Nacim Meslem, and Yves Candau. Reachability analysis of uncertain nonlinear systems using guaranteed set integration. *IFAC Proceedings Volumes*, 41(2):8972–8977, 2008. ISSN 1474-6670. doi: https://doi.org/10.3182/20080706-5-KR-1001.01515. URL https://www.sciencedirect.com/science/article/pii/S1474667016403940. 17th IFAC World Congress.

Scott Reed, Konrad Zolna, Emilio Parisotto, Sergio Gómez Colmenarejo, Alexander Novikov, Gabriel Barth-maron, Mai Giménez, Yury Sulsky, Jackie Kay, Jost Tobias Springenberg, Tom Eccles, Jake Bruce, Ali Razavi, Ashley Edwards, Nicolas Heess, Yutian Chen, Raia Hadsell, Oriol Vinyals, Mahyar Bordbar, and Nando de Freitas. A generalist agent. *Transactions on Machine Learning Research*, 2022. ISSN 2835-8856. URL https://openreview.net/forum?id=1ikK0kHjvj. Featured Certification, Outstanding Certification.

Gavin Rens, Jean-François Raskin, Raphaël Reynouad, and Giuseppe Marra. Online learning of non-markovian reward models, 2020. URL https://arxiv.org/abs/2009.12600.

Werner C. Rheinboldt and Azaria Paz. *Introduction to probabilistic automata.* Academic Press, 2014.

Spencer M. Richards, Felix Berkenkamp, and Andreas Krause. The lyapunov neural network: Adaptive stability certification for safe learning of dynamical systems. In Aude Billard, Anca Dragan, Jan Peters, and Jun Morimoto (eds.), *Proceedings of The 2nd Conference on Robot Learning*, volume 87 of *Proceedings of Machine Learning Research*, pp. 466–476. PMLR, 29–31 Oct 2018. URL https://proceedings.mlr.press/v87/richards18a.html.

Dorsa Sadigh and Ashish Kapoor. Safe control under uncertainty with probabilistic signal temporal logic. In *Proceedings of Robotics: Science and Systems XII*, June 2016. URL https://www.microsoft.com/en-us/research/publication/safe-control-uncertainty-probabilistic-signal-temporal-logic/.

Dorsa Sadigh, Eric S. Kim, Samuel Coogan, S. Shankar Sastry, and Sanjit A. Seshia. A learning based approach to control synthesis of markov decision processes for linear temporal logic specifications. In *53rd IEEE Conference on Decision and Control*, pp. 1091–1096, 2014. doi: 10.1109/CDC.2014.7039527.

Ali Salamati, Sadegh Soudjani, and Majid Zamani. Data-driven verification under signal temporal logic constraints. *IFAC-PapersOnLine*, 53(2):69–74, 2020. ISSN 2405-8963. doi: https://doi.org/10.1016/j.ifacol.2020.12.051. URL https://www.sciencedirect.com/science/article/pii/S2405896320303050. 21st IFAC World Congress.

Gaia Saveri, Laura Nenzi, Luca Bortolussi, and Jan Kretínský. stl2vec: Semantic and interpretable vector representation of temporal logic. In Ulle Endriss, Francisco S. Melo, Kerstin Bach, Alberto José Bugarín Diz, Jose Maria Alonso-Moral, Senén Barro, and Fredrik Heintz (eds.), *ECAI 2024 - 27th European Conference on Artificial Intelligence, 19-24 October 2024, Santiago de Compostela, Spain - Including 13th Conference on Prestigious Applications of Intelligent Systems (PAIS 2024)*, volume 392 of *Frontiers in Artificial Intelligence and Applications*, pp. 1381–1388. IOS Press, 2024. ISBN 978-1-64368-548-9. doi: 10.3233/FAIA240638. URL https://doi.org/10.3233/FAIA240638.

Sanjit A. Seshia, Ankush Desai, Tommaso Dreossi, Daniel J. Fremont, Shromona Ghosh, Edward Kim, Sumukh Shivakumar, Marcell Vazquez-Chanlatte, and Xiangyu Yue. Formal specification for deep neural networks. In Shuvendu K. Lahiri and Chao Wang (eds.), *Automated Technology for Verification and Analysis*, pp. 20–34, Cham, 2018. Springer International Publishing. ISBN 978-3-030-01090-4.

Ankit Shah, Pritish Kamath, Julie A Shah, and Shen Li. Bayesian inference of temporal task specifications from demonstrations. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018. URL https://proceedings.neurips.cc/paper_files/paper/2018/file/13168e6a2e6c84b4b7de9390c0ef5ec5-Paper.pdf.

Zhouxing Shi, Huan Zhang, Kai-Wei Chang, Minlie Huang, and Cho-Jui Hsieh. Robustness verification for transformers. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=BJxwPJHFwS.

Yasser Shoukry, Pierluigi Nuzzo, Alberto L. Sangiovanni-Vincentelli, Sanjit A. Seshia, George J. Pappas, and Paulo Tabuada. Smc: Satisfiability modulo convex optimization. In *Proceedings of the 20th International Conference on Hybrid Systems: Computation and Control*, HSCC '17, pp. 19–28, New York, NY, USA, 2017. Association for Computing Machinery. ISBN 9781450345903. doi: 10.1145/3049797.3049819. URL https://doi.org/10.1145/3049797.3049819.

Salomon Sickert, Javier Esparza, Stefan Jaax, and Jan Křetínský. Limit-deterministic büchi automata for linear temporal logic. In Swarat Chaudhuri and Azadeh Farzan (eds.), *Computer Aided Verification*, pp. 312–332, Cham, 2016. Springer International Publishing. ISBN 978-3-319-41540-6.

Gagandeep Singh, Timon Gehr, Matthew Mirman, Markus Püschel, and Martin Vechev. Fast and effective robustness certification. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018. URL https://proceedings.neurips.cc/paper_files/paper/2018/file/f2f446980d8e971ef3da97af089481c3-Paper.pdf.

Gagandeep Singh, Rupanshu Ganvir, Markus Püschel, and Martin Vechev. Beyond the single neuron convex barrier for neural network certification. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019a. URL https://proceedings.neurips.cc/paper_files/paper/2019/file/0a9fdbb17feb6ccb7ec405cfb85222c4-Paper.pdf.

Gagandeep Singh, Timon Gehr, Markus Püschel, and Martin Vechev. An abstract domain for certifying neural networks. *Proc. ACM Program. Lang.*, 3(POPL), January 2019b. doi: 10.1145/3290354. URL https://doi.org/10.1145/3290354.

Gagandeep Singh, Timon Gehr, Markus Püschel, and Martin Vechev. Robustness certification with refinement. In *International Conference on Learning Representations*, 2019c. URL https://openreview.net/forum?id=HJgeEh09KQ.

Vincent Sitzmann, Julien N.P. Martel, Alexander W. Bergman, David B. Lindell, and Gordon Wetzstein. Implicit neural representations with periodic activation functions. In *Proc. NeurIPS*, 2020.

Joar Skalse, Nikolaus H. R. Howe, Dmitrii Krasheninnikov, and David Krueger. Defining and characterizing reward hacking. In *Proceedings of the 36th International Conference on Neural Information Processing Systems*, NIPS '22, Red Hook, NY, USA, 2022. Curran Associates Inc. ISBN 9781713871088.

Stephen L Smith, Jana Tůmová, 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. doi: 10.1177/0278364911417911. URL `https://doi.org/10.1177/0278364911417911`.

Armando Solar-Lezama. The Sketching Approach to Program Synthesis. In Zhenjiang Hu (ed.), *Programming Languages and Systems*, pp. 4–13, Berlin, Heidelberg, 2009. Springer Berlin Heidelberg. ISBN 978-3-642-10672-9.

Armando Solar-Lezama. Introduction to Program Synthesis, 2023. URL `https://people.csail.mit.edu/asolar/SynthesisCourse/index.htm`.

Armando Solar-Lezama, Liviu Tancau, Rastislav Bodík, Sanjit A. Seshia, and Vijay A. Saraswat. Combinatorial sketching for finite programs. In *Proceedings of the 12th International Conference on Architectural Support for Programming Languages and Operating Systems, ASPLOS 2006, San Jose, CA, USA, October 21-25, 2006*, pp. 404–415, 2006. doi: 10.1145/1168857.1168907. URL `http://doi.acm.org/10.1145/1168857.1168907`.

Armando Solar-Lezama, Christopher Grant Jones, and Rastislav Bodík. Sketching concurrent data structures. In *Proceedings of the ACM SIGPLAN 2008 Conference on Programming Language Design and Implementation, Tucson, AZ, USA, June 7-13, 2008*, pp. 136–148, 2008. doi: 10.1145/1375581.1375599. URL `http://doi.acm.org/10.1145/1375581.1375599`.

Mohit Srinivasan, Amogh Dabholkar, Samuel Coogan, and Patricio A. Vela. Synthesis of control barrier functions using a supervised machine learning approach. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 7139–7145, 2020. doi: 10.1109/IROS45743.2020.9341190.

Saurabh Srivastava, Sumit Gulwani, and Jeffrey S. Foster. From program verification to program synthesis. *SIGPLAN Not.*, 45(1):313–326, January 2010. ISSN 0362-1340. doi: 10.1145/1707801.1706337. URL `https://doi.org/10.1145/1707801.1706337`.

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, 2022a. doi: 10.1109/LRA.2022.3146951.

Shiqi Sun, Yan Zhang, Xusheng Luo, Panagiotis Vlantis, Miroslav Pajic, and Michael M. Zavlanos. Formal verification of stochastic systems with relu neural network controllers. In *2022 International Conference on Robotics and Automation (ICRA)*, pp. 6800–6806, 2022b. doi: 10.1109/ICRA46639.2022.9811866.

Xiaowu Sun, Haitham Khedr, and Yasser Shoukry. Formal verification of neural network controlled autonomous systems. In *Proceedings of the 22nd ACM International Conference on Hybrid Systems: Computation and Control*, HSCC '19, pp. 147–156, New York, NY, USA, 2019. Association for Computing Machinery. ISBN 9781450362825. doi: 10.1145/3302504.3311802. URL `https://doi.org/10.1145/3302504.3311802`.

Youcheng Sun, Min Wu, Wenjie Ruan, Xiaowei Huang, Marta Kwiatkowska, and Daniel Kroening. Concolic testing for deep neural networks. In *Proceedings of the 33rd ACM/IEEE International Conference on Automated Software Engineering*, ASE '18, pp. 109–119, New York, NY, USA, 2018. Association for Computing Machinery. ISBN 9781450359375. doi: 10.1145/3238147.3238172. URL `https://doi.org/10.1145/3238147.3238172`.

Richard Sutton. The Bitter Lesson, 2019. URL `http://www.incompleteideas.net/IncIdeas/BitterLesson.html`.

Richard S Sutton. Reinforcement learning: An introduction. *A Bradford Book*, 2018.

Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks, 2014. URL https://arxiv.org/abs/1312.6199.

Andrew J. Taylor, Victor D. Dorobantu, Hoang M. Le, Yisong Yue, and Aaron D. Ames. Episodic learning with control lyapunov functions for uncertain robotic systems. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 6878–6884. IEEE, November 2019. doi: 10.1109/iros40897.2019.8967820. URL http://dx.doi.org/10.1109/IROS40897.2019.8967820.

Vincent Tjeng, Kai Y. Xiao, and Russ Tedrake. Evaluating robustness of neural networks with mixed integer programming. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=HyGIdiRqtm.

Matthias Althoff Tobias Ladner, Ahmed Rayen Mhadhbi. Towards formally verifying llms: Taming the nonlinearity of the transformer, 2025. URL https://openreview.net/forum?id=evDSvZBFRP.

C.J. Tomlin, J. Lygeros, and S. Shankar Sastry. A game theoretic approach to controller design for hybrid systems. *Proceedings of the IEEE*, 88(7):949–970, 2000. doi: 10.1109/5.871303.

Noah Topper, George Atia, Ashutosh Trivedi, and Alvaro Velasquez. Active grammatical inference for non-markovian planning. *Proceedings of the International Conference on Automated Planning and Scheduling*, 32(1):647–651, Jun. 2022. doi: 10.1609/icaps.v32i1.19853. URL https://ojs.aaai.org/index.php/ICAPS/article/view/19853.

Rodrigo Toro Icarte, Toryn Q. Klassen, Richard Valenzano, and Sheila A. McIlraith. Reward machines: Exploiting reward function structure in reinforcement learning. *Journal of Artificial Intelligence Research*, 73:173–208, January 2022. ISSN 1076-9757. doi: 10.1613/jair.1.12440. URL http://dx.doi.org/10.1613/jair.1.12440.

Hoang-Dung Tran, Diago Manzanas Lopez, Patrick Musau, Xiaodong Yang, Luan Viet Nguyen, Weiming Xiang, and Taylor T. Johnson. Star-based reachability analysis of deep neural networks. In Maurice H. ter Beek, Annabelle McIver, and José N. Oliveira (eds.), *Formal Methods – The Next 30 Years*, pp. 670–686, Cham, 2019a. Springer International Publishing. ISBN 978-3-030-30942-8.

Hoang-Dung Tran, Patrick Musau, Diego Manzanas Lopez, Xiaodong Yang, Luan Viet Nguyen, Weiming Xiang, and Taylor T Johnson. Parallelizable reachability analysis algorithms for feed-forward neural networks. In *2019 IEEE/ACM 7th International Conference on Formal Methods in Software Engineering (FormaliSE)*, pp. 51–60, 2019b. doi: 10.1109/FormaliSE.2019.00012.

Hoang-Dung Tran, Stanley Bak, Weiming Xiang, and Taylor T. Johnson. Verification of deep convolutional neural networks using imagestars. In Shuvendu K. Lahiri and Chao Wang (eds.), *Computer Aided Verification*, pp. 18–42, Cham, 2020a. Springer International Publishing. ISBN 978-3-030-53288-8.

Hoang-Dung Tran, Xiaodong Yang, Diego Manzanas Lopez, Patrick Musau, Luan Viet Nguyen, Weiming Xiang, Stanley Bak, and Taylor T. Johnson. NNV: The Neural Network Verification Tool for Deep Neural Networks and Learning-Enabled Cyber-Physical Systems. In Shuvendu K. Lahiri and Chao Wang (eds.), *Computer Aided Verification*, pp. 3–17, Cham, 2020b. Springer International Publishing. ISBN 978-3-030-53288-8.

Hoang-Dung Tran, Weiming Xiang, and Taylor T. Johnson. Verification approaches for learning-enabled autonomous cyber–physical systems. *IEEE Design and Test*, 39(1):24–34, 2022. doi: 10.1109/MDAT.2020.3015712.

Caterina Urban and Antoine Miné. A review of formal methods applied to machine learning, 2021. URL https://arxiv.org/abs/2104.02466.

Pashootan Vaezipoor, Andrew C Li, Rodrigo A Toro Icarte, and Sheila A. Mcilraith. Ltl2action: Generalizing ltl instructions for multi-task rl. In Marina Meila and Tong Zhang (eds.), *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 10497–10508. PMLR, 18–24 Jul 2021. URL `https://proceedings.mlr.press/v139/vaezipoor21a.html`.

Prashant Vaidyanathan, Rachael Ivison, Giuseppe Bombara, Nicholas A. DeLateur, Ron Weiss, Douglas Densmore, and Calin Belta. Grid-based temporal logic inference. In *2017 IEEE 56th Annual Conference on Decision and Control (CDC)*, pp. 5354–5359, 2017. doi: 10.1109/CDC.2017.8264452.

Moshe Y. Vardi. Automatic verification of probabilistic concurrent finite state programs. In *26th Annual Symposium on Foundations of Computer Science (sfcs 1985)*, pp. 327–338, 1985. doi: 10.1109/SFCS.1985.12.

Joris Verhagen, Lars Lindemann, and Jana Tumova. Temporally robust multi-agent stl motion planning in continuous time. In *2024 American Control Conference (ACC)*, pp. 251–258, 2024. doi: 10.23919/ACC60939.2024.10644216.

Cameron Voloshin, Abhinav Verma, and Yisong Yue. Eventual discounting temporal logic counterfactual experience replay. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett (eds.), *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 35137–35150. PMLR, 23–29 Jul 2023. URL `https://proceedings.mlr.press/v202/voloshin23a.html`.

Fujun Wang, Zining Cao, Lixing Tan, and Hui Zong. Survey on learning-based formal methods: Taxonomy, applications and possible future directions. *IEEE Access*, 8:108561–108578, 2020. doi: 10.1109/ACCESS.2020.3000907.

Yanwei Wang, Nadia Figueroa, Shen Li, Ankit Shah, and Julie Shah. Temporal logic imitation: Learning plan-satisficing motion policies from demonstrations. In *6th Annual Conference on Robot Learning*, 2022. URL `https://openreview.net/forum?id=ndYsaoyzCWv`.

Yanwei Wang, Tsun-Hsuan Wang, Jiayuan Mao, Michael Hagenow, and Julie Shah. Grounding language plans in demonstrations through counterfactual perturbations. In *The Twelfth International Conference on Learning Representations*, 2024. URL `https://openreview.net/forum?id=qoHeuRAcSl`.

Yuning Wang and He Zhu. Deductive synthesis of reinforcement learning agents for infinite horizon tasks. In *37th International Conference on Computer Aided Verification (CAV)*, 2025. URL `https://par.nsf.gov/biblio/10593428`.

Min Wen, Ivan Papusha, and Ufuk Topcu. Learning from demonstrations with high-level side information. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI-17*, pp. 3055–3061, 2017. doi: 10.24963/ijcai.2017/426. URL `https://doi.org/10.24963/ijcai.2017/426`.

Eric Wong and Zico Kolter. Provable defenses against adversarial examples via the convex outer adversarial polytope. In Jennifer Dy and Andreas Krause (eds.), *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 5286–5295. PMLR, 10–15 Jul 2018. URL `https://proceedings.mlr.press/v80/wong18a.html`.

Tichakorn Wongpiromsarn, Ufuk Topcu, and Richard M. Murray. Receding horizon control for temporal logic specifications. In *Proceedings of the 13th ACM International Conference on Hybrid Systems: Computation and Control*, HSCC '10, pp. 101–110, New York, NY, USA, 2010. Association for Computing Machinery. ISBN 9781605589558. doi: 10.1145/1755952.1755968. URL `https://doi.org/10.1145/1755952.1755968`.

Weiming Xiang and Taylor T. Johnson. Reachability analysis and safety verification for neural network control systems, 2018. URL `https://arxiv.org/abs/1805.09944`.

Weiming Xiang, Hoang-Dung Tran, Joel A. Rosenfeld, and Taylor T. Johnson. Reachable set estimation and safety verification for piecewise linear systems with neural network controllers. In *2018 Annual American Control Conference (ACC)*, pp. 1574–1579, 2018. doi: 10.23919/ACC.2018.8431048.

Yaqi Xie, Fan Zhou, and Harold Soh. Embedding symbolic temporal knowledge into deep sequential models. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 4267–4273, 2021. doi: 10.1109/ICRA48506.2021.9561952.

Zikang Xiong and Suresh Jagannathan. Manipulating neural path planners via slight perturbations. *IEEE Robotics and Automation Letters*, 9(6):5006–5013, June 2024. ISSN 2377-3774. doi: 10.1109/lra.2024.3387131. URL http://dx.doi.org/10.1109/LRA.2024.3387131.

Zikang Xiong, Joe Eappen, Ahmed H. Qureshi, and Suresh Jagannathan. Model-free neural lyapunov control for safe robot navigation. In *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 5572–5579, 2022. doi: 10.1109/IROS47612.2022.9981632.

Zikang Xiong, Daniel Lawson, Joe Eappen, Ahmed H. Qureshi, and Suresh Jagannathan. Co-learning planning and control policies constrained by differentiable logic specifications. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 14272–14278, 2024. doi: 10.1109/ICRA57147.2024.10610942.

Zhe Xu, Bo Wu, Aditya Ojha, Daniel Neider, and Ufuk Topcu. Active Finite Reward Automaton Inference and Reinforcement Learning Using Queries and Counterexamples. In Andreas Holzinger, Peter Kieseberg, A. Min Tjoa, and Edgar Weippl (eds.), *Machine Learning and Knowledge Extraction*, pp. 115–135, Cham, 2021. Springer International Publishing. ISBN 978-3-030-84060-0.

Beyazit Yalcinkaya, Niklas Lauffer, Marcell Vazquez-Chanlatte, and Sanjit A. Seshia. Compositional automata embeddings for goal-conditioned reinforcement learning, 2025. URL https://arxiv.org/abs/2411.00205.

Yoriyuki Yamagata, Shuang Liu, Takumi Akazaki, Yihai Duan, and Jianye Hao. Falsification of cyber-physical systems using deep reinforcement learning. *IEEE Transactions on Software Engineering*, 47(12):2823–2840, 2021. doi: 10.1109/TSE.2020.2969178.

Cambridge Yang, Michael L. Littman, and Michael Carbin. On the (in)tractability of reinforcement learning for ltl objectives. In Lud De Raedt (ed.), *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI-22*, pp. 3650–3658. International Joint Conferences on Artificial Intelligence Organization, 7 2022. doi: 10.24963/ijcai.2022/507. URL https://doi.org/10.24963/ijcai.2022/507. Main Track.

Guang Yang, Calin Belta, and Roberto Tron. Continuous-time signal temporal logic planning with control barrier functions. In *2020 American Control Conference (ACC)*, pp. 4612–4618, 2020. doi: 10.23919/ACC45564.2020.9147387.

Chanyeol Yoo and Calin Belta. Control with probabilistic signal temporal logic, 2015. URL https://arxiv.org/abs/1510.08474.

Dongjie Yu, Haitong Ma, Shengbo Li, and Jianyu Chen. Reachability constrained reinforcement learning. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato (eds.), *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 25636–25655. PMLR, 17–23 Jul 2022. URL https://proceedings.mlr.press/v162/yu22d.html.

Yu Zhang, Peter Tino, Ales Leonardis, and Ke Tang. A survey on neural network interpretability. *IEEE Transactions on Emerging Topics in Computational Intelligence*, 5(5):726–742, October 2021. ISSN 2471-285X. doi: 10.1109/tetci.2021.3100641. URL http://dx.doi.org/10.1109/TETCI.2021.3100641.

Zhenya Zhang, Deyun Lyu, Paolo Arcaini, Lei Ma, Ichiro Hasuo, and Jianjun Zhao. Falsifai: Falsification of ai-enabled hybrid control systems guided by time-aware coverage criteria. *IEEE Transactions on Software Engineering*, 49(4):1842–1859, 2023. doi: 10.1109/TSE.2022.3194640.

Hengjun Zhao, Xia Zeng, Taolue Chen, and Zhiming Liu. Synthesizing barrier certificates using neural networks. In *Proceedings of the 23rd International Conference on Hybrid Systems: Computation and Control*, HSCC '20, New York, NY, USA, 2020. Association for Computing Machinery. ISBN 9781450370189. doi: 10.1145/3365365.3382222. URL https://doi.org/10.1145/3365365.3382222.

Ziyuan Zhong, Davis Rempe, Yuxiao Chen, Boris Ivanovic, Yulong Cao, Danfei Xu, Marco Pavone, and Baishakhi Ray. Language-guided traffic simulation via scene-level diffusion. In *7th Annual Conference on Robot Learning*, 2023a. URL https://openreview.net/forum?id=nKWQnYkkwX.

Ziyuan Zhong, Davis Rempe, Danfei Xu, Yuxiao Chen, Sushant Veer, Tong Che, Baishakhi Ray, and Marco Pavone. Guided conditional diffusion for controllable traffic simulation. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 3560–3566, 2023b. doi: 10.1109/ICRA48891.2023.10161463.

Weichao Zhou and Wenchao Li. Safety-Aware Apprenticeship Learning. In Hana Chockler and Georg Weissenbacher (eds.), *Computer Aided Verification*, pp. 662–680, Cham, 2018. Springer International Publishing. ISBN 978-3-319-96145-3.

He Zhu, Zikang Xiong, Stephen Magill, and Suresh Jagannathan. An inductive synthesis framework for verifiable reinforcement learning. In *Proceedings of the 40th ACM SIGPLAN Conference on Programming Language Design and Implementation*, PLDI 2019, pp. 686–701, New York, NY, USA, 2019. Association for Computing Machinery. ISBN 9781450367127. doi: 10.1145/3314221.3314638. URL https://doi.org/10.1145/3314221.3314638.

Shibei Zhu, Francesco Belardinelli, and Borja González León. Evolutionary reinforcement learning for sparse rewards. In *Proceedings of the Genetic and Evolutionary Computation Conference Companion*, GECCO '21, pp. 1508–1512, New York, NY, USA, 2021. Association for Computing Machinery. ISBN 9781450383516. doi: 10.1145/3449726.3463142. URL https://doi.org/10.1145/3449726.3463142.

A Acronyms

DL Deep Learning

ML Machine Learning

FM Formal Method

FS Formal Specification

SM Specification Mining

RL Reinforcement Learning

IL Imitation Learning

IRL Inverse Reinforcement Learning

LTL Linear-time Temporal Logic

STL Signal Temporal Logic

STLCG Signal Temporal Logic Computation Graphs

TQTL Timed Quality Temporal Logic

STQL Spatio-Temporal Quality Logic

MDP Markov Decision Process

FSA Finite State Automaton

DBA Deterministic Büchi Automaton

LDBA Limit-Deterministic Büchi Automaton

NDBA Non-Deterministic Büchi Automaton

DRA Deterministic Rabin Automaton

CEGIS Counter-example Guided Inductive Synthesis

ReLU Rectified Linear Unit

NN Neural Network

RNN Recurrent Neural Network

GNN Graph Neural Network

LTI Linear Time-Invariant

MILP Mixed-Integer Linear Programming

ODE Ordinary Differential Equation

PDE Partial Differential Equation

MPC Model Predictive Control

CLF Control Lyapunov Function

CBF Control Barrier Function

TSSL Tree Spatial Superposition Logic

HJB Hamilton-Jacobi-Bellman

HJ Hamilton-Jacobi

SMC Satisfiability Modulo Convex Programming

SMT Satisfiability Modulo Theories