LEARNING REACHABILITY REWARDS FOR TEMPORAL-LOGIC REINFORCEMENT LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Reinforcement learning with Linear Temporal Logic (LTL) specifications faces severe credit-assignment challenges when learning infinite-horizon Büchi objectives. We propose RACE (Reachability-Aware Credit Estimation), a framework that uses accepting-set reachability as a surrogate for learning recurrent acceptance. RACE trains a discriminator on trajectories collected by the agent to predict whether the rollout continuation from a sampled state reaches acceptance. Its predictions provide dense, state-dependent rewards for policy learning. We extend this construction to intermediate stages of the limit-deterministic Büchi automaton (LDBA), allowing partially successful trajectories to provide useful feedback even when they do not reach acceptance. Automaton-induced potentials complement these learned rewards by guiding exploration. Our analysis establishes a scaled lower bound on an infinite-time reachability objective and provides conditional connections to Büchi satisfaction. Empirically, RACE improves exploration efficiency and task performance across continuous-control domains with complex LTL objectives.

1 INTRODUCTION

Linear Temporal Logic (LTL) provides a principled language for specifying reinforcement learning (RL) objectives with rich temporal structure (Sadigh et al., 2014; Hasanbeig et al., 2018; Camacho et al., 2019; Vaezipoor et al., 2021; De Giacomo et al., 2020; Voloshin et al., 2023). Such objectives include repeatedly alternating between subgoals while avoiding unsafe regions, or completing subtasks in a prescribed order. Infinite-horizon specifications can require compliant behavior indefinitely. Learning such behavior from finite interactions is particularly challenging in high-dimensional continuous domains, where long sequences of coordinated actions hinder exploration and credit assignment.

A common approach compiles the specification into a limit-deterministic Büchi automaton (LDBA) and constructs a product MDP, where satisfaction requires infinitely many accepting visits (Hasanbeig et al., 2020; Voloshin et al., 2023). Acceptance-based rewards offer little feedback before acceptance. Recent methods address this sparsity through automaton-induced potentials (Bagatella et al., 2024) and rewards for progress along accepting paths and cycles (Shah et al., 2024). A complementary opportunity is to learn state-dependent feedback from the acceptance and intermediate progress observed in the agent's own experience.

In this paper, we propose RACE (Reachability-Aware Credit Estimation), a framework that uses finite-horizon reachability as a surrogate for learning recurrent acceptance. The key idea is to turn the outcomes of self-explored trajectories into dense, state-dependent feedback for policy learning. An acceptance-based discriminator learns from whether acceptance is reached within the available lookahead window from each sampled state. We then exploit the LDBA structure to extend this feedback to intermediate stages, allowing trajectories that make partial progress to contribute to learning even when they do not reach acceptance. Our analysis characterizes the reachability surrogate, establishes its scaled lower-bound relationship to an infinite-time reachability objective, and provides conditional connections to Büchi satisfaction.

To realize this formulation, RACE combines learned progress rewards for exploitation with automaton-guided exploration. For exploitation, we extend the reachability discriminator to intermediate automaton stages, allowing both completed and partially successful rollout segments to

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

provide dense, state-dependent feedback. This construction draws on the stage-wise discriminator architecture of Fan & Zhu (2025), but changes its learning target from expert-occupancy matching to component-relative progress inferred from the agent's own experience. For exploration, we augment the automaton-induced potentials of Bagatella et al. (2024) with stage-aware feedback to guide the discovery of accepting behaviors. Together, these mechanisms connect structural guidance toward acceptance with learned rewards that exploit the progress observed in collected trajectories.

We evaluate RACE on navigation, locomotion, and robotic manipulation tasks with complex LTL objectives. The results show faster learning and higher eventual-discounted returns than the compared baselines, while ablations demonstrate the complementary contributions of stage-aware exploration and discriminator-based exploitation.

2 Related Work

Automaton-based RL represents task structure through product MDPs and logic-derived rewards (Sadigh et al., 2014; Bozkurt et al., 2020; Kantaros, 2022; Camacho et al., 2019). LDBA-based methods target recurrent acceptance (Hasanbeig et al., 2018; 2020; Cai et al., 2021), with dynamic and eventual discounting connecting surrogate optimization to satisfaction under suitable assumptions (Shao & Kwiatkowska, 2023; Voloshin et al., 2023). General LTL objectives nevertheless face limitations on PAC learning in unknown environments (Alur et al., 2022; Yang et al., 2021). Finite-trace approaches include LTL_f/LDL_f rewards evaluated on successive finite prefixes (De Giacomo et al., 2019) and SPECTRL's monitors and shaped rewards for finite-rollout tasks (Jothimurugan et al., 2019). RACE retains infinite-trace Büchi semantics: its finite reachability horizon bounds the look-ahead to acceptance, not the execution on which the specification is interpreted.

Reward machines (Icarte et al., 2018; 2022), quantitative semantics (Li et al., 2017; Balakrishnan & Deshmukh, 2019; Venkataraman et al., 2020), and hierarchical or goal-conditioned policies (Jothimurugan et al., 2021; Jackermeier & Abate, 2024; Qiu et al., 2024) further exploit task structure for credit assignment and reuse. Closely related approaches provide denser feedback within the Büchi setting: DRL[2] learns an automaton-level transition model and derives potential-based rewards (Bagatella et al., 2024), whereas CyclER shapes rewards using trajectory progress along accepting paths and cycles (Shah et al., 2024). RACE complements automaton-induced potentials with state-dependent discriminator rewards learned from rollout outcomes. Its stage-wise construction builds on Fan & Zhu (2025), but learns from self-explored progress rather than expert occupancy measures. Appendix A provides a broader discussion.

3 Background and Problem Setup

3.1 Linear Temporal Logic (LTL)

An LTL specification is evaluated on an infinite sequence of labels from 2^{AP}, where AP is a finite set of Boolean propositions (Pnueli, 1977). Boolean connectives describe conditions at a position; temporal operators constrain their occurrence along the sequence. In particular, $\mathsf{X}\varphi$ refers to the next position, $\mathsf{F}\varphi$ to some future position, and $\mathsf{G}\varphi$ to every position. The operator U expresses a condition that persists until another becomes true. Appendix B gives the infinite-trace semantics following Baier & Katoen (2008).

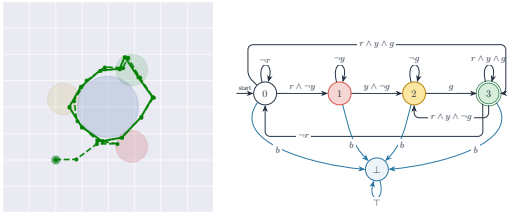

Figure 1: FlatWorld Cycle example, adapted from Fig. 1 of Fan & Zhu (2025). Left: a cyclic trajectory avoiding blue. Right: the LDBA for $\varphi = \mathsf{GF}(y \wedge \mathsf{XF}(g \wedge \mathsf{XF}r)) \wedge \mathsf{G}\neg b$, with accepting state 3 and a rejecting sink.

Running task. The FlatWorld example from Shah et al. (2024); Fan & Zhu (2025) illustrates why reaching a goal once is insufficient. The propositions y, g, r, b identify the colored regions in Fig. 1. The formula $\mathsf{GF}(y \wedge \mathsf{XF}(g \wedge \mathsf{XF}r)) \wedge \mathsf{G}\neg b$ requires the ordered visits to recur while excluding blue at every step. Completing one traversal gives

a finite indication of progress, but satisfaction depends on the continuation of the execution. This distinction motivates the local reachability signals studied below.

From LTL to LDBA. We represent φ by an LDBA whose accepting runs recognize its satisfying label sequences (Sickert et al., 2016). We use an explicit-action representation of the automaton's ϵ-choices.

Definition 1 (Limit Deterministic Büchi Automaton (LDBA)). *Write $\mathcal{L} = (\mathcal{B}, \Sigma \cup \mathcal{E}, P^{\mathcal{B}}, \mathcal{B}^\star, b_0)$, with finite automaton state space $\mathcal{B}$, initial state b_0, and accepting set $\mathcal{B}^\star$. Ordinary transitions consume labels in $\Sigma = 2^{AP}$; an enabled action in $\mathcal{E}$ selects an ϵ-transition without consuming a label. The transition map $P^{\mathcal{B}} : \mathcal{B} \times (\Sigma \cup \mathcal{E}) \to \mathcal{B}$ is understood on enabled state–input pairs. The state space has a disjoint partition $\mathcal{B}_N \cup \mathcal{B}_D$, where $\mathcal{B}^\star \subseteq \mathcal{B}_D$. Label transitions within $\mathcal{B}_D$ are deterministic and remain in $\mathcal{B}_D$; ϵ-actions resolve jumps from $\mathcal{B}_N$ into $\mathcal{B}_D$.*

After resolving enabled ϵ-choices, an automaton run is a sequence $(b_i)_{i\geq 0}$ generated by $P^{\mathcal{B}}$. Its success criterion concerns infinitely many visits, rather than arrival at a terminal goal.

Definition 2 (Büchi acceptance). *A run is accepting when the set $\{i \geq 0 : b_i \in \mathcal{B}^\star\}$ is infinite. Equivalently, every finite prefix has an accepting visit strictly later in the run.*

The translation ensures that a label sequence satisfies φ precisely when it admits an accepting automaton run. In the product construction, the policy selects any enabled ϵ-actions together with the environment actions.

3.2 Product MDP with LDBA

Let $\mathcal{M} = (S, A, P, \mu_0)$ describe the environment: $s_0 \sim \mu_0$ and $s_{t+1} \sim P(\cdot \mid s_t, a_t)$ for actions $a_t \in A$. The spaces S and A may be continuous. We omit a task reward from this tuple because the desired behavior is specified by φ; rewards for learning will be constructed separately.

The labeling map $\mathcal{F} : S \to 2^{AP}$ connects physical states to the specification (Voloshin et al., 2023; Vaezipoor et al., 2021; Bagatella et al., 2024). Augmenting s with the current automaton state b makes progress information available to the controller. We follow the product convention of Fan & Zhu (2025): environment actions update the automaton using the successor label, whereas an ϵ-action changes only the automaton state.

Definition 3 (Product MDP). *The controlled product process has state space $S^\times = S \times \mathcal{B}$ and action space $A^\times = A \cup \mathcal{E}$, restricted to actions enabled at the current state. Its initial distribution is $\mu_0^\times = \mu_0 \otimes \delta_{b_0}$. For $x = (s, b)$ and $x' = (s', b')$, its kernel is*

$$P^\times\big((s', b') \mid (s, b), a\big) = \begin{cases} P(s' \mid s, a), & \text{if } a \in A, \quad b' = P^{\mathcal{B}}(b, \mathcal{F}(s')) \\ 1, & \text{if } a \in \mathcal{E}, \quad b' = P^{\mathcal{B}}(b, a)\,,\ s = s' \\ 0, & \text{otherwise.} \end{cases} \tag{1}$$

Denote this product MDP by $\mathcal{M}^\times = (S^\times, A^\times, P^\times, \mu_0^\times)$. A product policy $\pi : S^\times \to \Delta(A^\times)$ determines both physical control and the resolution of automaton choices. We evaluate it by the probability of recurrent acceptance, $P(\pi \models \varphi) := \Pr_\pi(b_t \in \mathcal{B}^\star \text{ infinitely often})$. The control objective is $\pi^* \in \arg\max_{\pi\in\Pi} P(\pi \models \varphi)$, where Π denotes the space of all admissible policies.

4 Balancing Exploration and Exploitation with LTL Constraints

Our main objective is to learn a product MDP policy that satisfies the Büchi acceptance condition, i.e., visits accepting product states infinitely often.

4.1 Learning from Self-Explored Rollout Segments

Let $F \subseteq S^\times$ denote the set of accepting product states, i.e., $F := \{(s, b) \in S^\times \mid b \in \mathcal{B}^*\}$. Fix an initial distribution and a class Π of stationary Markov policies. Let $L \geq 1$ be a finite integer reachability horizon used to measure progress toward acceptance from a product state $x = (s, b)$.

Finite-horizon reachability. For any nonnegative integer horizon h and product state $x \in S^{\times}$, define

$$q_h^{\pi}(x) := \Pr_{\pi}\big(\exists\, k \in \{0, \ldots, h\} \text{ such that } x_k \in F \,\big|\, x_0 = x\big),$$

where the event includes the current state.

Infinite-time reachability objective. We evaluate finite-horizon reachability along the entire continuing trajectory, including after previous accepting visits. Let $d_{\infty,\gamma}^{\pi}$ be the joint law of (t, x_t) obtained by sampling $t \geq 0$ with probability $(1-\gamma)\gamma^t$ and observing x_t under π, for $0 < \gamma < 1$. Define

$$J_L^{\infty,\gamma}(\pi) := \mathbb{E}_{(t,x_t)\sim d_{\infty,\gamma}^{\pi}}[q_L^{\pi}(x_t)] = (1-\gamma)\sum_{t=0}^{\infty}\gamma^t \mathbb{E}_{\pi}[q_L^{\pi}(x_t)].$$

The factor $1-\gamma$ normalizes the weights to sum to one. Unlike eventual discounting, this objective discounts time rather than accepting visits.

Motivation from satisfaction optimality. Write $B(\pi) = \Pr_{\pi}(x_t \in F \text{ infinitely often})$ and $B^* = \sup_{\pi\in\Pi} B(\pi)$. Suppose expected waiting times to acceptance on satisfying trajectories are uniformly bounded by R, and expected total accepting-visit counts on failing trajectories by M, across Π (assumptions B1–B2 in Appendix C.4). Theorem 3 shows that any global maximizer $\pi_{L,\gamma}^*$ of $J_L^{\infty,\gamma}$, if one exists, satisfies

$$B^* - B(\pi_{L,\gamma}^*) \leq \frac{R}{L+1} + (1-\gamma)(L+1)M.$$

The bound vanishes as $L \to \infty$, $\gamma \uparrow 1$, and $(1-\gamma)(L+1) \to 0$. This motivates the objective, but its approximation quality depends jointly on L, γ, and potentially unknown temporal properties of the environment and policy class.

A practical lower-bound objective. We develop a practical learning approach for optimizing a finite-rollout lower bound without specifying the analytical horizon L. For training rollouts of length $T \geq 1$, every comparison horizon $L \geq T$ gives the same available suffix horizon

$$H(t) := \min\{L, T-t\} = T - t, \qquad 0 \leq t < T.$$

Restricting $d_{\infty,\gamma}^{\pi}$ to $t < T$ and renormalizing gives $d_{T,\gamma}^{\pi}$, with time weights γ^t/Z_T, where $Z_T = \sum_{t=0}^{T-1}\gamma^t$. Define

$$J_H(\pi) := \mathbb{E}_{(t,x_t)\sim d_{T,\gamma}^{\pi}}[q_{H(t)}^{\pi}(x_t)] = \frac{1}{Z_T}\sum_{t=0}^{T-1}\gamma^t \mathbb{E}_{\pi}[q_{H(t)}^{\pi}(x_t)].$$

We suppress the fixed T, γ in J_H. Theorem 1 establishes the lower bound; its construction depends only on rollout suffixes, not the numerical value of $L \geq T$.

Discriminator learning. Label each sampled state by the observable reachability event

$$Y_t := \mathbf{1}\{\exists\, k \in \{0, \ldots, H(t)\} : x_{t+k} \in F\},$$

whose conditional mean given (t, x_t) is $q_{H(t)}^{\pi}(x_t)$. Segmenting a rollout at accepting visits labels preceding segments positive and the non-accepting tail after the final visit negative. A sampled state already in F has label 1 because the event includes $k = 0$; if acceptance is never visited, all labels are zero.

To estimate J_H, fit a discriminator $q_\psi : S^{\times} \to (0, 1)$ under the same discounted temporal weighting. Binary cross-entropy minimization is equivalent to

$$\max_{\psi}\ \mathbb{E}_{d_{T,\gamma}^{\pi}}\big[Y_t \log q_\psi(x_t) + (1-Y_t)\log(1-q_\psi(x_t))\big]. \tag{2}$$

The expectation includes the continuation determining Y_t. Equivalently, uniformly sampled on-policy examples receive loss weights proportional to γ^t. This is an idealized matching-sampling loss; Appendix C.1 treats pooled data and sampling mismatch. The probability output provides the basic dense reward $R_{\text{disc}}(x_t) := q_\psi(x_t)$. Learning alternates rollout collection, labeling, discriminator fitting, and policy updates using this reward.

Theoretical characterization. For the discriminator identity, assume same-policy rollouts, matching discounted time sampling without label-dependent resampling, realizability, and exact population cross-entropy optimization (Appendix C.1).

Theorem 1 (Finite-horizon reachability surrogate)**.** *Under the discriminator assumptions just stated, the following hold for every finite $L \geq T$ and $0 < \gamma < 1$:*

1. $\mathbb{E}_{(t,x_t)\sim d^{\pi}_{T,\gamma}}[q_\psi(x_t)] = J_H(\pi)$;
2. $q^{\pi}_{H(t)}(x_t) \leq q^{\pi}_{L}(x_t)$ *for all* x_t;
3. $(1-\gamma^T)J_H(\pi) \leq J^{\infty,\gamma}_{L}(\pi)$.

We seek a policy maximizing the finite-horizon surrogate:

$$\max_{\pi\in\Pi} J_H(\pi). \tag{3}$$

Since $1-\gamma^T > 0$ is policy-independent, this maximizes the lower bound in part (iii) without knowing L. The theorem does not guarantee monotonic policy improvement or transfer the upper objective's satisfaction guarantee to lower-bound optimization. Appendix C.3 characterizes the gap, and Appendix C.4 gives conditions for it to close.

Connection to recurrent acceptance. For any threshold $\varepsilon \in (0,1)$,

$$(1-\varepsilon)\mathbf{1}_{\{q^{\pi}_{L}(x)<\varepsilon\}} \leq 1 - q^{\pi}_{L}(x).$$

Taking expectations under $d^{\pi}_{\infty,\gamma}$, dividing by $1-\varepsilon > 0$, and applying Theorem 1(iii) yields

$$\Pr_{(t,x_t)\sim d^{\pi}_{\infty,\gamma}}(q^{\pi}_{L}(x_t) < \varepsilon) \leq \frac{1 - J^{\infty,\gamma}_{L}(\pi)}{1-\varepsilon} \leq \frac{1-(1-\gamma^T)J_H(\pi)}{1-\varepsilon}. \tag{4}$$

For fixed T, γ, ε, increasing J_H tightens this upper bound on the infinite-time discounted mass of states with low L-step reachability. This gives a distribution-weighted connection to the uniform condition below, but a small mass does not exclude such states (Appendix C.2).

Theorem 2 (Uniform reachability implies recurrent acceptance)**.** *For any finite reachability horizon $L \geq T$, assume that the product-state process induced by the fixed policy π is (strong) Markov. Let* $\mathrm{Reach}(\pi)$ *denote the set of product states reachable under π from the initial distribution. Then*

$$\inf_{x\in\mathrm{Reach}(\pi)} q^{\pi}_{L}(x) \geq \varepsilon > 0 \quad\Longrightarrow\quad \Pr_{\pi}(x_t \in F \textit{ infinitely often}) = 1.$$

Restricting to $L \geq T$ is without loss of generality for this condition: any qualifying horizon can be enlarged to at least T (Appendix C.2). Neither L nor ε is an algorithm input. The distinction between occupancy-weighted and uniform reachability prevents an unconditional satisfaction guarantee from Eq. (4).

4.2 LDBA-GUIDED EXPLORATION

Learning the reachability signal in Sec. 4.1 requires exploring trajectories that reach acceptance, despite sparse and temporally extended LTL feedback. RACE combines automaton-guided exploration with discriminator-based exploitation, using a common notion of stage progress (Fig. 2).

Many LTL tasks decompose into stages: the FlatWorld Cycle automaton in Fig. 1, for instance, tracks successive region visits before acceptance. In general, however, distinct non-accepting states may be mutually reachable, so an automaton-state change need not indicate progress. We therefore group non-accepting states that are mutually reachable without passing through acceptance into a single stage. Moving between such groups records structural progress within a round, whereas transitions out of acceptance begin a new round.

Component-based stages. Let $\mathcal{B}_\perp$ contain the states from which acceptance is unreachable, and put $\mathcal{B}_{\mathrm{live}} = \mathcal{B} \setminus (\mathcal{B}^\star \cup \mathcal{B}_\perp)$. On the graph induced by $\mathcal{B} \setminus \mathcal{B}_\perp$, remove all outgoing edges from accepting states *before* computing the strongly connected components $\mathcal{C}$. Define $c : \mathcal{B} \setminus \mathcal{B}_\perp \to \mathcal{C}$ to map each state b to the unique component containing it, i.e., $c(b) = C$ if and only if $b \in C$. Each accepting state then forms a singleton terminal component; write $\mathcal{C}^\star = \{\{b_f\} : b_f \in \mathcal{B}^\star\}$. The component graph is acyclic; write $C \prec C'$ when a directed path leads from C to a distinct C'. This construction prevents cycles through acceptance from merging stages. The order describes progress within an

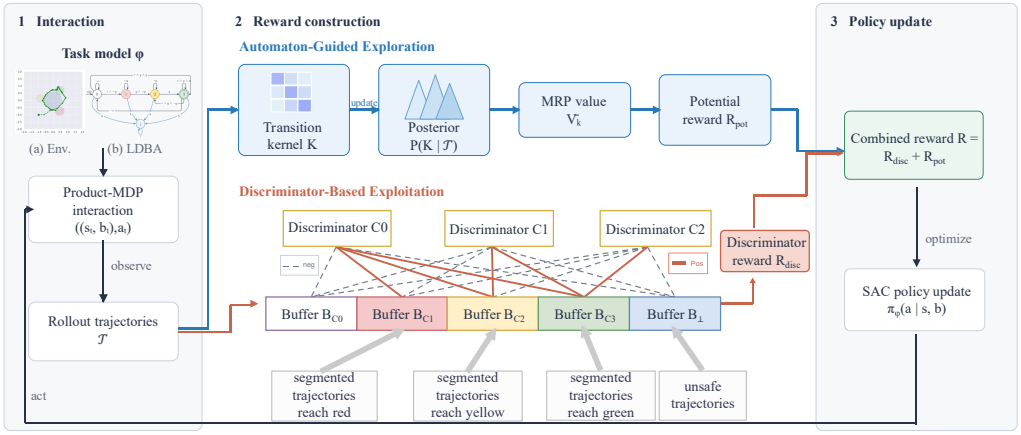

Figure 2: Overview of RACE. The stage-buffer schematic adapts the design of Fan & Zhu (2025). Here, agent-collected trajectories train component-wise discriminators and an automaton-based MRP; no demonstration buffer is used. Only non-accepting components have discriminator heads. Learned progress rewards and automaton potentials jointly guide policy optimization.

accepting round, from one accepting visit to the next strictly later visit; the initial prefix ends at the first accepting visit. Both exploration and exploitation use these stages. Edges are removed only in this auxiliary graph: the original LDBA and product-state dynamics, including transitions that begin a new round, remain unchanged.

In Fig. 3, the non-accepting cycle $b_1 \leftrightarrow b_2$ forms C_1, while b_3 and b_f form C_2 and $C_f = \{b_f\}$, respectively. Thus $C_1 \prec C_2 \prec C_f$; reaching C_f completes the round. The return $b_f \to b_1$ begins another round without merging these components.

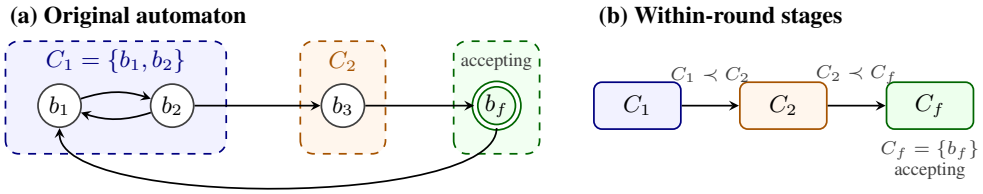

Figure 3: Component stages preserve accepting cycles. (a) An automaton graph with non-accepting states b_1, b_2, b_3 and accepting state b_f (double circle); transition conditions are omitted. Shaded components are computed after cutting outgoing edges from acceptance. (b) The resulting component DAG includes the accepting singleton $C_f = \{b_f\}$.

Following Bagatella et al. (2024), we construct an auxiliary Markov reward process over the original LDBA states using a stochastic matrix K, estimated from collected product-state transitions. Neither the automaton projection nor a component-level aggregation is assumed Markov. Its value function supplies an exploration potential under this surrogate model, not a satisfaction probability.

Prior work (Bagatella et al., 2024) rewards only accepting-state visitation. We additionally reward component progress in the auxiliary MRP $\bar{\mathcal{L}} = (\mathcal{B}, K, R^{\times}, \gamma^{\times}, \mathcal{B}^*, b_0)$. Trajectories follow $b_t \sim K(\cdot \mid b_{t-1})$. For a live successor b', write $C' = c(b')$. Define

$$R^{\times}(b, b') = \begin{cases} 1, & b' \in \mathcal{B}^*, \\ \textsc{Idx}(C')/\mathcal{N}(C'), & b' \in \mathcal{B}_{\text{live}},\ b \notin C', \\ 0, & \text{otherwise}, \end{cases} \qquad \gamma^{\times}(b, b') = \begin{cases} 1, & b' = b \notin \mathcal{B}^*, \\ \gamma, & \text{otherwise}. \end{cases} \tag{5}$$

For $C \in \mathcal{C} \setminus \mathcal{C}^\star$, with path lengths measured in edges, $\textsc{Idx}(C)$ is the longest path length from any source component to C, and $\mathcal{N}(C)$ is the longest source-to-accepting-component path length

through C. Every non-accepting component has $\mathcal{N}(C) > 0$, and its normalized score lies in $[0, 1)$. Appendix G gives the construction and establishes that the score increases along $\prec$ between non-accepting components.

Transitions into a non-accepting component from outside it, including from acceptance into a new round, receive its stage score. Movement within a non-accepting component or into structural failure receives zero; acceptance yields a unit reward. The discount remains tied to original automaton-state changes and accepting visits, not component changes. Thus non-accepting self-loops do not discount continuation, while movement between distinct states within a component is discounted without stage reward. This retains the value-existence conditions in Appendix E.

Define $\Gamma_t = \prod_{j=1}^{t-1} \gamma^{\times}(b_{j-1}, b_j)$, with $\Gamma_1 = 1$. The associated value function is

$$\bar{V}_K(b) = \mathbb{E}_K \left[\sum_{t=1}^{\infty} \Gamma_t \, R^{\times}(b_{t-1}, b_t) \;\middle|\; b_0 = b \right].$$

Non-accepting states that are absorbing under K, including rejecting sinks, have zero value. At the remaining states, the Bellman equation (Sutton et al., 1998) is

$$\bar{V}_K(b) = \sum_{b'} K(b, b') \left[R^{\times}(b, b') + \gamma^{\times}(b, b') \bar{V}_K(b') \right]. \tag{6}$$

Appendix E gives the corresponding boundary-aware linear system and establishes its unique solution. We use $\bar{V}_K$ as the potential for reward shaping (Ng et al., 1999):

$$R_{\mathrm{pot}}(b, b') = \gamma \, \bar{V}_K(b') - \bar{V}_K(b) \tag{7}$$

We combine the potential-based reward $R_{\mathrm{pot}}(b, b')$ with the acceptance-based discriminator reward $R_{\mathrm{disc}}(x) = q_\psi(x)$ introduced in Sec. 4.1 to obtain the following reward for policy learning:

$$R((s, b), (s', b')) = R_{\mathrm{disc}}((s', b')) + R_{\mathrm{pot}}(b, b') \tag{8}$$

Section 4.3 refines the discriminator reward using component-wise progress signals. We use a structured Dirichlet prior over LDBA transitions to initialize the kernel K and update it through a Bayesian framework during training based on experienced rollouts from the environment following (Bagatella et al., 2024), as visualized in Fig. 2. More details are provided in Appendix. E. In Appendix F, we further discuss the non-stationarity induced by the evolving kernel and its implications for convergence.

4.3 LDBA-GUIDED EXPLOITATION

Explored trajectories contain useful evidence of partial progress, even when they do not reach acceptance. To exploit this experience, we extend the acceptance-based discriminator introduced in Sec. 4.1 to stage-wise discriminators that use the LDBA's component structure to assess intermediate progress. These discriminators translate observed progress into dense, state-dependent rewards for subsequent policy learning.

Building on the stage-wise discriminator framework of Fan & Zhu (2025), we train one discriminator $q_{\psi,C} : S \to (0, 1)$ per non-accepting component $C \in \mathcal{C} \setminus \mathcal{C}^{\star}$, using progress labels from self-explored segments rather than expert occupancy targets. Each discriminator learns from segment-level labels to distinguish states associated with progress beyond component C from those associated with stalled progress or structural failure. Whereas the acceptance-based discriminator in Sec. 4.1 estimates finite-horizon reachability of the accepting set, these discriminators provide localized signals of progress relative to individual components.

For $b \in \mathcal{B}_{\mathrm{live}}$, define the component-wise predictor $q^{\mathrm{st}}_\psi(s, b) := q_{\psi, c(b)}(s)$, so automaton states in the same component share a discriminator. Adapting the stage-indexed reward of Fan & Zhu (2025) to these component scores and probability outputs gives

$$R_{\mathrm{disc}}((s, b)) = \frac{\textsc{Idx}(c(b)) + \beta \tanh\big(q^{\mathrm{st}}_\psi(s, b)\big)}{\mathcal{N}(c(b))} \tag{9}$$

where β controls the learned contribution. Each component has a buffer B_C, including accepting singletons; structurally failing segments enter $B_\perp$. Appendix G specifies disjoint training pools and

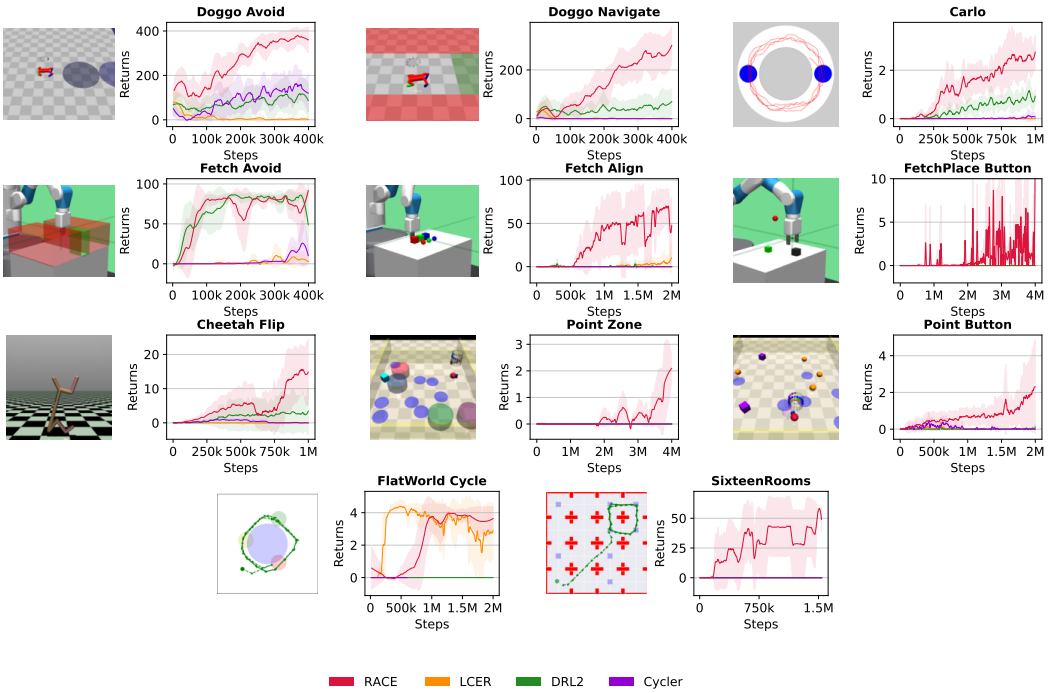

Figure 4: Returns under eventual discounting (Voloshin et al., 2023) comparing RACE and the baselines over 10 random seeds.

boundary rewards; Appendix C.5 characterizes the learned posteriors and their conditional connection to recurrent acceptance. The policy, critic, and exploration kernel still use b.

In the final reward formulation (equation 8), R_{disc} (equation 9) and R_{pot} (equation 7) balance structural guidance from the logical specification with data-driven feedback from observed trajectories, encouraging both directed exploration and effective exploitation toward satisfying the LTL objective.

As summarized in Algorithm 1, we alternate between training the discriminators and updating the policy using SAC (Haarnoja et al., 2018) with the combined reward R (equation 8). Following Fan & Zhu (2025), we use a fixed accepting-state Q-value target to credit segment completion independently of subsequent-round task returns. Implementation details and a theoretical characterization of this boundary rule are provided in Appendix D.

5 Experiments

Baselines. The comparison covers three state-of-the-art policy optimization methods for LTL objectives: counterfactual automaton-state replay in LCER (Voloshin et al., 2023), Bayesian automaton potentials in DRL^2 (Bagatella et al., 2024), and trajectory-progress shaping along accepting paths and cycles in CyclER (Shah et al., 2024).

Environments and Tasks. Our benchmarks, illustrated in Fig. 4, comprise navigation, manipulation, and locomotion tasks adapted from prior work (Voloshin et al., 2023; Bagatella et al., 2024; Fan & Zhu, 2025; Ji et al., 2023). Their LTL specifications combine ordered goal reaching, recurrent visitation, and safety constraints. In *Doggo Avoid* and *Doggo Navigate*, the agent must respectively reach a goal through a safe corridor or visit two zones in sequence. *Carlo* requires repeated visits to two waypoints without crashing. The manipulation tasks require reaching a target while avoiding unsafe regions (*Fetch Avoid*), positioning three cubes in sequence (*Fetch Align*), or repeatedly alternating between goal placement and button pressing (*FetchPlace Button*). For locomotion, *Cheetah Flip* requires repeated frontflips between two designated body configurations. *PointZone* and *PointButton* require recurrent visits to colored regions or designated buttons while avoiding hazards or gremlins. Finally, *FlatWorld Cycle* requires ordered, repeated visits to colored regions while avoid-

ing blue, and *SixteenRooms* requires repeated traversal of either of two prescribed routes without wall collisions. Appendix H provides the LTL formulas and environment details.

Results. Exact evaluation of infinite-horizon LTL satisfaction is impractical in our unknown continuous-state environments, where long-run policy occupancy and recurrent behavior are difficult to characterize. We therefore evaluate performance using eventual-discounted return (Voloshin et al., 2023) as a practical proxy for task satisfaction. This return assigns a unit reward at each LDBA-accepting visit and advances the discount only at these visits (Appendix P.2). Figure 4 reports the learning curves, with environment steps on the horizontal axis and eventual-discounted returns on the vertical axis.

Figure 4 shows substantial learning improvements for RACE on several challenging tasks. CyclER's performance remains low in high-dimensional environments such as *Cheetah Flip* and *PointButton*. Although CyclER rewards progress along accepting paths and cycles, discovering the low-level behaviors needed to realize that progress can remain difficult in high-dimensional settings. RACE combines automaton-induced exploration potentials with state-dependent discriminator rewards, providing complementary guidance for discovering and exploiting intermediate progress. Compared with LCER, which reuses observed transitions through counterfactual automaton-state relabeling, RACE additionally uses automaton-level values to direct exploration toward acceptance. DRL^2 also supplies this structural guidance, but RACE augments it with stage-aware potential shaping and component-wise discriminators. The former credits intermediate advances through the automaton, while the latter extracts localized learning signals from both completed and partially successful segments. These mechanisms results in the gains on tasks requiring multiple intermediate behaviors before acceptance. In *PointButton*, RACE completes three to four accepting rounds within 1,000 evaluation steps, whereas the compared baselines rarely complete one.

Direct behavioral evaluation further supports RACE's generalization to longer execution horizons: without additional training, it continues to complete accepting rounds beyond the training horizon and achieves higher mean accepting state visitations than DRL^2, LCER, and CyclER (Appendix M).

Ablation Studies. We evaluate the contributions of the staged potential reward $R_{\mathrm{pot}}((s,b))$, and the multistage discriminator reward $R_{\mathrm{disc}}((s,b))$. Specifically, RACE_noPot removes potential-based shaping, while RACE_noStagedPot removes the stage-aware terms based on $\textsc{Idx}(c(b_t))/\mathcal{N}(c(b_t))$ and the transition-dependent discounting $\gamma^{\times}$ in Eq. equation 5. RACE_noStagedDisc removes the stage-wise discriminator structure and trains a single discriminator to distinguish successful from failed segments. As shown in Fig. 7, removing either staged exploration or staged exploitation substantially degrades learning, confirming that both components are important for guiding progress toward LTL satisfaction. More details are provided in Appendix J.

We provide additional results on scalability and robustness to stochastic environments in Appendix K and Appendix L, as well as a comparison to the imitation learning baseline TiLoIL in Appendix N and goal-conditioned baseline DeepLTL in Appendix O.

6 Conclusion

We present RACE, an RL framework that uses learned finite-horizon progress signals to guide policy optimization for LTL-constrained tasks. Our analysis establishes a scaled lower-bound relationship between the reachability surrogate and an infinite-time reachability objective, quantifies their gap, and provides conditional connections to Büchi satisfaction. Empirical results support the effectiveness of the staged extension in continuous-control tasks.

Limitations and future work. First, RACE is built around finite-horizon surrogates that provide learnable feedback. The reachability surrogate yields a scaled lower bound on an infinite-time reachability objective, but even global optimization of this bound need not maximize Büchi satisfaction. Closing this gap and transferring optimality require additional assumptions; RACE therefore remains a practical surrogate-based algorithm. Second, automaton-level guidance does not eliminate difficult exploration within stages, and discriminator rewards can be affected by limited data coverage and mismatch between replay data and the current policy. Finally, the formulation assumes access to a ground-truth state-labeling function $\mathcal{F}$. Extending the approach to learned or noisy proposition labels while preserving reliable automaton progress and reward construction remains an important direction for future work.

AI USE STATEMENT

We used LLM-based generative AI tools, including OpenAI Codex, to assist with refining the presentation of theoretical claims and proofs, providing feedback on experimental methodology and interpretation, and revising the manuscript. We also used these tools to improve readability and structure, review related-work coverage, format LaTeX, and create or modify scientific figures and presentation artifacts. We did not use generative AI to generate synthetic datasets, collect or clean experimental data, implement the reported learning methods, execute experiments, or produce the reported numerical results. The authors reviewed all AI-assisted material: mathematical statements were checked against the derivations, literature claims against the cited sources, and textual and visual revisions against the underlying experimental artifacts. The authors take responsibility for the final text, claims, results, and artifacts in this paper.

ETHICS STATEMENT

This work studies reinforcement-learning algorithms using simulated control and navigation benchmarks. It does not involve human participants, personal data, or newly collected real-world datasets. Policies learned in simulation may behave unpredictably under distribution shift and should not be interpreted as certified for deployment in physical or safety-critical systems. Any real-world use would require additional validation, monitoring, and domain-specific safety measures.

REPRODUCIBILITY STATEMENT

The main paper specifies the learning objective, reward construction, and principal algorithmic components. The appendix provides the complete algorithm, assumptions and proofs for the theoretical results, environment and LTL-task definitions, hyperparameters, evaluation metrics, software tools, computational costs, wall-clock measurements, and additional ablations. The reported comparisons use the protocols and random-seed counts stated in the corresponding sections and figure captions.

REFERENCES

David Abel, Will Dabney, Anna Harutyunyan, Mark K Ho, Michael Littman, Doina Precup, and Satinder Singh. On the expressivity of markov reward. *Advances in Neural Information Processing Systems*, 34:7799–7812, 2021.

Rajeev Alur, Suguman Bansal, Osbert Bastani, and Kishor Jothimurugan. A framework for transforming specifications in reinforcement learning. In *Principles of Systems Design: Essays Dedicated to Thomas A. Henzinger on the Occasion of His 60th Birthday*, pp. 604–624. Springer, 2022.

Marco Bagatella, Andreas Krause, and Georg Martius. Directed exploration in reinforcement learning from linear temporal logic. *arXiv preprint arXiv:2408.09495*, 2024.

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

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. IEEE, 2019.

Alper Kamil Bozkurt, Yu Wang, Michael M Zavlanos, and Miroslav Pajic. Control synthesis from linear temporal logic specifications using model-free reinforcement learning. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 10349–10355. IEEE, 2020.

Mingyu Cai, Mohammadhosein Hasanbeig, Shaoping Xiao, Alessandro Abate, and Zhen Kan. Modular deep reinforcement learning for continuous motion planning with temporal logic. *IEEE robotics and automation letters*, 6(4):7973–7980, 2021.

Alberto Camacho, Oscar Chen, Scott Sanner, and Sheila McIlraith. Non-markovian rewards expressed in ltl: guiding search via reward shaping. In *Proceedings of the International Symposium on Combinatorial Search*, volume 8, pp. 159–160, 2017.

Alberto Camacho, Rodrigo Toro Icarte, Toryn Q Klassen, Richard Anthony Valenzano, and Sheila A McIlraith. Ltl and beyond: Formal languages for reward function specification in reinforcement learning. In *IJCAI*, volume 19, pp. 6065–6073, 2019.

Giuseppe De Giacomo, Luca Iocchi, Marco Favorito, and Fabio Patrizi. Foundations for restraining bolts: Reinforcement learning with LTLf/LDLf restraining specifications. In *Proceedings of the International Conference on Automated Planning and Scheduling*, volume 29, pp. 128–136, 2019. doi: 10.1609/icaps.v29i1.3549.

Giuseppe De Giacomo, Antonio Di Stasio, Moshe Vardi, Shufang Zhu, et al. Two-stage technique for ltlf synthesis under ltl assumptions. In *Proceedings of the 17th International Conference on Principles of Knowledge Representation and Reasoning*, pp. 304–314, 2020.

R. de Lazcano, K. Andreas, J. J. Tai, S. R. Lee, and J. Terry. Gymnasium robotics. GitHub, 2023. URL `http://github.com/Farama-Foundation/Gymnasium-Robotics`.

Xuchu Ding, Stephen L Smith, Calin Belta, and Daniela Rus. Optimal control of markov decision processes with linear temporal logic constraints. *IEEE Transactions on Automatic Control*, 59(5): 1244–1257, 2014.

Zining Fan and He Zhu. Imitation learning with temporal logic constraints. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL `https://openreview.net/forum?id=MKLcSmSTQI`.

Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International conference on machine learning*, pp. 1861–1870. PMLR, 2018.

Charles R Harris, K Jarrod Millman, Stéfan J Van Der Walt, Ralf Gommers, Pauli Virtanen, David Cournapeau, Eric Wieser, Julian Taylor, Sebastian Berg, Nathaniel J Smith, et al. Array programming with numpy. *Nature*, 585(7825):357–362, 2020.

Mohammadhosein Hasanbeig, Alessandro Abate, and Daniel Kroening. Logically-constrained reinforcement learning. *arXiv preprint arXiv:1801.08099*, 2018.

Mohammadhosein Hasanbeig, Daniel Kroening, and Alessandro Abate. Deep reinforcement learning with temporal logics. In *Formal Modeling and Analysis of Timed Systems: 18th International Conference, FORMATS 2020, Vienna, Austria, September 1–3, 2020, Proceedings 18*, pp. 1–22. Springer, 2020.

Rodrigo Toro Icarte, Toryn Klassen, Richard Valenzano, and Sheila McIlraith. Using reward machines for high-level task specification and decomposition in reinforcement learning. In *International Conference on Machine Learning*, pp. 2107–2116. PMLR, 2018.

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, 2022.

Mathias Jackermeier and Alessandro Abate. Deepltl: Learning to efficiently satisfy complex ltl specifications for multi-task rl. *arXiv preprint arXiv:2410.04631*, 2024.

Jiaming Ji, Borong Zhang, Jiayi Zhou, Xuehai Pan, Weidong Huang, Ruiyang Sun, Yiran Geng, Yifan Zhong, Josef Dai, and Yaodong Yang. Safety gymnasium: A unified safe reinforcement learning benchmark. *Advances in Neural Information Processing Systems*, 36:18964–18993, 2023.

Kishor Jothimurugan, Rajeev Alur, and Osbert Bastani. A composable specification language for reinforcement learning tasks. In *Advances in Neural Information Processing Systems*, volume 32, 2019.

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.

Yiannis Kantaros. Accelerated reinforcement learning for temporal logic control objectives. In *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 5077–5082. IEEE, 2022.

Jan Křetínskỳ, Tobias Meggendorfer, Salomon Sickert, and Christopher Ziegler. Rabinizer 4: From ltl to your favourite deterministic automaton. In *International Conference on Computer Aided Verification*, pp. 567–577. Springer, 2018.

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. IEEE, 2017.

Andrew Y Ng, Daishi Harada, and Stuart Russell. Theory and application to reward shaping. In *Proceedings of the Sixteenth International Conference on Machine Learning*, 1999.

Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. 2017.

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

Wenjie Qiu, Wensen Mao, and He Zhu. Instructing goal-conditioned reinforcement learning agents with temporal logic objectives. *Advances in Neural Information Processing Systems*, 36, 2024.

A. Ray, J. Achiam, and D. Amodei. Benchmarking safe exploration in deep reinforcement learning. GitHub, 2019. URL `https://github.com/openai/safety-gym`.

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. IEEE, 2014.

Ameesh Shah, Cameron Voloshin, Chenxi Yang, Abhinav Verma, Swarat Chaudhuri, and Sanjit A Seshia. Deep policy optimization with temporal logic constraints. *CoRR*, 2024.

Daqian Shao and Marta Kwiatkowska. Sample efficient model-free reinforcement learning from ltl specifications with optimality guarantees. *arXiv preprint arXiv:2305.01381*, 2023.

Salomon Sickert, Javier Esparza, Stefan Jaax, and Jan Křetínskỳ. Limit-deterministic büchi automata for linear temporal logic. In *International Conference on Computer Aided Verification*, pp. 312–332. Springer, 2016.

Richard S Sutton, Andrew G Barto, et al. *Reinforcement learning: An introduction*, volume 1. MIT press Cambridge, 1998.

M. Towers, J. K. Terry, A. Kwiatkowski, J. U. Balis, G. d. Cola, T. Deleu, M. Goulão, A. Kallinteris, A. KG, M. Krimmel, R. Perez-Vicente, A. Pierré, S. Schulhoff, J. J. Tai, A. T. J. Shen, and O. G. Younis. Gymnasium, 2023. URL `https://zenodo.org/record/8127025`. Accessed: 2025-01-28.

Pashootan Vaezipoor, Andrew C Li, Rodrigo A Toro Icarte, and Sheila A Mcilraith. Ltl2action: Generalizing ltl instructions for multi-task rl. In *International Conference on Machine Learning*, pp. 10497–10508. PMLR, 2021.

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

Cameron Voloshin, Abhinav Verma, and Yisong Yue. Eventual discounting temporal logic counterfactual experience replay. In *International Conference on Machine Learning*, pp. 35137–35150. PMLR, 2023.

Jun Wang, Hosein Hasanbeig, Kaiyuan Tan, Zihe Sun, and Yiannis Kantaros. Mission-driven exploration for accelerated deep reinforcement learning with temporal logic task specifications. *arXiv preprint arXiv:2311.17059*, 2023.

Zikang Xiong, Joe Eappen, Daniel Lawson, Ahmed H Qureshi, and Suresh Jagannathan. Co-learning planning and control policies using differentiable formal task constraints. *arXiv preprint arXiv:2303.01346*, 2023.

Cambridge Yang, Michael Littman, and Michael Carbin. On the (in) tractability of reinforcement learning for ltl objectives. *arXiv preprint arXiv:2111.12679*, 2021.

A More Related Work

Reinforcement Learning with Temporal Logic Objectives. Temporal logic provides a structured interface for specifying RL tasks beyond scalar Markovian rewards (Abel et al., 2021). Automaton-based formulations have been developed for both model-based control synthesis (Ding et al., 2014) and policy learning under unknown dynamics (Sadigh et al., 2014; Bozkurt et al., 2020; Kantaros, 2022). Related work compiles logical specifications into non-Markovian reward functions (Camacho et al., 2017; 2019), while reward machines expose their structure for task decomposition, shaping, and counterfactual learning (Icarte et al., 2018; 2022). For infinite-trace LTL objectives, LDBA-based approaches construct product MDPs in which satisfaction corresponds to recurrent acceptance (Hasanbeig et al., 2018; 2020; Cai et al., 2021). Dynamic and eventual discounting further connect the optimization of surrogate returns to satisfaction under suitable assumptions (Shao & Kwiatkowska, 2023; Voloshin et al., 2023). Complementary analyses establish limitations on PAC learning for general LTL objectives in unknown environments (Alur et al., 2022; Yang et al., 2021). Our work shares this infinite-horizon setting, but develops a learned reachability surrogate to provide feedback for policy optimization.

Finite-Trace Specifications and Quantitative Rewards. Another line of work formulates temporal objectives over finite execution histories. De Giacomo et al. (2019) use LTL_f/LDL_f specifications to generate automaton-based rewards evaluated on successive finite prefixes. SPECTRL (Jothimurugan et al., 2019) compiles compositional finite-rollout specifications into task monitors and shaped rewards that credit intermediate progress. Quantitative semantics also provide graded feedback through truncated LTL (Li et al., 2017) and signal temporal logic robustness (Balakrishnan & Deshmukh, 2019; Venkataraman et al., 2020). In contrast to adopting finite-trace satisfaction as the task objective, RACE preserves the infinite-trace Büchi specification and uses finite-horizon reachability as a surrogate for learning recurrent acceptance. Here, the finite reachability horizon L bounds the look-ahead to an accepting visit rather than the duration of the execution on which the specification is interpreted.

Exploration and Reward Shaping. Sparse acceptance-based rewards have motivated a range of techniques for improving exploration and credit assignment. Prior work uses accepting-frontier mechanisms (Hasanbeig et al., 2020), modular policies (Cai et al., 2021), automaton- and model-guided exploration (Wang et al., 2023), and joint planning and control with map-derived logical constraints (Xiong et al., 2023). Compositional and goal-conditioned approaches additionally exploit task structure to learn reusable behaviors (Jothimurugan et al., 2021; Jackermeier & Abate, 2024; Qiu et al., 2024). LCER (Voloshin et al., 2023) improves experience reuse through counterfactual automaton-state relabeling. More closely related to our shaping mechanism, DRL^2 (Bagatella et al., 2024) learns a Bayesian model of automaton transitions and derives potentials from the corresponding Markov reward process. CyclER (Shah et al., 2024) instead shapes rewards according to the progress of collected trajectories along accepting paths and cycles, with conditional guarantees relating its surrogate to LTL satisfaction.

RACE builds on automaton-induced potentials for exploration and complements them with state-dependent discriminator rewards learned from rollout outcomes. Rather than deriving feedback solely from automaton-level values, these discriminators use the agent's experience to distinguish states associated with successful acceptance or intermediate progress. Our stage-wise construction follows Fan & Zhu (2025), who use discriminators to match expert occupancy measures under temporal-logic constraints. We adapt this framework to exploit self-explored trajectories, learning localized progress signals without expert demonstrations.

B INFINITE-TRACE LTL SEMANTICS

For completeness, we state the standard semantics of LTL (Pnueli, 1977; Baier & Katoen, 2008). As in Fan & Zhu (2025), the specification concerns an infinite execution, even though the learning algorithm observes finite rollouts. The definitions below are background conventions, not a new logic or a finite-trace interpretation.

Formulas and labeled executions. For a finite proposition set AP, a sufficient collection of primitive operators is

$$\varphi ::= \mathsf{true} \mid p \mid \neg\varphi \mid (\varphi \wedge \psi) \mid \mathsf{X}\varphi \mid (\varphi \mathsf{U} \psi), \qquad p \in AP.$$

Disjunction and implication have their usual Boolean definitions. We write $\mathsf{false} = \neg\mathsf{true}$, $\mathsf{F}\varphi = \mathsf{true}\mathsf{U}\varphi$, and $\mathsf{G}\varphi = \neg\mathsf{F}\neg\varphi$. An environment execution $(s_i)_{i\geq 0}$ determines the word $w = \mathcal{F}(s_0)\mathcal{F}(s_1)\cdots \in (2^{AP})^\omega$. The judgment $w, i \models \varphi$ evaluates a formula on the suffix beginning at i; $w \models \varphi$ abbreviates $w, 0 \models \varphi$.

Interpretation. The constant true holds on every suffix. Atomic and Boolean formulas satisfy

$$\begin{aligned} w, i \models p &\iff p \in w_i, \\ w, i \models \neg\varphi &\iff w, i \not\models \varphi, \\ w, i \models \varphi \wedge \psi &\iff (w, i \models \varphi) \text{ and } (w, i \models \psi). \end{aligned}$$

The temporal clauses are

$$\begin{aligned} w, i \models \mathsf{X}\varphi &\iff w, i+1 \models \varphi, \\ w, i \models \varphi \mathsf{U} \psi &\iff \exists j \geq i :\ w, j \models \psi \text{ and } w, k \models \varphi \text{ for every } i \leq k < j. \end{aligned}$$

Thus $\mathsf{F}\varphi$ permits any future witness, including the current position, whereas $\mathsf{G}\varphi$ requires every suffix to satisfy φ.

Recurrence versus stabilization. The distinction relevant to RACE is

$$\begin{aligned} w \models \mathsf{GF}p &\iff \forall i \geq 0\ \exists j \geq i :\ p \in w_j, \\ w \models \mathsf{FG}p &\iff \exists i \geq 0\ \forall j \geq i :\ p \in w_j. \end{aligned}$$

The first condition allows arbitrarily long gaps between repeated visits; the second requires p to remain true after some point. Observing acceptance within a rollout supplies a training label for the reachability surrogate, but does not by itself certify either infinite-trace condition.

C PROOFS OF THEOREMS

Theorem 1 characterizes the basic acceptance-label discriminator and its lower bound on $J_L^{\infty,\gamma}$. Equation (4) bounds the infinite-time discounted mass of low-reachability states, while Theorem 2 gives a uniform sufficient condition for Büchi acceptance. We then decompose the lower-bound gap and establish an optimizer-to-satisfaction guarantee for the infinite-time objective under explicit assumptions. These are population objective results, not convergence or policy-update guarantees for the learning algorithm.

C.1 DISCRIMINATOR CHARACTERIZATION

For a stationary Markov policy π, let $q_h^\pi(x)$ be the probability of visiting F within times $0, \ldots, h$ from x. We retain the main text's notation $(t, x_t) \sim d_{T,\gamma}^\pi$ for sampled time–state pairs and use the following assumptions.

A1 (Rollouts and sampling). Let $L \geq 1$ be a finite integer reachability horizon and $T \geq 1$ the training rollout length, with $L \geq T$. Define $H(t) = \min(L, T - t) = T - t$. Rollouts $(x_0, \ldots, x_T)$ are prefixes of the same continuing process used in $J_L^{\infty,\gamma}$, with a fixed initial distribution and stationary Markov policy π; rollout truncation does not change the underlying dynamics. For fixed $0 < \gamma < 1$, sample $t \in \{0, \ldots, T-1\}$ with probability γ^t / Z_T, independently of the rollout, where

$Z_T = \sum_{t=0}^{T-1} \gamma^t = (1-\gamma^T)/(1-\gamma)$. Its joint law with x_t is $d^\pi_{T,\gamma}$, the conditional law of $d^\pi_{\infty,\gamma}$ given $t < T$. Thus $J_H(\pi) = Z_T^{-1} \sum_{t=0}^{T-1} \gamma^t \mathbb{E}_\pi[q^\pi_{H(t)}(x_t)]$. The reference sampling law is the same for discriminator fitting and J_H. Labels and J_H do not depend on the numerical value of $L \geq T$.

A2 (On-policy acceptance labels). The pairs (x_t, Y_t) are drawn from trajectories of π, without label-dependent resampling, with

$$Y_t = \mathbf{1}\{\exists j \in \{0, \dots, H(t)\} : x_{t+j} \in F\}.$$

A3 (Population optimization). The conditional label probability is realizable, and binary cross-entropy is minimized at the population level. For this idealized statement, outputs in $[0, 1]$ are allowed, including the limiting probabilities 0 and 1.

These assumptions concern the basic acceptance-label formulation. The stage-progress labels in Sec. 4.3 define a different classification target, analyzed in Appendix C.5.

Equivalent weighted loss under uniform sampling. Let $\mathbb{E}_{\text{uniform},\pi}$ denote expectation over a rollout of π and a uniformly sampled time $t \in \{0, \dots, T-1\}$, without label-dependent resampling. With $\ell_{\text{BCE}}(u, y) = -y \log u - (1-y)\log(1-u)$, the reference loss is exactly

$$\begin{aligned}\mathcal{L}_\gamma(q_\psi) &= \mathbb{E}_{d^\pi_{T,\gamma}}[\ell_{\text{BCE}}(q_\psi(x_t), Y_t)] \\ &= \mathbb{E}_{\text{uniform},\pi}\left[\frac{T\gamma^t}{Z_T}\ell_{\text{BCE}}(q_\psi(x_t), Y_t)\right].\end{aligned}$$

Here t is the original rollout time, not the time since the beginning of a segment. The factor T/Z_T does not affect the minimizer. The weighting aligns the starting-time distribution with J_H; it does not discount the event defining Y_t, correct behavior-policy mismatch, or undo label-dependent selection. Unweighted pooled training is covered by the mismatch extension below rather than assumed to satisfy A1–A2.

Proof of Theorem 1.

Proof. **(i) Expected discriminator identity.** For a fixed sampled state x, put $p(x) = \Pr_\pi(Y_t = 1 \mid x_t = x)$ under the sampling law above. The conditional cross-entropy

$$-p(x)\log q_\psi(x) - (1-p(x))\log(1-q_\psi(x))$$

is minimized at $q_\psi(x) = p(x)$, using the usual limiting convention at 0 and 1. Conditioning additionally on the sampled time t and using the Markov property gives

$$\mathbb{E}_\pi[Y_t \mid t, x_t] = q^\pi_{H(t)}(x_t).$$

Thus, for almost every sampled x,

$$\begin{aligned}q_\psi(x) &= \mathbb{E}_\pi[Y_t \mid x_t = x] \\ &= \mathbb{E}_{(t,x_t)\sim d^\pi_{T,\gamma}}[q^\pi_{H(t)}(x_t) \mid x_t = x].\end{aligned}$$

Thus the state-only discriminator estimates reachability averaged over the remaining horizons represented at that state. Taking expectations and applying the tower property proves part (i):

$$\mathbb{E}_{(t,x_t)\sim d^\pi_{T,\gamma}}[q_\psi(x_t)] = \mathbb{E}_{(t,x_t)\sim d^\pi_{T,\gamma}}[q^\pi_{H(t)}(x_t)] = J_H(\pi).$$

(ii) Pointwise lower bound. For every t and starting state x, the event of reaching F within $H(t)$ steps is contained in the event of reaching F within L steps. Therefore $q^\pi_{H(t)}(x) \leq q^\pi_L(x)$. The conditional representation above also implies $q_\psi(x) \leq q^\pi_L(x)$ on the sampled-state support.

(iii) Objective lower bound. Since $(1-\gamma^T)/Z_T = 1-\gamma$, applying part (ii) and adding the nonnegative tail gives

$$\begin{aligned}(1-\gamma^T)J_H(\pi) &= (1-\gamma)\sum_{t=0}^{T-1}\gamma^t \mathbb{E}_\pi[q^\pi_{H(t)}(x_t)] \\ &\leq (1-\gamma)\sum_{t=0}^{T-1}\gamma^t \mathbb{E}_\pi[q^\pi_L(x_t)] \leq J^{\infty,\gamma}_L(\pi).\end{aligned}$$

Parts (ii)–(iii) are objective-level statements and do not require A2–A3's discriminator fitting assumptions. □

Why a horizon average is necessary. The same product state can appear with different remaining horizons, so a state-only predictor need not equal each individual $q^{\pi}_{H(t)}(x)$. If the predictor instead receives the horizon, population cross-entropy identifies $q^{\pi}_{h}(x)$ on the joint state–horizon support. No such architecture change is needed for the expected identity in Theorem 1. A strictly increasing nonlinear transformation preserves the ordering of individual discriminator scores, but need not preserve the ordering of their policy-dependent expectations.

Extension to pooled-policy discriminator training. We now allow the training law to differ from A1–A2's same-policy, discounted reference. For a stationary target policy π, let P_π be the law of the labeled pair $(X, Y) = (x_t, Y_t)$ from its rollouts, using A1's reference time-sampling rule and the acceptance labels above. This reference distribution defines the target; training data need not be collected under π. Let Q be the population law of labeled samples used for discriminator training from a pool of behavior-policy rollouts. It includes time sampling, example selection, class balancing, and any normalized loss weighting, and is not the finite empirical measure of a particular buffer. Define

$$\delta_{\mathsf{Q}}(\pi) := \mathrm{TV}(\mathsf{P}_\pi, \mathsf{Q}), \qquad \mathrm{TV}(P, Q) := \sup_A |P(A) - Q(A)|.$$

Thus the mismatch concerns the joint state–label distribution, not only state visitation. It also covers uniformly sampled, unweighted examples when the reference objective uses discounted time sampling.

Proposition 1 (Pooled-data extension of Theorem 1)**.** *Retain the target-policy reference sampling and finite horizons $L \geq T$ above. Suppose A3 holds for the training law Q: its conditional label probability is realizable, and $q_\psi : S^{\times} \to [0, 1]$ minimizes population binary cross-entropy under Q. Then:*

1. *The discriminator satisfies $q_\psi(x) = \mathsf{Q}(Y = 1 \mid X = x)$ for Q_X-almost every x, where Q_X is the state marginal, and*

$$\left| \mathbb{E}_{(t,x_t)\sim d^{\pi}_{T,\gamma}}[q_\psi(x_t)] - J_H(\pi) \right| \leq 2\delta_{\mathsf{Q}}(\pi).$$

2. *For every $t \in \{0, \ldots, T-1\}$ and product state x, $q^{\pi}_{H(t)}(x) \leq q^{\pi}_{L}(x)$.*

3. *$(1-\gamma^T) J_H(\pi) \leq J^{\infty,\gamma}_L(\pi)$ for every $0 < \gamma < 1$.*

Consequently,

$$(1-\gamma^T)\Big(\mathbb{E}_{(t,x_t)\sim d^{\pi}_{T,\gamma}}[q_\psi(x_t)] - 2\delta_{\mathsf{Q}}(\pi)\Big) \leq (1-\gamma^T) J_H(\pi) \leq J^{\infty,\gamma}_L(\pi).$$

When $\mathsf{Q} = \mathsf{P}_\pi$, the exact identity in Theorem 1(i) is recovered.

Proof. The conditional cross-entropy calculation in the proof of Theorem 1, now under Q, gives $q_\psi(x) = p_{\mathsf{Q}}(x) := \mathsf{Q}(Y = 1 \mid X = x)$ almost surely under Q_X. The tower property therefore yields $\mathbb{E}_{\mathsf{Q}}[q_\psi(X)] = \mathbb{E}_{\mathsf{Q}}[Y]$. Under the reference law, the Markov property gives

$$\mathbb{E}_{\mathsf{P}_\pi}[Y] = \mathbb{E}_{(t,x_t)\sim d^{\pi}_{T,\gamma}}[q^{\pi}_{H(t)}(x_t)] = J_H(\pi).$$

For any measurable predictor $D : S^{\times} \to [0, 1]$, the triangle inequality and the total-variation bound for $[0, 1]$-valued functions imply

$$\begin{aligned} |\mathbb{E}_{\mathsf{P}_\pi}[D(X)] - J_H(\pi)| &\leq |\mathbb{E}_{\mathsf{P}_\pi}[D(X)] - \mathbb{E}_{\mathsf{Q}}[D(X)]| \\ &\quad + |\mathbb{E}_{\mathsf{Q}}[D(X)] - \mathbb{E}_{\mathsf{Q}}[Y]| \\ &\quad + |\mathbb{E}_{\mathsf{Q}}[Y] - \mathbb{E}_{\mathsf{P}_\pi}[Y]| \\ &\leq 2\delta_{\mathsf{Q}}(\pi) + |\mathbb{E}_{\mathsf{Q}}[D(X)] - \mathbb{E}_{\mathsf{Q}}[Y]|\,. \end{aligned} \tag{10}$$

Set $D = q_\psi$. Its mean residual under Q is zero, proving (i). Parts (ii) and (iii) follow from event containment and the discounted-prefix identity in Theorem 1; neither argument depends on the discriminator's training distribution. Multiplying the lower estimate in (i) by $1 - \gamma^T > 0$ gives the final bound. □

For a mixture $\mathsf{Q} = \sum_j a_j \mathsf{P}_{\mu_j}$ whose components use the same discounted time sampling and acceptance labels, with $a_j \geq 0$ and $\sum_j a_j = 1$, the pooled identity is $\mathbb{E}_{\mathsf{Q}}[q_\psi(X)] = \sum_j a_j J_H(\mu_j)$. This does not identify the discriminator with reachability under the target π, nor require the pool to correspond to a single stationary mixture policy. The target-policy bound is useful when $\delta_{\mathsf{Q}}(\pi)$ is controlled; pooling alone does not make it small. In particular, a pointwise bound $q_\psi(x) \leq q_L^\pi(x)$ is not asserted for pooled data. These statements accommodate off-policy discriminator data but do not establish convergence or monotonic improvement of SAC. The lower bound and recurrence implication concern a fixed policy, irrespective of how it is trained.

C.2 Uniform Reachability and Büchi Acceptance

Why $L \geq T$ is without loss of generality. Suppose a fixed policy satisfies the uniform reachability condition for some finite integer horizon $L_0 \geq 1$ and some $\varepsilon > 0$. Set $L = \max\{L_0, T\}$. By monotonicity of reachability probabilities in the horizon,

$$\inf_{x \in \mathrm{Reach}(\pi)} q_L^\pi(x) \geq \inf_{x \in \mathrm{Reach}(\pi)} q_{L_0}^\pi(x) \geq \varepsilon.$$

Thus restricting the recurrence condition to $L \geq T$ does not strengthen the requirement that some finite horizon satisfy the uniform bound. This argument concerns existence of a qualifying horizon, not invariance of the optimizer of $J_L^{\infty,\gamma}$ when L changes. The algorithm uses only the available horizons $H(t)$ and does not need to identify a qualifying L or ε.

Infinite-time distributional connection from Theorem 1. The discounted time–state distribution $d_{\infty,\gamma}^\pi$ in the main text satisfies, for every bounded measurable function f,

$$\mathbb{E}_{(t,x_t)\sim d_{\infty,\gamma}^\pi}[f(t,x_t)] = (1-\gamma)\sum_{t=0}^{\infty} \gamma^t \mathbb{E}_\pi[f(t,x_t)].$$

In particular, $\mathbb{E}_{d_{\infty,\gamma}^\pi}[q_L^\pi(x_t)] = J_L^{\infty,\gamma}(\pi)$. For any fixed policy π and threshold $\varepsilon \in (0,1)$, the pointwise inequality

$$(1-\varepsilon)\mathbf{1}\{q_L^\pi(x) < \varepsilon\} \leq 1 - q_L^\pi(x)$$

holds because $q_L^\pi(x) \in [0,1]$. Taking expectations under $d_{\infty,\gamma}^\pi$ gives

$$\begin{aligned}(1-\varepsilon)\Pr_{(t,x_t)\sim d_{\infty,\gamma}^\pi}(q_L^\pi(x_t) < \varepsilon) &\leq \mathbb{E}_{(t,x_t)\sim d_{\infty,\gamma}^\pi}[1 - q_L^\pi(x_t)] \\ &= 1 - J_L^{\infty,\gamma}(\pi) \\ &\leq 1 - (1-\gamma^T) J_H(\pi),\end{aligned}$$

where the last inequality is Theorem 1(iii). Dividing by $1-\varepsilon > 0$ proves Eq. (4). Both reference distributions use discount γ: $d_{\infty,\gamma}^\pi$ includes all starting times, whereas $d_{T,\gamma}^\pi$ conditions on $t < T$ and renormalizes. A small discounted mass of low-reachability states does not establish a uniform lower bound over all reachable states. Nor does tightening the upper bound imply that the actual low-reachability mass decreases monotonically across policies.

Additional finite-rollout bound. Averaging the same pointwise inequality under $d_{T,\gamma}^\pi$ instead and applying Theorem 1(ii) gives

$$\Pr_{(t,x_t)\sim d_{T,\gamma}^\pi}(q_L^\pi(x_t) < \varepsilon) \leq \frac{\mathbb{E}_{d_{T,\gamma}^\pi}[1 - q_L^\pi(x_t)]}{1-\varepsilon} \leq \frac{1 - J_H(\pi)}{1-\varepsilon}.$$

For example, $J_H(\pi) \geq 0.99$ implies that at most 2% of this finite-rollout probability mass has $q_L^\pi(x_t) < 0.5$. This auxiliary bound concerns $d_{T,\gamma}^\pi$, not the infinite-time occupancy in Eq. (4), and does not control states reached only after the rollout.

Proof of Theorem 2.

Proof. Assume the infimum bound holds for a fixed finite horizon $L \geq T$ and some $\varepsilon > 0$. Let $\mathcal{F}_t$ denote the history through time t. Fix $k \geq 0$, and let

$$A_m = \{x_j \notin F \text{ for every } j = k, \ldots, k + mL\}, \qquad m \geq 0.$$

On A_m, the state x_{k+mL} is outside F. The uniform reachability assumption and the Markov property imply

$$\Pr_{\pi}(A_{m+1} \mid \mathcal{F}_{k+mL}) \le (1-\varepsilon)\mathbf{1}_{A_m}.$$

Taking expectations and iterating yields $\Pr_\pi(A_m) \le (1-\varepsilon)^m$. Hence the probability of avoiding F forever from time k is zero. Intersecting the resulting probability-one events over all integer $k \ge 0$ proves infinitely many accepting visits almost surely. □

Scope of the condition. The infimum bound must hold for the same fixed horizon L over all reachable states of the fixed policy, including states reached after previous accepting visits. A qualifying pair (L, ε) may depend on both the environment and the policy, and need not exist for an arbitrary policy. For example, a reachable absorbing state outside F has $q_L^\pi(x) = 0$ for every finite L. A large average $J_H(\pi)$ does not by itself establish the uniform condition. Not needing to know L or ε is therefore an implementation property, not a guarantee that learning achieves the theorem's premise. The result establishes a sufficient condition for recurrent acceptance, not equivalence between finite-horizon surrogate maximization and infinite-horizon satisfaction maximization.

C.3 Infinite-Time Reachability and the Rollout Gap

We compare the basic rollout objective with $J_L^{\infty,\gamma}$, using A1's fixed initial distribution, continuing dynamics, and discounted reference sampling. This comparison is independent of discriminator fitting and of the policy-learning algorithm.

An auxiliary finite-rollout comparison. For a process started from x, let $\tau_F = \inf\{j \ge 0 : x_j \in F\}$, with $\inf \emptyset = \infty$. Introduce the auxiliary quantity

$$J_{L,T}(\pi) := \mathbb{E}_{(t,x_t)\sim d^\pi_{T,\gamma}}[q_L^\pi(x_t)].$$

Event containment gives the local horizon-truncation identity

$$J_{L,T}(\pi) - J_H(\pi) = \mathbb{E}_{(t,x_t)\sim d^\pi_{T,\gamma}}\left[\Pr_{\pi,x_t}\left(H(t) < \tau_F \le L\right)\right] \ge 0. \tag{11}$$

This comparison samples only the first T states. It is not the full gap to the infinite-time objective.

Proposition 2 (Exact lower-bound gap)**.** *For any stationary Markov policy, integers $L \ge T \ge 1$, and $0 < \gamma < 1$, define*

$$\Delta(\pi) := J_L^{\infty,\gamma}(\pi) - (1-\gamma^T) J_H(\pi).$$

Then

$$\Delta(\pi) = \Delta_{\text{look}}(\pi) + \Delta_{\text{tail}}(\pi), \tag{12}$$

where

$$\Delta_{\text{look}}(\pi) := (1-\gamma)\sum_{t=0}^{T-1}\gamma^t \mathbb{E}_\pi\left[\Pr_{\pi,x_t}\left(H(t) < \tau_F \le L\right)\right], \tag{13}$$

$$\Delta_{\text{tail}}(\pi) := (1-\gamma)\sum_{t=T}^{\infty}\gamma^t \mathbb{E}_\pi[q_L^\pi(x_t)]. \tag{14}$$

Both terms are nonnegative, and $0 \le \Delta_{\text{tail}}(\pi) \le \gamma^T$. The local term is $(1-\gamma^T)(J_{L,T}(\pi) - J_H(\pi))$.

Proof. Split the sum defining $J_L^{\infty,\gamma}$ at T and use

$$(1-\gamma^T) J_H(\pi) = (1-\gamma)\sum_{t=0}^{T-1}\gamma^t \mathbb{E}_\pi[q^\pi_{H(t)}(x_t)].$$

Subtracting this prefix gives the decomposition: the difference $q_L^\pi(x_t) - q^\pi_{H(t)}(x_t)$ is precisely the probability in Eq. (13). Event containment proves nonnegativity of Δ_{look}, and $0 \le q_L^\pi \le 1$ gives the tail bounds. Equation (11) and the shared discounted prefix weights give the final identity for the local term. □

The two terms distinguish truncating the local look-ahead window from omitting starting times beyond the rollout. Because the finite and infinite objectives share the same discount, there is no additional temporal-weighting term. The identity requires no assumption that failures become visible before T. Smallness of its terms, however, is not automatic. Even if every state is accepting, $J_H = J_L^{\infty,\gamma} = 1$ and $\Delta = \gamma^T$. Thus fixed T and $\gamma \uparrow 1$ do not make the scaled lower bound tight.

C.4 Optimality of the Infinite-Time Objective

An infinite time axis alone does not make a fixed-window objective Büchi-equivalent. We give explicit conditions under which its global optimizers approach optimal satisfaction. Throughout, comparisons are within the same class Π of stationary Markov policies and use the same initial distribution. Define

$$\mathcal{A} := \{x_t \in F \text{ infinitely often}\}, \qquad B(\pi) := \Pr_\pi(\mathcal{A}), \qquad B^* := \sup_{\pi\in\Pi} B(\pi).$$

Let

$$W_t := \inf\{k \geq 0 : x_{t+k} \in F\}, \qquad N_F := \sum_{t=0}^{\infty} \mathbf{1}_{\{x_t\in F\}}.$$

The complement $\mathcal{A}^c$ comprises trajectories with finitely many accepting visits; it does not mean that no accepting state is ever reached.

B1 (Uniform waiting times on satisfying trajectories). There is $0 \leq R < \infty$, independent of π, t, L, T, γ, such that

$$\mathbb{E}_\pi[W_t \mid \mathcal{A}] \leq R \quad \text{for every } \pi \in \Pi \text{ with } B(\pi) > 0 \text{ and every } t \geq 0.$$

B2 (Uniform accepting-visit counts on failing trajectories). There is $0 \leq M < \infty$, independent of π, L, T, γ, such that

$$\mathbb{E}_\pi[N_F \mid \mathcal{A}^c] \leq M \quad \text{for every } \pi \in \Pi \text{ with } B(\pi) < 1.$$

Conditions on probability-zero events are omitted. These are additional assumptions on the environment and policy class, not consequences of discriminator training. In particular, almost-sure finiteness of N_F on $\mathcal{A}^c$ does not imply B2.

Theorem 3 (Optimizer-to-satisfaction guarantee)**.** *Under B1–B2, for every integer $L \geq 1$, $0 < \gamma < 1$, and $\pi \in \Pi$,*

$$B(\pi) - \frac{R}{L+1} \leq J_L^{\infty,\gamma}(\pi) \leq B(\pi) + (1-\gamma)(L+1)M. \tag{15}$$

If $\pi^*_{L,\gamma} \in \arg\max_{\pi\in\Pi} J_L^{\infty,\gamma}(\pi)$ exists, then

$$B^* - B(\pi^*_{L,\gamma}) \leq \frac{R}{L+1} + (1-\gamma)(L+1)M. \tag{16}$$

Consequently, these optimizers approach optimal Büchi satisfaction as $L \to \infty$, $\gamma \uparrow 1$, and $(1-\gamma)(L+1) \to 0$.

Proof. Define the trajectory score

$$S_{L,\gamma} := (1-\gamma)\sum_{t=0}^{\infty} \gamma^t \mathbf{1}_{\{W_t \leq L\}} \in [0,1].$$

The Markov property gives $\mathbb{E}_\pi[\mathbf{1}_{\{W_t\leq L\}} \mid x_t] = q_L^\pi(x_t)$, so Tonelli's theorem yields $\mathbb{E}_\pi[S_{L,\gamma}] = J_L^{\infty,\gamma}(\pi)$.

On $\mathcal{A}$, Markov's inequality and the integer-valued waiting time imply

$$\Pr_\pi(W_t > L \mid \mathcal{A}) \leq \frac{\mathbb{E}_\pi[W_t \mid \mathcal{A}]}{L+1} \leq \frac{R}{L+1}.$$

Averaging with weights $(1-\gamma)\gamma^t$ shows $\mathbb{E}_\pi[S_{L,\gamma} \mid \mathcal{A}] \geq 1 - R/(L+1)$. On $\mathcal{A}^c$, each accepting visit can belong to at most $L+1$ of the windows $\{t, \ldots, t+L\}$. Therefore, pathwise,

$$\sum_{t=0}^{\infty} \mathbf{1}_{\{W_t\leq L\}} \leq (L+1)N_F, \qquad S_{L,\gamma} \leq (1-\gamma)(L+1)N_F.$$

Decomposing the expectation over $\mathcal{A}$ and $\mathcal{A}^c$, and using nonnegativity and $S_{L,\gamma} \leq 1$, proves Eq. (15).

For any $\pi \in \Pi$, global optimality and the two sides of that bound give

$$B(\pi) - \frac{R}{L+1} \leq J_L^{\infty,\gamma}(\pi) \leq J_L^{\infty,\gamma}(\pi^*_{L,\gamma}) \leq B(\pi^*_{L,\gamma}) + (1-\gamma)(L+1)M.$$

Taking the supremum over π proves Eq. (16); attainment of B^* is not required. □

The first error term reflects windows too short to capture recurrent acceptance; the second reflects finite accepting behavior on trajectories that eventually fail. For example, $\gamma = 1-(L+1)^{-2}$ makes both terms vanish as $L \to \infty$. For a policy within ζ of the global maximum of $J_L^{\infty,\gamma}$, the same proof adds ζ to the bound. This is an objective-optimization statement, not a claim that SAC, the learned reward, or the staged extension attains such an optimizer.

The result also explains why fixed L is insufficient in general. For $L = 1$, a deterministic cycle with one accepting state every ten steps has $B = 1$ but $\lim_{\gamma\uparrow 1} J_1^{\infty,\gamma} = 2/10$: the one-step window contains acceptance at only two phases of the cycle, including the accepting state itself. A policy that reaches an absorbing accepting state with probability $1/2$ and otherwise an absorbing rejecting state has $B = 1/2$ and limiting objective $1/2$. The infinite-time objective can therefore prefer the latter policy at fixed L.

When the lower-bound gap can close. B2 controls how many accepting visits a failing trajectory makes, but not how late they occur. A sufficient condition for controlling the rollout gap is the following stronger assumption. Let

$$\tau_{\text{last}} := \sup\{t \geq 0 : x_t \in F\},$$

with $\tau_{\text{last}} = -1$ when there are no accepting visits. Assume

B2' : $\quad \mathbb{E}_\pi[\tau_{\text{last}} + 1 \mid \mathcal{A}^c] \leq C < \infty \quad$ uniformly over $\pi \in \Pi$ with $B(\pi) < 1$,

where $C \geq 0$ is independent of L, T, γ. Since $N_F \leq \tau_{\text{last}} + 1$ on $\mathcal{A}^c$, B2' implies B2 with $M = C$, but additionally controls the timing of failed accepting behavior.

Proposition 3 (Gap closure under bounded failure transients)**.** *Under B1 and B2', for $L \geq T \geq 1$ and $0 < \gamma < 1$,*

$$0 \leq \Delta(\pi) \leq \gamma^T + (1-\gamma)C + (1-\gamma^T)\frac{R\log(T+1)}{T} \quad \textit{for every } \pi \in \Pi. \tag{17}$$

Moreover, whenever a global optimizer $\pi^*_{L,\gamma}$ exists, its guarantee strengthens to

$$B^* - B(\pi^*_{L,\gamma}) \leq \frac{R}{L+1} + (1-\gamma)C. \tag{18}$$

Thus both bounds vanish if $T \to \infty$, $L \geq T$, $\gamma \uparrow 1$, and $\gamma^T \to 0$.

Proof. On a failing trajectory, any window containing acceptance starts no later than τ_{last}. Hence

$$S_{L,\gamma} \leq (1-\gamma)\sum_{t=0}^{\tau_{\text{last}}} \gamma^t = 1 - \gamma^{\tau_{\text{last}}+1} \leq (1-\gamma)(\tau_{\text{last}} + 1).$$

The sum is zero for $\tau_{\text{last}} = -1$. By B2', $\mathbb{E}_\pi[S_{L,\gamma} \mid \mathcal{A}^c] \leq (1-\gamma)C$. Replacing the failure bound in Theorem 3 by this bound gives $J_L^{\infty,\gamma}(\pi) \leq B(\pi) + (1-\gamma)C$ and proves Eq. (18).

For the gap, put $c_T = 1 - \gamma^T$ and $U_T = Z_T^{-1}\sum_{t=0}^{T-1} \gamma^t \mathbf{1}_{\{W_t \leq H(t)\}}$, so that $\mathbb{E}_\pi[U_T] = J_H(\pi)$ and $\Delta(\pi) = \mathbb{E}_\pi[S_{L,\gamma} - c_T U_T]$. On satisfying trajectories, B1 implies

$$\begin{aligned} 1 - \mathbb{E}_\pi[U_T \mid \mathcal{A}] &\leq \frac{R}{Z_T}\sum_{t=0}^{T-1} \frac{\gamma^t}{T-t+1} \\ &\leq \frac{R}{T}\sum_{h=1}^{T} \frac{1}{h+1} \leq \frac{R\log(T+1)}{T}. \end{aligned}$$

For the second inequality, γ^t decreases with t whereas $1/(T-t+1)$ increases, so their normalized weighted average is no larger than the uniform average. This follows by expanding the pairwise differences, each of which has nonpositive product. Since $S_{L,\gamma} \leq 1$,

$$\mathbb{E}_\pi[S_{L,\gamma} - c_T U_T \mid \mathcal{A}] \leq \gamma^T + c_T \frac{R \log(T+1)}{T}.$$

On failing trajectories, $U_T \geq 0$ and the bound above give $\mathbb{E}_\pi[S_{L,\gamma} - c_T U_T \mid \mathcal{A}^c] \leq (1-\gamma)C$. Combining these conditional bounds proves the upper bound in Eq. (17); Proposition 2 supplies nonnegativity. □

For example, $L \geq T$ and $\gamma = 1 - T^{-1/2}$, for $T > 1$, give $\gamma^T \leq e^{-\sqrt{T}}$ and make both bounds vanish under B1 and B2'. This regime uses the stronger failed-trajectory timing assumption. In contrast, the sufficient regime $(1-\gamma)(L+1) \to 0$ in Theorem 3 alone does not certify gap closure: with $L \geq T$, it implies $\gamma^T \to 1$. The two bounds address different errors; an accurate infinite-time surrogate need not have a tight finite-rollout lower bound. The additional assumptions above specify one regime where both properties hold.

Interpretation and limits. Equation (12) characterizes the finite-to-infinite surrogate gap, while Theorem 3 connects surrogate optimality to satisfaction under additional assumptions. A small gap alone does not ensure satisfaction optimality.

Our guarantees do not contradict prior impossibility results. Alur et al. (2022) rule out PAC-MDP guarantees for the general class of safety specifications, while Yang et al. (2021) establish that sample-efficient PAC learnability over unrestricted unknown MDPs requires a finitary LTL objective. A central obstacle is that rare transitions can substantially change infinite-horizon satisfaction while remaining unobserved in finite data. Our analysis reflects this obstacle through the environment-dependent constants R, M, C, which need not admit common finite bounds across unrestricted environments. For example, in the rare-failure construction below, $M, C \geq 1/p$, so the satisfaction-error bounds deteriorate as the failure probability p approaches zero. Moreover, our optimizer-to-satisfaction results concern global maximizers of the population surrogate; they do not establish how many samples are needed to learn such policies. Thus, the results provide conditional objective-consistency guarantees without implying environment-independent finite-sample learning guarantees.

To illustrate why a small surrogate gap is insufficient, start in an accepting state that self-loops with probability $1-p$ and otherwise enters a rejecting state permanently. A length-T rollout stays accepting with probability $(1-p)^T$, close to one for small p. Nevertheless, for every $p > 0$, failure eventually occurs almost surely because $(1-p)^n \to 0$ as $n \to \infty$. Thus apparently successful finite rollouts coexist with $B = 0$; at $p = 0$, instead, $B = 1$.

Because the reachability windows include the current state, their success probabilities are one before failure and zero afterward. The discounted geometric series therefore gives, for $L \geq T$,

$$J_L^{\infty,\gamma} = \frac{1-\gamma}{1-\gamma+\gamma p}, \qquad \Delta = J_L^{\infty,\gamma}[\gamma(1-p)]^T.$$

For $p = 10^{-6}$, $\gamma = 0.99$, and $T = 2000$, we obtain $J_L^{\infty,\gamma} \approx 0.9999$ and $\Delta \approx 1.86 \times 10^{-9}$, yet $B = 0$. The finite-rollout lower bound and infinite-time surrogate agree closely without accurately approximating satisfaction probability.

This is consistent with Proposition 3: here $\mathbb{E}[N_F \mid \mathcal{A}^c] = \mathbb{E}[\tau_{\text{last}} + 1 \mid \mathcal{A}^c] = 1/p$, so B2 and B2' require $M, C \geq 1/p$. In the numerical instance, $(1-\gamma)C \geq 10^4$, making the satisfaction-error bound uninformative. Across unrestricted environments, M, C can diverge as $p \downarrow 0$, while long waits on satisfying trajectories can likewise make R unbounded. The algorithm does not require the values of these constants. However, success on finite rollouts alone does not establish the uniform bounds assumed in our analysis.

C.5 Extension to Multistage Discriminators

We extend the discriminator characterization and pooled-data guarantees in Appendix C.1 to component-wise discriminators, and adapt the recurrence analysis in Appendix C.2 to successive

component progress. We use the shared component framework introduced in Sec. 4.2 and formalized in Appendix G. The analysis retains the original product state $x = (s, b)$: sharing a discriminator within a component does not require the corresponding state aggregation to be Markov.

Component-progress targets. Throughout this subsection, discriminator heads and their conditional laws are indexed by non-accepting components $C \in \mathcal{C} \setminus \mathcal{C}^\star$. Define

$$G_C := \bigcup_{C \prec C'} C', \qquad q^\pi_{C,h}(x) := \Pr_{\pi,x}(\exists j \in \{0, \dots, h\} : b_j \in G_C)\,.$$

This is the probability of reaching a strictly downstream component, accepting or non-accepting, within h steps. Starting in C, any first accepting visit lies in a downstream accepting component. Probabilities are evaluated under the original product-state dynamics and can differ between (s, b) and (s, b') even when $c(b) = c(b')$. For $(t, x_t) \sim d^\pi_{T,\gamma}$ with $b_t \in C$, define the suffix label

$$Y^{\mathrm{prog}}_{C,t} := \mathbf{1}\{\exists j \in \{0, \dots, H(t)\} : b_{t+j} \in G_C\}\,.$$

Let $\rho_C(\pi) = \Pr_{d^\pi_{T,\gamma}}(b_t \in C)$. Whenever $\rho_C(\pi) > 0$, let $\mathsf{P}_{\pi,C}$ be the law of $(S, Y) = (s_t, Y^{\mathrm{prog}}_{C,t})$ conditional on $b_t \in C$, and set

$$p^\pi_C(s) := \mathsf{P}_{\pi,C}(Y = 1 \mid S = s), \qquad \bar{q}^\pi_{C,L}(s) := \mathbb{E}_{d^\pi_{T,\gamma}}\left[q^\pi_{C,L}(x_t) \mid s_t = s,\ b_t \in C\right].$$

The fixed reference parameters T, γ are suppressed in these two conditional averages. Components with zero probability contribute zero to the objectives below. These analytical suffix labels need not coincide with the whole-segment labels used in the buffers.

Training distributions and reward. At iteration k, let $\mathsf{Q}_{k,C}$ be the population law of a labeled environment state (S, Y) drawn for head C from the disjoint pools in Eq. (30). This law includes behavior-policy mixtures, segment assignment, within-pool sampling, and class balancing. The queried component specifies the head, not necessarily the automaton component at which the sampled occurrence was collected. Write $q_{k,C} := q_{\psi_k,C}$ for its probability output, allowing limiting values in $[0, 1]$; a shared component-conditioned network is included in this notation.

Consider the reward

$$R^{\mathrm{st}}_{k,g}(s, b) = a(s, b) + \lambda_b g(q_{k,c(b)}(s)), \qquad \lambda_b \geq 0,$$

where a is bounded and fixed and $g : [0, 1] \to \mathbb{R}$ is increasing and K_g-Lipschitz. On live states, Eq. (9) has $a(s, b) = \textsc{Idx}(c(b))/\mathcal{N}(c(b))$ and $\lambda_b = \beta/\mathcal{N}(c(b))$, with a positive normalizer shared within each component. Set $\lambda_b = 0$ outside $\mathcal{B}_{\mathrm{live}}$ and include the fixed boundary reward in a. Terms with $\lambda_b = 0$ are omitted, so no component head is evaluated outside its domain. Define

$$\begin{aligned}
\widehat{J}^{\mathrm{st}}_{k,g}(\pi) &:= \mathbb{E}_{d^\pi_{T,\gamma}}[a(x_t) + \lambda_{b_t} g(q_{k,c(b_t)}(s_t))],\\
J^{\mathrm{st}}_{H,g}(\pi) &:= \mathbb{E}_{d^\pi_{T,\gamma}}[a(x_t) + \lambda_{b_t} g(p^\pi_{c(b_t)}(s_t))],\\
J^{\mathrm{st}}_{L,T,g}(\pi) &:= \mathbb{E}_{d^\pi_{T,\gamma}}[a(x_t) + \lambda_{b_t} g(\bar{q}^\pi_{c(b_t),L}(s_t))].
\end{aligned}$$

All three use the discounted finite-rollout reference $d^\pi_{T,\gamma}$. Its state and component frequencies depend on π and are not held fixed in policy comparisons. The upper comparator uses a component-conditional average, not a statewise identification of the shared discriminator with $q^\pi_{C,L}(s, b)$.

For the probability-output reward in Eq. (9), $g(u) = \tanh(u)$ and $K_g = 1$. The analysis also covers a logit-based implementation: if $q_{\psi,C} = \sigma(f_C)$ and the reward uses $\tanh(f_C)$, then

$$g(u) = \tanh\left(\log \frac{u}{1-u}\right) = \frac{2u - 1}{u^2 + (1-u)^2}, \qquad K_g = 4,$$

with continuous limits at 0 and 1, since $g'(u) = 4u(1-u)/(u^2 + (1-u)^2)^2 \leq 4$.

Theorem 4 (Multistage discriminator characterization)**.** *Fix a stationary Markov policy π. Assume that the collection of training-law posteriors is jointly realizable by the discriminator architecture and that each $q_{k,C}$ minimizes population binary cross-entropy under $\mathsf{Q}_{k,C}$. Then the following hold.*

1. *Under the actual training law, the optimal probability output satisfies* $q_{k,C}(s) = \mathsf{Q}_{k,C}(Y = 1 \mid S = s)$ *almost surely under its state marginal.*

2. *If* $\mathsf{Q}_{k,C} = \mathsf{P}_{\pi,C}$ *for every non-accepting component with* $\rho_C(\pi) > 0$*, then* $q_{k,C}(s) = p^\pi_C(s)$ *and, for almost every sampled* s*,*

$$p^\pi_C(s) = \mathbb{E}_{d^\pi_{T,\gamma}}\left[q^\pi_{C,H(t)}(x_t) \mid s_t = s,\ b_t \in C\right] \le \bar{q}^\pi_{C,L}(s).$$

   Consequently, $\widehat{J}^{\mathrm{st}}_{k,g}(\pi) = J^{\mathrm{st}}_{H,g}(\pi) \le J^{\mathrm{st}}_{L,T,g}(\pi)$ *for every finite* $L \ge T$*.*

3. *For* $a = 0$*,* $\lambda_b = 1$ *on* $\mathcal{B}_{\mathrm{live}}$*, and* $g(u) = u$*, the ideal objective reduces to*

$$J^{\mathrm{prog}}_H(\pi) := \mathbb{E}_{d^\pi_{T,\gamma}}\left[\mathbf{1}_{\{b_t \in \mathcal{B}_{\mathrm{live}}\}} q^\pi_{c(b_t),H(t)}(x_t)\right] \le \mathbb{E}_{d^\pi_{T,\gamma}}\left[\mathbf{1}_{\{b_t \in \mathcal{B}_{\mathrm{live}}\}} q^\pi_{c(b_t),L}(x_t)\right].$$

   Terms outside $\mathcal{B}_{\mathrm{live}}$ *are defined as zero.*

Proof. For each head and environment-state input, conditional binary cross-entropy is minimized by the conditional positive-label probability, proving (i). Under the matching law in (ii), condition on the time and the full product state. The Markov property gives

$$\mathbb{E}_\pi[Y^{\mathrm{prog}}_{C,t} \mid t, x_t] = q^\pi_{C,H(t)}(x_t) \quad \text{when } b_t \in C.$$

Conditioning further on $s_t = s$ and $b_t \in C$ and applying the tower property yields the stated average over time and the original automaton states within C. Since $H(t) \le L$, pointwise event containment gives $q^\pi_{C,H(t)}(x_t) \le q^\pi_{C,L}(x_t)$. Taking the same conditional average proves $p^\pi_C(s) \le \bar{q}^\pi_{C,L}(s)$. Monotonicity of g and nonnegativity of λ_b preserve this inequality even when λ_b varies within a component; integration proves (ii). For (iii), $a = 0$, the coefficient is constant on live states, and g is the identity, so the tower property removes the component-conditional averages. □

Relation to segment labels. Part (i) applies to the specified disjoint pools. Part (ii) additionally requires matching suffix labels and reference sampling. A segment may have progressed before a sampled occurrence, and failure-priority assignment, incomparable-component exclusion, or class balancing may change its conditional label probability. Component sharing also averages over automaton states that can have different future behavior. These distinctions are included in $\mathsf{Q}_{k,C}$ and are not eliminated by small training loss. Moreover, visiting a downstream component need not entail reaching F, so this local comparison does not generally lower-bound the accepting-set objective $J^{\infty,\gamma}_L$ of Theorem 1.

Proposition 4 (Pooled-data approximation of staged rewards)**.** *At iteration* k*, hold the discriminator collection fixed. For each* $C \in \mathcal{C} \setminus \mathcal{C}^\star$*, let* $\mathcal{L}_{k,C}$ *be its population binary cross-entropy risk under* $\mathsf{Q}_{k,C}$*, and assume*

$$\mathcal{L}_{k,C}(q_{k,C}) - \inf_{f:S\to[0,1]} \mathcal{L}_{k,C}(f) \le \eta_{k,C}, \qquad \mathrm{TV}(\mathsf{P}_{\pi,C}, \mathsf{Q}_{k,C}) \le \delta_{k,C}(\pi)$$

for every candidate policy with $\rho_C(\pi) > 0$*, where the infimum is over measurable functions* $f : S \to [0,1]$*. Let* $\lambda^{\max}_C := \max_{b\in C} \lambda_b < \infty$ *and define*

$$E^{\mathrm{st}}_{k,g}(\pi) := K_g \sum_{C\in\mathcal{C}\setminus\mathcal{C}^\star} \rho_C(\pi)\lambda^{\max}_C \left(\sqrt{\eta_{k,C}/2} + 2\delta_{k,C}(\pi)\right).$$

Then

$$\left|\widehat{J}^{\mathrm{st}}_{k,g}(\pi) - J^{\mathrm{st}}_{H,g}(\pi)\right| \le E^{\mathrm{st}}_{k,g}(\pi). \tag{19}$$

Proof. Fix a component with positive reference probability. Write $P = \mathsf{P}_{\pi,C}$, $Q = \mathsf{Q}_{k,C}$, $p_P(s) = P(Y = 1 \mid S = s)$, and $p_Q(s) = Q(Y = 1 \mid S = s)$. Excess binary cross-entropy is the expected KL divergence between $\mathrm{Bern}(p_Q(S))$ and $\mathrm{Bern}(q_{k,C}(S))$. The Bernoulli Pinsker inequality and

Cauchy–Schwarz imply $\mathbb{E}_Q|q_{k,C}(S) - p_Q(S)| \le \sqrt{\eta_{k,C}/2}$. For measurable $u : S \to [-1, 1]$, the absolute-value dual representation gives

$$\begin{aligned}\mathbb{E}_P|q_{k,C}(S) - p_P(S)| &= \sup_{|u|\le 1} \mathbb{E}_P[u(S)(q_{k,C}(S) - Y)] \\ &\le \sup_{|u|\le 1} \mathbb{E}_Q[u(S)(q_{k,C}(S) - Y)] + 2\,\mathrm{TV}(P, Q) \\ &= \mathbb{E}_Q|q_{k,C}(S) - p_Q(S)| + 2\,\mathrm{TV}(P, Q) \\ &\le \sqrt{\eta_{k,C}/2} + 2\delta_{k,C}(\pi).\end{aligned}$$

The factor two follows because $u(S)(q_{k,C}(S) - Y) \in [-1, 1]$. After the common structural term cancels, condition the reward difference on $b_t \in C$. Since $\lambda_{b_t} \le \lambda_C^{\max}$,

$$\mathbb{E}_{d^\pi_{T,\gamma}}[\lambda_{b_t}|g(q_{k,C}(s_t)) - g(p^\pi_C(s_t))| \mid b_t \in C] \le \lambda_C^{\max} K_g\, \mathbb{E}_P|q_{k,C}(S) - p_P(S)|.$$

Multiply by $\rho_C(\pi)$ and sum over non-accepting components. Boundary rewards agree exactly and contribute no approximation error. The bound allows general state-dependent coefficients λ_b; Eq. (9) uses the special case in which they are constant within each component. □

The mismatch concerns the joint environment-state–label law and includes replay drift, temporal weighting, component-pool selection, sampling occurrences from other components, and whole-segment versus suffix labels. The result applies to separate component heads or a shared component-conditioned network whenever the excess-risk assumptions hold. Discriminators may be refitted at the next iteration. This is a reward-approximation result, not a policy-improvement or Büchi satisfaction guarantee.

A sufficient connection to recurrent acceptance. Use the component-to-acceptance path length $r(C)$ from Appendix G, and set

$$m := \max\big(\{1\} \cup \{r(C) : C \in \mathcal{C}\}\big).$$

The graph is finite and acyclic, so these ranks are finite and $C \prec C'$ implies $r(C') < r(C)$. Give accepting states rank zero and live product states (s, b) rank $r(c(b))$. This rank is used only until the next accepting visit; it may restart at a larger value after acceptance.

Proposition 5 (Uniform component progress implies recurrent acceptance)**.** *Fix a stationary policy whose original product-state process is strong Markov. Assume that every reachable non-accepting product state has $b \in \mathcal{B}_{\mathrm{live}}$ and, for some finite integer $\ell \ge 1$ and $p \in (0, 1]$,*

$$q^\pi_{c(b),\ell}((s, b)) \ge p \quad \textit{for every reachable } (s, b) \notin F.$$

Then, with $L_{\mathrm{st}} := \max\{T, m\ell\}$,

$$\inf_{x\in\mathrm{Reach}(\pi)} q^\pi_{L_{\mathrm{st}}}(x) \ge p^m > 0, \qquad \Pr_\pi(x_t \in F \textit{ infinitely often}) = 1.$$

Proof. Induct on j to show that every reachable product state of rank at most j reaches F within $j\ell$ steps with probability at least p^j. At rank zero it is already accepting. From a non-accepting $x = (s, b)$ of rank at most j, put $C = c(b)$ and let σ be the first hitting time of G_C. The assumed progress bound gives $\Pr_{\pi,x}(\sigma \le \ell) \ge p$. At that stopping time the process is accepting or in a strictly downstream component, hence has rank at most $j - 1$. Conditional on the stopped history, the strong Markov property and induction give probability at least p^{j-1} of acceptance within a further $(j - 1)\ell$ steps. Thus the probability of acceptance within $j\ell$ steps is at least p^j. Taking $j = m$ and enlarging the horizon to L_{st} gives the uniform accepting-set bound; Theorem 2 yields recurrent acceptance. □

The component construction supplies an acyclic rank, but it does not establish the uniform progress bound or avoidance of structurally rejecting states. These conditions concern every reachable original product state, including those reached after earlier accepting visits, not component-average discriminator scores. Neither the segment labels nor the choice of β guarantees them. Accepting cycles remain compatible with the result because each accepting visit starts a new round.

D Main Algorithm

D.1 SAC Updates and Replay

We adopt SAC (Haarnoja et al., 2018) as a practical off-policy optimizer with the accepting-state target assignment of Fan & Zhu (2025). Appendices C.1 and C.5 characterize pooled-data discriminator training and the staged extension. The objective-level results do not guarantee SAC convergence or improvement at each update.

Q-function update. Write $x = (s, b)$ and $x' = (s', b')$. We update $Q_\theta(x, a)$ by minimizing the Bellman residual over transitions sampled from replay buffer B:

$$J_Q(\theta) = \mathbb{E}_{(x,a,x')\sim B}\left[\frac{1}{2}\big(Q_\theta(x,a) - \widehat{y}\big)^2\right]. \tag{20}$$

For $C := 1/(1-\gamma)$, the target is

$$\widehat{y} = \begin{cases} C, & x \in F, \\ R(x,x') + \gamma\, \mathbb{E}_{a'\sim\pi_\phi(\cdot|x')}\left[Q_{\bar\theta}(x',a') - \alpha \log \pi_\phi(a' \mid x')\right], & x \notin F. \end{cases} \tag{21}$$

Here R is the combined reward in Eq. (8), $Q_{\bar\theta}$ is the target critic, and α is the entropy temperature. The first branch applies to the *current* accepting state and assigns the same target to every action; it replaces, rather than adds a bonus to, the ordinary Bellman target. This is a value boundary for learning and does not require physically resetting the environment at acceptance.

Policy update. The actor minimizes the usual replay-based SAC loss:

$$J_\pi(\phi) = \mathbb{E}_{x\sim B}\left[\mathbb{E}_{a\sim\pi_\phi(\cdot|x)}\left[\alpha \log \pi_\phi(a \mid x) - Q_\theta(x,a)\right]\right]. \tag{22}$$

Replay and segment labels. Algorithm 1 adapts the acceptance-segmented replay procedure of Fan & Zhu (2025), replacing demonstration initialization with agent-collected data and adding automaton-kernel learning for exploration. Alongside SAC replay B, RACE maintains a buffer B_C for every component, including accepting singletons, and a structural-failure buffer $B_\perp$, using the shared stages of Sec. 4.2. Appendix G specifies segment assignment and disjoint pools B_C^+, B_C^- for non-accepting heads. Each nonempty segment receives exactly one buffer assignment, while all collected transitions enter B. The policy and critic continue to use the full product state (s, b). Splitting discriminator data does not itself change the one-step SAC target; Eq. (21) specifies its acceptance-boundary modification. The matching discounted sampling in Theorem 1 is an idealized assumption, not a consequence of the Bellman discount.

D.2 Separating Credit Across Accepting Visits

We first isolate the effect of the value boundary. Fix a stationary policy π, a bounded reward $r(x, x')$, and $0 < \gamma < 1$, and omit entropy regularization. The reward is held fixed throughout this analysis; it may be a frozen learned reward. Write $V_C^\pi(x) := \mathbb{E}_{a\sim\pi(\cdot|x)}[Q_C^\pi(x,a)]$. The boosted policy-evaluation equations are

$$Q_C^\pi(x,a) = \begin{cases} C, & x \in F, \\ \mathbb{E}[r(x,x_1) + \gamma V_C^\pi(x_1) \mid x_0 = x, a_0 = a], & x \notin F, \end{cases}$$

Thus V_C^π is only a policy average of the action-value function, not an additional learned network, and $Q_C^\pi(x,a) = C$ for all actions at acceptance implies $V_C^\pi(x) = C$ there.

Proposition 6 (Accepting-state boundary decouples continuation values)**.** *Let $\tau_F := \inf\{k \ge 0 : x_k \in F\}$. For a bounded boundary function $v : F \to \mathbb{R}$, let V_v^π be the bounded solution with $V_v^\pi = v$ on F and the same policy and Bellman reward r as above off F. Then*

$$V_C^\pi(x) = \mathbb{E}_{\pi,x}\left[\sum_{k=0}^{\tau_F-1} \gamma^k r(x_k, x_{k+1}) + \gamma^{\tau_F} C\, \mathbf{1}_{\{\tau_F<\infty\}}\right], \tag{23}$$

$$V_C^\pi(x) - V_v^\pi(x) = \mathbb{E}_{\pi,x}\left[\gamma^{\tau_F}\big(C - v(x_{\tau_F})\big)\mathbf{1}_{\{\tau_F<\infty\}}\right]. \tag{24}$$

Algorithm 1 RACE Main Algorithm

Require: MDP $\mathcal{M}$, LDBA $\mathcal{L}$, transition kernel $\mathcal{K}$, fixed boundary rewards $r_\top, r_\bot$
Initialize policy π_ϕ, critic Q_θ, replay buffer B
Compute $\mathcal{B}_\bot$, components $\mathcal{C}$, accepting components $\mathcal{C}^\star$, map c, order $\prec$, and non-accepting scores $\textsc{SIdx}, \mathcal{N}$ (Sec. 4.2)
Initialize buffers B_C for all $C \in \mathcal{C}$, failure buffer $B_\bot$, and discriminators $q_{\psi,C}$ for $C \in \mathcal{C} \setminus \mathcal{C}^\star$
for each iteration **do**
  Sample trajectories $\mathcal{T}$ by executing π_ϕ in $\mathcal{M}^\times$ $\{\mathcal{M}^\times \equiv M \times \mathcal{L}\}$
  Add all collected transitions to B
  Split rollouts at new accepting visits; truncate at first rejecting entry and discard empty fragments
  for each resulting segment τ **do**
    $z \leftarrow \textsc{MaxStage}(\tau) \in \mathcal{C} \cup \{\bot\}$ via Eq. (29)
    $B_z \leftarrow B_z \cup \{\tau\}$
  for each gradient step for the discriminators **do**
    for each $C \in \mathcal{C} \setminus \mathcal{C}^\star$ with nonempty B_C^+ and B_C^- **do**
      Draw equal-size batches from B_C^+ (label 1) and B_C^- (label 0) via Eq. (30)
      Update $q_{\psi,C}$ using binary cross-entropy
  for each gradient step for the policy π_ϕ **do**
    Compute $R(x, x')$ via Eq. (8), evaluating the component reward at x'
    Update Q_θ via Eq. (20) and π_ϕ via Eq. (22) by SAC via samples from B
    Update posterior $P(\mathcal{K} \mid \mathcal{T})$ via Eq. (27)
    Compute high-level values $\bar{V}$ via Eq. (6)

On $\{\tau_F = \infty\}$, the sum is the infinite discounted sum and each boundary term is defined to be zero. In particular, the boosted value is independent of the continuation function v when the reward and pre-acceptance policy are fixed.

Proof. The boundary Bellman operator is a contraction with modulus at most γ, so each bounded fixed point exists and is unique. Iterate its recursion until $\tau_F \wedge n$. The remainder on $\{\tau_F > n\}$ is bounded in absolute value by $\gamma^n \|V_v^\pi\|_\infty$, which tends to zero. Bounded rewards and boundary values allow dominated convergence, giving the stopped-return expression with terminal value v. Substituting $v \equiv C$ gives Eq. (23). Subtracting the two expressions proves Eq. (24). □

Taking v to be the ordinary discounted continuation value on F explains the design. Poor performance after acceptance can lower the value of a segment that already reaches F. Replacing that continuation by C credits the completed segment without importing the current difficulty of later rounds. Subsequent segments can still provide training data. This is a statement about the boundary with fixed rewards, not about discriminator refitting, policy convergence, or preservation of the continuing objective's optimizer.

Entropy in the implemented SAC target. For the soft target in Eq. (21), the relevant policy average is instead

$$V^\pi_{C,\mathrm{soft}}(x) := \mathbb{E}_{a\sim\pi(\cdot|x)}\left[Q^\pi_{C,\mathrm{soft}}(x,a) - \alpha \log \pi(a \mid x)\right].$$

At an ideal fixed point, $Q^\pi_{C,\mathrm{soft}}(x,a) = C$ on F implies

$$V^\pi_{C,\mathrm{soft}}(x) = C + \alpha\mathcal{H}_\pi(x), \qquad \mathcal{H}_\pi(x) := -\mathbb{E}_{a\sim\pi(\cdot|x)}[\log \pi(a \mid x)].$$

Off F, its recursion uses the expected reward $r(x, x_1)+\alpha\mathcal{H}_\pi(x)$. If these entropy terms are bounded, the same stopped-return argument applies with this reward and boundary $C + \alpha\mathcal{H}_\pi$ on F. Hence the boost removes the learned future task-return continuation, but the action entropy at acceptance remains in the soft backup. The entropy-free results below do not silently identify this soft boundary with C.

D.3 When the Boundary Removes a Preference for Lingering

The next result makes the credit-assignment mechanism concrete. It considers a local decision problem with a fixed reward and continuation value, rather than asserting a global guarantee for the staged learner.

Proposition 7 (Preventing continuation-induced completion avoidance)**.** *Consider a non-accepting state x with two actions. Waiting receives reward r_{w} and returns to x; completing receives reward r_F and enters a fixed accepting state. Omit entropy and fix the ordinary continuation value at that accepting state to v. If*

$$r_F + \gamma v < \frac{r_{\mathrm{w}}}{1-\gamma} < r_F + \gamma C, \tag{25}$$

then waiting forever is the unique optimal stationary policy at x under the ordinary continuation, whereas completing immediately is the unique optimal stationary policy under the boosted boundary. This comparison includes stationary randomized policies.

Proof. Write G for the value of completing immediately, and let $p \in [0,1]$ be the stationary probability of waiting. The value at x is

$$V_p = p(r_{\mathrm{w}} + \gamma V_p) + (1-p)G = \frac{p r_{\mathrm{w}} + (1-p)G}{1-\gamma p}.$$

Consequently,

$$\frac{dV_p}{dp} = \frac{r_{\mathrm{w}} - (1-\gamma)G}{(1-\gamma p)^2}.$$

The first inequality in Eq. (25) makes this derivative positive for $G = r_F + \gamma v$, so $p = 1$ is uniquely optimal. The second makes it negative for $G = r_F + \gamma C$, so $p = 0$ is uniquely optimal. □

For a unit completion reward $r_F = 1$ and $C = 1/(1-\gamma)$, one has $r_F + \gamma C = C$. Thus the second inequality holds whenever $r_{\mathrm{w}} < 1$. For example, with $\gamma = 0.99$, $r_{\mathrm{w}} = 0.1$, and $v = 0$, indefinite waiting has value 10, ordinary completion has value 1, and boosted completion has value 100. The boundary reverses the local preference even though the reward is unchanged. For the actual staged and shaped reward, Eq. (25) must be checked with the corresponding rewards; neither a unit completion reward nor a subunit waiting reward follows automatically from Eq. (8). This proposition explains a possible benefit of the modification, not universal preference for acceptance or a Büchi satisfaction guarantee.

D.4 Idealized Compatibility with the Reachability Objective

The constant C also has an exact reachability interpretation under an additional, horizon-consistent idealization. Fix π and suppose that, with $h \geq 1$ steps remaining, the reward is the exact next-state reachability probability

$$r_h^\pi(x, x') := q_{h-1}^\pi(x').$$

This subsection uses the identity reward, without potential shaping or entropy, and a critic indexed by the remaining horizon. Write $V_h^\pi(x) := \mathbb{E}_{a\sim\pi(\cdot|x)}[Q_h^\pi(x,a)]$. Define $Q_0^\pi(x,a) := C\mathbf{1}_F(x)$, and for $h \geq 1$ let

$$Q_h^\pi(x,a) = \begin{cases} C, & x \in F, \\ \mathbb{E}\left[q_{h-1}^\pi(x_1) + \gamma V_{h-1}^\pi(x_1) \mid x_0 = x, a_0 = a\right], & x \notin F, \end{cases}$$

In particular, a non-accepting state at the rollout cutoff has value zero. These are analytical assumptions; the implemented state-only discriminator and critic are not assumed to satisfy them.

Proposition 8 (Boosted values under exact horizon-consistent rewards)**.** *Under the preceding idealization, for every $h \in \{0, \ldots, T\}$,*

$$V_h^\pi(x) = C q_h^\pi(x).$$

Therefore the policy-dependent averaged boosted value satisfies

$$\mathbb{E}_{(t,x_t)\sim d_{T,\gamma}^\pi}\left[V_{H(t)}^\pi(x_t)\right] = C J_H(\pi). \tag{26}$$

Since $C > 0$ is policy-independent, global maximizers of this averaged boosted-value objective are exactly the global maximizers of J_H, whenever they exist.

Proof. At $h = 0$, both sides are $C\mathbf{1}_F(x)$; at any accepting state they equal C for every horizon. At a non-accepting state, induction and $1 + \gamma C = C$ give

$$\begin{aligned} V_h^\pi(x) &= \mathbb{E}_{\pi,x}\left[q_{h-1}^\pi(x_1) + \gamma C q_{h-1}^\pi(x_1)\right] \\ &= C\,\mathbb{E}_{\pi,x}[q_{h-1}^\pi(x_1)] = C q_h^\pi(x), \end{aligned}$$

where the last equality is the finite-horizon reachability recursion off F. Averaging with the defining weights of J_H proves Eq. (26) and the optimizer statement. □

The result explains why the accepting-state constant can be compatible with a reachability objective, but it is not an optimizer-preservation theorem for the implemented SAC update. Theorem 1 identifies an *average* discriminator score, not a pointwise horizon-conditioned reward. Moreover, Eq. (26) averages values over $d_{T,\gamma}^\pi$; maximizing only the initial-state boosted value is a different objective. The implementation also uses staged rewards, shaping, entropy, and replay. Neither this idealization nor the boundary-separation and lingering results establish that its actor updates maximize J_H.

E Automaton Kernel and Value Computation

E.1 Optimistic Priors

Following Bagatella et al. (2024), we model each row of the aggregate kernel K as a categorical distribution. Let $\mathcal{J}_i$ index the successors allowed by the LDBA from b_i, with $m_i = |\mathcal{J}_i|$. Define the Dirichlet prior only over these coordinates:

$$(K_{i,j})_{j\in\mathcal{J}_i} \sim \mathrm{Dir}\big((\alpha_{i,j})_{j\in\mathcal{J}_i}\big), \qquad K_{i,j} = 0 \quad (j \notin \mathcal{J}_i),$$

where $\alpha_{i,j} = \alpha/m_i > 0$ for $j \in \mathcal{J}_i$. Forbidden coordinates are fixed at zero, not included as zero-parameter Dirichlet coordinates. A row with a single allowed successor is deterministic. The prior strength $\alpha > 0$ controls adaptation; larger values yield slower posterior adaptation.

Given the automaton projection $T = (b_t)_{t=0}^N$ of a collected N-step trajectory, the posterior under this surrogate categorical model is

$$P(K_i \mid T) \propto P(T \mid K_i)P(K_i) \tag{27}$$

In particular, for each LDBA state $b_i \in \mathcal{B}$,

$$(K_{i,j})_{j\in\mathcal{J}_i} \mid T \sim \mathrm{Dir}\big((\alpha_{i,j} + c_{i,j})_{j\in\mathcal{J}_i}\big),$$

where $c_{i,j}$ counts observed transitions from b_i to b_j, and forbidden entries remain zero. Updating $P(K \mid T)$ thus induces a posterior over auxiliary MRPs.

E.2 Boundary-Aware Value Computation

Because $R^\times(b, b')$ is a transition reward, its expected one-step reward under K is the vector

$$r_K(b) = \sum_{b'} K(b,b')R^\times(b,b'), \qquad M_K(b,b') = K(b,b')\gamma^\times(b,b').$$

Thus Eq. (6) has matrix form $\bar{V}_K = r_K + M_K\bar{V}_K$. The discount multiplies the continuation value, not the current transition reward, consistent with $\Gamma_1 = 1$ and the preceding-transition product defining Γ_t.

Let

$$Z_K = \{b \notin \mathcal{B}^* : K(b,b) = 1\}, \qquad U_K = \mathcal{B} \setminus Z_K.$$

Every state in Z_K generates only zero-reward self-loops, so its expected return is zero. This includes rejecting absorbing sinks. More generally, transitions into or within $\mathcal{B}_\perp$ receive zero automaton reward, and all structurally rejecting states have zero expected return. The full matrix $I - M_K$ is singular when Z_K is nonempty, but no full inverse is needed. Instead, impose the zero-value boundary and solve

$$\bar{V}_K\big|_{Z_K} = 0, \qquad \bar{V}_K\big|_{U_K} = \left(I - (M_K)_{U_K U_K}\right)^{-1} r_K\big|_{U_K}. \tag{28}$$

If U_K is empty, the value is identically zero.

Existence and uniqueness. For the transition discount in Eq. (5) and $0 < \gamma < 1$, an accepting-state row of M_K sums to γ. For each non-accepting $b \in U_K$, its row sums to

$$K(b,b) + \gamma\big(1 - K(b,b)\big) < 1,$$

because $K(b,b) < 1$. This remains true for transitions between distinct states in the same component: their stage reward is zero, but their discount is still γ. Restricting to columns in U_K can only decrease these row sums. Since the automaton is finite, $\kappa_K := \|(M_K)_{U_K U_K}\|_\infty < 1$, where the matrix norm is the maximum absolute row sum. Hence the reduced matrix is a contraction and its inverse exists, with

$$(I - (M_K)_{U_K U_K})^{-1} = \sum_{n=0}^{\infty} (M_K)^n_{U_K U_K}.$$

Multiplying this series by $r_K|_{U_K}$ gives the expected discounted transition-reward sum. Consequently, Eq. (28) is the unique Bellman solution satisfying the zero-value boundary and equals the value defined in the main text.

Shaping discount and sink transitions. The potential remains $R_{\mathrm{pot}}(b,b') = \gamma \bar{V}_K(b') - \bar{V}_K(b)$, using the ordinary discount γ of the SAC return. The transition-dependent $\gamma^\times$ is used only to construct the automaton potential. Entering a rejecting sink $b_\perp$ therefore gives $R_{\mathrm{pot}}(b, b_\perp) = -\bar{V}_K(b)$, while its self-loop gives zero shaping reward. This automaton potential is separate from the accepting-state Q-value boundary in Appendix D.

F Potential-Based Shaping and Relation to Reward Machines

Relation to Reward Machines. Potential-based reward shaping (PBRS) has been previously applied in the context of Reward Machines (Icarte et al., 2022). In that work, the shaping potential is derived from automaton-level value functions obtained via value iteration over a reward machine.

Our approach shares the intuition of using automaton-level structure to guide learning, but differs in a key aspect. In (Icarte et al., 2022), the shaping potential is derived from a *fixed reward model*, leading to a deterministic transition kernel after value iteration. In contrast, our method employs a *learned, adaptive kernel*, which evolves with the policy and data. This adaptivity enables more effective guidance in complex environments but comes at the cost of weakening strict policy-invariance guarantees.

Policy invariance of the shaping term. For a fixed kernel and discriminator reward, the following finite-horizon result establishes policy invariance when the shaping potential is zeroed at the terminal time.

Proposition 9 (Potential-based shaping invariance (finite horizon))**.** *Fix a finite horizon $\ell \geq 1$, a kernel K, a bounded discriminator reward R_{disc}, and $0 < \gamma < 1$. Let $\bar{V}_K$ be the automaton-level value function in Eq. (6), and define the time-indexed potential*

$$\Phi_t(b) = \begin{cases} \bar{V}_K(b), & 0 \leq t < \ell, \\ 0, & t = \ell. \end{cases}$$

For $t = 0, \ldots, \ell - 1$, define

$$R_{\mathrm{pot},t}(b_t, b_{t+1}) = \gamma \Phi_{t+1}(b_{t+1}) - \Phi_t(b_t).$$

Under the same product-state dynamics and initial distribution $\mu_0^\times$, the expected ℓ-step discounted returns with rewards R_{disc} and $R_{\mathrm{disc}} + R_{\mathrm{pot},t}$ have identical optimal policies.

Proof. For any rollout $(x_t)_{t=0}^{\ell}$, where $x_t = (s_t, b_t)$, the cumulative discounted shaping reward telescopes:

$$\begin{aligned} \sum_{t=0}^{\ell-1} \gamma^t R_{\mathrm{pot},t}(b_t, b_{t+1}) &= \sum_{t=0}^{\ell-1} \gamma^t \big(\gamma \Phi_{t+1}(b_{t+1}) - \Phi_t(b_t)\big) \\ &= -\Phi_0(b_0) + \gamma^\ell \Phi_\ell(b_\ell) = -\bar{V}_K(b_0). \end{aligned}$$

Taking expectations under the common initial distribution shifts every policy's return by the same constant $-\mathbb{E}_{\mu_0^\times}[\bar{V}_K(b_0)]$. Therefore policy ordering and the set of optimal policies are unchanged. Only Φ_ℓ is zero: the fixed function $\bar{V}_K$ retains its value at earlier occurrences of the same automaton state. □

Discussion. The finite-horizon result is conditional on the terminal-time convention above; it does not assert that the implementation applies this mask at rollout cutoffs. During training, the kernel (and hence $\bar{V}_K$) also evolves. The shaping term therefore serves as adaptive guidance rather than an unconditional policy-invariance guarantee for the full learning algorithm. Empirically, this adaptivity significantly improves learning efficiency, highlighting a practical trade-off between theoretical guarantees and performance.

Conditional stability under kernel updates. Updating K makes the shaping reward non-stationary. For count-based Dirichlet updates, the posterior mean is

$$K_{i,j}^{(k)} = \frac{\alpha_{i,j} + c_{i,j}^{(k)}}{\sum_{\ell \in \mathcal{J}_i}(\alpha_{i,\ell} + c_{i,\ell}^{(k)})} \quad (j \in \mathcal{J}_i), \qquad K_{i,j}^{(k)} = 0 \quad (j \notin \mathcal{J}_i).$$

If per-iteration count additions are bounded and every updated row has effective count mass at least $N^{(k)}$, then

$$\|K_{k+1} - K_k\|_\infty = O\Big(1/N^{(k)}\Big).$$

Transferring this rate to values requires control of the reduced system in Eq. (28). Specifically, assume a common boundary set $Z_{K_k} = Z$ and a uniform contraction bound $\|(M_{K_k})_{UU}\|_\infty \le \kappa < 1$ on $U = \mathcal{B} \setminus Z$. With fixed $R^\times$, $\gamma^\times$, and $R_{\max} = \max_{b,b'} |R^\times(b,b')| < \infty$, write $v_k = \bar{V}_{K_k}|_U$. Subtracting the two reduced Bellman equations gives

$$v_{k+1} - v_k = \big(I - (M_{K_{k+1}})_{UU}\big)^{-1} \Big[(r_{K_{k+1}} - r_{K_k})|_U + \big((M_{K_{k+1}})_{UU} - (M_{K_k})_{UU}\big) v_k\Big].$$

The inverse norm is at most $(1-\kappa)^{-1}$, $\|v_k\|_\infty \le R_{\max}/(1-\kappa)$, and

$$\|r_{K_{k+1}} - r_{K_k}\|_\infty \le R_{\max}\|K_{k+1} - K_k\|_\infty, \qquad \|M_{K_{k+1}} - M_{K_k}\|_\infty \le \|K_{k+1} - K_k\|_\infty.$$

Consequently,

$$\delta_k := \|\bar{V}_{K_{k+1}} - \bar{V}_{K_k}\|_\infty \le \frac{R_{\max}(2-\kappa)}{(1-\kappa)^2}\|K_{k+1} - K_k\|_\infty = O\Big(1/N^{(k)}\Big).$$

Thus value drift vanishes if these conditions hold and $N^{(k)} \to \infty$. Individual invertibility, bounded rewards, or growing data alone do not establish a uniform contraction margin. This conditional stability result is not a convergence guarantee for the overall SAC training process.

Experiment Results. To better understand this trade-off, we implemented a baseline RACE_VI where the potential function is constructed via value iteration over the LDBA, using the reward and discounting formulation in Eq. equation 5, analogous to Icarte et al. (2022). The transition kernel is defined directly from the LDBA structure. For each automaton state b_i, we consider only transitions permitted by the automaton. If there are m such successor states, each is assigned probability $1/m$, while all other transitions are assigned probability zero. This yields a fixed transition kernel K that is uniform over valid LDBA transitions and respects the automaton's structure. The result is shown in Fig. 5. Empirically, we observe that such a fixed kernel performs significantly worse than our adaptive kernel, suggesting that adaptivity is critical for effective exploration and exploitation in our setting.

G LDBA-GUIDED EXPLOITATION

The stage-wise discriminator and acceptance-segmented buffer design follow Fan & Zhu (2025). Here we formulate stages as components, specify disjoint progress-label pools, and use only agent-collected segments. The definitions below support both exploration and exploitation in Secs. 4.2–4.3; Appendix C.5 analyzes the resulting predictors rather than expert-occupancy matching.

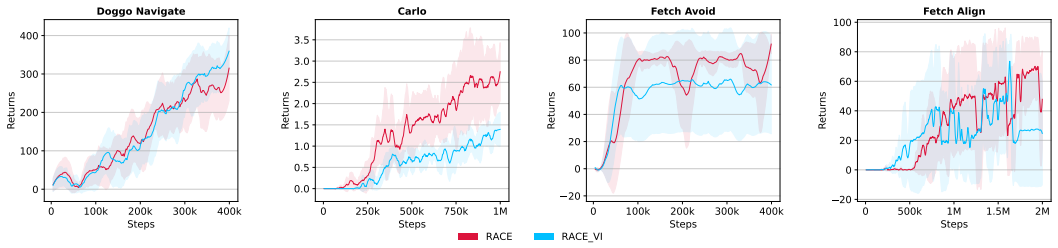

Figure 5: Comparison with VI-Based Potential (averaged over 10 random seeds using returns under eventual discounting)

Components and within-round progress. In the original LDBA graph, define

$$\mathcal{B}_\perp := \{b \in \mathcal{B} : \text{no directed path from } b \text{ reaches } \mathcal{B}^\star\}, \qquad \mathcal{B}_{\text{live}} := \mathcal{B} \setminus (\mathcal{B}^\star \cup \mathcal{B}_\perp).$$

Paths of length zero are allowed when testing reachability, so accepting states never belong to $\mathcal{B}_\perp$. This failure set includes rejecting absorbing states and closed rejecting cycles. Let E be the directed edge set of the original LDBA graph, including both labeled and ϵ-transitions. Form the auxiliary within-round graph

$$G_{\text{round}} = (\mathcal{B} \setminus \mathcal{B}_\perp, E_{\text{round}}), \qquad E_{\text{round}} := \{(b, b') \in E : b, b' \notin \mathcal{B}_\perp,\ b \notin \mathcal{B}^\star\}.$$

Thus all outgoing edges from accepting states, including self-loops, are removed before computing strongly connected components. Let $\mathcal{C}$ be these components and let $c : \mathcal{B} \setminus \mathcal{B}_\perp \to \mathcal{C}$ map each state to its component. The accepting components are exactly the terminal singletons

$$\mathcal{C}^\star := \{\{b_f\} : b_f \in \mathcal{B}^\star\}; \qquad \mathcal{B}_{\text{live}} = \bigcup_{C \in \mathcal{C} \setminus \mathcal{C}^\star} C.$$

Cutting edges only in this auxiliary graph prevents paths through acceptance from merging stages; the original LDBA retains all accepting cycles and transitions into subsequent rounds.

The condensation of G_{round} is a directed acyclic graph: it has an edge $C \to C'$ for distinct components exactly when $(b, b') \in E_{\text{round}}$ for some $b \in C$ and $b' \in C'$. Write $C \prec C'$ when a directed path leads from C to a distinct C', and $C \preceq C'$ when $C = C'$ or $C \prec C'$. These relations compare components, including accepting singletons, and describe progress within one accepting round. The original automaton state b remains part of the policy and critic input and of the exploration kernel; component sharing does not assert that $(s, c(b))$ is a Markov state.

Shared structural scores. Every non-accepting component reaches an accepting component: truncate an original path to acceptance at its first accepting visit, so none of its edges is removed by the boundary cut. A source component has no incoming edge in the component DAG. Paths used for non-accepting scores may start at any source, rather than requiring a path from the original initial state.

Measure path lengths in edges. For every $C \in \mathcal{C}$, define $r(C)$ as the longest path length from C to any accepting component. Accepting components have $r(C) = 0$, while non-accepting components have $r(C) \geq 1$. For $C \in \mathcal{C} \setminus \mathcal{C}^\star$, define $\textsc{Idx}(C)$ as the longest source-to-C path length and $\mathcal{N}(C)$ as the longest path from a source to any accepting component through C. A non-accepting source has index zero. Since a DAG path ending at C and one starting at C cannot share another vertex, longest prefixes and suffixes concatenate, giving

$$\mathcal{N}(C) = \textsc{Idx}(C) + r(C) > 0, \qquad 0 \leq \frac{\textsc{Idx}(C)}{\mathcal{N}(C)} < 1.$$

For any $C \prec C'$, $r(C') \leq r(C) - 1$. If both components are non-accepting, then also $\textsc{Idx}(C') \geq \textsc{Idx}(C) + 1$. Consequently,

$$\frac{\textsc{Idx}(C')}{\mathcal{N}(C')} > \frac{\textsc{Idx}(C)}{\mathcal{N}(C)}.$$

All states within a non-accepting component share these scores; accepting rewards are assigned separately and do not require evaluating this ratio at terminal components. Scalar scores do not make

incomparable components comparable: pool selection uses $\prec$, not numerical score comparisons. The exploration reward in Eq. (5) credits a live successor only when the predecessor lies outside its component, including entry from an accepting boundary. It is zero for movement within a non-accepting component and structural failure, and one on acceptance. The original-state discount $\gamma^{\times}$ is unchanged, as required by Appendix E.2's contraction argument.

Segment assignment. Split collected rollouts at accepting visits. An accepting state at the start of a segment is the boundary of the preceding round, not completion of the new round. For discriminator storage, terminate a segment at its first entry into $\mathcal{B}_\perp$ and discard any subsequent rejecting tail; keep all collected transitions in SAC replay. Discard empty fragments. For a resulting segment $\tau = ((s_0, b_0), \ldots, (s_n, b_n))$, $n \geq 1$, define

$$\textsc{MaxStage}(\tau) = \begin{cases} \perp, & b_n \in \mathcal{B}_\perp \quad \text{(structural failure)}, \\ c(b_n), & \text{otherwise} \quad \text{(new accepting visit or rollout cutoff)}. \end{cases} \tag{29}$$

For the second case, omit any accepting start state when considering within-round progress. After suppressing repetitions within a component, the remaining component sequence forms a path in the DAG, ending in an accepting singleton or a non-accepting cutoff component. Its endpoint is therefore the unique furthest visited component of the current round. Store τ in $B_{c(b_n)}$; distinct accepting components have distinct buffers. In the first case, store τ in $B_\perp$: failure takes precedence over intermediate progress. Accepting visits at the segment start do not determine its buffer assignment.

Component-wise discriminators and disjoint pools. Maintain B_C for every $C \in \mathcal{C}$, but train a probability predictor $q_{\psi,C}(s)$ only for $C \in \mathcal{C} \setminus \mathcal{C}^\star$. Its positive and negative segment pools are

$$B_C^+ = \bigcup_{C \prec C'} B_{C'}, \qquad B_C^- = B_\perp \cup \bigcup_{C' \preceq C} B_{C'}. \tag{30}$$

Positive segments end strictly downstream of C, in either a non-accepting component or an accepting singleton. Negative segments fail or end in C or an upstream component. Segments ending in incomparable components, including incomparable accepting singletons, are excluded from both pools. Any segment visiting C and subsequently completing acceptance within that round ends in a downstream accepting component and is therefore included in B_C^+. The pools are disjoint because $\prec$ is a strict partial order and each segment has a unique assignment. Draw equal-size positive and negative minibatches, uniformly over stored environment-state occurrences within each pool, and assign labels 1 and 0, respectively. If either pool is empty, skip that head's update. Sampled occurrences need not have automaton state in C: these are whole-segment labels, not necessarily future-progress labels from the sampled state. Their population interpretation and sampling mismatch are treated in Appendix C.5.

Reward evaluation. For $b \in \mathcal{B}_{\text{live}}$, evaluate $q_\psi^{\text{st}}(s, b) = q_{\psi,c(b)}(s)$ in Eq. (9), using the same component scores $\textsc{Idx}(c(b))$ and $\mathcal{N}(c(b))$ as exploration. The normalizer is positive by construction. At accepting and rejecting states, no head is evaluated: use fixed reward parameters $R_{\text{disc}}((s, b)) = r_\top$ for $b \in \mathcal{B}^\star$ and $R_{\text{disc}}((s, b)) = r_\perp$ for $b \in \mathcal{B}_\perp$. These task rewards are distinct from the accepting-state Q-value boundary in Appendix D. The coefficient β controls the learned correction; the structural ordering does not itself establish policy improvement or recurrent acceptance.

Shared component-conditioned architecture. Default heads have independent parameters for each non-accepting component. Following the state-conditioned alternative studied by Fan & Zhu (2025), we also consider parameter sharing, now conditioned on components: $q_{\psi,C}(s) = f_\psi([s, e_C])$, with a common probability-output network f_ψ and one-hot encoding $e_C \in \mathbb{R}^{|\mathcal{C} \setminus \mathcal{C}^\star|}$. The queried component selects its training pools; it need not match the component at which an occurrence was collected. Accepting components have buffers but no discriminator heads.

The results in Fig. 6 indicate that though using a shared discriminator leads to slightly lower performance compared to multiple discriminators, it still maintains reasonable performance, providing a viable trade-off between scalability and performance.

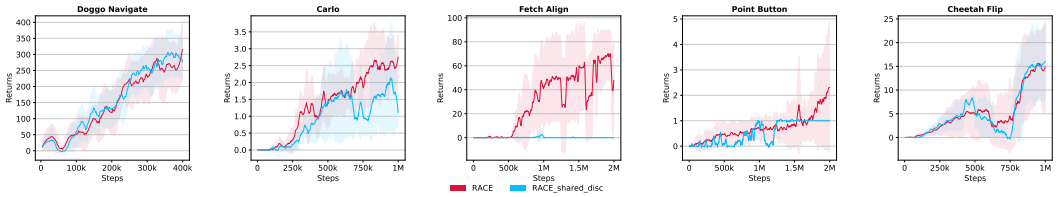

Figure 6: Ablation results for RACE using a shared component-conditioned discriminator, averaged over 10 random seeds using returns under eventual discounting.

H Environments and Tasks

The continuous-control suite includes the Carlo, Doggo, FlatWorld, Fetch, FetchPlace Button, Cheetah Flip, and SixteenRooms task designs used by Fan & Zhu (2025). We describe them here to make the evaluation self-contained, retaining citations to their underlying simulators. Point tasks use Safety-Gym (Ji et al., 2023). The settings below describe the present evaluation; sharing a task does not imply that all training horizons or protocols coincide with prior work. The episode horizons below describe the main experiments; Appendix M additionally evaluates the final policies over extended horizons.

Carlo Carlo tests recurrent waypoint tracking under bicycle-model vehicle dynamics. Each 500-step episode starts on the road centerline at angular position $\pi/4$. Steering and throttle form a two-dimensional action in $[-1, 1]^2$; the five observation coordinates encode position, velocity, and heading. The vehicle must travel counterclockwise through the blue waypoint regions wp_0 and wp_1, without entering the gray crash region. The specification is $\mathsf{GF}(wp_0 \wedge \mathsf{XF}wp_1) \wedge \mathsf{G}\neg crash$.

Doggo The SafetyGym Doggo platform (Ray et al., 2019) couples navigation with 12-DoF quadruped control. Its policy receives 66 observation coordinates and outputs 12 controls. We use 500-step episodes for two objectives: *Doggo Avoid*, $\mathsf{F}a \wedge \mathsf{G}\neg z$, requires reaching a distant target through a narrow safe corridor; *Doggo Navigate*, $\mathsf{F}(a \wedge \mathsf{XF}b)$, requires reaching two circular zones in sequence. The former stresses safety-constrained exploration, while the latter requires credit assignment across consecutive goals.

FlatWorld FlatWorld isolates spatial exploration from complex robot dynamics. A point agent starts at $(-1, -1)$ and moves according to $s_{t+1} = s_t + a_t/10$, with $s_t \in \mathbb{R}^2$ and $a_t \in [0, 1]^2$. Colored regions determine the propositions. The evaluation task *FlatWorld Cycle* uses $\mathsf{GF}(r \wedge \mathsf{XF}(g \wedge \mathsf{XF}y)) \wedge \mathsf{G}\neg b$: red, green, and yellow must recur in the prescribed order, while blue is unsafe. Each rollout contains at most 50 steps.

Fetch Using the Fetch simulator (de Lazcano et al., 2023), we control a 7-DoF manipulator through four-dimensional end-effector actions. The two tasks use 50-step episodes. *Fetch Avoid* provides a 10-dimensional observation and requires extending toward the green target without crossing the red lateral regions, expressed by $\mathsf{F}a \wedge \mathsf{G}\neg z$. *Fetch Align* uses a 45-dimensional observation that includes the three cubes. The arm must arrange them horizontally at the table center in the order specified by $\mathsf{F}(a \wedge \mathsf{XF}(b \wedge \mathsf{XF}c))$.

FetchPlace *FetchPlace Button* adds a changing placement target to the Fetch manipulation setting (de Lazcano et al., 2023). Pressing a button reveals the next goal location, and the arm must transfer a block there before returning to the button. The observation contains 31 coordinates, including button and goal information; the controller has four action dimensions and an episode budget of 500 steps. The seven-joint arm is tasked with grasping and repeated goal–button visits through $\mathsf{F}grasp \wedge \mathsf{GF}(GoalReached \wedge \mathsf{XF}ButtonReached)$.

HalfCheetah The *Cheetah Flip* task replaces forward locomotion with repeated frontflips in HalfCheetah (Towers et al., 2023). The six-degree-of-freedom robot moves in a vertical plane. Body-angle intervals define propositions b and d, corresponding respectively to front-leg and back-leg support. The formula $\mathsf{GF}(b \wedge \mathsf{XF}d)$ requires repeated alternation between these configurations within the continuing task; evaluation episodes are truncated after 1000 steps.

Table 1: Average active wall-clock training time in hours. The lowest runtime for each environment is highlighted in bold.

Environment	Horizon	RACE	LCER	DRL2	Cycler
Doggo Avoid	400K	**5.02**	8.22	5.61	6.63
Doggo Navigate	400K	**6.04**	8.11	7.40	8.11
Carlo	1M	12.29	7.94	8.00	**5.03**
Fetch Avoid	400K	4.76	3.61	**3.54**	4.40
Fetch Align	2M	**30.99**	40.16	55.59	44.54
FetchPlace Button	4M	71.70	**52.10**	57.29	–
Cheetah Flip	1M	13.99	**5.28**	11.65	7.59
Point Zone	2M	66.95	87.29	100.15	**48.44**
Point Button	2M	**59.55**	61.15	150.99	71.13
FlatWorld Cycle	2M	**26.72**	37.32	–	–
SixteenRooms	1.5M	35.51	**13.45**	36.56	22.19

Point Point robot from (Ji et al., 2023) is the agent, which has a two-dimensional action space $\mathcal{A} = [-1, 1]^2$. In *PointZone*, the robot must navigate the environment, which includes three differently colored goal regions while avoiding the 8 hazards. The robot receives an observation of lidar data that detects the presence of nearby objects at each timestep. The LTL task description instructs the agent to oscillate amongst visiting the three colored regions. In *PointButton*, this domain is a more challenging variant of the *PointZone* environment, in which the agent must press a set of small buttons distributed across a larger workspace while avoiding cube-shaped gremlins. The LTL specification requires the agent to press two designated buttons infinitely often, without making contact with any gremlin. The buttons are more scattered. The two tasks are defined by the following specifications: $\mathsf{GF}(button2 \wedge \mathsf{XF}button0) \wedge \neg grelim$ (repeatedly visit button0 and button2 while avoiding grelims) for *PointZone*, and $\mathsf{GF}(b \wedge \mathsf{XF}g \wedge \mathsf{XF}r) \wedge \neg harzard$ (visit red, blue, and green zones cyclically while avoiding harzards) for *PointButton*.

SixteenRooms This task combines the room layout of Jothimurugan et al. (2021) with the continuous point-agent setup used by Fan & Zhu (2025). Sixteen 8×8 rooms form a 4×4 grid connected by corridors. We index locations as $\texttt{room}_{i,j}$, counting i from left to right and j from bottom to top. The agent starts near the center of the bottom-left room, with random perturbations of its initial state.

The specification offers two recurrent routes while prohibiting wall collisions:

$$\begin{aligned}\Psi = \Big(&\mathsf{GF}(\texttt{room}_{1,2} \wedge \mathsf{XF}(\texttt{room}_{1,4} \wedge \mathsf{XF}(\texttt{room}_{3,4} \wedge \mathsf{XF}\texttt{room}_{3,2}))) \\ &\vee \mathsf{GF}(\texttt{room}_{3,1} \wedge \mathsf{XF}(\texttt{room}_{3,3} \wedge \mathsf{XF}(\texttt{room}_{3,4} \wedge \mathsf{XF}(\texttt{room}_{4,4} \wedge \mathsf{XF}\texttt{room}_{4,3}))))\Big) \\ &\wedge \mathsf{G}\neg\texttt{Wall}.\end{aligned}$$

The disjunction permits the controller to select a route rather than requiring both. Each branch specifies its own ordered, recurring room visits; the safety conjunct applies regardless of that choice.

I WALLCLCOCK TIME

Table 1 compares the observed training time of each method. RACE is fastest on five environments and remains competitive on several others. Since the experiments were executed under different hardware loads, the values represent observed runtime rather than a controlled efficiency benchmark.

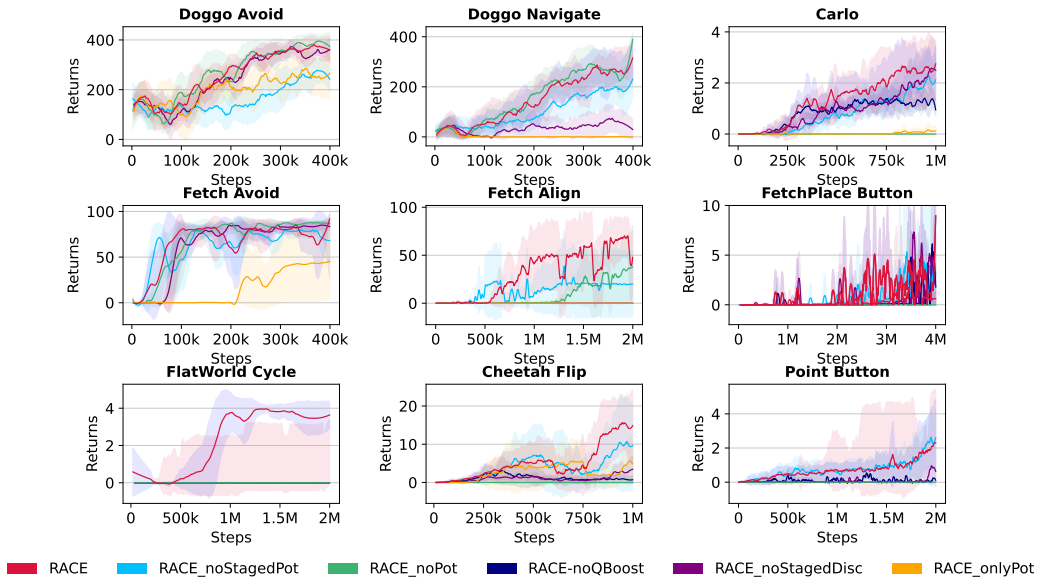

Figure 7: Ablation results for RACE averaged over 10 random seeds are reported using returns under eventual discounting.

J Ablation Study

We investigate how (a) staged potential rewards $R_{\text{pot}}((s,b))$ for exploration and (b) multistage discriminator rewards $R_{\text{disc}}((s,b))$ for exploitation each contribute to the overall performance. For part (a), we perform RACE_noPot using only $R_{\text{disc}}((s,b))$ for policy optimization, thereby removing all potential-based shaping. We further perform RACE_noStagedPot using a variant of the potential reward that omits the stage-aware shaping terms based on $\text{SIDX}(c(b_t))/\mathcal{N}(c(b_t))$ and the corresponding transition-dependent discounting $\gamma^{\times}$ in Eq. equation 5. In this variant, the agent no longer obtains intermediate guidance that reflects progress through the LDBA structure in exploration. For part (b), we perform the ablation RACE_noStagedDisc, which removes the stage-wise structure from the discriminator reward. In this variant, the discriminator is trained only to distinguish between successful and failed rollout segments, without conditioning on the LDBA stage or progress information in exploitation. We also include the ablation RACE_noQBoost, which removes the direct assignment of a large target Q value $\frac{1}{1-\gamma}$ at accepting product states. The results, depicted in Fig. 7, confirm the importance of both the staged potential rewards and the multistage discriminator structure in guiding the agent toward satisfying the LTL specification. Removing the stage-aware potential signal significantly slows exploration and reduces the agent's ability to make consistent progress through the automaton. Similarly, eliminating the staged discriminator severely degrades learning, as the agent no longer receives informative intermediate guidance aligned with acceptance progress, and consequently fails to reliably learn to satisfy the specification.

The ablation results clearly demonstrate the complementary roles of the main components of RACE. Removing the staged potential reward substantially weakens exploration, often preventing the agent from making steady progress through the LDBA; this effect is especially pronounced in long-horizon tasks such as Doggo Navigate, Fetch Align, Cheetah Flip, FlatWorld Cycle, Point Zone, and Point Button, where learning largely collapses without structured progress signals. Removing the multistage discriminator also leads to severe performance drops, as the reward signal becomes too coarse to provide guidance aligned with intermediate acceptance milestones, causing learning to stall even when exploration is possible, as seen in Doggo Navigate, Fetch Align, and Cheetah Flip. In contrast, removing the accepting-state value boost does not entirely prevent learning but consistently slows convergence and lowers final performance across tasks such as Cheetah Flip, FlatWorld Cycle, and Point Button, indicating that optimistic value targets help the agent commit to completing accepting behaviors rather than lingering near them. Overall, these results show that staged potential shaping, structured discriminator rewards, and ac-

ceptance value boosting address distinct challenges in long-horizon LTL-constrained reinforcement learning, and that their combination is critical for both efficient training and strong final satisfaction performance.

K Scalability Study

In the main paper, the largest LDBA considered has 10 states (16-room environment). To further evaluate scalability, we construct a family of LTL specifications in a FlatWorld environment with randomly positioned colored regions (r, g, y, b). We increase complexity by extending the length of sequential and cyclic subgoal compositions. Concretely, we consider:

a. Small (5 states):

$$\mathsf{F}(r \wedge \mathsf{XF}(g \wedge \mathsf{XF}(y \wedge \mathsf{XF}(b \wedge \mathsf{XF}r))))$$

b. Large (10 states):

$$\mathsf{F}(r \wedge \mathsf{XF}(g \wedge \mathsf{XF}(y \wedge \mathsf{XF}(b \wedge \mathsf{XF}(r \wedge \mathsf{XF}(g \wedge \mathsf{XF}(y \wedge \mathsf{XF}b)))))))$$

c. Larger (15 states):

$$\mathsf{F}(r \wedge \mathsf{XF}(g \wedge \mathsf{XF}(y \wedge \mathsf{XF}(b \wedge \mathsf{XF}(r \wedge \mathsf{XF}(g \wedge \mathsf{XF}(y \wedge \mathsf{XF}(b \wedge \mathsf{XF}(r \wedge \mathsf{XF}g)))))))))$$

Here $r \to g$ informally denotes visiting r then next g, not logical implication.

The Fig. 8 shows that RACE maintains stable performance as the LDBA size increases, with a gradual decrease in return as the task becomes more complex. Despite longer sequences and more stages, RACE continues to learn effectively, demonstrating robustness to moderately large specifications. In the 15-state setting, we observe higher variance, including temporary drops during training, which we attribute to increased task difficulty from longer sequential dependencies and more stages.

K.1 Sequential Scaling with Avoidance

We also consider a two-dimensional continuous-control environment with colored circular regions arranged along a fixed line. The agent must visit the r, g, and y regions in order while avoiding the unsafe b regions. Episodes last at most 50 steps. We extend the required sequence as follows; the labels count visits, not LDBA states.

a. Three states

$$\mathsf{F}(r \wedge \mathsf{XF}(g \wedge \mathsf{XF}y)) \wedge \mathsf{G}\neg b$$

b. Four states:

$$\mathsf{F}(r \wedge \mathsf{XF}(g \wedge \mathsf{XF}(y \wedge \mathsf{XF}r))) \wedge \mathsf{G}\neg b$$

c. Five states:

$$\mathsf{F}(r \wedge \mathsf{XF}(g \wedge \mathsf{XF}(y \wedge \mathsf{XF}(r \wedge \mathsf{XF}g)))) \wedge \mathsf{G}\neg b$$

d. Six states:

$$\mathsf{F}(r \wedge \mathsf{XF}(g \wedge \mathsf{XF}(y \wedge \mathsf{XF}(r \wedge \mathsf{XF}(g \wedge \mathsf{XF}y))))) \wedge \mathsf{G}\neg b$$

As shown in Figure 9, RACE learns the three-, four-, five-, and six-state specifications, reaching recent evaluation returns near 46, 45, 44, and 43 respectively.

L Stochastic Environments

Noise or stochastic transitions can introduce variance in the discriminator targets, since reachability becomes probabilistic. However, this does not make the reward misleading. Our discriminator is trained to estimate the likelihood of reaching accepting states, which remains well-defined under stochastic dynamics. In practice, the discriminator averages over this noise across many rollout segments (including replayed data), and remains a stable signal for exploitation.

For stochastic transitions, 1) we introduce sticky actions. At each step, with probability p, the environment ignores the agent's chosen action and instead repeats the previous action. This mechanism

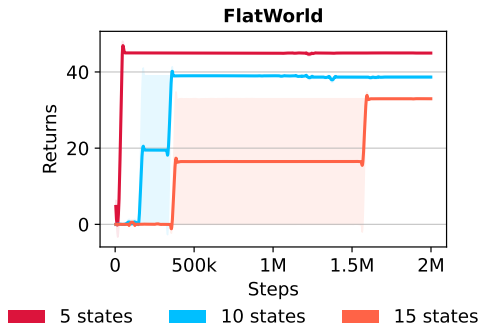

Figure 8: Performance on FlatWorld for large tasks.

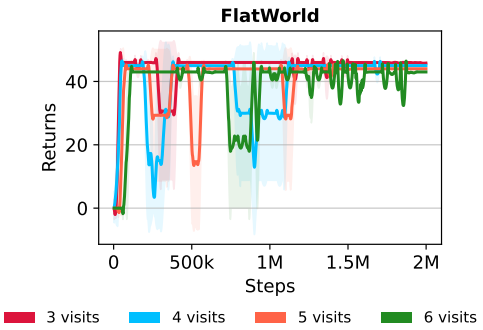

Figure 9: Performance on FlatWorld for large tasks.

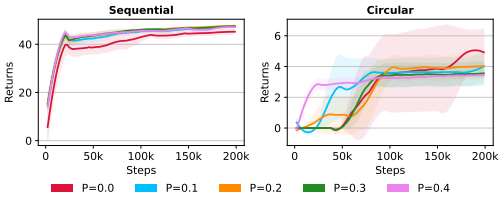

Figure 10: Evaluating RACE under different levels of sticky actions: At each step with probability p, the environment ignores the agent's chosen action and instead repeats the previous action.

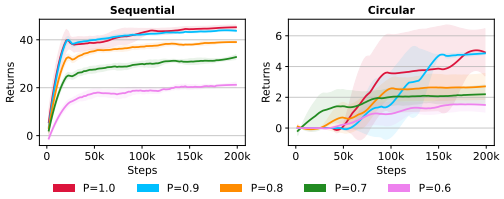

Figure 11: Evaluating RACE under different levels of stochastic state transitions: At each step in the MDP, the intended action is executed with probability p, while the remaining $1 - p$ is distributed among the other actions.

introduces both stochasticity and a mild form of partial observability, since transitions now depend on past actions. The test environments are borrowed from the sequential and circular tasks from Fig 15 in DRL2 (Bagatella et al., 2024). In the sequential tasks, the environment consists of horizontally aligned grid regions, each labeled with a different atomic proposition (a, b, c, ...). The agent starts from the left and must visit these regions in order. In the circular tasks, the environment is arranged in a cross-shaped layout with multiple labeled regions (a, b, c, d) surrounding a central obstacle. The agent must repeatedly follow a cyclic ordering of regions while avoiding the obstacle. As shown in Fig. 10, despite a slight performance drop compared to the deterministic setting, RACE remains robust and continues to learn effectively under stochastic dynamics. Moreover, RACE substantially outperforms DRL2 and LCER in these environments.

2) We also introduce stochastic actions. Specifically, at each environment step, the agent selects an intended action, which is executed with probability p. With the remaining probability $1 - p$, the environment instead executes one of the other available actions, chosen uniformly at random. By varying p, we obtain different levels of stochasticity: larger values of p correspond to more deterministic transitions, while smaller values introduce stronger action uncertainty. This experiment tests whether RACE can still learn policies that satisfy the LTL specification when the agent's actions do not always lead to the expected next state. As shown in Fig. 11, the results show that RACE remains effective under moderate stochasticity, indicating that its LDBA-guided shaping and exploration mechanism can tolerate noisy transitions and continue to guide the agent toward accepting states.

3) We also introduce stochastic labeling of atomic propositions. For noisy labeling, we inject uncertainty into atomic propositions by assigning the correct label for each state with probability 90%, and distributing the remaining 10% over incorrect labels in the Sequence and Circular tasks. In this setting, performance degrades significantly (Sequence: return ~ 8, Circular: return ~ 1). We attribute this to the fact that our current formulation operates in an MDP setting, whereas noisy propositions induce partial observability.

Table 2: Completed accepting rounds per episode for final policies evaluated at the original training horizon and an extended horizon, without further training. Entries are mean ± standard deviation across 10 training seeds. Horizon pairs give the original and extended episode lengths in environment steps and are shared across methods within each environment.

Environment (original/extended steps)	Method	Original-horizon rounds	Extended-horizon rounds
PointZone (1,000/1,500)	RACE	2.75 ± 0.68	3.39 ± 1.34
	DRL[2]	0.00 ± 0.00	0.00 ± 0.00
	LCER	0.00 ± 0.00	0.00 ± 0.00
	CyclER	0.00 ± 0.00	0.00 ± 0.00
Carlo (500/750)	RACE	2.40 ± 1.27	4.17 ± 2.11
	DRL[2]	1.00 ± 0.00	1.73 ± 0.85
	LCER	0.00 ± 0.00	0.00 ± 0.00
	CyclER	0.00 ± 0.00	0.00 ± 0.00
FlatWorld Cycle (50/90)	RACE	3.33 ± 1.49	5.67 ± 2.80
	DRL[2]	0.00 ± 0.00	0.00 ± 0.00
	LCER	2.97 ± 1.76	3.00 ± 2.00
	CyclER	0.00 ± 0.00	0.00 ± 0.00
Cheetah Flip (1,000/1,500)	RACE	14.63 ± 9.47	17.63 ± 12.06
	DRL[2]	2.50 ± 5.00	4.46 ± 12.38
	LCER	0.00 ± 0.00	0.00 ± 0.00
	CyclER	0.00 ± 0.00	0.00 ± 0.00

M Finite-Horizon Progress

We complement eventual-discounted returns with a direct evaluation of repeated task execution on *PointZone*, *Carlo*, *FlatWorld Cycle*, and *Cheetah Flip*. For each method, we evaluate the final policies obtained from 10 training seeds, counting visits to LDBA-accepting states within each episode. In these tasks, each accepting visit marks completion of the required task sequence and is counted as one accepting round. This count applies neither discounting nor learned or potential-based reward shaping.

To assess whether the learned behavior continues beyond the episode lengths used for training, we evaluate each final policy without further training at both the original training horizon and an extended horizon. A horizon specifies the maximum number of environment steps per episode, and all methods share the same pair of horizons within each environment. Table 2 reports completed rounds as mean ± standard deviation across the 10 seeds. Both columns evaluate final policies; the original-horizon column does not report rewards accumulated during learning.

RACE completes more rounds at the extended horizon in all four environments and achieves the highest mean extended-horizon count among the compared methods. In *FlatWorld Cycle*, increasing the horizon from 50 to 90 steps raises its mean count from 3.33 to 5.67 rounds; in *Cheetah Flip*, increasing the horizon from 1,000 to 1,500 steps raises the mean from 14.63 to 17.63. At the extended horizon, RACE completes 3.39 rounds on average in *PointZone*, where all three baselines record zero, and 4.17 in *Carlo*, compared with 1.73 for DRL[2] and zero for LCER and CyclER. These results provide direct evidence of continued task execution beyond the training horizon without retraining. They characterize behavior over the evaluated finite horizons, rather than certify infinite-horizon LTL satisfaction.

N Comparison with TiLoIL

We also include a direct comparison with TiLoIL (with/without demonstrations) in the Fig. 12. TiLoIL without demonstrations fails in most tasks, and even with demonstrations, it is consistently outperformed by RACE, showing the gains come from the new MRP-based exploration. Conversely, our ablation shows that removing the staged discriminator also significantly degrades performance,

indicating exploration alone is insufficient. Together, these results show that both components are necessary, with our exploration providing the key improvement over prior work.

O Comparison with DeepLTL

We evaluated DeepLTL (Jackermeier & Abate, 2024) as an additional baseline in the continuous-action *FlatWorld Cycle* and *PointZone* environments. We trained its goal-conditioned policy using the random reach–avoid curriculum. At each stage, the sampler independently selected one or two acceptable goal propositions and zero to two avoided propositions. The curriculum progressed from one-stage to two-stage and then three-stage tasks. Following the original sampling procedure, all environment propositions participated in this distribution. We trained DeepLTL for approximately 2M environment steps in *FlatWorld Cycle* and 15M steps in *PointZone*. The mean success rates over the final ten on-policy training rollouts were 89.6% and 90.0%, respectively.

After training, we froze each policy and conducted the evaluations summarized in Table 3. For the random evaluations, we generated new tasks from the same distribution used during training, with the sequence depth fixed to one, two, or three. We used 100 episodes per condition in *FlatWorld Cycle* and 20 episodes per condition in *PointZone*, with stochastic action sampling. We additionally evaluated the fixed sequences required by our target specifications, both with and without their persistent safety constraints.

In *FlatWorld Cycle*, DeepLTL achieved 95%, 88%, and 86% success on newly sampled one-, two-, and three-stage tasks, respectively. It also completed `green` $\rightarrow$ `red` $\rightarrow$ `yellow` in all 100 episodes when blue avoidance was removed. However, it failed in all 100 episodes when persistent blue avoidance was imposed. A separate single-goal diagnostic showed that red and yellow could each be reached while avoiding blue with 98% success, whereas green could not be reached while avoiding blue in any of the 100 episodes. Thus, the failure of the complete sequence was caused by obstacle-aware navigation to green rather than by an inability to compose multiple goals.

In *PointZone*, success decreased from 60% for one-stage tasks to 55% and 40% for two- and three-stage tasks. Single-goal tests showed 100% success for red and green but 0% for blue and purple. Consequently, DeepLTL failed to complete `blue` $\rightarrow$ `green` $\rightarrow$ `red` even without hazard avoidance, and all safety-constrained trials terminated with a hazard violation. In contrast, our method conditions learning and planning directly on the persistent safety requirements in the target LTL formula. Overall, these results show that high success on DeepLTL's random training distribution does not necessarily transfer to fixed recurring tasks whose safety propositions must remain false throughout execution.

In *SixteenRooms*, DeepLTL achieved success rates of 98%, 92%, 94%, and 89% on post-training random tasks with one to four stages, respectively. It reached each of the eight goal rooms in all single-goal episodes and completed both fixed four-room routes, `a` $\rightarrow$ `d` $\rightarrow$ `b` $\rightarrow$ `c` and `y` $\rightarrow$ `k` $\rightarrow$ `h` $\rightarrow$ `c`, in all episodes per route while avoiding walls. In a $1{,}000$-step evaluation, every episode completed at least one loop, with a mean of 8.02 loops and no wall violations. These results indicate that DeepLTL successfully learned both multi-stage navigation and repeated execution in *SixteenRooms*.

P Implementation Details

P.1 Methods and Algorithms

Policy optimization uses SAC (Haarnoja et al., 2018). The actor and critic sizes, and the width of each discriminator, follow the configuration in Fan & Zhu (2025); the discriminator inputs and sharing follow the component construction in this paper. Specifically:

- **Actor network**: a 4-layer multilayer perceptron (MLP) with hidden dimensions $(256, 256, 256)$.
- **Critic networks**: 3-layer MLPs with hidden dimensions $(256, 256)$.

Table 3: Evaluation of DeepLTL in our continuous-action environments.

Environment	Evaluation condition	Episodes	Success (%)	Violation (%)	Timeout (%)
FlatWorld Cycle	Random-1	100	95	0	5
	Random-2	100	88	0	12
	Random-3	100	86	1	13
	Fixed $\texttt{g} \rightarrow \texttt{r} \rightarrow \texttt{y}$	100	100	0	0
	Fixed $\texttt{g} \rightarrow \texttt{r} \rightarrow \texttt{y}$ with $\neg\texttt{blue}$	100	0	0	100
PointZone	Random-1	20	60	30	10
	Random-2	20	55	20	25
	Random-3	20	40	45	15
	Fixed $\texttt{b} \rightarrow \texttt{g} \rightarrow \texttt{r}$	20	0	0	100
	Fixed $\texttt{b} \rightarrow \texttt{g} \rightarrow \texttt{r}$ with $\neg\texttt{hazard}$	20	0	100	0
SixteenRooms	Random-1	100	98	2	0
	Random-2	100	92	8	0
	Random-3	100	94	5	1
	Random-4	100	89	11	0
	Fixed $\texttt{a} \rightarrow \texttt{d} \rightarrow \texttt{b} \rightarrow \texttt{c}$	100	100	0	0
	Fixed $\texttt{y} \rightarrow \texttt{k} \rightarrow \texttt{h} \rightarrow \texttt{c}$	100	100	0	0
	Recurring task: at least one loop within 1,000 steps	100	100	0	0

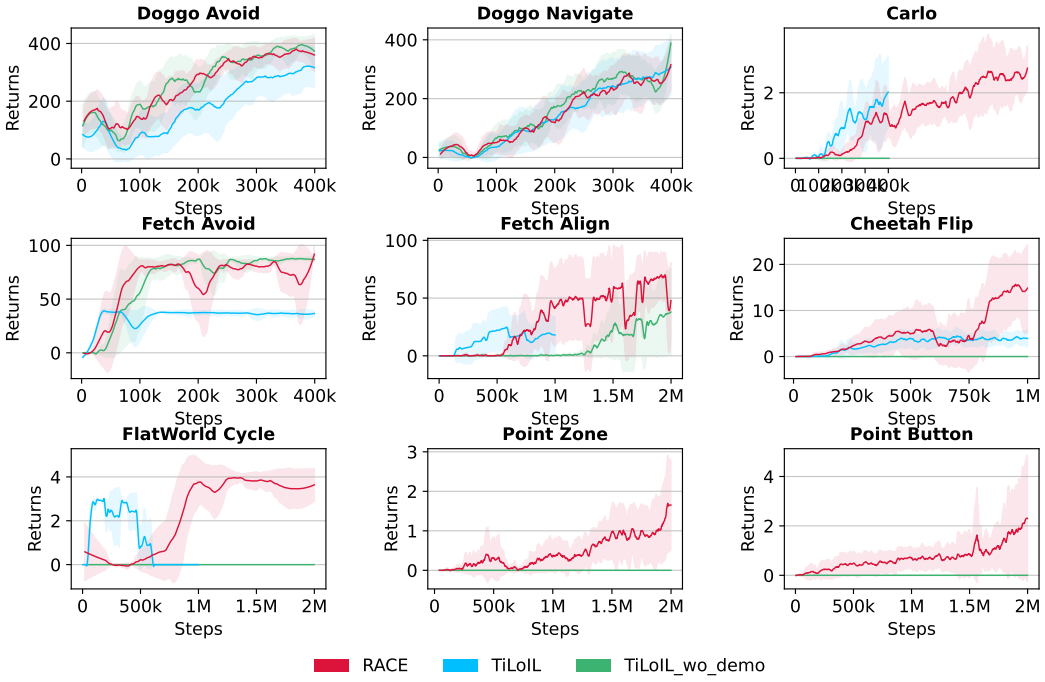

Figure 12: Comparison of RACE, TiLoIL, TiLoIL without demonstration. Our Fetch Avoid and Fetch Align have horizon L 100 while TiLoIL has 50.

- **Discriminator (reward) networks**: one probability-output head per non-accepting component, each a 2-layer MLP with hidden dimension 32; the shared alternative additionally receives the queried component encoding.

Table 4 lists the settings used here, including a default learning-start threshold of 4000 steps. *Cheetah Flip* instead uses 2×10^4 initial random-interaction steps before gradient updates. The numerical settings are reported explicitly rather than assumed to match the prior implementation in every detail.

Table 4: Hyperparameters for Q-learning and Soft Actor-Critic.

HYPERPARAMETER	VALUE
DISCOUNT FACTOR γ	0.99
ENTROPY COEFFICIENT α	0.2
REPLAY BUFFER SIZE	1×10^6
BATCH SIZE	64
LEARNING STARTS	4000
TARGET SMOOTHING COEFFICIENT τ	1×10^{-4}
Q-FUNCTION LEARNING RATE	3×10^{-4}
ACTOR LEARNING RATE	3×10^{-4}
CRITIC LEARNING RATE	3×10^{-4}
DISCRIMINATOR LEARNING RATE	3×10^{-4}
DISCRIMINATOR UPDATE FREQUENCY	EVERY STEP
TARGET NETWORK UPDATE FREQUENCY	EVERY STEP

P.2 METRICS

We evaluate performance using finite-rollout cumulative return under *eventual discounting* (Voloshin et al., 2023). Its infinite-horizon counterpart is

$$V_\pi^\gamma := \mathbb{E}_{\tau\sim\pi}\left[\sum_{i=0}^{\infty} \Gamma_i R^\times(b_i)\right],$$

where

$$R^\times(b_i) = \mathbf{1}_{\{b_i\in\mathcal{B}^*\}}, \quad \Gamma_0 = 1, \quad \Gamma_i = \prod_{t=0}^{i-1} \gamma^\times(b_t), \tag{31}$$

and

$$\gamma^\times(b_t) = \begin{cases} \gamma, & b_t \in \mathcal{B}^*, \\ 1, & \text{otherwise.} \end{cases} \tag{32}$$

Unlike the ordinary time discount in $J_L^{\infty,\gamma}$, eventual discounting advances its discount only at accepting visits. Its global optimizers approach optimal satisfaction under the conditions in Voloshin et al. (2023); a finite-rollout evaluation is a proxy metric, not a certified lower bound on satisfaction probability. All learning curves report the mean performance over 10 random seeds, with shaded regions indicating the corresponding variance.

P.3 TOOLS

The software stack consists of NumPy (Harris et al., 2020) for array operations, PyTorch (Paszke et al., 2017) for neural-network optimization, and Rabinizer (Křetínskỳ et al., 2018) for LTL-to-automaton translation. This is the same set of core tools reported by Fan & Zhu (2025).

P.4 COMPUTATIONAL COSTS

The reported hardware is an NVIDIA Quadro RTX 6000 GPU. Average run times are approximately 30 minutes for tabular experiments and 3 hours for continuous-control experiments, with comparable costs among methods within each setting. Task-specific wall-clock measurements are listed in Appendix I.