TOLLGATE: SAFE RESET-FREE REINFORCEMENT LEARNING VIA FEASIBILITY AND BUDGET GATING

Anonymous authors
Paper under double-blind review

ABSTRACT

Reinforcement learning typically relies on resets between attempts, which are readily available in simulation but costly in the real world. Prior work pairs a task-directed forward policy with a reset policy that learns to return the agent to the initial state, using estimated recoverability to determine handoffs. However, both task execution and recovery can incur safety costs, and within each attempt they draw on a single, shared safety budget. Recoverability therefore does not imply affordability: a state can remain recoverable while the remaining budget no longer covers the cost of recovery. We introduce TollGate, a budget-aware handoff framework that combines constrained learning of both policies with a dual-gate switching mechanism. An outcome-supervised feasibility gate predicts whether recovery will succeed, and a budget gate checks whether the predicted recovery cost would exhaust the remaining budget; either gate can trigger a handoff to the reset policy. On a grid world and on continuous-control tasks, TollGate trains both policies without demonstrations or a separately designed reset reward. TollGate learns the tasks, replaces the vast majority of manual resets, and drives safety costs below their thresholds. Together, these capabilities bring reinforcement learning a step closer to the real world, where neither manual resets nor unsafe exploration comes cheap.

1 INTRODUCTION

Reinforcement learning agents are typically trained in episodes: the agent attempts a task from an initial state, and the environment is reset before the next attempt. Resets keep exploration anchored to the task and make learning tractable. In simulation a reset is a line of code. On a physical system, each reset requires human intervention or purpose-built equipment to restore the initial configuration. That dependence bounds how much experience can be collected and how long training can run unattended. Reset-free RL removes it by learning two policies (Eysenbach et al., 2018; Kim et al., 2022; Sharma et al., 2022b; Patil et al., 2024). A *forward policy* attempts the task, and a *reset policy* returns the agent to the initial state on its own. Training can then continue with no one in the loop.

Yet with no one in the loop, there is also no one to stop the agent before it damages itself or its surroundings. Constrained RL represents safety-relevant events through a cost signal: a collision or a step into a hazardous region incurs cost. A policy is then trained to keep its expected cumulative cost below a cost threshold (Achiam et al., 2017; Ray et al., 2019). In a reset-free system every step can incur cost, whether the agent is attempting the task or recovering. A single attempt can tolerate only a limited amount of cost before someone has to be called in to end it with a manual reset. We call this allowance the per-attempt *safety budget* and define it formally in Section 3. Task execution and recovery draw on the same budget, which makes *when to hand off* a two-sided allocation decision. Hand off too late, and what is left may not cover the cost of recovery; the attempt then ends in the manual reset it was meant to avoid. Hand off too early, and budget and interaction time go to recovery while the task does not advance.

Existing methods time the handoff with a learned estimate of how well the policies are currently doing (Eysenbach et al., 2018; Kim et al., 2022; Patil et al., 2024). Such an estimate reports what a policy can achieve, not what achieving it will cost. Even the one method trained under an explicit safety constraint reads no cumulative quantity, since that constraint is fatal: a violation ends the

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.

attempt rather than accumulating within it (Gottwald et al., 2025). These rules therefore keep no account of cost, tracking neither what the attempt has already spent nor what recovery will add.

Consider the most direct of these estimates, the *feasibility* of recovery: can the reset policy still bring the agent back to the initial state? Feasibility alone does not settle what recovery will cost. Two recoveries can be equally likely to succeed and still differ sharply in cost: a short path that clips a hazard and a longer detour around it both reach the initial state. That recovery *can* succeed therefore does not tell us whether this attempt can still *afford* it, and no adjustment to when a feasibility signal fires can supply what is missing. The cost of recovery is information that feasibility cannot substitute for, and a handoff rule operating under a cumulative safety budget has to consult it.

We introduce TollGate, which makes the handoff budget-aware. Both policies are trained with constrained RL, using SAC-Lagrangian (Haarnoja et al., 2018; Ray et al., 2019), each under its own cost constraint. Such a constraint concerns expected cumulative cost; it does not cap what any single attempt spends. Guarding *this* attempt is left to the handoff rule. Two gates run while the forward policy acts, and either one hands control to the reset policy. The feasibility gate reads an outcome-supervised monitor of whether recovery will still succeed. The monitor is trained directly on the outcomes of observed reset trajectories, can be queried at every step, and does not feed back into either policy's learning objective. The budget gate adds the cost already spent in this attempt to a predicted cost of recovery, with an uncertainty margin, and fires once that sum reaches a safety threshold. Its design goal is to hand off while the budget can still cover the recovery. Both gates allocate control between the two policies rather than filtering the actions a policy proposes.

Beyond the handoff design, we analyse what each gate requires in order to work: what the budget gate's reserve must satisfy to cover the recovery it triggers, and what keeps the feasibility monitor supplied with supervision as the agent ventures further from the initial state.

Experiments on a grid world and continuous-control tasks adapted from Safety-Gymnasium (Ji et al., 2023) evaluate TollGate along three dimensions: task progress, the number of manual resets, and measured safety cost. TollGate learns the tasks without demonstrations or a separately designed reset reward. In the final 200k environment steps, autonomous resets account for 99.5–100% of resets on Point and 99.9–100% on Car across seeds. Manual resets concentrate early and fall to near zero by the end of training. Over the same final window, each seed's measured mean cost is below the cost threshold in both the task and reset phases.

In summary, our contributions are (i) a formulation of the budget-aware handoff problem, in which task execution and recovery draw on one per-attempt safety budget and feasibility alone does not settle affordability; (ii) TollGate, whose handoff rule couples recovery feasibility with a budget check, together with its outcome-supervised feasibility monitor; and (iii) conditions under which each gate does what it is designed to do.

2 Related Work

Reset-free RL seeks to reduce the human intervention required to prepare successive task attempts (Zhu et al., 2020; Sharma et al., 2021; 2022a;b; Kim et al., 2023). Learning a reset policy offers an alternative to manual resets, but also introduces a decision about when task execution should give way to recovery. Leave no Trace (LNT) makes this decision by thresholding the reset policy's value (Eysenbach et al., 2018). Example-based resets replace this value with a recursively learned success classifier, which also supplies the reset policy's learning signal (Kim et al., 2022; Eysenbach et al., 2021). RISC instead uses a bootstrapped success critic to reduce interaction in regions the agent has already mastered (Patil et al., 2024). These methods use learned capability to guide switching. Under a cumulative safety budget, however, knowing that recovery remains possible does not establish that the remaining budget can pay for it. Recovery cost is an additional quantity that a feasibility-based handoff must consider.

Cost-constrained policy learning provides one way to address safety during training (Achiam et al., 2017; Stooke et al., 2020; Yang et al., 2021). These policy-level constraints, however, do not determine how much a particular attempt can still spend before recovery becomes unaffordable. Recovery RL addresses a related decision by invoking a separate policy when constraint violations are likely (Thananjeyan et al., 2021). Recovery there aims to reach safety, whereas resetting here must reach the initial state to enable another task attempt. Safety also enters reset-free training through

fatal constraints in SRRL (Gottwald et al., 2025) and reach-avoid action filtering in Back to Base (Begzadić et al., 2025). Our setting allows safety costs to accumulate without immediately terminating interaction. Consequently, the handoff must account for expenditure during both task execution and recovery, rather than treating recovery itself as the end of the safety problem.

Saute RL incorporates the remaining safety budget into the state (Sootla et al., 2022), while budgeted MDPs condition values and policies on a cost allowance (Boutilier & Lu, 2016; Carrara et al., 2019). These approaches establish the relevance of budget information to control. TollGate focuses on its role in allocating control between two policies that spend the same safety budget: the handoff rule considers both expenditure so far and a predicted recovery reserve, alongside recovery feasibility. TollGate learns its feasibility signal directly from observed recovery trajectories, without bootstrapped targets. Unlike the classifier used for example-based reset-policy learning, this monitor informs handoffs but does not supply either policy's learning objective.

Together, these distinctions motivate budget-aware handoffs: reducing manual resets requires attention not only to whether the agent can recover, but also to the safety cost of doing so. TollGate brings that cost into the switching decision.

3 PRELIMINARIES

Task and policies. We consider a constrained Markov decision process (Achiam et al., 2017; Ray et al., 2019) with states s, actions a, dynamics P, a goal-reaching reward $r(s, a; g)$, and a nonnegative safety cost $c(s, a)$. Safety-relevant events incur cost without necessarily terminating interaction. The forward policy π_f pursues the task goal $g^\star$, while the reset policy π_r learns to recover to the initial state s_0. We assume task success is reported and proximity to s_0 can be measured; recovery succeeds on entry into a neighbourhood G_{stop} of s_0, subject to the budget termination rule below.

Reset-free interaction and intervals. The two policies alternate on a continuous stream of experience. A forward phase ends at task success, its horizon T_f, or an early handoff, after which π_r takes over in place. Successful recovery is an *autonomous reset*: π_f resumes from the state reached by π_r. A recovery timeout at horizon T_r or exhaustion of the safety budget instead requires a *manual reset*, a hard reset of the environment by an external intervention. An *interval* begins when π_f takes control and ends at successful recovery or a manual reset. Its cost includes both task execution and recovery. If the current interval began at step t_I, its cost spent so far is

$$A_t = \sum_{k=t_I}^{t-1} c_k. \tag{1}$$

Neither task success nor the handoff clears A_t; only the end of the interval does.

Constrained learning and the safety budget. Each policy is trained toward a phase-level constrained objective. For $p \in \{f, r\}$, let $J_r(\pi_p)$ and $J_c(\pi_p)$ denote expected reward and cumulative cost over its phase, with expectations over the phase-start states induced by interaction:

$$\max_{\pi_p} \; J_r(\pi_p) \quad \text{s.t.} \quad J_c(\pi_p) \le d. \tag{2}$$

The *cost threshold* d is shared by the two policies. These expectation constraints do not cap individual intervals. A separate *safety budget* C_{hard} terminates an interval after any step, in either phase, for which

$$A_t \ge C_{\text{hard}} \quad \Longrightarrow \quad \text{manual reset.} \tag{3}$$

This check precedes the success test. Because it follows the action, accumulated cost can exceed C_{hard} by at most the cost of that last step. Thus d concerns expected cost per phase, whereas C_{hard} limits realised expenditure through an interval-level stopping rule.

4 TOLLGATE: BUDGET-AWARE RESET-FREE LEARNING

4.1 BUDGET-AWARE HANDOFF DESIGN

Recovery feasibility alone cannot determine whether the remaining safety budget covers recovery. Consider two states from which the reset policy reaches s_0 with certainty when budget truncation is excluded. One recovery avoids hazards, while the other crosses a hazardous region and incurs safety cost. With the same remaining budget lying between their recovery costs, only the first recovery is affordable. A rule given only recovery feasibility and the remaining budget receives identical inputs in both cases. Budget-aware handoffs therefore require recovery-cost information in addition to feasibility.

TollGate combines constrained learning of both policies with two handoff gates. The feasibility gate reads an outcome-supervised monitor Q_F; the budget gate adds a predicted recovery reserve $\hat{U}_t$ to the cost already spent. Both query o_t^r, the observation with its goal channel redirected to s_0, at the reset policy's greedy action $a_t^r = \pi_r(o_t^r)$. After warmup, before each forward action, the rule is

$$\text{hand off} \iff Q_F(o_t^r, a_t^r) < \theta_F \ \vee \ A_t + \hat{U}_t > C_{\text{soft}}, \tag{4}$$

where θ_F is the feasibility threshold and C_{soft} the budget gate's *gate line*. Either trigger transfers control to π_r in place, allocating control between policies rather than filtering forward actions. The resulting recoveries supply new supervision for the monitor and experience for the reset learner.

4.2 OUTCOME-SUPERVISED FEASIBILITY GATE

The feasibility gate needs a signal tied directly to recovery outcomes. TollGate learns this signal from observed reset trajectories, separately from the reward and cost critics used to train the policies. As in recoverability-based switching (Eysenbach et al., 2018; Kim et al., 2022), each handoff determines both where the forward attempt ends and where recovery begins. Updating the monitor with these recoveries informs subsequent handoffs. This interaction can form an automatic curriculum, adjusting forward attempts as the estimated recovery capability changes.

We call the trajectory collected during one reset phase a recovery *segment*. Let G_F be a fixed neighbourhood of s_0, the monitor's *label set*. For a transition at step i, the label asks whether any next state observed from that transition onward enters G_F within the same segment:

$$y_i = \mathbf{1}\big[\exists\, k \geq i \text{ in the same segment}: \ s_{k+1} \in G_F\big]. \tag{5}$$

Labels are computed from the part of the segment observed when a batch is sampled. During an ongoing recovery, a zero label means that no such hit has yet been observed, not that recovery has failed. If that same recovery later enters G_F, earlier transitions can receive positive labels when sampled again. A hit during another segment does not change their labels. The target remains s_0, and the label does not test the budget.

In the continuous tasks, a separate network is trained by binary cross-entropy to predict these binary labels. Its targets come from observed trajectories, with no bootstrapping from network predictions. Its output is used only to decide handoffs.

An accepted recovery should provide positive supervision for the monitor. The relationship between the success set G_{stop} and the label set G_F determines when the labelling rule has this property.

Proposition 1 (Compatibility of success and labels)**.** *Suppose recovery success is declared on entry into G_{stop}, labels use the true-target suffix rule* (5)*, and the terminal transition of a completed, accepted successful segment remains in every sampled suffix. If*

$$G_{\text{stop}} \subseteq G_F,$$

then every sampled transition of that segment receives label 1 *after completion.*

The inclusion makes accepted success sufficient for positive labels. Positive labels can also come from a segment that enters G_F but fails to reach G_{stop} before timeout. How quickly the monitor obtains positive supervision therefore depends on the recovery attempts generated by handoffs.

Proposition 2 (Acquisition of positive feedback)**.** *Call a transfer to the reset policy together with the recovery that follows an* opportunity*. Consider a sequence of such opportunities. Suppose*

that, before the first positively labelled completed segment, each opportunity produces one with conditional probability at least $p \in (0, 1]$, given the entire history. If F_n is the event that the first n opportunities produce none, then

$$\Pr(F_n) \leq (1-p)^n.$$

For $\delta \in (0, 1)$, at least one such segment is obtained with probability at least $1 - \delta$ after

$$n \geq \left\lceil \frac{\log(1/\delta)}{p} \right\rceil$$

opportunities. Independence between opportunities is not required.

Together, these conditions distinguish labelling an accepted recovery from obtaining a recovery that supplies usable feedback. For the automatic curriculum to extend beyond already sampled states, handoffs and exploration must also provide feedback opportunities in new regions with positive conditional probability. Repeated supervision near s_0 alone does not establish this coverage.

4.3 Recovery-Cost Budget Gate

Even when recovery remains feasible, its cost may exceed the remaining safety budget. The budget gate supplies this missing information using the reset policy's cost critic. Its ensemble mean $\hat{V}_c(o^r, a)$ predicts expected discounted recovery cost, while the spread $\sigma(o^r, a)$ across heads serves as an uncertainty proxy (Lakshminarayanan et al., 2017). With $\kappa \geq 0$, the implemented *reserve* is

$$\hat{U}_t = \hat{V}_c(o^r_t, a^r_t) + \kappa\, \sigma(o^r_t, a^r_t), \qquad A_t + \hat{U}_t > C_{\text{soft}} \implies \text{hand off.} \tag{6}$$

The term $\kappa\sigma$ increases the predicted recovery reserve where ensemble members disagree. The gate compares spent cost plus this reserve with a line $C_{\text{soft}} \leq C_{\text{hard}}$, whereas the hard budget reacts only to cost already incurred. A gap between the two lines leaves room for the spent-plus-reserve sum to grow between checks.

To relate predictive handoffs to realised interval costs, the analysis below uses U_t for a reserve satisfying a simultaneous upper bound on undiscounted recovery costs. This coverage is an assumption of the analysis, beyond the discounted-cost prediction and uncertainty proxy used to construct $\hat{U}_t$. The proposition first bounds expenditure using this reserve and the handoff overshoot, then identifies the additional margin needed to exclude budget exhaustion.

Proposition 3 (Conditional interval-cost bound)**.** *Fix an interval, and let τ be any time at which control passes to π_r. For a possible transfer time t, let R_t denote the undiscounted cost of the recovery process started there, up to success, timeout or another ordinary termination, without truncation at C_{hard}. Suppose a reserve U_t, a number $\eta \geq 0$ and $\delta \in (0, 1)$ satisfy:*

- *(i) On an event E with $\Pr(E) \geq 1-\delta$, $R_t \leq U_t$ simultaneously throughout the forward phase, including its transfer time.*
- *(ii) The actual handoff satisfies $A_\tau + U_\tau \leq C_{\text{soft}} + \eta$.*

Then, on E,

$$A_\tau + R_\tau \leq A_\tau + U_\tau \leq C_{\text{soft}} + \eta.$$

If, in addition,

$$C_{\text{soft}} + \eta < C_{\text{hard}},$$

then nonnegative costs ensure that every cost prefix of the interval is below C_{hard}. In that case, with probability at least $1 - \delta$, this interval does not terminate through budget exhaustion; recovery may still time out.

The simultaneous coverage in (i) accounts for the adaptively chosen handoff. The overshoot allowance η in (ii) concerns growth between checks, rather than uncertainty at one query:

$$(A_{t+1} + U_{t+1}) - (A_t + U_t) = c_t + U_{t+1} - U_t.$$

It includes changes in the reserve caused by both the state and online learning. The one-step accounting requires a preceding non-crossing check and does not cover a trigger at the first post-warmup check. Detailed handoff cases are deferred to the appendix.

4.4 Constrained Policy Learning and Execution

Both policies are trained toward (2) by SAC-Lagrangian (Haarnoja et al., 2018; Ray et al., 2019). Forward and reset transitions are stored in separate replay buffers. In the continuous tasks, each has its own actor, reward critics, cost critic and multiplier $\lambda_p \geq 0$. The actor uses an entropy-regularised Lagrangian objective. Both multipliers use a phase-length-normalised cost signal. Each phase's mean per-step cost is scaled by the same fixed reference horizon. Experiments report the unscaled cumulative cost of each phase. The reset cost critic also supplies the budget gate's ensemble predictions.

The task interface allows both the observation's goal channel and the goal-reaching reward $r(s, a; g)$ to be redirected to s_0. The reset policy uses this observation o^r and the same reward evaluated toward s_0, its *reset reward*. Training uses no demonstrations or separately designed reset reward.

The grid implementation uses discrete-action learners, a per-cell running success rate in place of the neural monitor, and a two-head cost critic. Environment-specific parameters are listed in App. B.1; learner details are given in App. B. Algorithm 1 summarises execution.

Algorithm 1 TollGate training loop

Input: C_{hard}, C_{soft}, θ_F, κ, T_f, T_r.
Initialise (π_f, λ_f), (π_r, λ_r) with critics; Q_F; buffers $\mathcal{D}_f, \mathcal{D}_r$; phase $\leftarrow$ FWD; $A \leftarrow 0$
for each environment step **do**
  if phase $=$ FWD and past warmup **then**
    Evaluate (4); if either gate fires, phase $\leftarrow$ RST
  end if
  Act with π_{phase}; observe (r, c, s'); $A \leftarrow A + c$; store in $\mathcal{D}_{\text{phase}}$
  Update the acting policy and its critics; update Q_F if phase $=$ RST
  if $A \geq C_{\text{hard}}$: manual reset; phase $\leftarrow$ FWD; $A \leftarrow 0$
  else if phase $=$ FWD **and** (task success **or** T_f reached): phase $\leftarrow$ RST
  else if phase $=$ RST **and** $s' \in G_{\text{stop}}$: phase $\leftarrow$ FWD; $A \leftarrow 0$
  else if phase $=$ RST **and** T_r reached: manual reset; phase $\leftarrow$ FWD; $A \leftarrow 0$
end for

5 Experiments

We evaluate how TollGate combines task learning and autonomous recovery under cost constraints. The experiments examine how recovery cost informs handoffs (Section 5.1), how task learning and autonomous recovery develop under cost constraints (Section 5.2), and what each handoff gate contributes (Section 5.3).

The main learning experiments use navigation tasks in a grid world and in fixed-layout Point and Car environments (Fig. 1). The continuous environments are adapted from Safety-Gymnasium, a benchmark suite for safe reinforcement learning built on Safety Gym (Ray et al., 2019; Ji et al., 2023). The forward policy learns to reach the task goal; the reset policy learns to reach a neighbourhood of the initial state from the handoff position, enabling another attempt. Recovery starts after task completion, a phase timeout, or an early handoff. Both policies learn without demonstrations, using the same goal-reaching reward with the target redirected for recovery (Section 4.4). Hazard encounters incur safety cost in either phase, so the budget must cover both task execution and recovery.

Learning results are measured during training over five runs per configuration. Learning curves and the Car table report interquartile means (IQMs) with 95% bootstrap confidence intervals; the grid table reports means. Environment steps count interactions from both the forward and reset phases.

5.1 Recovery Cost and Budget-Aware Handoffs

A constructed grid illustrates why budget-aware handoffs need the cost of recovery as well as its feasibility (Section 4.1). A diagonal hazard band separates regions with different recovery costs (Fig. 2a). We use TollGate's grid-world implementation for this diagnostic. To distinguish the roles of the two gates, we fix the Lagrange multipliers at zero and disable hard-budget resets (App. C.1).

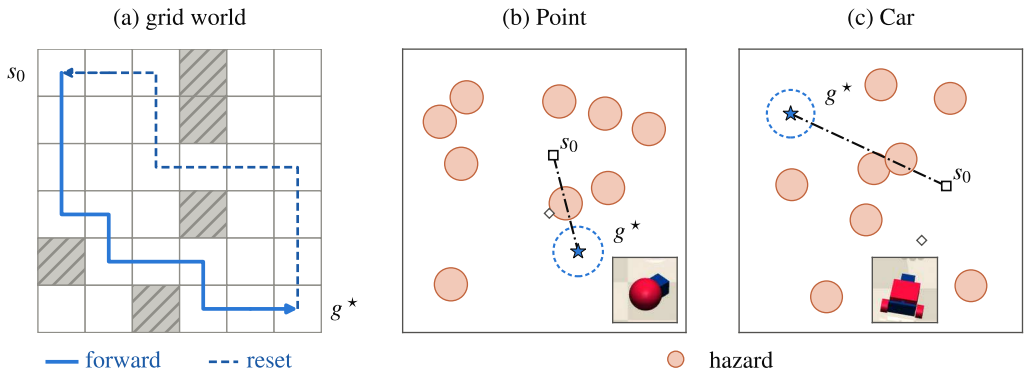


Figure 1: Navigation tasks for learning task execution and recovery. Hazards incur unit cost per entry on the grid and per step in the continuous tasks. Grid routes illustrate hazard-free paths for both phases; robot insets show appearance.

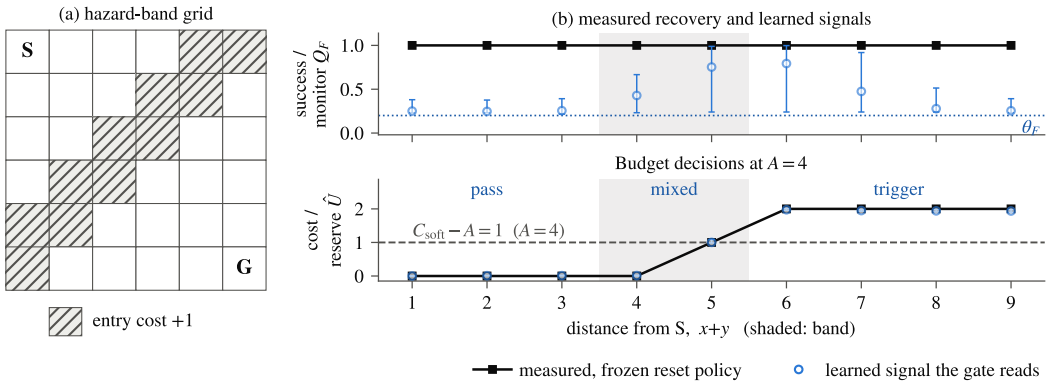


Figure 2: Frozen-policy recovery measurements are compared with learned gate signals (20 stochastic rollouts per state). Points average over five seeds and states at the same distance $x + y$ from S; whiskers show the min–max across these states and seeds.

After training, the frozen stochastic reset policy succeeds from every state in the sampled rollouts, without additional exploration. Every state passes the feasibility gate, while measured recovery cost rises across the band. The budget gate's reserve $\hat{U}$, the predicted recovery cost plus an uncertainty margin, reflects this difference (Fig. 2b). At the same spent cost $A = 4$, one unit below C_{soft}, the budget gate passes near-side states and triggers at far-side states, with mixed decisions at the boundary. Thus, states with the same feasibility verdict receive different handoff decisions because recovery consumes different amounts of the remaining safety budget.

5.2 Reset-Free Learning under Cost Constraints

With both policies trained under cost constraints, TollGate learns the navigation tasks in all three environments (Figs. 3a and 4a).

As training proceeds, learned recovery increasingly supports repeated task attempts. On the grid, TollGate uses an average of 37 manual resets across the run, compared with 300 for the episodic learner, which resets after every interval (Fig. 3b). On Point, autonomous resets account for 72–78% of resets over training and 99.5–100% over the final 200k environment steps. On Car, the corresponding ranges are 94–97% and 99.9–100% (Fig. 4b). By late training, almost all resets are autonomous. TollGate's handoffs thus support continued task practice with less manual intervention.

The grid comparison records mean total environment cost over training: TollGate accumulates 950 cost units, including recovery, against the episodic reference's 556. Figure 3c shows how assigning

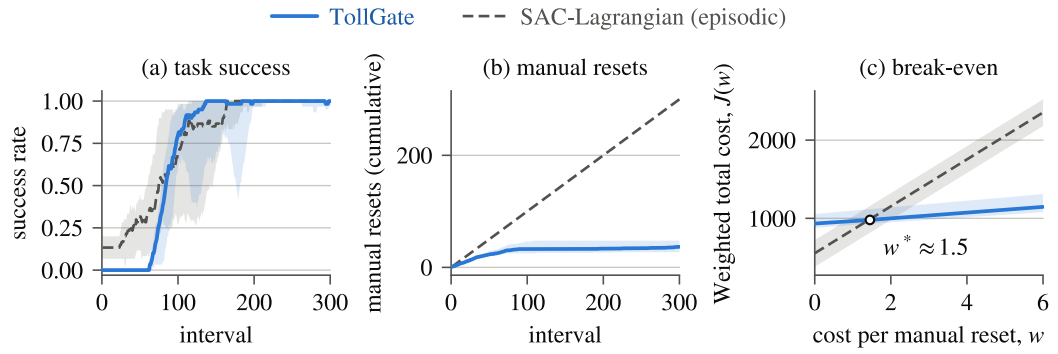


Figure 3: TollGate compared with episodic learning on the grid. Success uses a 20-interval moving average. $J(w)$ = total environment cost $+w\cdot$ manual resets, assigning cost w per reset. TollGate intervals include recovery, so interaction is unequal.

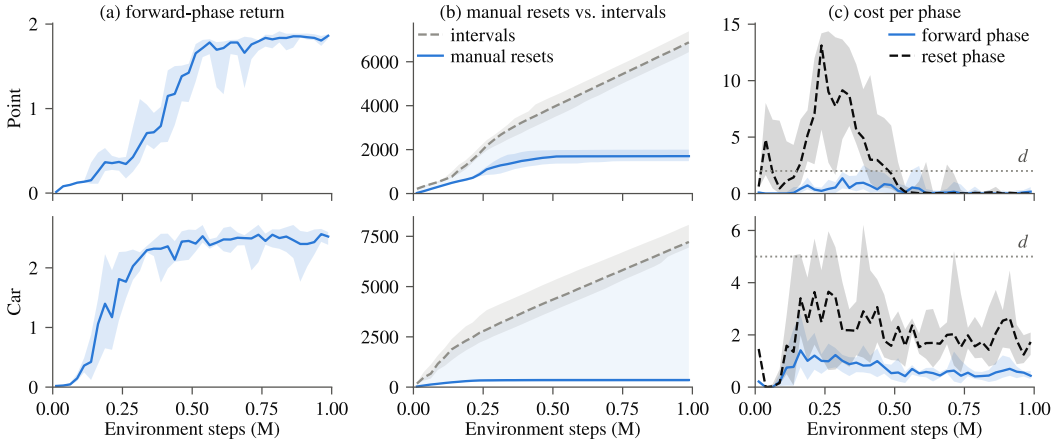


Figure 4: Task learning, manual intervention and phase costs during continuous training. Results are aggregated in windows of 25,000 environment steps.

a cost to manual resets changes this comparison. On the continuous tasks, realised phase cost settles below the cost threshold as training proceeds (Fig. 4c). Over the final 200k environment steps, every seed has mean cost below d in each phase. Together with the task results, these observations indicate that TollGate's constrained learners can learn effectively under gate-controlled interaction while reducing recovery cost.

5.3 Contributions of the Handoff Gates

Starting from the fixed reset-value threshold used by Leave no Trace (LNT) (Eysenbach et al., 2018), we examine the effects of using a feasibility monitor and then adding a budget gate. On the grid and Car, the LNT-style rule and our gate configurations share constrained learners, phase horizons and the hard budget (Tables 1 and 2).

The feasibility gate improves task learning relative to LNT-style switching at the tested thresholds. Both configurations omit the budget gate. Feasibility gate only solves every grid layout, compared with $1/5$ or $0/5$ for the LNT-style rule. On Car, the LNT-style rule combines a final-window forward-phase return of 0.02 with the fewest manual resets and zero phase cost. Feasibility gate only reaches 2.66 in the same window. These results suggest that outcome-supervised feasibility gating supports an effective automatic curriculum for task learning.

Adding the budget gate to feasibility gate only reduces dependence on manual resets. On Car, whole-run manual resets fall from 496 to 352. Over the final 200k environment steps, TollGate

Table 1: Handoff-rule comparisons on the grid. Success: final 50 intervals; solved: number of layouts with success ≥ 0.9. Cost and manual reset counts are accumulated over the full training run. Hard-cap rate: fraction of intervals reaching the cap, active in every configuration.

gating rule	success	solved	cost	hard-cap rate	manual resets
TollGate	**0.99**	**5/5**	**950**	**4.1%**	**37**
feasibility gate only	0.98	5/5	928	5.5%	43
budget gate only	0.83	3/5	1077	8.1%	49
LNT-style, $q_{\min} = -2$	0.20	1/5	937	10.0%	50
LNT-style, $q_{\min} = -1$	0.00	0/5	790	7.5%	39

Table 2: Handoff-rule comparisons on Car. Return and realised phase costs cover the final 200k environment steps; gate handoffs cover the whole run. Brackets give 95% confidence intervals.

	TollGate	feasibility gate only	budget gate only	LNT-style rule
manual resets (whole run)	352 [256, 408]	496 [446, 683]	480 [394, 915]	299 [261, 397]
manual resets (final 200k)	0 [0, 0.7]	15.3 [12.3, 32]	0.7 [0, 66]	0.3 [0, 8.3]
forward-phase return	2.50 [2.20, 2.57]	2.66 [2.58, 2.70]	2.63 [2.60, 2.76]	0.02 [0.01, 0.04]
forward-phase cost	0.52 [0.41, 0.80]	0.85 [0.57, 1.13]	0.75 [0.20, 1.02]	0.00 [0.00, 0.00]
reset-phase cost	1.82 [1.41, 2.78]	0.59 [0.45, 0.78]	1.10 [0.62, 1.28]	0.00 [0.00, 0.00]
total cost (whole run, $\times 10^3$)	19.9 [17.3, 20.6]	15.4 [13.4, 22.0]	19.4 [17.6, 27.9]	5.26 [4.49, 8.01]
gate handoffs ($\times 10^3$)	3.38 [3.12, 4.33]	3.21 [2.76, 5.78]	0.77 [0.70, 1.06]	118 [86.4, 148]

needs almost no manual resets, while feasibility gate only still needs 15.3; TollGate's mean reset-phase cost is higher in this window. On the grid, TollGate uses 13% fewer manual resets at 2.4% higher total environment cost, with both configurations solving every layout. Under cost constraints, feasibility-based switching supports task learning but still relies on manual resets. TollGate reduces this remaining dependence by including predicted recovery cost in the handoff decision.

Removing the feasibility gate while retaining the budget gate increases manual resets in both environments. Feasibility feedback therefore continues to reduce manual intervention even when recovery cost already guides handoffs.

6 Conclusion

We presented TollGate, a framework for cost-constrained reset-free learning. Both the forward and reset policies learn under cost constraints. An outcome-supervised feasibility gate and a budget gate determine handoffs. The feasibility gate assesses whether the reset policy can still reach the initial state. The budget gate accounts for cost already incurred and predicted recovery cost. Our theoretical analysis provides sufficient conditions for avoiding budget exhaustion within an interval and for obtaining useful recovery feedback.

On the tested grid and continuous-control tasks, TollGate learned the tasks with increasingly autonomous recovery. By late training, almost all resets in the continuous environments were autonomous, and the mean cost in each phase was below its cost threshold.

TollGate evaluates recovery from the current state without explicitly comparing alternative future handoff points. A short continuation could lead to a less costly recovery, but the intervening cost and risk must also be considered. Look-ahead handoff selection could explore this trade-off. Explicit experience sharing between the forward and reset policies is another direction for improving their coordination. For an autonomous learner, reaching the task goal is only part of the journey. It must also be able to recover and try again. TollGate asks the agent to set aside the fare home before moving on.

AI USE STATEMENT

We used generative AI tools to polish the manuscript, search for related work, and assist with writing data-analysis and plotting scripts. We reviewed all AI-assisted work and take full responsibility for the final content.

REPRODUCIBILITY STATEMENT

Section 4 and Algorithm 1 describe the handoff rules and training procedure. Appendix A provides proofs and states the assumptions and scope of the theoretical results. Appendix B reports environment and learning settings and details the continuous-control and grid-world implementations. Section 5 and the figure and table captions specify measurement windows and reporting conventions. Appendix C provides diagnostic protocols, gate-line sensitivity results, supplementary learning curves, and across-run variability, including per-seed late-training phase costs in Table 9.

REFERENCES

Joshua Achiam, David Held, Aviv Tamar, and Pieter Abbeel. Constrained policy optimization. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pp. 22–31. PMLR, 2017.

Azra Begzadić, Nikhil Uday Shinde, Sander Tonkens, Dylan Hirsch, Kaleb Ugalde, Michael C. Yip, Jorge Cortés, and Sylvia Herbert. Back to Base: Towards hands-off learning via safe resets with reach-avoid safety filters. In Necmiye Ozay, Laura Balzano, Dimitra Panagou, and Alessandro Abate (eds.), *Proceedings of the 7th Annual Learning for Dynamics and Control Conference*, volume 283 of *Proceedings of Machine Learning Research*, pp. 1154–1166. PMLR, 2025.

Craig Boutilier and Tyler Lu. Budget allocation using weakly coupled, constrained Markov decision processes. In *Proceedings of the 32nd Conference on Uncertainty in Artificial Intelligence*. AUAI Press, 2016.

Nicolas Carrara, Edouard Leurent, Romain Laroche, Tanguy Urvoy, Odalric-Ambrym Maillard, and Olivier Pietquin. Budgeted reinforcement learning in continuous state space. In *Advances in Neural Information Processing Systems 32*, pp. 9295–9305, 2019.

Benjamin Eysenbach, Shixiang Gu, Julian Ibarz, and Sergey Levine. Leave no Trace: Learning to reset for safe and autonomous reinforcement learning. In *International Conference on Learning Representations*, 2018.

Benjamin Eysenbach, Sergey Levine, and Ruslan Salakhutdinov. Replacing rewards with examples: Example-based policy search via recursive classification. In *Advances in Neural Information Processing Systems 34*, pp. 11541–11552, 2021.

Tristan Gottwald, Maximilian Schier, and Bodo Rosenhahn. Safe resetless reinforcement learning: Enhancing training autonomy with risk-averse agents. In Alessio Del Bue, Cristian Canton, Jordi Pont-Tuset, and Tatiana Tommasi (eds.), *Computer Vision – ECCV 2024 Workshops, Proceedings, Part XII*, volume 15634 of *Lecture Notes in Computer Science*, pp. 100–116. Springer, 2025. doi: 10.1007/978-3-031-92591-7_7.

Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 1856–1865. PMLR, 2018.

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. In *Advances in Neural Information Processing Systems 36: Datasets and Benchmarks Track*, 2023.

Jigang Kim, Jae Hyeon Park, Daesol Cho, and H. Jin Kim. Automating reinforcement learning with example-based resets. *IEEE Robotics and Automation Letters*, 7(3):6606–6613, 2022. doi: 10.1109/LRA.2022.3173039.

Jigang Kim, Daesol Cho, and H. Jin Kim. Demonstration-free autonomous reinforcement learning via implicit and bidirectional curriculum. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 16441–16457. PMLR, 2023.

Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. In *Advances in Neural Information Processing Systems 30*, pp. 6402–6413, 2017.

Darshan Patil, Janarthanan Rajendran, Glen Berseth, and Sarath Chandar. Intelligent switching for reset-free RL. In *International Conference on Learning Representations*, 2024.

Alex Ray, Joshua Achiam, and Dario Amodei. Benchmarking safe exploration in deep reinforcement learning. Technical report, OpenAI, 2019.

Archit Sharma, Abhishek Gupta, Sergey Levine, Karol Hausman, and Chelsea Finn. Autonomous reinforcement learning via subgoal curricula. In *Advances in Neural Information Processing Systems 34*, pp. 18474–18486, 2021.

Archit Sharma, Rehaan Ahmad, and Chelsea Finn. A state-distribution matching approach to non-episodic reinforcement learning. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 19645–19657. PMLR, 2022a.

Archit Sharma, Kelvin Xu, Nikhil Sardana, Abhishek Gupta, Karol Hausman, Sergey Levine, and Chelsea Finn. Autonomous reinforcement learning: Formalism and benchmarking. In *International Conference on Learning Representations*, 2022b.

Aivar Sootla, Alexander I Cowen-Rivers, Taher Jafferjee, Ziyan Wang, David H Mguni, Jun Wang, and Haitham Ammar. Saute RL: Almost surely safe reinforcement learning using state augmentation. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 20423–20443. PMLR, 2022.

Adam Stooke, Joshua Achiam, and Pieter Abbeel. Responsive safety in reinforcement learning by PID Lagrangian methods. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pp. 9133–9143. PMLR, 2020.

Brijen Thananjeyan, Ashwin Balakrishna, Suraj Nair, Michael Luo, Krishnan Srinivasan, Minho Hwang, Joseph E. Gonzalez, Julian Ibarz, Chelsea Finn, and Ken Goldberg. Recovery RL: Safe reinforcement learning with learned recovery zones. *IEEE Robotics and Automation Letters*, 6 (3):4915–4922, 2021. doi: 10.1109/LRA.2021.3070252.

Qisong Yang, Thiago D. Simão, Simon H. Tindemans, and Matthijs T. J. Spaan. WCSAC: Worst-case soft actor critic for safety-constrained reinforcement learning. In *Proceedings of the 35th AAAI Conference on Artificial Intelligence*, volume 35, pp. 10639–10646, 2021. doi: 10.1609/aaai.v35i12.17272.

Henry Zhu, Justin Yu, Abhishek Gupta, Dhruv Shah, Kristian Hartikainen, Avi Singh, Vikash Kumar, and Sergey Levine. The ingredients of real world robotic reinforcement learning. In *International Conference on Learning Representations*, 2020.

A Proofs and Supporting Analysis

A.1 Compatibility of Success and Labels

Proof of Proposition 1. Let ℓ be the index of the last transition in a completed, accepted successful recovery segment. Its next state satisfies $s_{\ell+1} \in G_{\text{stop}}$. By the assumed inclusion, $s_{\ell+1} \in G_F$. For any sampled transition i in this segment, $i \le \ell$, and the terminal transition remains in its observed suffix by hypothesis. Thus $k = \ell$ witnesses the event in (5), giving $y_i = 1$. The argument applies to every sampled transition of the completed segment. □

The terminal transition is essential to this argument. Before it is observed, an ongoing segment can still supply zero labels. Nor does a positive label imply an accepted recovery: a segment can visit G_F and then time out without reaching G_{stop}. The set inclusion is a sufficient condition for the stated implication, not a necessary condition for every possible recovery trajectory.

Consider a state $x \in G_{\text{stop}} \setminus G_F$ that is not a task-success state. Suppose the reset policy has a deterministic, zero-cost self-loop at x. After warmup, let each forward phase begin at x, with the monitor's queried value remaining below θ_F. The feasibility gate immediately hands control to the reset policy, whose next state is again x. The next-state success test in Algorithm 1 accepts the recovery, but its only suffix label is zero. The next interval repeats the same interaction, without a forward action. All recoveries succeed and no cost is incurred, yet neither positive labels nor task progress are obtained. This construction requires the stated dynamics and persistent gate decision; failure of set inclusion alone does not imply such a loop. Even with compatible sets, feedback from new states still requires opportunities to visit those states, as considered next.

A.2 Acquisition and Coverage of Recovery Feedback

Proof of Proposition 2. Let B_i be the event that opportunity i produces a completed segment containing a positive suffix label. Let $\mathcal{H}_i$ denote the full interaction and learning history through completion of opportunity i, with $\mathcal{H}_0$ the history at the start of the observation period. Write

$$F_0 = \Omega, \qquad F_i = \bigcap_{k=1}^{i} B_k^c.$$

Here F_{i-1} is measurable with respect to $\mathcal{H}_{i-1}$. The conditional probability assumption gives $\Pr(B_i^c \mid \mathcal{H}_{i-1}) \le 1 - p$ on F_{i-1}. Consequently,

$$\begin{aligned}\Pr(F_i) &= \mathbb{E}\big[\mathbf{1}_{F_{i-1}} \Pr(B_i^c \mid \mathcal{H}_{i-1})\big] \\ &\le (1-p)\Pr(F_{i-1}).\end{aligned}$$

Starting from $\Pr(F_0) = 1$ and iterating yields $\Pr(F_n) \le (1-p)^n$. Since $1 - p \le e^{-p}$, any integer

$$n \ge \left\lceil \frac{\log(1/\delta)}{p} \right\rceil$$

satisfies $\Pr(F_n) \le e^{-np} \le \delta$. The proof conditions on the full prior history at each opportunity and uses no independence between opportunities. □

The event B_i includes a segment that enters G_F and later ends without an accepted recovery. It does not require every transition in that segment to have a positive label. A lower bound on the marginal frequency of positive segments would not replace the history-conditional assumption used in the recursion.

Fix a finite collection of handoff-state regions $\mathcal{S}_1, \dots, \mathcal{S}_m$. Let $D_{i,j}$ be the event that opportunity i starts in $\mathcal{S}_j$, and let $F_{n,j}$ mean that the first n opportunities supply no completed positively labelled segment starting in that region. Suppose, on histories in $F_{i-1,j}$,

$$\Pr(D_{i,j} \mid \mathcal{H}_{i-1}) \ge \rho_j > 0, \qquad \Pr(B_i \mid D_{i,j}, \mathcal{H}_{i-1}) \ge p_j > 0.$$

Thus an opportunity supplies the required regional feedback with conditional probability at least $q_j = \rho_j p_j$. Applying the preceding recursion to each region and taking a union bound gives

$$\Pr\left(\bigcup_{j=1}^{m} F_{n,j}\right) \le \sum_{j=1}^{m}(1-q_j)^n \le me^{-nq_{\min}}, \qquad q_{\min} = \min_{1\le j\le m} q_j.$$

Hence $n \ge \lceil \log(m/\delta)/q_{\min} \rceil$ opportunities suffice to obtain such feedback from every region with probability at least $1-\delta$. This statement assumes that these opportunities occur and that the conditional coverage bounds hold throughout the relevant histories. Neither set compatibility nor a monitor threshold establishes the bounds on ρ_j or p_j.

To convert an opportunity count into an interaction bound, suppose a fixed $B < \infty$ bounds the environment steps from the start of the observation period to completion of the first opportunity, and between successive completed opportunities. The first n opportunities then finish within nB environment steps. This bound must include forward waiting as well as recovery; a reset-phase time limit alone does not bound the time until the next handoff. The resulting bound concerns acquisition of labelled trajectories, not their subsequent fit by the monitor or progress of the task policy.

A.3 Conditional Interval-Cost Bound

Proof of Proposition 3. Work on the simultaneous coverage event E in condition (i). Since coverage holds at every possible transfer time, it holds at the actual, adaptively selected time τ, giving $R_\tau \le U_\tau$. Condition (ii) then implies

$$A_\tau + R_\tau \le A_\tau + U_\tau \le C_{\text{soft}} + \eta.$$

This establishes the stated cost bound on an event of probability at least $1-\delta$.

Now assume additionally that $C_{\text{soft}} + \eta < C_{\text{hard}}$. Every forward cost prefix is at most A_τ, and every cost prefix after handoff is at most $A_\tau + R_\tau$, because individual costs are nonnegative. Thus no prefix reaches C_{hard}. Couple the process with a hard-budget check to the recovery used to define R_τ, using the same dynamics and action randomness until a possible budget termination. Such a termination would require a prefix of the latter process to reach C_{hard}, which the preceding inequality excludes. The interval therefore does not terminate through budget exhaustion on E. Its probability is at least $1-\delta$; the argument does not exclude a recovery timeout or another ordinary termination. □

Let $Y_t = A_t + U_t$. Suppose t_0 is a preceding check with $Y_{t_0} \le C_{\text{soft}}$, the handoff occurs after at most L environment steps, and throughout this intervening period

$$Y_{t+1} - Y_t = c_t + U_{t+1} - U_t \le \Delta$$

for some $\Delta \ge 0$. Summing the increments gives

$$Y_\tau = Y_{t_0} + \sum_{t=t_0}^{\tau-1}(c_t + U_{t+1} - U_t) \le C_{\text{soft}} + L\Delta.$$

Thus $\eta = L\Delta$ is sufficient for condition (ii) under these additional assumptions. The increment includes changes in the required recovery reserve caused by the next state and, for an online estimate, by learner updates. A bound on c_t alone is not a bound on this entire increment. The term $\kappa\sigma$ enlarges the implemented reserve at a query; η instead bounds overshoot of the spent-plus-reserve sum at handoff. A change in that uncertainty term can itself contribute to a change in the implemented sum.

The accounting concerns the actual transfer, whether caused by a gate, task success or the forward time limit.

1. If $Y_\tau \le C_{\text{soft}}$, condition (ii) holds with $\eta = 0$ for that transfer. This can occur at a feasibility-triggered handoff or an ordinary phase end, but neither cause alone implies that the sum is below the line.
2. For a crossing at consecutive active checks, a preceding non-crossing check and the increment bound give $L = 1$ and $\eta = \Delta$. In Algorithm 1, a detected trigger changes the acting

policy before another environment action, so there is no extra forward step after detection. The one step in this accounting is the step between checks that caused the crossing, not a delay after the trigger.

3. A trigger at the first post-warmup check need not have a preceding non-crossing check. Its handoff bound must instead be established directly, or from a bound over the entire unchecked period. The one-step argument does not supply it automatically.

These cases use $Y_t = A_t + U_t$, the reserve in the proposition. The implemented gate checks $A_t + \hat{U}_t$ instead. A non-crossing check of that implemented sum does not establish $Y_t \leq C_{\text{soft}}$ without an additional relation between the two reserves. Likewise, setting a gap between the gate line and the hard budget does not establish an increment bound. Given a valid η, the further conclusion excluding budget exhaustion requires $\eta < C_{\text{hard}} - C_{\text{soft}}$; equality is insufficient because the hard-budget test fires when the accumulated cost reaches C_{hard}.

For a recovery trajectory of length $\ell \leq H$, with nonnegative costs and a fixed finite bound H, define

$$R = \sum_{k=0}^{\ell-1} c_k, \qquad D_\gamma = \sum_{k=0}^{\ell-1} \gamma^k c_k, \qquad 0 < \gamma \leq 1.$$

Since $\gamma^{H-1} \leq \gamma^k \leq 1$ for every term,

$$\gamma^{H-1} R \leq D_\gamma \leq R, \qquad R \leq \gamma^{-(H-1)} D_\gamma.$$

If a critic gives the exact expectation of D_γ under this same recovery process, taking expectations yields

$$\mathbb{E}[R] \leq \gamma^{-(H-1)} \mathbb{E}[D_\gamma].$$

This is an expectation bound, not an upper bound on an individual realised cost. Approximation error can place a learned prediction on either side of its target. The implemented reserve does not apply this discount conversion, and ensemble spread by itself does not supply the simultaneous coverage event in condition (i).

The comparison also requires the same stopping rule. The quantity R_t in the proposition is not truncated at the hard budget. In the continuous learner, budget termination and reset success truncate cost bootstrap targets, whereas a reset timeout alone does not mark the cost target terminal. Thus the critic's target and the recovery process in the proposition differ in their termination semantics as well as in discounting.

Simultaneous coverage allows the handoff to depend on the trajectory and learned signals. Separate coverage claims at fixed times do not automatically give the same probability at an adaptively chosen transfer. The proposition concerns one interval; extending its probability statement to an entire training run would require coverage across the intervals as well. The empirical reset counts do not establish either coverage assumption.

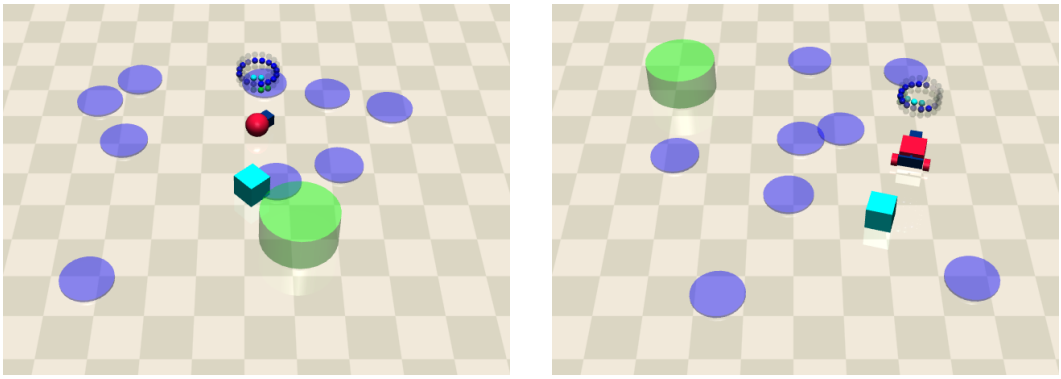
B Implementation and Experimental Protocol

B.1 Environments and Configuration

Tables 3 and 4 specify TollGate's main learning configurations. The grid uses run-specific layouts; Point and Car are adapted from the PointGoal1 and CarGoal1 environments in Safety-Gymnasium (Ji et al., 2023), respectively, using fixed layouts and reset-free interaction. Figure 5 shows their initial configurations. The constructed-grid diagnostic has separate settings in Section C.1.

In the continuous tasks, both phase horizons grow with the cumulative number of successful autonomous resets, using their respective maxima. Grid phase horizons are fixed. Gate warmup and the onset of learning are distinct settings.

For the forward cost critic, we oversample nonzero-cost transitions to counter the dilution of sparse cost signals under uniform replay.



(a) Point (b) Car

Figure 5: Simulator renders of the fixed layouts used in the continuous learning experiments, before any policy action. Red bodies are the agents, green cylinders mark task goals, and purple disks mark hazards.

Table 3: Environment and gate settings for TollGate in the main learning experiments.

Parameter	Grid	Point	Car
Layout	6×6	Fixed	Fixed
Grid hazard density	0.15	Not used	
Action space	Discrete (4)	Continuous	Continuous
Cost threshold d	3	2	5
Gate line C_{soft}	7	16	16
Safety budget C_{hard}	10	16	24
Feasibility threshold θ_F	0.20	0.30	0.20
Reserve coefficient κ		1	
Gate warmup (environment steps)	None	2,000	
Maximum forward horizon T_f	50	300	300
Maximum reset horizon T_r	50	250	400

Table 4: Learning settings for the main TollGate runs. Merged entries denote shared settings. Grid critic updates start at the stated buffer occupancy; continuous updates require the step threshold to be exceeded and a full minibatch. Both policies learn their entropy temperatures.

Learning parameter	Grid	Point	Car
	Discrete		
Policy learner	SAC-Lagrangian	SAC-Lagrangian	
Hidden-layer widths	$(128, 128)$	$(256, 256)$	
Network optimiser		Adam	
Replay capacity per policy	20,000	1,000,000	
Minibatch size	128	256	
Actor learning rate	3×10^{-4}	5×10^{-6}	
Reward/cost critic learning rate	3×10^{-4}	10^{-3}	
Monitor update	Tabular EMA	Binary cross-entropy	
Monitor EMA step / learning rate	0.2	10^{-3}	
Initial tabular monitor value	0.05	Not used	
Reward and cost discount factors		0.99	
Target-network mixing coefficient		0.005	
Actor/target update period (critic updates)		2	
Reward critics per policy		2	
Cost critics (forward / reset)	2 / 2	1 / 5	
Initial multiplier	0.1	0	
Multiplier update	Projected ascent	Adam + projection	
Multiplier learning rate	10^{-2}	10^{-4}	
Multiplier reference horizon	Not used	300	
Initial entropy temperature	0.2	1	
Entropy-temperature learning rate	10^{-4}	10^{-3}	
Critic-update threshold	300/buffer	1,000 environment steps	
Forward cost-stratified fraction	Not used	0.25	

B.2 Continuous-Control Learners

Each policy has its own Gaussian SAC actor, reward critics, cost critic, replay buffer, entropy temperature and Lagrange multiplier. The networks use ReLU hidden layers with the widths in Table 4. The reward target uses the smaller of the target reward-critic predictions. The forward cost critic has a single nonnegative output; the reset cost critic has nonnegative ensemble outputs, implemented with softplus. Each reset-cost head fits the same Bellman target, whose continuation value is the target ensemble mean. The ensemble's sample standard deviation supplies σ. Eligible environment steps update the active policy's critics; actor, entropy-temperature and target-network updates follow the delayed cadence in the table, counted separately for each policy. Each entropy target is minus the action dimension. The target-network coefficient is the weight on the new parameters, not the weight retained from the old target.

The monitor is a separate ReLU network taking the reset-view observation and action as inputs and producing a scalar logit. It is trained with Adam and binary cross-entropy, with a sigmoid applied for gate queries. After each eligible reset-policy update, a monitor minibatch is drawn from the reset buffer. A label is positive when any stored next-position in the sampled transition's currently observed same-phase suffix lies in G_F. Otherwise it is negative, including an unfinished suffix with no observed hit. Labels are recomputed when sampled, do not cross phase boundaries, and do not directly test the remaining safety budget. There is no bootstrap target for the monitor, and its loss does not update either policy or their reward and cost critics.

The reset observation redirects the goal channel to s_0; its reward uses the decrease in distance to s_0 plus an arrival bonus, following the task's goal-reaching reward form. Each nonempty phase updates its own multiplier with Adam, followed by projection onto the nonnegative half-line. The update uses $H_{\mathrm{ref}}(\sum_{t \in p} c_t)/L_p$, where L_p is the realised length of phase p and H_{ref} is the fixed reference horizon in Table 4. It does not substitute the dynamic phase limit for this reference. Reported EpCost is the unscaled sum $\sum_{t \in p} c_t$.

Reward and cost targets have separate terminal masks. Both stop bootstrapping at environment termination or detected recovery success; cost targets also stop at safety-budget exhaustion. The last forward transition is marked at a handoff or ordinary forward phase end, and its cost target is truncated at that boundary. A reset timeout alone does not make its bootstrap terminal. The training loop processes safety budget exhaustion before ordinary recovery success. After gate warmup, the gates query the reset-view observation at the reset policy's greedy action before a forward action is executed. A triggered handoff changes the active policy without an additional forward environment step.

B.3 Grid-World Learners

The grid uses separate discrete SAC-Lagrangian agents, each with a softmax actor, twin reward critics, twin linear-output cost critics and a replay buffer. Reward targets use the smaller reward prediction; cost targets and the actor's cost penalty use the larger cost prediction, averaged over the policy's action distribution. The networks use ReLU hidden layers. In addition to sampling from the actor, training uses uniform random actions with probability initially 1, decayed by a factor of 0.995 to a floor of 0.05 through terminal-transition and phase-end bookkeeping. The entropy target is $0.6 \log |\mathcal{A}|$, and the learned temperature is clipped to $[0.01, 0.5]$.

A per-cell table replaces the neural monitor. After a recovery attempt, only the cell where control passed to the reset policy is updated toward the binary recovery outcome using the EMA step in Table 4. No update is propagated along the rest of the recovery trajectory. Recovery success means reaching the initial cell. The gate compares the current cell's table entry with θ_F; unlike the continuous monitor, this table has no action input. Its initial value and threshold are given in the configuration tables.

The grid reserve averages over the reset policy's action distribution rather than querying a single greedy action. For its two cost heads,

$$\hat{V}_c(s) = \sum_a \pi_r(a \mid s) \frac{Q_{c,1}(s,a) + Q_{c,2}(s,a)}{2},$$
$$\sigma(s) = \sum_a \pi_r(a \mid s) \frac{|Q_{c,1}(s,a) - Q_{c,2}(s,a)|}{2}.$$

The budget gate then uses $A_t + \hat{V}_c(s) + \kappa\sigma(s)$. Each multiplier is updated by projected ascent on the difference between its recent recorded cost mean and d, without the continuous learner's phase-length normalisation. Updates use the latest at most 50 recorded costs, start once 10 records are available, and project onto $[0, 10]$. Terminal transitions record phase cost even when zero; other phase-end flushes append only positive cost totals. Forward replay uses the environment's terminal flag and reset replay uses arrival at the initial cell; neither adds the continuous learner's separate budget or phase-boundary cost mask.

B.4 Measurement and Aggregation

Learning results are measured during training. The grid experiments use five distinct layouts with one run per layout and 300 intervals per run; each continuous configuration uses five seeds on a fixed layout for one million environment steps. Environment steps include forward and reset phases; forward actions occupy 39–48% of steps in the TollGate continuous runs.

C Supplementary Experiments and Results

C.1 Constructed-Grid Diagnostic

The diagnostic uses the discrete learners in Section B.3, but fixes both Lagrange multipliers at zero and the entropy temperature at 0.05. Each handoff rule is trained for 300 intervals with five policy seeds on the same constructed layout; both phase horizons are 50 steps. A separate no-gate calibration run sets $C_{\text{soft}} = 5$ and the reference level $C_{\text{hard}} = 8$. Hard-budget resets are disabled in this diagnostic, so reaching the reference level does not itself terminate an interval.

The constructed grid uses zero-based row/column coordinates, with $S = (0, 0)$ and $G = (5, 5)$. Both endpoints are excluded from the layer profiles, which group states by $x + y$. The learned signals and frozen-policy measurements use the same checkpoints. For each cell and seed, the probe samples 20 recoveries from the frozen stochastic reset policy, without additional exploration or budget truncation, for at most 50 steps. It records recovery success and undiscounted recovery cost. Pass/Mixed/Trigger summarize individual budget-gate queries at fixed spent cost $A = 4$; they are not training handoff frequencies. The whiskers span the minimum and maximum over state–seed values rather than bootstrap confidence intervals. The tabular monitor is read against θ_F, not as a calibrated success probability.

Table 5: Training outcomes on the constructed grid, reported as min–max ranges over five seeds. Task success covers the final 50 intervals; cost and reference-level crossings cover all 300 intervals, including both phases. These are training measurements, distinct from the frozen-policy queries in Figure 2.

Handoff rule	Task success	Mean interval cost	Fraction of intervals with cost ≥ 8
TollGate	0.82–0.94	6.37–6.83	0.200–0.237
Feasibility gate only	1.00–1.00	6.93–7.44	0.247–0.297

Table 5 reports task performance alongside cost: the budget gate reduces expenditure in this diagnostic but also ends forward attempts earlier, lowering task success during training.

C.2 Grid Comparisons and Gate-Line Sensitivity

The main grid comparisons use $C_{\text{soft}} = 7$, selected for TollGate from a gate-line sweep. The budget-gate-only configuration uses the same gate line. Its task-success comparison with TollGate is therefore specific to that setting. Table 6 reports the complete TollGate sweep with the hard budget held fixed. The $C_{\text{soft}} = 9$ results reuse the earlier TollGate runs; the other gate lines use the same layout seeds and learning configuration.

Table 6: Grid gate-line sweep with $C_{\text{hard}} = 10$. Entries give means and standard deviations across five layouts (standard deviations use denominator n). Task success covers the final 50 intervals; solved counts layouts with success at least 0.9. Manual resets and total environment cost cover all 300 intervals, including both phases. The bold gate line is the main comparison setting.

Gate line C_{soft}	Task success	Solved	Manual resets	Total cost
5	0.908 ± 0.068	3/5	38.2 ± 13.9	968 ± 107
6	0.968 ± 0.045	4/5	37.0 ± 12.3	927 ± 87
7	0.988 ± 0.016	5/5	37.4 ± 9.1	950 ± 81
8	0.992 ± 0.010	5/5	40.4 ± 7.6	906 ± 54
9	0.972 ± 0.035	5/5	41.6 ± 10.4	912 ± 90

Table 7: Across-layout standard deviations for the means in Table 1, using denominator $n = 5$. Success covers the final 50 intervals; the other metrics cover the full run. Entries describe dispersion, not confidence intervals.

Handoff rule	Success	Total cost	Hard-cap rate (percentage points)	Manual resets
TollGate	0.016	81	0.93	9.1
Feasibility gate only	0.025	95	1.61	7.5
Budget gate only	0.257	176	2.03	10.9
LNT-style, $q_{\min} = -2$	0.398	170	2.76	12.4
LNT-style, $q_{\min} = -1$	0.000	175	2.82	12.7

Adding the budget gate to feasibility gating reduces manual resets on four layouts and leaves them unchanged on one; total cost is higher on three layouts and lower on two. With the budget gate

retained, adding the feasibility gate reduces manual resets on four layouts and increases them on one. These paired layout counts describe the selected setting rather than a uniform effect across layouts or gate lines.

C.3 Continuous-Task Results

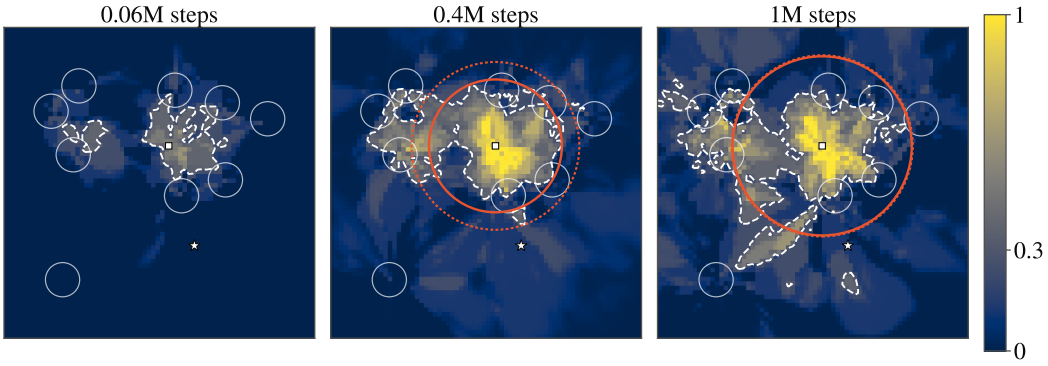


Figure 6: Feasibility-monitor maps from one Point run (seed 0). Colours show monitor outputs on a 70×70 position grid, averaged over eight headings using the reset policy's greedy action; the white dashed contour is $\theta_F = 0.3$. The square and star mark the initial position and task goal; thin circles mark hazards. Orange solid and dotted rings show the median and 90th percentile of starting distances for successful recoveries recorded in the preceding 20k environment steps. There are no such successes at 60k, so no orange rings appear in the first panel.

Figure 6 illustrates the evolution of the monitor during training, rather than a matched comparison of handoff rules. The colours are not calibrated recovery probabilities, and the distance rings summarise observed successful recoveries rather than certifying a recoverable region.

Table 8: Continuous control ($n = 5$ seeds per environment; cells are the min–max range over the five seeds). Manual resets and the overall autonomous-reset share cover the whole run; the indented row gives the final-200k autonomous-reset share. Forward-phase return and the two costs are converged quantities over environment steps $\geq$ 800k. Costs are the realised cost of a phase. The threshold d and the other constants are in App. B.1.

	Point	Car
solved	5/5	5/5
forward-phase return	1.77–1.88	2.11–2.57
manual resets	1430–2085	235–417
autonomous resets	72–78%	94–97%
final 200k steps	99.5–100%	99.9–100%
cost / forward phase	0.00–0.30	0.40–0.88
cost / reset phase	0.00–0.16	1.37–2.99

C.4 Car Handoff-Rule Comparisons

The budget-gate-only comparison is a post-hoc reuse of existing runs. The configuration audit checked the shared learners, phase horizons and cost limits, as well as the implementation differences between batches. The differing feasibility-label radius and monitor threshold are unused by the budget-only switching rule. Matched seed identifiers do not imply identical trajectories, so paired direction counts provide only descriptive evidence.

Feasibility gating queries the outcome monitor at reset actions. The LNT-style comparison is limited to the tested threshold and query protocol.

Table 9: Per-seed late-training mean realised phase costs for TollGate. Each entry averages recorded phase costs at environment steps $\geq$ 800k in the corresponding run, without phase-length normalisation.

	Point ($d = 2$)		Car ($d = 5$)	
Seed	Forward cost	Reset cost	Forward cost	Reset cost
0	0.011	0.013	0.884	2.992
1	0.011	0.026	0.401	1.368
2	0.302	0.163	0.514	2.344
3	0.000	0.000	0.431	1.482
4	0.128	0.147	0.625	1.648

In the main comparison, adding the budget gate to feasibility gating reduces whole-run manual resets in all five seeds, while final-window forward return is lower and reset-phase cost is higher in all five. Removing the feasibility gate while retaining the budget gate increases whole-run manual resets in all five seeds. Near-zero late-window IQMs do not imply that every seed requires almost no intervention: the budget-only run with seed 3 records 98 manual resets in the final 200k steps.

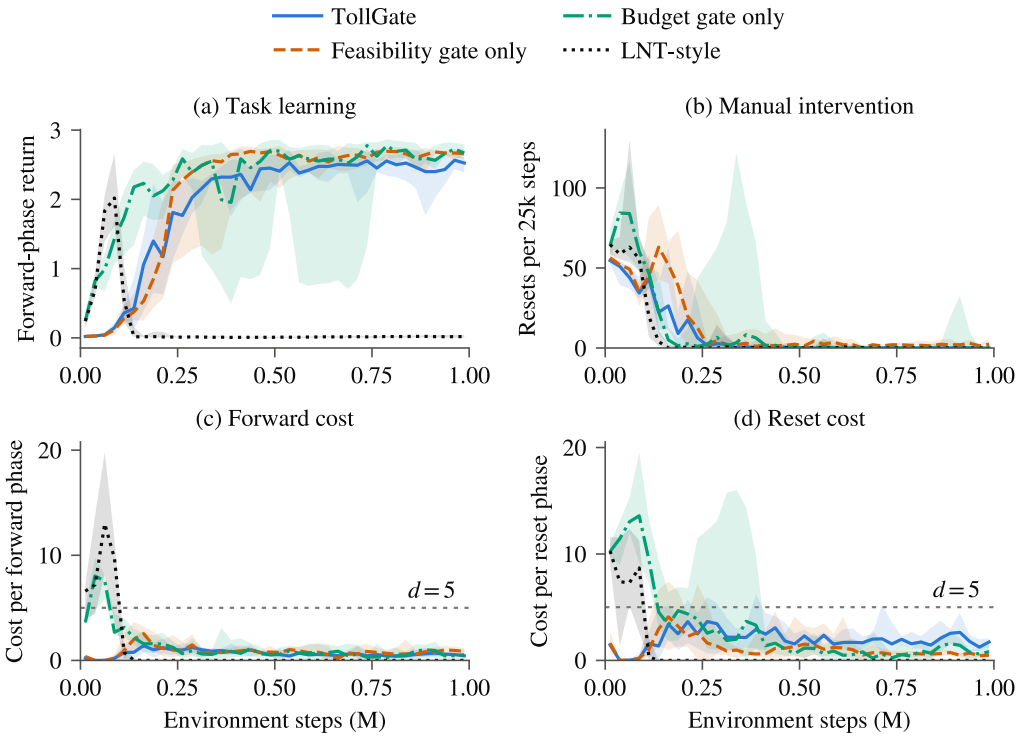


Figure 7: Car handoff-rule learning curves for the runs in Table 2. Lines are IQMs across five seeds; shading gives pointwise 95% percentile bootstrap intervals using whole-run resampling. Windows are 25k environment steps, open on the left and closed on the right. Returns and realised costs are averaged within each run by phase-end time; manual resets are increments of the cumulative counter, not cumulative totals. Both phases contribute to the horizontal axis. No additional smoothing is applied.