TRAM: Test-Time Risk Adaptation with Mixture of Agents

Mohamad Chehade
UT Austin
chehade@utexas.edu

Amrit Singh Bedi
University of Central Florida
amritbedi@ucf.edu

Souradip Chakraborty
MIT
soura24@mit.edu

Amy Zhang
UT Austin
amy.zhang@austin.utexas.edu

Hao Zhu
UT Austin
haozhu@utexas.edu

Abstract

Deployed reinforcement learning agents often face safety requirements that are specified only after training: new hazard maps, revised risk thresholds, or behavioral alignment constraints. We study zero-update deployment-time adaptation in which a fixed library of risk-neutral source policies must be reused under a newly specified reward–risk tradeoff. We propose **TRAM** (**T**est-time **R**isk **A**daptation with **M**ixture of **A**gents), a source-scored composition rule that evaluates each source under the target reward and an occupancy-based deployment risk, then selects actions using risk-adjusted source scores. Unlike training-time risk-sensitive methods tied to a fixed surrogate such as return variance, TRAM supports spatial barrier exposure, divergence to a reference behavior, and local volatility risks specified at test time. We make the surrogate nature of the method explicit: TRAM is not claimed to solve the full occupancy-control problem of the stitched policy, but it admits a measurable source-hull mismatch term connecting source-scored risk to realized risk. Experiments in gridworlds, MuJoCo Reacher, Safety-Gymnasium, and an inference-time LLM alignment setting show that TRAM improves deployment risk while preserving reward and requires no parameter updates at test time.

1 Introduction

Reinforcement learning systems are often deployed after training into settings where the safety specification changes. A robot may receive a new goal or temporary keep-out region, a safe-navigation agent may face a revised hazard map, or an assistant may be judged against a new behavioral reference. These changes are usually operational rather than architectural: the already-trained behaviors are still useful, but the desired reward–risk tradeoff is not known until deployment. Throughout, a *source policy* is a fixed, already-trained policy, and the *source library* is the set of such policies, trained once before deployment (*source training*). At deployment our method trains nothing: at each step it scores every source by its estimated value for the new task minus the requested risk penalty and acts with the best one. Retraining for each new specification is expensive, slow, and sometimes not allowed once a model is deployed. The problem is increasingly pressing as pretrained policy libraries and behavior foundation models are reused across tasks [34, 33, 42], and as alignment of large language models shifts from retraining toward inference-time control [28, 6, 39]. The core idea of this paper is to move the choice of risk from source training to deployment-time decision making: train reusable behaviors first, then compose them according to the risk requested at test time.

Why fixed training-time risk is brittle. Risk-sensitive RL typically chooses the risk model during training, such as variance, CVaR, or a fixed constraint. This is effective when the future risk is

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

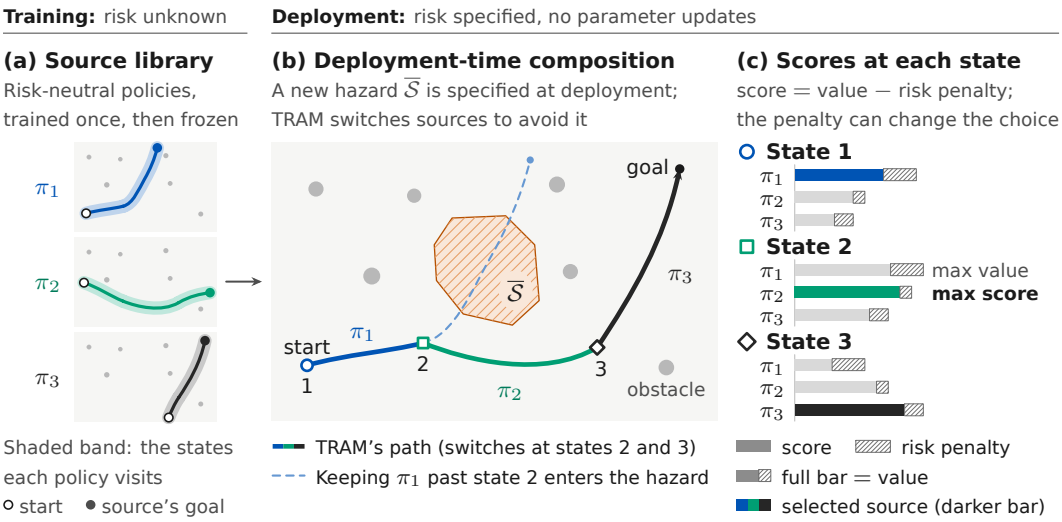


Figure 1: TRAM decouples source training from deployment-time risk specification. (a) Before deployment, a library of risk-neutral source policies is trained once and frozen; the shaded band around each route marks the states that policy visits. (b) At deployment, a new hazard $\overline{S}$ is specified. At each state, TRAM scores every source and acts with the best one, so the executed path is stitched from the source routes in (a); keeping π_1 past state 2 (dashed) would enter the hazard. (c) A source's score is its value $\max_a Q_T^{\pi_j}(s,a)$ (full bar) minus a fixed per-source risk penalty (hatched), Eq. 4.3. At state 2, π_1 has the highest value but π_2 the highest score, so the penalty changes the choice. TRAM selects the maximum by Eq. 4.4 and re-scores at the next state; all quantities are schematic, and no parameters are updated.

known, but brittle when the deployment risk changes from return variance to a spatial barrier, a behavioral reference, or a local volatility signal. A risk model chosen too early can over-specialize the source policies, remove useful behaviors, and leave no mechanism to recover when the deployment specification changes. Figure 1 summarizes the alternative: keep a reusable source library, then inject the requested risk only at deployment.

Proposed approach. We propose **TRAM** (**T**est-time **R**isk **A**daptation with **M**ixture of **A**gents), a zero-update method for adapting a fixed source library to newly specified deployment risks. TRAM trains source policies risk-neutrally to preserve behavioral coverage. At deployment, TRAM scores each candidate source action by combining target-task value with the requested occupancy-risk penalty, then selects the best source-action pair at the current state. This converts risk adaptation into fast source scoring rather than retraining, constrained RL, or source-policy fine-tuning. As shown in Figure 1, the active source can change along the trajectory, producing a locally stitched behavior that responds to the current state and risk specification. TRAM is therefore a tractable *source-scored surrogate*: it does not claim to exactly solve the full occupancy-control problem of the stitched policy. The main text focuses on the implementable scoring rule and its localized transfer guarantees (Theorems 4.1 and 4.3), while Appendix C analyzes the measurable mismatch between cached source risks and the realized stitched occupancy.

Contributions. **Problem:** we formalize zero-update deployment-time adaptation to newly specified occupancy risks. **Method:** we introduce TRAM, a tractable source-scored composition rule supporting barrier, divergence, and local-volatility risks at test time. **Theory:** we provide localized transfer bounds, a conditional source-hull comparison, and extensions for stitched occupancies and approximate evaluation. **Evidence:** we demonstrate TRAM across gridworlds, MuJoCo Reacher, Safety-Gymnasium, and LLM alignment without deployment-time parameter updates, including a comparison with a constrained policy retrained on the task.

2 Related Work

Zero-shot transfer and successor features. Zero-shot RL learns representations or policies that generalize without fine-tuning [27, 31, 15, 36, 44]. Successor features (SFs) exploit shared structure to evaluate policies under new rewards by dot products [1–3], and can be combined with learned

Table 1: Comparison with representative families. Columns: risk sensitivity, support for risk types beyond variance, cross-task transfer, use of shared task structure, and little/no test-time compute.

METHOD	RISK-AWARE	GENERAL RISK	TRANSFER	TASK STRUCT.	TEST-TIME
Risk-sensitive RL	✓	✗	✗	✗	✗
Zero-shot RL	✗	✗	✓	✗	✓
Dual RL	✓	✓	✗	✗	✗
Risk-aware Adaptation	✓	✗	✓	✗	✗
TRAM (ours)	✓	✓	✓	✓	✓

Refs. Risk-sensitive RL [4, 9, 18, 25, 26, 29, 37, 41, 49]; zero-shot RL [27, 31, 15, 36, 44]; dual RL [48]; risk-aware adaptation/SFs [12, 45, 40, 14, 11, 24, 17, 23, 1–3].

task-specific features [5]. Behavior foundation models scale this idea to pretrained policy families that are prompted with a reward or a few demonstrations at test time [34, 33, 42]. These methods support reward transfer, but they generally assume the evaluation objective is fixed once the target reward is known. TRAM keeps the zero-update benefit while adding a deployment-time risk term to the source score.

Risk-sensitive RL and safe adaptation. Risk-sensitive RL optimizes objectives involving variance, CVaR, constrained returns, or dual variables [4, 9, 18, 25, 26, 29, 37, 41, 49, 48]. Safety-transfer methods learn teachers, safety critics, reusable safe policies, or risk-sensitive options [45, 40, 14, 11, 24, 17, 23]. Constraint-conditioned and dual actor–critic approaches train over constraint families or adapt dual variables after the constraint is known [47, 21], and offline safe RL methods adjust to new cost thresholds at deployment through cost-conditioned sequence models or feasibility-guided diffusion policies [22, 51]. Distributionally robust RL hedges against dynamics or reward shifts within an ambiguity set fixed before training [16, 30, 52, 32], whereas TRAM adapts to a risk specified at deployment under fixed dynamics. These lines are complementary, but they usually require the relevant risk family during training or nontrivial optimization during adaptation.

Inference-time alignment and mixtures of agents. For language models, alignment is increasingly performed at inference time: controlled and reward-guided decoding steer a frozen model with a value or reward estimate [28, 20, 6], and mixture-of-agents methods combine several frozen models [46, 7]. Weight interpolation and constrained fine-tuning address multiple or safety-critical objectives at training time [35, 8], while optimizing a learned reward too strongly degrades true quality [10]. TRAM brings this inference-time, mixture-of-agents view to sequential decision-making, with an explicit risk term specified at deployment.

Risk-aware SFs and TRAM's positioning. Risk-aware successor features (RaSF) [12] combine SF transfer with a mean–variance risk proxy. TRAM differs in two ways: source policies are trained risk-neutrally to preserve reusable behavioral coverage, and the deployment risk is an occupancy-based functional supplied at test time, such as barrier exposure, KL-to-reference, or local volatility. Thus TRAM sits between zero-shot transfer and risk-aware decision-making: fixed sources are reused like SF/GPI, but action selection is risk-adjusted at inference by TRAM.

3 Problem Formulation

3.1 Deployment Setting

Deployment-time safety as constrained optimization. We model agent-environment interaction as an MDP $M = (\mathcal{S}, \mathcal{A}, p, R, \gamma)$ with state space $\mathcal{S}$, action space $\mathcal{A}$, transition kernel $p(\cdot \mid s, a)$, reward $R(s, a, s')$, and discount $\gamma \in [0, 1)$. A policy π induces action-value function $Q^\pi(s, a) = \mathbb{E}^\pi[G_t \mid S_t = s, A_t = a]$, where $G_t = \sum_{i=0}^{\infty} \gamma^i R_{t+i+1}$.

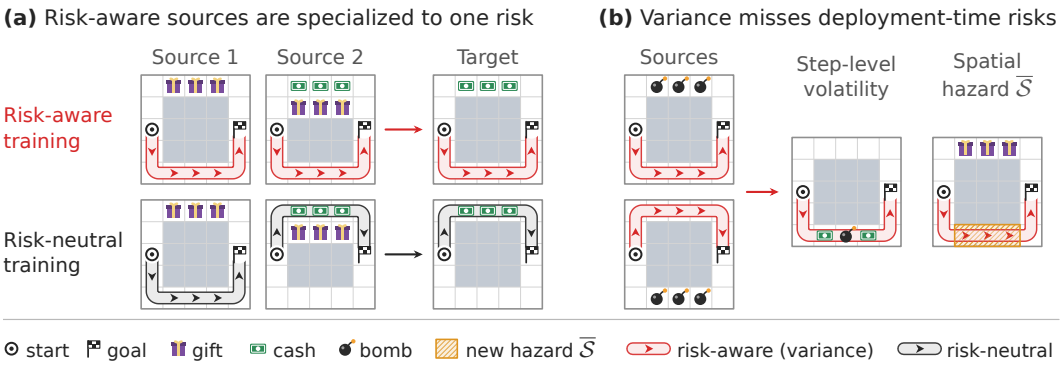


Figure 2: Motivating limitations of variance-based risk-aware transfer. (a) A risk-aware source is optimal for its training-time (variance) risk but specialized to it; when the deployment risk differs, it can miss the target-optimal route, while risk-neutral training preserves diverse reusable behaviors. **(b)** Trajectory-return variance can miss deployment-time risks: stochastic per-step rewards, spatial danger regions, and deployment-specific hazards are not reliably captured by return variance alone.

At deployment, the agent faces a target task $M_{\mathrm{T}} = (\mathcal{S}, \mathcal{A}, p_{\mathrm{T}}, R_{\mathrm{T}}, \gamma)$ with performance measured by expected return $J(\pi) = \mathbb{E}_{\pi}[\sum_{t\geq 0} \gamma^t R_{\mathrm{T}}(S_t, A_t, S_{t+1})]$. Deployment also introduces a safety or alignment requirement specified only at test time through a risk functional $\rho(\pi)$:

$$\max_{\pi} J(\pi) \quad \text{subject to} \quad \rho(\pi) \leq \delta, \tag{3.1}$$

where $\delta > 0$ is the risk tolerance. Since solving a new constrained RL problem at deployment can be too slow, we work with the penalized surrogate

$$\max_{\pi} \tilde{J}_c(\pi) = J(\pi) - c\,\rho(\pi), \quad c \geq 0. \tag{3.2}$$

Transfer setting: no retraining allowed. We assume access to fixed source policies $\{\pi_j\}_{j=1}^{n}$ trained on related source tasks $M_j = (\mathcal{S}, \mathcal{A}, p_j, R_j, \gamma)$. The finite-MDP analysis evaluates all policies under the target dynamics and assumes each source policy is optimal for its own source task under those dynamics. Changes to goals or hazard specifications can leave dynamics unchanged. When source values or occupancies are estimated using an approximate model, Appendix C.2 accounts for the evaluation error. The token-level LLM experiment illustrates the scoring principle and is outside the occupancy-MDP guarantees.

3.2 Why Variance-Based Risk Fails at Deployment

Many risk-aware RL and transfer methods optimize a risk proxy selected before deployment, commonly return variance, as in RaSF [12]. These methods are valuable when the risk specification is known in advance, but they expose three limitations when the risk is revealed or changed only at deployment.

Limitation 1: Conservative training destroys behavioral diversity. Variance-penalized or constraint-specialized sources can converge to uniformly cautious behaviors (Figure 2(a)), shrinking the repertoire available for test-time composition. Training sources risk-neutrally on varied tasks can retain useful behaviors, although diversity and source-hull inclusion are not automatic consequences of removing the training penalty.

Limitation 2: Variance degenerates in deterministic settings. With deterministic dynamics and policies, trajectory return variance can be zero (Figure 2(b), center), giving no safety signal even when step-level hazards exist. Variance-based selection can therefore reduce to risk-agnostic behavior precisely when predictable environments accumulate clear step-level risks.

Limitation 3: Variance misrepresents domain-specific risks. Return variance captures uncertainty in cumulative reward, not spatial exposure, behavioral drift, or worst-case contact with a new hazard (Figure 2(b), right). A low-variance path can cross a dangerous region, while a safer path may have higher benign stochasticity.

These limitations suggest three requirements for deployment-time adaptation: preserve a broad source library before the risk is known, accept risk functionals beyond return variance, and select

actions without retraining or re-solving a new constrained MDP. They also motivate reporting reward, requested risk, and source coverage separately rather than collapsing all outcomes into one safe-RL score. TRAM is designed around exactly these requirements.

4 TRAM: Test-Time Risk Adaptation via Source-Scored Composition

The limitations in Section 3.2 motivate a deployment-time method that separates source training from the risk model revealed at deployment. TRAM aims to preserve source diversity through risk-neutral training and introduces risk only through source scoring at test time.

Occupancy-based deployment risks. Deployment risk is often not just a property of the final return. In navigation, risk may be time spent near a barrier; in robotics, contact with a forbidden region; in power-system control, repeated operation near voltage or line-flow limits. These risks depend on where the policy spends discounted probability mass, not only on the variance of its cumulative reward. Using a fixed initial distribution μ_0 and target dynamics, TRAM therefore represents deployment risks as functionals of normalized discounted state-action occupancies d^π:

$$d^{\pi_j}(s,a) = (1-\gamma)\sum_{t\geq 0}\gamma^t \Pr_{\pi_j}(S_t = s, A_t = a \mid S_0 \sim \mu_0). \tag{4.1}$$

In implementation, global source occupancies are cached for the target dynamics, while values are evaluated by successor features, rollouts, or learned critics. This supports risks that return variance cannot encode: **spatial barrier exposure**, e.g.,

$$\rho_{\text{barrier}}(d) = -\log(\tau - d(\overline{\mathcal{S}})), \quad d(\overline{\mathcal{S}}) = \sum_{s,a} d(s,a)\mathbf{1}\{s \in \overline{\mathcal{S}}\}, \tag{4.2}$$

on its domain $d(\overline{\mathcal{S}}) < \tau$; **local volatility**, which measures per-step fluctuations under the occupancy; and **behavioral drift**, e.g., $\rho_{\text{KL}}(d) = \text{KL}(d\|\bar{d})$ to remain close to a safe reference behavior. These functionals are specified after source training and enter only through the deployment-time score. The Lipschitz guarantees below require the compared occupancies to remain in a domain with bounded risk variation: in particular, a log barrier must stay a positive distance from its singularity. Appendix A.3 states the relevant conditions.

Source-scored policy construction. Given fixed source policies $\{\pi_j\}_{j=1}^n$, TRAM computes for each source a target-task value estimate $Q_T^{\pi_j}(s,a)$ and a cached source-level risk $\rho(d^{\pi_j})$. The source-adjusted score and deployed action are the two core equations of TRAM:

$$S_j(s,a) = Q_T^{\pi_j}(s,a) - c\rho(d^{\pi_j}), \qquad c \geq 0, \tag{4.3}$$

$$\pi_T(s) \in \arg\max_{a\in\mathcal{A}} \max_{j\in[n]} S_j(s,a). \tag{4.4}$$

Algorithm 1 implements Eq. 4.4: cache source occupancies and risks, evaluate source values at the current state, and select the best risk-adjusted source action. The rule is deliberately a *tractable source-scored surrogate*; it reuses fixed source policies and cached source risks rather than repeatedly solving a new constrained RL problem. Appendix C gives a diagnostic for when the realized stitched occupancy differs from the source-hull proxy. An optional source screen does not certify the stitched policy's risk, because switching among eligible sources can change its occupancy.

This makes TRAM a modular deployment layer. Source training builds a library intended to provide broad behavioral coverage; deployment supplies the target reward, risk functional, tradeoff weight, and optional threshold. In new hazard maps, updated operating limits, or revised behavioral references, Algorithm 1 changes only the score used to select among fixed sources. The method is therefore most appropriate when source policies are already available, test-time updates are undesirable, and the requested risk can be evaluated or approximated per source.

Appendix D.5 discusses how to choose c and its relation to the tolerance δ.

Online computation. For linear successor features $Q_T^{\pi_j}(s,a) = \psi^{\pi_j}(s,a)^\top w_T$, online scoring costs $\mathcal{O}(n|\mathcal{A}|d)$ per state for n sources, $|\mathcal{A}|$ actions, and feature dimension d; if value tables are cached, it costs $\mathcal{O}(n|\mathcal{A}|)$. The risk term is a constant-time lookup after source occupancy caching. Thus the

Algorithm 1 TRAM: zero-update source-scored risk adaptation

Input: fixed sources $\{\pi_j\}_{j=1}^n$; target value specification; deployment risk ρ; weight $c \geq 0$; optional threshold δ

Before deployment: once per risk specification

Estimate source occupancies under the target dynamics; cache finite risks $\widehat{\rho}_j$

$\mathcal{J} \leftarrow [n]$, or $\mathcal{J} \leftarrow \{j : \widehat{\rho}_j \leq \delta\}$ if screening is used ▷ optional source screen

if $\mathcal{J} = \varnothing$ **then** stop ▷ the screen admits no deployment policy

At deployment: no source parameters are updated

for each deployment state s **do**

  for $j \in \mathcal{J}$ and $a \in \mathcal{A}$ **do**

    Evaluate $\widehat{Q}_T^{\pi_j}(s,a)$ by successor features, a critic, or rollouts

    $S_j(s,a) \leftarrow \widehat{Q}_T^{\pi_j}(s,a) - c\,\widehat{\rho}_j$ ▷ score, Eq. 4.3

  Execute $a^\star \in \arg\max_{a\in\mathcal{A}} \max_{j\in\mathcal{J}} S_j(s,a)$ ▷ select, Eq. 4.4

deployment-time work in TRAM is source scoring, not retraining, target-task policy optimization, or repeated global occupancy recomputation for every candidate stitched action. This complexity excludes source evaluation and occupancy estimation before scoring; a changed transition model can require new occupancy estimates even though source parameters remain fixed.

4.1 When and Why TRAM Works: Theoretical Foundations

We next give a compact theoretical account of the source-scored surrogate optimized by TRAM. The results isolate reward transfer quality, the deployment-risk penalty, and source-library coverage. The realized stitched-policy correction is handled separately in Appendix C, and the approximate-dynamics extension is given in Appendix C.2.

Traditional transfer learning bounds can require global reward similarity across all state-action pairs, often yielding pessimistic guarantees. TRAM enables a localized analysis through trajectory-specific mismatch terms. The analysis uses finite discounted MDPs, bounded rewards, common dynamics, stationary policies, and a nonempty retained source library. Write π^* for a statewise risk-neutral target optimum. We assume each source policy π_j is statewise optimal for its source task under the common dynamics, i.e., $Q_j^{\pi_j} = Q_j^*$. For expected one-step rewards r_T, r_j, define the localized reward discrepancy:

$$\Delta_\pi^{(T,j)}(s,a) = \mathbb{E}_\pi\left[\sum_{t\geq 0}\gamma^t |r_T(S_t,A_t) - r_j(S_t,A_t)| \,\middle|\, S_0 = s, A_0 = a\right]. \tag{4.5}$$

The transfer term contains discrepancies along both policies:

$$C_j(s,a) = \Delta_{\pi^*}^{(T,j)}(s,a) + \Delta_{\pi_j}^{(T,j)}(s,a).$$

Theorem 4.1 (Localized risk-adjusted guarantee)**.** *Assume exact source values and risks, and let ρ be finite and L-Lipschitz on a domain containing all source occupancies, d^{π^*}, and d^{π_T}. Then*

$$\widetilde{Q}_T^{\pi^*}(s,a) - \widetilde{Q}_T^{\pi_T}(s,a) \leq \min_j C_j(s,a) + 4Lc, \tag{4.6}$$

where $\widetilde{Q}_T^\pi = Q_T^\pi - c\rho(d^\pi)$ uses the actual global policy occupancy.

Corollary 4.2 (Reward-only guarantee)**.** *Under the same assumptions,*

$$Q_T^*(s,a) - Q_T^{\pi_T}(s,a) \leq \min_j C_j(s,a) + 2Lc. \tag{4.7}$$

Proof mechanism and scope. Set $\rho_{\min} = \min_j \rho(d^{\pi_j})$ and shift every score by $c\rho_{\min}$. This leaves the action rule unchanged and makes the source penalties nonnegative. Bellman monotonicity then gives $Q_T^{\pi_T} \geq \max_j\{Q_T^{\pi_j} - c(\rho(d^{\pi_j}) - \rho_{\min})\}$. Comparing each source with π^* yields the reward bound; comparing the two actual occupancy risks adds $2Lc$. Appendix B gives the proofs. The

comparator is risk-neutral, and neither guarantee asserts hard-constraint satisfaction or risk-optimality. Uniform source-score error η adds $2\gamma\eta/(1-\gamma)$ to both bounds (Appendix B.5). The convex-risk source-hull bridge in Appendix C.1 provides a separate diagnostic of realized risk.

4.2 Why Risk-Neutral Sources Are Useful

A central design claim in TRAM is that source policies should remain risk-neutral when the future deployment risk is unknown. Intuitively, risk-aware source training can prune away behaviors that look unsafe under one guessed risk but are valuable under a different future risk. The next result formalizes a conditional source-library design principle: broader coverage helps when the risk-neutral library's occupancy hull contains the hulls of the alternatives. Risk-neutral training alone does not guarantee that inclusion.

Theorem 4.3 (Conditional source-hull dominance)**.** *Let $S_\lambda \subseteq \mathcal{D}$ be source occupancy libraries, with $\lambda = 0$ denoting risk-neutral training. Assume $\mathrm{co}(S_\lambda) \subseteq \mathrm{co}(S_0)$ for all $\lambda \geq 0$, and that every hull admits an occupancy with $\langle d, r_T \rangle \geq V$. For a nonempty class $\mathcal{R}_L$ of finite L-Lipschitz risks on $\mathcal{D}$,*

$$\sup_{\rho \in \mathcal{R}_L} \mathcal{E}_{\rho,V}(S_0) = \inf_{\lambda \geq 0} \sup_{\rho \in \mathcal{R}_L} \mathcal{E}_{\rho,V}(S_\lambda), \tag{4.8}$$

where the nonnegative risk regret is

$$\mathcal{E}_{\rho,V}(S) = \inf_{\substack{d \in \mathrm{co}(S) \\ \langle d, r_T \rangle \geq V}} \rho(d) - \inf_{\substack{d \in \mathcal{D} \\ \langle d, r_T \rangle \geq V}} \rho(d).$$

The proof is feasible-set inclusion (Proposition B.13). It concerns the best occupancy in each hull, not the occupancy attained by greedy source switching. Hull inclusion must be established for the source libraries being compared.

4.3 Practical Implementation: Successor Features and Alternatives

TRAM needs repeated estimates of $Q_T^{\pi_j}(s,a)$ for many sources and actions. Successor features are useful because they turn this value computation into a fast dot product whenever rewards are linear in shared features. If $r(s,a,s') = \phi(s,a,s')^\top w$, then

$$\psi^\pi(s,a) = \mathbb{E}^\pi \left[\sum_{t \geq 0} \gamma^t \phi(S_t, A_t, S_{t+1}) \,\middle|\, S_0 = s, A_0 = a \right], \tag{4.9}$$

$$Q^\pi(s,a) = \psi^\pi(s,a)^\top w. \tag{4.10}$$

Thus source values can be evaluated under a new target reward without retraining the source policy or running a new Bellman solve.

Corollary 4.4 (Successor-feature transfer bound)**.** *Under Theorem 4.1's assumptions and $\|\phi(s,a,s')\|_2 \leq \phi_{\max}$,*

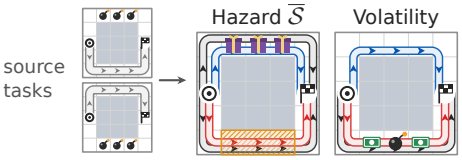
$$\widetilde{Q}_T^{\pi^*} - \widetilde{Q}_T^{\pi_T} \leq \frac{2\phi_{\max}}{1-\gamma} \min_j \|w_T - w_j\|_2 + 4Lc. \tag{4.11}$$

The factor two accounts for the two trajectory discrepancies. Successor features are an efficient implementation route, not a definitional requirement of TRAM. When linear reward structure is unavailable, $Q_T^{\pi_j}$ can be estimated with target-task rollouts, fitted value functions, or model-based evaluation, while the source-scored deployment rule remains unchanged. The score-error bound quantifies the additional effect of evaluation error. No source policy parameters are changed after deployment in any case.

5 Experiments

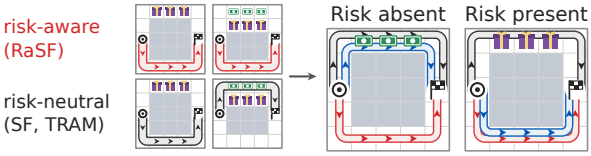
We evaluate TRAM in four settings using the zero-update source-scoring logic in Algorithm 1: gridworld mechanisms, Reacher with deep successor features, Safety-Gymnasium with deployment-time spatial risk, and LLM alignment with behavioral-risk adaptation.

(a) Risk types beyond trajectory variance



	Hazard $\overline{\mathcal{S}}$			Volatility	
	Barrier ↓	Entry $\overline{\mathcal{S}}$ ↓	$\mathbb{E}[G]$ ↑	$\mathrm{Var}[G]$ ↓	$\mathbb{E}[G]$ ↑
SF	1.12	57%	3.95	0.06	22.98
RaSF	1.89	100%	3.95	0.06	22.98
TRAM	**0.69**	**1%**	3.95	**0.00**	22.98

(b) Adaptation to changing risk conditions



	Risk absent	Risk present	
	$\mathbb{E}[G]$ ↑	$\mathbb{E}[G]$ ↑	$\mathrm{Var}[G]$ ↓
SF	**22.62**	**18.19**	104.87
RaSF	16.10	16.10	93.33
TRAM	22.49	16.10	**93.14**

Figure 3: Gridworld mechanism validation. **(a)** Spatial hazards and per-step volatility expose trajectory-variance limitations; TRAM succeeds through occupancy-based risk modeling while baselines fail. **(b)** TRAM exploits high-reward paths when risk is absent and becomes conservative when risk appears, while RaSF remains uniformly conservative due to training-time risk aversion. Small grids show source policies; in (b), red sources are trained risk-aware (used by RaSF) and black sources risk-neutral (used by SF and TRAM). G is the return; bold marks the best value in each column.

Baselines. We compare against risk-neutral generalized policy improvement (GPI) and risk-agnostic successor-feature transfer (SF) [1–3], risk-aware successor features (RaSF) [12], and, in Safety-Gymnasium, a frozen SafeDSR-style proxy [13] and a one-step SFT-CoP-style dual baseline inspired by transfer-Q/source-composition decoding and constrained adaptation [38, 48]. These baselines test whether reward transfer alone suffices, whether variance-style risk matches deployment hazards, and whether frozen or dual-style safety proxies can match Algorithm 1.

Across settings, the evaluation separates utility from the risk requested at deployment. This is important because a single return score can hide the failure mode TRAM targets: a source policy may be high-reward under the target task while accumulating the wrong kind of risk. We therefore report task performance together with the domain-specific risk measure used by the deployment score, and use the appendix diagnostics when the realized stitched behavior must be distinguished from cached source risks. The suite separates three stress tests: misspecified variance risk in small MDPs, hazards appearing after continuous-control source training, and reward-model overoptimization in decoding. In all cases, the deployed sources remain fixed.

5.1 Gridworld: Mechanism Validation

Controlled gridworlds test whether TRAM supports expressive risk models and avoids training-time conservatism. Risk appears as spatial hazards or per-step volatility. We compare TRAM against RaSF [12] and risk-agnostic SF [1].

Figure 3(a) demonstrates TRAM's expressive risk modeling. For spatial hazards, TRAM uses barrier risk to avoid danger zones while RaSF and SF fail because trajectory variance provides no spatial information. For per-step volatility, TRAM detects locally risky paths through step-level variance while trajectory-based approaches miss this fine-grained structure. Figure 3(b) shows adaptive risk sensitivity: TRAM exploits opportunities when risk-free but becomes conservative when risk appears, while RaSF remains uniformly conservative due to training-time risk aversion.

5.2 Continuous Control: Reacher

We evaluate scalability on Reacher (two-joint arm, 4D continuous state, MuJoCo physics). Four SFDQNs train risk-neutrally on different goals; at test time, new goals introduce danger zones. TRAM reduces barrier violations compared to risk-neutral transfer while maintaining goal performance. Successor features enable real-time adaptation through $Q_T^\pi(s,a) = \psi^\pi(s,a)^\top w_T$. This yields $\mathcal{O}(n|\mathcal{A}|d)$ dot-product scoring per step across n sources and feature dimension d. Details are in Appendix D.1.

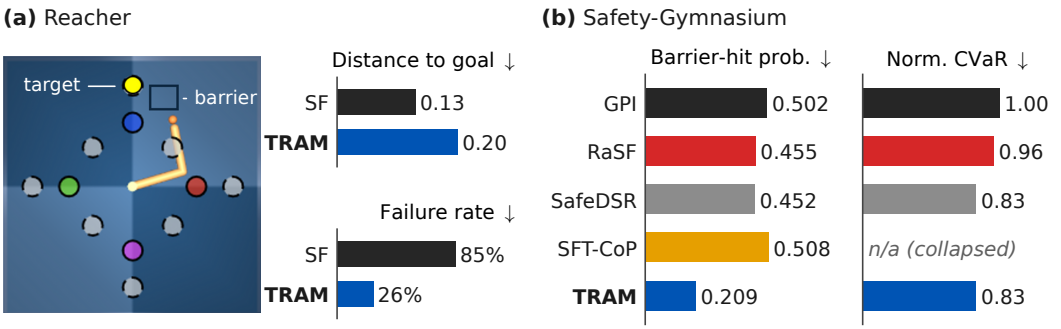


Figure 4: Continuous-control deployment-time risk results. (a) In Reacher, a new target goal and barrier are specified after source training; TRAM substantially reduces failures relative to risk-neutral SF transfer, with a modest increase in final distance to the goal. **(b)** In Safety-Gymnasium, TRAM directly scores the requested deployment barrier risk and achieves the lowest barrier-hit probability while matching the lowest normalized CVaR among valid methods.

Figure 4(a) visualizes the Reacher test setting and quantitative outcomes. TRAM prioritizes avoiding the deployment-time barrier, cutting the failure rate from 85% to 26%, while keeping the distance-to-goal comparable to risk-neutral transfer. The comparison asks whether a fixed risk-neutral library can be re-scored when a barrier appears after training, rather than whether a new safe controller can be learned from scratch.

5.3 Safety-Gymnasium: Deployment-Time Spatial Risk

To test a higher-dimensional continuous-control setting, we use Safety-Gymnasium `SafetyPointGoal0-v0` [19], a continuous-control safe-RL benchmark with a 16-dimensional observation representation. The target goal is fixed at $(0.7, 0.7)$ and a circular deployment barrier is placed at $(0.245, 0.245)$ with radius 0.18. All methods use the same source policy library and perform no parameter updates at deployment. We evaluate 1000 episodes with horizon 1000 and $\gamma = 0.99$. Additional benchmark details are in Appendix D.2.

Figure 4(b) shows that TRAM reduces barrier-hit probability from 0.502 to 0.209. Variance-based RaSF improves only modestly because return variance is not aligned with the circular barrier exposure. The frozen SafeDSR proxy is close on normalized CVaR but worse on hit probability, reflecting the limitation of a non-adaptive safety proxy. SFT-CoP is omitted from the normalized-CVaR comparison because it enters a collapsed-task regime and does not provide a valid reward–risk tradeoff. We report both barrier-hit probability and normalized CVaR because they capture different failure modes: direct contact with the deployment hazard and tail exposure among risky rollouts. This directly tests the central use case: a structured spatial hazard specified only at deployment, handled by source scoring rather than retraining.

Constraint-aware baseline and deployment cost. Under a separate held-out-goal protocol (Appendix D.3), constrained PPO trained on the deployment task for about 40 minutes reaches 99.7% success but a barrier-hit probability of 0.24, above its 10% target. TRAM trains nothing; after re-estimating source risks from rollouts, it costs about 2.6 s of inference per episode and reaches 0.086 at $c = 10^6$, though with only 10% success; smaller c trades risk for success (Table 2, Appendix D.5).

5.4 LLM Alignment: Behavioral Risk without Retraining

We also test the same principle in an inference-time LLM alignment setting [28, 7]. TRAM composes two pretrained 7B source models, Zephyr-Qwen-2-7B and Dolphin-Qwen-2-7B, under a new deployment-time reward/risk specification on Berkeley Nectar [53]. The target reward is given by a reward model, while deployment risk is a KL-style alignment penalty to a safe reference behavior. This differs from tabular and robotic control: the action space is text generation and adaptation must occur without fine-tuning source models. Additional protocol details are in Appendix E. This tests the same zero-update abstraction at the token level, as in Collab [7], using external reward and risk estimators.

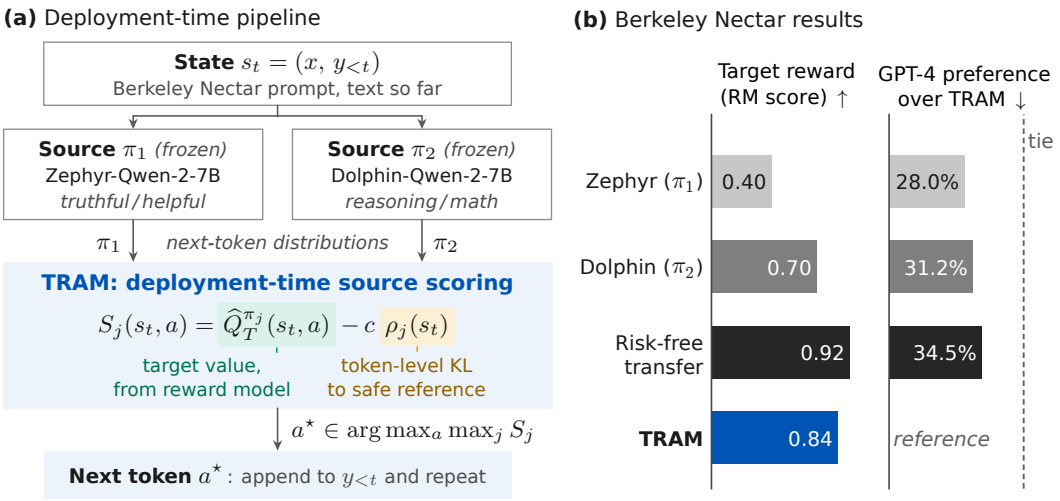


Figure 5: LLM alignment on Berkeley Nectar. The left panel summarizes the zero-update evaluation protocol using fixed source models, a target reward model, and a safe reference. The right panel shows that risk-free transfer maximizes the target reward-model score, while GPT-4 pairwise preferences favor TRAM overall.

Figure 5 illustrates why target-reward maximization alone can be misleading: the risk-free transfer policy obtains the highest reward-model score but is still preferred less often than TRAM by GPT-4 pairwise comparison. TRAM uses deployment-time behavioral risk to avoid overoptimizing the target reward model [10], showing that the same source-scored rule can support risks beyond variance and low-dimensional control.

This experiment is a scope test for the source-scored principle rather than an occupancy-control claim. Unlike the gridworld and continuous-control settings, the behavioral risk is evaluated on the sources' next-token distributions, not through discounted state-action occupancies. The common structure is deployment-time scoring: fixed sources remain unchanged, the target reward model supplies the utility term, and the newly requested alignment penalty shifts which source generates the next token. Operationally, each source proposes a reusable behavior, and deployment changes only the scoring criterion. The result should therefore be read as token-level risk adaptation, not as a Markov occupancy-risk theorem for language generation. This supports the broader claim that TRAM is most useful when risk specifications change after the source behaviors are already available.

6 Conclusion

We introduced TRAM, a zero-update deployment-time framework that composes risk-neutral source policies using source-scored reward–risk values. The formulation treats TRAM as a tractable source-scored surrogate rather than an exact optimizer of the stitched policy's full occupancy objective, with Appendix C linking cached source risk to realized risk through a measurable stitching-mismatch term. Empirically, TRAM improves deployment safety across gridworlds, Reacher, Safety-Gymnasium, and LLM alignment without retraining. TRAM separates predeployment learning of diverse behavior from deployment-time selection of the requested risk–reward tradeoff. **Limitations.** The main limitation is source-library coverage and risk-estimation fidelity, especially if all source policies are poor. Very safe operating points can also cost task success (Appendix D.5).

Acknowledgments and Disclosure of Funding

This work was supported in part by the U.S. National Science Foundation (NSF) under awards ECCS-1802319, 1751085, 2130706, and 2150596, and by the U.S. Army Research Office (ARO) under Grants W911NF2310266 and W911NF261A366. We also thank the Texas Advanced Computing Center (TACC) at The University of Texas at Austin for providing part of the computing resources used in this work. The authors declare no competing interests.

References

[1] André Barreto, Will Dabney, Rémi Munos, Jonathan J Hunt, Tom Schaul, Hado P van Hasselt, and David Silver. Successor features for transfer in reinforcement learning. *Advances in neural information processing systems*, 30, 2017.

[2] Andre Barreto, Diana Borsa, John Quan, Tom Schaul, David Silver, Matteo Hessel, Daniel Mankowitz, Augustin Zidek, and Remi Munos. Transfer in deep reinforcement learning using successor features and generalised policy improvement. In *International Conference on Machine Learning*, pages 501–510. PMLR, 2018.

[3] André Barreto, Shaobo Hou, Diana Borsa, David Silver, and Doina Precup. Fast reinforcement learning with generalized policy updates. *Proceedings of the National Academy of Sciences*, 117(48):30079–30087, 2020.

[4] Lorenzo Bisi, Luca Sabbioni, Edoardo Vittori, Matteo Papini, and Marcello Restelli. Risk-averse trust region optimization for reward-volatility reduction. *arXiv preprint arXiv:1912.03193*, 2019.

[5] Wilka Carvalho, Andre Saraiva, Angelos Filos, Andrew Kyle Lampinen, Loic Matthey, Richard L. Lewis, Honglak Lee, Satinder Singh, Danilo J. Rezende, and Daniel Zoran. Combining behaviors with the successor features keyboard. In *Advances in Neural Information Processing Systems*, volume 36, 2023.

[6] Souradip Chakraborty, Soumya Suvra Ghosal, Ming Yin, Dinesh Manocha, Mengdi Wang, Amrit Singh Bedi, and Furong Huang. Transfer Q-star: Principled decoding for LLM alignment. In *Advances in Neural Information Processing Systems*, volume 37, 2024.

[7] Souradip Chakraborty, Sujay Bhatt, Udari Madhushani Sehwag, Soumya Suvra Ghosal, Jiahao Qiu, Mengdi Wang, Dinesh Manocha, Furong Huang, Alec Koppel, and Sumitra Ganesh. Collab: Controlled decoding using mixture of agents for LLM alignment. In *International Conference on Learning Representations*, 2025.

[8] Josef Dai, Xuehai Pan, Ruiyang Sun, Jiaming Ji, Xinbo Xu, Mickel Liu, Yizhou Wang, and Yaodong Yang. Safe RLHF: Safe reinforcement learning from human feedback. In *International Conference on Learning Representations*, 2024.

[9] Yingjie Fei, Zhuoran Yang, Yudong Chen, Zhaoran Wang, and Qiaomin Xie. Risk-sensitive reinforcement learning: Near-optimal risk-sample tradeoff in regret. *Advances in Neural Information Processing Systems*, 33:22384–22395, 2020.

[10] Leo Gao, John Schulman, and Jacob Hilton. Scaling laws for reward model overoptimization. In *International Conference on Machine Learning*, pages 10835–10866, 2023.

[11] Javier García and Fernando Fernández. Probabilistic policy reuse for safe reinforcement learning. *ACM Transactions on Autonomous and Adaptive Systems (TAAS)*, 13(3):1–24, 2019.

[12] Michael Gimelfarb, André Barreto, Scott Sanner, and Chi-Guhn Lee. Risk-aware transfer in reinforcement learning using successor features. *Advances in Neural Information Processing Systems*, 34:17298–17310, 2021.

[13] Michael Girstl, Alexander Mattick, and Christopher Mutschler. Constrained reinforcement learning using successor representations. *Transactions on Machine Learning Research*, 2026. URL https://openreview.net/forum?id=6zUq7knzwA.

[14] David Held, Zoe McCarthy, Michael Zhang, Fred Shentu, and Pieter Abbeel. Probabilistically safe policy transfer. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5798–5805. IEEE, 2017.

[15] Irina Higgins, Arka Pal, Andrei Rusu, Loic Matthey, Christopher Burgess, Alexander Pritzel, Matthew Botvinick, Charles Blundell, and Alexander Lerchner. DARLA: Improving zero-shot transfer in reinforcement learning. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 1480–1490. PMLR, 2017.

[16] Garud N. Iyengar. Robust dynamic programming. *Mathematics of Operations Research*, 30(2):257–280, 2005.

[17] Arushi Jain, Khimya Khetarpal, and Doina Precup. Safe option-critic: learning safety in the option-critic architecture. *The Knowledge Engineering Review*, 36:e4, 2021.

[18] Arushi Jain, Gandharv Patil, Ayush Jain, Khimya Khetarpal, and Doina Precup. Variance penalized on-policy and off-policy actor-critic. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 7899–7907, 2021.

[19] Jiaming Ji, Jiayi Zhou, Borong Zhang, Jiemin Dai, Xuehai Pan, Ruiyang Sun, Weidong Huang, Yiran Geng, Mingyang Liu, and Yaodong Yang. Safety-gymnasium: A unified safe reinforcement learning benchmark. In *Advances in Neural Information Processing Systems Datasets and Benchmarks Track*, 2023.

[20] Maxim Khanov, Jirayu Burapacheep, and Yixuan Li. ARGS: Alignment as reward-guided search. In *International Conference on Learning Representations*, 2024.

[21] Junseo Lee, Jaeseok Heo, Dohyeong Kim, Gunmin Lee, and Songhwai Oh. Dual variable actor-critic for adaptive safe reinforcement learning. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2023.

[22] Zuxin Liu, Zijian Guo, Yihang Yao, Zhepeng Cen, Wenhao Yu, Tingnan Zhang, and Ding Zhao. Constrained decision transformer for offline safe reinforcement learning. In *International Conference on Machine Learning*, 2023.

[23] Daniel Mankowitz, Timothy Mann, Pierre-Luc Bacon, Doina Precup, and Shie Mannor. Learning robust options. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018.

[24] Daniel J Mankowitz, Aviv Tamar, and Shie Mannor. Situational awareness by risk-conscious skills. *arXiv preprint arXiv:1610.02847*, 2016.

[25] Shie Mannor and John N Tsitsiklis. Algorithmic aspects of mean–variance optimization in markov decision processes. *European Journal of Operational Research*, 231(3):645–653, 2013.

[26] Hongzi Mao, Shaileshh Bojja Venkatakrishnan, Malte Schwarzkopf, and Mohammad Alizadeh. Variance reduction for reinforcement learning in input-driven environments. *arXiv preprint arXiv:1807.02264*, 2018.

[27] Ofir Marom and Benjamin Rosman. Zero-Shot Transfer with Deictic Object-Oriented Representation in Reinforcement Learning. In *Advances in Neural Information Processing Systems*, volume 31, 2018.

[28] Sidharth Mudgal, Jong Lee, Harish Ganapathy, YaGuang Li, Tao Wang, Yanping Huang, Zhifeng Chen, Heng-Tze Cheng, Michael Collins, Trevor Strohman, Jilin Chen, Alex Beutel, and Ahmad Beirami. Controlled decoding from language models. In *International Conference on Machine Learning*, 2024.

[29] David Nass, Boris Belousov, and Jan Peters. Entropic risk measure in policy search. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1101–1106. IEEE, 2019.

[30] Arnab Nilim and Laurent El Ghaoui. Robust control of Markov decision processes with uncertain transition matrices. *Operations Research*, 53(5):780–798, 2005.

[31] Junhyuk Oh, Satinder Singh, Honglak Lee, and Pushmeet Kohli. Zero-Shot Task Generalization with Multi-Task Deep Reinforcement Learning. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 2661–2670. PMLR, 2017.

[32] Kishan Panaganti and Dileep Kalathil. Sample complexity of robust reinforcement learning with a generative model. In *International Conference on Artificial Intelligence and Statistics*, pages 9582–9602, 2022.

[33] Seohong Park, Tobias Kreiman, and Sergey Levine. Foundation policies with Hilbert representations. In *International Conference on Machine Learning*, 2024.

[34] Matteo Pirotta, Andrea Tirinzoni, Ahmed Touati, Alessandro Lazaric, and Yann Ollivier. Fast imitation via behavior foundation models. In *International Conference on Learning Representations*, 2024.

[35] Alexandre Ramé, Guillaume Couairon, Corentin Dancette, Jean-Baptiste Gaya, Mustafa Shukor, Laure Soulier, and Matthieu Cord. Rewarded soups: Towards Pareto-optimal alignment by interpolating weights fine-tuned on diverse rewards. In *Advances in Neural Information Processing Systems*, volume 36, 2023.

[36] Sahand Rezaei-Shoshtari, Charlotte Morissette, Francois R. Hogan, Gregory Dudek, and David Meger. Hypernetworks for zero-shot transfer in reinforcement learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pages 9579–9587, 2023. doi: 10.1609/aaai.v37i8.26146.

[37] Yun Shen, Michael J Tobia, Tobias Sommer, and Klaus Obermayer. Risk-sensitive reinforcement learning. *Neural computation*, 26(7):1298–1328, 2014.

[38] Harshit Sikchi, Qinqing Zheng, Amy Zhang, and Scott Niekum. Dual RL: Unification and new methods for reinforcement and imitation learning. In *International Conference on Learning Representations*, 2024.

[39] Charlie Snell, Jaehoon Lee, Kelvin Xu, and Aviral Kumar. Scaling LLM test-time compute optimally can be more effective than scaling parameters for reasoning. In *International Conference on Learning Representations*, 2025.

[40] Krishnan Srinivasan, Benjamin Eysenbach, Sehoon Ha, Jie Tan, and Chelsea Finn. Learning to be safe: Deep rl with a safety critic. *arXiv preprint arXiv:2010.14603*, 2020.

[41] Aviv Tamar, Dotan Di Castro, and Shie Mannor. Learning the variance of the reward-to-go. *Journal of Machine Learning Research*, 17(13):1–36, 2016.

[42] Andrea Tirinzoni, Ahmed Touati, Jesse Farebrother, Mateusz Guzek, Anssi Kanervisto, Yingchen Xu, Alessandro Lazaric, and Matteo Pirotta. Zero-shot whole-body humanoid control via behavioral foundation models. In *International Conference on Learning Representations*, 2025.

[43] Emanuel Todorov, Tom Erez, and Yuval Tassa. Mujoco: A physics engine for model-based control. In *2012 IEEE/RSJ international conference on intelligent robots and systems*, pages 5026–5033. IEEE, 2012.

[44] Ahmed Touati, Jérémy Rapin, and Yann Ollivier. Does zero-shot reinforcement learning exist? In *International Conference on Learning Representations*, 2023.

[45] Matteo Turchetta, Andrey Kolobov, Shital Shah, Andreas Krause, and Alekh Agarwal. Safe reinforcement learning via curriculum induction. *Advances in Neural Information Processing Systems*, 33:12151–12162, 2020.

[46] Junlin Wang, Jue Wang, Ben Athiwaratkun, Ce Zhang, and James Zou. Mixture-of-agents enhances large language model capabilities. In *International Conference on Learning Representations*, 2025.

[47] Yihang Yao, Zuxin Liu, Zhepeng Cen, Jiacheng Zhu, Wenhao Yu, Tingnan Zhang, and Ding Zhao. Constraint-conditioned policy optimization for versatile safe reinforcement learning. In *Advances in Neural Information Processing Systems*, 2023.

[48] Junyu Zhang, Amrit Singh Bedi, Mengdi Wang, and Alec Koppel. Cautious reinforcement learning via distributional risk in the dual domain. *IEEE Journal on Selected Areas in Information Theory*, 2(2): 611–626, 2021.

[49] Shangtong Zhang, Bo Liu, and Shimon Whiteson. Mean-variance policy iteration for risk-averse reinforcement learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 10905–10913, 2021. doi: 10.1609/aaai.v35i12.17302.

[50] Shuai Zhang, Heshan Devaka Fernando, Miao Liu, Keerthiram Murugesan, Songtao Lu, Pin-Yu Chen, Tianyi Chen, and Meng Wang. Sf-dqn: Provable knowledge transfer using successor feature for deep reinforcement learning. *arXiv preprint arXiv:2405.15920*, 2024.

[51] Yinan Zheng, Jianxiong Li, Dongjie Yu, Yujie Yang, Shengbo Eben Li, Xianyuan Zhan, and Jingjing Liu. Safe offline reinforcement learning with feasibility-guided diffusion model. In *International Conference on Learning Representations*, 2024.

[52] Zhengqing Zhou, Zhengyuan Zhou, Qinxun Bai, Linhai Qiu, Jose Blanchet, and Peter Glynn. Finite-sample regret bound for distributionally robust offline tabular reinforcement learning. In *International Conference on Artificial Intelligence and Statistics*, pages 3331–3339, 2021.

[53] Banghua Zhu, Evan Frick, Tianhao Wu, Hanlin Zhu, and Jiantao Jiao. Starling-7B: Improving LLM helpfulness & harmlessness with RLAIF, 2023. URL https://starling.cs.berkeley.edu/. Nectar dataset: https://huggingface.co/datasets/berkeley-nest/Nectar.

Appendix

TRAM: Test-Time Risk Adaptation with Mixture of Agents

A Linear Programming Views of RL and Occupancy Measures **15**

A.1 Policy Evaluation via Linear Programming . . . 15

A.2 Risk-Regularized Control via Occupancy Optimization . . . 15

A.3 Risk Functionals on Occupancies: Compact Catalog . . . 16

A.4 Connection to TRAM . . . 17

B Proofs and Technical Guarantees **17**

B.1 Basic Value-Difference Lemmas . . . 18

B.2 Generalized Improvement with Source-Level Penalties . . . 19

B.3 Localized Transfer Bounds . . . 19

B.4 Main-Text Corollaries and Source-Library Design . . . 20

B.5 Approximate Source Scores . . . 21

C Stitching Mismatch and Approximate-Dynamics Extensions **21**

C.1 Stitching Mismatch and the Source-Hull Bridge . . . 21

C.2 Approximate Dynamics and Source Evaluation . . . 22

D Continuous-Control and Safety-Gymnasium Experimental Details **23**

D.1 Reacher Implementation Details . . . 23

D.2 Safety-Gymnasium Benchmark Details . . . 23

D.3 Additional Safety-Gymnasium Experiments: Protocol . . . 23

D.4 Constraint-Aware Baseline and Deployment Cost . . . 24

D.5 Sensitivity to the Tradeoff Weight c . . . 25

D.6 Alternative Risk Functional: Log-Barrier . . . 25

D.7 Cost of Occupancy Estimation . . . 25

E LLM Alignment Experiment **26**

E.1 Setup . . . 26

E.2 Metrics and Interpretation . . . 26

F Reproducibility and Implementation Details **27**

F.1 Core Software Stack . . . 27

F.2 Computational Resources . . . 27

G Occupancy Storage, Risk Computation, and Runtime **27**

H Impact Statement **28**

A Linear Programming Views of RL and Occupancy Measures

A.1 Policy Evaluation via Linear Programming

Consider a finite MDP with state space $\mathcal{S}$, action space $\mathcal{A}$, discount $\gamma \in (0,1)$, transition $p(\cdot \mid s,a)$, reward $r(s,a,s')$, and initial distribution μ_0. Define

$$r(s,a) := \mathbb{E}_{s' \sim p(\cdot|s,a)}\big[r(s,a,s')\big], \tag{A.1}$$

$$\mathbb{P}_\pi(s' \mid s) := \sum_a \pi(a \mid s)\, p(s' \mid s,a). \tag{A.2}$$

Primal Q–LP (policy evaluation). For fixed π,

$$\begin{aligned} \min_{Q:\mathcal{S}\times\mathcal{A}\to\mathbb{R}} \quad & (1-\gamma)\,\mathbb{E}_{s_0\sim\mu_0,\, a_0\sim\pi(\cdot|s_0)}\big[Q(s_0,a_0)\big] \\ \text{s.t.} \quad & Q(s,a) \geq r(s,a) + \gamma\, \mathbb{E}_{\substack{s'\sim p(\cdot|s,a)\\ a'\sim\pi(\cdot|s')}}\big[Q(s',a')\big]. \end{aligned} \tag{A.3}$$

Dual: discounted occupancies. Introducing nonnegative multipliers $d(s,a) \geq 0$ yields

$$\begin{aligned} \max_{d\geq 0} \quad & \sum_{s,a} d(s,a)\, r(s,a) \\ \text{s.t.} \quad & d(s,a) = (1-\gamma)\mu_0(s)\pi(a \mid s) \\ & \qquad + \gamma \sum_{s',a'} d(s',a') p(s \mid s',a') \pi(a \mid s), \quad \forall (s,a). \end{aligned} \tag{A.4}$$

At optimality $d = d^\pi$. The normalized discounted occupancy is

$$d^\pi(s,a) = (1-\gamma) \sum_{t=0}^{\infty} \gamma^t \Pr_\pi(S_t = s, A_t = a \mid S_0 \sim \mu_0). \tag{A.5}$$

Strong duality gives $(1-\gamma)\,\mathbb{E}_{\mu_0,\pi}[Q^\pi(S_0,A_0)] = \langle d^\pi, r\rangle$. If $\nu(s) := \sum_a d(s,a) > 0$, a policy is recovered by $\pi(a \mid s) = d(s,a)/\nu(s)$.

A.2 Risk-Regularized Control via Occupancy Optimization

Define the *occupancy polytope*

$$\begin{aligned} \mathcal{D} = \Big\{ d \in \mathbb{R}^{\mathcal{S}\times\mathcal{A}}_{\geq 0} : \underbrace{\sum_a d(s,a)}_{\nu(s)} &= (1-\gamma)\mu_0(s) \\ &+ \gamma \sum_{s',a'} d(s',a')\, p(s \mid s',a'),\ \forall s \Big\}. \end{aligned} \tag{A.6}$$

Given a risk functional $\rho : \mathcal{D} \to \mathbb{R} \cup \{+\infty\}$ with nonempty finite domain, the penalized control problem for $c_{\mathrm{occ}} > 0$ is

$$\max_{d\in\mathcal{D}} \langle d, r\rangle - c_{\mathrm{occ}}\, \rho(d), \qquad c_{\mathrm{occ}} > 0, \tag{A.7}$$

whose optimizer, when it exists, induces $\pi^\star(a \mid s) = d^\star(s,a)/\nu^\star(s)$ wherever $\nu^\star(s) > 0$; actions at zero-occupancy states can be chosen arbitrarily. At $c_{\mathrm{occ}} = 0$, use the ordinary reward-only occupancy problem without evaluating $0 \cdot \rho(d)$ at infinite risks. To match the unnormalized-return objective in Eq. 3.2, use $c_{\mathrm{occ}} = (1-\gamma)c$. For one-step functionals, we use the occupancy-weighted expectation

$$\mathbb{E}_d[\phi(s,a,s')] := \sum_{s,a} d(s,a) \sum_{s'} p(s' \mid s,a)\, \phi(s,a,s'). \tag{A.8}$$

A.3 Risk Functionals on Occupancies: Compact Catalog

All functionals below depend *only* on d (and possibly a fixed reference $\bar{d}$) and thus fit A.7.

(R1) Set-occupancy constraints (barrier / chance proxy). For danger set $\overline{\mathcal{S}}$ and tolerance $\delta \in (0,1)$,

$$\rho_{\text{set}}(d) \;=\; -\log\left(\delta - \sum_{s\in\overline{\mathcal{S}}} \nu(s)\right), \quad \text{valid when } \sum_{s\in\overline{\mathcal{S}}} \nu(s) < \delta, \tag{A.9}$$

which penalizes discounted state-mass in $\overline{\mathcal{S}}$. This is an exposure surrogate, not the probability of ever hitting the danger set.

(R2) Per-step reward variance (local volatility). Let $\tilde{r}(s,a) := \mathbb{E}_{s'|s,a}[r(s,a,s')^2]$. Since $\mathbb{E}_d[r] = \langle d, r\rangle$ under the normalized occupancy,

$$\rho_{\text{var}}(d) \;=\; \langle d, \tilde{r}\rangle - \big(\langle d, r\rangle\big)^2, \tag{A.10}$$

the variance of a single reward drawn from the occupancy-weighted transition distribution. It can be positive for deterministic trajectories whose rewards vary across states. It is generally concave, not convex, as a function of d.

(R3) f-divergences to a safe reference (behavioral proximity). For a reference occupancy $\bar{d}$ with matching support, the KL case is

$$\rho_{\text{KL}}(d) = \sum_{s,a} d(s,a)\log\frac{d(s,a)}{\bar{d}(s,a)}. \tag{A.11}$$

Other f-divergences, such as χ^2, also depend only on the two occupancies. Rényi divergences provide another family of occupancy comparisons, with convexity depending on the order and argument.

(R4) Integral probability metrics (IPMs), e.g., MMD. Let Φ be a feature map into an RKHS $\mathcal{H}$:

$$\rho_{\text{MMD}}(d) = \Big\| \sum_{s,a} d(s,a)\Phi(s,a) - \sum_{s,a} \bar{d}(s,a)\Phi(s,a)\Big\|_{\mathcal{H}}. \tag{A.12}$$

This provides a nonparametric proximity to a safe/reference occupancy.

(R5) Entropic (exponential) per-step risk. Given a step risk cost $g(s,a,s')$ and $\theta > 0$,

$$\rho_{\text{ent}}(d) \;=\; \tfrac{1}{\theta}\log\Big(\mathbb{E}_d\big[e^{\theta\, g(s,a,s')}\big]\Big), \tag{A.13}$$

a tail-sensitive single-step penalty. For fixed g, the positive expectation is affine in d, so its logarithm is concave in d. Convexity in the losses must not be confused with convexity in occupancy.

(R6) Energy / actuation budgets (control cost). With an action embedding $u(s,a) \in \mathbb{R}^m$ and $G \succeq 0$,

$$\rho_{\text{energy}}(d) = \sum_{s,a} d(s,a)\, u(s,a)^\top G\, u(s,a), \tag{A.14}$$

which constrains power, wear, or acceleration magnitudes.

(R7) Transition smoothness / jerk (temporal regularity). Let $\varphi(s,a,s')$ encode finite differences (e.g., Δs); a control increment or jerk requires the relevant control history to be included in the state. Define

$$\rho_{\text{smooth}}(d) = \mathbb{E}_d\big[\|\varphi(s,a,s')\|_2^2\big], \tag{A.15}$$

penalizing oscillatory behaviors via occupancy-weighted transitions.

(R8) Reach–avoid proxy (linear temporal-logic surrogate). For target set $\mathcal{T}$ and avoid set $\overline{\mathcal{S}}$,

$$\rho_{\mathrm{RA}}(d) = \alpha \sum_{s\in\overline{\mathcal{S}}} \nu(s) + \beta \left(1 - \sum_{s\in\mathcal{T}} \nu(s)\right), \qquad \alpha, \beta \geq 0, \tag{A.16}$$

a linear-in-d proxy for STL/LTL reach–avoid objectives.

Design notes. (i) Each ρ above is a function of $(d, \bar{d})$ only; no return distributions or value iterations are needed. (ii) KL, the IPM norm, and the log barrier on its domain are convex in d; fixed-cost energy, smoothness, and reach–avoid penalties are linear in d. These choices make the penalized occupancy objective concave. The variance and entropic examples do not share that property. (iii) The analysis requires a finite Lipschitz constant on the occupancies being compared. Bounded linear costs satisfy this condition. A log barrier requires slack $\delta - \sum_{s\in\overline{\mathcal{S}}} \nu(s) \geq \varepsilon > 0$, giving $L \leq 1/\varepsilon$. KL has a bounded gradient on domains with density ratios bounded above and away from zero; positive reference mass alone does not make KL globally Lipschitz up to the boundary. A source satisfying these domain restrictions does not imply that the stitched policy satisfies them.

A.4 Connection to TRAM

In TRAM, each risk-neutral source π_j is evaluated on the *target* to obtain its occupancy d^{π_j} and risk score $\rho(d^{\pi_j})$. The risk-adjusted utility

$$\mathcal{U}_j(s,a) = Q^{\pi_j}(s,a) - c\,\rho\big(d^{\pi_j}\big) \tag{A.17}$$

drives per-state action selection (Eq. 4.4) without any parameter updates. Risk is specified through ρ and can be changed at deployment. Diverse sources can improve coverage, but risk-neutral training does not automatically maximize coverage of $\mathcal{D}$; the source-hull comparison in Theorem 4.3 requires explicit hull inclusion.

B Proofs and Technical Guarantees

Analysis setting. We consider finite discounted MDPs with bounded expected one-step rewards, common transition kernel p, discount $\gamma \in (0,1)$, and a fixed initial distribution μ_0. Policies are stationary. A superscript $*$ denotes a policy optimal from every state, not merely for one initial distribution. Each source policy π_j is statewise optimal for its source reward r_j under p, so $Q_j^{\pi_j} = Q_j^*$. Unless estimation error is stated explicitly, action values and source risks are exact. Any hard source screen leaves a nonempty, fixed library, and all source indices below refer to that retained library.

The normalized discounted occupancy is

$$d^\pi(s,a) = (1-\gamma) \sum_{t\geq 0} \gamma^t \Pr_\pi(S_t = s, A_t = a \mid S_0 \sim \mu_0).$$

Thus $\|d^\pi\|_1 = 1$ and $\langle d^\pi, r\rangle = (1-\gamma) J_r(\pi)$, where J_r is the unnormalized discounted return used in the main text. Conditional occupancies $d^{\pi|s,a}$ fix the initial state–action pair and satisfy

$$Q_r^\pi(s,a) = \frac{\langle d^{\pi|s,a}, r\rangle}{1-\gamma}.$$

Source risks use the global occupancy from μ_0, not this conditional occupancy. Write

$$\rho_j = \rho(d^{\pi_j}), \qquad \widetilde{Q}_i^\pi(s,a) = Q_i^\pi(s,a) - c\rho(d^\pi), \qquad c \geq 0.$$

The risk-adjusted quantity is a score, not a Bellman value function. Whenever a Lipschitz bound is invoked, ρ is finite and L-Lipschitz in ℓ_1 on a domain containing all occupancies being compared. Convexity is needed only for the source-hull bridge in Appendix C.1.

For tasks i, j, define

$$\Delta_\pi^{(i,j)}(s,a) = \mathbb{E}_\pi \left[\sum_{t\geq 0} \gamma^t |r_i(S_t,A_t) - r_j(S_t,A_t)| \,\middle|\, S_0 = s, A_0 = a \right].$$

B.1 Basic Value-Difference Lemmas

Lemma B.1 (Fixed-policy value difference)**.** *For any tasks i, j, policy π, and (s, a),*

$$|Q_i^\pi(s,a) - Q_j^\pi(s,a)| \le \Delta_\pi^{(i,j)}(s,a).$$

Proof. Under common dynamics, the two values integrate different rewards along the same trajectory distribution. Subtract their discounted-return expansions, apply the triangle inequality inside the expectation, and sum. Bounded rewards justify the interchange of expectation and the discounted sum. ■

Lemma B.2 (Optimal–optimal comparison)**.** *Let π_i^*, π_j^* be statewise optimal for tasks i, j. Then*

$$-\Delta_{\pi_j^*}^{(i,j)}(s,a) \le Q_i^*(s,a) - Q_j^*(s,a) \le \Delta_{\pi_i^*}^{(i,j)}(s,a),$$
$$|Q_i^*(s,a) - Q_j^*(s,a)| \le \max\{\Delta_{\pi_i^*}^{(i,j)}(s,a), \Delta_{\pi_j^*}^{(i,j)}(s,a)\}.$$

Proof. Statewise optimality gives $Q_j^* \ge Q_j^{\pi_i^*}$, so

$$Q_i^* - Q_j^* \le Q_i^{\pi_i^*} - Q_j^{\pi_i^*} \le \Delta_{\pi_i^*}^{(i,j)}.$$

Exchanging i and j gives $Q_j^* - Q_i^* \le \Delta_{\pi_j^*}^{(i,j)}$. These are bounds on opposite signs of the difference; the absolute-value bound uses their maximum. ■

Lemma B.3 (Transfer loss of a source policy)**.** *For any source policy π_j and target task i,*

$$0 \le Q_i^*(s,a) - Q_i^{\pi_j}(s,a) \le C_j^{(i)}(s,a), \quad C_j^{(i)} = \Delta_{\pi_i^*}^{(i,j)} + \Delta_{\pi_j}^{(i,j)}.$$

Proof. Insert the source-task values:

$$Q_i^* - Q_i^{\pi_j} = (Q_i^{\pi_i^*} - Q_j^{\pi_i^*}) + (Q_j^{\pi_i^*} - Q_j^{\pi_j}) + (Q_j^{\pi_j} - Q_i^{\pi_j}).$$

The first and third terms are at most the respective discrepancies by Lemma B.1. The middle term is nonpositive because π_j is optimal for task j: $Q_j^{\pi_j} = Q_j^* \ge Q_j^{\pi_i^*}$. The lower bound follows from target-task optimality. ■

Lemma B.4 (Occupancy diameter)**.** *For any policies π, π' evaluated with the same occupancy convention, $\|d^\pi - d^{\pi'}\|_1 \le 2$.*

Proof. Both occupancies are probability distributions, and the triangle inequality bounds their distance by the sum of their ℓ_1 norms. ■

Lemma B.5 (Lipschitz risk gap)**.** *For occupancies in the Lipschitz domain, $|\rho(d^\pi) - \rho(d^{\pi'})| \le L\|d^\pi - d^{\pi'}\|_1 \le 2L$.*

Proof. Apply the Lipschitz assumption and Lemma B.4. ■

B.2 Generalized Improvement with Source-Level Penalties

The following argument extends the Bellman monotonicity reasoning behind generalized policy improvement [1]; its constant shift is essential when source risks can be negative.

Definition B.6 (Bellman operators)**.** For the target reward r_T, define

$$(T_T^{\pi} Q)(s,a) = r_T(s,a) + \gamma \mathbb{E}_{s' \sim p(\cdot|s,a),\, b \sim \pi(\cdot|s')} Q(s',b).$$

It is monotone and a γ-contraction, with fixed point Q_T^{π}. The optimality operator replaces the inner policy expectation by $\max_b$.

Let $\rho_{\min} = \min_j \rho_j$ and

$$\alpha_j = c(\rho_j - \rho_{\min}) \geq 0, \qquad F(s,a) = \max_j \{Q_T^{\pi_j}(s,a) - \alpha_j\}.$$

Adding the common constant $c\rho_{\min}$ to all original TRAM scores does not change action selection, so TRAM is greedy with respect to F.

Lemma B.7 (Shifted source-score improvement)**.** *For exact source values and finite source risks, the greedy TRAM policy satisfies*

$$Q_T^{\pi_T}(s,a) \geq F(s,a) = \max_j \{Q_T^{\pi_j}(s,a) - c(\rho_j - \rho_{\min})\}.$$

No sign or convexity assumption on ρ is required.

Proof. For every source j, monotonicity and its Bellman equation imply

$$\begin{aligned} T_T^{\pi_T} F = T_T^* F &\geq T_T^{\pi_j} F \\ &\geq T_T^{\pi_j}(Q_T^{\pi_j} - \alpha_j) \\ &= Q_T^{\pi_j} - \gamma\alpha_j \geq Q_T^{\pi_j} - \alpha_j. \end{aligned}$$

Taking the maximum over j gives $T_T^{\pi_T} F \geq F$. Iterating this monotone contraction and taking the limit proves the claim. The identity $T_T^{\pi_j}(Q - \alpha) = T_T^{\pi_j} Q - \gamma\alpha$ is why a nonnegative shifted penalty is used. ■

B.3 Localized Transfer Bounds

Lemma B.8 (Risk-adjusted comparison with a source)**.** *For any source j,*

$$|\widetilde{Q}_T^{\pi^*} - \widetilde{Q}_T^{\pi_j}| \leq C_j^{(T)} + 2Lc,$$

where π^* is target-optimal and all quantities are evaluated at the same (s,a).

Proof. By Lemma B.3, the nonnegative reward-value gap is at most $C_j^{(T)}$. The absolute difference of the two risk penalties is at most $2Lc$ by Lemma B.5. Apply the triangle inequality. ■

Theorem B.9 (Localized bounds for the deployed policy)**.** *Under the analysis assumptions, set $C_j = C_j^{(T)}$. Then*

$$Q_T^* - Q_T^{\pi_T} \leq \min_j \{C_j + c(\rho_j - \rho_{\min})\} \leq \min_j C_j + 2Lc, \tag{B.1}$$

$$\widetilde{Q}_T^{\pi^*} - \widetilde{Q}_T^{\pi_T} \leq \min_j C_j + 4Lc. \tag{B.2}$$

Proof. Lemma B.7 gives, for every j,

$$Q_T^* - Q_T^{\pi_T} \leq Q_T^* - Q_T^{\pi_j} + c(\rho_j - \rho_{\min}) \leq C_j + c(\rho_j - \rho_{\min}).$$

Take the minimum and use $\rho_j - \rho_{\min} \le 2L$. For the second inequality, use the actual deployed-policy risk:

$$\widetilde{Q}_T^{\pi^*} - \widetilde{Q}_T^{\pi_T} = Q_T^* - Q_T^{\pi_T} + c\{\rho(d^{\pi_T}) - \rho(d^{\pi^*})\}.$$

The last term is at most $2Lc$. This step does not identify the deployed risk with any cached source risk. ■

Scope of the comparison. The comparator is a risk-neutral target optimum. These bounds do not establish optimality for the risk-regularized control problem or satisfaction of a hard risk threshold. They apply to the deployed policy only when its actual occupancy belongs to the stated Lipschitz domain. The convex source-hull bridge is a separate diagnostic, not a substitute for this domain condition.

B.4 Main-Text Corollaries and Source-Library Design

Proposition B.10 (Main-text localized guarantee)**.** *Theorem 4.1 follows with* $C_j = \Delta_{\pi^*}^{(T,j)} + \Delta_{\pi_j}^{(T,j)}$.

Proof. Apply Eq. B.2. Both trajectory discrepancies are required to compare a target-optimal policy with a source policy. ■

Proposition B.11 (Reward-only corollary)**.** *Corollary 4.2 has penalty term* $2Lc$.

Proof. Use Eq. B.1 directly. Removing the risk-adjusted terms from a looser bound would give an unnecessary additional penalty. ■

Proposition B.12 (Successor-feature corollary)**.** *If* $r_k(s,a,s') = \phi(s,a,s')^\top w_k$ *with* $\|\phi(s,a,s')\|_2 \le \phi_{\max}$*, then*

$$C_j(s,a) \le \frac{2\phi_{\max}}{1-\gamma}\|w_T - w_j\|_2.$$

Proof. The difference of the expected one-step rewards is bounded by $\phi_{\max}\|w_T - w_j\|_2$ using Cauchy–Schwarz and the common transition kernel. Each of the two trajectory discrepancies is therefore at most this quantity times $(1-\gamma)^{-1}$. ■

Proposition B.13 (Conditional source-hull dominance)**.** *Let* $S_\lambda \subseteq \mathcal{D}$ *be occupancy libraries and assume* $\mathrm{co}(S_\lambda) \subseteq \mathrm{co}(S_0)$ *for every* $\lambda \ge 0$*. Let every hull intersect* $\{d : \langle d, r_T\rangle \ge V\}$*. For any nonempty class* $\mathcal{R}_L$ *of finite L-Lipschitz risks on* $\mathcal{D}$*, define the nonnegative regret*

$$\mathcal{E}_{\rho,V}(S) = \inf_{\substack{d\in\mathrm{co}(S)\\ \langle d,r_T\rangle\ge V}} \rho(d) - \inf_{\substack{d\in\mathcal{D}\\ \langle d,r_T\rangle\ge V}} \rho(d).$$

Then

$$\sup_{\rho\in\mathcal{R}_L} \mathcal{E}_{\rho,V}(S_0) = \inf_{\lambda\ge 0} \sup_{\rho\in\mathcal{R}_L} \mathcal{E}_{\rho,V}(S_\lambda).$$

Proof. Hull inclusion enlarges the feasible set of the first infimum for S_0 and can only decrease it. The second infimum is common to all libraries, so $\mathcal{E}_{\rho,V}(S_0) \le \mathcal{E}_{\rho,V}(S_\lambda)$. Taking suprema preserves this inequality; inclusion of $\lambda = 0$ gives equality after the infimum over λ. Feasibility and the Lipschitz diameter bound make the regrets finite. ■

This is a conditional set-inclusion result. Risk-neutral training alone does not imply the required inclusion, and the result concerns best achievable risk in a source hull, not the performance of the greedy TRAM rule.

B.5 Approximate Source Scores

Proposition B.14 (Uniform score-error bound)**.** *Suppose $\widehat{\pi}$ is greedy with respect to $\widehat{M}(s,a) = \max_j\{\widehat{Q}_j(s,a) - c\widehat{\rho}_j\}$ and*

$$\sup_{j,s,a} |(\widehat{Q}_j(s,a) - c\widehat{\rho}_j) - (Q_T^{\pi_j}(s,a) - c\rho_j)| \le \eta.$$

For a fixed nonempty source library,

$$Q_T^{\widehat{\pi}} \ge F - \frac{2\gamma\eta}{1-\gamma}.$$

Consequently, each bound in Theorem B.9 holds for $\widehat{\pi}$ after adding $2\gamma\eta/(1-\gamma)$ to its right-hand side, provided the risk-domain assumption includes $d^{\widehat{\pi}}$.

Proof. Taking a maximum cannot increase the uniform error. Since F differs from the true max-score by a constant, greedy selection under $\widehat{M}$ satisfies $F(s,\widehat{\pi}(s)) \ge \max_a F(s,a) - 2\eta$. Thus

$$T_T^{\widehat{\pi}} F \ge T_T^* F - 2\gamma\eta \ge F - 2\gamma\eta.$$

Set $h = 2\gamma\eta/(1-\gamma)$. Constant-shift equivariance gives $T_T^{\widehat{\pi}}(F-h) \ge F - 2\gamma\eta - \gamma h = F - h$. Monotone contraction yields $Q_T^{\widehat{\pi}} \ge F - h$. The transfer bounds then follow exactly as in Theorem B.9. ■

The bound is for action values with the initial action fixed. If $\|\widehat{Q}_j - Q_T^{\pi_j}\|_\infty \le \epsilon_Q$ and $|\widehat{\rho}_j - \rho_j| \le \epsilon_\rho$, one may take $\eta = \epsilon_Q + c\epsilon_\rho$. Screening errors require a separate guarantee that the retained library is appropriate; this proposition only analyzes scoring within a fixed retained library.

C Stitching Mismatch and Approximate-Dynamics Extensions

C.1 Stitching Mismatch and the Source-Hull Bridge

The action rule in Eq. 4.4 can select different sources across states, so d^{π_T} need not equal any source occupancy or their selected mixture. Fix any weights $\omega_j \ge 0$ with $\sum_j \omega_j = 1$ and define

$$\bar{d}_T = \sum_j \omega_j d^{\pi_j}, \qquad \epsilon_{\text{stitch}} = \|d^{\pi_T} - \bar{d}_T\|_1.$$

One possible choice is the normalized discounted frequency of each selected source along deployed trajectories, with a fixed tie-breaking rule. The proposition also allows other mixture weights; it does not identify source-selection frequencies with mixture occupancy weights.

Proposition C.1 (Source-hull realized-risk bridge)**.** *If ρ is convex and L-Lipschitz on a convex domain containing the source occupancies and d^{π_T}, then*

$$\rho(d^{\pi_T}) \le \sum_j \omega_j \rho(d^{\pi_j}) + L\epsilon_{\text{stitch}}. \tag{C.1}$$

Proof. Lipschitzness gives $\rho(d^{\pi_T}) \le \rho(\bar{d}_T) + L\|d^{\pi_T} - \bar{d}_T\|_1$. Jensen's inequality gives $\rho(\bar{d}_T) \le \sum_j \omega_j \rho(d^{\pi_j})$. Combine the two inequalities. ■

For fixed weights, the mismatch term is zero exactly when $d^{\pi_T} = \bar{d}_T$. Membership in the source hull alone is insufficient to make a particular frequency-weighted mismatch zero. Minimizing the distance over all mixture weights gives the distance to the hull; a large distance for one chosen mixture does not prove a large distance to the entire hull.

The bridge does not apply to every occupancy functional. In particular, the variance and entropic examples in Appendix A.3 are generally concave in occupancy. Without convexity, Lipschitzness still gives $\rho(d^{\pi_T}) \le \rho(\bar{d}_T) + L\epsilon_{\text{stitch}}$, but the source-risk average cannot replace $\rho(\bar{d}_T)$.

Estimation and certification. Finite-rollout mismatch estimates are diagnostics unless their errors are bounded. For any chosen weights, if occupancy estimates satisfy $\|\widehat{d}_T - d^{\pi_T}\|_1 \le \zeta_T$ and $\|\widehat{d}_j - d^{\pi_j}\|_1 \le \zeta_j$, the triangle inequality gives

$$\epsilon_{\text{stitch}} \le \left\| \widehat{d}_T - \sum_j \omega_j \widehat{d}_j \right\|_1 + \zeta_T + \sum_j \omega_j \zeta_j.$$

If source-risk estimates also satisfy $|\widehat{\rho}_j - \rho(d^{\pi_j})| \le \xi_j$, a valid upper bound is

$$\rho(d^{\pi_T}) \le \sum_j \omega_j(\widehat{\rho}_j + \xi_j) + L\left(\left\| \widehat{d}_T - \sum_j \omega_j \widehat{d}_j \right\|_1 + \zeta_T + \sum_j \omega_j \zeta_j \right).$$

When the error bounds hold jointly with high probability, so does this inequality. Continuous-state applications need an appropriate occupancy representation and associated approximation bounds; empirical rollouts alone do not provide an exact certificate.

C.2 Approximate Dynamics and Source Evaluation

The common-dynamics analysis evaluates every source in the target MDP, where each source is assumed optimal for its own reward r_j. If a different model is also used to evaluate the target reward, its error affects greedy action selection as follows.

Theorem C.2 (Model-error extension)**.** *Suppose source j is evaluated exactly in model $(\widehat{p}_j, \widehat{r}_{T,j})$, yielding $\widehat{Q}_j = Q^{\pi_j}_{\widehat{p}_j, \widehat{r}_{T,j}}$, while the target uses (p_T, r_T). Assume uniformly over j, s, a that*

$$\|\widehat{p}_j(\cdot \mid s,a) - p_T(\cdot \mid s,a)\|_{\text{TV}} \le \delta_p, \quad |\widehat{r}_{T,j}(s,a) - r_T(s,a)| \le \delta_r, \quad \|\widehat{Q}_j\|_\infty \le B.$$

If source-risk errors are at most ϵ_ρ, define

$$\eta_{\text{model}} = \frac{\delta_r + 2\gamma B \delta_p}{1-\gamma} + c\epsilon_\rho.$$

Under the remaining risk-domain assumptions of Theorem 4.1, the policy $\widehat{\pi}$ greedy in these model-based scores obeys

$$Q^*_T - Q^{\widehat{\pi}}_T \le \min_j C_j + 2Lc + \frac{2\gamma}{1-\gamma}\eta_{\text{model}}, \tag{C.2}$$

$$\widetilde{Q}^{\pi^*}_T - \widetilde{Q}^{\widehat{\pi}}_T \le \min_j C_j + 4Lc + \frac{2\gamma}{1-\gamma}\eta_{\text{model}}. \tag{C.3}$$

All C_j, source occupancies, and true risks in these comparisons use the target dynamics.

Proof. Let $\widehat{V}_j(s) = \sum_a \pi_j(a \mid s)\widehat{Q}_j(s,a)$, so $\|\widehat{V}_j\|_\infty \le B$. Total variation yields

$$|\mathbb{E}_{p_T}\widehat{V}_j - \mathbb{E}_{\widehat{p}_j}\widehat{V}_j| \le 2B\delta_p.$$

Since $\widehat{Q}_j$ is the fixed point of the model Bellman operator, its residual under the target operator for π_j is at most $\delta_r + 2\gamma B\delta_p$. Contraction therefore gives

$$\|Q^{\pi_j}_T - \widehat{Q}_j\|_\infty \le \frac{\delta_r + 2\gamma B\delta_p}{1-\gamma}.$$

Adding the source-risk error gives a uniform score error of at most η_{model}. Proposition B.14 then proves both bounds. ■

If rewards and risks are evaluated exactly and only the transition model differs, the added action-value term is $4\gamma^2 B\delta_p/(1-\gamma)^2$. Both the value-evaluation error and the effect of choosing actions with erroneous scores must be accounted for. Additional critic approximation error can be added to η_{model}. For transition-dependent rewards, δ_r accounts for the change in their expected one-step value; it is zero only when these expectations are evaluated correctly.

D Continuous-Control and Safety-Gymnasium Experimental Details

This appendix records implementation details for the continuous-control results in Fig. 4 (Appendices D.1–D.2) and reports additional Safety-Gymnasium experiments: a constraint-aware baseline and deployment costs, sensitivity to c, an alternative risk functional, and the cost of occupancy estimation (Appendices D.3–D.7).

D.1 Reacher Implementation Details

Environment. REACHER (MuJoCo [43]) uses a planar 2-DOF arm with continuous joint-angle and velocity states. Following successor-feature transfer protocols [1, 12, 50], torques are discretized to nine actions corresponding to minimum, zero, or maximum torque on each joint. Rewards combine negative distance-to-goal with a small control penalty.

Source library. Four risk-neutral Successor-Feature DQNs (SFDQNs) are trained on different source goals. Each source learns successor features ψ^{π_j} so that the target value can be evaluated by the dot product

$$Q_T^{\pi_j}(s,a) = \psi^{\pi_j}(s,a)^\top w_T.$$

At deployment, the source parameters are frozen. TRAM only evaluates target values and the specified risk term for the fixed source library.

Deployment-time risk. The target task changes the goal and introduces a barrier region after source training. We use a barrier-exposure risk computed from target rollouts or cached occupancy estimates, with tradeoff weight $c = 5$. Thus the deployment-time computation is source scoring, not retraining or re-solving the target MDP. A failure is recorded when the arm's tip enters the barrier region, and the test is repeated 100 times.

Runtime. For n sources, $|\mathcal{A}|$ discrete actions, and feature dimension d, action selection costs $\mathcal{O}(n|\mathcal{A}|d)$ per state when successor-feature dot products are used. The risk term is a cached source-level lookup, so the online overhead over risk-neutral SF transfer is small.

D.2 Safety-Gymnasium Benchmark Details

We use Safety-Gymnasium `SafetyPointGoal0-v0` [19]. The target goal is fixed at $(0.7, 0.7)$ and the deployment risk is a circular barrier centered at $(0.245, 0.245)$ with radius 0.18. The observation representation is 16-dimensional. We evaluate 1000 episodes with horizon 1000 and discount $\gamma = 0.99$.

All methods use the same fixed source library and perform no deployment-time parameter updates. Risk-neutral GPI scores target reward only. RaSF uses a variance-style surrogate. The SafeDSR-style proxy is frozen and therefore cannot fully rescore behavior under the newly requested barrier risk. The SFT-CoP-style baseline enters a collapsed-task regime in this benchmark, so it is shown for barrier-hit probability in the main figure but omitted from the normalized-CVaR comparison.

The normalized-CVaR exposure in Fig. 4(b) rescales exposure relative to the risk-neutral GPI baseline so that lower values indicate safer behavior under the requested deployment-time barrier risk. This normalization is used only for visual comparability across methods.

D.3 Additional Safety-Gymnasium Experiments: Protocol

The experiments in Appendices D.4–D.7 use a separate protocol from Fig. 4(b), with a different held-out goal, so their numbers should not be compared with that figure directly.

Task. `SafetyPointGoal0-v0` with held-out target goal $(0.5, 0.5)$ and goal tolerance 0.02. The deployment barrier is a circle centered at $(0.245, 0.245)$ with radius 0.18 unless stated otherwise. Each evaluation uses 300 episodes with horizon $1{,}000$ and $\gamma = 0.99$. Success means reaching the goal; barrier-hit probability is the fraction of episodes that enter the barrier. All methods share one evaluator (goal, barrier geometry, horizon, success and hit detectors).

Source library and risk. The library contains 39 frozen SF-DQN source policies trained risk-neutrally on goals in the cardinal and diagonal directions; none is trained on the target goal. For each source we cache a policy-level risk, its barrier-hit probability under the deployment barrier, estimated

offline from rollouts of the frozen policy. TRAM uses the dynamic per-state rule of Algorithm 1: at every state it scores each source by its successor-feature value minus c times the cached risk and executes the best action.

Scale of c. The SF-DQN action values in this environment range from about 10^2 to 10^5, while the cached risk is a probability. Penalty weights that change the selected source are therefore of order 10^4 to 10^6; c is meaningful only relative to the value scale of the library.

Hardware. These experiments were run on NVIDIA GH200 Grace Hopper nodes, one node per job, on the node's Grace CPU (Appendix F). Wall-clock times exclude scheduler queueing.

D.4 Constraint-Aware Baseline and Deployment Cost

Constrained PPO. As a constraint-aware reference, we train a PPO policy directly on the deployment specification (goal $(0.5, 0.5)$, barrier radius 0.18). It uses a separate risk critic and an adaptive Lagrange multiplier that targets a barrier-hit probability of 0.10; the target is annealed from 0.30 to 0.10 over the first $400{,}000$ steps, and training runs for $750{,}000$ environment steps. Unlike TRAM, PPO observes the deployment reward and barrier during training, so it is an advantaged direct-training reference rather than a zero-update competitor. Table 2 compares the two under the common evaluator; returns are omitted because PPO is trained on a shaped deployment reward that TRAM's sources never see.

Table 2: Constraint-aware baseline. Safety-Gymnasium, held-out goal $(0.5, 0.5)$, barrier radius 0.18, 300 evaluation episodes, common evaluator. TRAM rows are means over three evaluation seeds (Table 3); PPO is one training run. Best value in each column in bold.

Method	Trained on deployment task	Success ↑	Barrier-hit prob. ↓
Constrained PPO	yes (750,000 steps)	**0.997**	0.240
TRAM, $c = 6.4 \times 10^4$	no	0.630	0.360
TRAM, $c = 1.28 \times 10^5$	no	0.366	0.290
TRAM, $c = 10^6$	no	0.102	**0.086**

PPO learns the task well (299 of 300 episodes reach the goal) but does not meet its 10% barrier-hit target. TRAM reaches a lower barrier-hit probability than PPO at $c = 10^6$, without any training on the deployment task, but at a large cost in task success; intermediate values of c trade the two off (Appendix D.5). The two methods therefore occupy different operating points rather than one dominating the other.

Deployment cost. Constrained PPO pays a full training run for each new specification: $750{,}000$ environment steps, about 40 minutes at its measured throughput of about 310 steps per second (a throughput-based estimate). TRAM performs no training at deployment. Its costs are:

- *Source library* (one-time, shared across specifications): training the frozen sources.
- *Risk estimation* (per specification in our implementation): rollouts of each frozen source under the new specification, about 0.94 s per $1{,}000$-step rollout per source on our hardware (Appendix D.7), parallel across sources.
- *Online scoring*: 2.646 ± 0.009 s per $1{,}000$-step episode over a 20-source library (2.65 ms per action; mean and 95% CI over 30 warm episodes), and 3.49 s for a cold first episode that includes loading the policies.

Because the cached risk in these experiments is a barrier-hit probability, risk estimation is repeated when the barrier changes. For risks that are functionals of cached state occupancies, such as barrier exposure (R1), it reduces to evaluating the functional on the cached occupancy.

Source switching. Logging the selected source at each step confirms that TRAM composes several sources within a rollout. In the dynamic log-barrier runs of Appendix D.6 at low c, the most frequently selected source accounts for about 81% of the $300{,}000$ evaluation steps, and two others account for about 14% and 5%.

D.5 Sensitivity to the Tradeoff Weight c

Choosing c. Problem 3.2 is the Lagrangian relaxation of 3.1: the risk attained by its optimum is nonincreasing in c, so each c selects an operating point on the reward–risk tradeoff, while the screening step of Algorithm 1 imposes a tolerance δ directly. Because c multiplies a risk that is compared with action values, its useful range depends on the value scale of the library (Appendix D.3). We use $c = 5$ in Reacher (Appendix D.1) and report the Safety-Gymnasium values below.

Main configuration. Table 3 sweeps c at barrier radius 0.18 over three independent evaluation seeds. Increasing c moves the deployed policy from higher return, success, and risk toward lower return, success, and risk, and the three seeds agree at every point. With $n = 3$, the spread is an exploratory measure of variability rather than a formal confidence interval.

Table 3: Sensitivity to c at the main configuration. Barrier radius 0.18; mean $\pm$ standard deviation over three evaluation seeds, 300 episodes each.

c	Return ↑	Success ↑	Barrier-hit prob. ↓
6.4×10^4	0.518 ± 0.028	0.630 ± 0.030	0.360 ± 0.020
1.28×10^5	0.268 ± 0.062	0.366 ± 0.031	0.290 ± 0.018
10^6	0.244 ± 0.023	0.102 ± 0.002	0.086 ± 0.010

Barrier radius. Table 4 repeats the sweep for five barrier radii with the same frozen library; only the cached risks are re-estimated for each barrier. Each cell is a single 300-episode evaluation. At every radius, the largest c gives the lowest barrier-hit probability and the lowest success. The response is not monotone at intermediate c (for example, $c = 2.56 \times 10^5$ at the larger radii), which reflects finite-sample evaluation and switching among sources whose risks are close; c should therefore be read as a tradeoff parameter rather than a guarantee of strictly decreasing risk.

Table 4: Sensitivity to c across barrier radii. Barrier-hit probability (left, lower is better) and success (right, higher is better) for one fixed source library; one 300-episode evaluation per cell.

	Barrier-hit probability ↓					Success ↑				
c / radius	0.03	0.09	0.18	0.24	0.30	0.03	0.09	0.18	0.24	0.30
3.2×10^4	0.247	0.303	0.470	0.590	0.867	0.963	0.740	0.757	0.783	0.893
6.4×10^4	0.117	0.173	0.340	0.457	0.563	0.780	0.607	0.617	0.633	0.647
1.28×10^5	0.073	0.177	0.270	0.403	0.450	0.597	0.363	0.330	0.437	0.433
2.56×10^5	0.067	0.203	0.327	0.423	0.553	0.350	0.307	0.273	0.277	0.390
10^6	0.040	0.057	0.117	0.153	0.190	0.227	0.213	0.120	0.120	0.087

D.6 Alternative Risk Functional: Log-Barrier

To test a different deployment risk on the same library, we replace the cached hit probability p_j by the log-barrier $-\log(1 - p_j)$ (the barrier of (R1) with $\delta = 1$, applied to the policy-level hit probability) at barrier radius 0.09.

Single-source selection. When one source is selected for the whole episode by the cached score (episode return minus c times the risk), the relevant scale of c is that of the returns, and a sweep over $c \in [0, 2]$ traces a clear frontier with distinct selected sources (Table 5, left).

Dynamic per-state selection. With the per-state rule of Algorithm 1, the value scale is that of the SF-DQN action values, and the transition occurs at $c \approx 10^4$. We repeated the three most informative values with three independent seeds, re-estimating the cached risks for each seed (Table 5, right).

Per-state composition reaches operating points that the single-source selections in this sweep do not: at $c = 1.6 \times 10^4$ it attains success 0.72 at hit probability 0.22, whereas the closest single-source selection attains success 0.46 at hit probability 0.29.

D.7 Cost of Occupancy Estimation

We measured the offline cost of estimating one frozen source's discounted occupancy in this continuous, non-enumerable setting. The occupancy is a fixed 80×80 histogram over the source's

Table 5: Log-barrier risk at barrier radius 0.09. Left: single-source selection, one 300-episode evaluation per row. Right: dynamic per-state selection, mean ± 95% confidence half-width over three seeds.

Single-source selection				Dynamic selection			
c	Return	Success	Hit prob.	c	Return	Success	Hit prob.
0	2.278	1.000	0.907	10^4	0.724 ± 0.025	0.879 ± 0.028	0.411 ± 0.078
1.0	1.202	0.627	0.497	1.2×10^4	0.667 ± 0.033	0.814 ± 0.038	0.289 ± 0.029
1.5	0.762	0.460	0.293	1.6×10^4	0.550 ± 0.038	0.720 ± 0.019	0.216 ± 0.006
2.0	0.280	0.130	0.033				

two successor features (goal progress and cost), accumulated with weights $(1-\gamma)\gamma^t$ over rollouts of horizon $1{,}000$. We ran five independent evaluation seeds of 200 rollouts each and compare the estimates after 50 and 100 rollouts with the 200-rollout estimate of the same seed (Table 6). Runtime is approximately linear in the number of rollouts, about 0.94 s per rollout. The 200-rollout estimate is a finite-sample reference, not ground truth, and the study covers one source policy.

Table 6: Occupancy-estimation cost for one source. Mean and 95% Student-t interval over five evaluation seeds. TV is the total-variation distance to the 200-rollout estimate of the same seed.

Rollouts	Wall-clock time (s)	TV to 200-rollout estimate
50	47.0 [46.7, 47.3]	0.0128 [0.0061, 0.0195]
100	94.0 [93.3, 94.6]	0.0021 [0, 0.0045]†
200	188.0 [186.7, 189.4]	reference

†Lower endpoint truncated at zero, since TV is nonnegative.

E LLM Alignment Experiment

E.1 Setup

We evaluate the deployment-time alignment task on Berkeley Nectar [53]. The source library contains two pretrained 7B models, Zephyr-Qwen-2-7B and Dolphin-Qwen-2-7B. The deployment target reward is produced by a reward model, while deployment risk is a KL-style behavioral penalty to a safe reference model. TRAM composes fixed source generations at inference time and does not fine-tune or update model parameters.

Instantiation. Generation is treated as a sequential decision process: the state is the prompt together with the text generated so far, and the action is the next token. The two fixed models are the source policies. Their values under the target reward model are estimated without parameter updates using the transfer-decoding value estimate of [6], and TRAM applies its selection rule at the token level through this transfer decoding. The deployment risk is the token-level KL divergence between a source's output distribution and that of the safe reference model, so continuations that drift from the reference are penalized. The rule of Eq. 4.4 then selects the continuation that maximizes estimated target value minus c times this KL penalty. As in other inference-time alignment methods [28, 20], this requires one forward pass per source and step; the cost is linear in the number of sources and parallel across them.

E.2 Metrics and Interpretation

The main quantitative summary is Fig. 5; we do not repeat the same values in an appendix table. We report both the scalar target-reward score and GPT-4 pairwise preference against TRAM. The second metric is included to detect reward-model overoptimization: a method can score high under the target reward while producing responses that are less preferred by a stronger evaluator.

Risk-free transfer maximizes the reward-model score but is not preferred over TRAM by GPT-4 in most pairwise comparisons. TRAM sacrifices some target-reward score to remain closer to the safe reference behavior. This supports the deployment-time claim: a new behavioral risk can be introduced after source models are trained and used to guide inference without parameter updates.

F Reproducibility and Implementation Details

This section records the available implementation details. The exact run configurations, all random seeds, complete model/checkpoint identifiers, LLM decoding and judging settings, and per-run compute times are not fully specified here. The descriptions support reimplementation of the scoring rule but do not constitute a complete recipe for reproducing every reported result.

F.1 Core Software Stack

Gridworld. The tabular experiments can be implemented with Python, NumPy, and SciPy. Source policies are trained by value iteration or Q-learning. Discounted occupancies can be computed either by solving the discounted occupancy linear system or by long-rollout discounted counts. Barrier, volatility, and KL-style risk functionals are then evaluated directly on these occupancies.

Reacher and continuous control. Reacher uses MuJoCo [43] with a Gymnasium-style environment interface. We discretize torques into nine actions, as in standard successor-feature transfer protocols [1, 12, 50]. Source policies are SFDQN agents trained risk-neutrally on different goals. A practical implementation uses Python 3.11, PyTorch 2.3 or newer, Gymnasium 0.29 or newer, MuJoCo 3.1 or newer, NumPy, SciPy, Pandas, and Matplotlib. During deployment, the target value is computed by $Q_T^{\pi_j}(s,a) = \psi^{\pi_j}(s,a)^\top w_T$, and the source-level risk term is a cached lookup.

Safety-Gymnasium. The SafetyPointGoal0-v0 experiment uses Safety-Gymnasium [19]. The implementation needs the Safety-Gymnasium package, Gymnasium, MuJoCo, PyTorch, NumPy, and Pandas. The target goal, circular barrier center/radius, horizon, discount, and number of episodes are specified in Appendix D.2. All methods use the same frozen source library, and no method updates source parameters at deployment.

LLM alignment. LLM experiments use Hugging Face Transformers, Accelerate, PyTorch with CUDA, and standard tokenizer/model loading utilities. The source library contains fixed 7B instruction-tuned models, the target reward is produced by a reward model, and the risk term is a KL-style distance to a safe reference behavior. Evaluation uses the same prompts and decoding budget across methods.

F.2 Computational Resources

Gridworld and Reacher experiments can run on CPU; our Reacher runs used an Intel Core i7-class CPU with 16 GB RAM. Safety-Gymnasium experiments were run on NVIDIA GH200 Grace Hopper nodes, each with a 72-core NVIDIA Grace CPU and a Hopper GPU with 96 GB of HBM3, using one node per job. Because the source networks are small SF-DQNs, these jobs run on the Grace CPU without GPU acceleration; the wall-clock times in Appendix D.4 and Appendix D.7 are measured on this hardware and exclude scheduler queueing. LLM evaluations use NVIDIA RTX A6000-class GPUs or comparable accelerators with enough memory for 7B model inference.

G Occupancy Storage, Risk Computation, and Runtime

TRAM stores one global discounted occupancy estimate per source policy. In tabular domains this is computed exactly by solving the occupancy linear system or the equivalent Q-LP. In continuous-control domains it is estimated once by discounted rollout counts or a learned occupancy model and cached before deployment. During deployment, the online risk score for a source is therefore a lookup of $\rho(d^{\pi_j})$ rather than a new occupancy computation.

The conditional occupancy $d^{\pi_j|s,a}$ appears only through value evaluation: it is the trajectory distribution underlying $Q_T^{\pi_j}(s,a)$. In successor-feature experiments, the value is computed by the dot product $\psi^{\pi_j}(s,a)^\top w_T$. With n sources, $|\mathcal{A}|$ discrete actions, and feature dimension d, this gives $\mathcal{O}(n|\mathcal{A}|d)$ online scoring per state. Without successor features, the same source-scored rule can be used with learned critics or rollout-based value estimators, but runtime then depends on that evaluator.

For experiments that report realized safety, we evaluate the deployed stitched policy by rollout and compute event-level risk metrics such as barrier-hit probability and exposure CVaR. The same rollouts can estimate the stitching mismatch in Eq. C.1, but certification additionally requires valid risk and occupancy estimation-error bounds as described in Appendix C.1.

H Impact Statement

This paper proposes TRAM, a test-time risk adaptation framework that composes pretrained (risk-neutral) policies to satisfy deployment-time safety specifications without retraining. The primary intended impact is to improve the safety and reliability of reinforcement learning systems in settings where acceptable risk changes after training, such as robotics, autonomous systems, and resource allocation under updated operational constraints. By enabling operators to express safety requirements through occupancy-based risk functionals (e.g., hazard avoidance, behavioral constraints, or local volatility), TRAM can reduce harmful behavior when conditions evolve. Potential negative impacts include misuse: the same ability to rapidly adapt behavior at deployment could be applied to optimize objectives that are misaligned with human values or organizational policies, particularly if the risk specification is chosen adversarially or if monitoring is absent. We recommend pairing TRAM with conservative safety envelopes, explicit constraint checks when available, and post-deployment monitoring and audits.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and Introduction state the zero-update deployment setting, define TRAM before using it, and explicitly describe the source-scored surrogate scope and stitching-mismatch limitation; the claims are supported by the theory in Section 4 and the experiments in Section 5.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: The paper states that TRAM is a source-scored surrogate rather than an exact optimizer of the stitched policy occupancy, discusses assumptions on source libraries and transition mismatch, and provides additional limitation analysis in Appendix C and Appendix C.2.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

   Justification: The theoretical statements include their assumptions in Section 4; complete proofs and extensions are provided in Appendix A, Appendix B, and Appendix C.2.

   Guidelines:

   - The answer [N/A] means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [No]

   Justification: The scoring rule and several environment settings are provided in Algorithm 1 and Appendices D–F, but the full run configurations and model identifiers are not supplied. These omissions limit exact reproduction of the reported results.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).

(d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [No]

   Justification: A public code/data release is not included with this paper. The paper instead provides reimplementation-oriented details, public benchmark names, software libraries, environment settings, and the complete deployment-time scoring rule in Appendix F.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [No]

   Justification: Task geometry, source counts, selected horizon/discount settings, and evaluation metrics are reported, but not all training hyperparameters, random seeds, prompt-selection rules, and decoding settings are specified.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [No]

   Justification: The main figures report aggregate performance metrics without error bars across independent seeds. The additional Safety-Gymnasium analyses in Appendix D.4–D.7 report variability across three or five evaluation seeds (standard deviations or 95% intervals) and 95% intervals for timing, but statistical reporting is not exhaustive across all experiments.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.
   - It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
   - For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
   - If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [No]

   Justification: Appendix F describes the software and hardware (including the NVIDIA GH200 nodes used for Safety-Gymnasium), and Appendix D.4 and Appendix D.7 report measured times for TRAM inference, occupancy estimation, and the retrained baseline. Total compute for every experiment is not reported.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics https://neurips.cc/public/EthicsGuidelines?

   Answer: [Yes]

   Justification: The work uses standard simulated benchmarks, public/pretrained models, and automated evaluation, and is intended to support safer deployment-time adaptation. No component of the research intentionally violates the NeurIPS Code of Ethics.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: The Impact Statement discusses intended positive impacts for safety and reliability, possible misuse through adversarial or misaligned risk specifications, and mitigation through conservative safety envelopes, explicit checks, monitoring, and audits.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
    - The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
    - If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: The submission does not release a new high-risk model, scraped dataset, or dual-use asset requiring controlled access. Existing pretrained models are used only as fixed source policies in the LLM alignment experiment.

Guidelines:

- The answer [N/A] means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [No]

    Justification: The paper cites the existing benchmarks, algorithms, datasets, and pretrained-model families used in the experiments, but it does not explicitly enumerate the license and terms of use for every software package, dataset, and model asset.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
    - If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [N/A]

    Justification: The paper does not introduce or release a new dataset, benchmark, model checkpoint, or software asset as a contribution of this submission.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: The experiments do not involve crowdsourcing or human-subject data collection. Pairwise preference evaluation in the LLM experiment is performed by an automated GPT-4 evaluator rather than recruited human participants.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: The research does not involve crowdsourcing or human-subject experiments, so IRB or equivalent approval is not applicable.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [Yes]

    Justification: The paper includes an LLM alignment experiment in Section 5 and Appendix E, describing the fixed 7B source models, target reward model, KL-style risk to a safe reference behavior, and GPT-4 pairwise preference evaluation. We also used in coding and methodology.

    Guidelines:

    - The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.