Surrogate-Augmented Deception in Reinforcement Learning (SAD-RL)

Joe Shymanski
The University of Tulsa
Tulsa, OK, United States
joe-shymanski@utulsa.edu

Scott Nivison
Air Force Research Laboratory
Eglin AFB, FL, United States
scott.nivison.1@us.af.mil

Sandip Sen
The University of Tulsa
Tulsa, OK, United States
sandip-sen@utulsa.edu

ABSTRACT

Reinforcement learning (RL) agents in adversarial environments risk being modeled and exploited by opponents that infer their goals or policies. We introduce *Surrogate-Augmented Deception in Reinforcement Learning* (SAD-RL), a framework that trains agents to resist such modeling by embedding a surrogate predictor into the learning loop and penalizing its accuracy. Rather than emphasizing mere unpredictability, SAD-RL promotes *strategic opacity*—learning behaviors that remain effective while defying opponent inference. We evaluate SAD-RL in two representative domains: a discrete *Adversarial Grid World* (AGW) and a continuous *Sharks and Minnows* (SaM) pursuit-evasion task. Across both settings, SAD-RL agents maintain high task performance while exhibiting measurable deception against surrogate models, achieving a better trade-off between effectiveness and opacity than conventional RL agents. We further analyze the trade-off between goal achievement and opacity, identifying distinct modes of balanced and over-deceptive behavior. Together, these results establish SAD-RL as a general and domain-agnostic approach for inducing emergent deception in reinforcement learning.

KEYWORDS

Reinforcement Learning; Multiagent Systems; Adversarial Learning; Deception; Opponent Modeling; Surrogate Models

ACM Reference Format:
Joe Shymanski, Scott Nivison, and Sandip Sen. 2026. Surrogate-Augmented Deception in Reinforcement Learning (SAD-RL). In *Proc. of the 25th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2026), Paphos, Cyprus, May 25 – 29, 2026*, IFAAMAS, 9 pages. https://doi.org/10.65109/KOTB1262

1 INTRODUCTION

Agents competing in adversarial or head-to-head settings may learn highly effective strategies. However, if those strategies become predictable, they can be exploited by opponents that infer their underlying decision rules. For instance, in cybersecurity, a defense agent that always responds to a specific threat in the same manner can be circumvented by an adaptive attacker. Initial success may give way to failure as adversaries learn and counter known tactics. To succeed in repeated encounters, an intelligent defender must plan effectively while actively defying opponent modeling efforts.

Proc. of the 25th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2026), C. Amato, L. Dennis, V. Mascardi, J. Thangarajah (eds.), May 25 – 29, 2026, Paphos, Cyprus. © 2026 International Foundation for Autonomous Agents and Multiagent Systems (www.ifaamas.org). https://doi.org/10.65109/KOTB1262

Reinforcement learning (RL) provides a powerful framework for discovering complex, high-performing policies through trial and error. Yet these policies are often opaque: their internal logic may be difficult to interpret, even for their designers. Explainable RL (XRL) seeks to address this opacity by constructing surrogate models, such as decision trees or linear approximators, that mimic an agent's behavior in an interpretable form [15, 16]. These surrogates sacrifice fidelity for transparency, offering valuable insight into a learned policy's structure and priorities.

In adversarial environments, however, interpretability can be a double-edged sword. If an opponent can accurately model and understand a policy, it gains the power to anticipate, manipulate, or exploit it. Many domains—from autonomous defense systems and competitive robotics to economic negotiation—therefore require agents that are not only competent but also strategically opaque. In such cases, predictability becomes a liability.

This work reverses the usual XRL objective. Instead of using surrogate models to make our policies interpretable, we embed them in training as adversarial observers to be defeated. By penalizing how well a surrogate can imitate the agent, we explicitly incentivize the policy to behave in ways that are harder to model. This approach encourages *emergent deception*: the spontaneous generation of behavior that misleads predictive models while still accomplishing the task at hand.

Overview of Contributions

We introduce **SAD-RL** (*Surrogate-Augmented Deception in Reinforcement Learning*), a general framework for training reinforcement learning agents to resist modeling and exploitation. The method integrates surrogate predictors directly into the learning loop, transforming interpretability tools into adversarial training instruments. Our key contributions are as follows:

- **Formulation:** We define the SAD-RL objective and training procedure, which augment the learner's reward with a penalty proportional to surrogate model accuracy. This transforms explainability into a controllable deception mechanism via a tunable weight λ.
- **Architecture and Algorithm:** We present a domain- and policy-agnostic training loop (Algorithm 1) and architecture diagram (Figure 1) illustrating how surrogate feedback integrates seamlessly with standard RL pipelines.
- **Empirical Validation:** We evaluate SAD-RL in two representative adversarial domains—a discrete *Adversarial Grid World* (AGW) and a continuous *Sharks and Minnows* (SaM) pursuit-evasion task—showing that SAD-RL agents maintain strong task performance while achieving measurable deception against decision and regression tree surrogates.

Analysis and Metrics: We analyze the trade-off between task performance and deception using ϵ-tolerant surrogate accuracy, identifying distinct modes of balanced and over-deceptive behavior.

Together, these contributions establish SAD-RL as a principled, domain-agnostic foundation for studying strategic opacity and deception in reinforcement learning.

2 RELATED WORKS

2.1 Predictability and Explainability

Gil *et al.* investigate methods for making agent behavior more predictable and legible to teammates in collaborative settings [6]. Although the goals of predictability and deception are at odds, their discussion of the trade-offs between legibility and unpredictability is relevant, especially in adversarial or mixed-motive environments. Notably, their introduction of a tunable λ parameter to control behavioral predictability could be inverted to generate intentionally deceptive trajectories.

Xiong *et al.* introduce XRL-Bench, a benchmark and taxonomy for evaluating explainability in reinforcement learning [16]. They focus on objective metrics such as fidelity, stability, and fairness, and critique over-reliance on subjective measures like user satisfaction. Their work reinforces the need for quantitative metrics when evaluating surrogate models and explainability mechanisms in RL.

Sieusahai and Guzdial propose the use of interpretable surrogate models—specifically decision trees combined with sprite visualizations—to explain deep Q-networks (DQNs) trained on pixel-level input [15]. They introduce a fidelity metric based on perturbing important features and measuring resulting action changes, which provides a useful technique for surrogate validation.

Taken together, these works view surrogate modeling as a tool for transparency. SAD-RL inverts this perspective: rather than seeking faithful surrogates, it leverages surrogate fidelity as an adversarial pressure that the agent must learn to minimize. This inversion transforms interpretability from a *post hoc* analytic tool into an online, dynamic training mechanism.

2.2 Deception

Schneider *et al.* examine how machine learning models can deceive users through selective or misleading explanations [13]. While their focus is on the misuse of XAI tools in human-AI interaction, their findings suggest that deception can emerge not only through behavior but also through interpretability mechanisms—a concern that warrants attention in safety-critical settings.

Kim *et al.* formalize two classes of deceptive strategies—diversionary and targeted—by defining optimization problems with a tunable deception parameter β [8]. Their work evaluates deception in a multiagent moving target defense (MTD) setting, assuming that the adversary learns via inverse reinforcement learning (IRL). They track opponent misclassification rates as their metric for deception. However, the assumption of uniform agent capabilities may limit generalization to broader adversarial scenarios.

In another work, Kim *et al.* introduce *equivocal deception*, where the goal is to leave the opponent equally uncertain about the agent's true intent and another possible goal [7]. They compare this approach to targeted and diversionary deception and find that equivocal deception offers a no-regret tradeoff, albeit under the same IRL adversary assumption.

Nichols *et al.* propose Adversarial RRT*, a planning algorithm that generates near-optimal but less predictable paths [12]. Their framework measures deception using observer confidence and models the planner-observer interaction as a form of recursive reasoning akin to level-k thinking. This structure parallels adversarial reinforcement learning in settings where agents must anticipate how their intentions will be inferred.

Birmpas *et al.* explore optimal follower strategies in Stackelberg games [1]. While their analysis yields useful theoretical insights, the assumptions of fixed leader-follower roles and static environments with full observability diverge significantly from the dynamic and partially observable adversarial scenarios studied in this work.

Collectively, prior deception research defines deception as an explicit optimization objective or information asymmetry engineered by the designer. In contrast, SAD-RL induces deception implicitly through a secondary learning signal that rewards being hard to model, allowing deceptive behavior to emerge naturally during training rather than being hand-crafted.

2.3 Adversarial Attacks

Fujimoto *et al.* study an adversarial setting in which an attacker maximizes the entropy of a victim's action distribution, effectively inducing confusion or unpredictability in the victim policy [5]. The attacker may do this to slow the progress of the victim in order to buy time for its own policy's success. This entropy-maximization objective does not require modeling the victim's reward function and serves as a valuable baseline for evaluating robustness against "reward-free" adversarial policies. Though most effective against static victims, this work highlights a minimal-assumption approach to adversarial interference.

While such approaches focus on destabilizing external agents, SAD-RL embeds the adversary within the agent's own training loop, transforming external attack pressure into an internalized regularizer. This reframing bridges adversarial robustness, explainability, and deception, filling a gap between interpretability-focused XRL and opponent modeling in multiagent RL.

Across these areas, no prior work explicitly integrates surrogate modeling into the reinforcement learning process as a source of deceptive pressure. SAD-RL unifies insights from explainable RL, adversarial training, and deceptive planning to produce agents that are both effective and strategically opaque. The next section formalizes this framework and describes how surrogate fidelity can be operationalized as a deception-aware training signal.

3 METHODOLOGY

3.1 Surrogate Models in Adversarial RL

Surrogate models are lightweight, interpretable learners, such as decision or regression trees, commonly used in explainable reinforcement learning (XRL) to approximate and interpret complex black-box policies. For example, Coppens *et al.* distill deep RL policies into soft decision trees to reveal which features drive agent

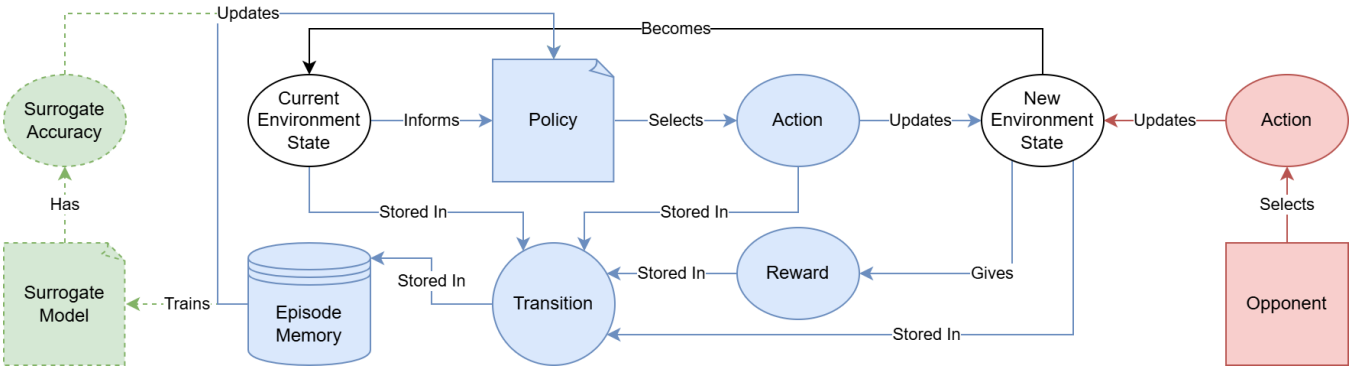


Figure 1: SAD-RL architecture diagram. The blue components represent our agent's RL training loop, while those in red represent the adversary. The dotted green shapes are our novel augmentations to the training process, where the surrogate model is trained to mimic our agent's policy.

behavior [2]. These models offer transparency in high-stakes environments, enabling oversight, debugging, and human-agent coordination.

The same tools that make policies explainable can also render them vulnerable. In adversarial or multi-agent settings where opponents observe behavior across repeated interactions, it is plausible that they would train surrogates to predict and exploit an agent's decision-making strategy. This risk is especially relevant in mixed-initiative systems, where interpretability is not just a feature but also a potential vector for manipulation.

We invert this typical use: rather than interpreting our agents with surrogates, we treat surrogates as stand-ins for adversarial models. Assuming an intelligent opponent might use a decision tree (or similar learner) to model our policy, we proactively train the agent to resist such modeling.

Operationally, we integrate a surrogate into the training loop. At the end of each episode, a surrogate is trained on the agent's observed state-action pairs, and its predictive accuracy determines an additional penalty in the reward function. This introduces a secondary objective: in addition to maximizing task performance, the agent is incentivized to minimize surrogate fidelity. The resulting policies not only perform well, but do so through behavior that is harder to model, enabling emergent deception.

3.2 Threat Model and Design Objectives

We adopt a static-observer threat model during training: the opponent does not learn online but is assumed to fit a predictive surrogate to recent episodes and exploit that model. SAD-RL therefore treats the surrogate as a proxy for the opponent's modeling capability and penalizes the agent whenever the surrogate attains high predictive fidelity. This isolates the effect of modeling pressure without confounds from co-adaptation.

Our design goals are threefold: (i) preserve task competency, (ii) reduce modelability (strategic opacity), and (iii) remain algorithm- and domain-agnostic. In practice, SAD-RL augments the terminal reward with a deception term weighted by $\lambda \in [0, 1]$, enabling a smooth performance-opacity trade-off and yielding interpretable Pareto-like fronts in evaluation.

3.3 SAD-RL Architecture

SAD-RL enhances a standard reinforcement learning pipeline by incorporating a surrogate-based deception mechanism into the training loop. During training, the agent interacts with the environment by observing its current state, selecting an action according to its policy, and transitioning to a new state while receiving a reward. Each transition $(s_t, a_t, r_t, s_{t+1}, d_t)$ is stored for policy updates. After an episode concludes, the RL algorithm computes returns and updates the policy.

SAD-RL modifies this process by introducing a surrogate during the update phase. The surrogate is trained on the episode's state-action pairs to approximate the agent's policy. Its fidelity (how accurately it mimics the agent's behavior) is then used to augment the reward signal. Specifically, the agent receives a penalty proportional to the surrogate's fidelity, incentivizing behaviors that are harder for the surrogate—and, by extension, a modeling opponent—to predict. This yields a multi-objective RL setup: maximize task performance while minimizing predictability (equivalently, maximize deception). Figure 1 summarizes the full pipeline.

SAD-RL is compatible with a wide range of reinforcement learning paradigms (e.g., PPO, DQN, DDPG), as it modifies only the reward structure rather than the underlying policy optimization procedure. To formalize this process, Algorithm 1 presents a domain- and policy-agnostic description of the SAD-RL training loop. The algorithm mirrors the standard RL cycle shown in Figure 1. This pseudocode highlights how the surrogate's predictive accuracy is translated into a deception penalty that modifies the terminal reward before the policy update, making the approach compatible with both on- and off-policy algorithms.

Algorithm 1 illustrates how SAD-RL can be integrated into any standard RL architecture without altering the underlying optimization paradigm. The surrogate and policy operate asynchronously: the surrogate is trained only after each episode and does not interfere with step-by-step action selection. This separation ensures that SAD-RL remains algorithm-agnostic, allowing it to be implemented alongside diverse training strategies such as PPO, DQN, or DDPG with minimal modification.

Algorithm 1 SAD-RL Training Procedure

Require: policy π_θ, surrogate g_ψ, deception weight $\lambda \geq 0$, accuracy metric $\mathrm{Acc}(\cdot) \in [0,1]$, horizon H

for episode $e = 1, 2, \ldots, E$ **do**
  Initialize environment; receive s_1
  Initialize episode buffer $\mathcal{B}_e \leftarrow \emptyset$
  for $t = 1$ **to** H **do**
    Observe s_t
    Sample $a_t \sim \pi_\theta(\cdot \mid s_t)$
    Execute a_t
    Observe r_t, next state s_{t+1}, terminal flag $d_t \in \{0, 1\}$
    Append $(s_t, a_t, r_t, s_{t+1}, d_t)$ to $\mathcal{B}_e$
    if $d_t = 1$ **then break**
    end if
  end for
  Compute outcome reward $R_{\text{task}} \in \mathbb{R}$ for the episode
  Select disjoint subsets $\mathcal{T}_e, \mathcal{V}_e \subseteq \mathcal{B}_e$
  Fit g_ψ on $\mathcal{T}_e$ to predict actions from states: $g_\psi : s \mapsto \hat{a}$
  $A_e \leftarrow \mathrm{Acc}\Big(\{(g_\psi(s), a) \mid (s, a, \cdot, \cdot, \cdot) \in \mathcal{V}_e\}\Big)$
  $R_{\text{final}} \leftarrow R_{\text{task}} - \lambda \cdot A_e$ ▷ task–deception trade-off
  Replace terminal reward:
  Let T_e be the index of the last tuple in $\mathcal{B}_e$
  Set $(s_{T_e}, a_{T_e}, r_{T_e}, s_{T_e+1}, d_{T_e}) \leftarrow (s_{T_e}, a_{T_e}, R_{\text{final}}, s_{T_e+1}, d_{T_e})$
  Update π_θ using $\mathcal{B}_e$ ▷ if off-policy: add to replay buffer
end for

In our experiments, the abstract terminal reward R_{final} in Algorithm 1 is instantiated using the following formulation:

$$R_{\text{final}} = [(1-\lambda)R_{\text{task}} + \lambda(1-2A_e)](H - T_e), \quad R_{\text{final}} \in [-H, H]. \quad (1)$$

Here, $A_e \in [0,1]$ is the predictive accuracy of the surrogate model, measured on a held-out validation split as

$$A_e = \frac{1}{|\mathcal{V}_e|} \sum_{(s,a)\in\mathcal{V}_e} \mathbf{1}\big[\, \|g_\psi(s) - a\|_\infty \leq \epsilon \,\big], \quad (2)$$

where $\epsilon = 0.5$ in all continuous-action experiments and equality is used for discrete actions. Additionally, $\lambda \in [0,1]$ controls the trade-off between task performance and deception, H is the maximum episode length, and T_e is the terminal timestep. This formulation rewards early task completion while penalizing predictability, encouraging agents to remain effective yet difficult to model.

We instantiate g_ψ as lightweight learners that balance interpretability and efficiency. For discrete action spaces, we use decision trees of depth 3; for continuous control, regression trees of depth 7. Surrogates are trained offline after each episode on the agent's state-action pairs, so fitting does not interfere with online exploration. Accuracy $A_e \in [0,1]$ is computed with the ϵ-tolerant criterion above, and deception weight λ governs the performance-opacity trade-off. Unless stated otherwise, we sweep λ to produce trade-off curves in Section 4. To verify that deception is not simply a by-product of higher policy entropy, we later analyze the relationship between per-episode entropy and deception (Section 4.3).

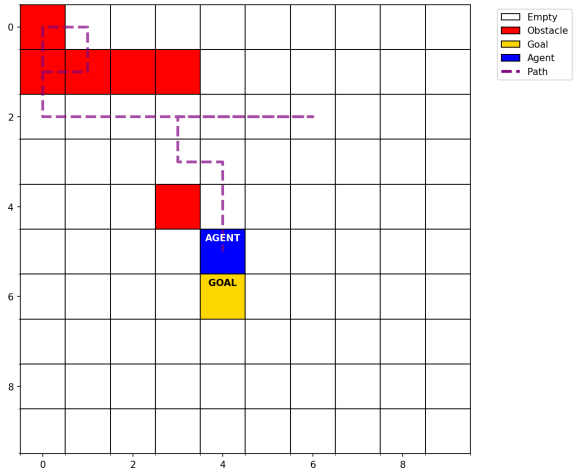


Figure 2: Snapshot of an AGW episode.

3.4 Adversarial Grid World (AGW)

3.4.1 Environment Summary. The AGW domain is a discrete 2D environment where a navigating agent starts in the top-left corner and aims to reach a randomly placed goal that is sufficiently far away. An adversarial observer, unaware of the goal's location, attempts to block the agent by placing obstacles with a 15% chance each timestep. The observer starts by placing obstacles randomly, then as the episode progresses, it attempts to block the agent's progress based on its recent history and most common directions. The navigator's outcome state s_n is encoded as a 1 for reaching the goal and -1 for failing to do so. Each episode lasts for a maximum of 100 steps. Figure 2 shows a single timestep from an AGW episode.

3.4.2 Observation and Action Spaces.

- Observations: 13 dimensions normalized to $[-1, 1]$
- Actions: 4 discrete moves (up, down, left, right)

3.4.3 Surrogates. We use decision trees of maximum depth 3—which measure impurity using entropy—as surrogates.

3.4.4 Training Setup. We trained nine deep Q-network (DQN) models [10], each with a different λ value. Each uses a standard architecture: two hidden layers of 64 nodes with ReLU activations, ϵ-greedy exploration, and an Adam optimizer with a learning rate of .001.

3.5 Sharks and Minnows (SaM)

3.5.1 Environment Summary. SaM is a continuous 2D arena with a minnow trying to reach the opposite boundary while a shark attempts to intercept it. Both agents spawn at random locations on opposite walls, obey Dubins path constraints [3], and move at constant speeds. Entities that collide with an outer wall are eliminated, except for minnows who reach the goal wall. The minnow's terminal reward R_{task} is encoded as -1 if it dies, 0 if it survives without reaching the goal, and 1 if it successfully reaches the target wall. Each episode spans a simulated duration of 10 minutes, with updates occurring at 10 Hz, resulting in a maximum of 6000 discrete timesteps per episode. Figures 6a and 6b in the Results section display final agent trajectories within the SaM environment.

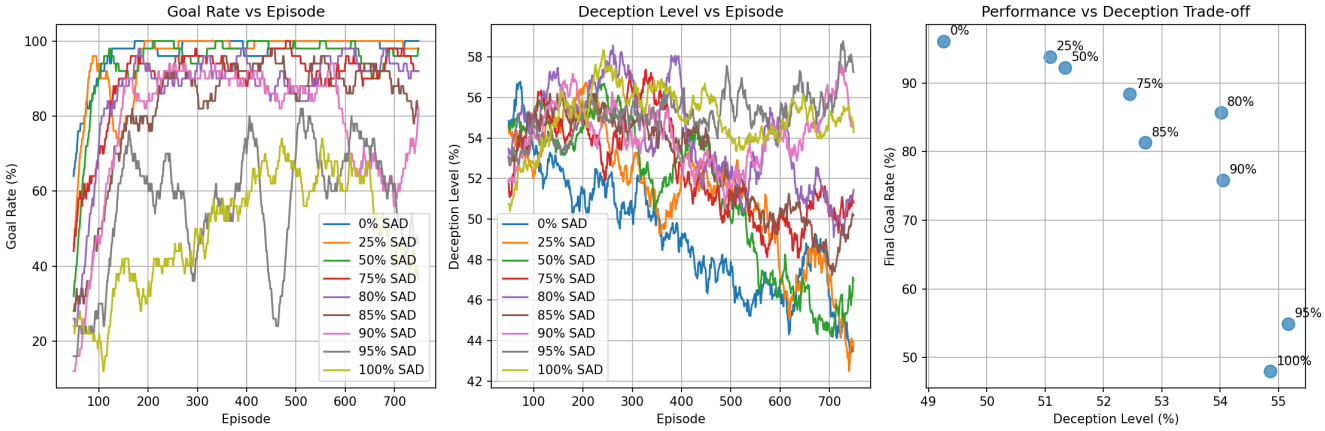


Figure 3: Analysis of AGW training on nine different SAD-RL models. The left plot portrays goal rate over time, the middle shows deception over time, and the rightmost plot shows the tradeoff curve between the two metrics.

3.5.2 Observation and Action Spaces.

- Observation: 12 features normalized to $[-1, 1]$
- Actions: 2D continuous control (speed, heading change)

3.5.3 Surrogates. We use regression trees of maximum depth 7 that are evaluated using ϵ-tolerant accuracy as described above.

3.5.4 Dense Rewards. In addition to the sparse SAD-RL rewards at the end of each episode (Equation 1), the minnow is rewarded at each timestep for its normalized proximity to the goal. This encourages forward progress and penalizes excessive stagnation or retreating.

3.5.5 Curriculum Training. To establish a robust baseline policy for the minnow agent, we employed curriculum learning, a common approach for creating stable strategies in RL [11]. Thus, the minnow learns against increasingly difficult sharks:

(1) Stationary shark
(2) Slow pure pursuit shark
(3) Half-speed pure pursuit shark
(4) Equal-speed pure pursuit shark

Here, a pure pursuit shark attempts to point directly at the minnow's position every timestep. This fixed policy is just difficult enough to both allow minnows to learn evasive strategies while facing consequences for failure.

3.5.6 Training Setup. We initially tried deep deterministic policy gradient (DDPG) [9] and twin delayed DDPG (TD3) [4] approaches, but with little success. Instead, we trained four agents using proximal policy optimization (PPO) [14] by varying the value of λ:

(1) Baseline ($\lambda = 0.00$) — no surrogate augmentation
(2) 25% SAD ($\lambda = 0.25$)
(3) 50% SAD ($\lambda = 0.50$)
(4) 75% SAD ($\lambda = 0.75$)

Each agent was trained from the same post-curriculum base policy for 5000+ episodes with seven checkpoints, producing 28 models. Each uses a standard architecture: actor and critic neural networks comprised of two 64-node hidden layers with tanh activations, an ϵ clipping factor of .2, and an Adam optimizer with a learning rate of .0003.

4 RESULTS

4.1 Adversarial Grid World (AGW)

Figure 3 summarizes the effect of varying the deception weight λ across nine DQN-trained models. The left subplot shows mean goal rate, the middle shows mean deception, and the rightmost shows the trade-off curve between the two objectives. Each curve is a rolling mean over 50 episodes. As λ increases, surrogate accuracy drops and trajectories become more irregular, indicating that higher deception pressure produces harder-to-model behavior.

Figure 4 aggregates training episodes into four categories: *good and deceptive*, *good but predictable*, *bad yet deceptive*, and *bad and predictable*. The green (successful and deceptive) mass peaks near $\lambda = 0.8$, suggesting a sweet spot where agents achieve both objectives concurrently. Orange episodes (*bad yet deceptive*) also grow at high λ, reflecting occasional over-deception (see Section 5).

Figure 5 visualizes three representative models ($\lambda \in \{0.0, 0.8, 1.0\}$) in the two-objective space. Each point is a rolling mean over 75 episodes. The $\lambda = 0.8$ model concentrates near the top-right quadrant—a Pareto-like frontier of performance and opacity—whereas $\lambda = 1.0$ begins to trade effectiveness for unpredictability. This convex, frontier-like trend matches the controllable trade-off implied by Equation 1.

4.2 Sharks and Minnows (SaM)

Figures 6a and 6b show representative trajectories. The baseline minnow follows a relatively direct route, pulling the shark inward before veering toward the goal. By contrast, the SAD-RL minnow with $\lambda = 0.50$ advances briefly toward the shark, then executes a gentle sinusoidal diversion before slipping past—qualitatively resembling baiting and luring tactics in predator-prey pursuit.

Each model was evaluated over 300 randomized test episodes with equal-speed sharks. Figure 7 plots average win proportion versus deception ($1 - \bar{A}$, where $\bar{A}$ is mean surrogate accuracy).

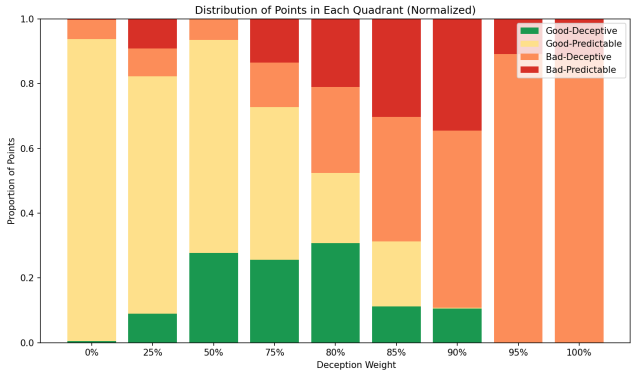


Figure 4: AGW training episodes categorized by their ability to jointly achieve task success and deception. Green bars indicate episodes succeeding at both objectives, red indicate failures at both, and yellow/orange indicate partial success.

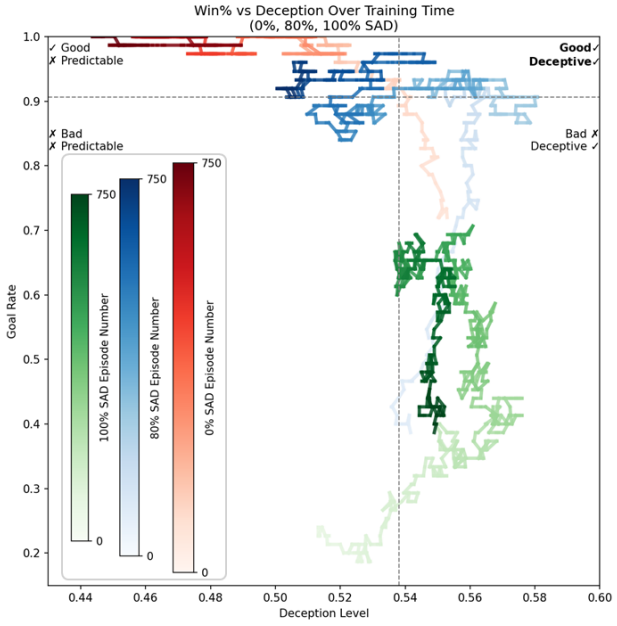
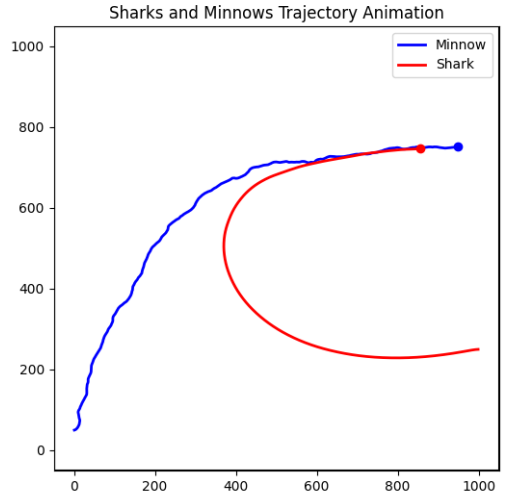
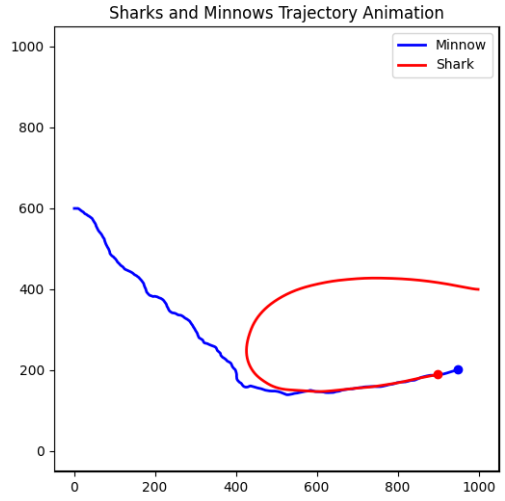


Figure 5: AGW win rate vs. deception over training time. Darker lines represent later episodes. Dotted lines denote median values across all models. The $\lambda = 0.8$ agent approaches the top-right "good and deceptive" region.

The $\lambda = 0.50$ model consistently occupies the top-right quadrant, indicating high task success and strong resistance to modeling. Elliptical contours (one to three standard deviations) reveal distinct behavioral modes as λ increases: low-λ models cluster near predictable success, whereas high-λ models drift toward deceptive but inconsistent play.



(a) Baseline minnow trajectory.



(b) SAD-RL minnow trajectory with $\lambda = 0.5$.

Figure 6: SaM minnow trajectories under different λ settings.

4.3 Not Just Randomness

A potential concern is that SAD-RL's deception might arise from higher action entropy—that is, stochasticity rather than structured misdirection. We therefore computed per-episode action entropies and surrogate deception scores for each trained model. Within each fixed λ, the Spearman correlation between entropy and deception was negligible ($|\rho| < 0.17$). Pooling all episodes produced a moderate positive correlation ($\rho \approx 0.50$), but this effect disappeared when controlling for λ (partial $\rho = 0.017$, $p > 0.8$), indicating that the pooled trend reflects differences between policy families rather

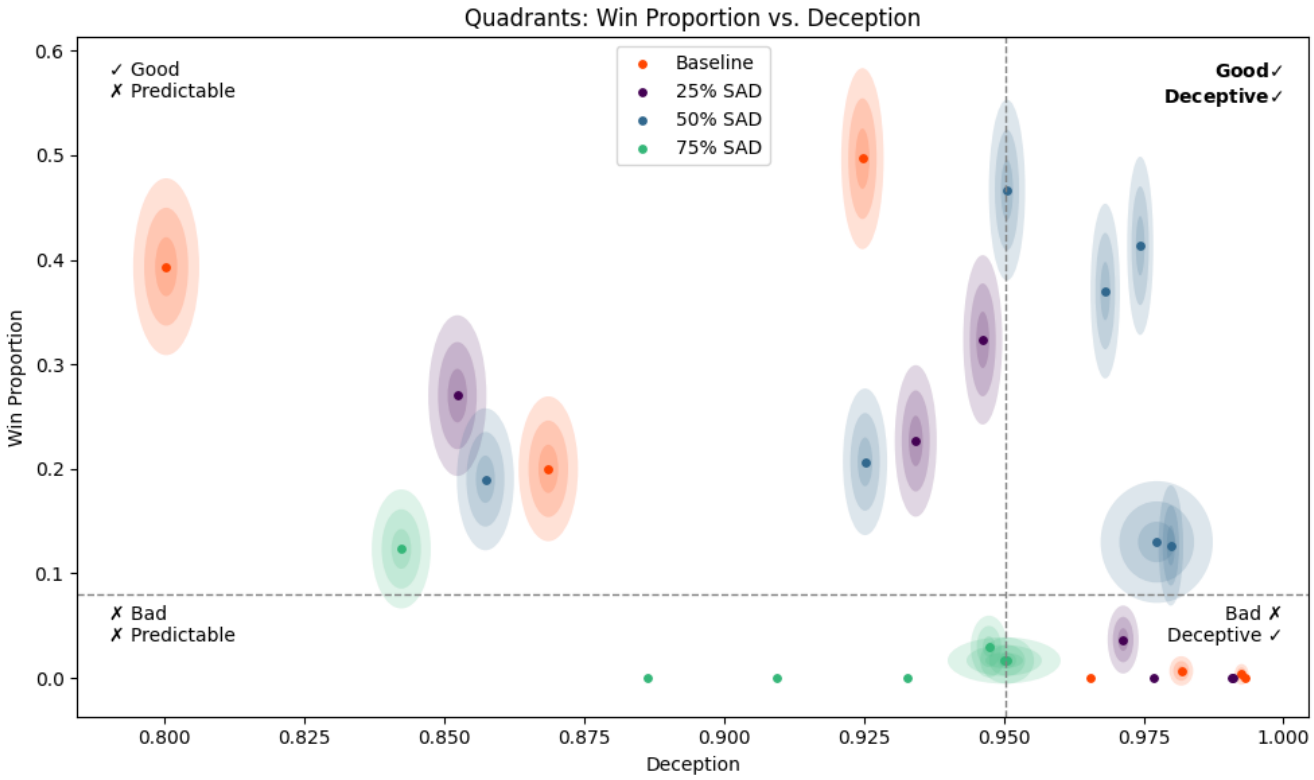


Figure 7: Win proportion vs. deception for all SaM models. The $\lambda = 0.50$ agent lies in the top-right quadrant, balancing effectiveness and opacity.

than random variation within them. An entropy-matched control—constructed by injecting Gaussian noise into the baseline agent to reproduce SAD(0.50)'s mean entropy—failed to achieve comparable deception, confirming that SAD-RL yields structured, goal-directed opacity rather than random dithering. Together, these analyses provide quantitative evidence that SAD-RL yields structured, goal-directed opacity rather than entropy-driven stochasticity.

4.4 Cross-Domain Trends

Across both environments, moderate deception weights ($\lambda \in [0.5, 0.8]$) yield the best balance between goal achievement and surrogate misprediction. Higher values push agents toward erratic, low-reward behaviors, while $\lambda = 0$ reproduces standard RL. These results collectively demonstrate that SAD-RL's surrogate penalty induces controllable, emergent deception that generalizes across discrete and continuous domains.

5 DISCUSSION

Our experiments in both AGW and SaM domains support SAD-RL's central claim: penalizing surrogate-model accuracy during training encourages the emergence of deceptive yet performant agent behaviors.

In AGW, the trade-off between deception and performance (Figure 3) follows a nonlinear, convex trend similar to a Pareto frontier, where improvement in one objective often comes at the expense of the other. Adjusting the surrogate weight λ enables flexible navigation of this frontier. Models trained with $\lambda = 0.80$ achieved the best balance, spending the largest proportion of training episodes in the "good and deceptive" quadrant (Figure 4) and clustering near the top right of the objective space (Figure 5).

In the SaM domain, we observed qualitatively distinct patterns between baseline and SAD-RL agents. The baseline minnow (Figure 6a) pursues a direct, high-speed path—effective but predictable once observed. By contrast, the SAD-RL minnow (Figure 6b) advances briefly toward the shark before curving away in a faint sinusoidal pattern, baiting the predator inward before escaping along the boundary. This illustrates *emergent deception*: misleading an opponent through subtle, structured deviations rather than random noise.

Figure 7 quantifies this effect. Baseline models tend to occupy the "good but predictable" or "bad but deceptive" regions, while SAD-RL agents—especially at $\lambda = 0.50$—concentrate in the top-right quadrant, balancing task success with opacity. These findings confirm that deception-aware training can produce agents that are both effective and resilient to adversarial modeling.

Across both environments, we also observe agents that are *deceptive but poor performers*. As shown in the lower-right quadrants of Figures 5 and 7, excessive deception pressure can lead to erratic

or evasive behaviors that harm performance. This underscores the inherent tension between competence and opacity and the need to carefully tune λ. Future extensions could automate this balance through meta-learning or adaptive weighting.

SAD-RL reframes *strategic opacity* in reinforcement learning. Whereas prior work often equates unpredictability with entropy, SAD-RL shows that opacity can emerge systematically from adversarial pressure on modelability. The framework thus bridges explainability and deception: the same surrogate tools that reveal policies can also be repurposed to teach them how to hide intent.

Methodologically, surrogate fidelity was quantified via ϵ-tolerant accuracy, counting a prediction as correct only if it lay within a small threshold of the true action. This binary metric enabled consistent cross-domain comparison and avoided scale-dependent measures such as R^2. It grounds deception in a reproducible statistic rather than subjective interpretation, aligning SAD-RL with the rigor of modern XRL evaluation frameworks.

Together, these results suggest that SAD-RL induces emergent deceptive behavior without domain-specific heuristics. Embedding modeling pressure directly into the reward structure generalizes across tasks and architectures, allowing agents to learn deception organically as a byproduct of resisting predictability.

These findings also rule out the possibility that SAD-RL's deception merely reflects randomness. Even when action entropy was artificially matched to that of a SAD-RL agent, baseline policies failed to achieve similar deception. This supports the interpretation that SAD-RL produces *goal-directed opacity*—strategic unpredictability tuned to mislead modeling adversaries—rather than noise-driven stochasticity.

Several limitations remain. The adversaries in our experiments followed static policies and the surrogates were intentionally shallow to isolate the effects of deception pressure. These design choices clarify causality but limit generality: deeper or adaptive surrogates could alter the dynamics substantially. Real-world opponents also learn and co-adapt, introducing non-stationarity absent from our setup. Examining SAD-RL under co-evolving or competitive regimes is an important next step toward robust multiagent deception.

A key open question is how SAD-RL-trained agents perform when both sides actively learn to model and counter each other. We hypothesize that agents trained to minimize surrogate fidelity will exhibit greater robustness against adaptive opponents, though this remains to be tested. Exploring these interactions—and the equilibrium between opacity and adaptation—offers a promising direction for future research on deception in reinforcement learning.

6 CONCLUSION

We introduced **SAD-RL** (*Surrogate-Augmented Deception in Reinforcement Learning*), a framework that trains agents to resist adversarial modeling by penalizing surrogate fidelity during learning. In contrast to traditional explainable RL approaches that seek transparency, SAD-RL inverts this paradigm to promote *strategic opacity*: behavior that remains effective while defying opponent inference.

Across two adversarial environments—a continuous pursuit-evasion task and a discrete adversarial grid world—SAD-RL agents outperformed baseline RL in both deception and task performance. Intermediate surrogate weights ($\lambda = 0.50$ or 0.80) produced the most balanced outcomes, forming a controllable Pareto frontier between performance and opacity. These results confirm that surrogate-driven training pressure alone can induce emergent deceptive behavior without adversarial imitation, handcrafted strategies, or domain-specific assumptions.

SAD-RL thus provides a principled, domain-agnostic method for training resilient RL agents. By transforming interpretability mechanisms into adversarial training tools, it redefines transparency as a controllable design variable rather than a fixed property. This perspective opens the door to agents that are not merely high-performing but strategically elusive—laying a foundation for deception-aware AI in competitive and safety-critical domains.

Supplementary Material. All simulation code, trained models, analysis scripts, and any other supplementary materials are hosted in a dedicated GitHub repository, found here: https://github.com/Shymkis/SAD-RL.

7 FUTURE WORK

Several extensions merit further exploration:

- **Training the Adversary:** Train the opposing agent (e.g., the shark) with its own surrogate penalty, enabling symmetric deception and co-evolutionary dynamics.
- **Adaptive Opponents:** Evaluate SAD-RL-trained agents against adaptive adversaries that iteratively refine surrogates during play, testing robustness under non-stationary learning.
- **Surrogate Diversity:** Investigate alternative surrogate types—random forests, LIME, SHAP, or neural predictors—and their impact on emergent deception styles.
- **Multiagent Scenarios:** Extend SAD-RL to multiagent systems (e.g., teams of minnows or sharks) to study coordinated or collective deception.
- **Cross-Domain Validation:** Apply SAD-RL to new adversarial domains such as automated negotiation, moving-target defense, or capture-the-flag to assess generality.
- **Comparative Benchmarking:** Conduct head-to-head comparisons with other deception-aware approaches, including policy regularization, entropy shaping, and goal obfuscation.
- **Policy-Gradient Integration:** Incorporate surrogate loss directly into actor-critic or policy-gradient updates to improve stability and sample efficiency.
- **Adaptive λ Optimization:** Develop methods to automatically tune the deception weight λ via meta-learning or Pareto-based adaptive scheduling.

Overall, SAD-RL establishes deception as a first-class optimization signal in reinforcement learning, demonstrating that the ability to resist being modeled can itself be learned—laying the groundwork for future systems that are both intelligent and intentionally inscrutable.

ACKNOWLEDGMENTS

We would like to thank the Air Force Research Laboratory and the Cyber Fellows program at The University of Tulsa for supporting this work.

REFERENCES

[1] Georgios Birmpas, Jiarui Gan, Alexandros Hollender, Francisco Marmolejo, Ninad Rajgopal, and Alexandros Voudouris. 2020. Optimally Deceiving a Learning Leader in Stackelberg Games. In *Advances in Neural Information Processing Systems*, Vol. 33. Curran Associates, Inc., 20624–20635. https://proceedings.neurips.cc/paper/2020/hash/ed383ec94720d62a939bfb6bdd98f50c-Abstract.html

[2] Youri Coppens, Kyriakos Efthymiadis, Tom Lenaerts, Ann Nowé, Tim Miller, Rosina Weber, and Daniele Magazzeni. 2019. Distilling deep reinforcement learning policies in soft decision trees. In *Proceedings of the IJCAI 2019 workshop on explainable artificial intelligence*. 1–6.

[3] Lester E Dubins. 1957. On curves of minimal length with a constraint on average curvature, and with prescribed initial and terminal positions and tangents. *American Journal of mathematics* 79, 3 (1957), 497–516.

[4] Scott Fujimoto, Herke Hoof, and David Meger. 2018. Addressing function approximation error in actor-critic methods. In *International conference on machine learning*. PMLR, 1587–1596.

[5] Ted Fujimoto, Timothy Doster, Adam Attarian, Jill Brandenberger, and Nathan Hodas. 2021. Reward-Free Attacks in Multi-Agent Reinforcement Learning. https://doi.org/10.48550/arXiv.2112.00940 arXiv:2112.00940 [cs].

[6] Roman Chiva Gil, Daniel Jarne Ornia, Khaled A. Mustafa, and Javier Alonso Mora. 2024. Predictability Awareness for Efficient and Robust Multi-Agent Coordination. https://doi.org/10.48550/arXiv.2411.06223 arXiv:2411.06223 [cs].

[7] Yerin Kim, Alexander Benvenuti, Bo Chen, Mustafa Karabag, Abhishek Kulkarni, Nathaniel D. Bastian, Ufuk Topcu, and Matthew Hale. 2024. Defining and Measuring Deception in Sequential Decision Systems: Application to Network Defense. In *MILCOM 2024 - 2024 IEEE Military Communications Conference (MILCOM)*. 1–6. https://doi.org/10.1109/MILCOM61039.2024.10773660 ISSN: 2155-7586.

[8] Yerin Kim, Alexander Benvenuti, Bo Chen, Mustafa Karabag, Abhishek Kulkarni, Nathaniel D. Bastian, Ufuk Topcu, and Matthew Hale. 2025. Deceptive Sequential Decision-Making via Regularized Policy Optimization. https://doi.org/10.48550/arXiv.2501.18803 arXiv:2501.18803 [cs].

[9] Timothy P Lillicrap, Jonathan J Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. 2015. Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971* (2015).

[10] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. 2013. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602* (2013).

[11] Sanmit Narvekar, Bei Peng, Matteo Leonetti, Jivko Sinapov, Matthew E Taylor, and Peter Stone. 2020. Curriculum learning for reinforcement learning domains: A framework and survey. *Journal of Machine Learning Research* 21, 181 (2020), 1–50.

[12] Hayden Nichols, Mark Jimenez, Zachary Goddard, Michael Sparapany, Byron Boots, and Anirban Mazumdar. 2022. Adversarial Sampling-Based Motion Planning. *IEEE Robotics and Automation Letters* 7, 2 (April 2022), 4267–4274. https://doi.org/10.1109/LRA.2022.3148464

[13] Johannes Schneider, Christian Meske, and Michalis Vlachos. 2023. Deceptive XAI: Typology, Creation and Detection. *SN Computer Science* 5, 1 (Dec. 2023), 81. https://doi.org/10.1007/s42979-023-02401-z

[14] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347* (2017).

[15] Alexander Sieusahai and Matthew Guzdial. 2021. Explaining Deep Reinforcement Learning Agents in the Atari Domain through a Surrogate Model. *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment* 17, 1 (Oct. 2021), 82–90. https://doi.org/10.1609/aiide.v17i1.18894

[16] Yu Xiong, Zhipeng Hu, Ye Huang, Runze Wu, Kai Guan, XingChen Fang, Ji Jiang, Tianze Zhou, YuJing Hu, Haoyu Liu, Tangjie Lyu, and Changjie Fan. 2024. XRL-Bench: A Benchmark for Evaluating and Comparing Explainable Reinforcement Learning Techniques. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. ACM, Barcelona Spain, 6073–6082. https://doi.org/10.1145/3637528.3671595