Predictive Red Teaming: Breaking Policies Without Breaking Robots

Anirudha Majumdar[1,2], **Mohit Sharma**[1], **Dmitry Kalashnikov**[1], **Sumeet Singh**[1], **Pierre Sermanet**[1], **and Vikas Sindhwani**[1]
[1]Google DeepMind, [2]Princeton University
animajumdar@google.com

Abstract: Visuomotor policies trained via imitation learning are capable of challenging manipulation tasks, but are often extremely brittle to lighting, visual distractors, and object locations. These vulnerabilities can depend unpredictably on the specifics of training, and are challenging to expose without expensive hardware evaluations. We propose *predictive red teaming*: discovering vulnerabilities of a policy with respect to environmental factors, and predicting the corresponding performance degradation *without* hardware evaluations in off-nominal scenarios. In order to achieve this, we develop RoboART: an automated red teaming (ART) pipeline that (1) modifies nominal observations using generative image editing to vary environmental factors, and (2) predicts performance under each variation using a policy-specific anomaly detector executed on edited observations. Experiments across 500+ hardware trials in twelve off-nominal conditions for visuomotor diffusion policies demonstrate that RoboART predicts performance degradation with high accuracy (less than 0.19 average difference between predicted and real success rates). We also demonstrate how predictive red teaming enables *targeted data collection*: fine-tuning with data collected under conditions predicted to be adverse boosts baseline performance by 2–7x. Website: URL.

Keywords: Red teaming, evaluation, targeted data collection

1 Introduction

Is it possible to expose the vulnerabilities of a given robot policy with respect to changes in environmental factors such as lighting, visual distractors, and object placement *without performing hardware evaluations in these scenarios*? As we seek to deploy robots in environments with ever-increasing complexity, it becomes imperative to develop scalable methods for predicting how well they will generalize when faced with unseen scenarios. Performing hardware evaluations to discover vulnerabilities — which can depend in surprising ways on the specifics of policy training and architecture — is often prohibitively expensive to set up and execute in diverse scenarios.

As an example, consider a visuomotor diffusion policy [1] trained for pick-and-place tasks (Fig. 1) via behavior cloning using a large dataset: over 3K+ demonstrations with varied objects, locations, and visual distractors. Can we predict that the policy will generalize well to a change in the table height by a few centimeters (due to the presence of 2D spatial variations in training data) compared to when a human is standing closer to the table than seen during training? If so, what is the absolute degradation of success rates in each case? As it turns out, the prediction above is incorrect: the success rate degrades from $\sim 65\%$ under nominal conditions to $\sim 10\%$ by changing the table height, and remains roughly constant with a human close to the table. Predicting the relative and absolute impact of other factors (e.g., lighting, distractors, etc.; Fig. 2) can be even more challenging.

In this paper, we formalize the problem of *predictive red teaming*: discovering vulnerabilities of a given policy with respect to changes in environmental factors, and predicting the (relative or absolute) degradation in performance *without* performing hardware evaluations in off-nominal scenarios. The ability to perform predictive red teaming has a number of important consequences. First, it enables *targeted deployment*: by understanding the envelope of conditions that will yield satisfactory

9th Conference on Robot Learning (CoRL 2025), Seoul, Korea.

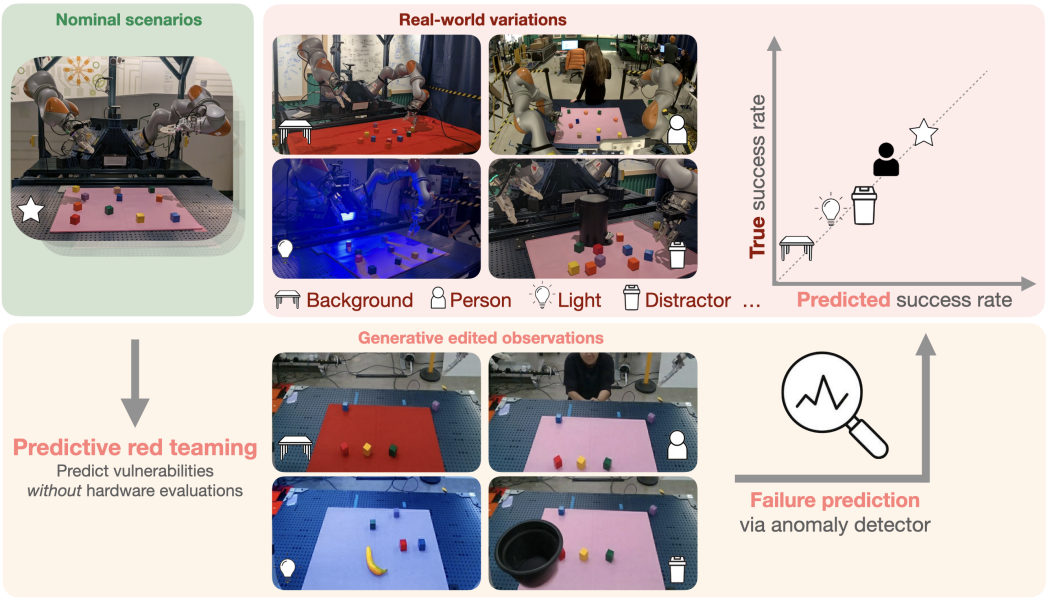


Figure 1: We consider *predictive red teaming*: discovering vulnerabilities of a policy with respect to environmental factors and predicting the corresponding performance degradation *without* hardware evaluations in off-nominal scenarios. Our approach modifies nominal observations using generative image editing to reflect changes in environmental factors (e.g., background, lighting, injecting humans and other distractors), and predicts the resulting performance degradation via anomaly detection.

performance, we can choose where the policy is deployed. Second, it enables *policy comparison*: knowing the relative vulnerabilities of different policies allows us to select one that is more likely to meet deployment needs. Third, it enables *targeted data collection*: if we know that certain environmental conditions degrade performance more than others, we can re-train the policy with additional data from the adverse conditions to patch vulnerabilities.

Contribution 1 (`RoboART`). We introduce `RoboART`— robotics automated red teaming (ART) — an approach to predictive red teaming for visuomotor policies via generative image editing and anomaly detection. In contrast to prior work on finding vulnerabilities of policies [2–4], `RoboART` does not require hardware experiments in off-nominal scenarios or the ability to simulate such scenarios (which can take months-long human effort to set up accurately in a new environment for RGB policies). The pipeline for `RoboART` has two main steps: *edit* and *predict* (Fig. 1). We use generative image editing [5–8] to modify a set of nominal observations by varying different environmental factors of interest via language instructions, and then predict the degradation in performance induced by each factor using *anomaly detection*. Specifically, we find that a simple anomaly detector that computes distances in policy embedding space between edited observations and a set of nominal observations (with an anomaly threshold computed using *conformal prediction* [9]) is surprisingly predictive of both relative and absolute performance degradation.

Contribution 2 (*Demonstration for visuomotor diffusion policies*). We evaluate `RoboART` using 500+ hardware experiments that vary 12 environmental factors for two visuomotor diffusion policies with significantly different architectures. We find that `RoboART` predicts performance degradation with a high degree of accuracy, e.g., correctly predicting that the changed table height will degrade performance significantly more than a human distractor. The difference between predicted and real success rates averaged across the twelve factors is 0.1 and 0.19 respectively for the two policies.

Contribution 3 (*Targeted data collection*). We demonstrate how predictive red teaming enables targeted data collection. Co-finetuning the policy with data from the three conditions predicted to be the most adverse boosts performance in these conditions by 2–7x. Moreover, targeted data collection also yields *cross-domain generalization*: the performance of the policy is improved by 2–5x even for conditions where we did not collect data.

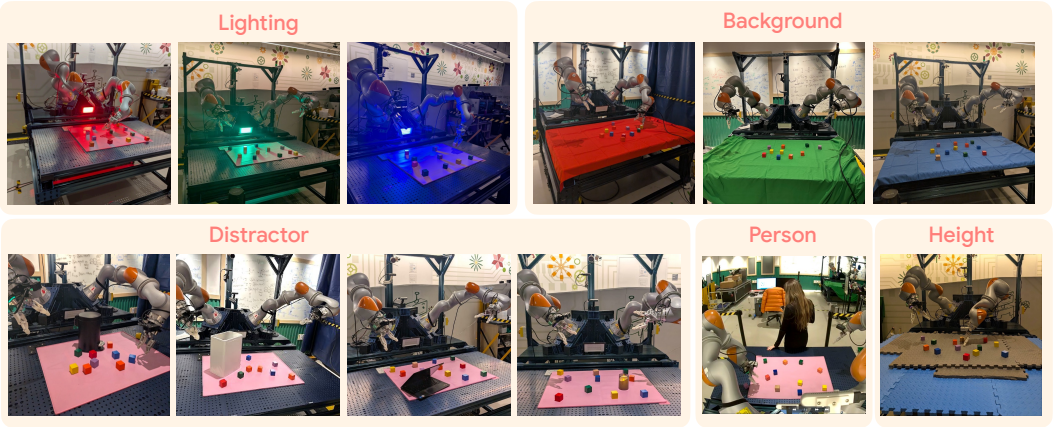


Figure 2: We evaluate RoboART's predictions using 500+ hardware trials in twelve off-nominal conditions.

2 Related Work

Red teaming. Red teaming refers to the process of finding vulnerabilities in systems, e.g., multi-modal large language models (LLMs) [10–22]. Red teaming for robotics is still nascent, and has largely focused on *scenario generation* for autonomous driving [23]. In the context of manipulation, recent work has considered finding flaws in robotic foundation models in terms of *instruction generalization* [24] in simulated environments and *jailbreaking* LLM-powered robots [25]. Unlike [24, 25], our focus is on finding environmental factors (e.g., background colors, lighting, object locations) that degrade the performance of a policy. In contrast to [2–4, 26] — which also assess generalization with respect to exogenous factors — our approach does not require hardware experiments in off-nominal scenarios or the ability to simulate such scenarios, which can take extensive (e.g., months-long) human effort to set up accurately for RGB-based policies. To our knowledge, RoboART is the first approach to predictive red teaming to rely only on access to the policy, training data, and text descriptions of desired environmental changes.

Anomaly detection and failure prediction. Methods for *failure prediction* seek to foresee failures as the robot is operating, e.g., via reachability analysis [27–29], control barrier functions [30], formal methods [31], or learned predictors [32–35]. A related line of work on *anomaly detection* seeks to detect conditions that are far from nominal and may thus induce failures [36–40]. Our approach to predictive red teaming uses conformal prediction-based anomaly detection [9, 41–45], which allows one to provide statistical assurances on the false positive rate of detection. All of the prior work mentioned above on failure prediction and anomaly detection develops methods that operate at *runtime* in order to detect possible failures and take remedial measures. To our knowledge, our work is the first to demonstrate that anomaly detection techniques can be used *offline* to accurately forecast performance in different off-nominal scenarios using generative image editing.

Generative image editing. Prior work in robotics uses generative image editing [5, 46–50] for data augmentation [51–55], generating sub-goals for image-conditioned policies [56, 57], and runtime observation editing for visual generalization [58]. In this work, we utilize a *language-conditioned* image editing model (Imagen 3 [5]) to generate image observations that reflect changes in various environmental factors (Fig. 1). By modifying real robot observations with targeted edits (e.g., "change the background to red" or "add a trash can to the scene"), we are able to generate synthetic observations with a high degree of realism.

3 Problem: Predictive Red Teaming

We formally introduce the problem of predictive red teaming: *exposing vulnerabilities of a given policy with respect to environmental factors such as lighting, visual distractors, and object locations, and predicting their impact on performance without performing any hardware evaluations in these off-nominal scenarios*.

Nominal scenarios. In each episode, the robot is deployed in a scenario ξ, which is defined as a partially observable Markov decision process (POMDP) initialized in a particular state. Let $\mathcal{D}_{\text{nom}}$ be a distribution over scenarios that captures nominal variations in all environmental factors (e.g., objects that the robot may encounter, lighting conditions, background colors, etc.) and tasks (via the reward function). We do not assume knowledge of $\mathcal{D}_{\text{nom}}$, except a dataset S_{nom} of observations collected from nominal scenarios $\xi \sim \mathcal{D}_{\text{nom}}$.

Inputs to the red team. The *red team* is provided a deterministic or stochastic policy π that maps observations $o_t \in \mathcal{O}$ to actions $a_t \in \mathcal{A}$, along with the dataset S_{nom} of nominal observations. Our focus in this paper will be on visuomotor policies trained via imitation learning; in this setting, S_{nom} can consist of observations from the training dataset for π. We also assume access to a set S_{val} of nominal observations that were held out when training π. The specific approach we present in this paper will only require nominal observations $S_{\text{nom}} \cup S_{\text{val}}$ collected at the *start* of episodes. The red team is provided the ability to query π on arbitrary observations, potentially with white-box access to internal representations of the policy.

Goal: predictive red teaming. The red team's goal is to expose vulnerabilities of π with respect to various environmental factors $f \in F$ chosen by the red team. These factors may be arbitrarily fine-grained (e.g., the introduction of a particular distractor or specific background) and may go beyond purely visual changes and require changes in actions (e.g., a change to the table height). Formally, let $\mathcal{D}_f$ be a distribution of scenarios where a factor f has changed relative to the nominal distribution $\mathcal{D}_{\text{nom}}$. Let R^{π}_{nom} be the expected reward of π for scenarios $\xi \sim \mathcal{D}_{\text{nom}}$, and let R^{π}_{f} be the expected reward for $\mathcal{D}_f$. For simplicity, we will assume henceforth that rewards are bounded in $[0, 1]$. Knowing R^{π}_{nom}, we consider two problems: (1) *rank* the factors $f \in F$ by performance degradation, and (2) predict the *absolute* performance R^{π}_{f}, $\forall f \in F$. The former problem is important for targeted data collection, while the latter helps understand the envelope of acceptable performance.

4 Predictive Red Teaming via Image Editing and Anomaly Detection

We introduce `RoboART` (**Robo**tics **A**uto-**R**ed-**T**eaming): a method for predictive red teaming using generative image editing and anomaly detection. We focus on visuomotor policies that rely on RGB observations. The two main steps of RoboART are shown in Fig. 1. First, we use generative image editing to modify the nominal observations in S_{val} to reflect changes in various factors of interest. For each factor, we then predict the performance degradation of the policy using anomaly detection.

4.1 Generative Image Editing

Selection of environmental factors. The red team first selects a set F of environmental factors that have the potential to degrade the performance of the given policy π. This set can be arbitrarily fine-grained in its contents (e.g., specific lighting conditions, distractor objects, background colors, etc.). The specific factors of interest will depend on the deployment needs of the policy and plausible environmental changes that the robot may encounter.

Generating edited observations. For each factor $f \in F$, we modify observations in the nominal set S_{val} to reflect a change in f. We leverage state-of-the-art generative image editing tools, which have the capacity to take detailed language instructions as input in order to produce realistic and globally consistent edits. In this work, we specifically utilize `Imagen 3` [5], which can generate edits with a high degree of realism and precision, e.g., changing the color of the background, adding a human in the scene, and changing lighting conditions. For robots with multiple cameras (e.g., a wrist camera in addition to an overhead camera), we edit each observation independently with the same prompt. As seen in Fig. 1, the edited images are highly realistic and maintain per-view global consistency in lighting, shadows, and overall composition of the scene (see Sec. 7 for a discussion of multi-view consistency). Full prompts along with additional examples are provided in Appendix A.

VLM critic. Diffusion-based image editing models can generate multiple edited images given the same input image and prompt. These variations often differ in terms of their quality and adherence to the prompt. In order to ensure that edited observations accurately reflect the desired change, we generate a batch of four edited images per input, and utilize a vision-language model (VLM) as a

critic. We prompt the VLM with the original and edited images, and ask it to judge if any of the options accurately reflect the desired change; if so, the VLM is tasked with choosing the best one (if not, we simply discard the observation from our set). See App. B for an example and the prompt.

4.2 Predicting Performance via Anomaly Detection

At the end of the image editing process, the red team has a set S_f of edited observations for each environmental factor $f \in F$. The second component of RoboART (Fig. 1) uses S_f to predict the performance degradation induced by each factor f. Our key idea is to utilize techniques from *anomaly detection*: for each observation in S_f, we quantify how "close" it is to nominal observations in S_{nom} from the perspective of the policy π. If this distance is above a threshold computed using *conformal prediction* [9], the observation is flagged as an anomaly. The primary hypothesis is that one can define such a policy-specific anomaly detector that predicts performance degradation: $R^\pi_f \approx 1 - \alpha^\pi_f$, where R^π_f is the expected reward under factor f (Sec. 3) and α^π_f is the anomaly rate for f, i.e., the proportion of edited observations in S_f flagged as anomalous according to a threshold chosen to ensure $R^\pi_{\text{nom}} \approx 1 - \alpha^\pi_{\text{nom}}$ (where α^π_{nom} is the proportion of nominal observations flagged as anomalous).

Anomaly detection. Next, we further describe how to compute the anomaly rate α^π_f for each factor f using the edited observations S_f. In this work, we utilize policy embedding distances as a method for quantifying how far from nominal a given observation is. This choice is motivated by the prior success of embedding-based methods in anomaly detection (see, e.g., [39, 59]) and the simplicity of implementation. Let $\phi_\pi(o)$ be a latent representation produced by the policy π for a given observation o (e.g., the output of an intermediate layer of the network). In our experiments in Sec. 5, we employ policies parameterized using diffusion models and utilize the context vector provided to the denoising process as our latent representation; see Appendix C for details. Using ϕ_π, we can define a policy-specific anomaly score $s_\pi(o, S_{\text{nom}})$ that quantifies how far from nominal the observation o is. A simple choice is to define s_π as the mean of the k-nearest neighbor (k-NN) cosine distances between the embedding $\phi_\pi(o)$ and the embeddings computed for the nominal observations in S_{nom}. Intuitively, this anomaly score quantifies how dissimilar a given observation is compared to similar training observations from the perspective of the policy.

For each factor $f \in F$, we compute the anomaly score for all edited observations $o \in S_f$. The anomaly rate for a factor f is then defined as the proportion of observations flagged as anomalous according to a threshold τ: $\alpha^\pi_f := |\{o \in S_f \mid s_\pi(o, S_{\text{nom}}) > \tau\}| / |S_f|$.

Anomaly threshold. The anomaly threshold τ is chosen to ensure that α^π_{nom} (the anomaly rate for *nominal* observations) predicts the nominal success rate R^π_{nom} of the policy: $R^\pi_{\text{nom}} \approx 1 - \alpha^\pi_{\text{nom}}$. Given access to a validation set S_{val} with n_{val} nominal observations, one can simply choose τ such that the proportion of these flagged as anomalous is $1 - R^\pi_{\text{nom}}$. A more sophisticated approach uses conformal prediction [9], which chooses τ as the $\lceil (n_{\text{val}} + 1) R^\pi_{\text{nom}} \rceil / n_{\text{val}}$ empirical quantile of the set of anomaly scores for the validation set. This choice upper bounds the probability that *unseen* nominal observations are flagged as anomalous to $1 - R^\pi_{\text{nom}}$ [60]. The use of the same threshold τ for predicting failures across different factors is justified by the fact that we are always computing k-NN distances of a given observation to the *same* set of nominal observations: if an observation is far from nominal (as quantified by τ), it is predicted to lead to failure. We summarize the key steps of RoboART in Algorithm 1.

Algorithm 1 RoboART: Robotics Auto Red Teaming

Input: Policy π with nominal performance R^π_{nom}, nominal observations $S_{\text{nom}} \cup S_{\text{val}}$
Select environmental factors F
Conformal prediction:
  Compute anomaly scores for S_{val} using π embeddings:
    $\Lambda_{\text{val}} := \{s_\pi(o, S_{\text{nom}}) \mid o \in S_{\text{val}}\}$
  Compute anomaly threshold τ using Λ_{val} to bound the nominal anomaly rate to $\alpha^\pi_{\text{nom}} := 1 - R^\pi_{\text{nom}}$
for $f \in F$ **do**
    Generate edits S_f and filter with VLM
    Compute anomaly rate:
      $\alpha^\pi_f := |\{o \in S_f \mid s_\pi(o, S_{\text{nom}}) > \tau\}| / |S_f|$
    Predict performance:
      $R^\pi_{f,\text{pred}} := 1 - \alpha^\pi_f$
end for

5 Experiments

We evaluate our framework using 500+ hardware trials that vary twelve environmental factors (Fig. 2) for two visuomotor diffusion policies with significantly different architectures. These experiments investigate: (1) How well does RoboART identify vulnerabilities and predict policy performance when relevant environmental factors are varied? (2) How effective is RoboART in enabling policy improvement via targeted data collection? (3) How good of a proxy is anomaly detection for performance degradation in different environmental conditions?

Hardware setup. Fig. 1 visualizes the hardware platform: a bimanual manipulator with two Kuka IIWA arms and Weiss grippers (the experiments only utilize the left arm). For sensing, we use a dual camera setup with a fixed overhead camera and another camera mounted on the left wrist.

Training data. We use trajectory optimization to automatically collect training data: 3K+ demonstrations for grasping objects in a broad set of nominal conditions, i.e., with fixed lighting, with a fixed pink background on a table, and an object set that consists of blocks, plush toys, small cans, and artificial fruits. Data augmentation is performed via random crops and color jittering. Additional details are in Appendix C. We highlight that the chosen task (grasping) is relatively easy to learn, and hence makes the problem of red teaming *more challenging*; trained policies demonstrate a nontrivial degree of generalization, but are also vulnerable in ways that are hard to intuit.

Policies. We consider two policies that vary significantly in their overall architecture. The first policy, π_{hyb}, uses a hybrid policy architecture inspired by [61], which combines the benefits of trajectory optimization for free-space planning with the reactive nature of closed-loop visuomotor diffusion policies [1]. The latent embedding (used by RoboART for anomaly detection) is a vector in $\mathbb{R}^{512}$ that encodes the robot's visual and proprioceptive observations, along with a keypoint selected by the robot's operator to specify the object to grasp. We also separately train a vanilla diffusion policy [1], π_{dfn}, with a latent embedding vector in $\mathbb{R}^{515\times513}$. Details of the policies are provided in Appendix C. Both policies achieve approximately 65% success for nominal conditions, as measured by 30 trials (each) that vary objects, their locations, and the target object.

Environmental factors. We choose a set F of twelve environmental factors that reflect common vulnerabilities of visuomotor policies trained via behavior cloning. These are shown in Fig. 2, and include: (1–3) three changes to the **lighting** conditions (red, green, blue), (4–6) three changes to the color (red, green, blue) of the table **background** on which objects are placed, (7–10) four **distractor** objects (black and white trash can, laptop, candle) that partially occlude other objects, (11) a distractor in the form of a **person** close to the table, and (12) a change to the **height** of the table (which changes the location of objects relative to the overhead camera). We highlight that these factors include both purely visual changes and ones that impact the robot's actions (e.g., moving around the large distractors or adjusting to a different table height). We evaluate predictions from RoboART by executing both policies in 20+ episodes for each factor; this allows us to estimate the success rates $R_f^{\pi_{\text{hyb}}}$ and $R_f^{\pi_{\text{dfn}}}, \forall f \in F$. The subsequent results thus include 500+ hardware trials.

5.1 How accurately does RoboART identify vulnerabilities and predict policy performance?

We first evaluate how well RoboART predicts the performance degradation induced by each of the twelve environmental factors for the different policies. We utilize two metrics to evaluate RoboART, which correspond to the two versions of predictive red teaming described in Sec. 3: (1) **Spearman rank correlation** [62]: which is a value $\rho \in [-1, 1]$ that measures how accurately RoboART *ranks* the different factors by performance degradation, and (2) **Average prediction error**: which measures how accurately RoboART predicts the *absolute success rates* for the different factors by computing $\frac{1}{|F|}\sum_{f\in F}|R_f^{\pi} - R_{f,\text{pred}}^{\pi}|$.

Implementation. In order to make predictions using RoboART, we generate a set S_f of 100 edited observations for each factor f using first time-step observations from a held-out portion of training episodes. Examples of edits and complete prompts are provided in Fig. 1 and Appendix A. We compute the resulting anomaly rates α_f^{π} using S_f for each policy as described in Alg. 1. We take the anomaly score $s_{\pi}(o, S_{\text{nom}})$ to be the mean of the k-nearest neighbor cosine distances (in the

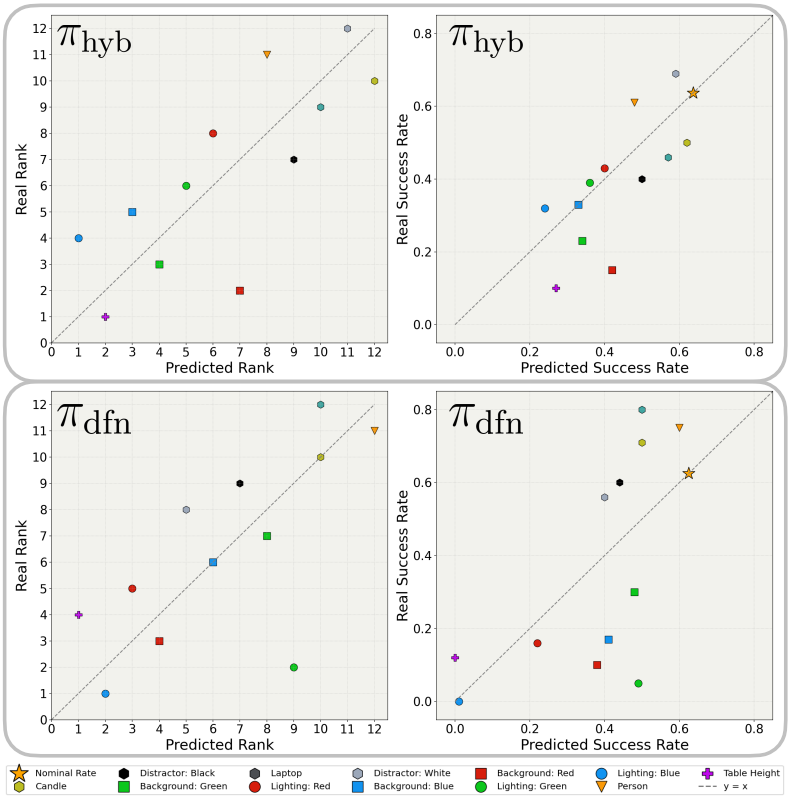


Figure 3: Evaluating predictions from RoboART for π_{hyb} (top panel) and π_{dfn} (bottom panel). Left: Comparison of true (estimated) rankings of different environmental factors by performance degradation with predictions made by RoboART. Right: Comparison of true (estimated) success rates with predictions from RoboART.

respective policy embedding space) to a set S_{nom}, which is chosen to be a subset of first-time-step observations from the training episodes.

Results. Fig. 3 evaluates predictions made by RoboART for π_{hyb} (top row) and π_{dfn} (bottom row). Fig. 3-left compares the rankings of different environmental factors $f \in F$ predicted by RoboART with the true rankings as measured by the 20+ hardware evaluations for each factor (a lower rank corresponds to a lower success rate). Fig. 3-right compares the absolute success rates predicted by RoboART with the true (estimated) success rates. As the figure illustrates, the predictions for both rankings and absolute performance are strongly correlated with the true rankings and success rates. For example, RoboART successfully predicts that the performance of π_{hyb} is relatively robust to object or person distractors, moderately degraded by background and lighting changes, and strongly degraded by changing the table height. RoboART also successfully predicts that π_{dfn} is more vulnerable than π_{hyb} to certain factors such as blue lighting and a change in the table height.

Table 1 evaluates the predictions made by RoboART for both policies. The Spearman ρ indicates a strong correlation between predicted and actual rankings of factors, while the average prediction error is under 0.19 for both policies (roughly in the range of noise when estimating success rates from ~ 20 trials).

RoboART	π_{hyb}	π_{dfn}
Spearman $\rho \in [-1, 1]$	0.8 (↑)	0.7 (↑)
Av. prediction error $\in [0, 1]$	0.10 (↓)	0.19 (↓)

Table 1: Quantitative evaluation of success rates predicted by RoboART compared with real success rates.

Ablations. For the results above, we use $k = 5, |S_{\text{nom}}| = 3000$ for π_{hyb} and $k = 10, |S_{\text{nom}}| = 500$ for π_{dfn}. We provide results from ablating the values k and $|S_{\text{nom}}|$ in Appendix D. Generally, we find that predictions for π_{hyb} remain accurate when varying $|S_{\text{nom}}|$ with small k, while predictions for π_{dfn} (which has a significantly higher dimensional embedding space) benefit from either having a smaller value of $|S_{\text{nom}}|$ or larger values of k.

5.2 How effective is RoboART in enabling policy improvement via targeted data collection?

Next, we evaluate how well RoboART enables *policy improvement* via targeted data collection. We collect around 1hr of training data ($\approx$ 100 trajectories) for π_{hyb} with the three environmental factors that RoboART predicts the highest performance degradation for: blue lighting, change in the table height, and blue table background. We then co-finetune our initial policy π_{hyb} on the nominal data combined with the small amount of new off-nominal data. During co-finetuning, each mini-batch consists of 80% of the original data mixture and 20% from the new off-nominal data.

The fine-tuned policy $\pi^{\text{ft}}_{\text{hyb}}$ is evaluated in nominal conditions, the three conditions for which we collected data, and also the other background and lighting conditions. Videos of $\pi^{\text{ft}}_{\text{hyb}}$ are available in the supplement. Fig. 4 shows improved performance in nominal conditions and a 2–7x improvement in off-nominal conditions under which training data was collected. Significantly, the targeted data collection also yields *cross-domain generalization*: the policy improves by 2–5x even for background and lighting conditions where we did not collect additional data. This highlights the benefits of targeting data collection towards adverse scenarios via predictive red teaming.

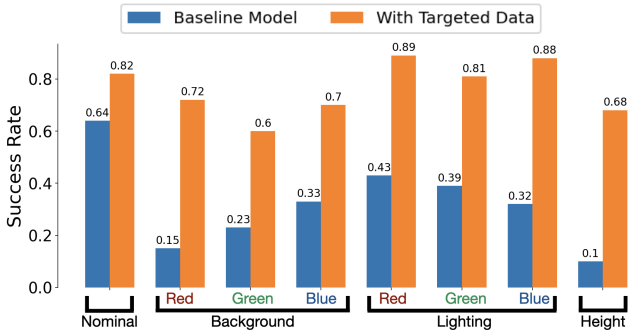


Figure 4: Fine-tuning with data collected under conditions predicted to be adverse shows cross-domain generalization and boosts baseline performance by 2–7x.

5.3 How accurately does anomaly detection predict performance degradation?

Our final set of experiments evaluate the anomaly detection component of RoboART in isolation from the image editing pipeline. To this end, instead of executing the embedding-based anomaly detector on the set S_f of *edited* observations, we execute the detector on the set S^{real}_f composed of *real* robot observations collected from the first time step of the 20+ episodes where the factor f is varied. We then compute the corresponding anomaly rates $\alpha^{\pi}_{f,\text{real}}$ ($\forall f \in F$). Predicted success rates for each factor are computed as $R^{\pi}_{f,\text{anom}} := 1 - \alpha^{\pi}_{f,\text{real}}$ and compared with the (estimated) true success rates. Additional implementation details are provided in Appendix E.

Table 2 presents the Spearman ρ and average prediction errors. Appendix E presents figures analogous to Fig. 3. While we observe high values of ρ and low values of the prediction error for both policies, we note that the predictions $R^{\pi}_{f,\text{anom}}$ are made using $5\times$ fewer observations than predictions from the full RoboART pipeline ($\sim$ 20 real observations vs. 100 edited observations), thus making them significantly more susceptible to noise.

Anomaly detector	π_{hyb}	π_{dfn}
Spearman $\rho \in [-1, 1]$	0.6 (↑)	0.8 (↑)
Av. prediction error $\in [0, 1]$	0.20 (↓)	0.21 (↓)

Table 2: Evaluating predictions of success rates made from anomaly rates computed using real observations.

6 Conclusions

We have considered the problem of predictive red teaming: given access to observations from nominal scenarios, discovering vulnerabilities of a policy with respect to unseen changes in environmental factors and predicting the resulting performance degradation. Our approach — Robotics Auto Red Teaming (RoboART) — modifies nominal observations via generative image editing to reflect changes in environmental factors, and then uses a policy embedding-based anomaly detector to predict performance. Experiments across 500+ trials for visuomotor policies demonstrate RoboART's ability to (i) identify factors that significantly impact performance, (ii) predict the relative and absolute impact of factors, and (iii) enable policy improvement via targeted data collection.

7 Limitations and Future Work

We discuss limitations of our approach and promising future directions that may address them.

Edit-to-real gap. While state-of-the-art image editing tools are capable of producing realistic edits (especially with careful prompting), there are still gaps in realism for certain environmental factors. For example, edits that reflect lighting changes (Fig. 1) do not modify the shadows of objects as real lighting changes do. We expect that our method will benefit from the significant investments in improving image editing models for commercial applications. Beyond single-view realism, a more challenging limitation is ensuring *multi-view* consistency. As seen in Appendix A, edited observations from the overhead and wrist cameras do not represent a consistent geometry for the new object. One exciting possibility is to utilize recent *3D scene editing* tools based on Gaussian Splatting [63, 64] that allow for edits with multi-view consistency. Scene editing may also allow us to go beyond RGB observations and edit depth channels.

Anomaly-to-failure gap. Our approach utilizes the anomaly rate as a predictor for performance degradation. However, as seen in Sec. 5.3, these predictions are not perfectly accurate. One avenue for future work is to perform edits on observations from multiple time-steps within each episode, and to compute anomaly rates based on these sequences (rather than only utilizing the first time-step from episodes, as we currently do). We are also interested in exploring other methods from the vast literature on anomaly detection to identify techniques that may serve as better predictors of performance degradation (see, e.g., [65] for a recent empirical study).

Hidden environmental factors. One limitation of `RoboART` is that it requires changes in environmental factors to be reflected in visual observations of the robot. As such, `RoboART` will not identify vulnerabilities with changes that are completely hidden (e.g., changing the mass of objects without changing their visual appearance). In such cases, predictive red teaming via simulation is a promising avenue. We note that our method is *not* meant to provide more accurate predictions compared to a carefully constructed simulator that faithfully simulates policy execution under a range of conditions. Instead, our work is motivated by the challenge of constructing a faithful simulator, which can take months-long effort for RGB-based policies, e.g., by matching visual backdrops via green-screening and matching dynamics via extensive system-id [3]. In contrast, our method does not require any raw materials other than a set of nominal observations and a generative image editing tool. To our knowledge, our work is the first to provide such a light-weight method for predictive red teaming.

Multi-round predictive red teaming. `RoboART` currently chooses a single set F of environmental factors at the beginning of predictive red teaming. A more sophisticated approach could *iteratively* explore the space of environmental factors: choose an initial set F, make predictions for these, and expand the set iteratively to include factors that are similar to ones that were predicted to yield poor performance.

As we seek to deploy robots in challenging real-world applications, it is essential that we develop scalable methods for predicting the limits of their performance. We hope that formalizing the problem of predictive red teaming and providing a baseline in the form of `RoboART` spurs further research along this underexplored direction.

Acknowledgements

We are grateful to Shixin Luo and Suraj Kothawade for extremely helpful pointers on the image editing pipeline. We thank Alex Irpan for insightful feedback on an early version of the paper. We are grateful to Dave Orr, Anca Dragan, and Vincent Vanhoucke for continuous guidance on safety-related topics; and Frankie Garcia and Dawn Bloxwich on responsible development.

References

[1] C. Chi, Z. Xu, S. Feng, E. Cousineau, Y. Du, B. Burchfiel, R. Tedrake, and S. Song. Diffusion policy: Visuomotor policy learning via action diffusion. *The International Journal of Robotics Research*, 2023.

[2] W. Pumacay, I. Singh, J. Duan, R. Krishna, J. Thomason, and D. Fox. THE COLOSSEUM: A benchmark for evaluating generalization for robotic manipulation. 2024.

[3] X. Li, K. Hsu, J. Gu, K. Pertsch, O. Mees, H. R. Walke, C. Fu, I. Lunawat, I. Sieh, S. Kirmani, et al. Evaluating real-world robot manipulation policies in simulation. *arXiv preprint arXiv:2405.05941*, 2024.

[4] S. Sagar, J. Duan, S. Vasudevan, Y. Zhou, H. B. Amor, D. Fox, and R. Senanayake. From mystery to mastery: Failure diagnosis for improving manipulation policies. 2025.

[5] J. Baldridge, J. Bauer, M. Bhutani, N. Brichtova, A. Bunner, K. Chan, Y. Chen, S. Dieleman, Y. Du, Z. Eaton-Rosen, et al. Imagen 3. *arXiv preprint arXiv:2408.07009*, 2024.

[6] P. Esser, S. Kulal, A. Blattmann, R. Entezari, J. Müller, H. Saini, Y. Levi, D. Lorenz, A. Sauer, F. Boesel, et al. Scaling rectified flow transformers for high-resolution image synthesis. In *Proceedings of the International Conference on Machine Learning*, 2024.

[7] L. Zhang, A. Rao, and M. Agrawala. Adding conditional control to text-to-image diffusion models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3836–3847, 2023.

[8] C. Saharia, W. Chan, S. Saxena, L. Li, J. Whang, E. L. Denton, K. Ghasemipour, R. Gontijo Lopes, B. Karagol Ayan, T. Salimans, et al. Photorealistic text-to-image diffusion models with deep language understanding. *Advances in neural information processing systems*, 35: 36479–36494, 2022.

[9] V. Vovk, A. Gammerman, and G. Shafer. *Algorithmic Learning in a Random World*, volume 29. Springer, 2005.

[10] J. Achiam, S. Adler, S. Agarwal, L. Ahmad, I. Akkaya, F. L. Aleman, D. Almeida, J. Altenschmidt, S. Altman, S. Anadkat, et al. GPT-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

[11] G. Team, R. Anil, S. Borgeaud, J.-B. Alayrac, J. Yu, R. Soricut, J. Schalkwyk, A. M. Dai, A. Hauth, K. Millican, et al. Gemini: a family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*, 2023.

[12] D. Ganguli, L. Lovitt, J. Kernion, A. Askell, Y. Bai, S. Kadavath, B. Mann, E. Perez, N. Schiefer, K. Ndousse, et al. Red teaming language models to reduce harms: Methods, scaling behaviors, and lessons learned. *arXiv preprint arXiv:2209.07858*, 2022.

[13] Y. Bai, A. Jones, K. Ndousse, A. Askell, A. Chen, N. DasSarma, D. Drain, S. Fort, D. Ganguli, T. Henighan, et al. Training a helpful and harmless assistant with reinforcement learning from human feedback. *arXiv preprint arXiv:2204.05862*, 2022.

[14] A. Wei, N. Haghtalab, and J. Steinhardt. Jailbroken: How does LLM safety training fail? *Advances in Neural Information Processing Systems*, 36, 2024.

[15] E. Perez, S. Huang, F. Song, T. Cai, R. Ring, J. Aslanides, A. Glaese, N. McAleese, and G. Irving. Red teaming language models with language models. *arXiv preprint arXiv:2202.03286*, 2022.

[16] S. Tong, E. Jones, and J. Steinhardt. Mass-producing failures of multimodal systems with language models. *Advances in Neural Information Processing Systems*, 36, 2024.

[17] A. Zou, Z. Wang, N. Carlini, M. Nasr, J. Z. Kolter, and M. Fredrikson. Universal and transferable adversarial attacks on aligned language models. *arXiv preprint arXiv:2307.15043*, 2023.

[18] P. Chao, A. Robey, E. Dobriban, H. Hassani, G. J. Pappas, and E. Wong. Jailbreaking black box large language models in twenty queries. *arXiv preprint arXiv:2310.08419*, 2023.

[19] X. Liu, N. Xu, M. Chen, and C. Xiao. AutoDAN: Generating stealthy jailbreak prompts on aligned large language models. *arXiv preprint arXiv:2310.04451*, 2023.

[20] A. Mehrotra, M. Zampetakis, P. Kassianik, B. Nelson, H. Anderson, Y. Singer, and A. Karbasi. Tree of attacks: Jailbreaking black-box LLMs automatically. *arXiv preprint arXiv:2312.02119*, 2023.

[21] M. Li, L. Li, Y. Yin, M. Ahmed, Z. Liu, and Q. Liu. Red teaming visual language models. *arXiv preprint arXiv:2401.12915*, 2024.

[22] Y. Liu, C. Cai, X. Zhang, X. Yuan, and C. Wang. Arondight: Red teaming large vision language models with auto-generated multi-modal jailbreak prompts. In *Proceedings of the 32nd ACM International Conference on Multimedia*, pages 3578–3586, 2024.

[23] W. Ding, C. Xu, M. Arief, H. Lin, B. Li, and D. Zhao. A survey on safety-critical driving scenario generation—a methodological perspective. *IEEE Transactions on Intelligent Transportation Systems*, 24(7):6971–6988, 2023.

[24] S. Karnik, Z.-W. Hong, N. Abhangi, Y.-C. Lin, T.-H. Wang, and P. Agrawal. Embodied red teaming for auditing robotic foundation models. *arXiv preprint arXiv:2411.18676*, 2024.

[25] A. Robey, Z. Ravichandran, V. Kumar, H. Hassani, and G. J. Pappas. Jailbreaking LLM-controlled robots. *arXiv preprint arXiv:2410.13691*, 2024.

[26] J. Z.-Y. He, D. S. Brown, Z. Erickson, and A. Dragan. Quantifying assistive robustness via the natural-adversarial frontier. In *Conference on Robot Learning*, pages 1865–1886. PMLR, 2023.

[27] A. K. Akametalu, J. F. Fisac, J. H. Gillula, S. Kaynama, M. N. Zeilinger, and C. J. Tomlin. Reachability-based safe learning with gaussian processes. In *Proceedings of the 53rd IEEE Conference on Decision and Control*, pages 1424–1431, 2014.

[28] K.-C. Hsu, A. Z. Ren, D. P. Nguyen, A. Majumdar, and J. F. Fisac. Sim-to-lab-to-real: Safe reinforcement learning with shielding and generalization guarantees. *Artificial Intelligence*, 314:103811, 2023.

[29] K.-C. Hsu, H. Hu, and J. F. Fisac. The safety filter: A unified view of safety-critical control in autonomous systems. *Annual Review of Control, Robotics, and Autonomous Systems*, 7, 2023.

[30] A. D. Ames, X. Xu, J. W. Grizzle, and P. Tabuada. Control barrier function based quadratic programs for safety critical systems. *IEEE Transactions on Automatic Control*, 62(8):3861–3876, 2016.

[31] M. Alshiekh, R. Bloem, R. Ehlers, B. Könighofer, S. Niekum, and U. Topcu. Safe reinforcement learning via shielding. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018.

[32] A. Farid, D. Snyder, A. Z. Ren, and A. Majumdar. Failure prediction with statistical guarantees for vision-based robot control. In *Proceedings of Robotics: Science and Systems (RSS)*, 2022.

[33] A. Xie, F. Tajwar, A. Sharma, and C. Finn. When to ask for help: Proactive interventions in autonomous reinforcement learning. *Advances in Neural Information Processing Systems*, 35: 16918–16930, 2022.

[34] C. Gokmen, D. Ho, and M. Khansari. Asking for help: Failure prediction in behavioral cloning through value approximation. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 5821–5828, 2023.

[35] H. Liu, S. Dass, R. Martín-Martín, and Y. Zhu. Model-based runtime monitoring with interactive imitation learning. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 4154–4161, 2024.

[36] C. Richter and N. Roy. Safe visual navigation via deep learning and novelty detection. In *Proceedings of Robotics: Science and Systems (RSS)*, 2017.

[37] R. Sinha, A. Sharma, S. Banerjee, T. Lew, R. Luo, S. M. Richards, Y. Sun, E. Schmerling, and M. Pavone. A system-level view on out-of-distribution data in robotics. *arXiv preprint arXiv:2212.14020*, 2022.

[38] M. Salehi, H. Mirzaei, D. Hendrycks, Y. Li, M. H. Rohban, and M. Sabokrou. A unified survey on anomaly, novelty, open-set, and out-of-distribution detection: Solutions and future challenges. *arXiv preprint arXiv:2110.14051*, 2021.

[39] R. Sinha, A. Elhafsi, C. Agia, M. Foutter, E. Schmerling, and M. Pavone. Real-time anomaly detection and reactive planning with large language models. *arXiv preprint arXiv:2407.08735*, 2024.

[40] V. Sindhwani, H. Sidahmed, K. Choromanski, and B. Jones. Unsupervised anomaly detection for self-flying delivery drones. In *2020 IEEE international conference on robotics and automation (ICRA)*, pages 186–192. IEEE, 2020.

[41] R. Laxhammar and G. Falkman. Online learning and sequential anomaly detection in trajectories. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(6):1158–1173, 2013.

[42] R. Luo, S. Zhao, J. Kuck, B. Ivanovic, S. Savarese, E. Schmerling, and M. Pavone. Sample-efficient safety assurances using conformal prediction. *The International Journal of Robotics Research*, 43(9):1409–1424, 2024.

[43] R. Sinha, E. Schmerling, and M. Pavone. Closing the loop on runtime monitors with fallback-safe MPC. In *2023 62nd IEEE Conference on Decision and Control (CDC)*, pages 6533–6540. IEEE, 2023.

[44] H. Huang, S. Sharma, A. Loquercio, A. Angelopoulos, K. Goldberg, and J. Malik. Conformal policy learning for sensorimotor control under distribution shifts. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 16285–16291. IEEE, 2024.

[45] M. Zhao, R. Simmons, H. Admoni, A. Ramdas, and A. Bajcsy. Conformalized interactive imitation learning: Handling expert shift and intermittent feedback. *arXiv preprint arXiv:2410.08852*, 2024.

[46] J. Betker, G. Goh, L. Jing, T. Brooks, J. Wang, L. Li, L. Ouyang, J. Zhuang, J. Lee, Y. Guo, et al. Improving image generation with better captions. *OpenAI*, 2023.

[47] A. Nichol, P. Dhariwal, A. Ramesh, P. Shyam, P. Mishkin, B. McGrew, I. Sutskever, and M. Chen. Glide: Towards photorealistic image generation and editing with text-guided diffusion models. *arXiv preprint arXiv:2112.10741*, 2021.

[48] T. Yu, R. Feng, R. Feng, J. Liu, X. Jin, W. Zeng, and Z. Chen. Inpaint anything: Segment anything meets image inpainting. *arXiv preprint arXiv:2304.06790*, 2023.

[49] H. Ling, K. Kreis, D. Li, S. W. Kim, A. Torralba, and S. Fidler. Editgan: High-precision semantic image editing. *Advances in Neural Information Processing Systems*, 34:16331–16345, 2021.

[50] J. Zhu, Y. Shen, D. Zhao, and B. Zhou. In-domain GAN inversion for real image editing. In *European conference on computer vision*, pages 592–608. Springer, 2020.

[51] Z. Chen, S. Kiami, A. Gupta, and V. Kumar. Genaug: Retargeting behaviors to unseen situations via generative augmentation. *arXiv preprint arXiv:2302.06671*, 2023.

[52] T. Yu, T. Xiao, A. Stone, J. Tompson, A. Brohan, S. Wang, J. Singh, C. Tan, J. Peralta, B. Ichter, et al. Scaling robot learning with semantically imagined experience. *arXiv preprint arXiv:2302.11550*, 2023.

[53] H. Bharadhwaj, J. Vakil, M. Sharma, A. Gupta, S. Tulsiani, and V. Kumar. Roboagent: Generalization and efficiency in robot manipulation via semantic augmentations and action chunking. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 4788–4795, 2024.

[54] L. Y. Chen, C. Xu, K. Dharmarajan, M. Z. Irshad, R. Cheng, K. Keutzer, M. Tomizuka, Q. Vuong, and K. Goldberg. RoVi-Aug: Robot and viewpoint augmentation for cross-embodiment robot learning. *arXiv preprint arXiv:2409.03403*, 2024.

[55] Z. Chen, Z. Mandi, H. Bharadhwaj, M. Sharma, S. Song, A. Gupta, and V. Kumar. Semantically controllable augmentations for generalizable robot learning. *The International Journal of Robotics Research*, page 02783649241273686, 2024.

[56] K. Black, M. Nakamoto, P. Atreya, H. Walke, C. Finn, A. Kumar, and S. Levine. Zero-shot robotic manipulation with pretrained image-editing diffusion models. *arXiv preprint arXiv:2310.10639*, 2023.

[57] D. Shah, A. Sridhar, N. Dashora, K. Stachowicz, K. Black, N. Hirose, and S. Levine. Vint: A foundation model for visual navigation. *arXiv preprint arXiv:2306.14846*, 2023.

[58] A. J. Hancock, A. Z. Ren, and A. Majumdar. Run-time observation interventions make vision-language-action models more visually robust. *arXiv preprint arXiv:2410.01971*, 2024.

[59] R. Luo, R. Sinha, Y. Sun, A. Hindy, S. Zhao, S. Savarese, E. Schmerling, and M. Pavone. Online distribution shift detection via recency prediction. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 16251–16263. IEEE, 2024.

[60] A. N. Angelopoulos and S. Bates. A gentle introduction to conformal prediction and distribution-free uncertainty quantification. *arXiv preprint arXiv:2107.07511*, 2021.

[61] S. Belkhale, Y. Cui, and D. Sadigh. Hydra: Hybrid robot actions for imitation learning. In *Proceedings of the Conference on Robot Learning*, pages 2113–2133, 2023.

[62] J. H. Zar. Spearman rank correlation. *Encyclopedia of Biostatistics*, 7, 2005.

[63] J.-K. Chen and Y.-X. Wang. Proedit: Simple progression is all you need for high-quality 3D scene editing. *arXiv preprint arXiv:2411.05006*, 2024.

[64] Y. Bao, T. Ding, J. Huo, Y. Liu, Y. Li, W. Li, Y. Gao, and J. Luo. 3d Gaussian splatting: Survey, technologies, challenges, and opportunities. *arXiv preprint arXiv:2407.17418*, 2024.

[65] C. Xu, T. K. Nguyen, P. Miller, R. Lee, P. Shah, R. A. Ambrus, H. Nishimura, and M. Itkina. Uncertainty-aware failure detection for imitation learning robot policies. In *CoRL Workshop on Safe and Robust Robot Learning for Operation in the Real World*.

[66] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

[67] M. Ryoo, A. Piergiovanni, A. Arnab, M. Dehghani, and A. Angelova. Tokenlearner: Adaptive space-time tokenization for videos. *Advances in neural information processing systems*, 34: 12786–12797, 2021.

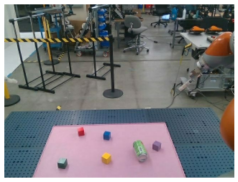
[68] W. Peebles and S. Xie. Scalable diffusion models with transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4195–4205, 2023.

[69] M. Reuss, M. Li, X. Jia, and R. Lioutikov. Goal-conditioned imitation learning using score-based diffusion policies. *arXiv preprint arXiv:2304.02532*, 2023.

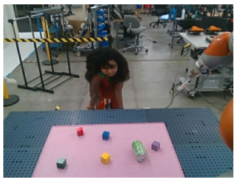
[70] T. Karras, M. Aittala, T. Aila, and S. Laine. Elucidating the design space of diffusion-based generative models. *Advances in neural information processing systems*, 35:26565–26577, 2022.

A Image Editing: Examples and Prompts

Examples of different edits applied to both the overhead camera and the wrist camera are shown in Figure 5. Below, we provide complete prompts used to generate the edited observations for each environmental condition.

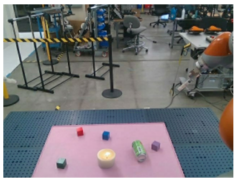


(a) Nominal overhead (left) and wrist (right) cameras.



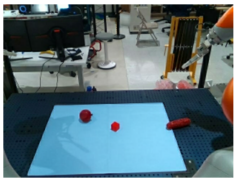
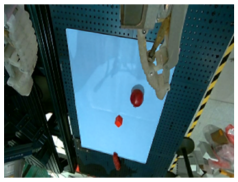
(b) Distractor: person.

(c) Distractor: trash can.

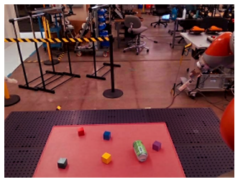


(d) Distractor: candle.

(e) Distractor: laptop.

(f) Background: blue.



(g) Lighting: red.

(h) Table height (changed color followed by zoom).

Figure 5: Examples of different edits applied to nominal overhead and wrist camera observations.

Full prompts for edits

Person:
Add a person to the image. Specifically, add a person behind the blue platform, realistically interacting with the platform and fitting seamlessly into the existing environment. Preserve all other aspects of the image, including the different objects on the mat, other background elements, and the overall composition. The lighting should remain consistent. The new person should be realistically rendered with all details of the person including their face, clothing, and any other visible parts shown in exquisite clarity and detail. Only the person should be added.
Large distractor (e.g., trash cans):
Add a large <target color> <target object> at the edge of the pink mat, so that it doesn't modify or occlude any of the objects on the pink mat. Specifically, add a <target color> <target object> that is larger than any objects at the edge of the pink mat, fitting realistically and seamlessly into the existing scene. Preserve all details of the objects on the mat, their poses, and the overall composition of the image. The <target color> <target object> should be realistically and exquisitely rendered and should not occlude any of the objects on the pink mat. The lighting should remain consistent. Only the <target color> <target object> at the edge of the pink mat should be added.
Small distractor (e.g., candle):
Modify image <image> as described below: Add a small scented candle on the pink mat, so that it doesn't modify or occlude any of the objects on the mat. Specifically, add a scented candle with roughly the same size as the objects on the pink mat, fitting realistically and seamlessly into the existing scene. Preserve all details of the composition of the image. The scented candle should be realistically and exquisitely rendered and should not occlude any of the objects on the pink mat. The lighting should remain consistent. Only the scented candle should be added.
Background color:
Modify image <image> as described below: change the color of the pink mat that objects are on to <target color>. Preserve the different objects on the mat, and all other aspects of the image including the lighting and the overall composition.
Lighting (overhead camera):
Modify image <image> as described below: Colorize the bottom half of the image with an extremely intense <target color> hue. Preserve the existing composition, details, and textures of the objects in the scene, including the ones on the pink mat and the background. Only the shadows and color palette should be altered to reflect an extremely intense <target color> light, maintaining the style of the original image. The overall lighting should remain consistent, with shadows and highlights adjusted to match the new color palette. Make sure that the hue for the bottom half of the image is changed to intense <target color>, including for the objects on the table.
Lighting (wrist camera):
Modify image <image> as described below: Colorize the entire image with an extremely intense <target color> color tone. Preserve the existing composition, details, and textures of the objects in the scene, including the ones on the pink mat and the background. Only the shadows and color palette should be altered to reflect an extremely intense <target color> light, maintaining the style of the original image. The overall lighting should remain consistent, with shadows and highlights adjusted to match the new color palette. Make sure that the color for the entire image is changed to intense <target color>.
Table height:
Change the color of the pink mat to <target color>. Preserve all other aspects of the image, including the different objects on the mat, the lighting, and the overall composition. Only the color of the pink mat should be altered to <target color>, maintaining its shape, size, and position. [We then apply a zoom to the portion of the image that contains the table in order to simulate a change in the height of the table.

B Filtering Edits with a Vision-Language Model

For each nominal observation, we generate a batch of four candidate edited observations via the image editing model. We then use a vision-language model (VLM) in order to judge if any of the

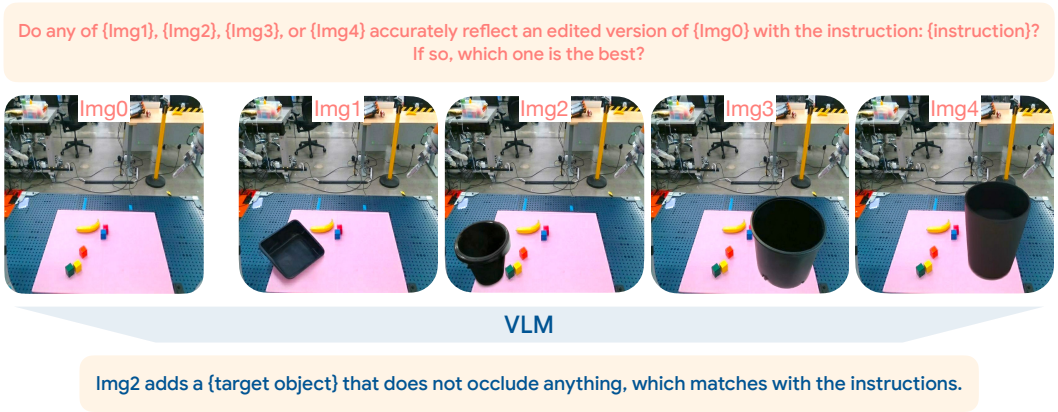


Figure 6: A vision-language model ensures that the edited image reflects the desired change.

options accurately reflect the desired change; if so, the VLM is tasked with choosing the best one (if not, we simply discard the observation from our set). An example is shown in Fig. 6. The full prompt for the VLM — which involves chain-of-thought reasoning — is provided below. We use the Gemini Pro 1.5 VLM [11] for our experiments.

Prompt for filtering edits with a VLM

Here is the original image I have: <`original image`>. Do any of Image 0: <`Image 0`>, Image 1: <`Image 1`>, Image 2: <`Image 2`>, or Image 3 <`Image 3`> accurately reflect an edited version of the original image with the instruction "<`short edit instruction`>"? Give your reasoning and then answer with a True or False. If True, provide the index (0,1,2,3) of the best image.

The variable <`short edit instruction`> contains a shortened version (e.g., "Change the color of the pink mat to <`target color`>") of the full prompt provided to the image editing model. We find that providing the full prompt (instead of a shortened version) can lead the VLM to be overly critical and filter out many acceptable edits.

C Training and Policy Details

C.1 Training Data Collection

Training data collection. For training our policies, we collect 3K+ demonstrations for grasping tasks on the hardware. Specifically, we use trajectory optimization-based motion planners to automatically collect a large set of training data. Our data collection pipeline uses the overhead camera to obtain a 3D point cloud of the scene. We segment the point cloud into multiple objects and randomly choose different objects to pick using the left arm. We use automated success detection to segment these trajectories. For each episode, we further automatically annotate keypoints for the object the policy should grasp; these are used as additional context for the robot policy in addition to camera and proprioceptive observations. All demonstrations are collected in nominal conditions, i.e., with fixed lighting, with a fixed pink background on a table, and an object set that consists of blocks, plush toys, small cans, and artificial fruits.

C.2 Hybrid Policy Architecture

Hybrid policy. We consider two policies that vary significantly in their overall architecture. The first policy π_{hyb} uses a hybrid policy architecture inspired by [61], which aims to utilize the benefits of trajectory optimization-based approaches for free space planning together with the reactive nature of closed-loop visuomotor diffusion policies. We achieve this by using two separate heads in our

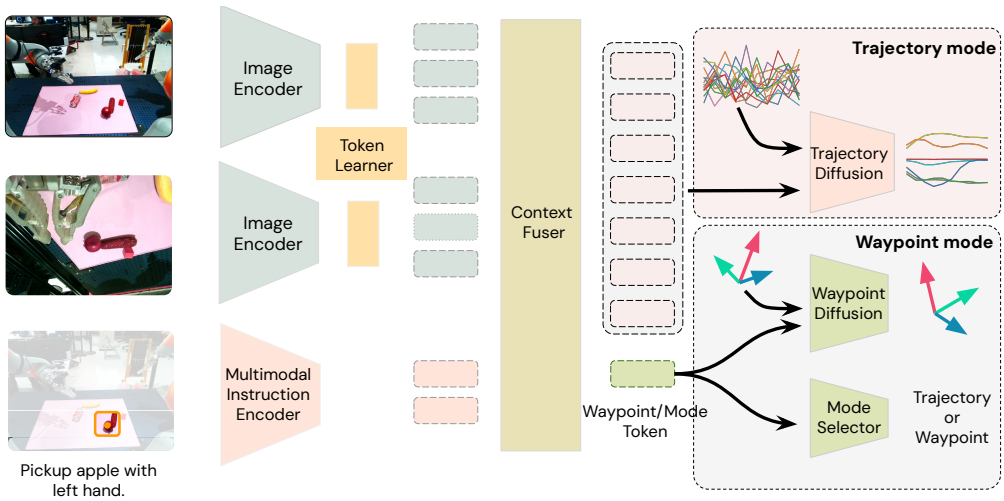


Figure 7: The policy architecture used for two different — hybrid and diffusion — policy implementation. Our unified architecture consists of a trajectory mode which predicts the continuous joint space actions and a waypoint mode which predicts a single SE(3) waypoint that the arm should reach to. We use two different policies. 1. *hybrid policy* uses both the trajectory and waypoint mode and selects between them to execute the action. 2. *diffusion policy* only uses the trajectory mode and directly predicts the joint space trajectory for the robot to follow.

policy architecture (see Fig. 7), where each head represents an *action mode*. These two different action modes correspond to:

1. a waypoint action mode which outputs a single waypoint ($w \in SE(3)$), and
2. a trajectory action mode which outputs a dense sequence of robot joint angles ($q_i \in \mathbb{R}^{14}$).

In addition to these policy heads we also output a mode selection scalar which defines which action mode should be executed at any given time. In order to execute the waypoint action we use a trajectory optimization approach based on sequential quadratic programming (SQP), and execute the output trajectory for a fixed number of steps before re-querying the policy. By contrast, in order to execute the trajectory action we simply interpolate through the joint commands outputted by the network. Importantly, during training both policy heads are trained *simultaneously*, i.e., each input data item is labelled with a waypoint action (extracted using an object closeness heuristic) and a dense trajectory action (which we directly extract from the robot logs). We supervise the mode selection scalar to output the waypoint action mode when the arm is far away from any object and the trajectory action mode in all other scenarios.

Vision encoder. Our policy architecture uses pre-trained ViT [66] encoders to encode the image observations from each image. We use separate models for each camera observation (overhead and wrist). We reduce the number of tokens from each ViT using a TokenLearner layer [67]. We encode proprioceptive features using a multi-layer perceptron (MLP) with a single hidden layer.

Instruction encoder. The robot is instructed to grasp a target object using semantic keypoints. Specifically, we extract a small patch (64×64) from the overhead camera view around a keypoint that is selected by the robot operator. We encode this patch using a small coordinate convolution-based neural network. Since we train a multi-skill policy we encode the skill that the robot needs to perform using a continuous embedding. The semantic keypoint representation is concatenated with the skill embedding to form the instruction tokens.

Context Fuser. The observation tokens, the instruction tokens and proprioceptive tokens are fused together using a context fuser which uses a stack of self-attention based transformer layers. We also additionally add a readout token, which we refer to as the waypoint-mode token. At the end of the context fuser we get a set of fused observation-instruction embeddings as well as the embedding

Hyperparam	Value
train steps	500K
optimizer	AdamW
warmup	linear upto 10K steps
learning rate	1e-4
learnign rate decay	constant
weight decay	1e-4
trajectory (action) horizon	10

Table 3: Hyperparameters used to train the different policies used.

for the readout token. The observation-instruction embeddings are used to predict the trajectory and thus passed into the trajectory diffusion transformer. Alternatively, the waypoint-mode embedding is used by the waypoint diffusion transformer to predict the $SE(3)$ waypoint as well as to predict the current mode for the robot. The observation-instruction embeddings are used by `RoboART` for anomaly detection.

Diffusion. For both trajectory diffusion and waypoint diffusion we use a Transformer decoder-based denoiser [68]. The denoiser takes as input noisy action embeddings together with a diffusion timestep embedding. These noisy actions and timestep embeddings cross-attend to the context embeddings (either the context tokens for trajectory diffusion or waypoint embedding for waypoint prediction). After multiple layers of alternating between self-attention and cross-attention the diffusion transformer outputs the denoised trajectory or waypoint action (as desired).

C.3 Diffusion Policy

Our diffusion policy architecture π_{dfn} uses a standard diffusion policy [1, 69] to directly output the joint angles to control the robot. Our base architecture is similar to π_{hyb} (described above) wherein we only use the trajectory mode, i.e., only the trajectory diffusion head is used to predict robot trajectories. The rest of the architecture including the vision encoders and the multi-modal instruction encoder are common between π_{dfn} and π_{hyb}. However, unlike π_{hyb}, π_{dfn} does not include a readout token (waypoint/mode token) within the context fuser.

C.4 Training and Inference Details

Table 3 shows the common set of hyper-parameters used to train each of our policies. We use a batch size of 256 during training. As shown in Figure 7 for the high dimensional image observations we use an additional token learner to reduce the number of image tokens. We use 128 tokens for each image observation. For our diffusion model we use a continuous time implementation based on [70]. Similar to [70] we use a second order Heun solver to solve the flow ODE. We use 30 ODE steps during inference. As shown in Table 3, we use an action horizon of 10. Since we collect our training data at 10Hz this corresponds to 1 second of robot motion. During inference, we open-loop rollout entire 10 steps before querying the policy again. During evaluation we use a maximum of 30 policy steps before we stop policy evaluation. For our targeted data collection experiment Section 5.2, we use a much smaller learning rate of $5e-6$ and a linear warmup of 4K steps. We finetune the policy for a total 20K steps.

D Ablations: Score Function and Size of Nominal Rerence Set

D.1 Ablations for `RoboART`

`RoboART` uses an anomaly score function $s_\pi(o, S_{\text{nom}})$ that computes the mean of the k-nearest neighbor cosine distances in policy embedding space. The set S_{nom} consists of embeddings computed for a random subset of the training data. Intuitively, this anomaly score quantifies how dissimilar a given observation is compared to similar training observations from the perspective of the policy. In the

tables below, we provide results from varying the size of S_{nom} and the value of k for each policy. Each table compares the predictions made by `RoboART` with the actual (empirically measured) performance by computing the Spearman rank correlation and the average prediction error, as described in Sec. 5.1. Generally, we find that predictions for π_{hyb} remain accurate when varying $|S_{\text{nom}}|$ with small k, while predictions for π_{dfn} (which has a significantly higher dimensional embedding space) benefit from either having a smaller value of $|S_{\text{nom}}|$ or larger values of k.

Hybrid policy π_{hyb}

$\lvert S_{\text{nom}}\rvert = 3000$	$k=1$	$k=5$	$k=10$
Spearman ρ (↑)	0.72	**0.78**	0.78
Av. pred. err. (↓)	0.12	**0.1**	0.12

$\lvert S_{\text{nom}}\rvert = 2000$	$k=1$	$k=5$	$k=10$
Spearman ρ (↑)	0.72	**0.79**	0.68
Av. pred. err. (↓)	0.12	**0.12**	0.13

$\lvert S_{\text{nom}}\rvert = 1000$	$k=1$	$k=5$	$k=10$
Spearman ρ (↑)	**0.76**	0.65	0.63
Av. pred. err. (↓)	**0.12**	0.13	0.17

$\lvert S_{\text{nom}}\rvert = 500$	$k=1$	$k=5$	$k=10$
Spearman ρ (↑)	**0.69**	0.63	0.56
Av. pred. err. (↓)	**0.12**	0.16	0.19

$\lvert S_{\text{nom}}\rvert = 200$	$k=1$	$k=5$	$k=10$
Spearman ρ (↑)	**0.72**	0.65	0.49
Av. pred. err. (↓)	**0.14**	0.18	0.23

Vanilla diffusion policy π_{dfn}

$\lvert S_{\text{nom}}\rvert = 3000$	$k=1$	$k=5$	$k=10$	$k=25$	$k=50$	$k=100$	$k=200$
Spearman ρ (↑)	0.59	0.52	0.56	0.66	0.59	**0.67**	0.66
Av. pred. err. (↓)	0.22	0.21	0.21	0.20	0.20	**0.19**	0.20

$\lvert S_{\text{nom}}\rvert = 2000$	$k=1$	$k=5$	$k=10$	$k=25$	$k=50$	$k=100$	$k=250$
Spearman ρ (↑)	0.55	0.52	0.53	0.64	0.64	**0.69**	0.17
Av. pred. err. (↓)	0.21	0.20	0.20	0.20	0.20	0.20	0.25

$\lvert S_{\text{nom}}\rvert = 1000$	$k=1$	$k=5$	$k=10$	$k=25$
Spearman ρ (↑)	**0.63**	0.52	0.60	0.59
Av. pred. err. (↓)	0.21	0.20	**0.19**	0.20

$\lvert S_{\text{nom}}\rvert = 500$	$k=1$	$k=5$	$k=10$
Spearman ρ (↑)	0.58	0.66	**0.71**
Av. pred. err. (↓)	0.21	0.19	**0.19**

$\lvert S_{\text{nom}} \rvert = 200$	$k = 1$	$k = 5$	$k = 10$
Spearman ρ (↑)	0.61	0.64	**0.75**
Av. pred. err. (↓)	0.19	0.20	**0.19**

E Predicting Performance From Anomalies

Fig. 8 compares the true (estimated) rankings of different environmental factors and success rates with rankings and success rates predicted by executing the anomaly detector on $\sim$ 20 real observations collected from each of the twelve off-nominal settings. Specifically, predicted success rates for each factor are computed as $R^{\pi}_{f,\text{anom}} := 1 - \alpha^{\pi}_{f,\text{real}}$. For anomaly detection, we use $k = 10, |S_{\text{nom}}| = 3000$ for π_{hyb} and $k = 10, |S_{\text{nom}}| = 200$ for π_{dfn}. The anomaly threshold for π_{hyb} is computed using conformal prediction as described in Sec. 4.2 in order to bound the anomaly rate in nominal conditions to $1 - R^{\pi_{\text{hyb}}}_{\text{nom}}$. For π_{dfn}, we found this procedure to yield an anomaly threshold that is too conservative (i.e., flagging most observations in the different off-nominal scenarios as anomalous). This sensitivity may be due to the relatively small number $n_{\text{val}} = 70$ of nominal observations we used to compute the anomaly threshold and the very high dimensionality of the embedding space ($\mathbb{R}^{515\times513}$) of π_{dfn}. In order to correct for this, we computed the anomaly threshold with a slightly higher estimate of the nominal success rate (0.8 vs. 0.65), i.e., using conformal prediction to bound the anomaly rate in nominal conditions to $1 - 0.8$ rather than $1 - 0.65$. Fig. 8 shows a strong correlation between predicted and realized performance. We note that the predictions $R^{\pi}_{f,\text{anom}}$ are made using $5\times$ fewer observations than predictions from the full RoboART pipeline ($\sim$ 20 real observations vs. 100 edited observations), thus making them significantly more susceptible to noise.

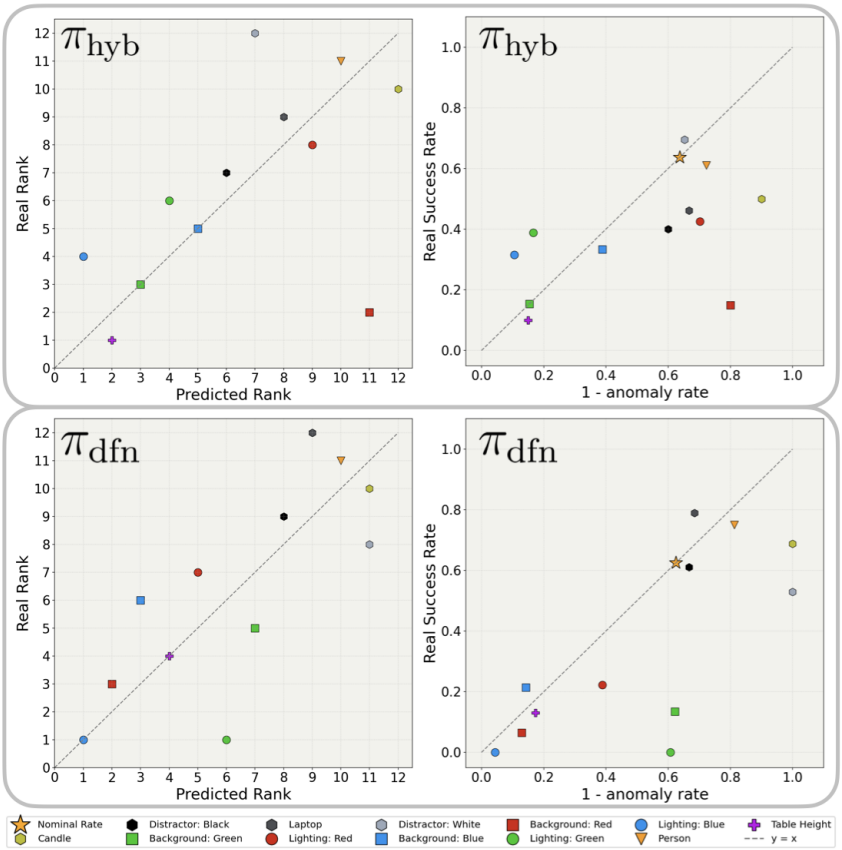


Figure 8: Evaluating predictions made from anomaly rates computed using real observations.