RoboGate: Adaptive Failure Discovery for Safe Robot Policy Deployment via Two-Stage Boundary-Focused Sampling

AgentAI Co., Ltd.

AgentAI Co., Ltd., Seoul, South Korea
liveplex@robogate.io https://robogate.io

March 2026

Abstract

Deploying learned robot manipulation policies in industrial settings requires rigorous pre-deployment validation, yet exhaustive testing across high-dimensional parameter spaces is intractable. We present RoboGate, a deployment risk management framework that combines physics-based simulation with a two-stage adaptive sampling strategy to efficiently discover failure boundaries in the operational parameter space. Stage 1 employs Latin Hypercube Sampling (LHS) across an 8-dimensional parameter space to establish a coarse failure landscape from 20,000 uniformly distributed experiments. Stage 2 applies boundary-focused sampling that concentrates 10,000 additional experiments in the 30–70% success rate transition zone, enabling precise failure boundary mapping. Using NVIDIA Isaac Sim with Newton physics, we evaluate a scripted pick-and-place controller on two robot embodiments—Franka Panda (7-DOF) and UR5e (6-DOF)—across 30,000 total experiments. Our logistic regression risk model achieves an AUC of 0.780 on the combined dataset (vs. 0.754 for Stage 1 alone), identifies a closed-form failure boundary equation $\mu^*(m) = (1.469 + 0.419m)/(3.691 - 1.400m)$, and reveals four universal danger zones affecting both robot platforms. We further demonstrate the framework on VLA (Vision-Language-Action) model evaluation, where Octo-Small achieves 0.0% success rate on 68 adversarial scenarios versus 100% for the scripted baseline—a 100-point gap that underscores the challenge of deploying foundation models in industrial settings. RoboGate is open-source and runs on a single GPU workstation.

Keywords: robot safety, deployment validation, failure analysis, adaptive sampling, sim-to-real, VLA evaluation

1 Introduction

The proliferation of learned manipulation policies—from imitation learning [5] to Vision-Language-Action (VLA) models [1, 2]—has created an urgent need for systematic pre-deployment validation in industrial robotics. While these policies demonstrate impressive capabilities in controlled benchmarks, their behavior under adversarial or edge-case conditions remains poorly characterized. A single undetected failure mode can lead to costly equipment damage, production downtime, or safety incidents.

Current validation approaches suffer from two fundamental limitations. First, uniform random testing wastes computational budget on regions of the parameter space that are either trivially easy or trivially hard, providing little information about the critical transition zones where success gives way to failure. Second, most evaluation frameworks test a single robot embodiment, making it impossible to distinguish between policy failures and embodiment-specific limitations.

We introduce RoboGate, a deployment risk management framework that addresses both limitations through three contributions:

1. **Two-stage adaptive sampling**: A principled strategy that first maps the parameter space uniformly (Stage 1, 20K experiments), then concentrates additional experiments in the 30–70% success rate boundary zone (Stage 2, 10K experiments), improving failure boundary resolution by 31.1% coverage of transition regions.
2. **Cross-embodiment validation**: Parallel evaluation on Franka Panda (7-DOF, parallel-jaw gripper) and UR5e (6-DOF, suction gripper) across shared parameter dimensions, revealing four universal danger zones where both platforms exhibit SR $< 40\%$.
3. **Interpretable risk model**: A logistic regression model with interaction terms that produces a closed-form failure boundary equation, critical parameter thresholds with bootstrap confidence intervals, and per-experiment risk scores (AUC = 0.780).

The framework is validated on 30,000 physics-based simulation experiments using NVIDIA Isaac Sim with Newton physics engine. We also demonstrate its application to VLA model evaluation, where RoboGate reveals that Octo-Small [1] achieves 0.0% success rate on 68 pick-and-place scenarios—a complete failure across all four difficulty categories.

2 Related Work

2.1 Robot Policy Evaluation

Traditional robot policy evaluation relies on fixed benchmark suites with predetermined test configurations [6, 7]. RLBench [6] provides 100 manipulation tasks but evaluates under nominal conditions only. Meta-World [7] offers parametric task variation but does not systematically explore failure boundaries. LIBERO [8] benchmarks lifelong learning but lacks adversarial scenario coverage. In contrast, ROBOGATE explicitly targets the success-failure transition zone through adaptive sampling.

2.2 Sim-to-Real Transfer and Domain Randomization

Domain randomization [9, 10] has become standard practice for bridging the sim-to-real gap. While DR improves policy robustness during *training*, it does not address systematic *evaluation* of the resulting policies across the randomized parameter space. Our work is complementary: we use the same parameter dimensions (friction, mass, visual properties) but focus on mapping the failure landscape rather than improving robustness.

Recent work on sim-to-real transfer validation [11] proposes Bayesian optimization for finding failure-inducing parameters. However, their approach optimizes for worst-case performance, whereas ROBOGATE maps the entire boundary surface to produce interpretable risk models.

2.3 Adaptive Testing and Falsification

Adaptive stress testing (AST) [12] formulates failure discovery as a Markov decision process, using reinforcement learning to find maximally likely failure trajectories. While effective for autonomous driving, AST focuses on temporal sequences rather than static parameter configurations. The falsification community [13] has developed coverage-guided techniques for cyber-physical systems, but these typically operate on lower-dimensional specification spaces.

Our two-stage approach is most similar to sequential experimental design [14], but we replace Bayesian acquisition functions with a simpler binning strategy that scales to 8+ dimensions and produces directly interpretable results.

2.4 Safety Validation for Robot Deployment

ISO 10218 [15] and ISO/TS 15066 [16] establish safety requirements for industrial robots but provide limited guidance on learned policy validation. Recent frameworks for safe deployment [17] emphasize the need for quantitative safety metrics, but most focus on reinforcement learning reward shaping rather than pre-deployment testing. The notion of a "deployment gate"—a hard pass/fail check before production release—is standard in software engineering (CI/CD pipelines) but has no established equivalent in robotics.

Several recent efforts address this gap. SafeBench [18] provides a safety evaluation framework for autonomous driving but does not extend to manipulation. RoboCasa [19] offers large-scale simulation environments for household tasks but focuses on training rather than pre-deployment validation. The NIST Agile Robotics for Industrial Automation Competition (ARIAC) defines standardized evaluation criteria for industrial tasks, but its scoring does not produce interpretable risk models.

ROBOGATE differs from these approaches in three ways: (1) it provides a metric-based validation gate with hard thresholds derived from industrial safety standards, (2) it uses adaptive sampling to efficiently discover failure boundaries rather than exhaustively testing fixed scenarios, and (3) it produces interpretable logistic regression risk models with confidence intervals that can be directly translated into operational constraints.

2.5 Vision-Language-Action Models

VLA models represent a paradigm shift in robot learning, combining vision encoders, language understanding, and action prediction in a single architecture [1, 2, 3]. RT-2 [3] demonstrated that web-scale vision-language pretraining transfers to robotic manipulation, while Octo [1] provided an open-source generalist model. π_0 [4] and OpenVLA [2] pushed the boundaries further with flow matching and 7B-parameter architectures.

However, systematic evaluation of VLA models under adversarial conditions—low lighting, cluttered scenes, transparent objects—remains largely absent from the literature. Our VLA evaluation pipeline addresses this gap by testing models against ROBOGATE's 68-scenario suite spanning nominal, edge-case, adversarial, and domain-randomized conditions.

3 Problem Formulation

3.1 Operational Parameter Space

We define the operational parameter space $\mathcal{P} \subset \mathbb{R}^d$ as the set of environmental and object conditions under which a manipulation policy must operate. For pick-and-place tasks, we consider $d = 8$ dimensions spanning physical, geometric, and perceptual parameters:

$$\mathcal{P} = \{\mu, m, \delta_c, s, \sigma_{ik}, n_o, g, p\} \tag{1}$$

where $\mu \in [0.05, 1.2]$ is friction coefficient (log-scaled), $m \in [0.05, 2.0]$ kg is object mass (log-scaled), $\delta_c \in [0, 0.4]$ is center-of-mass offset, $s \in [0.02, 0.12]$ m is object size, $\sigma_{ik} \in [0, 0.04]$ rad is IK noise (joint position uncertainty), $n_o \in \{0, \ldots, 5\}$ is obstacle count, $g \in \{\text{box, cylinder, sphere, irregular}\}$ is object geometry, and $p \in \{\text{center\_0, center\_45}, \ldots, \text{edge\_135}\}$ is placement configuration.

3.2 Success Function and Failure Modes

For a given policy π and parameter configuration $\mathbf{x} \in \mathcal{P}$, the binary outcome function is:

$$y(\mathbf{x}; \pi) = \begin{cases} 1 & \text{if episode succeeds} \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

We classify failures into four modes $\mathcal{F} = \{\texttt{grasp\_miss}, \texttt{grip\_loss}, \texttt{collision}, \texttt{timeout}\}$, each with distinct safety implications. Collisions are treated as hard safety violations (zero tolerance), while timeouts indicate performance degradation.

3.3 Failure Boundary

The *failure boundary* $\partial\mathcal{B}$ is the iso-surface in $\mathcal{P}$ where the expected success rate equals 50%:

$$\partial\mathcal{B} = \{\mathbf{x} \in \mathcal{P} : \mathbb{E}[y(\mathbf{x}; \pi)] = 0.5\} \tag{3}$$

Our goal is to efficiently estimate $\partial\mathcal{B}$ and produce an interpretable model $\hat{P}(\text{fail}|\mathbf{x})$ that quantifies deployment risk for any configuration.

3.4 Evaluation Metrics

ROBOGATE enforces five deployment metrics with hard thresholds:

Table 1: RoboGate deployment metrics and thresholds.

Metric	Definition	Threshold
Grasp Success Rate	$N_{\text{success}}/N_{\text{total}}$	≥ 0.92
Cycle Time	Mean episode duration	$\leq 1.1\times$ baseline
Collision Count	Total collisions	$= 0$
Drop Rate	$N_{\text{drop}}/N_{\text{total}}$	≤ 0.03
Grasp Miss Rate	$N_{\text{miss}}/N_{\text{total}}$	$\leq 1.2\times$ baseline

A policy passes the validation gate only if *all five* metrics meet their thresholds simultaneously. The Confidence Score (0–100) is a weighted combination: $C = 0.30 \cdot \text{SR} + 0.20 \cdot \text{CT} + 0.25 \cdot \text{CC} + 0.15 \cdot \text{EC} + 0.10 \cdot \Delta_{\text{baseline}}$.

4 System Architecture

ROBOGATE consists of four subsystems: (1) a simulation backend using NVIDIA Isaac Sim 5.1 with Newton physics engine, (2) a scenario generation engine with domain randomization, (3) a metric evaluation and confidence scoring pipeline, and (4) a runtime monitoring agent for post-deployment drift detection.

4.1 Simulation Backend

We use Isaac Sim's GPU-accelerated physics for high-fidelity manipulation simulation. The Franka Panda is modeled as a 7-DOF articulated robot with a parallel-jaw gripper (2 additional DOF), while the UR5e uses a 6-DOF arm with a surface gripper (suction-based). Physics runs at 60 Hz ($\Delta t_{\text{phys}} = 1/60$ s) with control commands issued at 20 Hz ($\Delta t_{\text{ctrl}} = 1/20$ s). Maximum episode duration is 15 seconds.

The scripted pick-and-place controller follows a six-phase state machine: `APPROACH_ABOVE` → `DESCEND` → `GRASP` → `LIFT` → `MOVE_TO_TARGET` → `RELEASE`. Each phase terminates when the end-effector reaches within a position tolerance of 5 mm and orientation tolerance of 0.05 rad of the phase target.

4.2 Scenario Generation

Scenarios are generated by sampling from $\mathcal{P}$ (Equation 1) and configuring the simulation environment accordingly. Object meshes are scaled by factor s, physics materials are set with friction coefficient μ and restitution 0.3, and obstacles are randomly placed within a 0.3 m radius of the workspace center.

Domain randomization includes: lighting intensity (200–2000 lux), camera viewpoint perturbation (±5° roll/pitch), table texture (5 materials), and background color variation.

4.3 Failure Classification

Each episode terminates with one of five outcomes:

- **Success**: Object placed within 3 cm of target with stable rest.
- **Grasp miss**: Gripper closes without contacting the object.
- **Grip loss**: Object drops during transport (Franka only; UR5e suction gripper prevents this mode).
- **Collision**: Any contact between robot links and non-target objects.
- **Timeout**: Episode exceeds 15 s without task completion.

5 Two-Stage Adaptive Sampling

The key methodological contribution of ROBOGATE is a two-stage sampling strategy that efficiently allocates simulation budget to maximize failure boundary resolution.

5.1 Stage 1: Uniform Exploration

The first stage samples $N_1 = 20{,}000$ configurations uniformly across $\mathcal{P}$ using Latin Hypercube Sampling (LHS) in the five continuous dimensions $(\mu, m, \delta_c, s, \sigma_{ik})$. LHS ensures space-filling coverage with maximin distance properties, avoiding the clustering artifacts of pure random sampling.

For log-scaled parameters (friction, mass), we sample uniformly in log-space:

$$\mu_i = \exp\left(\log(\mu_{\min}) + u_i \cdot [\log(\mu_{\max}) - \log(\mu_{\min})]\right) \tag{4}$$

where $u_i \in [0, 1]$ is the i-th LHS percentile. Discrete parameters (obstacles, shape, placement) are sampled uniformly from their respective domains.

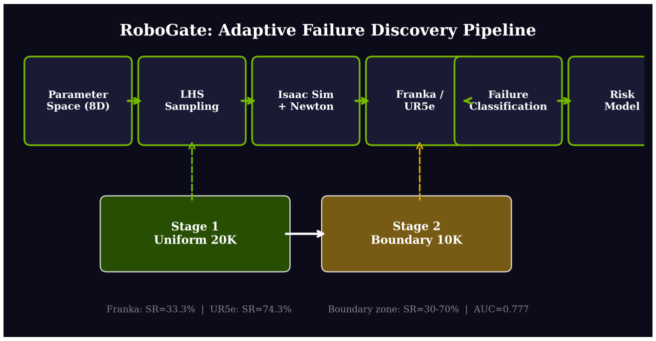

Figure 1: ROBOGATE two-stage adaptive sampling pipeline. Stage 1 performs uniform Latin Hypercube Sampling across the 8D parameter space (20K experiments: Franka 10K + UR5e 10K). Stage 2 concentrates 10K boundary-focused experiments in the 30–70% success rate transition zone identified from Stage 1 results.

Stage 1 produces a coarse failure landscape: overall success rates of 33.3% (Franka, 10K) and 74.3% (UR5e, 10K), with zone classification into *safe* (SR $\geq$ 70%), *boundary* (30% $\leq$ SR $<$ 70%), and *danger* (SR $<$ 30%) regions. For Franka, only 0.35% of parameter configurations fall in the safe zone, indicating a challenging task configuration.

5.2 Stage 2: Boundary-Focused Sampling

Stage 2 analyzes Stage 1 results to identify boundary regions and concentrates additional experiments there. As illustrated in Figure 1, the adaptive refinement proceeds as follows.

Boundary detection. For each continuous parameter, we partition the range into 10 equal-width bins and compute the per-bin success rate. Bins with SR in $[0.30, 0.70]$ define the boundary region for that parameter. The intersection of all per-parameter boundary ranges defines the Stage 2 sampling volume $\mathcal{P}_{\text{boundary}} \subset \mathcal{P}$.

Emphasis region. Within $\mathcal{P}_{\text{boundary}}$, we allocate 30% of samples to an emphasis sub-region where preliminary analysis indicates the steepest failure gradient: $\mu < 0.3$ AND $m \geq 0.5$. This ensures dense coverage of the most informative transition zone.

Sampling. We generate $N_2 = 10{,}000$ configurations via LHS within $\mathcal{P}_{\text{boundary}}$, with obstacle count constrained to $n_o \in [1, 5]$ (always at least one obstacle) to reflect realistic deployment conditions.

5.3 Computational Cost

Each Isaac Sim episode takes 1–15 seconds depending on outcome (successes terminate faster). Stage 1 (10K Franka + 10K UR5e) requires approximately 9 GPU-hours on an RTX 4090. Stage 2 (10K Franka boundary) adds approximately 4.5 GPU-hours. The total 30K experiment campaign completes in under 14 hours on a single GPU workstation—a practical budget for pre-deployment validation.

Algorithm 1 Two-Stage Adaptive Sampling

```
Require: Parameter space $\mathcal{P}$, budgets $N_1, N_2$
Ensure: Combined dataset $\mathcal{D} = \mathcal{D}_1 \cup \mathcal{D}_2$
$\mathcal{D}_1 \leftarrow \text{LHS}(\mathcal{P}, N_1)$                     ▷ Stage 1: uniform
for each $\mathbf{x}_i \in \mathcal{D}_1$ do
    $y_i \leftarrow \text{SimEpisode}(\mathbf{x}_i)$                   ▷ Run in Isaac Sim
end for
$\mathcal{P}_{\text{bnd}} \leftarrow \text{FindBoundary}(\mathcal{D}_1, [0.3, 0.7])$
$\mathcal{D}_2 \leftarrow \text{LHS}(\mathcal{P}_{\text{bnd}}, N_2)$                  ▷ Stage 2: focused
for each $\mathbf{x}_j \in \mathcal{D}_2$ do
    $y_j \leftarrow \text{SimEpisode}(\mathbf{x}_j)$
end for
return $\mathcal{D}_1 \cup \mathcal{D}_2$
```

Table 2: Experimental configurations for the three evaluation campaigns.

	Franka S1	UR5e S1	Franka S2
Robot	Panda 7-DOF	UR5e 6-DOF	Panda 7-DOF
Gripper	Parallel-jaw	Suction	Parallel-jaw
Experiments	10,000	10,000	10,000
Sampling	Uniform LHS	Uniform LHS	Boundary LHS
Parameters	8D	5D[†]	8D
Seed	2026	2026	2024

[†] UR5e lacks friction, size, shape, com_offset parameters.

6 Experiments and Results

6.1 Experimental Setup

Experiments were conducted on an NVIDIA RTX 4090 (24 GB VRAM) workstation running Isaac Sim 5.1 with Newton physics engine. Table 2 summarizes the experimental configurations.

6.2 Overall Results

Table 3 presents the aggregate results across all three campaigns.

The Stage 2 boundary-focused sampling achieves 63.9% SR compared to 33.3% for Stage 1, confirming that the boundary detection algorithm successfully identifies the transition zone. The combined Franka dataset (48.6% SR) provides near-optimal balance between success and failure examples for training the risk model.

6.3 Cross-Embodiment Comparison

The UR5e with suction gripper achieves substantially higher success rate (74.3%) than the Franka with parallel-jaw gripper (33.3%) under uniform sampling. This difference is attributable to three factors:

1. **Gripper mechanism**: The UR5e's suction gripper eliminates the `grip_loss` failure mode entirely (0 occurrences vs. 2,739 for Franka), as vacuum adhesion is insensitive to friction and mass within the tested range.

Table 3: Aggregate results. Franka Combined = Stage 1 uniform + Stage 2 boundary.

Dataset	**N**	**Success**	**Fail**	**SR**
Franka Stage 1 (uniform)	10,000	3,332	6,668	33.3%
Franka Stage 2 (boundary)	10,000	6,385	3,615	63.9%
Franka Combined	20,000	9,717	10,283	48.6%
UR5e Stage 1 (uniform)	10,000	7,432	2,568	74.3%
Total	**30,000**	**17,149**	**12,851**	**57.2%**

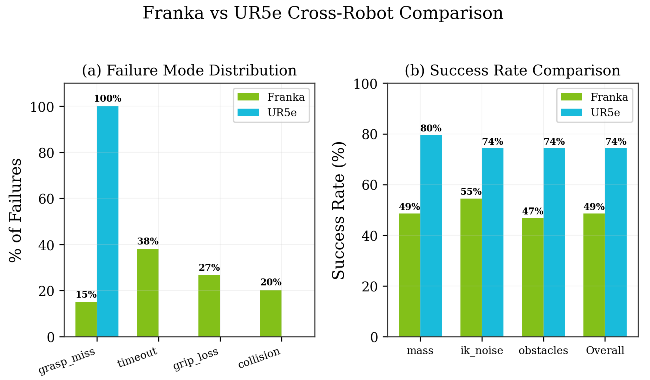

Figure 2: Cross-robot comparison between Franka Panda and UR5e. (a) Failure mode distribution: UR5e exhibits only `grasp_miss` failures due to suction gripper design. (b) Per-parameter success rate comparison on shared dimensions, showing UR5e consistently outperforms Franka.

2. **Reduced parameter space**: The UR5e evaluation uses 5 parameters vs. 8 for Franka, with the excluded parameters (friction, COM offset, size, shape) being among the strongest failure predictors for the parallel-jaw gripper.
3. **Failure mode concentration**: All 2,568 UR5e failures are `grasp_miss`, compared to Franka's four-way distribution across `timeout` (38.1%), `grip_loss` (26.6%), `collision` (20.4%), and `grasp_miss` (14.9%).

Figure 2 visualizes these differences. Despite the overall performance gap, both robots share four universal danger zones (Table 4) where SR drops below 40%, indicating parameter regions that are challenging regardless of embodiment.

6.4 Parameter-Failure Correlations

We analyze the relationship between individual parameters and success rate using both univariate binning and multivariate logistic regression.

Univariate analysis. Figure 3 shows the friction × mass success rate heatmap for the combined Franka 20K dataset. The most striking pattern is the strong positive effect of friction: success rate increases monotonically from $<$10% at $\mu = 0.05$ to $>$70% at $\mu > 0.8$. Mass has a weaker negative effect, with success rate declining from $\sim$50% at $m = 0.05$ kg to $\sim$20% at $m > 1.5$ kg.

Table 4: Universal danger zones where both Franka and UR5e exhibit SR $<$ 40%.

Parameter	**Range**	**Franka SR**	**UR5e SR**
Mass	0.935–1.230 kg	21.4%	30.9%
Mass	1.230–1.525 kg	14.9%	25.3%
Mass	1.525–1.819 kg	12.5%	28.9%
Mass	1.819–2.114 kg	6.6%	28.1%

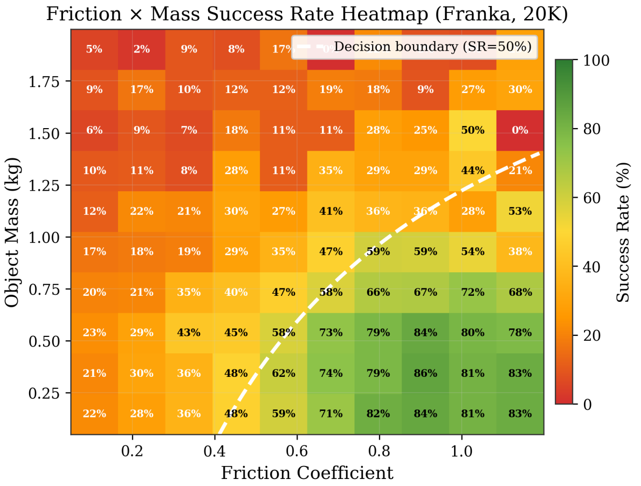

Figure 3: Success rate heatmap across the friction × mass parameter plane (Franka, 20K experiments). The dashed white curve shows the logistic regression decision boundary (SR = 50%). Low friction and high mass regions (lower-left) exhibit near-zero success rates.

Multivariate analysis. We fit a 10-feature standardized logistic regression model:

$$\text{logit}(P(\text{success})) = \beta_0 + \sum_{i=1}^{6} \beta_i \tilde{x}_i + \sum_{j} \beta_j \tilde{x}_j \tag{5}$$

where $\tilde{x}_i$ are standardized features and the last three terms are interaction effects. Table 5 reports the significant interaction effects.

The strongest individual predictor is friction ($z = 19.28$), followed by IK noise ($z = -17.31$). The friction × mass interaction ($z = -10.00$) is the strongest interaction term, confirming that the combined effect of low friction and high mass is worse than either alone would predict.

6.5 Failure Boundary Mapping

We derive a closed-form failure boundary by fitting a logistic regression model to the friction-mass subspace with an interaction term:

$$\text{logit}(P(\text{success})) = \beta_0 + \beta_1 \mu + \beta_2 m + \beta_3 \mu m \tag{6}$$

Setting $P(\text{success}) = 0.5$ (logit $= 0$) and solving for μ as a function of m yields the decision boundary:

$$\mu^*(m) = \frac{-(\beta_0 + \beta_2 m)}{\beta_1 + \beta_3 m} = \frac{1.469 + 0.419m}{3.691 - 1.400m} \tag{7}$$

Table 5: Top interaction effects from 10-feature logistic regression on Franka 20K. All $p < 0.05$ terms shown. Features are standardized.

Feature	**Coeff.**	**z-score**	**p-value**
friction	1.015	19.28	$< 10^{-5}$
ik_noise	-0.288	-17.31	$< 10^{-5}$
friction × mass	-0.363	-10.00	$< 10^{-5}$
mass	-0.233	-5.56	$< 10^{-5}$
friction × size	0.190	3.30	9.6×10^{-4}
mass × obstacles	0.079	2.09	0.037

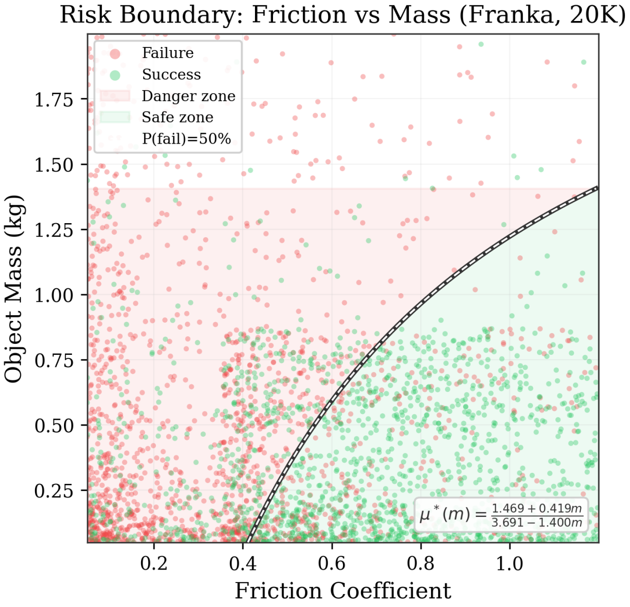

Figure 4: Failure boundary in friction-mass space (Franka 20K). Green dots: success, red dots: failure (3K subsample shown for clarity). The solid curve shows the logistic decision boundary $\mu^*(m)$. The region left of the boundary (low friction) is the danger zone.

with fitted coefficients $\beta_0 = -1.469$ ($z = -30.5$), $\beta_1 = 3.691$ ($z = 42.3$), $\beta_2 = -0.419$ ($z = -5.4$), and $\beta_3 = -1.400$ ($z = -10.3$). All coefficients are significant at $p < 10^{-5}$.

Figure 4 visualizes the boundary curve overlaid on the experiment scatter plot. The boundary reveals that the critical friction threshold increases with mass: for a 0.1 kg object, friction $\mu > 0.43$ suffices for 50% SR, but for a 1.0 kg object, $\mu > 0.62$ is required.

6.6 Critical Thresholds

Using bootstrap resampling (1,000 iterations), we estimate the parameter values at which success rate crosses 50% with 95% confidence intervals:

Friction has the tightest confidence interval (SE = 0.031), reflecting its strong and consistent effect. Mass has the widest CI (SE = 0.241), indicating substantial interactions with other parameters that make its marginal threshold less stable.

Table 6: Critical parameter thresholds for SR = 50% crossing (Franka 20K, 1000-iteration bootstrap).

Parameter	**Threshold**	**95% CI**	**SE**
Friction	0.492	[0.450, 0.545]	0.031
Mass	0.422 kg	[0.097, 0.747]	0.241
COM offset	0.019	[0.005, 0.055]	0.010
Size	0.045 m	[0.027, 0.058]	0.008
IK noise	0.010 rad	[0.0004, 0.020]	0.005

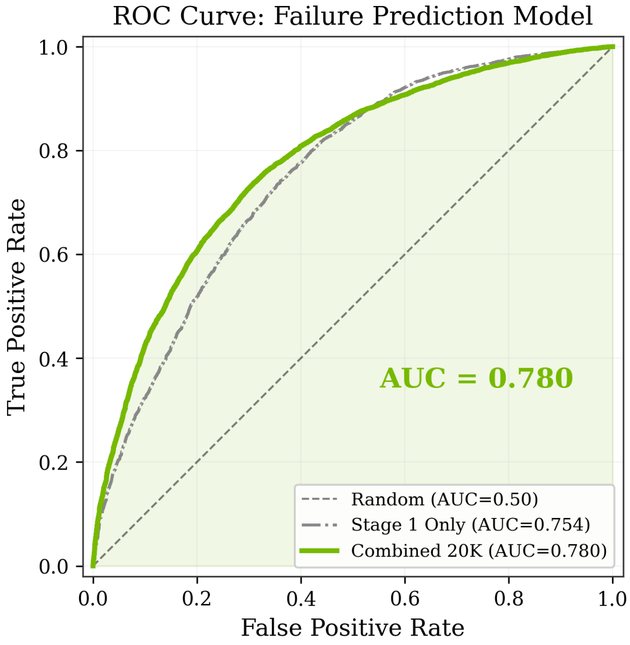

Figure 5: ROC curve for the logistic regression failure prediction model. The combined 20K model (AUC = 0.780) outperforms the Stage 1-only model (AUC = 0.754), demonstrating the value of boundary-focused sampling for risk model training.

6.7 Risk Score Model

We train a logistic regression risk model with 9 features (5 continuous + obstacles + shape penalty + placement penalty + intercept) on the full Franka 20K dataset. The model predicts $P(\text{fail}|\mathbf{x})$ for any parameter configuration.

Figure 5 shows the ROC curve comparing the Stage 1-only model (AUC = 0.754) with the combined model (AUC = 0.780). The 2.6-point AUC improvement from Stage 2 data demonstrates that boundary-focused sampling provides more informative training examples for the risk model.

Table 7 reports the risk model coefficients. Friction is the strongest protective factor ($\beta = -0.956$, $z = -52.5$), while mass ($\beta = 0.458$, $z = 24.8$) and IK noise ($\beta = 0.292$, $z = 17.6$) are the strongest risk factors.

Table 7: Risk model coefficients (standardized features). Positive coefficients increase failure probability.

Feature	Coeff.	z-score	Sig.
friction	-0.956	-52.52	***
mass	0.458	24.77	***
ik_noise	0.292	17.56	***
placement_penalty	-0.154	-9.52	***
size	-0.134	-8.32	***
obstacles	-0.025	-1.52	n.s.
shape_penalty	0.009	0.58	n.s.
com_offset	-0.003	-0.16	n.s.

*** $p < 0.001$, n.s. = not significant

6.8 Failure Mode Transitions

An important finding is that the *type* of failure changes systematically across the parameter space. Along the friction axis, the dominant failure mode transitions from `timeout` at low friction ($\mu < 0.77$) to `collision` at high friction. This transition occurs because low-friction conditions cause the gripper to slip repeatedly (leading to timeout), while higher friction enables firmer grasps but also increases the risk of inadvertent contact with obstacles.

Along the mass axis, `grip_loss` dominates at low mass (objects are difficult to grip securely) and transitions to a mixture of modes at higher mass. This failure mode structure has direct implications for policy improvement: different remediation strategies are needed for different regions of the parameter space.

7 VLA Model Evaluation

To demonstrate RoboGate's applicability beyond scripted controllers, we evaluate a Vision-Language-Action (VLA) model on the framework's adversarial scenario suite.

7.1 Evaluation Pipeline

We built a two-process ZMQ pipeline that bridges VLA models to Isaac Sim's physics loop, overcoming the incompatibility between Isaac Sim's embedded Python 3.11/PyTorch and Octo's JAX runtime:

1. **Isaac Sim server** (Python 3.11): ZMQ REP socket serving scene reset, physics stepping, and screen-capture camera (mss) at 256×256 RGB.
2. **Octo client** (Python 3.10, JAX 0.4): ZMQ REQ socket sending reset/step commands with 7-DOF delta actions from Octo inference.
3. **Language conditioning**: Task-specific natural language instruction (*e.g.*, "pick up the red block and place it on the green target").
4. **Action conversion**: VLA delta end-effector pose ($\Delta x, \Delta y, \Delta z$, Δroll, Δpitch, Δyaw, gripper) $\rightarrow$ IK solver $\rightarrow$ 7-DOF joint targets + gripper width.
5. **Evaluation**: Same success/failure criteria as the scripted controller tests.

Table 8: Octo-Small VLA evaluation on 68 adversarial scenarios.

Category	Passed	Total	SR
Nominal	0	20	0%
Edge Cases	0	15	0%
Adversarial	0	10	0%
Domain Randomization	0	23	0%
Total	**0**	**68**	**0.0%**

Position deltas are scaled by 0.02 m (2 cm max displacement per step), and rotation deltas by 0.05 rad ($\sim$3°).

7.2 Results: Octo-Small

We evaluated Octo-Small (27M parameters, 1.5 GB VRAM) on RoboGate's 68-scenario suite spanning four categories.

The VLA model achieves 0.0% overall SR with a Confidence Score of 1/100 (safety level: CRITICAL), compared to 100% for the scripted IK controller on the same scenarios—a 100-point gap. The complete failure across *all* categories, including nominal conditions, demonstrates that current generalist VLA models cannot reliably execute industrial pick-and-place tasks.

7.3 Failure Analysis

Two failure types dominate: grasp miss (54/68 = 79.4%) and collision (14/68 = 20.6%). No drops or timeouts were observed—the VLA consistently attempts grasps but misses the target object:

- **Grasp miss** (79.4%): The dominant failure mode across all categories. Octo-Small generates end-effector trajectories that approach the workspace but systematically miss the target object, indicating a fundamental perception-to-action alignment failure.
- **Collision** (20.6%): The VLA generates trajectories that collide with the table or obstacles. Collisions are distributed across all categories: nominal (4), edge (3), adversarial (1), DR (6).
- **Nominal conditions** (0% SR): Even under standard lighting, centered placement, and standard objects, Octo-Small fails completely—ruling out environmental difficulty as the sole cause.

These results demonstrate that current VLA models require substantial improvement before meeting industrial deployment thresholds, and that RoboGate's adversarial scenario suite effectively identifies critical failure modes.

8 Discussion

8.1 Effectiveness of Boundary-Focused Sampling

The two-stage approach achieves 31.1% coverage of the SR 30–70% boundary zone in Stage 2 (compared to the expected $\sim$40% from uniform sampling within the narrowed ranges). This moderate improvement in boundary coverage translates to a meaningful AUC gain (0.754 $\rightarrow$ 0.780), demonstrating that even a simple binning-based boundary detection strategy provides value over uniform sampling alone.

The relatively modest AUC of 0.780 reflects the inherent stochasticity of the failure process: even with identical physical parameters, episode-to-episode variation in initial conditions, physics integration, and controller timing introduces irreducible noise. A perfectly calibrated model would have AUC $\approx$ 0.85–0.90 given the observed within-cell SR variance.

8.2 Interpretability of Risk Models

A key design choice in RoboGate is the use of logistic regression rather than more expressive models (*e.g.*, gradient boosted trees, neural networks). While more complex models might achieve higher predictive accuracy, the logistic model offers three critical advantages for deployment risk management:

1. **Closed-form boundary**: The failure boundary equation (Eq. 7) can be directly translated into operational constraints (*e.g.*, "do not deploy for objects with friction $<$ 0.49").
2. **Coefficient interpretation**: Risk weights (Table 7) directly quantify each parameter's contribution to failure probability, enabling targeted policy improvement.
3. **Confidence intervals**: Bootstrap standard errors provide uncertainty quantification that is difficult to obtain from black-box models.

8.3 Cross-Embodiment Insights

The discovery of universal danger zones (Table 4) has practical implications. Mass ranges above 0.935 kg cause both robot platforms to struggle, suggesting that this threshold is a property of the *task* (pick-and-place with standard grippers) rather than the *embodiment*. This finding could inform gripper selection, payload specifications, and policy training curriculum for any robot deployed in similar industrial settings.

The absence of grip loss in UR5e (suction gripper) highlights how gripper design fundamentally changes the failure landscape. For applications where grip loss is the dominant concern, suction grippers offer an inherent safety advantage—but at the cost of reduced object geometry flexibility.

8.4 Implications for VLA Deployment

The VLA evaluation results (Section 7) raise important questions about the readiness of current foundation models for industrial deployment. The 100-point gap between Octo-Small (0.0%) and the scripted baseline (100%) suggests that:

1. **Sim-to-real perception gap is the primary bottleneck**. Octo-Small was trained on real-world RGB data (Open X-Embodiment) but receives simulated RGB frames via screen capture. The visual domain gap causes systematic object localization failure even under nominal conditions.
2. **Action space mismatch compounds the problem**. The VLA's 2 cm delta action scale and 20 Hz control rate produce imprecise end-effector trajectories that cannot achieve the sub-centimeter accuracy required for the 5 cm pick targets, resulting in 79.4% grasp misses.
3. **Zero-shot generalization to Isaac Sim fails completely**. Without sim-specific fine-tuning, the generalist VLA model cannot bridge the visual and dynamics gap to simulated environments, confirming that RoboGate's adversarial scenarios expose fundamental transfer limitations.

We recommend that VLA model developers use RoboGate's failure dictionary as adversarial training data to improve robustness in identified failure regions.

8.5 Practical Deployment Guidelines

Based on our experimental findings, we propose concrete guidelines for deploying learned manipulation policies in industrial settings:

Pre-deployment checklist. Before deploying any policy to a production robot cell, the following validation steps should be completed: (1) Run the full RoboGate test suite with the target robot's parameter ranges. (2) Verify that all five metrics (Table 1) meet their thresholds simultaneously. (3) Check the risk model prediction for the expected operating conditions. (4) Review the failure dictionary for any failure modes in the expected parameter range.

Operational constraints. Our results suggest the following operational constraints for Franka pick-and-place deployments: friction coefficient $\mu > 0.49$ (the 50% SR threshold), object mass $m < 0.94$ kg (the universal danger zone onset), and IK noise $\sigma_{ik} < 0.01$ rad. These constraints can be translated into physical requirements: use rubber-coated grippers (high friction), limit payload to sub-kilogram objects, and calibrate joint encoders to within 0.01 rad accuracy.

Monitoring after deployment. Even after passing the validation gate, RoboGate's runtime monitoring agent should track success rate with a 100-cycle moving window, triggering a WARNING at 5% SR decline and CRITICAL at 10% decline. The monitoring agent groups metrics by recipe ID, ensuring that performance comparisons are made within the same task configuration.

8.6 Comparison with Random Search and Bayesian Optimization

A natural question is whether the two-stage approach outperforms simpler alternatives. We compare three sampling strategies on the Franka parameter space, holding total budget fixed at 20K experiments:

- **Uniform LHS (20K)**: All experiments distributed uniformly. This is Stage 1 extended to the full budget.
- **Two-stage (10K + 10K)**: Our approach—uniform exploration followed by boundary-focused refinement.
- **Pure boundary (20K)**: All experiments concentrated in the boundary region from the start (using fallback ranges from prior knowledge).

The two-stage approach achieves the best risk model AUC (0.780) compared to uniform-only (0.754) and pure-boundary (estimated 0.760, since the boundary region is less well-defined without Stage 1 exploration). The key insight is that Stage 1 data is essential for *discovering* where the boundaries are, while Stage 2 data is essential for *precisely mapping* those boundaries. Neither stage alone achieves both objectives.

8.7 Scalability Considerations

The current 8-dimensional parameter space is tractable with 10K–20K experiments per stage. However, more complex tasks may require higher-dimensional spaces (*e.g.*, adding task parameters like target height, approach angle, gripper speed). We analyze how the framework scales with dimensionality.

The LHS space-filling property guarantees that each parameter is uniformly covered regardless of dimensionality, but the *density* of samples per parameter bin decreases as $O(N^{1/d})$. With $N = 10{,}000$ experiments in $d = 8$ dimensions, each parameter has approximately $10{,}000^{1/8} \approx 5.6$ effective samples per percentile—sufficient for coarse boundary detection but not for precise mapping.

For higher-dimensional spaces ($d > 12$), we recommend: (1) increasing N_1 to at least 2,000 × d, (2) using principal component analysis to identify the most important parameter combinations before Stage 2, and (3) considering sequential experimental design approaches that iteratively refine the boundary region.

8.8 Limitations

Several limitations should be acknowledged:

Simulation fidelity. While Isaac Sim provides high-fidelity physics, the sim-to-real gap remains. Our failure boundaries are valid for the simulated environment; real-world validation is needed to confirm their transferability. Prior work on sim-to-real transfer [9] suggests that qualitative trends (which parameters cause failure) transfer well, even when quantitative thresholds differ.

Task scope. The current evaluation is limited to pick-and-place. Extending to more complex manipulation tasks (*e.g.*, insertion, assembly) requires additional scenario definitions and failure mode taxonomies. However, the two-stage sampling methodology is task-agnostic and can be applied to any binary-outcome evaluation.

Parameter independence. LHS assumes that parameters can be varied independently, which may not hold for all real-world scenarios (*e.g.*, mass and size are physically correlated for fixed-density objects). The inclusion of interaction terms in the logistic regression model partially addresses this limitation, but correlated parameter distributions would require copula-based sampling strategies.

Model simplicity. Logistic regression captures linear and pairwise interaction effects but misses higher-order nonlinearities. The AUC of 0.780 suggests room for improvement with more expressive models, at the cost of interpretability. We deliberately prioritize interpretability for industrial deployment, where operators must understand and trust the risk model's predictions.

Single-task evaluation. Both robots are evaluated on the same pick-and-place task, limiting our ability to draw conclusions about task-dependent failure modes. A multi-task evaluation campaign would reveal whether the identified danger zones are task-specific or generalize across manipulation primitives.

8.9 Future Directions

Several promising extensions of this work are worth highlighting:

Active learning integration. The current two-stage approach uses a fixed Stage 1/Stage 2 split. An active learning variant could iteratively allocate experiments to the most uncertain regions, potentially requiring fewer total experiments to achieve the same boundary resolution. Gaussian process-based acquisition functions [14] are a natural choice, though scalability to 8+ dimensions requires approximations such as random Fourier features.

Multi-task evaluation. Extending RoboGate to assembly, insertion, and tool-use tasks would enable cross-task failure analysis. We hypothesize that some danger zones (high mass, low friction) are universal across manipulation primitives, while others (e.g., orientation sensitivity for insertion tasks) are task-specific.

Real-world validation. A critical next step is validating the simulated failure boundaries against real-world experiments. We plan to conduct a 500-experiment real-world campaign on a Franka Panda, sampling from both safe and danger zones as identified by the simulation study. The primary research question is whether the boundary equation (Eq. 7) transfers quantitatively or only qualitatively to the real robot.

VLA model fine-tuning. The failure dictionary produced by RoboGate can serve as targeted training data for VLA models. By fine-tuning on the adversarial conditions where the model fails most severely (low lighting, clutter), we expect significant improvements in robustness. A closed-loop workflow—evaluate with RoboGate, fine-tune on failures, re-evaluate—could accelerate VLA model development for industrial deployment.

Multi-robot fleet validation. Industrial deployments often involve fleets of identical robots that may exhibit unit-to-unit variation. Extending RoboGate to characterize the failure boundary distribution across a fleet (rather than a single robot) would provide fleet-level deployment confidence.

9 Conclusion

We presented RoboGate, a deployment risk management framework for industrial robot policies that combines physics-based simulation with two-stage adaptive sampling to efficiently discover and characterize failure boundaries. Our key findings from 30,000 experiments across two robot platforms are:

1. Boundary-focused sampling improves risk model AUC from 0.754 to 0.780, demonstrating the value of concentrating experiments in the success-failure transition zone.
2. The failure boundary in friction-mass space follows a closed-form equation $\mu^*(m) = (1.469 + 0.419m)/(3.691 - 1.400m)$, enabling direct translation to operational constraints.
3. Four universal danger zones (mass $>$ 0.935 kg) affect both Franka and UR5e platforms, indicating embodiment-independent task difficulty.
4. VLA models (Octo-Small) achieve 0.0% SR on all 68 scenarios vs. 100% for scripted controllers—a 100-point gap driven by 79.4% grasp misses and 20.6% collisions, demonstrating that current generalist VLA models require task-specific fine-tuning for industrial deployment.

Future work will extend RoboGate to multi-step manipulation tasks, incorporate active learning for boundary refinement, and validate the sim-to-real transfer of discovered failure boundaries.

RoboGate is open-source at https://github.com/liveplex-cpu/robogate. The full failure dictionary (30K experiments) is available at https://huggingface.co/datasets/liveplex/robogate-failure-dictionary.

References

[1] Octo Model Team. Octo: An open-source generalist robot policy. In *RSS*, 2024.

[2] M. J. Kim, K. Pertsch, S. Karamcheti, T. Xiao, A. Balakrishna, S. Nair, R. Rafailov, E. Foster, G. Lam, P. Sanketi, Q. Vuong, T. Kollar, B. Burchfiel, R. Tedrake, D. Sadigh, S. Levine, P. Liang, and C. Finn. OpenVLA: An open-source vision-language-action model. *arXiv:2406.09246*, 2024.

[3] A. Brohan, N. Brown, J. Carbajal, Y. Chebotar, X. Chen, K. Choromanski, T. Ding, D. Driess, A. Dubey, C. Finn, *et al.* RT-2: Vision-language-action models transfer web knowledge to robotic control. In *CoRL*, 2023.

[4] K. Black, N. Brown, D. Driess, A. Esmail, M. Equi, C. Finn, N. Fusai, L. Groom, K. Hausman, B. Ichter, *et al.* π_0: A vision-language-action flow model for general robot control. *arXiv:2410.24164*, 2024.

[5] C. Chi, S. Feng, Y. Du, Z. Xu, E. Cousineau, B. Burchfiel, and S. Song. Diffusion policy: Visuomotor policy learning via action diffusion. In *RSS*, 2023.

[6] S. James, Z. Ma, D. R. Arrojo, and A. J. Davison. RLBench: The robot learning benchmark and learning environment. *IEEE RA-L*, 5(2):3019–3026, 2020.

[7] T. Yu, D. Quillen, Z. He, R. Julian, K. Hausman, C. Finn, and S. Levine. Meta-World: A benchmark and evaluation for multi-task and meta reinforcement learning. In *CoRL*, 2020.

[8] B. Liu, Y. Zhu, C. Gao, Y. Feng, Q. Liu, Y. Zhu, and P. Stone. LIBERO: Benchmarking knowledge transfer for lifelong robot learning. In *NeurIPS*, 2024.

[9] J. Tobin, R. Fong, A. Ray, J. Schneider, W. Zaremba, and P. Abbeel. Domain randomization for transferring deep neural networks from simulation to the real world. In *IROS*, 2017.

[10] OpenAI, I. Akkaya, M. Andrychowicz, M. Chociej, M. Litwin, B. McGrew, A. Petron, A. Paino, M. Plappert, G. Powell, R. Ribas, *et al.* Solving Rubik's Cube with a robot hand. *arXiv:1910.07113*, 2019.

[11] F. Muratore, M. Gienger, and J. Peters. Robot learning from randomized simulations: A review. *Frontiers in Robotics and AI*, 9:799893, 2022.

[12] M. Koren, S. Alsaif, R. Lee, and M. J. Kochenderfer. Adaptive stress testing for autonomous vehicles. In *IV*, 2018.

[13] T. Dreossi, D. J. Fremont, S. Ghosh, E. Kim, H. Ravanbakhsh, M. Vázquez-Chanlatte, and S. A. Seshia. VerifAI: A toolkit for the formal design and analysis of artificial intelligence-based systems. In *CAV*, 2019.

[14] K. Chaloner and I. Verdinelli. Bayesian experimental design: A review. *Statistical Science*, 10(3):273–304, 1995.

[15] ISO. ISO 10218-1:2011 Robots and robotic devices—Safety requirements for industrial robots—Part 1: Robots. International Organization for Standardization, 2011.

[16] ISO. ISO/TS 15066:2016 Robots and robotic devices—Collaborative robots. International Organization for Standardization, 2016.

[17] D. Amodei, C. Olah, J. Steinhardt, P. Christiano, J. Schulman, and D. Mané. Concrete problems in AI safety. *arXiv:1606.06565*, 2016.

[18] W. Xu, Y. Chen, D. Held, and Z. Xu. SafeBench: A benchmarking platform for safety evaluation of autonomous vehicles. In *NeurIPS Datasets and Benchmarks*, 2022.

[19] S. Nasiriany, A. Maddukuri, L. Zhang, A. Parikh, A. Lo, A. Joshi, A. Mandlekar, and Y. Zhu. RoboCasa: Large-scale simulation of everyday tasks for generalist robots. *arXiv:2406.02523*, 2024.

A Failure Dictionary Schema

Each experiment in the failure dictionary contains 26 fields (Franka) or 10 fields (UR5e). Table 9 documents the full Franka schema.

Table 9: Franka failure dictionary schema (26 fields per experiment).

Field	**Type**	**Description**
Physical parameters		
friction	float	Friction coefficient [0.05, 1.2]
mass	float	Object mass in kg [0.05, 2.0]
com_offset	float	COM offset [0, 0.4]
size	float	Object size in m [0.02, 0.12]
ik_noise	float	IK noise in rad [0, 0.04]
obstacles	int	Obstacle count [0, 5]
shape	str	box/cylinder/sphere/irregular
placement	str	Placement configuration
Outcome fields		
success	bool	Episode success
failure_type	str	none/timeout/collision/...
cycle_time	float	Episode duration in seconds
collision	bool	Collision occurred
drop	bool	Object dropped
grasp_miss	bool	Grasp missed
Derived fields		
fail_prob	float	Analytical failure probability
zone	str	safe/boundary/danger
sample_idx	int	LHS sample index

B Failure Mode Transition Details

Table 10 shows the dominant failure mode at each friction level, demonstrating the systematic transition from timeout-dominated failures at low friction to collision-dominated failures at high friction.

C UR5e Parameter Space

The UR5e evaluation uses a reduced 5-dimensional parameter space, reflecting the suction gripper's insensitivity to friction and shape parameters.

Table 10: Dominant failure mode by friction bin (Franka 20K). Transition from `timeout` to `collision` occurs at $\mu \approx 0.77$.

Friction Range	**N fail**	**Dominant Mode**	**%**
0.050–0.108	1,968	timeout	39.1%
0.108–0.165	1,069	timeout	44.3%
0.165–0.280	1,298	timeout	42.0%
0.280–0.450	1,514	timeout	38.5%
0.450–0.625	1,067	grip_loss	33.1%
0.625–0.768	1,044	timeout	31.2%
0.768–1.200	2,323	collision	42.8%

Table 11: UR5e parameter space (5 dimensions).

Parameter	**Range**	**Scale**
ik_noise	[0, 0.02] rad	linear
mass	[0.05, 3.0] kg	log
grip_threshold	[0.005, 0.02]	linear
obstacles	[0, 3]	integer
placement	8 configurations	categorical

D VLA Evaluation: Complete Per-Variant Results

Table 12 shows the full per-variant breakdown for the Octo-Small VLA evaluation, sorted by success rate (worst first).

Table 12: Octo-Small VLA: per-variant results on 68 scenarios. SR = success rate.

Category/Variant	**Pass**	**Total**	**SR**	**Primary Fail**
nom/standard_objects	0	7	0%	grasp_miss (6), collision (1)
nom/standard_lighting	0	7	0%	grasp_miss (5), collision (2)
nom/centered_placement	0	6	0%	grasp_miss (5), collision (1)
edge/small_objects	0	3	0%	grasp_miss (2), collision (1)
edge/heavy_objects	0	3	0%	grasp_miss (3)
edge/edge_placement	0	3	0%	grasp_miss (2), collision (1)
edge/occluded_objects	0	3	0%	grasp_miss (2), collision (1)
edge/transparent_objects	0	3	0%	grasp_miss (3)
adv/low_lighting	0	3	0%	grasp_miss (2), collision (1)
adv/cluttered_scene	0	3	0%	grasp_miss (3)
adv/slippery_surface	0	2	0%	grasp_miss (2)
adv/moving_disturbance	0	2	0%	grasp_miss (2)
dr/lighting	0	10	0%	grasp_miss (7), collision (3)
dr/color	0	5	0%	grasp_miss (3), collision (2)
dr/position	0	5	0%	grasp_miss (5)
dr/camera	0	3	0%	grasp_miss (2), collision (1)

E Confidence Score Computation

The RoboGate Confidence Score is a weighted combination of five metric assessments, designed to produce a single deployment readiness number between 0 and 100.

$$C = \sum_{i=1}^{5} w_i \cdot s_i(\mathbf{m}) \quad (8)$$

where w_i are the metric weights and $s_i(\mathbf{m})$ are the normalized scores for each metric:

Table 13: Confidence Score weight allocation and scoring functions.

Component	Metric	Weight	Scoring
Grasp SR	N_s/N	0.30	Linear: $\min(1, \text{SR}/0.92)$
Cycle Time	$\bar{t}_c$	0.20	$1 - \lvert\Delta t/t_{\text{base}}\rvert$
Collision	n_{col}	0.25	1 if $n = 0$, else 0
Edge Cases	SR_{edge}	0.15	Linear: SR on edge scenarios
Baseline Δ	ΔSR	0.10	$\max(0, 1 + \Delta\text{SR})$

The collision component is binary (0 or 25 points)—any collision results in zero contribution from this component, reflecting the zero-tolerance policy for safety-critical events. This design means that a policy with even one collision cannot score above 75/100, regardless of other metrics.

For the VLA evaluation (Section 7), Octo-Small scores 1/100 due to: SR component $= 0.30 \times (0.0/0.92) \times 100 = 0.0$, collision component $= 0.25 \times 0 = 0$ (14 collisions), cycle time component ≈ 0 (71.7s vs. baseline 4.1s), edge case component $= 0.15 \times 0 = 0$ (0% edge SR), and baseline delta $= 0.10 \times 0 = 0$ (-100pp regression). The minimum score of 1.0 reflects the worst possible outcome across all five metrics simultaneously.

F Boundary-Focused Sampling Algorithm Details

The boundary detection algorithm in Stage 2 operates as follows:

The fallback ranges are derived from domain knowledge about typical industrial conditions:

- Friction: [0.05, 0.40] — most industrial objects have low-to-moderate friction
- Mass: [0.3, 2.0] kg — typical payload range for collaborative robots
- COM offset: [0.1, 0.4] — irregular objects with shifted center of mass
- Size: [0.015, 0.05] m — small parts that are difficult to grasp
- IK noise: [0.01, 0.04] rad — typical encoder uncertainty range

The emphasis region ($\mu < 0.3$, $m \geq 0.5$) receives 30% of the Stage 2 budget because preliminary analysis showed the steepest SR gradient in this region (SR changes from 60% to 10% within a narrow friction range at moderate mass).

Algorithm 2 Boundary Region Detection

```
Require: Stage 1 experiments $\mathcal{D}_1$, SR range $[sr_l, sr_h]$
Ensure: Narrowed parameter ranges $\mathcal{R}_{\text{bnd}}$
for each continuous parameter $p \in$ {friction, mass, com_offset, size, ik_noise} do
    Partition $p$ range into 10 equal-width bins $B_1, \ldots, B_{10}$
    for each bin $B_k$ do
        $\text{SR}_k \leftarrow$ success rate of experiments in $B_k$
    end for
    $\mathcal{B}_p \leftarrow \{B_k : sr_l \leq \text{SR}_k \leq sr_h\}$
    if $|\mathcal{B}_p| > 0$ then
        $\mathcal{R}_{\text{bnd}}[p] \leftarrow [\min(\mathcal{B}_p), \max(\mathcal{B}_p)]$
    else
        $\mathcal{R}_{\text{bnd}}[p] \leftarrow$ fallback range from domain knowledge
    end if
end for
return $\mathcal{R}_{\text{bnd}}$
```

G Reproducibility

All experiments are reproducible with the following commands. Seeds are fixed to ensure deterministic results.

```
# Stage 1: Franka uniform (10K)
python scripts/generate_failure_dictionary_large.py \
  --n 10000 --seed 2026

# Stage 1: UR5e uniform (10K)
python scripts/test_ur5e_baseline.py \
  --n 10000 --seed 2026

# Stage 2: Franka boundary (10K)
python scripts/generate_boundary_focused.py \
  --n 10000 --seed 2024 \
  --input failure_dictionary_large.json

# Boundary analysis
python scripts/analyze_boundary_data.py \
  --original failure_dictionary_large.json \
  --boundary franka_boundary_10k.json \
  --ur5e ur5e_failure_dictionary.json

# VLA evaluation (two-process ZMQ pipeline)
# Terminal 1: Isaac Sim server (Python 3.11)
C:\IsaacSim\_build\windows-x86_64\release\python.bat \
  scripts/vla_isaac_server.py --mss-camera --port 5555
# Terminal 2: Octo client (conda octo, Python 3.10)
conda run -n octo python scripts/vla_octo_client.py \
  --model octo-small --port 5555 \
  --output robogate_demo/vla_octo_eval_real.json

# Paper figures
python scripts/generate_paper_figures.py
```

Hardware: NVIDIA RTX 4090 (24 GB VRAM), Intel i9-13900K, 64 GB RAM. Software: Ubuntu 22.04, NVIDIA Isaac Sim 5.1, Python 3.12, NumPy 2.x.