Robust Motion Planning under Sim-to-Real Uncertainty via Active Hypothesis Elimination

June 10, 2025

1 Introduction

Physics-based simulators have become indispensable tools in robotic research and development, offering safe, cost-effective platforms for training controllers and evaluating planning strategies. These simulators model the robot's dynamics and its environment to a high degree of detail, enabling the synthesis of sophisticated behaviors before any hardware is used.

For example, consider a robotic manipulator tasked with transferring a deformable object—such as a piece of cloth or a container of liquid—from one bin to another in a cluttered environment. Simulating this task may involve handling complex contact dynamics, soft-body interactions, or fluid behavior using an engine like NVIDIA Flex, MuJoCo, or PyBullet with customized physics extensions. Within the simulator, it is possible to compute collision-free, dynamically feasible trajectories that succeed under nominal model parameters.

However, transferring these trajectories to real hardware introduces several challenges. First, simulators inevitably suffer from inaccuracies due to unknown physical parameters: friction coefficients, compliance properties, actuator delays, sensor latencies, or fluid viscosity may all differ in reality. Second, real-world observations often exhibit noise and partial observability, while simulators typically offer perfect state feedback. Lastly, simulators cannot fully capture phenomena such as wear-and-tear, thermal effects, or unmodeled nonlinearities.

These discrepancies give rise to what is known as the *sim-to-real gap*, where strategies that perform reliably in simulation may fail unpredictably on the real robot. Traditional methods for closing this gap include domain randomization, system identification, and learning robust policies across wide parameter ranges. However, these approaches often require extensive real-world data or generalization beyond what a planner or policy was trained on.

Our approach. We introduce an algorithmic framework that bridges the sim-to-real gap by treating model uncertainty as a structured planning problem. Instead of learning a universal policy, we leverage a physics simulator to generate candidate plans and iteratively refine our belief over the true system through active real-world execution. Our contributions are as follows:

- We propose a planning strategy based on sampling a finite set of candidate dynamics models and choosing trajectories that maximize the probability of success across this belief set.
- We execute these plans in the real world and use the outcomes to reject inconsistent models—those that fail to match the observed behavior.
- We resample locally around the surviving models to densify the hypothesis set and converge toward the true parameters.
- We formulate this process as a chance-constrained optimization problem and show that, under deterministic dynamics, it yields safety guarantees and provable convergence properties.

This method is particularly appealing in settings where the number of real-world trials is severely limited. It provides a practical, explainable, and computationally tractable pathway to adapt simulation-generated plans to real-world execution with minimal data collection and without retraining large-scale models.

2 Problem Setup

Let $\Theta = \{\theta_1, \ldots, \theta_N\}$ be a finite set of simulation parameters sampled from a prior over plausible real-world dynamics. Let $\mathcal{B}_t \subseteq \Theta$ denote the belief set at iteration t, assumed to contain the true parameter θ^*. At each iteration, we:

1. Plan a trajectory τ_t that maximizes the expected success rate over $\mathcal{B}_t$.

2. Execute τ_t in the real world, observe outcome $o_t \in \{\text{success}, \text{failure}\}$.

3. Eliminate parameters in $\mathcal{B}_t$ that are inconsistent with o_t.

4. Resample new parameters by perturbing those remaining in $\mathcal{B}_t$.

We assume deterministic dynamics and binary outcomes.

3 Algorithm

Algorithm 1 Active Robust Planning via Hypothesis Elimination

Initialize belief set $\mathcal{B}_0 = \{\theta_1, \ldots, \theta_N\}$
for each iteration $t = 0, 1, 2, \ldots$ **do**
  Plan trajectory $\tau_t = \arg\max_\tau \frac{1}{|\mathcal{B}_t|} \sum_{\theta \in \mathcal{B}_t} \mathbb{I}[\text{success}(\tau; \theta)]$
  Execute τ_t in the real world, observe outcome o_t
  Update belief: $\mathcal{B}_{t+1} = \{\theta \in \mathcal{B}_t : \text{sim}(\tau_t; \theta) = o_t\}$
  Resample around $\mathcal{B}_{t+1}$ to replenish sample set

4 Formal Properties and Proof Sketches

We assume that the real world corresponds to some unknown parameter $\theta^* \in \Theta$, and that outcomes from executing a trajectory are deterministic functions of θ.

4.1 Property 1: Safety via Chance Constraints

Proposition 1. *If τ satisfies*

$$\hat{p}_t(\textit{success}(\tau)) := \frac{1}{|\mathcal{B}_t|} \sum_{\theta \in \mathcal{B}_t} \mathbb{I}[\textit{success}(\tau; \theta)] \geq \delta$$

and $\theta^* \in \mathcal{B}_t$, then the probability that τ succeeds on the real system is at least δ.

Sketch. Under the assumption that $\theta^* \sim \text{Uniform}(\mathcal{B}_t)$, the probability of success is the fraction of belief models predicting success. The estimate $\hat{p}_t$ is exact due to determinism. □

4.2 Property 2: Belief Densification

Proposition 2. *Assume resampling perturbs parameters in $\mathcal{B}_t$ using a kernel with local support. Then, for any neighborhood $B_\epsilon(\theta^*)$, the density of samples in $B_\epsilon(\theta^*)$ increases in expectation over time.*

Sketch. Each resampling step draws from a mixture centered on $\mathcal{B}_t$. Since $\theta^* \in \mathcal{B}_t$ and is never eliminated, future samples concentrate near it. □

4.3 Property 3: Parameter Convergence

Proposition 3. *Under the assumptions above, the empirical distribution $p_t(\theta)$ over parameters converges in probability to a distribution $p_\infty(\theta)$ with mean $\mathbb{E}_{p_\infty}[\theta] \approx \theta^*$ and vanishing variance.*

Sketch. Follows from consistent elimination (only incorrect hypotheses are removed) and local resampling. As more observations are made, $p_t(\theta)$ becomes increasingly peaked near θ^*. □

5 Implemention-oriented algorithm

Algorithm 2 Sim2Real Planning with Active Rejection and Resampling

Input: Initial parameters $\Theta = \{\theta_1, \ldots, \theta_N\}$, threshold δ
Initialize belief set $\mathcal{B} \leftarrow \Theta$
for each iteration $t = 0, 1, 2, \ldots$ **do**
  Generate candidate trajectories $\mathcal{T}$
  for each $\tau \in \mathcal{T}$ **do**
    successes $\leftarrow 0$
    for each $\theta \in \mathcal{B}$ **do**
      if simulate(τ, θ) == "success" **then**
        successes $\leftarrow$ successes + 1
    success_rate$[\tau] \leftarrow$ successes / $|\mathcal{B}|$
  $\tau^* \leftarrow \arg\max_{\tau \in \mathcal{T}}$ success_rate$[\tau]$
  if success_rate$[\tau^*] < \delta$ **then**
    break {No safe trajectory found}
  Execute τ^* on real robot, observe $o \in \{\text{success}, \text{failure}\}$
  // **Reject inconsistent parameters**
  $\mathcal{B}_{\text{new}} \leftarrow \{\theta \in \mathcal{B} : \text{simulate}(\tau^*, \theta) = o\}$
  // **Resample to maintain diversity**
  $\mathcal{B} \leftarrow \mathcal{B}_{\text{new}} \cup \text{perturb}(\mathcal{B}_{\text{new}})$