Robotics: Science and Systems 2026
Sydney, Australia, July 13–17, 2026

Betting for Sim-to-Real Performance Evaluation

Yujia Chen*
Department of Computer Science
Iowa State University
Ames, Iowa 50011–1090
Email: yjchen@iastate.edu

Zaid Mahboob*
Department of Computer Science
Iowa State University
Ames, Iowa 50011–1090
Email: zaidm@iastate.edu
*Equal contribution

Bowen Weng†
Department of Computer Science
Iowa State University
Ames, Iowa 50011–1090
Email: bweng@iastate.edu
†Corresponding author

***Abstract*—Roboticists are, in some sense, gamblers: every sim-to-real deployment is a wager that simulation-implied performance will carry over to real-world hardware. If one is gambling anyway, one might as well do it as "professionals". Among the many sim-to-real bets one could place in the robotics context (e.g., policy training), this paper focuses on *performance evaluation*: how to obtain accurate and efficient estimates of real-world robot behavior from limited real-world trials. We formalize this as a sequential betting problem and show that the bet-weighted estimator induced by Kelly-optimal stakes provably outperforms Monte Carlo whenever a bank of simulators carries predictive advantage. The optimal bet, we prove, requires only the one-dimensional payoff distribution rather than a generative model of the real system, in contrast to state-of-the-art sim-to-real evaluation methods that hinge on simulator-to-reality fidelity. We translate this principle into a practical algorithm that maintains a bank of randomized simulators reweighted via Cover's universal portfolio principle, and we prove that wealth growth itself serves as a runtime diagnostic for whether such bets are working. We demonstrate the effectiveness of the proposed methods using both synthetic examples and cross-fidelity computational simulators. Most strikingly, on a SO-ARM101 manipulator it estimates real-world placement accuracy more accurately than Monte Carlo on hardware data, using only synthetic distributions with no kinematic or task-level relation to the robot, a seemingly unconventional sim-to-real transfer that becomes natural and feasible under the proposed betting perspective. Programs for reproducing empirical results are available at https://github.com/ISUSAIL/Bet4Sim2Real.**

DISCLAIMER

In this paper, terms such as "gambler", "gambling", "betting", and "bets" are used strictly as theoretical metaphors within the context of robot testing and performance evaluation. They refer only to abstract mathematical constructs for sequential decision-making and uncertainty modeling. Under no circumstances does this work involve, promote, or relate to real-world financial gambling, monetary transactions, or any form of betting activity, within or outside the robotics context.

I. Introduction

Although rarely stated explicitly, roboticists often behave much like "gamblers" [1], at least occasionally, and in a constructive sense. One develops theory-inspired or data-driven algorithms, controllers, software pipelines, and hardware stacks largely within simulators or synthetic computational-aided environments, and then deploys them on real machines with the hope, though rarely with a formal guarantee, that performance will carry over. This transfer is guided by an implicit and continually updated notion of confidence: as real-world data accumulate, our trust in the simulator's predictions is revised, reinforced, or corrected. To the best of the authors' knowledge, this paper represents the first attempt to *formalize and theorize* this longstanding practice of "*betting*" within sim-to-real workflows in robotics.

A. The sim-to-real performance evaluation problem

We focus on a specific yet fundamental aspect of sim-to-real transfer: *performance evaluation* [2, 3, 4, 5, 6]. Given a *fixed* policy, controller, or robotic system, whether model-based, learning-based, or hybrid, as it stands, one seeks to *efficiently* obtain an *accurate* estimate of its expected performance under a certain real-world environment. This challenge arises in almost every corner of robotics: benchmarking learning algorithms [7, 8, 9], evaluating safety-critical behaviors [10, 11], validating controllers [12, 13], comparing algorithmic or policy variants [14, 15], or supporting certification and regulatory processes [6, 13, 16, 17, 18].

Formally speaking, consider a probability distribution P over the measurable space $(\mathcal{X}, \mathcal{F})$. Let us further assume, albeit somewhat unrealistically in general but quite common in robotics, that P represents an *expensive* real-world data-generating distribution of various states concerning the robot being tested as well as environmental factors.

Given a bounded performance scoring function $\psi : \mathcal{X} \to \mathcal{M} \subset \mathbb{R}$, consider the performance evaluation target of inference as the scalar mean

$$\mu \triangleq \mathbb{E}_{x\sim P}[\psi(x)]. \tag{1}$$

When ψ is a Boolean indicator of failure versus non-failure, μ reduces to the risk (i.e., the failure probability) [2, 6, 11]. In practice, many other performance measures fall within the same expectation-based formulation, such as normalized rewards [19, 20], bounded stability scores [10, 21], or clipped cost signals. W.l.o.g., we take $\mathcal{M} = [0, 1]$ (via an affine rescaling/normalization of ψ if necessary [22]) for the remainder of the analysis.

This work was supported in part by Presidential Interdisciplinary Research Seed Grant at Iowa State University and in part by financial assistance award 70NANB25H125 from U.S. Department of Commerce, National Institute of Standards and Technology.

In general, to estimate the mean μ, we have the Monte Carlo inference [23, 24, 25] as

$$\hat{\mu}_{\text{MC}} = \frac{1}{n}\sum_{1}^{n}\psi(x_i), x_i \sim P \text{ i.i.d.} \tag{2}$$

Despite the theoretical completeness of Monte Carlo estimation (e.g., $\lim_{n\to\infty}\hat{\mu}_{\text{MC}} = \mu$ almost surely [24, 26]), plain Monte Carlo is often not a practically "efficient" evaluation method, especially when extreme-performance outcomes occur with very small probability under P (the well-known long-tail or "curse of rarity" phenomenon [27]). This inefficiency becomes particularly pronounced when P denotes the distribution of states induced by *real-world* trials on *real robots*, where tests are costly, time-consuming, safety-constrained, and often subject to hardware wear, environmental variability, or operational risk [5, 11, 28, 29].

Simulators, therefore, could play a crucial role. They offer abundant, inexpensive, and controlled observations that can supplement or guide real-world testing. Consider a parameterized *simulator* as a family of other distributions $\{Q_\theta : \theta \in \Theta\}$ over the same measurable space mentioned above. By configuring θ differently, one can have different distributions of Q. Contrary to the real-world distribution P, let's further assume Q_θ corresponds to a family of relatively "cheaper" distributions from which we can affordably draw large numbers of samples. This setup motivates our central question: *can we leverage the low cost and flexible configurability of Q_θ to perform accurate and efficient inference on P with theoretical guarantees?*

B. Literature review

We note that this paper is not the first attempt to address the above research question. A substantial body of prior work has explored related directions, generally falling into two complementary perspectives [30]: (i) *variance reduction*, and (ii) *bias correction*.

The variance-reduction perspective typically relies on *weighted sampling*, with *importance sampling* (IS) as a canonical example [31, 32, 33]:

$$\hat{\mu}_{\text{IS}} = \frac{1}{n}\sum_{i=1}^{n}\psi(x_i)\frac{p(x_i)}{p'(x_i)}, \qquad x_i \sim P' \text{ i.i.d.} \tag{3}$$

Here, the auxiliary distribution P' is chosen to oversample regions that contribute most to μ. Under standard regularity conditions [32, 34], IS yields unbiased estimators with reduced variance relative to vanilla Monte Carlo (2). In robotics, however, both P and P' typically correspond to real-world data collection, with the simulator Q_θ serving mainly to guide the construction of a better P' [2, 10, 12, 35]. Consequently, the most effective simulator under this view is one that closely matches P, since tighter alignment enables more accurate variance reduction.

Bias-correction approaches instead seek to implicitly [36] or explicitly [3] model and correct the discrepancy between the real distribution P and a simulator Q_θ. These methods interpret the bias $\mathbb{E}_P[\psi] - \mathbb{E}_{Q_\theta}[\psi]$ as a structured, sample-dependent quantity that can be learned or parameterized. Concretely, one estimates a correction function $b_\theta(x)$, for example via paired real–sim or predictor–real evaluations as in prediction-powered inference (PPI) [37, 38], or via learned correlators or control variates [39, 40]. The correction is then used to construct a bias-adjusted estimator, either by evaluating corrected simulator samples or by calibrating predictor-based estimates using real data, yielding an estimator of the generic form

$$\hat{\mu}_{\text{BC}} = \frac{1}{n}\sum_{i=1}^{n}\big(\psi(x_i) + b_\theta(x_i)\big), \qquad x_i \sim Q_\theta \text{ i.i.d.}, \tag{4}$$

where this expression represents a canonical simulator-centric bias-correction form; predictor-based variants such as PPI can be written in an equivalent correction-based form. Although conceptually distinct from IS, bias-correction methods share a similar implication: their theoretical optimum favors $Q_\theta = P$, in which the correction term b_θ vanishes and the estimator becomes an unbiased Monte-Carlo estimator under P.

Note neither approach mentioned above fundamentally exploits the scalability of modern robotic simulation: simulated data grows cheaply and rapidly, while real-world data remains scarce, preventing simulation abundance from translating into proportional gains in reliable real-world performance estimation [11, 29, 32, 37]. Moreover, from a theoretical standpoint, these lines of work suggest a common conclusion: for *performance evaluation*, a "good" simulator is one that closely approximates reality. Yet this conclusion contrasts sharply with another highly successful paradigm in sim-to-real robotics. In *policy transfer*, where the policy is continually learned and adapted in the hope of achieving high performance in the real world, "wrong" but diverged simulators have enabled dramatic progress by deliberately embracing mismatch. Techniques like domain randomization [20, 41, 42, 43, 44, 45, 46], adversarial perturbations [10, 47], and robustness-oriented training inject structured variability to promote generalization across the reality gap, despite limited theoretical understanding [3, 43].

What if the very ingredients that drive the success of robust sim-to-real policy transfer, its deliberate distributional mismatch, structured variability, and controlled uncertainty, are in fact *beneficial* for performance evaluation as well, and in certain regimes even *preferable* to perfect simulator alignment? Such a possibility appears incompatible with classical variance-reduction or bias-correction viewpoints, which rely on minimizing or compensating for simulator–reality discrepancies. The *betting* perspective introduced in this paper suggests a different interpretation. Within a betting framework, predictive advantage, rather than perfect fidelity, is what matters. A simulator that is "wrong but informative", because it induces directional predictive signals or structured uncertainty, may provide a stronger edge than one that attempts to mimic reality exactly but carries no exploitable variance. This re-framing opens the door to a unified viewpoint in which tools traditionally associated with policy transfer (e.g., domain randomization) may also play a principled and theoretically

grounded role towards (provably) accurate and efficient sim-to-real *performance evaluation*.

C. Main contributions

To the best of the authors' knowledge, this paper makes the first attempt at formulating the sim-to-real performance evaluation problem as a *betting* problem.

- We first introduce a *sequential betting formalism* (Algorithm 1) for sim-to-real performance inference, in which simulator-informed bets induce a *bet-weighted estimator* of the target performance measure (1).
- We then show that, under this mechanism, the classical Kelly framework for optimal betting approximately *coincides* with the statistically optimal strategy for efficient estimation (Theorems 1 and 2). That is, the strategy that maximizes wealth growth through betting is exactly the one that achieves the best sampling efficiency and estimation accuracy for the robot's real-world performance under our proposed framework.
- We translate these theoretical insights into a practically executable sim-to-real testing algorithm inspired by *Cover's universal portfolio principle* (Algorithm 2). By maintaining and adaptively reweighting a bank of randomized simulators using proper scoring rules, the proposed method systematically approximates the classical Kelly betting strategy.
- We further prove that wealth growth itself provides a diagnostic signal for assessing the quality of this approximation (Theorem 3), thereby endowing the practical algorithm with theoretical performance guarantees in real settings.
- Finally, we demonstrate the effectiveness and robustness of the proposed framework through a series of examples, ranging from controlled synthetic experiments to real robotic applications in legged locomotion and manipulation across both simulated and physical environments.

The proposed framework fundamentally leverages the scalability of modern simulation in robotics and offers a new theoretical perspective on understanding the role of domain randomization and simulation variance in real-world performance evaluation.

D. Terminology from betting and finance

Throughout this paper we adopt a few standard terms from the literature on sequential betting and finance. For clarity, we summarize them here in plain language.

- Bet: In finance and gambling, a *bet* is a stake placed on an uncertain outcome (e.g., wagering \$10 on a stock going up), with the size of the stake expressing how confident the bettor is. In this paper, we keep the confidence interpretation but drop the money: a bet is a quantitative expression of how strongly to rely on a single prediction at a given round, with no monetary stakes.
- Payoff: In finance, a *payoff* is the return a bettor receives once the outcome is realized — positive if the bet was correct (e.g., \$30 back from a \$10 bet at 3:1 odds), negative otherwise. We preserve the sign convention: the round's payoff is positive when the prediction pointed in the correct direction and negative when it did not, with magnitude reflecting how large the prediction error was (see Algorithm 1 later).
- Edge: In finance, a strategy has *edge* when, on average, its expected return is strictly positive (e.g., a trading model whose predictions are right more often than chance has edge over a coin-flip strategy). We keep this meaning unchanged: a strategy has edge when its expected payoff conditioned on the past is strictly positive. "No edge" means no useful predictive signal beyond chance.
- Predictive Advantage: This is essentially the same as *edge* above. But in this specific paper, when stated more generally: a simulator has *predictive advantage* when its output gives us useful information about real-world performance. This is different from how simulators are usually treated in variance reduction or bias-correction as mentioned above. Theorem 1 makes this precise.
- Wealth. In a typical betting process, wealth is a running total of capital that compounds when bets pay off and depletes when they don't (e.g., start with \$100, win three 1:1 bets in a row, end with \$800). In this paper, we preserve the accumulation mechanism but not the monetary interpretation: wealth here is a bookkeeping diagnostic that summarizes how informative the strategy has been so far. It is not a weight on simulators; rather, sustained wealth growth provides statistical evidence against the absence of edge (see Theorem 3 later).

Notations. The set of real numbers is denoted by $\mathbb{R}$. $\mathbb{Z}$ denotes the set of positive integers and $\mathbb{Z}_k = \{1, \dots, k\}$ for some $k \in \mathbb{Z}$. $|S|$ denotes the cardinality of a set S. $a \wedge b = \min(a, b)$ for some $a, b \in \mathbb{R}$.

II. Main Method

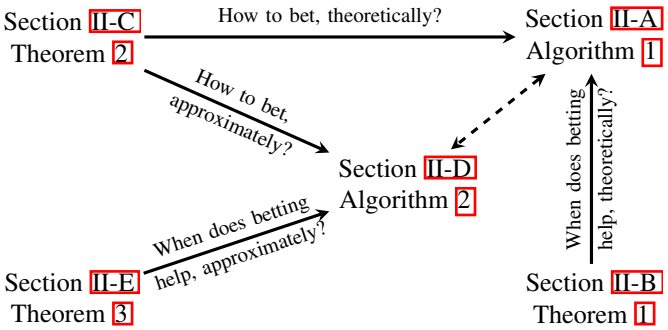

Fig. 1: A roadmap of Section II. Theoretical results (Theorems 1–3) establish when and how betting improves estimation; algorithms translate these insights into practice. Arrows show logical dependencies.

Proofs of all Theorems in this section can be found in the supplementary material.

A. An abstract sequential betting algorithm

An abstract version of the proposed sequential betting algorithm for sim-to-real performance inference is given in Algorithm 1. It places a standard *betting* mechanism at its core. At each round t, before observing the real sample but after drawing auxiliary information from the accumulated simulator

samples, a bet b_t is chosen as a fraction of the (future) payoff, reflecting the algorithm's belief about the correctness of a forthcoming prediction (lines 5–8). If the prediction is subsequently verified by the real outcome, the gambler's *wealth* increases by the corresponding payoff Y_t; if not, the wealth decreases proportionally (lines 12–13). If $b_t = 0$ for all t, the bet-weighted estimator is undefined; setting $b_t = 1$ for all t recovers the Monte Carlo estimator. It is important to point out that this *betting* setup does reflect common practices in robotic sim-to-real workflows, in some conceptual respects. Practitioners routinely decide whether real-world tests are worthwhile based on simulation confidence [3, 43, 48]. As real-world samples accumulate, they also jointly affects the utilization of the simulator and the resultant bet.

On the other hand, the algorithm also departs from standard betting practice and robot performance evaluation in two important ways. The first atypical component is the *double betting* mechanism (lines 5-8) (conceptually close to [49]), which determines a "directional" bet on the correlation between the upcoming real-world sample and the running bet-weighted mean of the previously observed real-world measurements. This mechanism is intentionally introduced to align the betting procedure with the performance–estimation objectives of this paper. Moreover, for the same reason, the algorithm outputs a *bet-weighted estimate* (line 15) in place of the more traditional equally weighted estimator such as the Monte Carlo mean in (2).

Finally, Algorithm 1 is not immediately executable as stated, since critical implementation details remain unspecified. The determination of bet sizes b_t is described only conceptually as being "based on S_t and $\mathcal{D}$" without concrete formulas. Despite these gaps (which shall all be detailed in Section II-D later), we first seek to address a more fundamental question: under what conditions does the sequential betting framework, as abstractly described in Algorithm 1, outperform the Monte Carlo estimator? Our approach is to impose assumptions directly on the *properties* of the betting sequence rather than prescribing specific mechanisms for generating these bets. **This separates the question of *(i) when betting helps* (a statistical question about bet properties) from *(ii) how to generate good bets* (an algorithmic design question).**

Algorithm 1 Abstract Betting

1: **Given:** P, Q_θ, $\psi : \mathcal{X} \to \mathcal{M}$, $T \in \mathbb{Z}$
2: **Initialize:** $\mathcal{D}=\emptyset$, $\tau_0 \in \mathcal{M}$, Wealth $W = 1$, $S_0 = \emptyset$
3: **For** $t = 1, 2, \ldots, T$:
4: Sample $\tilde{S}_t \sim Q_\theta$, $S_t \leftarrow S_{t-1} \cup \tilde{S}_t$
5: Based on S_t and $\mathcal{D}$:
6: compute bet $b_t^{\uparrow} \in [0,1]$ on $\psi(x_t) > \tau_{t-1}$
7: compute bet $b_t^{\downarrow} \in [0,1]$ on $\psi(x_t) < \tau_{t-1}$
8: $b_t = \max\{b_t^{\uparrow}, b_t^{\downarrow}\}$
9: **If** $b_t > 0$:
10: Sample $x_t \sim P$
11: $\mathcal{D}.\texttt{append}((x_t, b_t))$
12: $Y_t = \begin{cases} |\psi(x_t) - \tau_{t-1}|, & \text{if the bet is correct,} \\ -|\psi(x_t) - \tau_{t-1}|, & \text{otherwise.} \end{cases}$
13: $W \leftarrow W(1 + b_t Y_t)$
14: $\tau_t = \sum_{(x,b)\in\mathcal{D}} b \cdot \psi(x) / \sum_{(x,b)\in\mathcal{D}} b$
15: **Output:** $\hat{\mu}_{BW} = \tau_t$

B. When does betting help, theoretically?

Regardless of how the sequence of bets is generated, the following theorem establishes a universal performance guarantee concerning bet-weighted performance estimate. It precisely characterizes when the bet-weighted estimator (line 15 of Algorithm 1) achieves higher efficiency, and therefore greater accuracy, than the standard Monte Carlo estimator (2).

Theorem 1 (Bet-weighted inference V.S. Monte Carlo efficiency)**.** *Consider the abstract sequential betting algorithm (Algorithm 1) with the bet-weighted estimator* $\hat{\mu}_{\mathrm{BW}} = \sum_{t=1}^{T} w_t \psi(x_t)$ *where* $w_t = b_t / \sum_j b_j$*. Correspondingly, the Monte Carlo estimator* $\hat{\mu}_{MC}$ *is defined as in (2) and the target mean* μ *in (1). For a fixed sample budget* n *from* P*, the bet-weighted estimator achieves lower expected squared error* $\mathbb{E}[(\hat{\mu}_{\mathrm{BW}} - \mu)^2] < \mathbb{E}[(\hat{\mu}_{MC} - \mu)^2]$ *if and only if:*

$$\underbrace{\frac{\sigma^2}{n} - \sum_{t=1}^{n} w_t^2 \sigma_t^2}_{\textit{Variance Reduction}} > \underbrace{\left[\sum_{t=1}^{n} w_t \beta(w_t)\right]^2}_{\textit{Bias Penalty}}, \tag{5}$$

where $\sigma_t^2 = Var(\psi(x_t)|w_t)$ *is the conditional variance given the weight,* $\beta(w) = \mathbb{E}[\psi(x_t)|w_t = w] - \mu$ *is the weight–outcome dependence term, and* $\sigma^2 = Var_{x\sim P}[\psi(x)]$ *is the variance of the measure function from the distribution* P*.*

To this end, the central question becomes how to design sequential double–betting procedures that realize the variance–reduction guarantees established in Theorem 1. Concretely, this requires instantiating the bet-generation components left unspecified in Algorithm 1.

C. How to make good bets, theoretically?

If the discussion were purely about betting rather than robotics, the answer would be straightforward: use the classical Kelly principle [50, 51]. Kelly betting, originally developed in information theory [50] and later adopted in finance and gambling [51, 52], addresses a simple yet fundamental question: *"When repeatedly placing bets that are slightly favorable, what fraction of my wealth should I wager each time so that my wealth grows as quickly as possible while controlling risk?"*

The central insight is that one should wager *more* when the perceived advantage is high and *less* when the outcome uncertainty is large. In its most compact form, the Kelly rule prescribes bet size $\propto$ advantage/uncertainty (variance), which is precisely an inverse-variance scaling [52, 53].

A slightly more formal and sequential view of the Kelly principle comes from a small-stakes (Taylor-expanded) and small-edge approximation applied conditionally on past information [50, 54, 55]. Consider a betting process similar to Algorithm 1 in which, at round t, a fraction $b_t \in [0,1]$ is

wagered and a random payoff $Y_t \in \mathbb{R}$ is received. Conditioned on the past history $\mathcal{F}_{t-1}$, the log-wealth increment admits the second-order Taylor expansion

$$\log(1 + b_t Y_t) \;=\; b_t Y_t - \frac{1}{2} b_t^2 Y_t^2 + O(|b_t Y_t|^3), \tag{6}$$

which is accurate when the small-stakes condition $|b_t Y_t| \ll 1$ holds. Taking conditional expectations and retaining the leading terms yields the approximate conditional log-growth

$$b_t \, \mathbb{E}[Y_t \mid \mathcal{F}_{t-1}] \;-\; \frac{1}{2} b_t^2 \, \mathbb{E}[Y_t^2 \mid \mathcal{F}_{t-1}]. \tag{7}$$

Differentiating and setting the derivative to zero gives

$$b_t^* \;\approx\; \frac{\mathbb{E}[Y_t \mid \mathcal{F}_{t-1}]}{\mathbb{E}[Y_t^2 \mid \mathcal{F}_{t-1}]} = \frac{\mu_t}{\sigma_t^2 + \mu_t^2} \approx \frac{\mu_t}{\sigma_t^2}, \tag{8}$$

where the final approximation follows under the small-edge condition $\mu_t^2 \ll \sigma_t^2$ [55]. Note the small-stake and small-edge assumptions are classical in the information-theoretic and financial settings where the Kelly criterion was originally developed [50, 54]. In the robotics performance-evaluation setting considered here, these conditions are even easier to satisfy, since the "wagers" are not real monetary stakes but mathematical constructs that can be scaled arbitrarily without affecting the underlying performance measure.

The remainder of this section will show that this same principle underlies the optimal variance-reduction strategy in our setting. In fact, we will see that the Kelly-optimal betting rule approximately coincides with the choice of weights that minimizes the estimation error of the bet-weighted mean, thereby connecting successful wealth growth to improved statistical efficiency of Algorithm 1.

Theorem 2 (Kelly-style betting induces optimal bet-weighted inference)**.** *Let $\hat{\mu}_{\mathrm{BW}} = \sum_{t=1}^{T} w_t \, \psi(x_t)$ be the bet-weighted estimator of Theorem 1, where $w_t = b_t / \sum_{j=1}^{T} b_j$ and $\sigma_t^2 = \mathrm{Var}(\psi(x_t) \mid \mathcal{F}_{t-1})$. Suppose that at each round t the bet size b_t is chosen according to the Taylor-expanded Kelly rule (8), so that b_t is proportional to $\mathbb{E}[Y_t \mid \mathcal{F}_{t-1}]/\mathrm{Var}(Y_t \mid \mathcal{F}_{t-1})$, which we denote by κ_{t-1}/σ_t^2 with $\kappa_{t-1} := \mathbb{E}[Y_t \mid \mathcal{F}_{t-1}]$. Then the induced normalized weights satisfy $w_t \propto \kappa_{t-1}/\sigma_t^2$. Moreover, if the predictive edge is (approximately) constant, i.e., $\kappa_{t-1} \approx \kappa$ for all t, then the induced weights reduce (approximately) to inverse-variance weighting, $w_t \propto 1/\sigma_t^2$, hence they (approximately) maximize the variance-reduction term $\sigma^2/T - \sum_{t=1}^{T} w_t^2 \sigma_t^2$ in (5).*

Theorem 2 applies to a broad class of Kelly-style updates with κ_{t-1} as an $\mathcal{F}_{t-1}$-measurable "edge" factor and σ_t^2 as the conditional outcome variance. The constant-edge approximation $\kappa_{t-1} \approx \kappa$ holds most accurately in early-to-mid rounds when $|\tau_t - \mu|$ remains large; as $\tau_t \to \mu$, the edge diminishes and Kelly betting naturally reduces stakes, transitioning the estimator toward stability rather than aggressive weighting. In the specific case arising in Algorithm 1, the payoff is defined as $Y_t \;=\; B_t \, (\psi(x_t) - \tau_{t-1})$, where $B_t \in \{+1, -1\}$ encodes whether the algorithm bets "up" or "down". Under this *regression*-style payoff, $\mathrm{Var}(Y_t \mid \mathcal{F}_{t-1}) = \mathrm{Var}(\psi(x_t) \mid \mathcal{F}_{t-1}) = \sigma_t^2$, so the Kelly denominator coincides exactly with the variance term used in Theorem 1. The Kelly "edge" then becomes

$$\kappa_{t-1} = B_t\big(\mathbb{E}[\psi(x_t) \mid \mathcal{F}_{t-1}] - \tau_{t-1}\big) = B_t(\mu - \tau_{t-1}). \tag{9}$$

A practical formulation of Kelly betting is expressed as

$$b_t = \lambda \frac{\kappa_{t-1}}{\sigma_t^2}. \tag{10}$$

$\lambda \in (0, 1]$ is known as the *Kelly fraction*, controlling the level of risk exposure [56]. The choice $\lambda = 1$ corresponds to the *full Kelly* strategy, which maximizes asymptotic log-wealth growth [52], while $\lambda = 0.5$ is often referred to as the *half-Kelly* strategy, offering more conservative behavior with reduced variance and improved robustness in finite-sample settings.

Remark 1. *It may be tempting to interpret the ideal Kelly update (10) as implicitly selecting the "true" real-world simulator, since the optimal bet (10) uses the real distribution. This interpretation is* incorrect. *Kelly betting requires only the* payoff distribution *of $\psi(x_t) - \tau_{t-1}$, specifically its conditional mean and variance, and does not depend on any generative or dynamical model of the real system. Thus, even in the ideal case, the object that Kelly optimizes against is not a simulator of reality, but the one-dimensional payoff law.*

From a practical standpoint, however, the "ideal" Kelly rule (10) is not directly implementable, since neither the true predictive advantage nor the conditional variances σ_t^2 are known a priori. More ironically, this creates a circular dependency: constructing an accurate estimate of μ would itself require prior knowledge of μ. This limitation was already implicit in Kelly's original formulation [50], which assumes that the gambler knows the true odds or outcome probabilities, an assumption rarely satisfied in practice. In the sim-to-real performance evaluation problem studied in this paper, this is precisely where the simulator re-enters the picture.

D. How to make good bets, approximately?

In the attempt to *approximate* the "ideal" Kelly conditions, our main proposal is, again, inspired by classical ideas in finance and gambling, most notably *Cover's universal portfolio principle* [57, 58]. By maintaining and updating a bank of experts, each representing a distinct simulator or predictive hypothesis, the algorithm adaptively reweights these experts according to their observed betting performance. This universal-portfolio–style aggregation admits a classical guarantee [59, 60, 61]: its cumulative performance is within logarithmic regret of the best expert in the bank [60], despite that expert being unknown a priori (formal adaptation to our setting is left to future work).

Specifically, consider a bank of simulator instances $\{Q_{\theta^{(k)}}\}_{k=1}^{K}$ with K "expert" betting strategies. The k-th expert can be instantiated by any lightweight simulator configuration or update rule, for example, a domain-randomized parameter setting, an adversarially stressed model instance, or any other plausible (or even not necessarily plausible) hypothesis about the real-world dynamics. Each expert proposes a predictive

distribution for the performance measure, summarized by $\mu^{(k)} := \mathbb{E}_{Q_{\theta^{(k)}}}[\psi(x)]$ and $(\sigma^{(k)})^2 := \mathrm{Var}_{Q_{\theta^{(k)}}}(\psi(x))$. The universal portfolio idea then maintains a *score-weighted mixture* over all experts, allowing the algorithm to asymptotically compete with the best simulator in hindsight without knowing in advance which simulator is informative. Consistent with both Cover's construction and classical gambling theory, each simulator is assigned a nonnegative weight $\pi_t^{(k)}$ with $\sum_k \pi_t^{(k)} = 1$. A weighted simulator mixture is then used as an approximation of the ideal Kelly as

$$m_t = \sum_{k=1}^{K} \pi_t^{(k)} \mu^{(k)}, v_t = \sum_{k=1}^{K} \pi_t^{(k)} (\sigma^{(k)})^2, b_t = \frac{\lambda |m_t - \tau_{t-1}|}{v_t} \wedge 1, \tag{11}$$

with betting direction $B_t = \mathrm{sign}(m_t - \tau_{t-1})$. Here $\lambda \in (0, 1]$ is the Kelly fraction. This produces a *single combined bet* at each round, exactly as in Algorithm 1.

After observing the real-world outcome $y_t = \psi(x_t)$, for each simulator instance k, we maintain a cumulative log-score $s_t^{(k)}$ updated as

$$s_t^{(k)} = s_{t-1}^{(k)} - \eta\, \ell\left(y_t; \mu^{(k)}, (\sigma^{(k)})^2\right), \tag{12}$$

where $\eta > 0$ is a positive learning-rate (inverse-temperature) parameter that controls how aggressively evidence from real-world outcomes is accumulated, analogous to the learning-rate or risk-sensitivity parameters in classical exponential-weights algorithms and betting-based sequential decision frameworks [54, 61, 62]. $\ell(\cdot)$ is the Gaussian log-score $\ell(y; \mu, \sigma^2) = \frac{1}{2}\left[\log(2\pi\sigma^2) + \frac{(y-\mu)^2}{\sigma^2}\right]$, i.e., the negative log-likelihood of y under the normal model $\mathcal{N}(\mu, \sigma^2)$. Note the Gaussian log-score is standard in universal portfolios [57]; alternative scores (Beta, Brier [63]) are left to future work. The normalized trust weights are then obtained via softmax normalization,

$$\pi_t^{(k)} = \frac{\exp(s_t^{(k)})}{\sum_{j=1}^{K} \exp(s_t^{(j)})}. \tag{13}$$

Importantly, this update does not seek to identify a simulator that matches the real world; rather, it maintains a distribution over simulators that are most informative for predicting real-world outcomes, even when all simulators are misspecified.

Putting the above components together yields Algorithm 2, an explicitly executable instantiation of Algorithm 1. At each round, simulator experts are updated via a universal score-weighted mixture, their predictive moments are aggregated into (m_t, v_t), a Kelly-style bet b_t is computed using (11), and the real-world payoff is incorporated into the bet-weighted estimator τ_t. No other changes are made to Algorithm 1 concerning the abstract betting loop, payoff, or estimator; only the approximation of the Kelly quantities is specified.

Finally, although Algorithm 2 assumes a fixed simulator bank, an important direction for future work is to adaptively revise simulators when performance degrades, for instance when sustained declines in wealth are observed. As in finance, such behavior may indicate model misspecification, but in sim-to-real settings the appropriate update mechanism is inherently application-dependent. This leads to the final challenge addressed in this section: how do we know, in practice, when the approximate bet is actually working?

Algorithm 2 Approximated Betting

Given: P, $\{Q_{\theta^{(k)}}\}_{k=1}^{K}$ for some $K \in \mathbb{Z}$, $\psi : \mathcal{X} \to \mathcal{M}$, $T \in \mathbb{Z}, \eta > 0$
Initialize: $\mathcal{D} = \emptyset$, $\tau_0 \in \mathcal{M}$, $W = 1$, $S_0 = \emptyset$, $s_0^{(k)} = 0$, $\pi_0^{(k)}, \forall k$ s.t. $\sum_{k=1}^{K} \pi_0^{(k)} = 1$
For $t = 1, 2, \ldots, T$:
  Sample $\tilde{S}_t^{(k)} \sim Q_{\theta^{(k)}}$ for each k, $S_t \leftarrow S_{t-1} \cup \bigcup_{k=1}^{K} \tilde{S}_t^{(k)}$
  Based on S_t and $\mathcal{D}$:
    compute m_t, v_t, b_t *by* (11)
    compute bet direction $\mathit{sign}(m_t - \tau_{t-1})$
  If $b_t > 0$:
    line 10-14 in Algorithm 1
    Update log-score $s_t^{(k)}$ *by* (12) *for all* $k \in \mathbb{Z}_K$
    Update weights $\pi_t^{(k)}$ *by* (13) *for all* $k \in \mathbb{Z}_K$
Output: $\hat{\mu}_{\mathrm{BW}} = \tau_t$

E. When does betting help, approximately?

One possible approach is to invoke Theorem 1, provided that the required variance and bias terms can be estimated with provable guarantees. While this may be feasible in principle, we were unable to obtain such estimates in this paper and will consider it as future work. The answer we provide is remarkably simple yet deeply characteristic of the betting perspective: *wealth*.

Theorem 3 (Growing wealth implies effective estimate, statistically)**.** *Let $(\mathcal{F}_t)_{t=0}^{T}$ be a filtration and let $(Y_t)_{t=1}^{T}$ be the realized payoff of the betting strategy at round t (as defined in Algorithm 1, line 12), with $Y_t \geq -1$ almost surely. At each round t, a predictable betting strategy (measurable w.r.t. $\mathcal{F}_{t-1}$) chooses a stake $b_t \in [0, 1]$, then observes Y_t and updates the wealth via $W_0 = 1, W_{t+1} = W_t(1 + b_t Y_t), t = 0, \ldots, T-1$. Suppose the following* no-advantage (no-edge) null hypothesis *holds:*

$$H_0 : \quad \mathbb{E}[Y_t \mid \mathcal{F}_{t-1}] \leq 0 \quad \textit{for all } t = 1, \ldots, T, \tag{14}$$

i.e., conditioned on the past, the betting strategy has no systematic positive expected gain. Then for every $\alpha \in (0, 1)$,

$$\mathbb{P}_{H_0}\left(W_T \geq 1/\alpha\right) \leq \alpha. \tag{15}$$

The theorem itself relies only on the no-edge condition (14) and makes no assumptions about how the bets are generated [64, 65]. When combined with Theorem 1 and the Kelly inverse-variance equivalence, this motivates interpreting H_0 as the "bad" regime in which the betting strategy has no predictive advantage and the bet-weighted estimator offers no improvement over the Monte Carlo estimator.

Remark 2 (Interpreting wealth growth)**.** *The guarantee (15) should not be interpreted as a probability of outperforming*

Monte Carlo (i.e., observing wealth w does not imply improvement with probability $1-1/w$). Instead, sustained wealth growth provides increasing evidence that the betting strategy is extracting predictive signal from the data, indicating operation in a regime where bet-weighted estimation is expected to outperform plain Monte Carlo. A more refined probabilistic interpretation, for example via sequential factorization of wealth growth, is left to future work.

Remark 3 (Why a diagnostic, not a regret bound)**.** *Theorem 3 is a finite-sample, distribution-free statement as (15) holds for every T under the stated null, with no asymptotic limit and no assumptions beyond $Y_t \geq -1$. It is also a diagnostic signal applied* after *or alongside the execution of the betting. A natural alternative would be a regret bound for Algorithm 2 via Cover's universal portfolio framework [57, 58], comparing the wealth of the score-weighted mixture (11)–(13) to that of the best single expert in hindsight. Such a bound is achievable in principle and is left to future work; however, regret bounds, even when finite-sample, are typically worst-case in nature and may not be tight at the small sample budgets relevant to real-robot testing. We therefore present Theorem 3 as a per-trial empirical signal that is directly actionable.*

F. Discussions

The proposed method addresses a different aspect of the performance-evaluation problem than variance reduction (e.g., IS) or bias-correction (e.g., PPI) approaches. IS relies on likelihood ratios reweighting the sampling process itself, while bias-correction methods, such as PPI, use real data to calibrate or debias estimates so that simulator outputs approximate the real distribution. Our betting framework instead treats simulators as sources of predictive advantage: simulator bias is neither estimated nor removed, but explicitly tolerated and traded against variance when informative edge is present. From this perspective, betting complements IS and bias-correction methods by operating in regimes where simulator fidelity is limited but simulation is abundant. In particular, unlike existing approaches that are bottlenecked by scarce real-world data, the proposed framework directly embraces arbitrarily large, even mismatched, simulator banks and can benefit from their diversity. The framework also admits natural composition with IS and PPI; we leave such hybrids to future work.

The method involves only two interpretable hyperparameters: the learning rate $\eta \in \mathbb{R}_{\geq 0}$ and the Kelly fraction $\lambda \in (0, 1]$. Empirical sensitivity to their choices is examined in Section III. A more complete theoretical characterization is of future interest.

Remark 4. *The construction in Algorithm 2 does not require any expert $Q_{\theta^{(k)}}$ to match P at the testing environment, task, or robot level. Theorem 1's variance-reduction condition is a property of the induced weights and the conditional payoff statistics, not of per-test fidelity. Whether a given bank satisfies it for a given target is an empirical question that Theorem 3 lets us check at runtime. As a result, though counter-intuitive, the expert bank in Algorithm 2 does* not *need to simulate the same robot or policy being tested. This has been theoretically justified in this section and will be further demonstrated empirically in Section III (and with Remark 5).*

Finally, addressing the four questions posed as titles of previous subsections, we show that (II-B) betting helps when it provides predictive advantage that reduces estimator variance; (II-C) its optimal form (approximately) coincides with the classical small-stake Kelly criterion; (II-D) practical approximations can be realized via a simulator bank updated with proper scoring rules inspired by Cover's universal portfolio principle; and (II-E) sustained wealth growth approximately implies effective estimates.

III. Case Studies

The empirical studies illustrate the theoretical insights of Section II, highlighting both effective and failure regimes. Accordingly, our comparisons focus primarily on the Monte Carlo and various proposed betting-based estimators, including the ideal Kelly strategy described in Section II-C and its practical approximations implemented in Algorithm 2. Additional comparisons with IS and PPI variants are provided in the supplementary material. As both require case-specific adaptations and are not applicable to all scenarios, our most careful fair implementations show consistent advantages for the proposed method. Programs that allow direct replications of the presented results can be found in the repository [1].

A. Synthetic examples

This section characterizes the real-world distribution P and simulators Q_θ using a diverse family of parameterized synthetic distributions (e.g., Beta distributions, truncated normal distributions, Gaussian mixtures, Bernoulli, bimodal, and uniform-spike distributions (details can be found in the affiliated code and supplementary material) adapted from [6]) over the $[0, 1]$ interval. The performance measure adopted in these cases is $\psi(x) = x$.

a) Configurations: For the "real-world" distributions, we consider three collections of synthetic distribution variants with cardinalities 6, 20, and 40, denoted as `Real_6`, `Real_20`, and `Real_40`, respectively. Each collection is a strict subset of the next, constructed incrementally by progressively enriching the support. For the banks of simulator experts used in the approximated Kelly betting, we construct four collections of synthetic distribution variants with sizes 17, 35, 94, and 172, denoted as `Sim_17_biased`, `Sim_35`, `Sim_94`, and `Sim_172`. `Sim_17_biased` is intentionally skewed and biased toward the left portion of the distributional support, whereas the other three banks provide increasingly dense coverage over the entire range with varying resolutions. Importantly, none of the distribution variants in the `Sim` banks overlap with those used in the `Real` collections. For the implementation of ideal Kelly variants, we adopt the ground-truth mean and ground truth variance (which upper bounds the conditional variance σ_t^2).

[1] https://github.com/ISUSAIL/Bet4Sim2Real

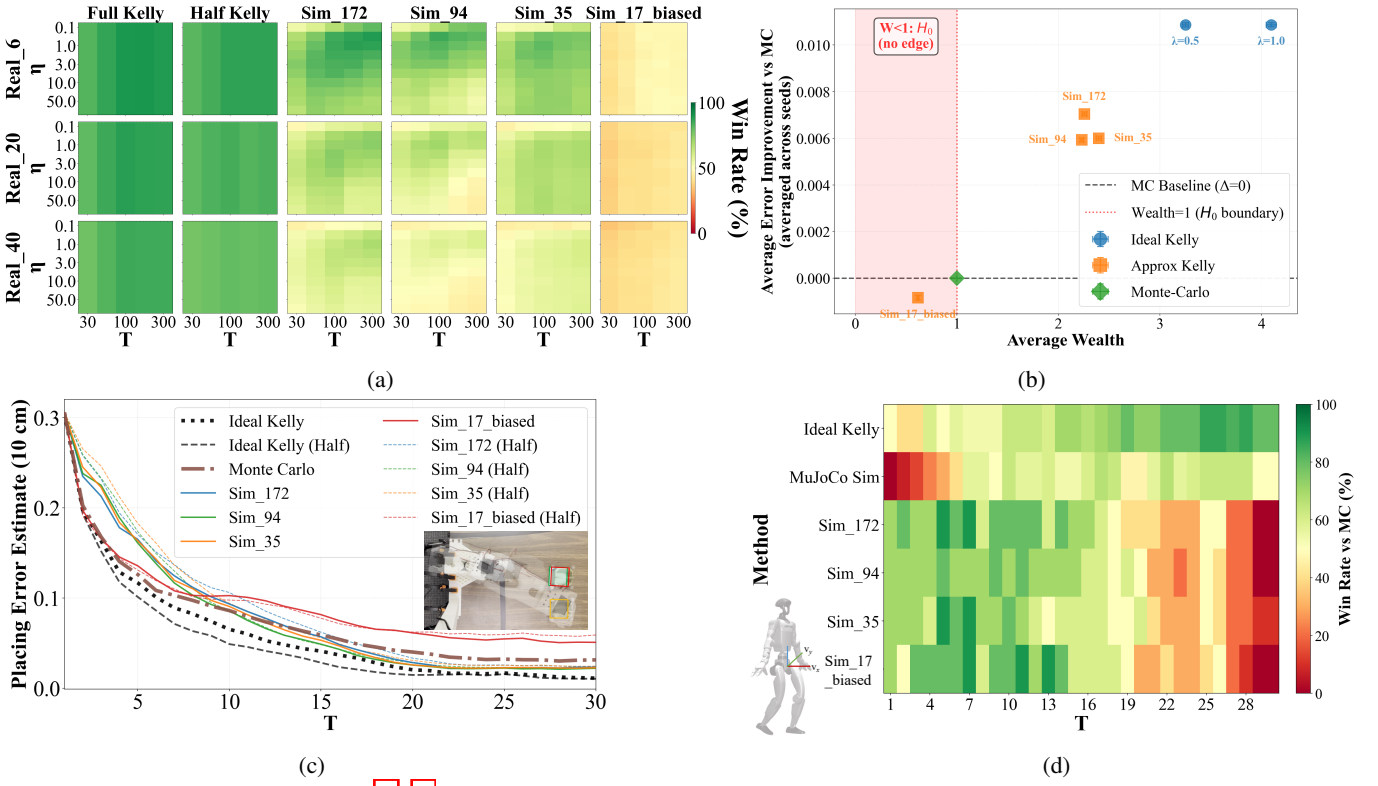

Fig. 2: Experiment results for Section III: 2a: win-rate comparison of all methods against the Monte Carlo baseline across different `Real` synthetic distributions, learning rates, and numbers of rounds; 2b: average wealth across methods on the `Real_6` distributions, providing empirical support for Theorem 3; 2c: Round-by-round estimates of the placement error for Section III-B across different methods, the *(Half)* label denotes half Kelly betting ($\lambda = 0.5$); 2d: Round-by-round win rate comparison for Section III-C across different methods. The maximum number of rounds is set to $T = 30$, reflecting commonly adopted trial counts in standardized real-world robotic testing, particularly for manipulators [13, 17, 66].

For each evaluated method (Monte Carlo, ideal Kelly with different Kelly fraction values, and approximated Kelly under different configurations), and for each task (corresponding to a specific distribution within one of the `Real` collections), the experiment is repeated over 100 independent random seeds. For each seed, the estimation error is evaluated against the ground-truth mean, as dictated by the known parameters of the underlying distribution. A bet-weighted estimate is considered a "win" if its estimation error is smaller than that of the Monte Carlo estimator on the same task, using the same random seed and the same number of rounds T (i.e., the same number of real-world samples). The overall results of the win rate evaluation are shown in Fig. 2a related to a variety of learning rates $\eta \in \{0.1, 0.5, 1.0, 2.0, 3.0, 4.0, 5.0, 10.0, 20.0, 50.0, 100.0\}$ for (12) and different rounds $T \in \{30, 50, 100, 200, 300\}$. Each cell in the heatmap of Fig. 2a reports the win rate of a given method on a specific task, computed over 100 random seed trials. For the approximated Kelly variants, the win rate is evaluated for the corresponding choice of η and number of rounds T. Finally, for the evaluation against `Real_6`, the average wealth achieved by each Kelly betting variant, aggregated over all 100 random seed trials and across different choices of η and T, is reported in Fig. 2b. The red shaded region indicates the no-edge null hypothesis dictated by (14) in Theorem 3.

b) Analysis and Discussions: Overall, the empirical results support the theoretical claims in Section II from multiple perspectives. As shown in Fig. 2a, ideal Kelly betting consistently dominates Monte Carlo by a clear margin across all settings, with performance improving as the number of rounds T increases due to more stable estimation from additional real-world samples. This trend aligns with the asymptotic nature of the theoretical analysis. For the approximated Kelly method (Algorithm 2), the largest simulator bank, `Sim_172`, generally achieves the best performance; however, more experts do not necessarily imply better results, as `Sim_35` often outperforms `Sim_94`. The learning rate η also influences performance, though these effects are descriptive for the studied cases, and a systematic characterization of their roles remains an important direction for future work. The wealth analysis in Fig. 2b empirically supports Theorem 3.

B. From synthetic simulators to real-world pick-and-place

This section evaluates the pick-and-place positioning accuracy of an imitation-learning-based policy on a SO-ARM101 manipulator [67] (the policy was trained from tele-operated trajectories collected on the same physical robot, and therefore no naturally "matching" simulator exists for this task [68]).

a) Configuration: In each trial, a green bucket is placed at a fixed location and moved to a target position offset by 102 mm, with a maximum trial duration of 60 s. Positioning accuracy is measured as the Euclidean distance between the target center and the bucket center using the OptiTrack motion capture system with five Prime^{x} 22 high-speed cameras

(≤ 0.15 mm accuracy). To enforce an i.i.d. setting, each trial includes a full power cycle and a cooldown time ≥ 60 s. The ground-truth error is estimated via Monte Carlo using 119 trials, achieving a relative half-width of 0.13 at a 95% confidence level. To align the performance measure ψ between the tracking task and the synthetic simulators, the placement error is expressed in units of 10 cm and normalized (to $[0, 1]$ interval) by a maximum distance of 30 cm. For the Monte Carlo baseline, we evaluate 90 trials, each constructed from a contiguous window of 30 samples spanning the i-th to the $(i + 30)$-th samples of the aforementioned ground-truth process. For each trial, we apply ideal Kelly betting with full and half Kelly fractions, as well as the approximated Kelly method using the same synthetic `Sim` variants as in the previous example. The estimation error, averaged over all trials as a function of T, is shown in Fig. 2c.

b) Analysis and Discussion: Overall, ideal Kelly remains dominant, with half Kelly being more stable in the early stages, although both variants empirically converge to similar estimation error levels. Most `Sim` variants outperform Monte Carlo, particularly in later rounds as simulator weights adapt over time. In contrast, `Sim_17_biased` remains ineffective due to its pronounced mismatch with the task distribution.

Remark 5. *Although synthetic distributions are effective in both examples, the pick-and-place study in particular highlights a key property of the bet-weighted estimator: its effectiveness depends on* distributional *signal rather than task-specific or application-level alignment. This observation should not be interpreted as advocating the replacement of application-specific simulators with synthetic ones. In practice, synthetic banks often require broad distributional coverage and may be less efficient than well-correlated sim-to-real simulators, a point further illustrated in Section III-C.*

C. Sim-to-sim locomotion tracking performance evaluation

In this section, we demonstrate the proposed method by evaluating the velocity-tracking performance of a Unitree G1 humanoid robot controlled by a reinforcement-learning-based locomotion policy [69, 70] in the MuJoCo simulator [71].

a) Configuration: : Each test trial lasts 4 s, starting from a fixed initial pose and executing two sequential velocity commands of 2 s each, specified by desired planar velocities $(v_x, v_y) \in [-1, 1]^2$. The velocity-tracking error is defined as the mean ℓ_2-norm of the difference between measured and commanded velocities over the 4 s horizon, normalized by a maximum speed of 1.5 m/s. In this example, the "real-world" is a fixed MuJoCo environment with specified parameters (robot mass, damping coefficients, friction, and a fixed random seed for velocity commands). The simulator side consists of a domain-randomized bank of 10 experts with varying parameters and random seeds, none matching the real instance. The experts' estimate update follows Algorithm 2. We also reuse the synthetic `Sim` variants from Section III-A. The ground-truth tracking error is estimated via sufficient sampling using 552 trials, achieving a relative half-width below 0.02 at a 95% confidence level. We then evaluate bootstrapped trials using different betting-based methods. Each method is repeated over 30 independent random seeds. As in Section III-A, a method is counted as a "win" if its error estimate outperforms Monte Carlo on the same samples. Results are shown in Fig. 2d, with learning rate $\eta = 5.0$ throughout.

b) Analysis and Discussion: Ideal Kelly betting remains the dominant method overall. Among the two classes of approximated Kelly variants, synthetic `Sim` distributions often perform better in the early rounds, largely due to noisy or "lucky" early guesses. In contrast, the bank of domain-randomized MuJoCo-based simulators may start with less favorable initial estimates, but converge rapidly and dominate in later rounds. Comparison on this specific case against a PPI variant [38] can be found in the supplementary material.

IV. Conclusion: Limitations and Future Work

Beyond the theoretical limitations discussed in Remark 2, the practical betting scheme in Section II-D currently lacks theoretical guarantees characterizing how well and to what extent it approximates the ideal Kelly strategy. While such guarantees appear attainable in principle (e.g., by leveraging classical regret bounds, mixture optimality results, or asymptotic efficiency analyses from Cover's universal portfolio framework [58] (see Remark 3)), formal development is left to future work. Another limitation (also common in robot testing) is the i.i.d. sampling assumption, which may not naturally hold in robotic experiments and can require additional effort to enforce (e.g., Section III-B). Extending the framework to non-stationary settings with independent but non-identically distributed samples is an important and plausible [6] direction for future work. On the empirical side, our demonstrations do not yet include large-scale, computationally intensive simulators (e.g., those with rich visual sensing or high-dimensional perception). While our theory applies broadly to scalar mean performance measures (and thus remains compatible with dimension-reduced measurements), extending the empirical evaluation to these settings is an important direction for future work.

Finally, to the best of the authors' knowledge, this work is the first to explicitly connect *betting* with sim-to-real robotics. While the present study focuses on expectation-based performance evaluation for a fixed system, the underlying mechanisms and insights extend beyond mean estimation and may facilitate policy training, sim-to-real transfer, and online adaptation. In particular, the betting perspective provides a principled way to exploit predictive advantage and structured simulator variability beyond pure evaluation.

Acknowledgment

The author Bowen Weng would like to express his sincere gratitude to the first responders and firefighters of the City of Iowa Falls at Iowa, U.S.A. Their courageous actions rescued his family from a life-threatening situation and, in a very real sense, resolved a dangerous bet he made, one that also, unintentionally, made this paper possible.

References

[1] Seth Hutchinson. Model-Based Methods in Today's Data-Driven Robotics Landscape. Invited talk, 2026. The remark on roboticists "betting" before real-world experiments appears at 5:45.

[2] Shuo Feng, Xintao Yan, Haowei Sun, Yiheng Feng, and Henry X Liu. Intelligent driving intelligence test for autonomous vehicles with naturalistic and adversarial environment. *Nature Communications*, 12(1):1–14, 2021.

[3] Abhishek Kadian, Joanne Truong, Aaron Gokaslan, Alexander Clegg, Erik Wijmans, Stefan Lee, Manolis Savva, Sonia Chernova, and Dhruv Batra. Sim2real predictivity: Does evaluation in simulation predict real-world performance? *IEEE Robotics and Automation Letters*, 5(4):6670–6677, 2020.

[4] Fabio Muratore, Michael Gienger, and Jan Peters. Assessing transferability from simulation to reality for reinforcement learning. *IEEE transactions on pattern analysis and machine intelligence*, 43(4):1172–1183, 2019.

[5] Bowen Weng, Guillermo A Castillo, Yun-Seok Kang, and Ayonga Hereid. Towards standardized disturbance rejection testing of legged robot locomotion with linear impactor: A preliminary study, observations, and implications. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 9946–9952. IEEE, 2024.

[6] Bowen Weng, Linda Capito, Guillermo A. Castillo, and Dylan Khor. Rethink Repeatable Measures of Robot Performance with Statistical Query. *IEEE Transactions on Robotics*, 42:561–578, 2025. doi: 10.1109/TRO.2025.3645934.

[7] Rishabh Agarwal, Max Schwarzer, Pablo Samuel Castro, Aaron C Courville, and Marc Bellemare. Deep reinforcement learning at the edge of the statistical precipice. *Advances in neural information processing systems*, 34: 29304–29320, 2021.

[8] Yan Duan, Xi Chen, Rein Houthooft, John Schulman, and Pieter Abbeel. Benchmarking deep reinforcement learning for continuous control. In *International conference on machine learning*, pages 1329–1338. PMLR, 2016.

[9] Hadas Kress-Gazit, Kunimatsu Hashimoto, Naveen Kuppuswamy, Paarth Shah, Phoebe Horgan, Gordon Richardson, Siyuan Feng, and Benjamin Burchfiel. Robot learning as an empirical science: Best practices for policy evaluation. *arXiv preprint arXiv:2409.09491*, 2024.

[10] Bowen Weng, Guillermo A Castillo, Wei Zhang, and Ayonga Hereid. On the comparability and optimal aggressiveness of the adversarial scenario-based safety testing of robots. *IEEE Transactions on Robotics*, 39(4): 3299–3318, 2023.

[11] Nidhi Kalra and Susan M Paddock. Driving to safety: How many miles of driving would it take to demonstrate autonomous vehicle reliability? *Transportation research part A: policy and practice*, 94:182–193, 2016.

[12] Shuo Feng, Haowei Sun, Xintao Yan, Haojie Zhu, Zhengxia Zou, Shengyin Shen, and Henry X Liu. Dense reinforcement learning for safety validation of autonomous vehicles. *Nature*, 615(7953):620–627, 2023.

[13] Bernard Roth. Performance evaluation of manipulators from a kinematic viewpoint. *NBS Special Publication*, 459:39–62, 1976.

[14] Joseph A Vincent, Haruki Nishimura, Masha Itkina, Paarth Shah, Mac Schwager, and Thomas Kollar. How generalizable is my behavior cloning policy? a statistical approach to trustworthy performance evaluation. *IEEE Robotics and Automation Letters*, 2024.

[15] Cédric Colas, Olivier Sigaud, and Pierre-Yves Oudeyer. A hitchhiker's guide to statistical comparisons of reinforcement learning algorithms. *arXiv preprint arXiv:1904.06979*, 2019.

[16] American National Standards Institute/Robotic Industries Association. ANSI/RIA R15.05: Industrial Robots and Robot Systems – Performance Characteristics, 1992.

[17] International Organization for Standardization. ISO 9283: Manipulating Industrial Robots – Performance Criteria and Related Test Methods, 1998.

[18] Michiel R. van Ratingen. The Euro NCAP safety rating. In Alexander Piskun, editor, *Karosseriebautage Hamburg 2017*, pages 11–20, Wiesbaden, 2017. Springer Fachmedien Wiesbaden. ISBN 978-3-658-18107-9.

[19] Richard S Sutton, Andrew G Barto, et al. *Reinforcement learning: An introduction*, volume 1. MIT press Cambridge, 1998.

[20] Xue Bin Peng, Marcin Andrychowicz, Wojciech Zaremba, and Pieter Abbeel. Sim-to-real transfer of robotic control with dynamics randomization. In *2018 IEEE international conference on robotics and automation (ICRA)*, pages 3803–3810. IEEE, 2018.

[21] SangHo Hyon and Gordon Cheng. Passivity-based full-body force control for humanoids and application to dynamic balancing and locomotion. In *2006 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 4915–4922. IEEE, 2006.

[22] Stephen Boyd and Lieven Vandenberghe. *Convex optimization*. Cambridge university press, 2004.

[23] Nicholas Metropolis and Stanislaw Ulam. The monte carlo method. *Journal of the American statistical association*, 44(247):335–341, 1949.

[24] Nicholas Metropolis, Arianna W Rosenbluth, Marshall N Rosenbluth, Augusta H Teller, and Edward Teller. Equation of state calculations by fast computing machines. *The journal of chemical physics*, 21(6):1087–1092, 1953.

[25] W Keith Hastings. Monte Carlo sampling methods using Markov chains and their applications. *Biometrika*, 57(1): 97–109, 1970.

[26] JM Hammersley and DC Handscomb. Monte carlo methods. *Ltd., London*, 40:32, 1964.

[27] Henry X Liu and Shuo Feng. Curse of rarity for autonomous vehicles. *nature communications*, 15(1): 4808, 2024.

[28] Afsoon Afzal, Claire Le Goues, Michael Hilton, and

Christopher Steven Timperley. A study on challenges of testing robotic systems. In *2020 IEEE 13th international conference on software testing, validation and verification (ICST)*, pages 96–107. IEEE, 2020.

[29] Philip Koopman and Michael Wagner. Challenges in autonomous vehicle testing and validation. *SAE International Journal of Transportation Safety*, 4(1):15–24, 2016.

[30] Søren Asmussen and Peter W Glynn. Rare-event simulation. In *Stochastic Simulation: Algorithms and Analysis*, pages 158–205. Springer, 2007.

[31] Herman Kahn and Theodore E Harris. Estimation of particle transmission by random sampling. *National Bureau of Standards applied mathematics series*, 12:27–30, 1951.

[32] Søren Asmussen and Peter W Glynn. *Stochastic simulation: algorithms and analysis*, volume 57. Springer, 2007.

[33] Matthew O'Kelly, Aman Sinha, Hongseok Namkoong, Russ Tedrake, and John C Duchi. Scalable end-to-end autonomous vehicle testing via rare-event simulation. *Advances in neural information processing systems*, 31, 2018.

[34] Sourav Chatterjee and Persi Diaconis. The sample size required in importance sampling. *The Annals of Applied Probability*, 28(2):1099–1135, 2018.

[35] Mark Koren, Saud Alsaif, Ritchie Lee, and Mykel J Kochenderfer. Adaptive stress testing for autonomous vehicles. In *2018 IEEE Intelligent Vehicles Symposium (IV)*, pages 1–7. IEEE, 2018.

[36] Yevgen Chebotar, Ankur Handa, Viktor Makoviychuk, Miles Macklin, Jan Issac, Nathan Ratliff, and Dieter Fox. Closing the sim-to-real loop: Adapting simulation randomization with real world experience. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 8973–8979. IEEE, 2019.

[37] Anastasios N Angelopoulos, Stephen Bates, Clara Fannjiang, Michael I Jordan, and Tijana Zrnic. Prediction-powered inference. *Science*, 382(6671):669–674, 2023.

[38] Apurva Badithela, David Snyder, Lihan Zha, Joseph Mikhail, Matthew O'Kelly, Anushri Dixit, and Anirudha Majumdar. Reliable and scalable robot policy evaluation with imperfect simulators. *arXiv preprint arXiv:2510.04354*, 2025.

[39] Rajesh Ranganath, Sean Gerrish, and David Blei. Black box variational inference. In *Artificial intelligence and statistics*, pages 814–822. PMLR, 2014.

[40] Rachel Luo, Heng Yang, Michael Watson, Apoorva Sharma, Sushant Veer, Edward Schmerling, and Marco Pavone. Sim2Val: Leveraging Correlation Across Test Platforms for Variance-Reduced Metric Estimation. *arXiv preprint arXiv:2506.20553*, 2025.

[41] Josh Tobin, Rachel Fong, Alex Ray, Jonas Schneider, Wojciech Zaremba, and Pieter Abbeel. Domain randomization for transferring deep neural networks from simulation to the real world. In *2017 IEEE/RSJ international conference on intelligent robots and systems (IROS)*, pages 23–30. IEEE, 2017.

[42] Jie Tan, Tingnan Zhang, Erwin Coumans, Atil Iscen, Yunfei Bai, Danijar Hafner, Steven Bohez, and Vincent Vanhoucke. Sim-to-Real: Learning Agile Locomotion for Quadruped Robots. In *Robotics: Science and Systems*, 2018.

[43] Fabio Muratore, Christian Eilers, Michael Gienger, and Jan Peters. Data-efficient domain randomization with bayesian optimization. *IEEE Robotics and Automation Letters*, 6(2):911–918, 2021.

[44] Ilge Akkaya, Marcin Andrychowicz, Maciek Chociej, Mateusz Litwin, Bob McGrew, Arthur Petron, Alex Paino, Matthias Plappert, Glenn Powell, Raphael Ribas, et al. Solving rubik's cube with a robot hand. *arXiv preprint arXiv:1910.07113*, 2019.

[45] Zhehui Huang, Zhaojing Yang, Rahul Krupani, Baskın Şenbaşlar, Sumeet Batra, and Gaurav S Sukhatme. Collision avoidance and navigation for a quadrotor swarm using end-to-end deep reinforcement learning. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 300–306. IEEE, 2024.

[46] A survey on transfer learning, author=Pan, Sinno Jialin and Yang, Qiang. *IEEE Transactions on knowledge and data engineering*, 22(10):1345–1359, 2009.

[47] Lerrel Pinto, James Davidson, Rahul Sukthankar, and Abhinav Gupta. Robust adversarial reinforcement learning. In *International conference on machine learning*, pages 2817–2826. PMLR, 2017.

[48] Akshara Rai, Rika Antonova, Franziska Meier, and Christopher G Atkeson. Using simulation to improve sample-efficiency of Bayesian optimization for bipedal robots. *Journal of machine learning research*, 20(49): 1–24, 2019.

[49] Steven R Howard, Aaditya Ramdas, Jon McAuliffe, and Jasjeet Sekhon. Time-uniform Chernoff bounds via nonnegative supermartingales. *Probability Surveys*, 17: 257–317, 2020.

[50] John L Kelly. A new interpretation of information rate. *the bell system technical journal*, 35(4):917–926, 1956.

[51] Edward O Thorp. Portfolio choice and the Kelly criterion. In *Stochastic optimization models in finance*, pages 599–619. Elsevier, 1975.

[52] Edward O Thorp. Understanding the Kelly criterion. In *The Kelly capital growth investment criterion: theory and practice*, pages 509–523. World Scientific, 2011.

[53] Louis M Rotando and Edward O Thorp. The Kelly criterion and the stock market. *The American Mathematical Monthly*, 99(10):922–931, 1992.

[54] Thomas M Cover. *Elements of information theory*. John Wiley & Sons, 1999.

[55] Leonard C MacLean, William T Ziemba, and George Blazenko. Growth versus security in dynamic investment analysis. *Management Science*, 38(11):1562–1585, 1992.

[56] Bill Ziemba. Good and bad properties of the Kelly criterion. *The Best of Wilmott*, page 65, 2006.

[57] Thomas M Cover. Universal portfolios. *Mathematical finance*, 1(1):1–29, 1991.

[58] Thomas M Cover and Erik Ordentlich. Universal portfolios with side information. *IEEE Transactions on Information Theory*, 42(2):348–363, 2002.

[59] Paul H Algoet and Thomas M Cover. Asymptotic optimality and asymptotic equipartition properties of log-optimum investment. *The Annals of Probability*, pages 876–898, 1988.

[60] Nick Littlestone and Manfred K Warmuth. The weighted majority algorithm. *Information and computation*, 108 (2):212–261, 1994.

[61] Nicolo Cesa-Bianchi and Gábor Lugosi. *Prediction, learning, and games*. Cambridge university press, 2006.

[62] Yoav Freund and Robert E Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of computer and system sciences*, 55 (1):119–139, 1997.

[63] Tilmann Gneiting and Adrian E Raftery. Strictly proper scoring rules, prediction, and estimation. *Journal of the American Statistical Association*, 102(477):359–378, 2007.

[64] Vladimir G Vovk. A game of prediction with expert advice. In *Proceedings of the eighth annual conference on Computational learning theory*, pages 51–60, 1995.

[65] Aaditya Ramdas, Peter Grünwald, Vladimir Vovk, and Glenn Shafer. Game-theoretic statistics and safe anytime-valid inference. *Statistical Science*, 38(4):576–601, 2023.

[66] International Organization for Standardization. ISO 18646: Robots and Robotic Devices – Performance Criteria and Related Test Methods for Service Robots, 2016.

[67] The Robot Studio. SO-ARM100: Open-Source Robotic Arm Platform. https://github.com/TheRobotStudio/SO-ARM100, 2024. Accessed: 2025-01-XX.

[68] Tony Z. Zhao, Vikash Kumar, Sergey Levine, and Chelsea Finn. Learning Fine-Grained Bimanual Manipulation with Low-Cost Hardware. In *Proceedings of Robotics: Science and Systems*, Daegu, Republic of Korea, July 2023. doi: 10.15607/RSS.2023.XIX.016.

[69] Unitree Robotics. Unitree RL Gym. https://github.com/unitreerobotics/unitree_rl_gym, 2024.

[70] Clemens Schwarke, Mayank Mittal, Nikita Rudin, David Hoeller, and Marco Hutter. RSL-RL: A learning library for robotics research. *arXiv preprint arXiv:2509.10771*, 2025.

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