World Translation: Minimizing Sim-to-Real Gap with Backward Dynamics Extraction and Unpaired Domain Translation

Author Names Omitted for Anonymous Review. Paper-ID 350

***Abstract*—The simulation-to-reality gap remains a fundamental challenge in robotics, particularly under partial observability, where time-varying hidden variables, such as contact, collision and external disturbances, significantly affect dynamics but cannot be predicted from observation history. Learned dynamics models offer a promising path to minimize this gap by capturing domain-specific characteristics directly from data. We observe that simulators and learned models have complementary strengths: simulators explicitly model transient physics but suffer from inaccurate persistent parameters, while learned models capture domain-specific characteristics but cannot infer discontinuously evolving hidden variables. We propose World Translation, which combines these strengths by reformulating sim-to-real gap minimization as unpaired domain translation over learned dynamics features. Our approach consists of two stages: (1) Backward Dynamics Extraction encodes the effects of hidden variables from observed state transitions into a compact representation, using transition outcomes rather than forward prediction from history; (2) Unpaired Domain Translation maps these features across domains, preserving hidden variable semantics while transforming domain characteristics. This enables the simulator to serve as a source of hidden variable information while correcting for domain mismatch. Experiments on locomotion and manipulation tasks show that World Translation achieves more accurate dynamics modeling than baselines, with the largest gains when hidden variables evolve discontinuously.**

I. Introduction

Simulation has become the foundational infrastructure of the sim-to-real robot learning paradigm [1, 2, 3]. However, since these fashions prevail, it is always hindered by the sim-to-real gaps. For example, inaccurate modeling of joint motor characteristics can lead to ineffective quadruped locomotion policy [1]. Existing approaches address this mismatch through several dominant paradigms: domain randomization, system identification and dynamics model learning. Domain randomization [4, 5, 6] trains policies to be robust across varied dynamics parameters, but this robustness comes at the cost of optimality, producing conservative policies [7, 8]. System identification [9, 10] estimates physical parameters to align simulation with reality, but remains limited to predefined parameter spaces that cannot capture unmodeled dynamics. In contrast, the learned dynamics models [11, 12, 13, 14], including the residual ones [15, 16, 17], provide a distinct perspective. These approaches learn latent dynamics directly from data, implicitly capturing domain-specific characteristics, which offer a more flexible and promising alternative for minimizing sim-to-real gap by providing higher-fidelity dynamics modeling that can bypass the modeling limits of traditional simulators, like the predefined parameter spaces and idealized analytical formulations.

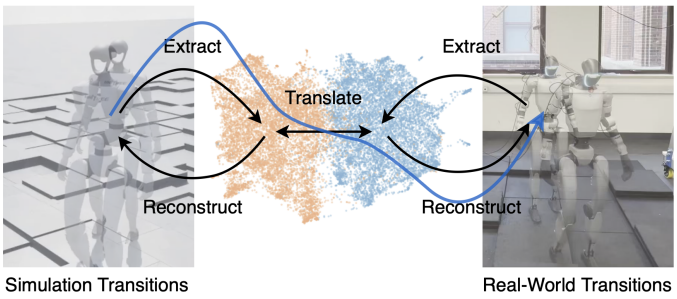


Fig. 1. World Translation bridges the sim-to-real gap by extracting dynamics features from observed transitions and translating them across domains. Black arrows are used during training; the blue arrow illustrates the inference pipeline, where simulation dynamics features are translated and reconstructed to predict real-world transitions.

Despite their promise, learned dynamics models face a fundamental challenge from unobservable time-varying factors such as contact, collision, and external disturbances, which significantly affect dynamics but cannot be reliably inferred from observation history. And we refer to such factors as *hidden variables*. Unlike time-invariant domain characteristics (e.g., motor response, mass distribution, friction parameters) that remain constant throughout execution, hidden variables may change discontinuously between timesteps. A collision has no observable precursor; it simply occurs. Standard forward-prediction architectures assume hidden variables can be inferred from past observations, which breaks when these factors evolve unpredictably. Crucially, while hidden variables cannot be predicted in advance, their effects are observable in transition outcomes: a stumble reveals that a collision occurred. Forward prediction marginalizes over these latent causes rather than inferring them from the consequent effect. This reveals a critical complementarity: simulators explicitly model these transient physics but suffer from inaccurate persistent parameters like friction, while learned models capture domain-specific characteristics but cannot infer transient signals. We argue that these two types of unobservable factors, time-variant (hidden variables) and time-invariant (domain characteristics), require fundamentally different treatment.

We propose World Translation, a novel dynamics model framework that combines these complementary strengths by reformulating the sim-to-real gap minimization as an unpaired domain translation problem over learned dynamics features. Our approach consists of two components. First, *backward dy-*

namics extraction encodes the effects of unobservable factors from observed state transitions into a compact representation. Unlike forward prediction, which infers next states from past observations, backward extraction uses the observed outcome to capture what actually happened, accepting discontinuous evolutions without temporal smoothness assumptions. Second, *unpaired domain translation* maps these dynamics features across domains by cycle-consistent adversarial training [18]. Like image translation [18, 19, 20] that preserves semantics while transforming style, dynamics translation preserves time-variant hidden variable information while transforming domain characteristics. In this formulation, we observe a transition in the simulation, extract its dynamics features, translate them to the target domain, and decode to predict what the corresponding real-world transition would be, where the simulation serves as a source of information about time-variant factors.

Our contributions are summarized as follows.

- We propose backward dynamics extraction, which encodes the effects of time-variant unobservable factors (hidden variables) from observed transitions rather than predicting forward from past observations, eliminating temporal smoothness assumptions.
- We reformulate sim-to-real dynamics modeling as unpaired domain translation over learned dynamics features, which preserves hidden variable information while transforming domain characteristics and further enables the simulator to serve as a source of hidden variable information while correcting for domain mismatch.
- We demonstrate that the proposed World Translation achieves more accurate dynamics modeling than forward prediction methods on humanoid, quadruped, and manipulator platforms, with the largest gains when hidden variables evolve discontinuously, resulting in improved downstream policy performance.

II. Related Work

A. Sim-to-Real Transfer

Sim-to-real transfer is commonly facilitated through domain randomization [4, 5, 6, 21] and system identification [9, 10]. Domain randomization achieves robustness at the cost of optimality [7], while system identification is limited to predefined parameter spaces. Recent work combines these approaches [22, 23], but all operate within fixed parameter spaces that cannot capture unmodeled dynamics.

B. Learned Dynamics Models

Learned dynamics models offer an alternative path to sim-to-real transfer by creating more realistic simulations from data, bypassing the modeling limits of predefined parameter spaces. Early Gaussian process methods like PILCO [24] demonstrated sample efficiency, while neural network approaches like PETS [25] addressed scalability. The Dreamer family [11, 12, 13] learns latent dynamics from high-dimensional observations, with DayDreamer [26] demonstrating learning directly on physical robots. TD-MPC2 [14] combines world models with model predictive control for diverse continuous control tasks. However, these approaches use forward prediction $p(o_{t+1}|o_{1:t}, a_{1:t})$ to handle partial observability, assuming hidden variables can be inferred from history. This breaks when hidden variables cannot be determined from past observations, such as terrain contact or external disturbances.

C. Residual Dynamics and Residual Action

Residual learning learns from real-world data to improve dynamics modeling, and augments simulators with learned corrections: residual dynamics methods [15, 16, 27, 17] correct simulator outputs, while residual action methods [28, 29, 30] transform inputs. While these approaches utilize simulators, they do not exploit the information simulators provide about underlying physical processes. Their potential is thus bounded by what pure learned dynamics models can achieve.

D. Unpaired Domain Translation

CycleGAN [18] introduced cycle-consistent adversarial training for unpaired image-to-image translation, preserving semantics while transforming domain-specific style. This framework has been applied to visual sim-to-real transfer [31, 32], focusing on visual adaptation and does not address the dynamics domain gap. Our work applies unpaired translation to learned dynamics features rather than images, where image translation preserves semantic content while transforming visual style, dynamics translation preserves hidden variable semantics while transforming domain characteristics, enabling simulation to serve as a source of hidden variable information.

III. Problem Formulation

A. System Dynamics and Hidden Variables

We formalize the distinction between time-varying factors and time-invariant domain characteristics. Consider a robotic system defined by its full state $\mathbf{s}_t \in \mathcal{S}$ and actions $\mathbf{a}_t \in \mathcal{A}$. Under full observability, the dynamics are Markovian, satisfying $\mathbf{s}_{t+1} = g(\mathbf{s}_t, \mathbf{a}_t)$, where $\mathcal{S}$ is state space and $\mathcal{A}$ represents action space. In practice, we typically only have access to partial observations $\mathbf{o}_t \in \mathcal{O}$ and $\mathcal{O}$ represents the observation space. Modeling dynamics solely via $\mathbf{o}_{t+1} = g(\mathbf{o}_t, \mathbf{a}_t)$ is typically non-Markovian because identical observation-action pairs can produce different outcomes due to hidden states. To model the time-variant factors affecting dynamics and restore determinism over observation space, we introduce hidden variables $\mathbf{h}_t \in \mathcal{H}$. We define that $\mathbf{h}_t$ captures the unobserved component of the full state that may vary unpredictably during execution, which means full state $\mathbf{s}_t = [\mathbf{o}_t, \mathbf{h}_t]$. By conditioning on $\mathbf{h}_t$, the dynamics become:

$$\mathbf{o}_{t+1} = f(\mathbf{o}_t, \mathbf{a}_t, \mathbf{h}_t). \tag{1}$$

The hidden variable $\mathbf{h}_t$ serves as the latent factor that disambiguates transition outcomes. While $p(\mathbf{o}_{t+1}|\mathbf{o}_t, \mathbf{a}_t)$ is highly stochastic, the mapping becomes deterministic when conditioned on $\mathbf{h}_t$. Critically, $\mathbf{h}_t$ may exhibit discrete or non-smooth transitions between successive timesteps. For instance, a collision at time t might occur without any predictive

cues at $t-1$, making it impossible to anticipate from the previous observation. Unlike standard POMDP that relies on a well-defined state transition model $p(\mathbf{s}_{t+1}|\mathbf{s}_t, \mathbf{a}_t)$, our setting remains agnostic to the evolution of $\mathbf{h}_t$. We dispense with assumptions regarding $p(\mathbf{h}_{t+1}|\mathbf{h}_t)$, requiring only that the latent state's impact be identifiable through the observed transition $(\mathbf{o}_t, \mathbf{a}_t) \rightarrow \mathbf{o}_{t+1}$. We use hidden *variables* rather than *states* that denote quantities with well-defined dynamics in control theory or *events* that imply discrete occurrences.

B. Backward Dynamics Extraction

As a learned dynamics model, our goal is to predict $\mathbf{o}_{t+1}$. As formulated in (1), this requires knowledge of hidden variables $\mathbf{h}_t$. While $\mathbf{h}_t$ cannot be directly observed, their effect on dynamics is detectable through state transitions. We introduce a *dynamics feature* $\mathbf{Z}_t = \Phi(\mathbf{h}_t)$ to represent this information in latent space.

$$(\mathbf{o}_t, \mathbf{a}_t, \mathbf{o}_{t+1}) \xrightarrow{\text{encode}} \mathbf{Z}_t = \Phi(\mathbf{h}_t), \tag{2}$$

$$(\mathbf{o}_t, \mathbf{a}_t, \mathbf{Z}_t) \xrightarrow{\text{decode}} \mathbf{o}_{t+1}. \tag{3}$$

We call this backward extraction because it uses future information $\mathbf{o}_{t+1}$ to infer $\mathbf{Z}_t$ at time t.

However, this extraction alone does not enable dynamics prediction. $\mathbf{Z}_t$ requires observing the outcome $\mathbf{o}_{t+1}$, which is precisely what we wish to predict. To make $\mathbf{Z}_t$ useful, we need an alternative source, such as a simulator where transitions *can* be observed, from which we extract dynamics features and translate them to the target system.

C. Unpaired Domain Translation

The backward extraction in Section III-B assumes a single domain where $\mathbf{Z}_t = \Phi(\mathbf{h}_t)$. However, when applying this framework across domains, for example simulation and reality, the same hidden variable $\mathbf{h}_t$ produces different observed transitions in each domain. Therefore, the dynamics feature extracted from transitions is not solely a function of $\mathbf{h}_t$, but also depends on which domain generated the transition. We introduce domain-specific characteristics $\mathbf{c}$ to explicitly capture these systematic differences. With h_t and c, the dynamics can be formulated as $\mathbf{o}_{t+1} = f(\mathbf{o}_t, \mathbf{a}_t, \mathbf{h}_t, \mathbf{c})$. $\mathbf{c}$ is an implicit learned representation capturing domain-specific effects, including unmodeled phenomena. Figure 2 illustrates how $\mathbf{h}_t$ and $\mathbf{c}$ relate to other unobservable factors.

In this case, the dynamics feature (now denoted $\mathbf{z}_t$ for the multi-domain setting) is jointly determined by both hidden variables and domain characteristics:

$$(\mathbf{o}_t, \mathbf{a}_t, \mathbf{o}_{t+1}) \xrightarrow{\text{encode}} \mathbf{z}_t = \phi(\mathbf{h}_t, \mathbf{c}), \tag{4}$$

$$(\mathbf{o}_t, \mathbf{a}_t, \mathbf{z}_t) \xrightarrow{\text{decode}} \mathbf{o}_{t+1}, \tag{5}$$

where ϕ extends Φ by incorporating domain characteristics into the encoding. Note that $\mathbf{h}_t$ and $\mathbf{c}$ are not explicitly separable within $\mathbf{z}_t$ and their effects are mixed in the learned representation. For the same hidden variable $\mathbf{h}_t$, different domains produce different dynamics features: $\mathbf{z}_t^A = \phi(\mathbf{h}_t, \mathbf{c}^A)$ and $\mathbf{z}_t^B = \phi(\mathbf{h}_t, \mathbf{c}^B)$.

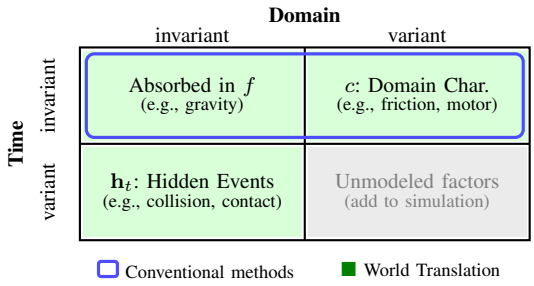


Fig. 2. Taxonomy of unobservable factors. Conventional dynamics models implicitly learn time-invariant domain characteristics ($\mathbf{c}$) but cannot explicitly capture time-variant hidden events ($\mathbf{h}_t$) when they evolve discontinuously. World Translation captures $\mathbf{h}_t$ through backward extraction, enabling generalization across unpredictable hidden variable evolutions. Factors that are both domain-variant and time-variant (e.g., a payload present in reality but absent in simulation) represent unmodeled phenomena that World Translation cannot recover from the source system.

In this case, to predict o_{t+1}^B in a target domain B, the problem becomes how to infer $\mathbf{z}_t^B$, with the help of $\mathbf{z}_t^A$ from a source domain A. So, we seek to learn a mapping $G_{A\rightarrow B}$ such that:

$$G_{A\rightarrow B}(\phi(\mathbf{h}_t, \mathbf{c}^A)) \approx \phi(\mathbf{h}_t, \mathbf{c}^B), \tag{6}$$

effectively transforming the domain-specific component while preserving the hidden variable component, despite their entanglement in the representation. This becomes an *unpaired domain translation* problem: learning a mapping that preserves hidden variable semantics $\mathbf{h}_t$ while transforming domain characteristics $\mathbf{c}$ without paired supervision, where "paired" means corresponding to the *same underlying hidden variable* $\mathbf{h}_t$, not merely matching $(\mathbf{o}_t, \mathbf{a}_t)$. While we can collect transitions with identical observations and actions across domains, we cannot ensure the underlying hidden variables are identical. Because such physical interaction cannot be precisely reproduced.

D. Combined Objective

By combining backward dynamics extraction in Section III-B and unpaired domain translation in Section III-C, we formulate World Translation framework. Given unpaired transition datasets $\mathcal{D}^A$ and $\mathcal{D}^B$ from two domains, World Translation seeks to learn: (1) A dynamics feature $\mathbf{z}_t$ that captures the effect of hidden variables $\mathbf{h}_t$ on state transitions; (2) A mapping $G_{A\rightarrow B}$ that translates dynamics features across domains while preserving hidden variable semantics such that observing a transition in domain A allows predicting what the corresponding transition would be in domain B:

$$\mathbf{o}_{t+1}^A \xrightarrow[\mathbf{o}_t, \mathbf{a}_t]{\text{encode}} \mathbf{z}_t^A \xrightarrow{G_{A\rightarrow B}} \mathbf{z}_t^B \xrightarrow[\mathbf{o}_t, \mathbf{a}_t]{\text{decode}} \mathbf{o}_{t+1}^B. \tag{7}$$

IV. Method

We instantiate the World Translation framework in Section III using a VAE with auxiliary regularization for backward dynamics extraction and a CycleGAN [18] for unpaired domain translation. The following subsections detail each component's architecture, training objectives, and deployment procedure.

A. Backward Dynamics Extraction

We implement the dynamics encoder-decoder using a variational autoencoder. We first describe the core architecture, then analyze two challenges that arise during training and introduce auxiliary objectives to address them.

The encoder E_ϕ maps a transition tuple to a latent distribution $(\boldsymbol{\mu}, \log \boldsymbol{\sigma}^2) = E_\phi(\mathbf{o}_t, \mathbf{a}_t, \mathbf{o}_{t+1})$, where the latent code is sampled as $\mathbf{z}_t \sim \mathcal{N}(\boldsymbol{\mu}, \mathrm{diag}(\boldsymbol{\sigma}^2))$. The decoder D_θ reconstructs the next observation given the current state, action, and latent code $\hat{\mathbf{o}}_{t+1} = D_\theta(\mathbf{o}_t, \mathbf{a}_t, \mathbf{z}_t)$. Ideally, the encoder learns to encode hidden variables $\mathbf{h}_t$ and domain characteristics $\mathbf{c}$ into $\mathbf{z}_t$, as both are required by the decoder to reconstruct $\mathbf{o}_{t+1}$. However, two challenges prevent this from happening naturally.

However, this paradigm faces two critical challenges, namely, direct encoding of $\mathbf{o}_{t+1}$ and information leakage in $(\mathbf{o}_t, \mathbf{a}_t)$. For the challenge of direct encoding of $\mathbf{o}_{t+1}$, the encoder might cheat by directly encoding $\mathbf{o}_{t+1}$ into $\mathbf{z}_t$, making reconstruction trivial but $\mathbf{z}_t$ uninformative about actual hidden dynamics. We introduce a blind decoder D_ψ that attempts to predict the next observation from $\mathbf{z}_t$ alone: $\tilde{\mathbf{o}}_{t+1} = D_\psi(\mathbf{z}_t)$. If the blind decoder can accurately predict $\mathbf{o}_{t+1}$ from $\mathbf{z}_t$ without access to $(\mathbf{o}_t, \mathbf{a}_t)$, then $\mathbf{z}_t$ contains direct observation information rather than abstract dynamics features. The encoder minimizes reconstruction error while simultaneously *maximizing* the blind decoder's prediction error. The blind decoder tries to minimize its own error, creating an adversarial objective:

$$\mathcal{L}_{\text{blind}} = \lambda_b \exp\left(-\frac{\|\mathbf{o}_{t+1} - \tilde{\mathbf{o}}_{t+1}\|^2}{\sigma_b^2}\right) \tag{8}$$

The exponential form provides bounded gradients and saturates smoothly when the blind decoder error is already high, avoiding unnecessary optimization pressure. The temperature σ_b controls sensitivity. The blind decoder is trained separately with detached gradients from the encoder.

For the second challenge of information leakage in $(\mathbf{o}_t, \mathbf{a}_t)$, ideally, $(\mathbf{o}_t, \mathbf{a}_t)$ would be sampled independently of hidden variables and domain characteristics, containing no information about $\mathbf{h}_t$ or $\mathbf{c}$. However, the policy collecting the data likely reacts to both, producing different trajectory distributions across domains. Consequently, $(\mathbf{o}_t, \mathbf{a}_t)$ contains partial information about both $\mathbf{h}_t$ and $\mathbf{c}$. The encoder only needs to extract the *remaining* information into $\mathbf{z}_t$. During translation, $(\mathbf{o}_t, \mathbf{a}_t)$ remains unchanged while $\mathbf{z}_t$ is translated. The partial $\mathbf{h}_t$ information in $(\mathbf{o}_t, \mathbf{a}_t)$ is preserved, and the translated $\mathbf{z}_t$ carries the rest, and thus the complete $\mathbf{h}_t$ semantics are maintained after domain translation. In addition, since the decoder can potentially infer domain-specific information from $(\mathbf{o}_t, \mathbf{a}_t)$ alone, the encoder has no incentive to encode $\mathbf{c}$ into $\mathbf{z}_t$. But cross-domain translation requires $\mathbf{z}_t$ to contain domain-specific content. We introduce two auxiliary components: the domain classifier and the FiLM decoder. We add a domain classifier C_ξ that predicts which domain a latent code originates from $\hat{d} = C_\xi(\mathbf{z}_t)$. By requiring the classifier to succeed, we force $\mathbf{z}_t$ to encode domain-specific information:

$$\mathcal{L}_{\text{cls}} = \lambda_c \cdot \text{BCE}(C_\xi(\mathbf{z}_t), d), \tag{9}$$

where $d \in \{0, 1\}$ is the domain label. Even with domain information in $\mathbf{z}_t$, the decoder might still infer $\mathbf{c}$ from $(\mathbf{o}_t, \mathbf{a}_t)$ and ignore that in $\mathbf{z}_t$. We replace the basic MLP decoder with Feature-wise Linear Modulation (FiLM) [33], where $\mathbf{z}_t$ conditions each layer:

$$\begin{aligned} D_\theta(\mathbf{o}_t, \mathbf{a}_t, \mathbf{z}_t) &= \text{FiLM}(\mathbf{o}_t, \mathbf{a}_t; \mathbf{z}_t) \\ &= \boldsymbol{\gamma}(\mathbf{z}_t) \odot f(\mathbf{o}_t, \mathbf{a}_t) + \boldsymbol{\beta}(\mathbf{z}_t), \end{aligned} \tag{10}$$

where f denotes the main network processing $(\mathbf{o}_t, \mathbf{a}_t)$, and $\mathbf{z}_t$ generates scale $\boldsymbol{\gamma}$ and shift $\boldsymbol{\beta}$ parameters. This architectural choice encourages the decoder to actively use information from $\mathbf{z}_t$. While these auxiliary components do not theoretically guarantee the decoder uses domain information from $\mathbf{z}_t$, we find empirically that they are effective, as reported in Section V-B2.

Finally, the full training objective combines all components and can be formulated as (11).

$$\begin{aligned} \mathcal{L}_{\text{VAE}} = & \underbrace{\|\mathbf{o}_{t+1} - \hat{\mathbf{o}}_{t+1}\|^2}_{\text{reconstruction}} + \underbrace{\beta \cdot \text{KL}\left(q_\phi(\mathbf{z}_t) \| p(\mathbf{z}_t)\right)}_{\text{regularization}} \\ & + \underbrace{\lambda_b \exp\left(-\frac{\|\mathbf{o}_{t+1} - \tilde{\mathbf{o}}_{t+1}\|^2}{\sigma_b^2}\right)}_{\text{blind penalty}} \\ & + \underbrace{\lambda_c \cdot \text{BCE}(C_\xi(\mathbf{z}_t), d)}_{\text{domain classification}} \end{aligned} \tag{11}$$

B. Unpaired Domain Translation

Between simulation domain S and real-world domain R, we implement domain translation using CycleGAN [18] with generators $G_{S \to R}$ and $G_{R \to S}$ (residual MLPs) and discriminators D_S and D_R. The generators are trained with standard adversarial loss, cycle consistency loss (ensuring $\mathbf{z}^S \approx G_{R \to S}(G_{S \to R}(\mathbf{z}^S))$), and optional identity loss. This preserves hidden variable semantics in $\mathbf{z}_t$ while transforming domain characteristics.

C. Training and Deployment Pipeline

Algorithm 1 summarizes the training procedure. We train the VAE and CycleGAN jointly, with encoder-decoder shared across both domains. This shared representation ensures that dynamics features from different domains occupy a common latent space, making translation meaningful. Alternatively, the two stages can be trained separately, where the VAE is first trained until convergence, and then the encoder is frozen and CycleGAN is trained on the extracted dynamics features.

At deployment, we apply the trained models during simulation. Given a simulator transition $(\mathbf{o}_t, \mathbf{a}_t, \mathbf{o}^{\text{S}}_{t+1})$, we encode, translate, and decode to obtain the aligned observation $\mathbf{o}^{S \to R}_{t+1}$, then *overwrite* the simulator state rather than merely transforming observations. This ensures the policy trains on trajectories that evolve according to translated dynamics. Domain randomization is disabled during training with alignment to

Algorithm 1: World Translation Training

Initialize encoder E_ϕ, decoder D_θ, blind decoder D_ψ, classifier C_ξ. Initialize generators $G_{S\to R}, G_{R\to S}$ and discriminators D_S, D_R
while *not converged* **do**
  `// Backward Dynamics Extraction` Sample transitions $\{(\mathbf{o}_t, \mathbf{a}_t, \mathbf{o}_{t+1})\} \sim \mathcal{D}_S \cup \mathcal{D}_R$
  Encode: $\mathbf{z}_t \sim q_\phi(\mathbf{z}_t \mid \mathbf{o}_t, \mathbf{a}_t, \mathbf{o}_{t+1})$
  Decode: $\hat{\mathbf{o}}_{t+1} = D_\theta(\mathbf{o}_t, \mathbf{a}_t, \mathbf{z}_t)$
  Blind predict: $\tilde{\mathbf{o}}_{t+1} = D_\psi(\mathbf{z}_t)$
  Classify: $\hat{d} = C_\xi(\mathbf{z}_t)$
  Update ϕ, θ, ξ: minimize $\mathcal{L}_{\text{VAE}}$
  Update ψ: minimize $\|\mathbf{o}_{t+1} - \tilde{\mathbf{o}}_{t+1}\|^2$ with $\mathbf{z}_t$ detached
  `// Unpaired Domain Translation` Sample domain-separated batches from $\mathcal{D}_S$ and $\mathcal{D}_R$
  Encode: $\mathbf{z}^S \sim q_\phi(\mathbf{z} \mid \mathcal{D}_S)$, $\mathbf{z}^R \sim q_\phi(\mathbf{z} \mid \mathcal{D}_R)$
  Translate: $\hat{\mathbf{z}}^R = G_{S\to R}(\mathbf{z}^S)$, $\hat{\mathbf{z}}^S = G_{R\to S}(\mathbf{z}^R)$
  Cycle: $\tilde{\mathbf{z}}^S = G_{R\to S}(\hat{\mathbf{z}}^R)$, $\tilde{\mathbf{z}}^R = G_{S\to R}(\hat{\mathbf{z}}^S)$
  Update $G_{S\to R}, G_{R\to S}$: minimize $\mathcal{L}_{\text{adv}} + \lambda_{\text{cyc}}\mathcal{L}_{\text{cyc}} + \lambda_{\text{id}}\mathcal{L}_{\text{id}}$
  Update D_S: minimize $-\mathbb{E}[\log D_S(\mathbf{z}^S)] - \mathbb{E}[\log(1 - D_S(\hat{\mathbf{z}}^S))]$
  Update D_R: minimize $-\mathbb{E}[\log D_R(\mathbf{z}^R)] - \mathbb{E}[\log(1 - D_R(\hat{\mathbf{z}}^R))]$
end

ensure deterministic simulation dynamics. Implementation details of state decomposition and coordinate frame conversions are provided in the Appendix.

V. Experiments

We evaluate the proposed World Translation method across multiple robotic platforms with various tasks. We try to answer the following questions with our experiments:

- Can World Translation accurately predict dynamics, especially under unpredictable hidden variable evolution?
- Does the learned representation capture meaningful structure about hidden variables and domains?
- What components of our method are essential?
- Does our improved dynamics modeling lead to better downstream policy transfer?

We conduct a comprehensive suite of experiments to evaluate the effectiveness of World Translation. We begin by benchmarking the dynamics modeling accuracy to verify the effectiveness of our method. To provide deeper insights, we conducted representation analysis to validate whether the learned representation functions as expected. This is followed by an ablation study to isolate the contribution of key modules. Finally, we conducted real-robot deployment as the ultimate validation, trying to demonstrate our framework's capacity to minimize the sim-to-real gap through enhanced physical performance.

A. Dynamics Modeling Accuracy

1) Baselines and Protocol: For each platform, we create source and target domains in Isaac Lab with different physics parameters, such as stiffness, damping, mass and friction, to simulate the sim-to-real domain gap for more controllable evaluation. Transition data is collected using fixed pre-trained policies.

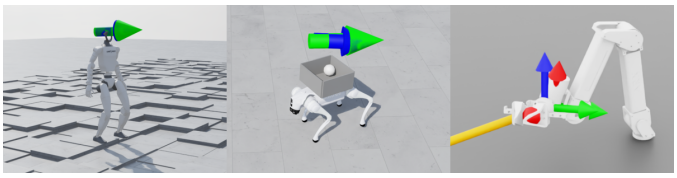


Fig. 3. Simulation environments. Left: G1 humanoid on irregular terrain. Middle: Go2 quadruped with payload (ball in box). Right: R5 manipulator under external forces.

For the evaluation protocol, we measure single-step prediction error under teacher-forcing, where the model receives ground-truth observations at each step. Autoregressive evaluation introduces confounds: using ground-truth actions causes methods with prediction error to receive inappropriate actions (computed for a different state), while recomputing actions per method couples policy behavior with dynamics error. Teacher-forcing isolates the dynamics modeling accuracy by providing identical inputs to all methods. The baselines we compared are listed in the following.

- Direct Prediction (DirectPred): Predicts from $(\mathbf{o}_t, \mathbf{a}_t)$ alone; error reflects hidden variable influence $\mathbf{h}_t$.
- Raw Simulation (RawSim): Raw simulator output; error reflects influence of domain characteristics $\mathbf{c}$ mismatch.
- Residual Dynamics (ResDyn): Predicts residual corrections using observation history for forward inference.

2) Benchmark Task: Standard continuous control benchmarks (e.g., dm_control [34]) are designed for fully observable environments or smoothly-varying latent dynamics, and do not provide paired source-target domains for evaluating sim-to-real transfer. Our contribution specifically addresses scenarios with discontinuous hidden variables. To evaluate this, we design custom tasks on robotic platforms that systematically vary hidden variable influence. For each robot (Figure 3), we design two conditions. In the first condition, the hidden variables $\mathbf{h}_t$ have minimal impact on dynamics (low-$\mathbf{h}_t$), while in another condition, $\mathbf{h}_t$ significantly affects transitions (high-$\mathbf{h}_t$).

- G1 Humanoid: Velocity tracking on flat terrain (low-$\mathbf{h}_t$) vs. irregular stepping blocks with unpredictable foot contacts (high-$\mathbf{h}_t$).
- Go2 Quadruped: Locomotion without payload (low-$\mathbf{h}_t$) vs. with 5kg ball in mounted box creating unpredictable interaction forces (high-$\mathbf{h}_t$).
- R5 Manipulator: End-effector pose tracking in its workspace (low-$\mathbf{h}_t$) vs. under 0–200N external force disturbances (high-$\mathbf{h}_t$).

3) Results: Figure 4 shows the dynamics modeling accuracy across platforms. The two reference baselines reveal the structure of each task: Direct Prediction error reflects hidden variable influence, while Raw Simulation error reflects domain gap severity. World Translation consistently outperforms both references across all conditions. It recovers hidden variable information (improving over Direct Prediction) while correcting domain mismatch (improving over Raw Simulation). In high-$\mathbf{h}_t$ conditions where DirectPred error is substantial, World

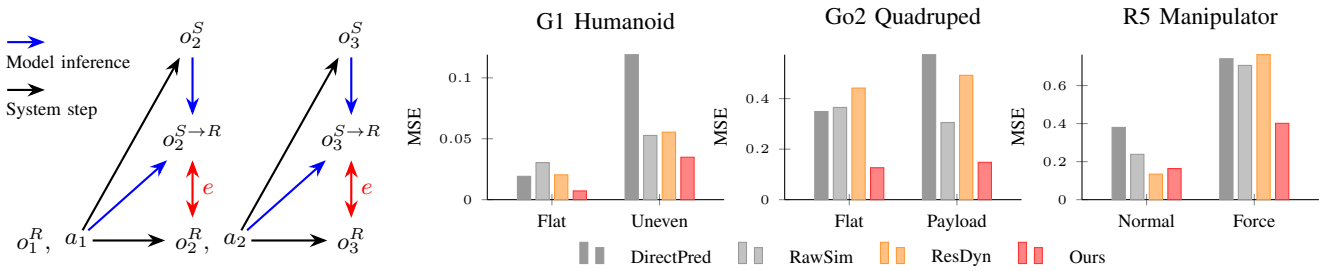


Fig. 4. Dynamics modeling evaluation. Left: teacher-forcing protocol where the model predicts translated observations $o^{S\to R}$ from simulator transitions, with error e measured against ground-truth. Right: prediction error (MSE) across three platforms. For each robot, the left bar group shows the low-$\mathbf{h}_t$ task, and the right shows the high-$\mathbf{h}_t$ task. Two reference baselines characterize different challenges: DirectPred (predicting from $(\mathbf{o}_t, \mathbf{a}_t)$ alone) reflects hidden variable influence; RawSim reflects domain gap. These references are not directly comparable in magnitude but provide a within-condition context. World Translation consistently improves over both baselines, with the largest gains where hidden variable influence is strongest.

TABLE I
HIDDEN VARIABLE ($\mathbf{h}_t$) PREDICTION R^2 FROM DIFFERENT REPRESENTATIONS. COMBINED $(\mathbf{o}_t, \mathbf{a}_t, \mathbf{z}_t)$ RESULTS COMPARE W/O (BASIC VAE) VS FULL MODEL, SHOWING THAT AUXILIARY COMPONENTS DO NOT COMPROMISE $\mathbf{h}_t$ PRESERVATION.

Input	w/o	Full
Baseline		
$(\mathbf{o}, \mathbf{a})^S$	0.73	
$(\mathbf{o}, \mathbf{a})^R$	0.66	
$(\mathbf{o}_t, \mathbf{a}_t, \mathbf{o}_{t+1})^S$	0.78	
$(\mathbf{o}_t, \mathbf{a}_t, \mathbf{o}_{t+1})^R$	0.73	
Combined $(\mathbf{o}_t, \mathbf{a}_t, \mathbf{z}_t)$		
$(\mathbf{o}, \mathbf{a})^S, \mathbf{z}^S$	0.81	0.80
$(\mathbf{o}, \mathbf{a})^R, \mathbf{z}^R$	0.68	0.72
$(\mathbf{o}, \mathbf{a})^S, \mathbf{z}^{S\to R}$	0.67	0.68
$(\mathbf{o}, \mathbf{a})^R, \mathbf{z}^{R\to S}$	0.75	0.78
$(\mathbf{o}, \mathbf{a})^S, \mathbf{z}^{S\to R\to S}$	0.78	0.78
$(\mathbf{o}, \mathbf{a})^R, \mathbf{z}^{R\to S\to R}$	0.67	0.67

Translation achieves the largest improvements (G1 Uneven: 0.035 vs. DirectPred 0.119; R5 Force: 0.401 vs. DirectPred 0.741). Residual Dynamics shows mixed results: it improves over RawSim in some conditions but shows limited gains over DirectPred in high-$\mathbf{h}_t$ tasks. Forward prediction from observation history can partially correct domain characteristics but struggles to recover discontinuous hidden variables, which is where backward extraction helps most.

B. Representation Analysis

To understand what the learned dynamics features indeed capture and why auxiliary components are necessary, we analyze the representation from multiple perspectives. We focus on the ARX R5 manipulator task in Section V-A2, where external forces serve as hidden variables and motor parameters serve as domain characteristics. In this case, hidden variables and domain characteristics are tightly coupled, as the force's effect on dynamics depends on the motor response. This represents a challenging case for disentanglement. Discrete hidden variables like contact events in locomotion would be more naturally separable from motor parameters.

1) Hidden Variable Information $\mathbf{h}_t$*:* To verify that transitions preserve hidden variable information, we train lightweight decoders to predict the ground-truth hidden variable $\mathbf{h}_t$ (normalized external force) from different input representations. Table I reports the R^2 score for predicting force magnitude. Several observations stand out:

- Information leakage: $(\mathbf{o}_t, \mathbf{a}_t)$ alone explains 66–73% of $\mathbf{h}_t$ variance, confirming the information leakage discussed in Section IV.
- Combined matches transition: $(\mathbf{o}_t, \mathbf{a}_t, \mathbf{z}_t)$ achieves 68–81% R^2, matching the full transition's predictive power (73–78%).
- Translation preserves $\mathbf{h}_t$: After translation, combined representations maintain similar predictive power (67–78% R^2). The $\mathbf{h}_t$ information is largely preserved through translation.
- Auxiliary components preserve $\mathbf{h}_t$: Comparing w/o and Full columns, the results are nearly identical. The domain classifier and FiLM *add* domain information $\mathbf{c}$ to $\mathbf{z}_t$ without compromising $\mathbf{h}_t$ preservation.

2) Domain Characteristic Information $\mathbf{c}$*:* To verify that our auxiliary components successfully encourage $\mathbf{z}_t$ to encode domain characteristics, we conduct domain classification experiments. We train a binary classifier as a probe to measure how much domain information each representation contains. The classifier is trained with ground-truth domain labels and directly predicts S or R from different input representations. We evaluate on: (1) $(\mathbf{o}_t, \mathbf{a}_t)$ alone to measure baseline information leakage, (2) $\mathbf{z}_t$ alone to measure how much domain information the encoder captures, and (3) combined $(\mathbf{o}_t, \mathbf{a}_t, \mathbf{z}_t)$ to measure whether $\mathbf{z}_t$ adds domain information beyond $(\mathbf{o}_t, \mathbf{a}_t)$. For the combined input, $(\mathbf{o}_t, \mathbf{a}_t)$ is always from the source domain to match the practical application scenario where we observe transitions in one domain and translate $\mathbf{z}_t$ to another. Our results displayed in table II show:

- Information leakage exists: $(\mathbf{o}_t, \mathbf{a}_t)$ alone achieves 79–85% domain classification accuracy, confirming that observation-action distributions differ across domains.
- Domain information in $\mathbf{z}_t$: Without auxiliary components (w/o), $\mathbf{z}_t$ achieves only 64–67% accuracy, even *worse* than $(\mathbf{o}_t, \mathbf{a}_t)$. With auxiliary components (Full), $\mathbf{z}_t$ achieves 94%, significantly exceeding $(\mathbf{o}_t, \mathbf{a}_t)$. This confirms the encoder has no incentive to encode domain characteristics without explicit encouragement.
- Translation effect on $\mathbf{z}_t$: Without auxiliary components, $\mathbf{z}^{S\to R}$ achieves only 43% accuracy as R (below chance), indicating translation fails. With auxiliary components,

TABLE II

DOMAIN CLASSIFICATION ACCURACY FROM DIFFERENT REPRESENTATIONS. "W/O" DENOTES THE BASIC VAE WITHOUT DOMAIN CLASSIFIER AND FILM; "FULL" DENOTES THE COMPLETE MODEL.

Input	Target	w/o	Full
Information in $(\mathbf{o}_t, \mathbf{a}_t)$			
$(\mathbf{o}, \mathbf{a})^S$	S	0.7894	
$(\mathbf{o}, \mathbf{a})^R$	R	0.8479	
Information in $\mathbf{z}_t$			
$\mathbf{z}^S$	S	0.6729	0.9407
$\mathbf{z}^R$	R	0.6412	0.9365
$\mathbf{z}^{S\to R}$	R	0.4288	0.9124
$\mathbf{z}^{R\to S}$	S	0.4936	0.8327
$\mathbf{z}^{S\to R\to S}$	S	0.7015	0.8427
$\mathbf{z}^{R\to S\to R}$	R	0.6099	0.9820
Combined $(\mathbf{o}_t, \mathbf{a}_t, \mathbf{z}_t)$			
$(\mathbf{o}, \mathbf{a})^S, \mathbf{z}^S$	S	0.7857	0.9443
$(\mathbf{o}, \mathbf{a})^R, \mathbf{z}^R$	R	0.8326	0.9414
$(\mathbf{o}, \mathbf{a})^S, \mathbf{z}^{S\to R}$	R	0.2702	0.9070
$(\mathbf{o}, \mathbf{a})^R, \mathbf{z}^{R\to S}$	S	0.2404	0.8309
$(\mathbf{o}, \mathbf{a})^S, \mathbf{z}^{S\to R\to S}$	S	0.7952	0.8459
$(\mathbf{o}, \mathbf{a})^R, \mathbf{z}^{R\to S\to R}$	R	0.8227	0.9837

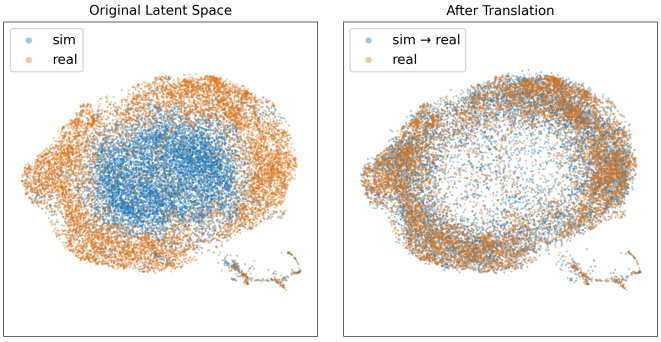


Fig. 5. UMAP visualization of dynamics features on the R5 manipulator task. Left: original latent space shows clear separation between sim (blue) and real (orange) domains, reflecting domain characteristics $\mathbf{c}$ encoded in $\mathbf{z}_t$. Right: after translation, $\mathbf{z}^{S\to R}$ (blue) aligns with real domain $\mathbf{z}^R$ (orange), demonstrating successful cross-domain mapping.

$\mathbf{z}^{S\to R}$ achieves 91% as R, showing successful domain transformation.

- Combined $(\mathbf{o}, \mathbf{a}), \mathbf{z}$ after translation: Without auxiliary components, $(\mathbf{o}, \mathbf{a})^S, \mathbf{z}^{S\to R}$ achieves only 27% as R because the $(\mathbf{o}, \mathbf{a})^S$ signal dominates. With auxiliary components, it achieves 91% as R, where the translated $\mathbf{z}_t$ successfully overrides the domain signal in $(\mathbf{o}, \mathbf{a})$.

Auxiliary components are essential for translation to succeed: without them, translated representations cannot override the domain signal in $(\mathbf{o}_t, \mathbf{a}_t)$, making cross-domain prediction infeasible.

3) Latent Space Visualization: We visualize the latent space using UMAP [35] in Figure 5. The original latent space shows clear separation between simulation and real domains, reflecting the domain characteristics $\mathbf{c}$ successfully encoded into $\mathbf{z}_t$ by our auxiliary components. This separation is necessary for the translation network to learn meaningful cross-domain mappings. After translation, the translated sim features $\mathbf{z}^{S\to R}$ align with real domain features $\mathbf{z}^R$, confirming that the translation successfully transforms dynamics representations across domains. In addition, as shown in Table I, the w/o and full columns are nearly identical for combined $(\mathbf{o}_t, \mathbf{a}_t, \mathbf{z}_t)$,

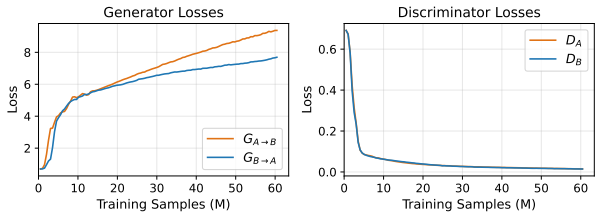


Fig. 6. Training collapse in observation-level translation. Discriminator losses (right) drop to near-zero while generator losses (left) diverge, indicating the discriminators trivially distinguish domains using $(\mathbf{o}_t, \mathbf{a}_t)$ without learning meaningful dynamics mappings.

TABLE III

ABLATION OF TRANSLATION COMPONENTS. CYCLE ERROR MEASURES ROUND-TRIP RECONSTRUCTION $\|\mathbf{z} - G(G(\mathbf{z}))\|$; IDENTITY ERROR MEASURES UNNECESSARY MODIFICATION $\|\mathbf{z}^R - G_{S\to R}(\mathbf{z}^R)\|$.

Variant	Cycle Error	Identity Error	Pred. Error
Full Model	0.034	0.233	0.401
w/o Cycle Consistency	0.273	0.004	0.681
w/o Identity Loss	0.026	0.463	0.483

confirming that our auxiliary components *add* domain information $\mathbf{c}$ to $\mathbf{z}_t$ without *removing* hidden variable $\mathbf{h}_t$.

C. Ablation Study

We study the necessity of key design choices in World Translation. Experiments are conducted on the ARX R5 manipulator unless otherwise noted.

1) Dynamics Extraction: World Translation performs cross-domain prediction through Equation (7). At deployment, this pipeline can be encapsulated as a single mapping:

$$(\mathbf{o}_t, \mathbf{a}_t, \mathbf{o}^A_{t+1}) \xrightarrow{G_{A\to B}} \mathbf{o}^B_{t+1} \tag{12}$$

To make clear if the intermediate dynamics feature $\mathbf{z}_t$ is necessary or not, we train a generator that maps $(\mathbf{o}_t, \mathbf{a}_t, \mathbf{o}^A_{t+1}) \to \mathbf{o}^B_{t+1}$ without explicit dynamics extraction. The discriminator judges transition tuples $(\mathbf{o}_t, \mathbf{a}_t, \mathbf{o}_{t+1})$ to distinguish real from generated samples. We observed that the direct observation-level translation suffers from immediate training collapse, as illustrated in Figure 6, where the discriminator losses drop to near-zero while generator losses diverge. This occurs because $(\mathbf{o}_t, \mathbf{a}_t)$ alone achieves 79–85% domain classification accuracy (Section V-B2), allowing discriminators to distinguish domains without examining $\mathbf{o}_{t+1}$. With no useful gradient signal, the generators cannot learn meaningful cross-domain mappings. By abstracting to $\mathbf{z}_t$, our approach isolates dynamics-relevant information from domain-leaking $(\mathbf{o}_t, \mathbf{a}_t)$, enabling stable adversarial training.

2) Translation Components: We ablate components of the CycleGAN translation network, and the experiment results are listed in Table III. Without cycle consistency, cycle error increases 8 times, from 0.034 to 0.273, and prediction error degrades significantly. Identity error drops to near-zero, suggesting the generator becomes overly conservative, meaning it avoids modifying inputs but consequently fails to perform meaningful cross-domain translation. Without identity loss, identity error doubles, from 0.233 to 0.463, indicating the generator unnecessarily distorts in-domain samples.

TABLE IV
DOWNSTREAM POLICY PERFORMANCE ON GO2 WITH PAYLOAD. ALL METHODS TRACK THE SAME ANGULAR VELOCITY COMMAND SEQUENCE.

Method	Tracking (rad/s)	Stability (rad)	Power (W)
No Adaptation	0.2475	0.0517	28.07
Domain Rand.	0.3306	0.0694	21.95
World Translation	0.1786	0.0513	29.55

D. Downstream Policy Performance on Real Robot

We conducted real robot experiment to further evaluate whether improved dynamics modeling translates to better policy transfer in a practical sim-to-real deployment scenario. The real-world experiment setup is illustrated in Figure 7.

1) Setup: We evaluate on the real Go2 quadruped with the payload task described earlier in Section V-A2, where a ball moves freely inside a box mounted on the robot. All methods start from a base policy trained in simulation without domain randomization. This base policy is then fine-tuned using different adaptation strategies. The employed baseline for comparison are shown below.

- No Adaptation: The base policy is deployed directly without fine-tuning.
- Domain Randomization: Fine-tuned with randomized physics parameters (base mass -1 to $+3$ kg, friction 0.3–1.2, actuator gains 0.8–1.25$\times$) and additive observation noise for 1000 steps.
- World Translation (Ours): Fine-tuned using translated dynamics. We collect 20 minutes of real-world transitions using the base policy, train the translation model, then fine-tune the policy for 1000 steps using translated dynamics predictions.

2) Metrics: All policies track the same recorded commanded angular velocity sequence, enabling controlled comparison. We evaluate using metrics computable from proprioceptive observations, each corresponding to a reward term during training:

- Angular Velocity Tracking Error: RMSE between commanded and measured angular velocity. Lower is better.
- Orientation Stability: Root mean square of pitch and roll standard deviations, $\sqrt{\sigma_{\text{pitch}}^2 + \sigma_{\text{roll}}^2}$. Lower is better.
- Average Power: Mean joint power consumption $\frac{1}{T}\int |\boldsymbol{\tau} \cdot \dot{\mathbf{q}}|\, dt$. Lower is better.

3) Results: Table IV shows downstream policy performance. World Translation achieves the lowest tracking error (0.18 rad/s vs. 0.25 for No Adaptation and 0.33 for Domain Randomization) while maintaining similar stability to the No Adaptation baseline. Domain Randomization exhibits the highest tracking error but lowest power consumption, suggesting overly conservative behavior that sacrifices responsiveness for energy efficiency. World Translation and No Adaptation show comparable stability and power consumption, with the primary difference being tracking accuracy.

These results provide empirical evidence that improved dynamics modeling can translate to better command tracking in downstream policy transfer, though the effects on other metrics are modest.



Fig. 7. Real-world deployment of the Go2 quadruped with the payload task. Left: simulation setup for training, right: real-world deployment for evaluation.

VI. DISCUSSION

World Translation shares limitations common to learned dynamics models: it requires diverse training data, prediction accuracy depends on information recoverable from observations, and it cannot extrapolate to unseen dynamics regimes.

Several limitations are specific to our approach. First, translation is more challenging when h_t and c are tightly coupled; external forces in the R5 task (section V-A2) affect dynamics through domain-specific motor responses, whereas discrete contact events in locomotion are more naturally separable. Second, our state overwriting procedure leaves hidden simulator states unchanged, which may cause inconsistencies for simulators with complex internal dependencies. Third, our current implementation trains on single-step transitions. Prediction errors compound over multi-step rollouts. Finally, our indirect optimization through cycle consistency may sacrifice prediction accuracy compared to direct supervision, though it enables learning from unpaired data.

More broadly, our reliance on implicit representations creates challenges throughout the pipeline. The learned z_t lacks interpretable structure, making training difficult to diagnose when competing objectives become unstable, and validation impossible without indirect proxies like downstream policy performance. Developing more interpretable dynamics representations remains an open direction.

VII. CONCLUSION AND FUTURE WORK

We presented World Translation, a novel dynamics model learning framework that addresses sim-to-real transfer by reformulating dynamics alignment as unpaired domain translation. Backward dynamics extraction captures hidden variable effects from observed transitions without temporal smoothness assumptions, while cycle-consistent translation preserves these semantics across domains. Experiments on humanoid, quadruped, and manipulator platforms demonstrate improved dynamics modeling over forward prediction baselines, with the largest gains when hidden variables evolve discontinuously. Limitations include sensitivity to tight coupling between $\mathbf{h}_t$ and $\mathbf{c}$, and error compounding over multi-step rollouts.

Potential future extensions include combining with autoregressive training to handle compounding errors, incorporating visual observations for vision-based policies, online model updates during deployment, and combining World Translation with action correction methods where translated dynamics provide target states.

References

[1] J. Hwangbo, J. Lee, A. Dosovitskiy, D. Bellicoso, V. Tsounis, V. Koltun, and M. Hutter, "Learning agile and dynamic motor skills for legged robots," *Science Robotics*, vol. 4, no. 26, p. eaau5872, 2019.

[2] J. Lee, J. Hwangbo, L. Wellhausen, V. Koltun, and M. Hutter, "Learning quadrupedal locomotion over challenging terrain," *Science robotics*, vol. 5, no. 47, p. eabc5986, 2020.

[3] R. Singh, A. Allshire, A. Handa, N. Ratliff, and K. Van Wyk, "Dextrah-rgb: Visuomotor policies to grasp anything with dexterous hands," *arXiv preprint arXiv:2412.01791*, 2024.

[4] 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 *2017 IEEE/RSJ international conference on intelligent robots and systems (IROS)*. IEEE, 2017, pp. 23–30.

[5] X. B. Peng, M. Andrychowicz, W. Zaremba, and P. Abbeel, "Sim-to-real transfer of robotic control with dynamics randomization," in *2018 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2018, pp. 3803–3810.

[6] 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 preprint arXiv:1910.07113*, 2019.

[7] J. Josifovski, M. Malmir, N. Klarmann, B. L. Žagar, N. Navarro-Guerrero, and A. Knoll, "Analysis of randomization effects on sim2real transfer in reinforcement learning for robotic manipulation tasks," in *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2022, pp. 10 193–10 200.

[8] W. Xiao, S. Lyu, Z. Gong, R. Wang, and D. Wang, "Learning robotic policy with imagined transition: Mitigating the trade-off between robustness and optimality," *arXiv preprint arXiv:2503.10484*, 2025.

[9] L. Ljung, "System identification," in *Signal analysis and prediction*. Springer, 1998, pp. 163–173.

[10] Y. Chebotar, A. Handa, V. Makoviychuk, M. Macklin, J. Issac, N. Ratliff, and D. Fox, "Closing the sim-to-real loop: Adapting simulation randomization with real world experience," in *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, 2019, pp. 8973–8979.

[11] D. Hafner, T. Lillicrap, I. Fischer, R. Villegas, D. Ha, H. Lee, and J. Davidson, "Learning latent dynamics for planning from pixels," in *International conference on machine learning*. PMLR, 2019, pp. 2555–2565.

[12] D. Hafner, T. Lillicrap, J. Ba, and M. Norouzi, "Dream to control: Learning behaviors by latent imagination," in *International Conference on Learning Representations (ICLR)*, 2020.

[13] D. Hafner, J. Pasukonis, J. Ba, and T. Lillicrap, "Mastering diverse domains through world models," *arXiv preprint arXiv:2301.04104*, 2023.

[14] N. Hansen, H. Su, and X. Wang, "Td-mpc2: Scalable, robust world models for continuous control," *arXiv preprint arXiv:2310.16828*, 2023.

[15] F. Golemo, A. A. Taiga, A. Courville, and P.-Y. Oudeyer, "Sim-to-real transfer with neural-augmented robot simulation," in *Conference on Robot Learning*. PMLR, 2018, pp. 817–828.

[16] E. Heiden, D. Millard, E. Coumans, Y. Sheng, and G. S. Sukhatme, "Neuralsim: Augmenting differentiable simulators with neural networks," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021, pp. 9474–9481.

[17] N. Sontakke, H. Chae, S. Lee, T. Huang, D. W. Hong, and S. Hal, "Residual physics learning and system identification for sim-to-real transfer of policies on buoyancy assisted legged robots," in *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2023, pp. 392–399.

[18] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros, "Unpaired image-to-image translation using cycle-consistent adversarial networks," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 2223–2232.

[19] M.-Y. Liu, T. Breuel, and J. Kautz, "Unsupervised image-to-image translation networks," *Advances in neural information processing systems*, vol. 30, 2017.

[20] X. Huang, M.-Y. Liu, S. Belongie, and J. Kautz, "Multimodal unsupervised image-to-image translation," in *Proceedings of the European conference on computer vision (ECCV)*, 2018, pp. 172–189.

[21] J. Tan, T. Zhang, E. Coumans, A. Iscen, Y. Bai, D. Hafner, S. Bohez, and V. Vanhoucke, "Sim-to-real: Learning agile locomotion for quadruped robots," in *Robotics: Science and Systems (RSS)*, 2018.

[22] G. Tiboni, K. Arndt, and V. Kyrki, "Dropo: Sim-to-real transfer with offline domain randomization," *Robotics and Autonomous Systems*, vol. 166, p. 104432, 2023.

[23] B. Mehta, M. Diaz, F. Golemo, C. J. Pal, and L. Paull, "Active domain randomization," in *Conference on Robot Learning*. PMLR, 2020, pp. 1162–1176.

[24] M. Deisenroth and C. E. Rasmussen, "Pilco: A model-based and data-efficient approach to policy search," in *Proceedings of the 28th International Conference on machine learning (ICML-11)*, 2011, pp. 465–472.

[25] K. Chua, R. Calandra, R. McAllister, and S. Levine, "Deep reinforcement learning in a handful of trials using probabilistic dynamics models," *Advances in neural information processing systems*, vol. 31, 2018.

[26] P. Wu, A. Escontrela, D. Hafner, P. Abbeel, and K. Goldberg, "Daydreamer: World models for physical robot learning," in *Conference on robot learning*. PMLR, 2023, pp. 2226–2240.

[27] J. Gao, M. Y. Michelis, A. Spielberg, and R. K. Katzschmann, "Sim-to-real of soft robots with learned residual physics," *IEEE Robotics and Automation Letters*,

2024.

[28] J. Hanna and P. Stone, "Grounded action transformation for robot learning in simulation," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 31, no. 1, 2017.

[29] H. Karnan, S. Desai, J. P. Hanna, G. Warnell, and P. Stone, "Reinforced grounded action transformation for sim-to-real transfer," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2020.

[30] T. He, J. Luo, W. He, T. Xiao, Y. Zhang, Z. Shi, S. Fidler, Y. Zhu, and L. Liu, "Asap: Aligning simulation and real-world physics for learning agile humanoid whole-body skills," in *Robotics: Science and Systems (RSS)*, 2025.

[31] K. Rao, C. Harris, A. Irpan, S. Levine, J. Ibarz, and M. Khansari, "Rl-cyclegan: Reinforcement learning aware simulation-to-real," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 11 157–11 166.

[32] D. Ho, K. Rao, Z. Xu, E. Jang, M. Khansari, and Y. Bai, "Retinagan: An object-aware approach to sim-to-real transfer," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021, pp. 10 920–10 926.

[33] E. Perez, F. Strub, H. De Vries, V. Dumoulin, and A. Courville, "Film: Visual reasoning with a general conditioning layer," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 32, no. 1, 2018.

[34] Y. Tassa, Y. Doron, A. Muldal, T. Erez, Y. Li, D. d. L. Casas, D. Budden, A. Abdolmaleki, J. Merel, A. Lefrancq *et al.*, "Deepmind control suite," *arXiv preprint arXiv:1801.00690*, 2018.

[35] L. McInnes, J. Healy, and J. Melville, "Umap: Uniform manifold approximation and projection for dimension reduction," *arXiv preprint arXiv:1802.03426*, 2018.