Learning Interactive Real-World Simulators

Mengjiao Yang[†,◇], **Yilun Du**[♮], **Kamyar Ghasemipour**[◇],
Jonathan Tompson[◇], **Dale Schuurmans**[◇,‡], **Pieter Abbeel**[†]
[†]UC Berkeley, [◇]Google DeepMind, [♮]MIT, [‡]University of Alberta
sherryy@{berkeley.edu, google.com}
universal-simulator.github.io

Abstract

Generative models trained on internet data have revolutionized how text, image, and video content can be created. Perhaps the next milestone for generative models is to simulate realistic experience in response to actions taken by humans, robots, and other interactive agents. Applications of a real-world simulator range from controllable content creation in games and movies, to training embodied agents purely in simulation that can be directly deployed in the real world. We explore the possibility of learning a universal simulator (UniSim) of real-world interaction through generative modeling. We first make the important observation that natural datasets available for learning a real-world simulator are often rich along different axes (e.g., abundant objects in image data, densely sampled actions in robotics data, and diverse movements in navigation data). With careful orchestration of diverse datasets, each providing a different aspect of the overall experience, UniSim can emulate how humans and agents interact with the world by simulating the visual outcome of both high-level instructions such as "open the drawer" and low-level controls such as "move by x, y" from otherwise static scenes and objects. There are numerous use cases for such a real-world simulator. As an example, we use UniSim to train both high-level vision-language planners and low-level reinforcement learning policies, each of which exhibit zero-shot real-world transfer after training purely in a learned real-world simulator. We also show that other types of intelligence such as video captioning models can benefit from training with simulated experience in UniSim, opening up even wider applications. Video demos can be found at universal-simulator.github.io.

1 Introduction

Generative models trained on internet data can now produce highly realistic text [1], speech [2], image [3], and video [4]. Perhaps the ultimate goal of generative models is to be able to simulate every aspect of the human experienced world, from how cars are driven on a street to how furniture is assembled and meals prepared. With a comprehensive real-world simulator, humans can "interact" with diverse scenes and objects, robots can learn from simulated experience without risking physical damage, and a vast amount of "real-world" data can be simulated to train other types of machine intelligence.

One roadblock to building such a real-world simulator lies in the datasets that are available. While there are billions of texts, images, and video snippets available on the internet, different datasets cover different information axes, and these have to be brought together to simulate realistic experience of the world. For instance, paired text-image data contains rich scenes and objects but little movement [5, 6, 7], video captioning and question answering data contain rich high-level activity descriptions but little low-level movement detail [8, 9], human activity data contains rich human action but little mechanical motion [10, 11], and robotics data contains rich robot action but are limited in quantity [12, 13]. Since different datasets are curated by different industrial or research communities for different tasks,

NeurIPS 2023 Workshop on Generalization in Planning (GenPlan 2023)

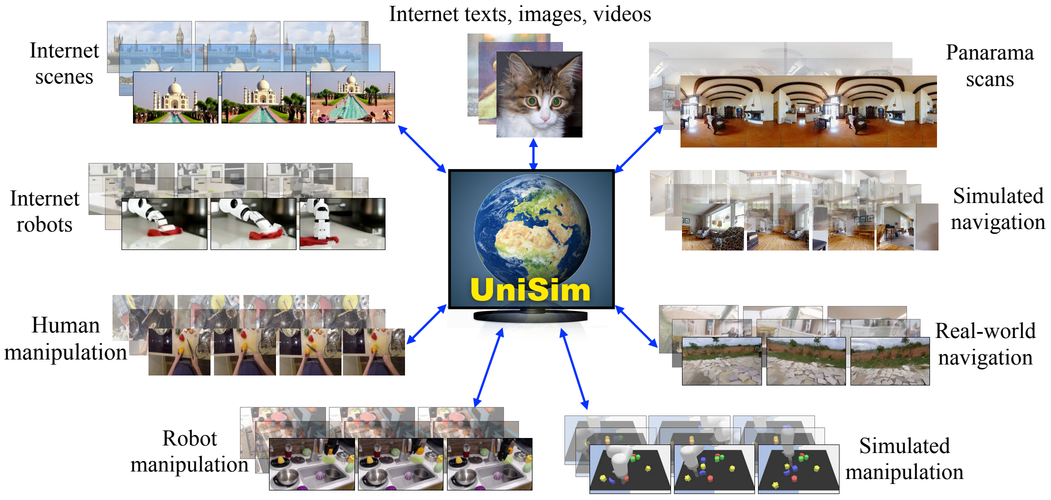

Figure 1: **A universal Simulator (UniSim).** UniSim is a simulator of the real-world that learns from broad data rich in different axes including objects, scenes, human activities, motions in navigation and manipulation, panorama scans, and simulations and renderings.

divergence in information is natural and hard to overcome, posing difficulties to building a real-world simulator that seeks to capture realistic experience of the world we live in.

In this work, we take the first steps towards building a universal simulator (UniSim) of real-world interaction through generative modeling. Specifically, we propose to combine a wealth of data—ranging from internet text-image pairs, to motion and action rich data from navigation, manipulation, human activities, robotics, and data from simulations and renderings—in a conditional video generation framework. With careful orchestration of data rich along different axes, we show that UniSim can successfully merge the different axes of experience and generalize beyond the data, enabling rich interaction through fine-grained motion control of otherwise static scenes and objects. In simulating long-horizon interactions with UniSim, we develop a fundamental connection between autoregressive video generation during inference and performing rollouts in a partially observable Markov decision processes (POMDPs) [14, 15]. As a result, UniSim can simulate long-horizon interactions consistently across video generation boundaries.

While the potential applications of UniSim are vast, we demonstrate a few practical use cases centered around using simulated experience from UniSim. We first demonstrate how an embodied vision-language planner can be trained to complete long-horizon goal-conditioned tasks through hindsight relabeling of simulated experience [16]. In addition to high-level planning, we further illustrate how UniSim can enable learning low-level control policies by leveraging model-based reinforcement learning [17]. We show that both the high-level vision-language planner and the low-level control policy, while trained purely in simulation, can generalize to real robot settings in a zero-shot manner, achieving one step towards bridging the sim-to-real gap in embodied learning [18, 19, 20]. This is enabled by using simulators that are nearly visually indistinguishable from the real world. Lastly, we note that UniSim can be used to simulate rare events where data collection is expensive or dangerous (e.g., crashes in self-driving cars). Such simulated videos can then be used to improve other machine intelligence such as rare event detectors, suggesting broad applications of UniSim beyond embodied learning. The main contributions can be summarized as follows:

- We take the first step toward building a universal simulator (UniSim) of real-world interaction by combining diverse datasets rich in different axes—such as objects, scenes, actions, motions, language, and motor controls—in a unified video generation framework.
- We establish the connection between conditional video generation and partially observable Markov decision process (POMDP), and leverage multi-frame history conditioning to simulate consistent long-horizon interactions from otherwise static scenes and objects.
- We illustrate how UniSim can simulate realistic experiences for training embodied planners, low-level control policies, and video captioning models, equipping these other forms of machine intelligence with the ability to generalize to the real world when trained purely in simulation, thereby bridging the sim-to-real gap.

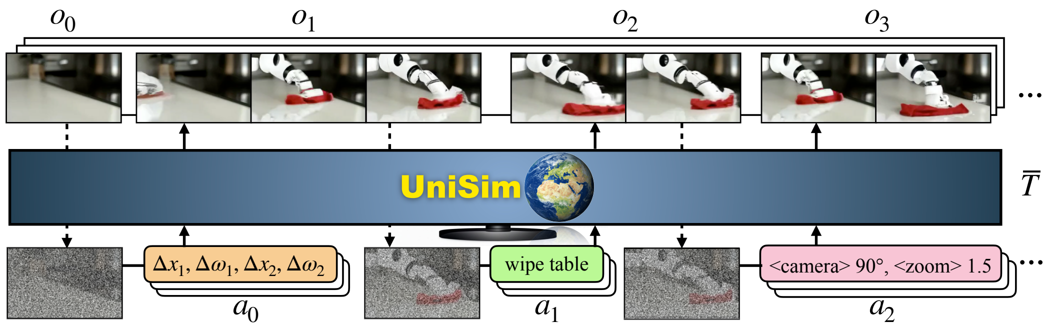

Figure 2: **Training and inference of UniSim.** UniSim ($\overline{T}$) is a video diffusion model trained to predict the next (variable length) observation frames (o_1) given the noisy version of the previous observation (o_0) and action input (a_0). UniSim can handle actions of varying modalities such as motor controls of varying length ($\Delta x_1, \Delta\omega_1, \Delta x_2, ...$), language descriptions of the action ("wipe table"), and actions extracted from camera motions and other sources. Dotted arrows add noise to the true video during training or to the previously generated video during inference to autoregressively rollout observations in supporting long-horizon interactions.

2 Learning Interactive Real-World Simulators

The major differences between an interactive real-world simulator and typical video generation models are that a simulator requires support for (1) a diverse set of actions and (2) long-horizon rollouts. In this section, we first enable action-rich interaction by combining datasets rich in different axes through joint training, and then enable long-horizon interaction through history conditioning inspired by rollouts in a POMDP.

2.1 Orchestrating Datasets Rich in Different Axes

A realistic world simulator should be able to simulate diverse scenes, objects, human activities, robot actions, camera motions, other aspects of the world. While this seems difficult, there already exist billions of text, image, and video samples on the internet, as well as various robotic, 3D, and navigation datasets scattered across institutions. The main difficulty comes down effectively extracting information from broad datasets rich in these different axes and fusing this information into a single learned simulator.

Extracting Information from Broad Data. Although data exists across many different modalities on the Internet, our focus in this paper will be on visual observations of the world and actions that cause changes to these visual observations. Note that this choice inevitably misses states that are not visual (e.g., temperature-dependent friction), but we only focus on problems that can be visually captured. In such a scenario, if we can express the two modalities in terms of a universal interface that relates videos and text, we can fuse the information between different datasets by training a simulator that operates through this universal interface. Thus, the key challenge is to extract then fuse observations and actions from different types of datasets into a common format, which we describe below. The datasets we included in this study are as follows (further details of the datasets used to train UniSim are given in Appendix 9).

- **Simulated execution and renderings.** While annotating actions for real-world videos is expensive, simulation engines such as Habitat [21] and Language Table [22] are able to render a wide variety of actions. We use datasets previously collected from these simulators, i.e., Habitat object navigation using HM3D [23] and Language Table Data from [24] to train UniSim. We extract text descriptions as actions when available. For simulated continuous control actions, we encode them via language embeddings and concatenate the text embeddings with discretized control values.
- **Real robot data.** An increasing amount of video data of real-robot executions paired with task descriptions such as the Bridge Data [25, 26] and data that enabled RT-1 and RT-2 [27] are becoming increasingly available. Despite low-level control actions often being different across robots, the task descriptions can serve as high-level actions in UniSim. We further include discretize continuous controls actions when available similar to simulated robotics data.
- **Human activity videos.** Human activity data such as Ego4D [11], EPIC-KITCHENS [28], and Something-Something [29] have videos filled with human activities. Different from low-level robot controls, these activities are high-level actions that humans take to interact with the world. But these actions are sometimes provided as labels for video classification or activity recognition

tasks [29]. In this case, we convert the video labels into text actions. In addition, we subsample the videos to construct chunks of observations at a frame rate that captures meaningful actions.

- **Panorama scans.** There exists a wealth of 3D scans such as Matterport3D [30]. These static scans do not contain actions. We construct actions (e.g., turn left) by truncating panorama scans and utilize information such as change in camera poses between two images.
- **Internet text-image data.** Paired text-image datasets such as LAION [31] contain rich static objects without actions. However, even though the images are static, the text labels often contain motion information such as "a person walking". In addition, internet text-image data can describe a richer set of objects than other datasets above, making them good candidates for training UniSim. To use text-image data in UniSim, we treat individual images as single-frame videos and text labels as actions.

For each of these datasets, we process text tokens into continuous representations using T5 language model embeddings [32] to better fuse with continuous actions such as robot controls.

Fuse Information into UniSim. Given the observation and action data extracted from the broad datasets above, we train a diffusion model (architecture and training details in Appendix 10) to predict observations conditioned on actions and noisy previous observations as shown in Figure 2. During training, Gaussian noise with a fixed schedule is added to the true previous observation as a part of the forward process of the diffusion model [33], then UniSim learns to denoise the previous noisy observation to the next observation conditioned on the input action. Since the observations from different environments have all been converted to videos, while actions of different modalities (e.g., text descriptions, motor controls, camera angles) have all been converted to continuous embeddings, UniSim can learn a single world model across all datasets.

2.2 Enabling Long-Horizon Interactions through Rollouts in POMDP

While combining diverse data might enable rich interaction, the true value of a simulator like UniSim lies in simulating long episodes to enable optimizing decisions through search [34], planning [35], optimal control [36], or reinforcement learning [37]. In this section, we show that inference in UniSim is analogous to performing rollouts in a partially observable Markov decision process (POMDP) [14]. This connection enables UniSim to support learning decision making policies with established algorithms.

Real World as A POMDP. A POMDP can be defined as a tuple $\mathcal{M} := \langle S, A, \mathcal{O}, \mathcal{R}, \mathcal{T}, \mathcal{E} \rangle$ consisting of state, action, and observation spaces as well as reward, transition, and observation emission functions. A POMDP can characterize interactions with the real world, where $s_t \in S$ is the true state of the world, $o_t \in O$ contains video frames, and $a_t \in A$ contains actions carried out by humans or agents, all at interactive step t. A policy π can learn to choose actions that lead to high rewards through interacting with $\mathcal{M}$.

UniSim as Transition Function. Given an observation o_t from $\mathcal{M}$ at interaction step t, UniSim can parametrize the transition function $\mathcal{T}$ to sample the next observation $o_{t+1} \sim \overline{\mathcal{T}}(\cdot|o_t, a_t)$ conditioned o_t and action input a_t. Note that the distribution of temporally extended observations can be factorized into segments:

$$\overline{\mathcal{T}}(o_l|o_{l-1}, a_{l-1}) = \overline{\mathcal{T}}(o_t, o_{t+1}|o_{t-1}, a_{t-1}, a_t) = \overline{\mathcal{T}}(o_t|o_{t-1}, a_{t-1})\overline{\mathcal{T}}(o_{t+1}|o_t, a_t), \quad (1)$$

where $o_l = [o_t, o_{t+1}]$, $a_{l-1} = [a_{t-1}, a_t]$, and $o_{l-1} = o_{t-1}$. This enables the modeling of dynamics at any temporal control frequency by chaining together actions, and allows high-level abstract action descriptions (e.g., "move left") and low-level motor controls (e.g., $\Delta x, \Delta\omega$) to be jointly modeled within the same framework. Temporally extended actions have been found to be beneficial in various settings such as learning hierarchical policies [38, 39], skills, and options [40, 41]. Rolling out a policy π in $\mathcal{M}$ corresponds to generating the next video segment conditioned on the (noisy) previously generated video and a new action input. As a result, UniSim can simulate arbitrarily long interactions by conditionally generating each video segment autoregressively. Note that while UniSim only models $\mathcal{T}$, reward signals $\mathcal{R}$ can be extracted from the generated videos for optimizing π, as we illustrate in Section 4.2 below.

Parametrizing and training UniSim. To instantiate the UniSim method outlined in Figure 2, we use diffusion models [42, 33] to parametrize $\overline{\mathcal{T}}(o_t|o_{t-1}, a_{t-1})$. Specifically, the reverse process learns a denoising model $\epsilon_\theta(o_t^{(k)}, k|o_{t-1}, a_{t-1})$ that denoises a previous observation into the next observation using K denoising steps. We concatenate the last four frames from o_{t-1} with initial noise samples $o_t^{(K)} \sim \mathcal{N}(0, I)$ channelwise to serve as conditional inputs to the denoising model. To

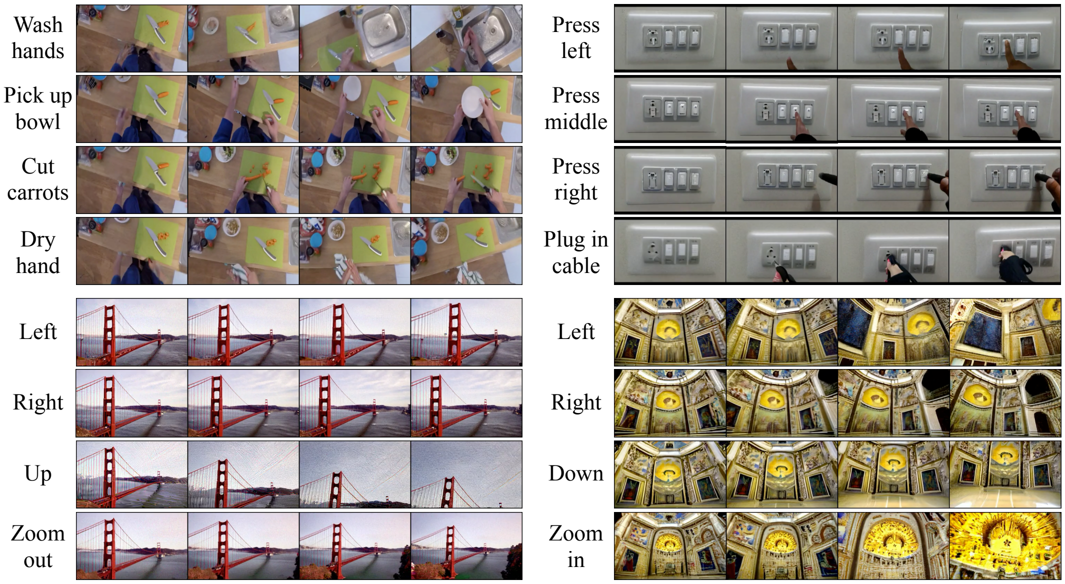

Figure 3: **Action-rich simulations.** UniSim can support manipulation actions such as "cut carrots", "wash hands", and "pickup bowl" from the same initial frame (top left) and other navigation actions.

condition on an action a_{t-1}, we leverage classifier-free guidance [43]. The final $\overline{\mathcal{T}}(o_t|o_{t-1}, a_{t-1})$ is parametrized by the variance schedule:

$$\epsilon_\theta(o_t^{(k)}, k|o_{t-1}, a_{t-1}) = (1+\eta)\epsilon_\theta(o_t^{(k)}, k|o_{t-1}, a_{t-1}) - \eta\epsilon_\theta(o_t, k|o_{t-1}), \tag{2}$$

where η controls action conditioning strength. With this parametrization, we train ϵ_θ by minimizing

$$\mathcal{L}_{\text{MSE}} = \left\| \epsilon - \epsilon_\theta\Big(\sqrt{1-\beta^{(k)}}o_t + \sqrt{\beta^{(k)}}\epsilon,\ k \Big| o_{t-1}, a_{t-1}\Big)\right\|^2,$$

where $\epsilon \sim \mathcal{N}(0, I)$, and $\beta^{(k)} \in \mathbb{R}$ are a set of K different noise levels for each $k \in [1, K]$. Given the learned ϵ_θ, an observation o_t can be generated by sampling from the initial distribution $o_t^{(K)} \sim \mathcal{N}(0, I)$ and iteratively denoising according to the following process for k from K to 0

$$o_t^{(k-1)} = \alpha^{(k)}(o_t^{(k)} - \gamma^{(k)}\epsilon_\theta(o_t^{(k)}, k|o_{t-1}, a_{t-1})) + \xi, \quad \xi \sim \mathcal{N}\big(0, \sigma_k^2 I\big), \tag{3}$$

where $\gamma^{(k)}$ is the denoising step size, $\alpha^{(k)}$ is a linear decay on the current denoised sample, and σ_k is a time varying noise level that depends on $\alpha^{(k)}$ and $\beta^{(k)}$.

Training Policies using UniSim. With the approximated dynamics model $\overline{\mathcal{T}}$ parametrized by a denoising model ϵ_θ, we can then optimize policies using planning, search, or reinforcement learning algorithms by sampling from $\overline{\mathcal{T}}$. Using UniSim as an environment to train policies has a few advantages including unlimited environment access (through parallelizable video servers), real-world like observations (through photorealistic diffusion outputs), and flexible temporal control frequencies (through temporally extended actions across low-level robot controls and high-level text actions).

3 Simulating Real-World Interactions

We now demonstrate UniSim in emulating real-world manipulation and navigation environments by simulating both action-rich and long-horizon interactions for both humans and robots.

3.1 Action-Rich, Long-Horizon, and Diverse Interactions

Action-Rich Simulation. We first demonstrate action-rich interactions with UniSim through natural language actions. Figure 3 shows simulation of human manipulation and navigation starting from the same initial observation (left-most column). We can instruct a person in the initial frame to perform various kitchen tasks (top left), press different switches (top right), or navigate scenes (bottom). We note that the model only trained on generic internet data, without action-rich manipulation data such as EPIC-KITCHENS [28], fails to simulate action-rich manipulations (see Appendix 12).

Long-Horizon Simulation. Next, we illustrate 8 *sequential* interactions with UniSim in Figure 4. Specifically, we condition the simulation of each interaction on previous observation and new language action as described in Section 2.2. UniSim successfully preserves objects manipulated

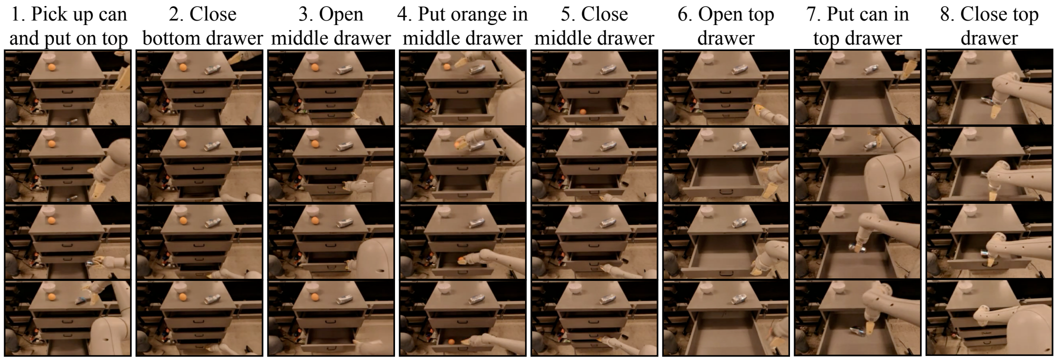

Figure 4: **Long-horizon simulations.** UniSim sequentially simulates 8 interactions autoregressively. The simulated interactions maintain temporal consistency across long-horizon interactions, correctly preserving objects and locations (can on counter in column 2-7, orange in drawer in column 4-5).

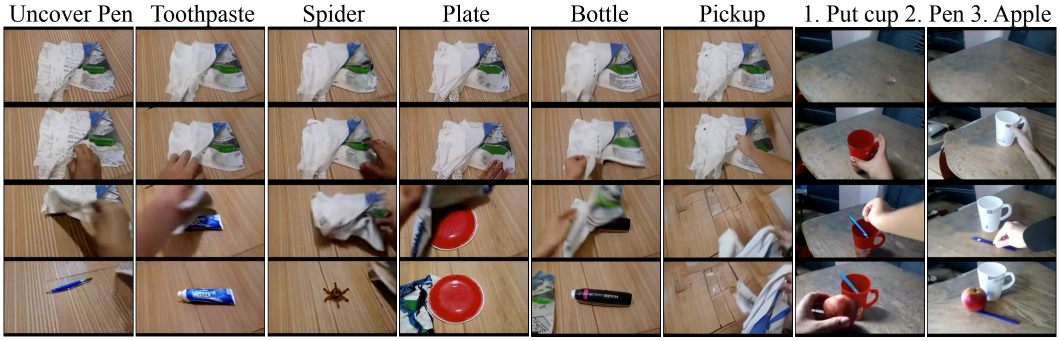

Figure 5: **Diverse and stochastic simulations.** On the left, we use text to specify the object being revealed by suffixing "uncovering" with the object name. On the right, we only specify "put cup" or "put pen", and cups and pens of different colors are sampled as a result of the stochastic sampling process during video generation.

Condition	FID ↓	FVD ↓	IS ↑	CLIP ↑
1 frame	59.47	315.69	3.03	22.55
4 distant	34.89	237	3.43	22.62
4 recent	**34.63**	**211.3**	**3.52**	**22.63**

Table 1: **Ablations of history conditioning** using FVD, FID, and Inception score, and CLIP score on Ego4D. Conditioning on multiple frames is better than on a single frame, and recent history has an edge over distant history.

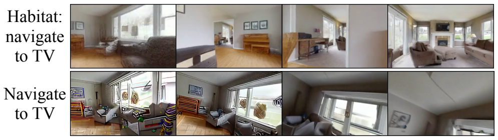

Figure 6: **Simulations of low-data domains** using the Habitat object navigation using HM3D dataset [23] with only 700 training examples. Prefixing language actions with dataset identifier leads to videos that completes the action (top).

by previous instructions (e.g., the orange and can are preserved in the drawers in Columns 4, 5, 7, 8 after being put in the drawers). Additional results on long-horizon interaction are included in Appendix 8.1.

Diversity and Stochasticity in UniSim In addition to supporting action-rich and long-horizon interactions, UniSim can also support highly diverse and stochastic environment transitions, such as diversity in objects being revealed after removing the towel on top (Figure 5 left), diversity in object colors and locations (cups and pens in Figure 5 right), and real-world variabilities such as wind and change in camera angles. We can use language actions to specify the appearance of diverse objects, and leverage the stochastic sampling process of video generation to support environment stochasticity such as wind and camera angles. Since diffusion models are flexible in capturing multi-modal distributions, they can generate diverse samples representing highly stochastic environments. Note that stochasticity associated with the sampling process cannot be directly controlled by users, which corresponds to the uncontrollable environment stochasticity from Dichotomy of Control [44] that is ubiquitous in real-world environments (e.g., wind).

3.2 Ablation and Analysis

Frame Conditioning Ablations. We ablate over choices of past frames to condition on (detailed in Section 2.2) using the Ego4D dataset [11], which contains egocentric movement requiring proper handling of observation history. We compare UniSim trained by conditioning on a different number of past frames and report generative modeling metrics (e.g., FID, FVD) in Table 1. We observe that conditioning on more frames from the past is better than conditioning on a single frame, but conditioning on history that is too far in the past (4 frames with exponentially increasing distances) can hurt performance. We found increasing the number of conditioning frames beyond 4 did not further improve performance on Ego4D, but it could be helpful for applications that require memory from distant past (e.g., navigation for retrieval).

Simulating Low-Data Domains. During joint training of UniSim on diverse data, we found that naïvely combining datasets of highly varying size can result in low generation quality in low-data domains. While we can increase the weight of these domains in the data mixture during training, we found that attaching a domain identifier such as the name of the dataset to the actions being conditioned on improves generation quality in low-data domains, as shown in Figure 6.

4 Applications of UniSim

Having learned a realistic simulator of the real world, we now demonstrate how UniSim can be used to train other types of machine intelligence such as embodied planners, reinforcement learning agents, and vision-language models through simulating highly realistic experiences.

4.1 Training Long-Horizon Embodied Planner through Hindsight Labeling.

One of the recent advances in learning embodied agents has been the adoption of language models or vision language models (VLM) as policies or planners that can operate in image or text based observation and action spaces [45, 46, 47]. One major challenge in learning such agents lies in the need for large amounts of data from the real world. The labor intensity in data collection only increases as tasks increase in horizon and complexity. Below, we demonstrate how UniSim can generate large amounts of training data for VLM policies through hindsight relabeling.

Setup and Baseline. We use data from the Language Table environment [22] for learning geometric rearrangements of blocks on a table. The dataset consists of 160k simulated trajectories and 440k real trajectories where each trajectory contains a language instruction (e.g., "move blue cube to the right"), a sequence of visuomotor controls, and a sequence of image frames corresponding to the execution of the task. The original trajectories have short horizons (e.g., only moving one block). We train an image-goal conditioned VLM policy to predict language instructions and the motor controls from the start and goal images using the PALM-E architecture [46] (See details in Appendix 11.1). For the baseline, the goal is set to the last frame of the original short-horizon trajectories. During each evaluation run, we set the goal by modifying the location of 3-4 blocks in the Language Table Simulation environment, and measure the blocks' distance to their goal states after executing 5 instructions using the VLM policy. We define the reduction in distance to goal (RDG) metric as

$$\text{RDG} = \frac{\|s_0 - s_{\text{goal}}\|_2 - \|s_T - s_{\text{goal}}\|_2}{\|s_0 - s_{\text{goal}}\|_2}, \tag{4}$$

where s_T represents the underlying block locations after executing the policy, s_0 and s_{goal} represents the initial and goal block locations.

Generating Hindsight Data with UniSim. To use UniSim for long-horizon tasks, we draw inspiration from hindsight relabeling [48]. Specifically, we create a total of 10k long-horizon trajectories from UniSim by doing rollouts in UniSim 3-5 times per trajectory, where each rollout corresponds to one scripted language instruction similar to the original dataset. We then use the final frame from each long-horizon rollout as a goal input and the scripted language instructions as supervision for training the VLM policy.

Results on Zero-shot Real-World Transfer. The true value of UniSim lies in simulating the real world. Figure 7 shows that the language plans produced by the VLM, the generated videos from UniSim according to the language plans, and the executions on the real robot. The policy purely trained in UniSim can directly perform long-horizon tasks in the real world in a zero-shot manner. See additional sim-to-real results with zero-shot real-world transfer in Appendix 8.2.

Results on Simulated Evaluation. In addition to testing real-world transfer, we also conduct simulator based evaluation to compare the reduction in distance to goal (RDG) of the VLM policy

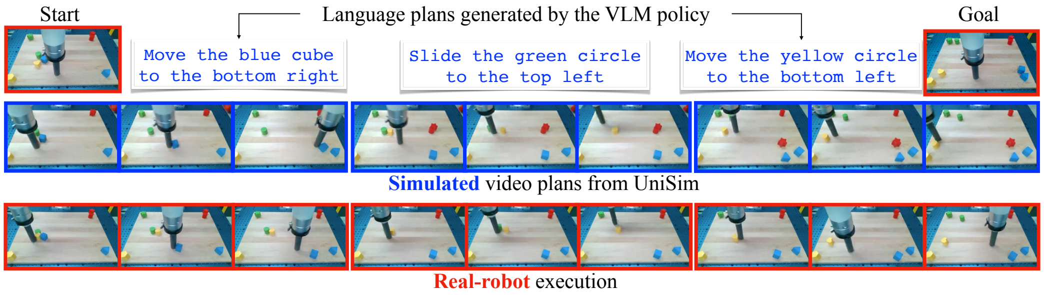
Start
Language plans generated by the VLM policy
Goal
Move the blue cube to the bottom right
Slide the green circle to the top left
Move the yellow circle to the bottom left
Simulated video plans from UniSim
Real-robot execution

Figure 7: **Long-horizon plans from UniSim.** A VLM poliy generates high-level language plans (first row) which are executed in UniSim (middle row) similar to how they are executed in the real world (bottom row) using the Language Table robot. The VLM trained on data from UniSim can plan for long-horizon tasks by successfully moving three blocks (blue, green, yellow) to match their target location in the goal image.

	RDG (moved)	RDG (all)
VLM-BC	0.11 ± 0.13	0.07 ± 0.11
UniSim-Hindsight	**0.34** ±0.13	**0.34** ± 0.13

Table 2: **Evaluation of long-horizon plans.** Reduction in distance to goal (RDG) defined in Equation 4 across 5 evaluation runs of VLM trained in UniSim simulated long-horizon data (bottom row) compared to VLM trained on original short-horizon data (top row). Using UniSim performs much better both in RGD of moved blocks (left) and RGD in all blocks (right).

	Succ. rate (all)	Succ. rate (pointing)
VLA-BC	0.58	0.12
UniSim-RL	**0.81**	**0.71**

Table 3: **Evaluation of RL policy.** Percentage of successful simulated rollouts (out of 48 tasks) using the VLA policy with and without RL finetuning on Language Table (assessed qualitatively using video rollouts in UniSim). UniSim-RL improves the overall performance, especially in pointing-based tasks which contain limited expert demonstrations.

trained on UniSim's generated long-horizon data to the VLM policy trained on the original short-horizon data in Table 2. The VLM trained using long-horizon data generated by UniSim performs 3-4 times better than the VLM trained on original short-horizon data in completing long-horizon goal-conditioned tasks.

4.2 Real-World Simulator for Reinforcement Learning

Reinforcement learning (RL) has achieved superhuman performance on difficult tasks such as playing Go and Atari games [34, 49], but has limited real world applications due to the lack of a realistic environment simulator [50]. We investigate whether UniSim can enable effective training of RL agents by providing the agent with a realistic simulator that can be accessed in parallel.

Setup. We finetune the PaLI 3B vision-language model [51] to predict low-level control actions (joint movements in $\Delta x, \Delta y$) from an image observation and a task description (e.g., "move the blue cube to the right") using the behavioral cloning (BC) loss to serve as the low-level control policy and the baseline, which we call the vision-language-action (VLA) policy similar to [47]. Because UniSim can take low-level control actions as input, we can directly conduct model-based rollouts in UniSim using control actions outputted by VLA policy. To acquire reward information, we use the number of steps-to-completion from the training data as a proxy reward to train a model that maps the current observation to learned reward. We then use the REINFORCE algorithm [52] to optimize the VLA policy, treating the rollouts from UniSim as the on-policy rollouts from the real environment and use the learned reward model to predict rewards from simulated rollouts. See details of RL training in Appendix 11.2.

Results. We first assess the quality of UniSim in simulating real-robot executions by applying low-level control actions (e.g., $\Delta x = 0.05, \delta y = 0.05$) repeatedly for 20-30 environment steps to move the endpoint left, right, down, up, and diagonally in Figure 8 (top two rows). We see that the simulated rollouts capture both the endpoint movements and the physics of collision. To compare the RL policy trained in UniSim to the BC policy, we qualitatively assessed the simulated rollouts in UniSim. Table 3 shows that RL training significantly improves the performance of the VLA policy across a wide set of tasks, especially in tasks such as "point to blue block". We then directly deploy the RL policy trained in UniSim onto the real robot in zero-shot, and observe successful task executions as shown in Figure 8 (bottom row). Additional results on zero-shot transfer to real robot can be found in Appendix 8.3.

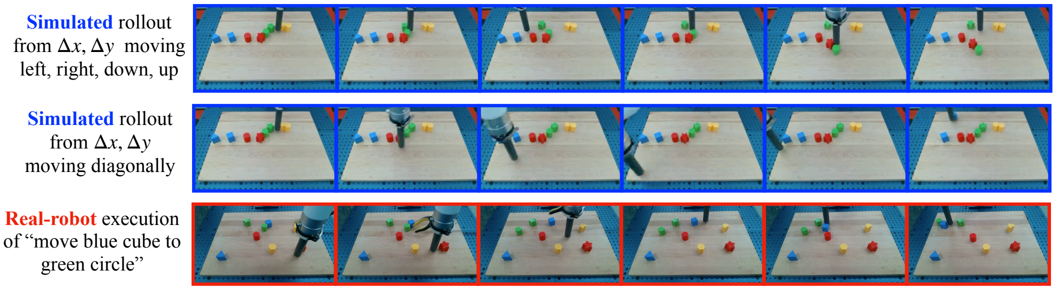
Simulated rollout from Δx, Δy moving left, right, down, up
Simulated rollout from Δx, Δy moving diagonally
Real-robot execution of "move blue cube to green circle"

Figure 8: **[Top] Simulation from low-level controls**. UniSim supports low-level control actions as inputs to move endpoint horizontally, vertically, and diagonally. **[Bottom] Real-robot execution of an RL policy** trained in UniSim and zero-shot onto the real Language Table task. The RL policy can successfully complete the task of "moving blue cube to green circle".

4.3 Realistic Simulator for Broader Vision-Language Tasks.

UniSim can generate training data for other machine intelligence such as detectors of rare events. This is especially useful when natural data is rare or difficult to collect (e.g., footage of crimes or accidents). We provide such a proof-of-concept by training vision-language models on purely generated data from UniSim, and observe significant performance benefits in video captioning tasks.

Setup. We finetune PaLI-X [53], a VLM with 55B parameters pretrained on a broad set of image, video, and language tasks, to caption a set of videos generated by UniSim using texts from the training split of ActivityNet Captions [9]. We measure the CIDEr score of the finetuned model on the test split of ActivityNet Captions as well as other captioning tasks following the same setup as [53]. See finetuning details of PaLI-X in Appendix 11.3.

	Activity	MSR-VTT	VATEX	SMIT
No finetune	15.2	21.91	13.31	9.22
Activity	54.90	24.88	36.01	16.91
UniSim	46.23	**27.63**	**40.03**	**20.58**

Table 4: **VLM trained in UniSim** to perform video captioning tasks. CIDEr scores for PaLI-X finetuned only on simulated data from UniSim compared to no finetuning and finetuning on true video data from ActivityNet Captions. Finetuning only on simulated data has a large advantage over no finetuning and transfers better to other tasks than finetuning on true data.

Results. We compare PaLI-X finetuned on purely generated videos to PaLI-X without finetuning and PaLI-X finetuned on original ActivityNet Captions in Table 4. Purely finetuning on generated data drastically improves the captioning performance from no finetuning at all on ActivityNet (15.2 to 46.23), while achieving 84% performance of finetuning on true data. Furthermore, PaLI-X finetuned on generated data transfers better to other captioning tasks such as MSR-VTT [8], VATEX [54], and SMIT [55] than PaLI-X finetuned on true data, which tends to overfit to ActivityNet. These results suggest that UniSim can serve as an effective data generator for improving broader vision-language models.

5 Related Work

Internet-Scale Generative Models. Language models trained on internet text succeed at text-based tasks [1, 56] but not physical tasks, which requires perception and control. Internet-scale generative models can synthesize realistic images and videos [57, 4, 58, 59], but have mostly been applied to generative media [60] as opposed to empowering sophisticated agents capable of multi-turn interactions. [61] shows video generation can serve as policies, but the major bottleneck for policy learning often lies in limited access to real-world environments [50]. We focus on this exact bottleneck by learning universal simulators of the real world, enabling realistic and unlimited "environment" access for training sophisticated agents interactively.

Learning World Models. Learning an accurate world model in reaction to control inputs has been a long-standing challenge in model-based planning, optimization, and reinforcement learning [17, 62, 36]. Most systems choose to learn dynamics models in lower dimensional state spaces as opposed to in the pixel space [63, 64, 65, 66], which limits knowledge sharing across systems. With large transformer architectures, learning image-based world models became plausible [67, 68, 69, 70, 71, 72], but mostly in games or simulated domains with visually simplistic and abundant data. In video generation, previous works have leveraged text prompts [73, 74], driving videos [75, 76], 3D geometries [77, 78], physical simulations [79], frequency information [80], and user annotations [81] to introduce movements into videos. However, they focus on generating domain specific videos as opposed to building a universal simulator that can further improve other agents as in UniSim.

6 Limitations and Conclusion

We have shown it is possible to learn a universal simulator of the real world in response to various action inputs ranging from texts to robot controls. UniSim can simulate highly realistic experiences for interacting with humans and training autonomous agents. UniSim requires large compute to train similar to other modern foundation models. Despite this disadvantage, we hope UniSim will instigate broad interest in learning and applying real-world simulators to improve machine intelligence.

7 Acknowledgements

We thank Hanjun Dai and Doina Precup for reviewing this manuscript. We gratefully acknowledges the support of a Canada CIFAR AI Chair, NSERC and Amii, NSF GRFP, and the support from Berkeley BAIR industrial consortion.

References

[1] OpenAI. Gpt-4 technical report, 2023. 1, 9

[2] Zalán Borsos, Raphaël Marinier, Damien Vincent, Eugene Kharitonov, Olivier Pietquin, Matt Sharifi, Dominik Roblek, Olivier Teboul, David Grangier, Marco Tagliasacchi, et al. Audiolm: a language modeling approach to audio generation. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 2023. 1

[3] Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. Hierarchical text-conditional image generation with clip latents. *arXiv preprint arXiv:2204.06125*, 2022. 1

[4] Jonathan Ho, William Chan, Chitwan Saharia, Jay Whang, Ruiqi Gao, Alexey Gritsenko, Diederik P Kingma, Ben Poole, Mohammad Norouzi, David J Fleet, et al. Imagen video: High definition video generation with diffusion models. *arXiv preprint arXiv:2210.02303*, 2022. 1, 9, 22

[5] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V 13*, pages 740–755. Springer, 2014. 1

[6] Christoph Schuhmann, Romain Beaumont, Richard Vencu, Cade Gordon, Ross Wightman, Mehdi Cherti, Theo Coombes, Aarush Katta, Clayton Mullis, Mitchell Wortsman, et al. Laion-5b: An open large-scale dataset for training next generation image-text models. *Advances in Neural Information Processing Systems*, 35:25278–25294, 2022. 1

[7] Xiaohua Zhai, Alexander Kolesnikov, Neil Houlsby, and Lucas Beyer. Scaling vision transformers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12104–12113, 2022. 1, 23

[8] Jun Xu, Tao Mei, Ting Yao, and Yong Rui. Msr-vtt: A large video description dataset for bridging video and language. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5288–5296, 2016. 1, 9

[9] Ranjay Krishna, Kenji Hata, Frederic Ren, Li Fei-Fei, and Juan Carlos Niebles. Dense-captioning events in videos. In *Proceedings of the IEEE international conference on computer vision*, pages 706–715, 2017. 1, 9, 24

[10] Antoine Miech, Dimitri Zhukov, Jean-Baptiste Alayrac, Makarand Tapaswi, Ivan Laptev, and Josef Sivic. Howto100m: Learning a text-video embedding by watching hundred million narrated video clips. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 2630–2640, 2019. 1

[11] Kristen Grauman, Andrew Westbury, Eugene Byrne, Zachary Chavis, Antonino Furnari, Rohit Girdhar, Jackson Hamburger, Hao Jiang, Miao Liu, Xingyu Liu, et al. Ego4d: Around the world in 3,000 hours of egocentric video. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18995–19012, 2022. 1, 3, 7, 21

[12] Sudeep Dasari, Frederik Ebert, Stephen Tian, Suraj Nair, Bernadette Bucher, Karl Schmeckpeper, Siddharth Singh, Sergey Levine, and Chelsea Finn. Robonet: Large-scale multi-robot learning. *arXiv preprint arXiv:1910.11215*, 2019. 1

[13] Ajay Mandlekar, Yuke Zhu, Animesh Garg, Jonathan Booher, Max Spero, Albert Tung, Julian Gao, John Emmons, Anchit Gupta, Emre Orbay, et al. Roboturk: A crowdsourcing platform for robotic skill learning through imitation. In *Conference on Robot Learning*, pages 879–893. PMLR, 2018. 1

[14] George E Monahan. State of the art—a survey of partially observable markov decision processes: theory, models, and algorithms. *Management science*, 28(1):1–16, 1982. 2, 4

[15] Dimitri P. Bertsekas. *Dynamic Programming: Deterministic and Stochastic Models*. Prentice-Hall, Inc., One Lake Street, Upper Saddle River, NJ, United States, January 1987. 2

[16] Marcin Andrychowicz, Filip Wolski, Alex Ray, Jonas Schneider, Rachel Fong, Peter Welinder, Bob McGrew, Josh Tobin, Pieter Abbeel, and Wojciech Zaremba. Hindsight experience replay. In *Advances in Neural Information Processing Systems*. 2017. 2

[17] Richard S. Sutton. Learning to predict by the methods of temporal differences. *Machine Learning*, 1988. 2, 9

[18] Andrei A Rusu, Matej Večerík, Thomas Rothörl, Nicolas Heess, Razvan Pascanu, and Raia Hadsell. Sim-to-real robot learning from pixels with progressive nets. In *Conference on robot learning*, pages 262–270. PMLR, 2017. 2

[19] Wenshuai Zhao, Jorge Peña Queralta, and Tomi Westerlund. Sim-to-real transfer in deep reinforcement learning for robotics: a survey. In *2020 IEEE symposium series on computational intelligence (SSCI)*, pages 737–744. IEEE, 2020. 2

[20] Erica Salvato, Gianfranco Fenu, Eric Medvet, and Felice Andrea Pellegrino. Crossing the reality gap: A survey on sim-to-real transferability of robot controllers in reinforcement learning. *IEEE Access*, 9:153171–153187, 2021. 2

[21] Manolis Savva, Abhishek Kadian, Oleksandr Maksymets, Yili Zhao, Erik Wijmans, Bhavana Jain, Julian Straub, Jia Liu, Vladlen Koltun, Jitendra Malik, et al. Habitat: A platform for embodied ai research. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 9339–9347, 2019. 3

[22] Corey Lynch and Pierre Sermanet. Language conditioned imitation learning over unstructured data. *arXiv preprint arXiv:2005.07648*, 2020. 3, 7, 21

[23] Santhosh Kumar Ramakrishnan, Aaron Gokaslan, Erik Wijmans, Oleksandr Maksymets, Alexander Clegg, John M Turner, Eric Undersander, Wojciech Galuba, Andrew Westbury, Angel X Chang, Manolis Savva, Yili Zhao, and Dhruv Batra. Habitat-matterport 3d dataset (HM3d): 1000 large-scale 3d environments for embodied AI. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2021. 3, 6, 21

[24] Corey Lynch, Ayzaan Wahid, Jonathan Tompson, Tianli Ding, James Betker, Robert Baruch, Travis Armstrong, and Pete Florence. Interactive language: Talking to robots in real time. *IEEE Robotics and Automation Letters*, 2023. 3

[25] Frederik Ebert, Yanlai Yang, Karl Schmeckpeper, Bernadette Bucher, Georgios Georgakis, Kostas Daniilidis, Chelsea Finn, and Sergey Levine. Bridge data: Boosting generalization of robotic skills with cross-domain datasets. *arXiv preprint arXiv:2109.13396*, 2021. 3, 21

[26] Homer Walke, Kevin Black, Abraham Lee, Moo Jin Kim, Max Du, Chongyi Zheng, Tony Zhao, Philippe Hansen-Estruch, Quan Vuong, Andre He, et al. Bridgedata v2: A dataset for robot learning at scale. *arXiv preprint arXiv:2308.12952*, 2023. 3

[27] Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Joseph Dabis, Chelsea Finn, Keerthana Gopalakrishnan, Karol Hausman, Alex Herzog, Jasmine Hsu, et al. Rt-1: Robotics transformer for real-world control at scale. *arXiv preprint arXiv:2212.06817*, 2022. 3, 21, 26

[28] Dima Damen, Hazel Doughty, Giovanni Maria Farinella, Sanja Fidler, Antonino Furnari, Evangelos Kazakos, Davide Moltisanti, Jonathan Munro, Toby Perrett, Will Price, et al. Scaling egocentric vision: The epic-kitchens dataset. In *Proceedings of the European conference on computer vision (ECCV)*, pages 720–736, 2018. 3, 5, 21

[29] Raghav Goyal, Samira Ebrahimi Kahou, Vincent Michalski, Joanna Materzynska, Susanne Westphal, Heuna Kim, Valentin Haenel, Ingo Fruend, Peter Yianilos, Moritz Mueller-Freitag, et al. The" something something" video database for learning and evaluating visual common sense. In *Proceedings of the IEEE international conference on computer vision*, pages 5842–5850, 2017. 3, 4, 21

[30] Angel Chang, Angela Dai, Thomas Funkhouser, Maciej Halber, Matthias Niessner, Manolis Savva, Shuran Song, Andy Zeng, and Yinda Zhang. Matterport3d: Learning from rgb-d data in indoor environments. *arXiv preprint arXiv:1709.06158*, 2017. 4

[31] Christoph Schuhmann, Richard Vencu, Romain Beaumont, Robert Kaczmarczyk, Clayton Mullis, Aarush Katta, Theo Coombes, Jenia Jitsev, and Aran Komatsuzaki. Laion-400m: Open dataset of clip-filtered 400 million image-text pairs. *arXiv preprint arXiv:2111.02114*, 2021. 4, 21

[32] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, Peter J Liu, et al. Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.*, 21(140):1–67, 2020. 4, 21

[33] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *Advances in Neural Information Processing Systems*, 2020. 4

[34] David Silver, Thomas Hubert, Julian Schrittwieser, Ioannis Antonoglou, Matthew Lai, Arthur Guez, Marc Lanctot, Laurent Sifre, Dharshan Kumaran, Thore Graepel, Timothy P. Lillicrap, Karen Simonyan, and Demis Hassabis. Mastering chess and shogi by self-play with a general reinforcement learning algorithm. *arXiv preprint arXiv:1712.01815*. 4, 8

[35] Michael Montemerlo and Sebastian Thrun. *FastSLAM: A scalable method for the simultaneous localization and mapping problem in robotics*, volume 27. Springer, 2007. 4

[36] Shengwei Wang and Xinqiao Jin. Model-based optimal control of vav air-conditioning system using genetic algorithm. *Building and Environment*, 35(6):471–487, 2000. 4, 9

[37] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*, 2013. 4

[38] Leslie Pack Kaelbling and Tomás Lozano-Pérez. Hierarchical task and motion planning in the now. In *2011 IEEE International Conference on Robotics and Automation*, pages 1470–1477. IEEE, 2011. 4

[39] Ofir Nachum, Shixiang Shane Gu, Honglak Lee, and Sergey Levine. Data-efficient hierarchical reinforcement learning. *Advances in neural information processing systems*, 31, 2018. 4

[40] Richard S Sutton, Doina Precup, and Satinder Singh. Between mdps and semi-mdps: A framework for temporal abstraction in reinforcement learning. *Artificial intelligence*, 112(1-2):181–211, 1999. 4

[41] George Konidaris and Andrew Barto. Skill discovery in continuous reinforcement learning domains using skill chaining. *Advances in neural information processing systems*, 22, 2009. 4

[42] Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *International Conference on Machine Learning*, pages 2256–2265. PMLR, 2015. 4

[43] Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. *arXiv preprint arXiv:2207.12598*, 2022. 5

[44] Mengjiao Yang, Dale Schuurmans, Pieter Abbeel, and Ofir Nachum. Dichotomy of control: Separating what you can control from what you cannot. *arXiv preprint arXiv:2210.13435*, 2022. 6

[45] Yuqing Du, Olivia Watkins, Zihan Wang, Cédric Colas, Trevor Darrell, Pieter Abbeel, Abhishek Gupta, and Jacob Andreas. Guiding pretraining in reinforcement learning with large language models. *arXiv preprint arXiv:2302.06692*, 2023. 7

[46] Danny Driess, Fei Xia, Mehdi SM Sajjadi, Corey Lynch, Aakanksha Chowdhery, Brian Ichter, Ayzaan Wahid, Jonathan Tompson, Quan Vuong, Tianhe Yu, et al. Palm-e: An embodied multimodal language model. *arXiv preprint arXiv:2303.03378*, 2023. 7, 23

[47] Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Xi Chen, Krzysztof Choromanski, Tianli Ding, Danny Driess, Avinava Dubey, Chelsea Finn, et al. Rt-2: Vision-language-action models transfer web knowledge to robotic control. *arXiv preprint arXiv:2307.15818*, 2023. 7, 8, 23

[48] Paulo Rauber, Avinash Ummadisingu, Filipe Mutz, and Jürgen Schmidhuber. Hindsight policy gradients. In *International Conference on Learning Representations*, 2019. 7

[49] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *nature*, 518(7540):529–533, 2015. 8

[50] Gabriel Dulac-Arnold, Daniel Mankowitz, and Todd Hester. Challenges of real-world reinforcement learning. *arXiv preprint arXiv:1904.12901*, 2019. 8, 9

[51] Xi Chen, Xiao Wang, Soravit Changpinyo, AJ Piergiovanni, Piotr Padlewski, Daniel Salz, Sebastian Goodman, Adam Grycner, Basil Mustafa, Lucas Beyer, et al. Pali: A jointly-scaled multilingual language-image model. *arXiv preprint arXiv:2209.06794*, 2022. 8, 23

[52] Ronald J Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8:229–256, 1992. 8, 24

[53] Xi Chen, Josip Djolonga, Piotr Padlewski, Basil Mustafa, Soravit Changpinyo, Jialin Wu, Carlos Riquelme Ruiz, Sebastian Goodman, Xiao Wang, Yi Tay, et al. Pali-x: On scaling up a multilingual vision and language model. *arXiv preprint arXiv:2305.18565*, 2023. 9, 24

[54] Xin Wang, Jiawei Wu, Junkun Chen, Lei Li, Yuan-Fang Wang, and William Yang Wang. Vatex: A large-scale, high-quality multilingual dataset for video-and-language research. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4581–4591, 2019. 9

[55] Mathew Monfort, SouYoung Jin, Alexander Liu, David Harwath, Rogerio Feris, James Glass, and Aude Oliva. Spoken moments: Learning joint audio-visual representations from video descriptions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14871–14881, 2021. 9

[56] Rohan Anil, Andrew M Dai, Orhan Firat, Melvin Johnson, Dmitry Lepikhin, Alexandre Passos, Siamak Shakeri, Emanuel Taropa, Paige Bailey, Zhifeng Chen, et al. Palm 2 technical report. *arXiv preprint arXiv:2305.10403*, 2023. 9

[57] Chenfei Wu, Lun Huang, Qianxi Zhang, Binyang Li, Lei Ji, Fan Yang, Guillermo Sapiro, and Nan Duan. Godiva: Generating open-domain videos from natural descriptions. *arXiv preprint arXiv:2104.14806*, 2021. 9

[58] Uriel Singer, Adam Polyak, Thomas Hayes, Xi Yin, Jie An, Songyang Zhang, Qiyuan Hu, Harry Yang, Oron Ashual, Oran Gafni, et al. Make-a-video: Text-to-video generation without text-video data. *arXiv preprint arXiv:2209.14792*, 2022. 9

[59] Mengjiao Yang, Yilun Du, Bo Dai, Dale Schuurmans, Joshua B Tenenbaum, and Pieter Abbeel. Probabilistic adaptation of text-to-video models. *arXiv preprint arXiv:2306.01872*, 2023. 9

[60] Chenshuang Zhang, Chaoning Zhang, Mengchun Zhang, and In So Kweon. Text-to-image diffusion model in generative ai: A survey. *arXiv preprint arXiv:2303.07909*, 2023. 9

[61] Yilun Du, Mengjiao Yang, Bo Dai, Hanjun Dai, Ofir Nachum, Joshua B Tenenbaum, Dale Schuurmans, and Pieter Abbeel. Learning universal policies via text-guided video generation, 2023. *URL https://arxiv. org/abs/2302.00111*, 2023. 9, 23

[62] Leslie Pack Kaelbling, Michael L Littman, and Anthony R Cassandra. Planning and acting in partially observable stochastic domains. *Artificial intelligence*, 101(1-2):99–134, 1998. 9

[63] Norm Ferns, Prakash Panangaden, and Doina Precup. Metrics for finite markov decision processes. In *UAI*, volume 4, pages 162–169, 2004. 9

[64] Alessandro Achille and Stefano Soatto. A separation principle for control in the age of deep learning. *Annual Review of Control, Robotics, and Autonomous Systems*, 1:287–307, 2018. 9

[65] Timothée Lesort, Natalia Díaz-Rodríguez, Jean-Franois Goudou, and David Filliat. State representation learning for control: An overview. *Neural Networks*, 108:379–392, 2018. 9

[66] Pablo Samuel Castro. Scalable methods for computing state similarity in deterministic markov decision processes. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 10069–10076, 2020. 9

[67] Danijar Hafner, Timothy Lillicrap, Mohammad Norouzi, and Jimmy Ba. Mastering atari with discrete world models. *arXiv preprint arXiv:2010.02193*, 2020. 9

[68] Chang Chen, Yi-Fu Wu, Jaesik Yoon, and Sungjin Ahn. Transdreamer: Reinforcement learning with transformer world models. *arXiv preprint arXiv:2202.09481*, 2022. 9

[69] Younggyo Seo, Kimin Lee, Stephen L James, and Pieter Abbeel. Reinforcement learning with action-free pre-training from videos. In *International Conference on Machine Learning*, pages 19561–19579. PMLR, 2022. 9

[70] Vincent Micheli, Eloi Alonso, and François Fleuret. Transformers are sample efficient world models. *arXiv preprint arXiv:2209.00588*, 2022. 9

[71] Ziyi Wu, Nikita Dvornik, Klaus Greff, Thomas Kipf, and Animesh Garg. Slotformer: Unsupervised visual dynamics simulation with object-centric models. *arXiv preprint arXiv:2210.05861*, 2022. 9

[72] Danijar Hafner, Jurgis Pasukonis, Jimmy Ba, and Timothy Lillicrap. Mastering diverse domains through world models. *arXiv preprint arXiv:2301.04104*, 2023. 9

[73] Sihyun Yu, Kihyuk Sohn, Subin Kim, and Jinwoo Shin. Video probabilistic diffusion models in projected latent space. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18456–18466, 2023. 9

[74] Daquan Zhou, Weimin Wang, Hanshu Yan, Weiwei Lv, Yizhe Zhu, and Jiashi Feng. Magicvideo: Efficient video generation with latent diffusion models. *arXiv preprint arXiv:2211.11018*, 2022. 9

[75] Aliaksandr Siarohin, Stéphane Lathuilière, Sergey Tulyakov, Elisa Ricci, and Nicu Sebe. Animating arbitrary objects via deep motion transfer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2377–2386, 2019. 9

[76] Yaohui Wang, Di Yang, Francois Bremond, and Antitza Dantcheva. Latent image animator: Learning to animate images via latent space navigation. *arXiv preprint arXiv:2203.09043*, 2022. 9

[77] Chung-Yi Weng, Brian Curless, and Ira Kemelmacher-Shlizerman. Photo wake-up: 3d character animation from a single photo. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 5908–5917, 2019. 9

[78] Tianfan Xue, Jiajun Wu, Katherine L Bouman, and William T Freeman. Visual dynamics: Stochastic future generation via layered cross convolutional networks. *IEEE transactions on pattern analysis and machine intelligence*, 41(9):2236–2250, 2018. 9

[79] Yung-Yu Chuang, Dan B Goldman, Ke Colin Zheng, Brian Curless, David H Salesin, and Richard Szeliski. Animating pictures with stochastic motion textures. In *ACM SIGGRAPH 2005 Papers*, pages 853–860. 2005. 9

[80] Zhengqi Li, Richard Tucker, Noah Snavely, and Aleksander Holynski. Generative image dynamics, 2023. 9

[81] Zekun Hao, Xun Huang, and Serge Belongie. Controllable video generation with sparse trajectories. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7854–7863, 2018. 9

[82] Peter Anderson, Qi Wu, Damien Teney, Jake Bruce, Mark Johnson, Niko Sünderhauf, Ian Reid, Stephen Gould, and Anton Van Den Hengel. Vision-and-language navigation: Interpreting visually-grounded navigation instructions in real environments. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3674–3683, 2018. 21

[83] Chao Jia, Yinfei Yang, Ye Xia, Yi-Ting Chen, Zarana Parekh, Hieu Pham, Quoc Le, Yun-Hsuan Sung, Zhen Li, and Tom Duerig. Scaling up visual and vision-language representation learning with noisy text supervision. In *International conference on machine learning*, pages 4904–4916. PMLR, 2021. 21

[84] Özgün Çiçek, Ahmed Abdulkadir, Soeren S Lienkamp, Thomas Brox, and Olaf Ronneberger. 3d u-net: learning dense volumetric segmentation from sparse annotation. In *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2016: 19th International Conference, Athens, Greece, October 17-21, 2016, Proceedings, Part II 19*, pages 424–432. Springer, 2016. 22

[85] Jonathan Ho, Tim Salimans, Alexey Gritsenko, William Chan, Mohammad Norouzi, and David J Fleet. Video diffusion models. *arXiv preprint arXiv:2204.03458*, 2022. 22

[86] Yi Tay, Mostafa Dehghani, Vinh Q Tran, Xavier Garcia, Jason Wei, Xuezhi Wang, Hyung Won Chung, Dara Bahri, Tal Schuster, Steven Zheng, et al. Ul2: Unifying language learning paradigms. In *The Eleventh International Conference on Learning Representations*, 2022. 23

[87] Yuval Tassa, Yotam Doron, Alistair Muldal, Tom Erez, Yazhe Li, Diego de Las Casas, David Budden, Abbas Abdolmaleki, Josh Merel, Andrew Lefrancq, et al. Deepmind control suite. *arXiv preprint arXiv:1801.00690*, 2018. 23

[88] Teng Wang, Ruimao Zhang, Zhichao Lu, Feng Zheng, Ran Cheng, and Ping Luo. End-to-end dense video captioning with parallel decoding. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6847–6857, 2021. 24

Appendix

In this Appendix we provide additional qualitative results on long-horizon simulation of human and robot interactions (Section 8.1), zero-shot sim-to-real transfer results on long-horizon planning (Section 8.2), and zero-shot sim-to-real transfer of the RL policy (Section 8.3). We also provided details on the dataset used to train UniSim in Section 9, the model architecture and training details of UniSim in Section 10, and the details of the three experimental setups for applications of UniSim in Section 11. Finally, we provide failed examples when UniSim is not jointly trained on broad datasets (Section 12). Video demos can be found at anonymous-papers-submissions.github.io

8 Additional Results

8.1 Additional Long-Horizon Interaction

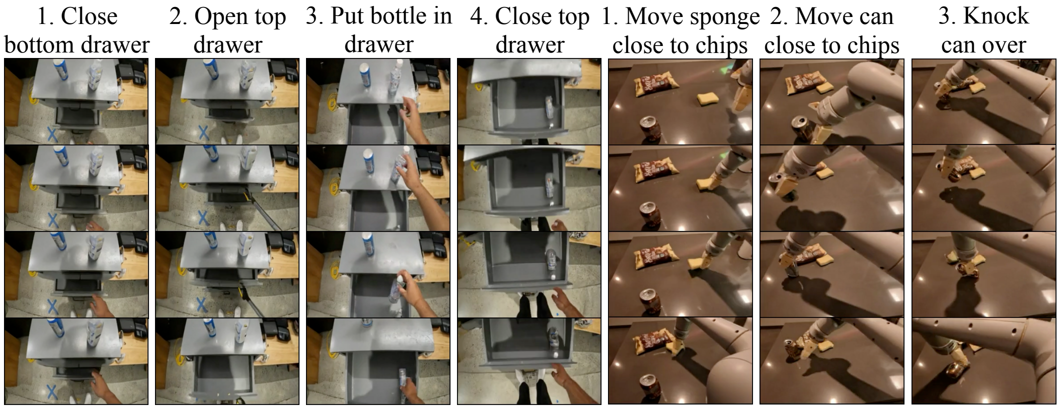

Figure 9: Additional results on long-horizon interaction with humans and robots similar to Figure 4. UniSim can generate consistent video rollouts across 3-4 high-level language actions.

8.2 Additional Real-Robot Results for Long-Horizon Planning

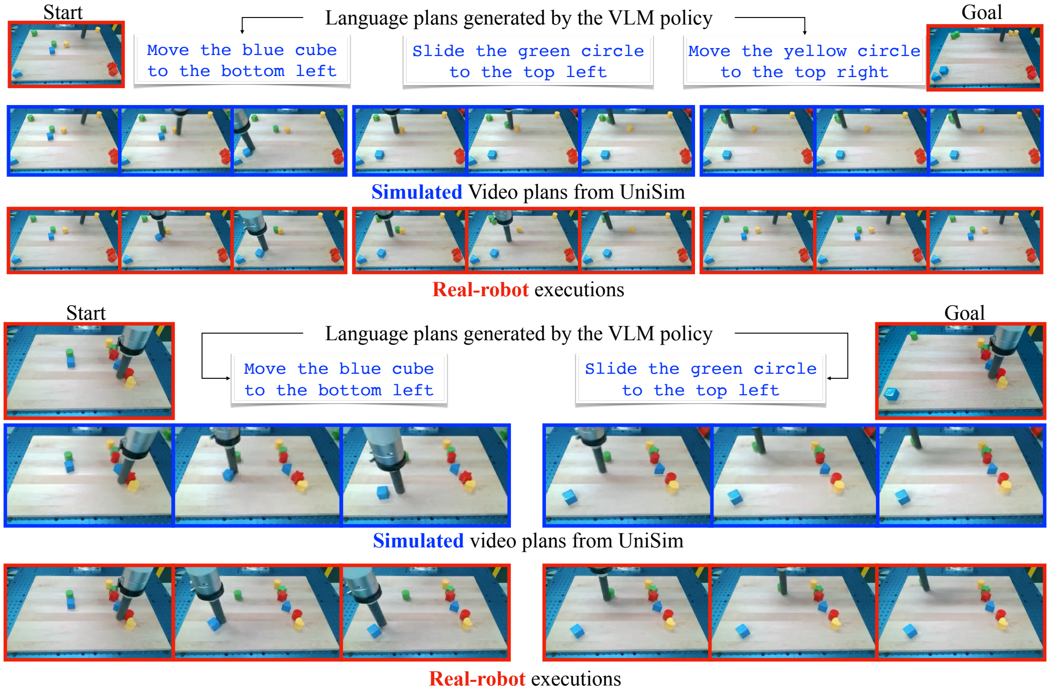
Start
Language plans generated by the VLM policy
Goal
Move the blue cube to the bottom left
Slide the green circle to the top left
Move the yellow circle to the top right
Simulated Video plans from UniSim
Real-robot executions
Start
Language plans generated by the VLM policy
Goal
Move the blue cube to the bottom left
Slide the green circle to the top left
Simulated video plans from UniSim
Real-robot executions

Figure 10: Additional results (similar to Figure 7) on applying UniSim to train vision-language planners to complete long-horizon tasks. VLM finetuned with hindsight labeled data is able to generate long-horizon instructions that moves two or three blocks successfully to match their location in the goal image.

8.3 Additional Results on Learning RL Policy in UniSim

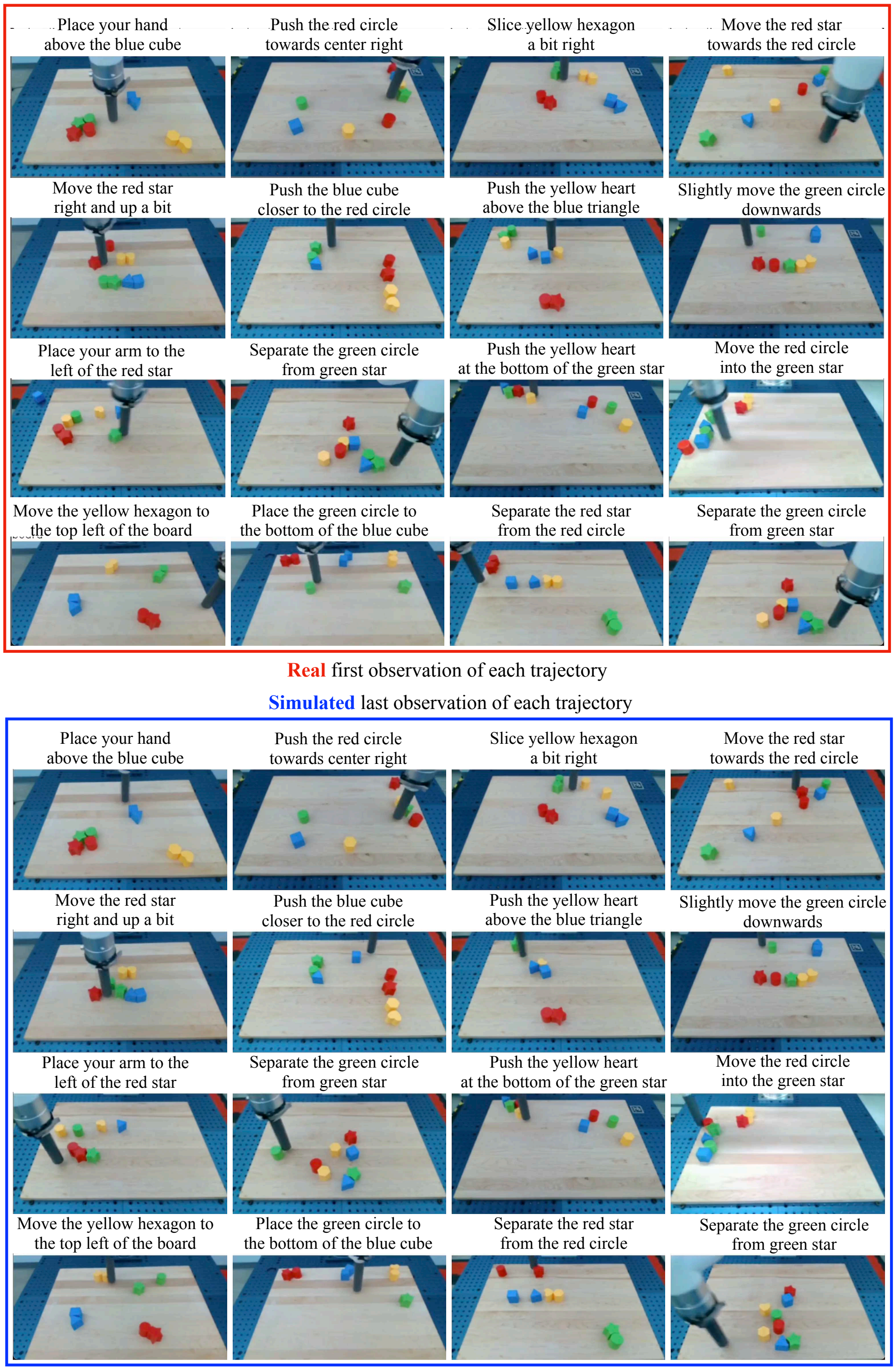
Place your hand above the blue cube
Push the red circle towards center right
Slice yellow hexagon a bit right
Move the red star towards the red circle
Move the red star right and up a bit
Push the blue cube closer to the red circle
Push the yellow heart above the blue triangle
Slightly move the green circle downwards
Place your arm to the left of the red star
Separate the green circle from green star
Push the yellow heart at the bottom of the green star
Move the red circle into the green star
Move the yellow hexagon to the top left of the board
Place the green circle to the bottom of the blue cube
Separate the red star from the red circle
Separate the green circle from green star
Real first observation of each trajectory
Simulated last observation of each trajectory
Place your hand above the blue cube
Push the red circle towards center right
Slice yellow hexagon a bit right
Move the red star towards the red circle
Move the red star right and up a bit
Push the blue cube closer to the red circle
Push the yellow heart above the blue triangle
Slightly move the green circle downwards
Place your arm to the left of the red star
Separate the green circle from green star
Push the yellow heart at the bottom of the green star
Move the red circle into the green star
Move the yellow hexagon to the top left of the board
Place the green circle to the bottom of the blue cube
Separate the red star from the red circle
Separate the green circle from green star

Figure 11: First real observations and last simulated observations of rolling out the RL policy trained in UniSim.

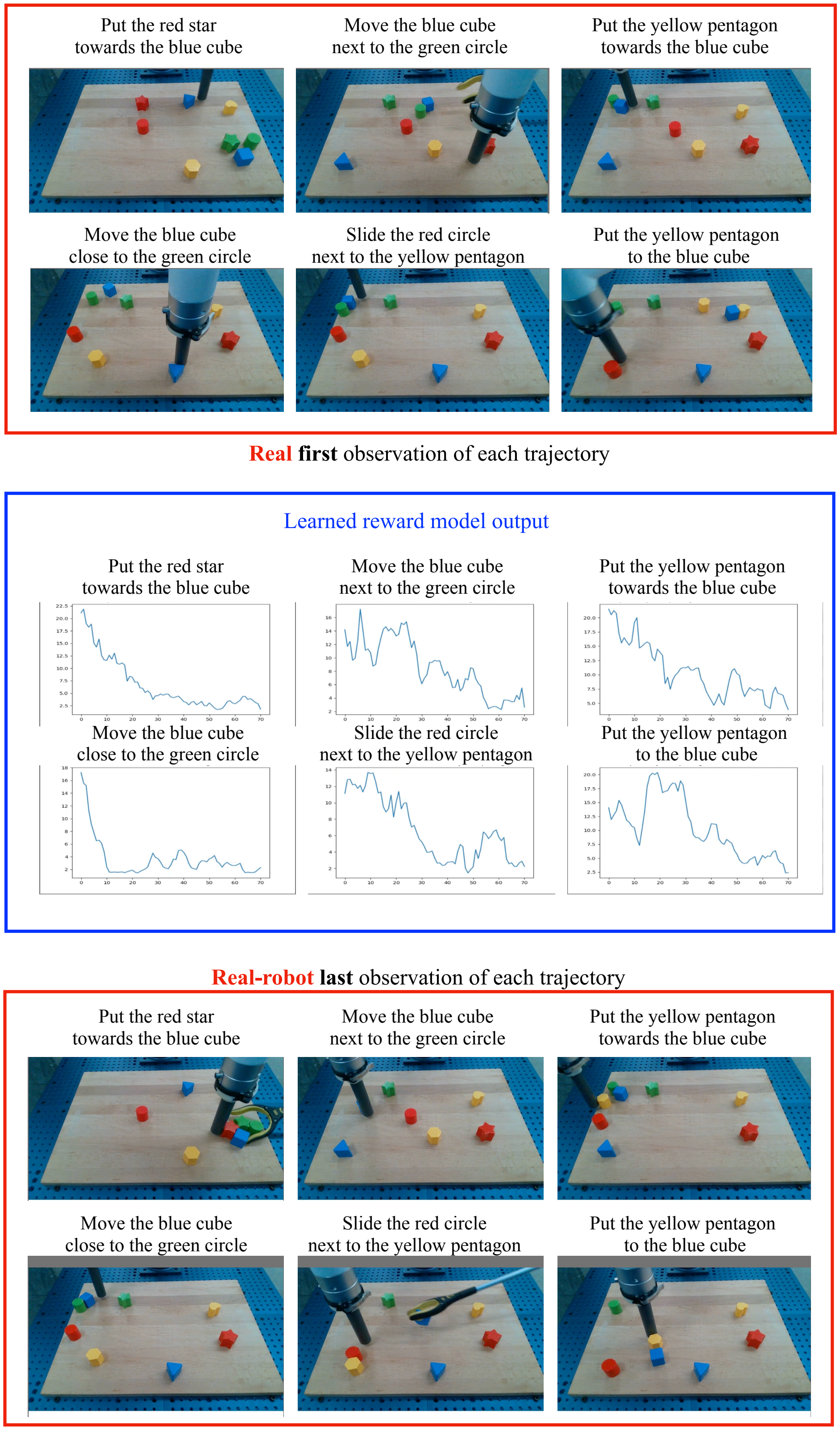

Figure 12: First real observations and last real observations of executing the RL policy trained from UniSim in the real world in zero-shot. Middle plot also shows the output of the learned reward model (steps-to-completion) during policy execution, where step 0 corresponds to the top plot (initial observation) and step 70 corresponds to the bottom plot (final observation).

9 Datasets

We provide the datasets used to train UniSim below, including dataset name, number of training examples (approximate), and weight in the data mixture. Miscellaneous data are collections of datasets that have not been published. Some of these datasets have been processed into train and validation split, hence the number of training examples may differ from the original data size. When text are available in the original dataset, we use T5 language model embeddings [32] to preprocess the text into continuous representations. When low-level controls are available in the original dataset, we encode them both as text and normalize then discretize them into 4096 bins contatenated with language embeddings (if present). The choice of mixture weights are either 0.1 or 0.05 without careful tuning. How data mixture weights affect simulation performance is an interesting line of future work.

	Dataset	**# Examples**	**Weight**
Simulation	Habitat HM3D [23]	710	0.1
	Language Table sim [22]	160k	0.05
Real Robot	Bridge Data [25]	2k	0.05
	RT-1 data [27]	70k	0.1
	Language Table real [22]	440k	0.05
	Miscellaneous robot videos	133k	0.05
Human activities	Ego4D [11]	3.5M	0.1
	Something-Something V2 [29]	160k	0.1
	EPIC-KITCHENS [28]	25k	0.1
	Miscellaneous human videos	50k	0.05
Panorama scan	Matterport Room-to-Room scans [82]	3.5M	0.1
Internet text-image	LAION-400M [31]	400M	0.05
	ALIGN [83]	400M	0.05
Internet video	Miscellaneous videos	13M	0.05

Table 5: Dataset name, number of training examples, and mixture weights used for training UniSim.

10 Architecture and Training

We the 3D U-Net architecture [84, 85] to parametrize the UniSim video model. We apply the spatial downsampling pass followed by the spatial upsampling pass with skip connections to the downsampling pass activations with interleaved 3D convolution and attention layers as in the standard 3D U-Net. The video models in UniSim consist of one history conditioned video prediction model as the base and two additional spatial super-resolution models similar to [4]. The history conditioned base model operates at temporal and spatial resolution $[16, 24, 40]$, and the two spatial super-resolution models operate at spatial resolution $[24, 40] \rightarrow [48, 80]$ and $[48, 80] \rightarrow [192, 320]$, respectively. To condition the base video model on the history, we take 4 frames from the previous video segment and concatenate them channelwise to the noise samples inputted to the U-Net. We employ temporal attention for the forward model to allow maximum modeling flexibility but temporal convolution to the super-resolution models for efficiency reasons similar to [4]. The model and training hyperparamters of UniSim are summarized in Table 6.

Hyperparameter	**Value**
Base channels	1024
Optimizer	Adam ($\beta_1 = 0.9, \beta_2 = 0.99$)
Channel multipliers	1, 2, 4
Learning rate	0.0001
Blocks per resolution	3
Batch size	256
Attention resolutions	6, 12, 24
Num attention heads	16, 16, 8
Conditioning embedding dimension	4096
Conditioning embedding MLP layers: 4	
Conditioning token length	64
EMA	0.9999
Dropout	0.1
Training hardware	512 TPU-v3 chips
Training steps	1000000
Diffusion noise schedule	cosine
Noise schedule log SNR range	[-20, 20]
Sampling timesteps	256
Sampling log-variance interpolation	$\gamma = 0.1$
Weight decay	0.0
Prediction target	ϵ

Table 6: Hyperparameters for training the UniSim diffusion model.

11 Details of Experimental Setups

11.1 Details of Long-Horizon Planning

PALM-E VLM Policy. We modify the original PALM-E 12B model [46] to condition on a goal image as additional input before decoding the text actions. The VLM is finetuned on either the original short horizon data or the long horizon simulated data using 64 TPUv3 chips for 1 day. The supervision for short-horizon baseline is the single step language instruction in the original data, whereas the supervision for long-horizon UniSim data is the scripted long-horizon language instructions chained together that generated the video data. Other model architecture and training details follow [46].

Simulated evaluation. In setting up goal in the simulated environments, a subset of 3-4 blocks (randomly selected) are moved by 0.05, 0.1, or 0.2 along the x,y axes (randomly selected). The original observation space has $x \in [0.15, 0.6]$ and $y \in [-0.3048, 0.3048]$. So the modification of goal location corresponds to meaningful block movements. For executing the long-horizon VLM policy trained on UniSim data, we first sample one language instruction from the VLM, predict a video of 16 frames, and use a separately trained inverse dynamics model similar to [61] to recover the low-level control actions, which we found to slightly outperform directly regressing on control actions from language outputs of the VLM. We execute 5 instructions in total, and measure the final distance to goal according to the ground truth simulator state. We 5 evaluations each with a different random seed for sampling the initial state and resetting the goal, and report the mean and standard error in Table 2.

11.2 Details of RL Policy Training

Stage 1 (Supervised Learning)

Model Architecture The PaLI 3B model trained on Language-Table uses a Vision Transformer architecture G/14 [7] to process images, and the encoder-decoder architecture of UL2 language model [86] for encoding task descriptions and decoding tokens which can represent language, control actions, or other values of interest (described below).

Objectives In the first stage of training, using a dataset of demonstrations, we finetune the pretrained PaLI 3B vision language model checkpoint [51] with the following tasks:

- **Behavioral Cloning:** Given observations and task instruction, predict the demonstration action. The continuous actions of the Language-Table domain are discretized into the form "+1 -5", and represented using extra tokens from the PaLI model's token vocabulary. As an example, "+1 -5" is represented by the token sequence (`<extra_id_65>, <extra_id_1>, <extra_id_66>, <extra_id_5>`).
- **Timestep to Success Prediction:** Given observations and task instruction, predict how many timesteps are left until the end of episode (i.e. success). Similar to actions, the number of steps remaining is represented via extra tokens from the PaLI model's token vocabulary.
- **Instruction Prediction:** Given the first and last frame of an episode, predict the task instruction associated with that episode.

We use learning rate 0.001, dropout rate 0.1, and batch size 128 to finetune the PaLI 3B model for 300k gradient steps with 1k warmup steps on both the simulated and real Language Table dataset similar to RT-2 [47].

Stage 2 (RL Training)

Reward Definition As mentioned above, during Stage 1, given an observation and goal, the PaLI model is finetuned to predict how many timesteps are left until the demonstration episode reaches a success state. Let us denote this function by $d(o, g)$. The reward we use during RL training is defined as $r(o_t, a_t, o_{t+1}, g) = -[d(o_{t+1}, g) - d(o_t, g)] \cdot \mathcal{C}$, where $\mathcal{C} > 0$ is a small constant used to stabilize training ($\mathcal{C} = 5e-2$ in this work). Intuitively, this reward tracks if from timestep t to $t+1$ the policy arrived closer to accomplishing the desired goal. Before starting Stage 2, we make a copy of the Stage 1 model checkpoint and keep it frozen to use as the reward model for RL training.

Environment Definition To implement video generation as environment transitions, we expose the inference interface of the video generation model through remote procedure call, and use the DeepMind RL Environment API (also known as DM Env API) [87] to wrap the remote procedure call in the step function of the environment. When the environment is reset to start a new episode, a goal instruction is randomly sampled from the ones available in the dataset of demonstrations used in Stage 1.

RL Method We initialize the RL trained policy using the Stage 1 checkpoint, which as mentioned was also trained with a Behavioral Cloning objective. A collection of actor processes perform policy rollouts in the video generation environment, and add rewards to the trajectories using the reward model defined above. The policy is updated using the REINFORCE [52] objective, i.e. $\nabla_\pi \mathcal{L}(o_t, a_t, g) = \nabla_\pi \log \pi(a_t|o_t, g) \cdot \left[\sum_{i=t}^{T} \gamma^{i-t} \cdot r(o_i, a_i, o_{i+1}, g)\right]$, where $\mathcal{L}(o_t, a_t, g)$ represents the loss associated with the observation-action pair (o_t, a_t) in an episode with the goal g. The actors are rate limited to prevent generated trajectories from being very off-policy. We report the hyperparameters associated with RL training in Table 7.

Hyperparameter	**Value**
Max steps per episode	100
Number of actor processes	64
Number of image history stack	2
Learner batch size	64
Discounting factor γ	0.9

Table 7: Hyperparameters for training the VLA RL policy using the ACME framework.

11.3 Details of Video Captioning

Note that even though UniSim is a video based simulator trained to condition on past history, we can achieve text-only conditioning by inputting placeholder frames such as white images while increasing the classifier-free guidance strength on text. We found this to work well in generating videos purely from captions of ActivityNet Captions. For generating data to train VLMs, we take the training split of ActivityNet Captions which consists of 30,740 text-video examples after the 50/25/25% train/val1/val2 split as in [53]. For each of the 30,740 text, we generate 4 videos from UniSim, and use the text labels as supervision in finetuning PaLI-X. As a result, we have 4X amount of the original training data (in terms the number of videos). In addition, we found the generated videos to generally align better semantically than the original ActivityNet Captions videos, which could contain noise and ambiguous videos that could be labeled differently. We use ground truth temporal proposals at evaluation following [53] and [9]. Following [53] and [88], we use the val1 split for validation and val2 split for testing.

12 Failed Simulations without Joint Training

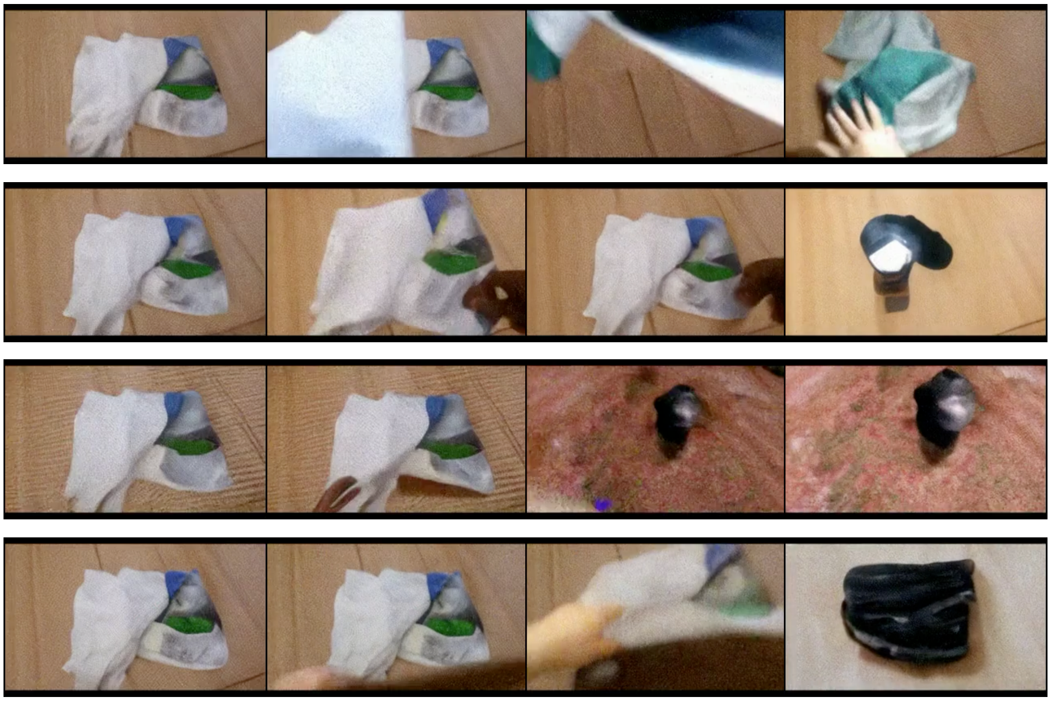

Figure 13: Failed environment simulation from the action “uncover bottle” without training on broad data as in UniSim. Top two videos are generated from only training on SSV2. Bottom two videos are generated from only training on generic internet data (without SSV2, EpicKitchen, Ego4D, and various robotics dataset).

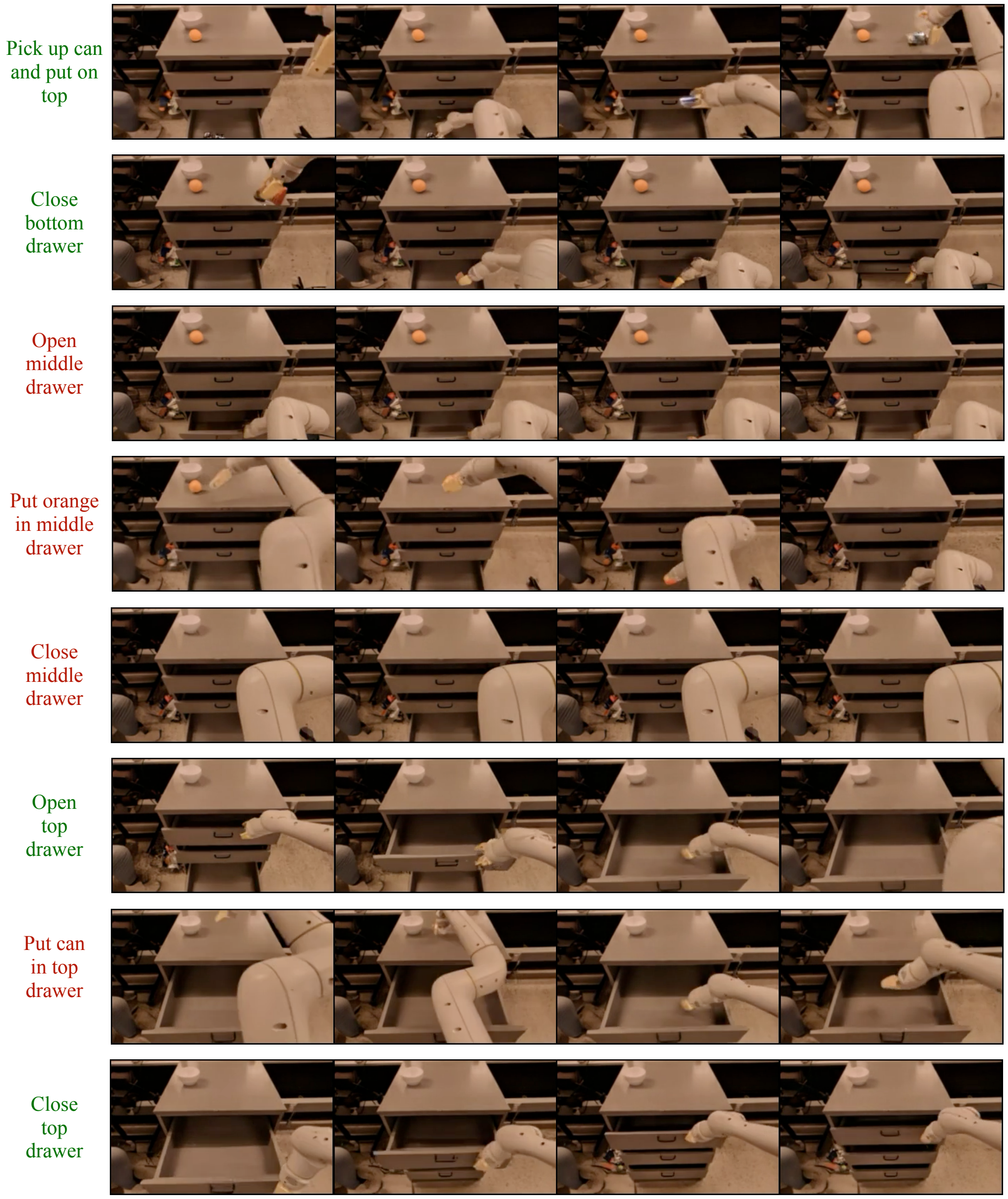

Figure 14: When the text-to-video model behind UniSim is only trained on data from [27] as opposed incorporating broad data from the internet and other manipulation datasets, long-horizon interaction simulations fail half of the time (red text).