TWIST: Teacher-Student World Model Distillation for Efficient Sim-to-Real Transfer

Jun Yamada, Marc Rigter, Jack Collins, Ingmar Posner

***Abstract*— Model-based RL is a promising approach for real-world robotics due to its improved sample efficiency and generalization capabilities compared to model-free RL. However, effective model-based RL solutions for vision-based real-world applications require bridging the sim-to-real gap for any world model learnt. Due to its significant computational cost, standard domain randomisation does not provide an effective solution to this problem. This paper proposes *TWIST* (Teacher-Student World Model Distillation for Sim-to-Real Transfer) to achieve efficient sim-to-real transfer of vision-based model-based RL using distillation. Specifically, *TWIST* leverages state observations as readily accessible, privileged information commonly garnered from a simulator to significantly accelerate sim-to-real transfer. Specifically, a teacher world model is trained efficiently on state information. At the same time, a matching dataset is collected of domain-randomised image observations. The teacher world model then supervises a student world model that takes the domain-randomised image observations as input. By distilling the learned latent dynamics model from the teacher to the student model, *TWIST* achieves efficient and effective sim-to-real transfer for vision-based model-based RL tasks. Experiments in simulated and real robotics tasks demonstrate that our approach outperforms naive domain randomisation and model-free methods in terms of sample efficiency and task performance of sim-to-real transfer.**

I. Introduction

Deep reinforcement learning (RL) has been applied successfully to challenging control problems such as dexterous manipulation [1], locomotion [2], and Atari [3]. A particularly promising approach is *model-based* RL, which learns a *world model* of the environment, and utilises this model for planning or policy optimisation. Compared to *model-free* approaches, model-based RL holds the potential for broader generalisation [4], improved sample efficiency [5], [6], and faster adaptation to new tasks [7], [8]. However, while model-based RL algorithms have been highly successful in simulated environments [9], [10], their application to real-world robots remains limited due to the need for unsafe or costly data collection [11] to train a world model in the real world.

Instead of training an RL agent directly in the real world, *sim-to-real transfer* is a common approach: learning a policy from easily accessible simulated data and deploying it in the real environment. In real-world environments, we often do not have access to accurate state information, and therefore we wish to learn a policy that utilises images as inputs. To overcome the gap between the simulator and the real world, *domain randomisation* (DR) is often employed. DR exposes the policy to a wide range of simulated environments during training to improve generalisation to the real environment. However, a significant drawback of DR is that policy training on randomised environments requires much more data [12]. Therefore, RL with DR can be extremely computationally intensive and may require weeks of computation time for training to converge [1].

Applied AI Lab, Oxford Robotics Institute, University of Oxford
Correspondence to: jyamada@robots.ox.ac.uk
https://twist-sim2real.github.io/

The vast majority of existing work on sim-to-real transfer is applied to model-free RL [13], [14], [12], [15]. In this work, we address the uninvestigated area of sim-to-real transfer for model-based RL trained from images. By leveraging model-based RL algorithms, we benefit from the improved sample efficiency of model-based approaches [5]. However, to address the sim-to-real gap, it is still necessary to apply DR. Similar to applying DR to the model-free case, naïvely applying DR to model-based approaches increases the amount of data required to train a suitable world model, and is therefore computationally very demanding [8].

To address this, we propose **T**eacher-Student **W**orld model **Di**stillation for **S**im-to-Real **T**ransfer (*TWIST*). *TWIST* leverages privileged information in a simulator to achieve efficient and robust sim-to-real transfer for model-based RL. In particular, *TWIST* utilises two world models, a *teacher* and a *student*, to learn the environment. The input to the *teacher* is state information that is only accessible within the simulator. The teacher model is therefore unaffected by appearance changes as introduced by DR and can learn to represent the environment dynamics within a compact latent space much more efficiently than a vision-based model. The teacher model then supervises a *student* world model by encouraging it to encode domain-randomised image observations to the same latent representation as the teacher. We demonstrate that *TWIST* provides efficient and effective sim-to-real transfer for model-based RL, outperforming the standard DR-based approach almost by an order of magnitude in terms of success rate when applied to real-world manipulation tasks.

Our general approach of combining world model distillation with DR is applicable to any model-based RL algorithm. In our implementation, we specifically use the DreamerV2 model architecture [9] to learn the world models and associated policies, and apply our approach to a set of simulated and real robotics environments. We show that our approach successfully achieves transfer to real-world environments, and outperforms naïve DR and model-free approaches in terms of sample efficiency and performance. Our work demonstrates empirically, that there is significant potential for sim-to-real transfer of model-based RL, extending its applicability to a wide range of real-world robotics applications.

II. Related Works

The key concepts that *TWIST* builds upon include model-based RL, sim-to-real transfer, and distillation using privileged information. We review the relevant literature of each of these concepts in turn.

Model-based RL has emerged as a promising approach to solving complex control problems by leveraging a learned dynamics model [16], [17], [18]. To achieve the desired behaviour, the dynamics model (or "world" model) can be used for planning [5], [19], [10], [20], or policy optimisation [16], [9], [21], [22]. To handle partially observable environments [23] with high-dimensional observations such as images, a common approach is to employ a recurrent state-space model (RSSM) [24], [16], [17], which predicts transitions in a compact latent space with a recurrent module. Despite considerable success on simulated environments, such as Atari [25] and DMControl [26], applications of vision-based model-based RL to real-world robotics tasks remain limited due to the need for a large number of samples to train the world model [27], [28]. Existing works on model-based RL from images for robotics [27], [28] build upon a suite of Dreamer algorithms [29], [9], [21], which achieves state-of-the-art performance on simulated domains by optimising a policy using only synthetic data generated by the model. DayDreamer [27] relies upon either state information or discretised action-spaces to simplify robotics tasks, and to facilitate learning a model from data collected directly in the real world. Existing approaches to transferring Dreamer from simulation to real robots either require state information [30] or only demonstrate transfer to near-identical real-world environments [28].

Sim-to-real transfer [31] trains a policy using simulated data, and deploys the policy in the real world. Existing approaches to sim-to-real transfer utilise techniques such as domain randomisation (DR) [32], system identification [33], and domain adaptation [34]. DR is a particularly simple, yet effective approach to expose agents to a wide range of instances of the same environment by randomising visual and dynamics parameters. By training policies using DR, agents become more robust to domain mismatches [32]. Previous work on sim-to-real transfer using DR has been primarily applied to model-free RL methods [13], [35], [36] or imitation learning [37].

Compared to sim-to-real transfer of model-free RL algorithms, model-based sim-to-real methods remain relatively unexplored. To our knowledge, [30] is the only work to transfer a model-based method across the sim-to-real gap. The authors accomplish this using a state-based Dreamer model that requires privileged information *in the real world*. Enabling sim-to-real transfer of Dreamer from image observations will help to unleash the potential of model-based RL for real-world applications where state information is not available.

Leveraging **privileged information** to accelerate the training of policies is a common approach. Specifically, [13], [35] utilise information asymmetric actor-critic methods to train the critic faster via access to the privileged information while providing only images for the actor.

Another common technique to make use of the privilege information is **Distillation**, which transfers knowledge about a task from one or multiple teachers to a student. In RL, knowledge transfer is generally achieved via *policy* distillation: training a student policy to imitate a teacher policy [38], [14], [39], [40], [41]. Our work is most closely related in spirit to [38], [14], [41] in that distillation and DR are used to efficiently train a teacher policy from privileged information and distil it into a student policy for sim-to-real transfer. However, for distillation, the prior works focus on model-free RL, which often requires additional trajectories collected by either the teacher or student policy to match the action distribution.

In contrast to these works, we consider model-based RL conditioned on image observations and introduce a novel method for *world model* distillation. Our approach achieves knowledge transfer by supervising a student world model instead of a policy without the need for additional data collection during the distillation. We demonstrate that our approach achieves strong performance for sim-to-real transfer in both simulated and real environments.

III. Preliminaries

In this section, we describe our problem setting and the Dreamer model-based RL algorithm [29], [9], [21]. We implement our approach using Dreamer as it is a commonly used state-of-the-art model-based RL algorithm that demonstrates the capability of our world model distillation approach.

A. Problem Formulation

The real environment is a partially observable Markov Decision Process represented by the tuple $(\mathcal{S}, \mathcal{O}, \mathcal{A}, \mathcal{P}, \mathcal{I}, r, \gamma, \mathbb{S})$, where: $\mathcal{S}$ is a set of continuous states, $\mathcal{O}$ is a set of image observations, $\mathcal{A}$ is a set of continuous actions, $\mathcal{P} : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to \mathbb{R}$ is the transition function, $\mathcal{I} : \mathcal{O} \times \mathcal{S} \to \mathbb{R}$ is the observation function, $r : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is the reward function, γ is the discount factor, and $\mathbb{S}$ is the initial state distribution. The goal is to maximise the expected discounted reward $\mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t r_t\right]$.

In our problem setting, we do not have access to the real environment during training. Instead, we have access to a simulator that approximates the real environment. In the simulator, we have direct access to privileged information, $s \in \mathcal{S}$, in addition to randomised image observations $o \in O$.

B. Dreamer

Dreamer [9], [21] is a model-based RL method that learns a world model from pixels or state observations and trains an actor-critic agent by leveraging imagined trajectories from the world model.

a) World Model: Dreamer uses a Recurrent State Space Model (RSSM) [17] to learn the dynamics of environments, consisting of the following modules:

$$\text{RSSM} \begin{cases} \text{Sequence model:} & h_t = f_\phi\left(h_{t-1}, z_{t-1}, a_{t-1}\right) \\ \text{Representation model:} & z_t \sim q_\phi\left(z_t \mid h_t, x_t\right) \\ \text{Dynamics predictor:} & \hat{z}_t \sim p_\phi\left(\hat{z}_t \mid h_t\right) \\ \text{Reward predictor:} & \hat{r}_t \sim p_\phi\left(\hat{r}_t \mid h_t, z_t\right) \\ \text{Decoder:} & \hat{x}_t \sim p_\phi\left(\hat{x}_t \mid h_t, z_t\right) \end{cases} \tag{1}$$

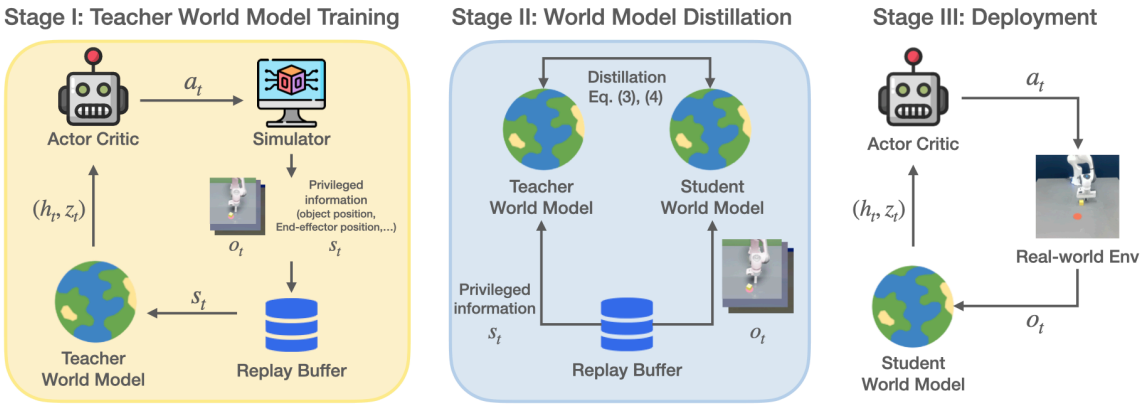

Fig. 1: Overview of *TWIST*. While a teacher world model is trained from privileged information, domain-randomised image observations are collected for distillation. The teacher supervises a student trained from the domain-randomised images to imitate the compact latent states of the teacher. The student world model is then transferred to real-world environments.

All modules are implemented as neural networks parameterised by ϕ. In the RSSM, the state is jointly represented by a recurrent deterministic component, h_t, and a stochastic component represented by a categorical distribution. At each step, the RSSM uses h_t to compute two distributions over the stochastic state: z_t and $\hat{z}$. The stochastic posterior state z_t encodes information about the current input observation x_t, while the prior state $\hat{z}_t$ is a prediction of the posterior state z_t without access to the current input observation. Therefore, by learning to predict $\hat{z}_t$, the model learns to predict the dynamics of the environment. Given the posterior state, the decoder and reward predictor are trained to reconstruct the current input observation x_t and the reward r_t, respectively. These models are jointly learned by minimising the negative variational lower bound [42].

$$\mathcal{L}(\theta) \doteq \mathbb{E}_{q_\theta(s_{1:T}|a_{1:T},x_{1:T})}\Big[\sum_{t=1}^{T}(-\ln p_\theta(x_t \mid h_t, z_t) - \ln p_\theta(r_t \mid h_t, z_t) + \beta KL\left[q_\theta(z_t \mid h_t, x_t) \| p_\theta(\hat{z}_t \mid h_t)\right]\Big] \tag{2}$$

Once the model has been trained, it can be rolled out without access to any input observations by utilising the prior $\hat{z}$ in place of the posterior z. This enables the model to generate unlimited synthetic or *imagined* trajectories of the form: $\{\{h_t, \hat{z}_t, a_t, r_t\}_{t=0}^{t=T}\}$, where T is the time horizon for imagination.

b) Actor-Critic Learning: To learn a policy, Dreamer leverages an actor-critic algorithm that is trained using synthetic data generated by the world model. Given a particular RSSM state $(h_t, \hat{z}_t)$, the critic is trained to predict the total expected reward. The actor (i.e. the policy) is trained to output a distribution over actions, $\pi(a_t|h_t, \hat{z})$, that maximises the total expected reward given the current state.

IV. TWIST: Teacher-Student World Model Distillation for Sim-to-Real Transfer

Dreamer is capable of efficiently solving diverse vision-based continuous control tasks in simulated environments by explicitly learning a task-agnostic world model. To transfer Dreamer to real-world robotics tasks, domain randomisation (DR) is required to bridge the gap between simulation and real-world environments. However, DR dramatically increases the number of samples, and therefore computation time, required for training. To address this issue, we propose *TWIST* (**T**eacher-Student **W**orld Model **Di**stillation for **S**im-to-Real **T**ransfer) to efficiently train a world model for vision-based tasks in simulation which readily transfers into real-world environments. In this section, we describe our approach to distilling the teacher to the student world model (see Fig. 1).

A. Overview

A simulator affords access to state information in addition to domain-randomised images. *TWIST* leverages this privileged information in order to accelerate the sim-to-real transfer of model-based RL. Specifically, *TWIST* initially trains a teacher world model and associated policy based on state information. Because the teacher learns from state information, an accurate world model and strong policy can be trained from only a small number of samples.

However, in real-world environments, privileged information is not usually available. To overcome this issue, the teacher is distilled into a vision-based student world model. While training the teacher from the state observations s_t, privileged information easily accessible in simulation, a matching dataset of domain-randomised image observations o_t is generated, denoted as $\mathcal{D} = \{(s_t, o_t, a_t, r_t), ...\}$. The student is trained to imitate the RSSM latent states of the teacher while operating on the corresponding domain-randomised raw pixel inputs o_t from the dataset $\mathcal{D}$. Aligning these representations enables effective knowledge transfer and achieves sample-efficient sim-to-real transfer.

B. World Model Distillation

Given the teacher world model trained on state information, the teacher supervises the student to imitate the dynamics of the environment. Specifically, the student is trained to imitate the prior distribution $p(\hat{z}_t^{\text{teacher}}|h_t^{\text{teacher}})$, posterior distribution $q(z_t^{\text{teacher}}|h_t^{\text{teacher}}, s_t)$, and deterministic representations h_t^{teacher} of the teacher for a trajectory τ

of length L sampled from the dataset, $\mathcal{D}$:

$$\mathcal{L}_{\text{distil}}(\tau) = \mathbb{E}_{\{(a_t,o_t,s_t)\}_{t=k}^{k+L}\sim\mathcal{D}} \sum_{t=k}^{k+L} \Bigg[\underbrace{||h_t^{\text{teacher}} - h_t^{\text{student}}||_2^2}_{\text{Deterministic representation distillation}} + \underbrace{\mathbb{KL}[p_\theta(\hat{z}_t^{\text{student}}|h_t^{\text{student}})||p_\phi(\hat{z}_t^{\text{teacher}}|h_t^{\text{teacher}})]}_{\text{Prior distillation}} + \underbrace{\mathbb{KL}[q_\theta(z_t^{\text{student}}|h_t^{\text{student}}, o_t)||q_\phi(z_t^{\text{teacher}}|h_t^{\text{teacher}}, s_t)]}_{\text{Posterior distillation}} \Bigg] \tag{3}$$

where ϕ and θ represent the parameters of the teacher and student world model, respectively. Note that the parameter of the teacher world model ϕ is frozen during the distillation.

In addition to distilling the two stochastic distributions and deterministic representations, we further derive a training signal for distribution alignment by matching imagined rollouts in both the teacher and the student models. (Algorithm 1). Specifically, a set of initial latent states in each world model is computed by embedding the trajectories τ sampled from the dataset $\mathcal{D}$ (see lines 6 and 7). Starting from the initial states of the teacher, we then generate an imagined rollout $\hat{\tau}^{\text{teacher}} = \{(\hat{z}_i^{\text{teacher}}, h_i^{\text{teacher}}, a_i^{\text{teacher}})\}_{i=t}^{t+H}$ with the time horizon H using the policy π learned with the teacher model (line 8). We also collect an imagined trajectory $\tau^{student}$ in the student world model by replaying the same sequence of actions $\{a_i^{\text{teacher}}\}_{i=1}^{H}$ used for trajectory imagination in the teacher (line 10). Then, we align the prior distribution $p(\hat{z}_t|h_t)$ and deterministic representation h_t in the trajectories generated by the teacher and student world model:

$$\mathcal{L}_{\text{imagined}}(\hat{\tau}^{student}, \hat{\tau}^{\text{teacher}}) = \sum_{i=1}^{H} \Bigg[\underbrace{||h_i^{\text{teacher}} - h_i^{student}||_2^2}_{\text{Deterministic representation distillation}} + \underbrace{\mathbb{KL}[p_\theta(\hat{z}_i^{\text{student}}|h_i^{\text{student}})||p_\phi(\hat{z}_i^{\text{teacher}}|h_i^{\text{teacher}})]}_{\text{Distillation in Imagination}} \Bigg] \tag{4}$$

where $(h_i^{\text{teacher}}, z_i^{\text{teacher}})$ and $(h_i^{\text{student}}, z_i^{\text{student}})$ are the i^{th} entries in $\hat{\tau}^{\text{teacher}}$ and $\hat{\tau}^{\text{student}}$ respectively. To ensure diversity in the imagined trajectories, random noise is added to the action a_t sampled from the policy $\pi(a_t|h_t^{\text{teacher}}, \hat{z}_t^{\text{teacher}})$ when rolling it out in the teacher world model. This bootstraps the trajectories in the dataset $\mathcal{D}$; thus, the student can imitate the prior distribution and deterministic representation of the teacher more accurately. The loss function for world model distillation is therefore $\mathcal{L} = \mathcal{L}_{\text{distill}} + \mathcal{L}_{\text{imagined}}$. Our experimental results demonstrate that, after distillation, an actor trained in the teacher world model successfully transfers to real-world environments as the student world model is trained to imitate the RSSM latent states of the teacher.

V. Implementation Details

Our encoder and decoder of the teacher world model consists of three fully connected hidden layers with 512 units and ELU activation. We use the same architecture for the encoder, decoder, and actor-critic agent of vision-based world models as those used in [9]. For distillation, a trajectory of length $L = 50$ is sampled from the dataset $\mathcal{D}$ (see Eq. 3) and an imagined trajectory of length $H = 15$ is generated (see Eq. 4). All of the agents are trained on a single GeForce RTX 3090 for $500K$ environment steps.

Algorithm 1 TWIST: Teacher-Student World Model Distillation for Sim-To-Real Transfer

1: **Inputs:** Dataset $\mathcal{D} = \{(s_i, o_t, a_t, r_t), ...\}$; Teacher world model W_ϕ^{teacher}; Policy $\pi(a_t|h_t, \hat{z}_t)$
2: **Initialise:** Student world model $W_\theta^{\text{student}}$
3: **while** distilling world model **do**
4: $\quad\tau = \{(a_t, o_t, s_t)\}_{t=k}^{k+L} \sim D$
5: $\quad$Compute $\mathcal{L}_{\text{distill}}$ via Eq. 3 using τ
6: $\quad\mathcal{Z}_\tau^{\text{teacher}} = \{z_t^{\text{teacher}}\}_{t=k}^{k+L} \leftarrow q_\phi(\tau)$
7: $\quad\mathcal{Z}_\tau^{\text{student}} = \{z_t^{\text{student}}\}_{t=k}^{k+L} \leftarrow q_\theta(\tau)$
8: $\quad\hat{\tau}^{teacher}$ = IMAGINE(W_ϕ^{teacher}, Z_τ^{teacher})
9: $\quad A^{\text{teacher}} \leftarrow \{a_i\}_{i=1}^{H}$ in $\hat{\tau}^{\text{teacher}}$
10: $\quad\hat{\tau}^{\text{student}}$ = IMAGINE($W_\theta^{\text{student}}$, Z_τ^{student}, A^{teacher})
11: $\quad$Compute $\mathcal{L}_{\text{imagined}}$ via Eq. 4
12: $\quad\theta \leftarrow \theta - \alpha\nabla_\theta(\mathcal{L}_{\text{distill}} + \mathcal{L}_{\text{imagined}})$
13: **function** IMAGINE(W, Z_{init}, A = None)
14: $\quad$**if** A is None **then** ▷ Imagination in W^{teacher}
15: $\quad\quad\hat{\tau} \leftarrow$ rollout π for H steps from $z \in Z_{\text{init}}$ in W
16: $\quad$**else** ▷ Imagination in W^{student}
17: $\quad\quad\hat{\tau} \leftarrow$ rollout $a \in A$ from $z \in Z$ in W
18: $\quad$**return** $\hat{\tau}$ ▷ $\hat{\tau} = \{(\hat{z}_i, h_i, a_i)\}_{i=1}^{H}$

VI. Experiments

The efficacy of *TWIST* for sim-to-real transfer is evaluated through experiments in both simulated and real-world environments. The experiments aim to answer the following questions: (1) does *TWIST* enable efficient sim-to-real transfer for model-based RL using DR? and (2) does the distillation for imagined trajectories improve the task performance compared to performing distillation only on the original dataset?

A. Baselines

We compare *TWIST* against several competitive baselines, including Dreamer agents with different training methods and model-free RL. *Oracle* is a Dreamer agent trained from privileged information. The performance of the oracle agent is an upper bound on the performance of our method. Since we do not have access to state information in real-world settings, we only provide the performance of the oracle approach in the experiments conducted in simulation environments. *Dreamer w/ DR* is a vision-based Dreamer agent trained with naive DR. *Dreamer w/o DR* is an agent trained without DR. *Dreamer State Recon.* is a vision-based Dreamer agent trained to reconstruct state information from domain-randomised image observations, which is an alternative way of leveraging privileged information. Lastly, *Asymmetric SAC* [13] is a sample-efficient state-of-the-art model-free RL algorithm suitable for DR. While the critic network is trained from privileged information, the policy is trained from domain-randomised image observations.

B. Simulated Results

Firstly, we empirically demonstrate the efficacy of *TWIST* on a set of continuous control tasks in the Distracting Control

Tasks (500K Steps)	Oracle Dreamer	TWIST	Dreamer w/o DR	Dreamer w/ DR	Dreamer State Recon.	Asymmetric SAC
Cup Catch	936.6 ± 0.1	856.6 ± 29.6	150.7 ± 80.3	744.3 ± 93.8	627.0 ± 194.7	873.0 ± 11.1
Cartpole, Balance	992.9 ± 1.7	954.5 ± 37.4	349.3 ± 19.0	590.8 ± 23.0	869.7 ± 40.4	353.1 ± 18.6
Cheetah Run	597.1 ± 24.3	506.0 ± 54.2	206.5 ± 40.3	476.4 ± 61.1	391.6 ± 4.5	222.5 ± 20.4
Hopper Stand	501.1 ± 38.9	483.3 ± 118.9	42.1 ± 11.2	471.8 ± 48.4	358.2 ± 34.4	57.7 ± 86.8
Walker Walk	800.4 ± 53.7	665.8 ± 80.3	182.7 ± 3.4	394.6 ± 25.1	491.0 ± 145.6	439.4 ± 57.0
Finger, Easy Turn	904.4 ± 32.8	798.0 ± 50.3	182.5 ± 26.0	440.7 ± 35.7	553.4 ± 42.2	304.4 ± 22.5

TABLE I: Averaged episodic rewards and standard deviation obtained from 100 trials with 3 seeds in the Distracting Control Suite. The evaluation is conducted using held-out environments.

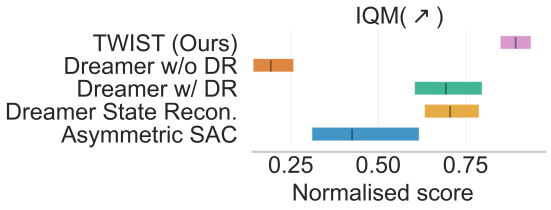

Fig. 2: Aggregated Interquartile Mean (IQM) of normalised episodic rewards with 95% bootstrap CI based on 5 tasks from 100 trials with 3 seeds evaluated using held-out environments in Distracting Control. The lack of overlap with the CIs between TWIST and the baseline methods indicates that the difference is statistically significant.

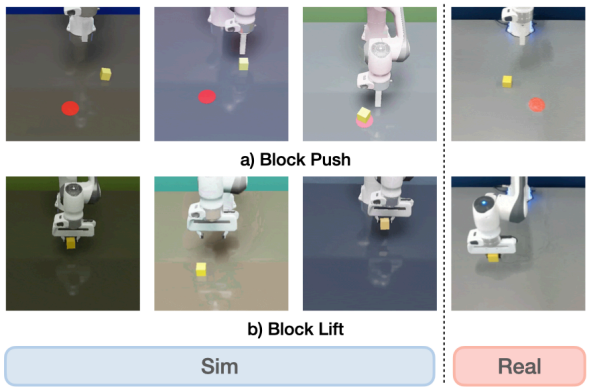

Fig. 3: Sim-to-real manipulation tasks. (a) *Block Push*: A Franka Panda arm pushes the yellow block towards the red goal marker. (b) *Block Lift*: The arm grasps the yellow block and lifts it 10cm above the table

Suite [43], an extended version of the DMControl [26].

1) Experiment Setup: First, a teacher world model and a policy are trained using Dreamer from ground-truth state information. During training, domain-randomised images are collected by randomizing the background texture used in prior work [44] and the colour of objects every timestep for diverse data acquisition. After training the teacher, we use the domain-randomised image observations to distil the state-based teacher world model into a vision-based student world model. For evaluation, we sample the object colours from the same distribution as training, but the background texture is sampled from a held-out test distribution. Therefore, the distribution of environments for evaluation is different to the training time environments. Note that DR is applied only at the beginning of the episode for the evaluation because the textures are usually consistent at test time.

2) Results: Table I reports the average episodic rewards for six continuous control tasks evaluated on hold-out scenes from the Distracting Control Suite. *TWIST* outperforms the baseline approaches, including model-free RL, often by significant margins. While *Asymmetric SAC* shows comparable performance on the simple *Cup Catch* task, it does not perform well on more complex tasks because the policy struggles to learn task-relevant information efficiently from domain-randomised images due to its visual complexity. *Dreamer State Recon.* and *Dreamer w/ DR* demonstrate better performance among the baselines. However, learning task-relevant information and the actor-critic agent jointly on limited samples is often challenging, resulting in worse performance compared to our approach. *Dreamer w/o DR* does not perform well in any of the six tasks due to the lack of generalisation to unseen scenes.

To assess the statistical significance of our results, Fig. 2 reports Interquartile Mean (IQM) of normalised episodic rewards with 95% bootstrap confidence interval (CI) aggregated across 5 tasks in Distracting Control, computed using [45]. The episodic rewards of each task are normalised by the performance of *Oracle Dreamer* to aggregate the results and validate the efficacy of our method. As shown in Fig. 2, our method is substantially more performant than the baselines. The lack of overlap with the CIs of the baseline method further indicates that this difference is statistically significant.

C. Sim-to-Real Transfer for Manipulation Tasks

In this section, we consider sim-to-real transfer for manipulation tasks to verify the effectiveness of *TWIST* in the real world.

1) Experimental setup: In our experiments, a Franka Panda robot is used. In real-world experiments, RGB image observations are taken from a RealSense D435i camera. In the simulation, agents are trained in Omniverse Isaac Orbit [46] powered by Omniverse Isaac Sim [47]. DR is applied to the brightness of the light and texture of the robot body, background, table, and objects every timestep to collect diverse image observations. Further, the friction of objects is randomised in every episode. The action space of the policy is a delta-position of the end-effector in Cartesian coordinates with a maximum delta of 2cm.

2) Tasks: We conduct experiments to showcase the successful sim-to-real transfer capability of *TWIST*, focusing on the *block push* and *block lift* tasks (see Fig. 3). The objective of the *block push* task is to push a $4\text{cm} \times 4\text{cm}$ cube towards a designated red goal marker. If the distance between the centre of the cube and the goal marker is less than 5cm at the end of the episode, then the trial is considered successful. The cube and goal marker positions are uniformly sampled. For the block push task, we replace the robot's hand with a

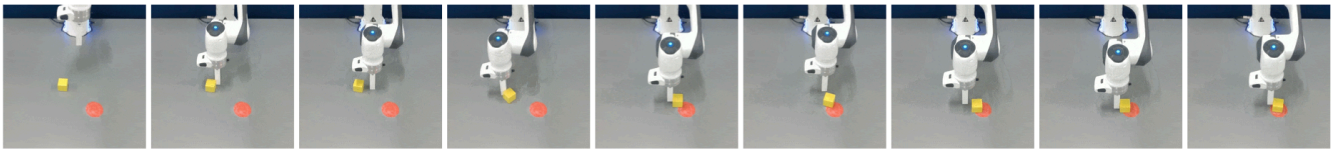

Fig. 4: Example rollouts of the proposed method on the real-world Block Push task. Our method successfully transfers the student world model and solves the block push task in the real world.

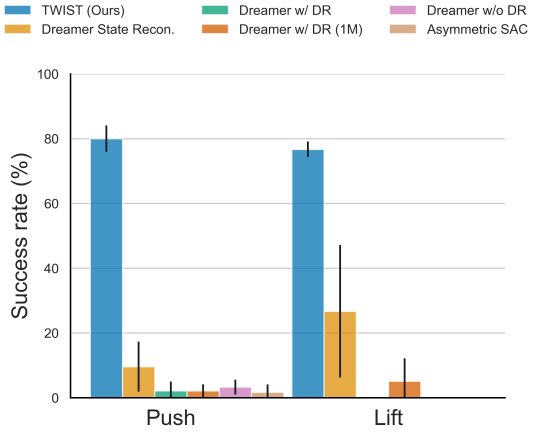

Fig. 5: Success rate on real-world tasks. The success rate and standard deviation are calculated from 20 trials with 3 seeds. *TWIST* significantly outperform baselines including naïve Dreamer with DR and model-free RL.

3D-printed peg to push the block because the original robot's hand often occludes the block from the third-person camera.

The goal of the *block lift* task is to grasp the cube and lift it 10cm above the tabletop by the end of the episode. To train agents in simulation, we define a dense reward function tailored to each task. Privileged information available in these tasks includes end-effector position, object pose, and L2 distance between the object and goal position. Additionally, in the *block lift* task, a grasp state is used to determine whether the object is grasped or not. The episode length of these tasks is 150 timesteps. In real-world experiments, we randomise the camera position and brightness of the scene randomly to ensure robustness of the distilled agents.

3) Results: The success rate for each task across 20 trials averaged over 3 seeds is reported in Fig. 5. Compared to the baselines, including naïve Dreamer with DR and model-free RL, *TWIST* demonstrates significantly better success rates in both *block push* and *lift* tasks. In particular, the block push task requires an accurate dynamics model to successfully push the box towards the goal marker, indicating that our world model is successfully distilled and transferred from simulation to real-world environments. The baseline methods often fail to solve the task, because those methods require more samples to successfully train agents in simulation with DR [8]. *Dreamer State Recon.* shows a better success rate than other baselines. However, it still struggles to learn task-relevant information in image observations effectively while exploring environments for solving manipulation tasks. Although naïve Dreamer agent with DR is also trained from 1M samples (*Dreamer w/ DR* (1M)), its success rate on the *block push* and *block lift* tasks remains low, indicating the sample inefficiency of the naïve DR approach.

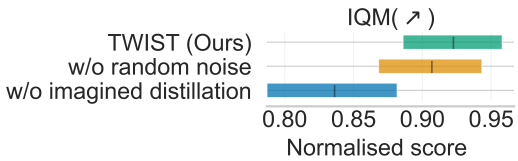

Fig. 6: Interquartile Mean (IQM) of normalised episodic rewards with 95% bootstrap CI to ablate the key components of the proposed distillation method in the Distracting Control Suite. The following variants are compared: (1) the full proposed method, (2) without random noise to actions for imagined distillation, (3) without imagined distillation.

D. Ablation Study

We ablate the distillation for imagined trajectories (*imagined distillation*) (see Eq. 4) and random noise added to actions for the imagined distillation in the Distracting Control Suite. We report normalised aggregated Interquartile Mean (IQM) with a 95% bootstrap confidence interval. As shown in Fig. 6, the CI for *our method* and *our method w/o imagined distillation* do not overlap, indicating that the difference in performance is statistically significant. On the other hand, the gap between *our method* and *our method w/o random noise* is smaller but still notable in practice. These results highlight that distillation using imagined rollouts is particularly important for successful world model distillation.

VII. Conclusion

We propose *TWIST* for efficient sim-to-real transfer of model-based RL. Specifically, a teacher world model trained from privileged information supervises a student world model taking as input domain-randomised image observations to mimic the compact latent states of the teacher. Our experiments demonstrate successful distillation from the teacher world model to the student world model with domain randomisation in simulated environements and further show the efficient and robust sim-to-real transfer for robot manipulation tasks into real-world domains.

TWIST is therefore a significant step towards unlocking the benefits of model-based RL for real-world applications. In future work we will look to explore fine-tuning the distilled world model from few real-world image observations to efficiently acquire new skills in the real world.

Acknowledgment

This work was supported by a UKRI/EPSRC Programme Grant [EP/V000748/1]. We would also like to thank the University of Oxford for providing Advanced Research Computing (ARC) and the SCAN facility in carrying out this work (`http://dx.doi.org/10.5281/zenodo.22558`).

REFERENCES

[1] 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.

[2] T. Haarnoja, S. Ha, A. Zhou, J. Tan, G. Tucker, and S. Levine, "Learning to walk via deep reinforcement learning," *Robotics: Science and Systems*, 2019.

[3] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, *et al.*, "Human-level control through deep reinforcement learning," *nature*, vol. 518, no. 7540, pp. 529–533, 2015.

[4] T. Yu, A. Kumar, R. Rafailov, A. Rajeswaran, S. Levine, and C. Finn, "Combo: Conservative offline model-based policy optimization," *Advances in neural information processing systems*, vol. 34, pp. 28 954–28 967, 2021.

[5] 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.

[6] 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.

[7] R. Sekar, O. Rybkin, K. Daniilidis, P. Abbeel, D. Hafner, and D. Pathak, "Planning to explore via self-supervised world models," in *International Conference on Machine Learning*. PMLR, 2020, pp. 8583–8592.

[8] M. Rigter, M. Jiang, and I. Posner, "Reward-free curricula for training robust world models," *arXiv preprint arXiv:2306.09205*, 2023.

[9] D. Hafner, T. Lillicrap, M. Norouzi, and J. Ba, "Mastering atari with discrete world models," 2022.

[10] J. Schrittwieser, I. Antonoglou, T. Hubert, K. Simonyan, L. Sifre, S. Schmitt, A. Guez, E. Lockhart, D. Hassabis, T. Graepel, *et al.*, "Mastering atari, go, chess and shogi by planning with a learned model," *Nature*, vol. 588, no. 7839, pp. 604–609, 2020.

[11] S. Levine, A. Kumar, G. Tucker, and J. Fu, "Offline reinforcement learning: Tutorial, review, and perspectives on open problems," *arXiv preprint arXiv:2005.01643*, 2020.

[12] S. Salter, D. Rao, M. Wulfmeier, R. Hadsell, and I. Posner, "Attention-privileged reinforcement learning," in *Conference on Robot Learning*. PMLR, 2021, pp. 394–408.

[13] L. Pinto, M. Andrychowicz, P. Welinder, W. Zaremba, and P. Abbeel, "Asymmetric actor critic for image-based robot learning," 2017.

[14] J. Brosseit, B. Hahner, F. Muratore, M. Gienger, and J. Peters, "Distilled domain randomization," *arXiv preprint arXiv:2112.03149*, 2021.

[15] S. James, P. Wohlhart, M. Kalakrishnan, D. Kalashnikov, A. Irpan, J. Ibarz, S. Levine, R. Hadsell, and K. Bousmalis, "Sim-to-real via sim-to-sim: Data-efficient robotic grasping via randomized-to-canonical adaptation networks," 2019.

[16] D. Ha and J. Schmidhuber, "World models," *arXiv preprint arXiv:1803.10122*, 2018.

[17] 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.

[18] R. S. Sutton, "Dyna, an integrated architecture for learning, planning, and reacting," *ACM Sigart Bulletin*, vol. 2, no. 4, pp. 160–163, 1991.

[19] R. Y. Rubinstein, "Optimization of computer simulation models with rare events," *European Journal of Operational Research*, vol. 99, no. 1, pp. 89–112, 1997. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0377221796003852

[20] G. Williams, A. Aldrich, and E. Theodorou, "Model predictive path integral control using covariance variable importance sampling," 2015.

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

[22] M. Rigter, B. Lacerda, and N. Hawes, "RAMBO-RL: Robust adversarial model-based offline reinforcement learning," *Advances in Neural Information Processing Systems*, 2022.

[23] L. P. Kaelbling, M. L. Littman, and A. R. Cassandra, "Planning and acting in partially observable stochastic domains," *Artificial intelligence*, vol. 101, no. 1-2, pp. 99–134, 1998.

[24] J. Schmidhuber, "Reinforcement learning in markovian and non-markovian environments," *Advances in neural information processing systems*, vol. 3, 1990.

[25] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller, "Playing atari with deep reinforcement learning," 2013.

[26] Y. Tassa, Y. Doron, A. Muldal, T. Erez, Y. Li, D. de Las Casas, D. Budden, A. Abdolmaleki, J. Merel, A. Lefrancq, T. Lillicrap, and M. Riedmiller, "Deepmind control suite," 2018.

[27] P. Wu, A. Escontrela, D. Hafner, K. Goldberg, and P. Abbeel, "Daydreamer: World models for physical robot learning," 2022.

[28] Y. Seo, J. Kim, S. James, K. Lee, J. Shin, and P. Abbeel, "Multi-view masked world models for visual robotic manipulation," *arXiv preprint arXiv:2302.02408*, 2023.

[29] D. Hafner, T. Lillicrap, J. Ba, and M. Norouzi, "Dream to control: Learning behaviors by latent imagination," in *International Conference on Learning Representations*, 2020. [Online]. Available: https://openreview.net/forum?id=S1lOTC4tDS

[30] A. Brunnbauer, L. Berducci, A. Brandstátter, M. Lechner, R. Hasani, D. Rus, and R. Grosu, "Latent imagination facilitates zero-shot transfer in autonomous racing," in *2022 International Conference on Robotics and Automation (ICRA)*. IEEE, 2022, pp. 7513–7520.

[31] W. Zhao, J. P. Queralta, and T. Westerlund, "Sim-to-real transfer in deep reinforcement learning for robotics: a survey," in *2020 IEEE symposium series on computational intelligence (SSCI)*. IEEE, 2020, pp. 737–744.

[32] 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.

[33] M. Lutter, J. Silberbauer, J. Watson, and J. Peters, "Differentiable physics models for real-world offline model-based reinforcement learning," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*, 2021, pp. 4163–4170.

[34] K. Bousmalis, A. Irpan, P. Wohlhart, Y. Bai, M. Kelcey, M. Kalakrishnan, L. Downs, J. Ibarz, P. Pastor, K. Konolige, *et al.*, "Using simulation and domain adaptation to improve efficiency of deep robotic grasping," in *2018 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2018, pp. 4243–4250.

[35] S. Salter, D. Rao, M. Wulfmeier, R. Hadsell, and I. Posner, "Attention privileged reinforcement learning for domain transfer," 2020. [Online]. Available: https://openreview.net/forum?id=HygW26VYwS

[36] O. M. Andrychowicz, B. Baker, M. Chociej, R. Jozefowicz, B. McGrew, J. Pachocki, A. Petron, M. Plappert, G. Powell, A. Ray, *et al.*, "Learning dexterous in-hand manipulation," *The International Journal of Robotics Research*, vol. 39, no. 1, pp. 3–20, 2020.

[37] S. James, A. J. Davison, and E. Johns, "Transferring end-to-end visuomotor control from simulation to real world for a multi-stage task," in *Conference on Robot Learning*. PMLR, 2017, pp. 334–343.

[38] I.-C. A. Liu, S. Uppal, G. S. Sukhatme, J. J. Lim, P. Englert, and Y. Lee, "Distilling motion planner augmented policies into visual control policies for robot manipulation," in *Conference on Robot Learning*. PMLR, 2022, pp. 641–650.

[39] A. A. Rusu, S. G. Colmenarejo, C. Gulcehre, G. Desjardins, J. Kirkpatrick, R. Pascanu, V. Mnih, K. Kavukcuoglu, and R. Hadsell, "Policy distillation," 2016.

[40] W. M. Czarnecki, R. Pascanu, S. Osindero, S. M. Jayakumar, G. Swirszcz, and M. Jaderberg, "Distilling policy distillation," 2019.

[41] T. Chen, J. Xu, and P. Agrawal, "A system for general in-hand object re-orientation," in *Conference on Robot Learning*. PMLR, 2022, pp. 297–307.

[42] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," *arXiv preprint arXiv:1312.6114*, 2013.

[43] A. Stone, O. Ramirez, K. Konolige, and R. Jonschkowski, "The distracting control suite – a challenging benchmark for reinforcement learning from pixels," 2021.

[44] N. Hansen and X. Wang, "Generalization in reinforcement learning by soft data augmentation," 2021.

[45] R. Agarwal, M. Schwarzer, P. S. Castro, A. C. Courville, and M. Bellemare, "Deep reinforcement learning at the edge of the statistical precipice," *Advances in Neural Information Processing Systems*, vol. 34, 2021.

[46] M. Mittal, C. Yu, Q. Yu, J. Liu, N. Rudin, D. Hoeller, J. L. Yuan, R. Singh, Y. Guo, H. Mazhar, A. Mandlekar, B. Babich, G. State, M. Hutter, and A. Garg, "Orbit: A unified simulation framework for interactive robot learning environments," *IEEE Robotics and Automation Letters*, vol. 8, no. 6, pp. 3740–3747, 2023.

[47] NVIDIA, "Nvidia isaac sim." [Online]. Available: https://developer.nvidia.com/isaac-sim