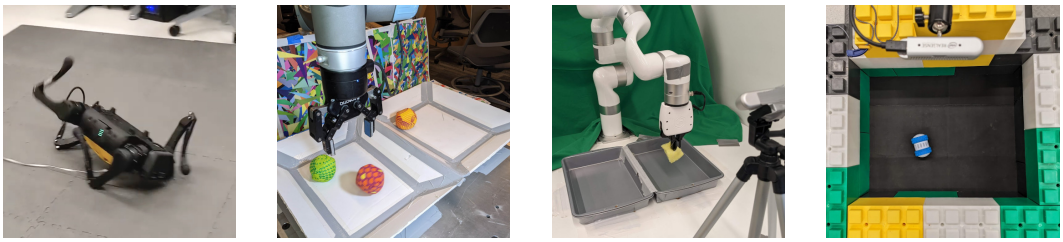
DayDreamer: World Models for Physical Robot Learning

Anonymous Author(s)
Affiliation
Address
email

Abstract: To solve tasks in complex environments, robots need to learn from experience. Deep reinforcement learning is a common approach to robot learning but requires a large amount of trial and error to learn, limiting its deployment in the physical world. As a consequence, many advances in robot learning rely on simulators. However, learning in simulation fails to capture the complexity of the real world, is prone to simulator inaccuracies, and the resulting behaviors do not adapt to changes in the real world. The Dreamer algorithm has recently shown great promise for learning from small amounts of interaction by planning within a learned world model, outperforming pure reinforcement learning in video games and simulated control domains. The world model learns to predict the outcomes of potential actions, reducing the amount of trial and error needed in the real environment. However, it is unknown whether Dreamer can facilitate faster learning on physical robots. In this paper, we apply Dreamer to four robots and tasks to learn online and directly in the real world, without any simulators. On a quadruped robot, Dreamer learns to roll off its back, stand up, and walk from scratch and without resets in only one hour. Afterwards, we manually push the robot, showing that the continuously learning robot adapts its behavior within 10 minutes to withstand pushes or quickly roll over and gets back on its feet. On two different robotic arms, Dreamer learns to pick and place multiple objects directly from camera images and sparse rewards, approaching human performance. On a wheeled robot, Dreamer learns to navigate to goal positions purely from camera inputs, automatically resolving ambiguity about the robot orientation by integrating sensory inputs over time. All experiments in this paper use the same algorithm and hyperparameters, demonstrating the generality and robustness of the approach. Results suggest that Dreamer is capable of online learning in the real world, establishing a strong baseline. We release our infrastructure as a platform for future applications of world models to robot learning.

Submitted to the 6th Conference on Robot Learning (CoRL 2022). Do not distribute.



(a) A1 Quadruped Walking (b) UR5 Visual Pick Place (c) XArm Visual Pick Place (d) Sphero Navigation

Figure 1: We leverage Dreamer to solve a range of tasks on different robots from scratch in the real world, all with the same hyper parameters and without simulators. The tasks span a range of diverse challenges, including high-dimensional continuous control, object perception from raw pixels, discrete control, and goal reaching. These results demonstrate the generality and sample-efficiency of world models and establish a strong baseline for robot learning.

1 Introduction

Teaching robots to solve complex tasks in the real world is a foundational problem of robotics research. Deep reinforcement learning (RL) offers a common approach to robot learning that enables robots to improve their behavior over time through trial and error. However, current algorithms require too much interaction with the environment to learn successful behaviors, making them impractical for many real world tasks. Recently, modern world models have shown great promise for data efficient learning in simulated domains and video games (Hafner et al., 2019; 2020). Learning world models from past experience enables robots to imagine the future outcomes of potential actions, reducing the amount of trial and error in the real environment needed to learn successful behaviors.

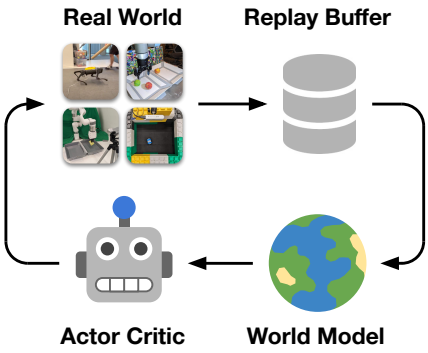
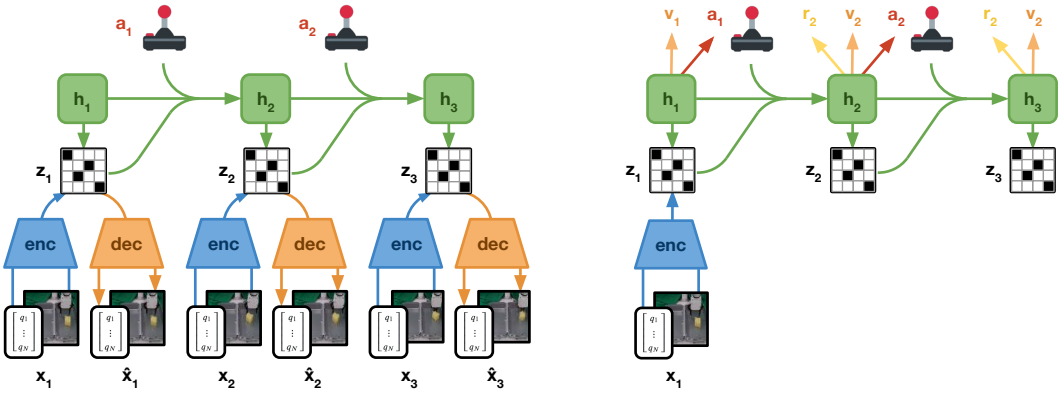


Figure 2: Dreamer follows a simple pipeline for continual online learning on robot hardware without simulators. The current learned policy collects experience on the robot. This experience is added to the replay buffer. The world model is trained on replayed off-policy sequences through supervised learning. An actor critic algorithm optimizes a neural network policy from imagined rollouts in the latent space of the world model. Our implementation parallelizes data collection and training to allow for low-latency action computation and to train quickly regardless of the control frequency on the robot.

While learning accurate world models can be challenging, they offer a range of compelling properties for robot learning. By predicting future outcomes, world models allow for planning and behavior learning given only small amounts of real world interaction (Gal et al., 2016; Ebert et al., 2018). Moreover, world models summarize general dynamics knowledge about the environment that, once learned, could be reused for a wide range of downstream tasks (Sekar et al., 2020). World models also learn representations that fuse multiple sensor modalities and integrate them into latent states, removing the need for manual state estimation. Finally, representation learning enables world models to generalize well from available offline data (Yu et al., 2021), which can further accelerate learning in the real world.

Despite the promises of world models, learning accurate world models for the real world is a big open challenge. In this paper, we leverage recent advances of the Dreamer world model for learning on a variety of robots and tasks in the most straight-forward and fundamental problem setting: end-to-end online reinforcement learning directly in the real world, without simulators or demonstrations. As shown in Figure 2, Dreamer learns a world model from a replay buffer of past experience, learns behaviors from rollouts imagined in the latent space of the world model, and continuously interacts with the environment to explore and improve its behaviors. Our aim is to push the limits of robot learning directly in the real world and offer a robust platform to enable future work that develops the benefits of world models for robot learning. We summarize the key contributions as follows:

- **Dreamer on Robots** We leverage Dreamer to apply modern world models to four robots, demonstrating successful learning directly in the real world, without introducing new complex algorithms. The tasks cover a range of challenges, including different action spaces, sensory modalities, and reward structures.
- **Walking in 1 Hour** We teach a quadruped from scratch in the real world to roll off its back, stand up, and walk in only 1 hour. Afterwards, we find that the robots adapts to being pushed within 10 minutes, learning to withstand pushes or quickly roll over and get back on its feet.
- **Visual Pick and Place** We train a robot arm to perform multi-object pick and place that requires learning to localize objects from camera images, fusing the images with proprioceptive readings, and learning end-to-end from sparse rewards. The resulting policy outperforms strong model-free learning algorithms and approaches human performance.
- **Infrastructure** We open source the software infrastructure for all our experiments, which supports different action spaces and sensory modalities, offering a flexible platform for future research of world models for robot learning in the real world.



(a) World Model Learning (b) Behavior Learning

Figure 3: **Neural Network Training** We leverage the Dreamer algorithm (Hafner et al., 2019; 2020) for fast robot learning in real world. Dreamer consists of two neural network components. **Left:** The world model follows the structure of a deep Kalman filter that is trained on sub sequences drawn from the replay buffer. The **encoder** fuses all sensory modalities into discrete codes. The decoder reconstructions the inputs from the codes, providing a rich learning signal and enabling human inspection of model predictions. A **recurrent state-space model** (RSSM) is trained to predict future codes given actions, without observing intermediate inputs. **Right:** The world model enables massively parallel policy optimization from imagined rollouts in the compact latent space using a large batch size, without having to reconstruct sensory inputs. Dreamer trains a **policy network** and **value network** from the imagined rollouts and a learned **reward function**.

2 Approach

We leverage the Dreamer algorithm (Hafner et al., 2019; 2020) for online learning on physical robots, without the need for simulators. This section summarizes the general algorithm, as well as details on the training architecture and sensor fusion needed for the robotics experiments. An overview of the approach is shown in Figure 2. Specifically, Dreamer learns a world model from a replay buffer of past experiences, uses an actor critic algorithm to learn behaviors from trajectories predicted by the learned model, and deploys its behavior in the environment to grow the replay buffer. To meet latency requirements and avoid slowing down training in environments with slow control frequency, we decouple learning updates from data collection. In our implementation, a learner thread continuously trains the world model and actor critic behavior, while an actor thread in parallel computes actions for environment interaction.

World Model Learning The world model is a deep neural network that learns the environment dynamics, as shown in Figure 3 (left). Because sensory inputs can be high-dimensional modalities like camera images, we predict ahead in the representation space rater than directly predicting future observations. This reduces accumulating errors and enables massively parallel training with a large batch size, without having to generate sensory observations. The world model is based on the Recurrent State-Space Model (RSSM; Hafner et al., 2018), which consists of four components:

$$\begin{aligned} &\text{Encoder Network:} && \mathrm{enc}_\theta(s_t \mid s_{t-1}, a_{t-1}, x_t) && \text{Decoder Network:} && \mathrm{dec}_\theta(s_t) \approx x_t \\ &\text{Dynamics Network:} && \mathrm{dyn}_\theta(s_t \mid s_{t-1}, a_{t-1}) && \text{Reward Network:} && \mathrm{rew}_\theta(s_{t+1}) \approx r_t \end{aligned} \tag{1}$$

Physical robots are often equipped with multiple sensors of different modalities, such as proprioceptive joint readings, force sensors, and high-dimensional inputs such as RGB and depth camera images. The encoder network fuses all sensory inputs x_t together into the stochastic representations z_t. The dynamics model learns to predict the sequence of stochastic representations by using its recurrent state h_t. The decoder reconstructs the sensory inputs to provide a rich signal for learning representations and enable human inspect of model predictions, but is not needed while learning behaviors from latent rollouts. In our experiments, the robot has to discover task rewards by interacting with the real world, which the reward network learns to predict. Using manually specified rewards as a function of the decoded sensory inputs is also possible. We optimize all components of the world model jointly by stochastic backpropagation (Kingma and Welling, 2013; Rezende et al., 2014).

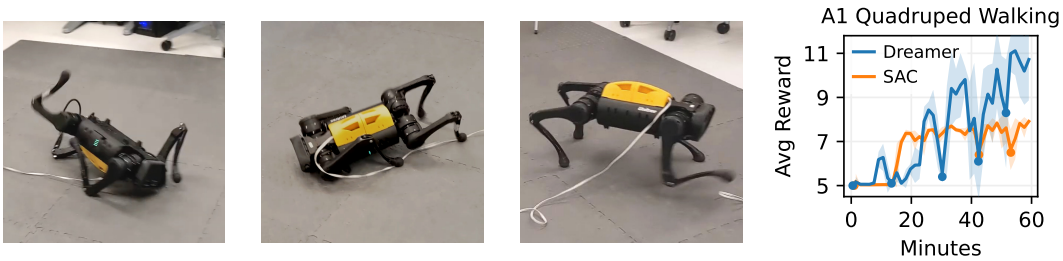


Figure 4: **A1 Quadruped Walking** Starting from lying on its back with the feet in the air, Dreamer learns to roll over, stand up, and walk in 1 hour of real world training time, without simulators or resets. In contrast, SAC only learns to roll over but neither to stand up nor to walk. For SAC, we also had to help the robot out of a dead-locked leg configuration during training. On the right we show trained curves for both SAC and Dreamer. The maximum reward is 14. The filled circles indicate times where the robot fell on its back, requiring to over time learn a robust strategy for getting back up. After 1 hour of training, we start pushing the robot and find that it adapts its behavior within 10 minutes to withstand light pushes and quickly roll back on its feet for hard pushes.

Actor Critic Learning While the world model represents task-agnostic knowledge about the dynamics, the actor critic algorithm learns a behavior that is specific to the task at hand. As shown in Figure 3 (right), the we learn behaviors from rollouts that are predicted in the latent space of the world model, without decoding observations. This enables massively parallel behavior learning with typical batch sizes of 16K, similar to specialized modern simulators (Makoviychuk et al., 2021). The actor critic algorithm consists of two neural networks:

$$\text{Actor Network:} \quad \pi(a_t \mid s_t) \qquad \text{Critic Network:} \quad v(s_t) \tag{2}$$

The role of the actor network is to learn a distribution over successful actions a_t for each latent model state s_t that maximizes the sum of future predicted task rewards. The critic network learns to predict the sum of future task rewards through temporal difference learning (Sutton and Barto, 2018). This is important because it allows the algorithm to take into account rewards beyond the planning horizon of $H = 16$ steps to learn long-term strategies. Given a predicted trajectory of model states, the critic is trained to regress the return of the trajectory. A simple choice would be to compute the return as the sum of N intermediate rewards plus the critic's own prediction at the next state. To avoid the choice of an arbitrary value for N, we instead compute λ-returns, which average over all $N \in [1, H-1]$ and are computed as follows:

$$V_t^\lambda \doteq r_t + \gamma\Big((1-\lambda)v(s_{t+1}) + \lambda V_{t+1}^\lambda\Big), \quad V_H^\lambda \doteq v(s_H). \tag{3}$$

While the critic network is trained to regress the λ-returns, the actor network is trained to maximize them. Different gradient estimators are available for computing the policy gradient for optimizing the actor, such as Reinforce (Williams, 1992) and the reparameterization trick (Kingma and Welling, 2013; Rezende et al., 2014) that directly backpropagates return gradients through the differentiable dynamics network (Henaff et al., 2019). Following Hafner et al. (2020), we choose reperameterization gradients for continuous control tasks and Reinforce gradients for tasks with discrete actions. In addition to maximizing returns, the actor is also incentivized to maintain high entropy to prevent collapse to a deterministic policy and maintain some amount of exploration throughout training:

$$\mathcal{L}(\pi) \doteq -\operatorname{E}\big[\textstyle\sum_{t=1}^{H} \ln \pi(a_t \mid s_t)\, \operatorname{sg}(V_t^\lambda - v(s_t)) + \eta \operatorname{H}\big[\pi(a_t \mid s_t)\big]\big] \tag{4}$$

We optimize the actor and critic using the Adam optimizer (Kingma and Ba, 2014). To compute the λ-returns, we use a slowly updated copy of the critic network as common in the literature (Mnih et al., 2015). The gradients of the actor and critic do not affect the world model, as this would lead to incorrect and overly optimistic model predictions. All hyperparameters of the world model are left at their default values reported in the hyperparameter table in the work of Hafner et al. (2020), except for the training frequency which inapplicable because we parallelize data collection and network optimization so that both progress independently as fast as they can.

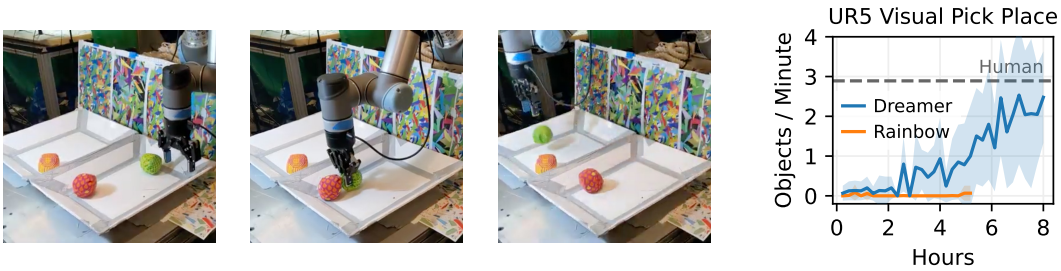


Figure 5: **UR5 Multi Object Visual Pick and Place** This task requires learning to locate three ball objects from third-person camera images, grasp them, and move them into the other bin. The arm is free to move within and above the bins and sparse rewards are given for grasping a ball and for dropping it in the opposite bin. The environment requires the world model to learn multi-object dynamics in the real world and the sparse reward structure poses a challenge for policy optimization. Dreamer overcomes the challenges of visual localization and sparse rewards on this task, learning a successful strategy within a few hours of autonomous operation.

3 Experiments

We evaluate Dreamer on four tasks across four robotics platforms and compare the results to algorithmic and human baselines. The chosen environments are representative of common RL problem settings, including state-based continuous control, continuous control from pixels, and discrete control from proprioceptive observations and pixels. Learned world models have various properties that make them well suited for robot learning. The goal of the experiments is to to evaluate whether the recent successes of learned world models enable sample-efficient robot learning directly in the real world. Specifically, we aim to answer the following research questions:

- Does Dreamer enable robot learning directly in the real world, without simulators?
- Does Dreamer succeed across various robot platforms, sensory modalities, and action spaces?
- How does the data-efficiency and asymptotic performance of Dreamer compare to previous general robot learning algorithms and human performance?

Implementation We use the same implementation of Dreamer with identical hyperparameters across all four environments, which, gracefully handles observations of different modalities, enabling off-the-shelf deployment to different robot embodiments. We make use of an asynchronous actor and learner formulation, which is essential in environments with high control rates, such as the quadrupedal walking task.

Baselines We compare to a strong learning algorithm for each of our experimental setups. The A1 quadruped robot uses continuous actions and low-dimensional inputs, allowing us to compare to SAC (Haarnoja et al., 2018a;b), a popular algorithm for data-efficient continuous control. For the visual pick and place experiments on the XArm and UR5 robots, inputs are images and pripriocepive readings and actions are discrete, suggesting algorithms from the DQN (Mnih et al., 2015) line of work as baselines. We choose Rainbow (Hessel et al., 2018) as a powerful representative of this category, an algorithm that combines many improvements of DQN. To input the proprioceptive readings, we concatenate them as broadcasted planes to the RGB channels of the image, a common practice in the literature (Schrittwieser et al., 2019). In addition we compare against a human operator controlling the robot arm through the robot control interface, which provides an approximate upper bound for the robot performance. For the Sphero navigation task, inputs are images and actions are continuous. The state-of-the-art baseline in this category is DrQv2 (Yarats et al., 2021), which uses image augmentation to increase sample-efficiency.

3.1 A1 Quadruped Walking

Setup This high-dimensional continuous control task requires training a quadruped robot to roll over from its back, stand up, and walk forward at a fixed target velocity. Prior work in quadruped locomotion requires either extensive training in simulation under domain randomization, using recovery controllers to avoid unsafe states, or defining the action space as parameterized trajectory

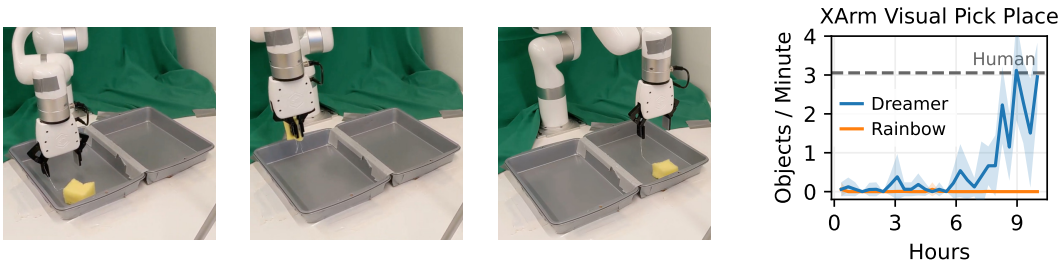


Figure 6: **XArm Visual Pick and Place** The XArm is an affordable robot arm that operates slower than the UR5. To demonstrate successful learning on this robot, we use a third-person RealSense camera with RGB and depth modalities, as well as proprioceptive inputs for the robot arm, requiring the world model to learn sensor fusion. The pick and place task uses a soft object. While soft objects would be challenging to model accurately in a simulator, Dreamer learns avoids this issue by directly learning on the real robot without simulator. While Rainbow (Hessel et al., 2018) converges to the local optimum of grasping and ungrasping the object in the same bin, Dreamer learns a successful pick and place policy from sparse rewards in under 10 hours.

generators that restrict the space of motions. In contrast, we train in the end-to-end reinforcement learning setting directly on the robot, without simulators or resets. We use the Unitree A1 robot that consists of twelve direct drive motors. The motors are controlled at 20 Hz via continuous actions that represent motor angles that are realized by a PD controller on the hardware. The state consists motor angles, orientations, and angular velocities. To protect the motors, we filter out high-frequency motor commands through a Butterworth filter. Due to space constraints, we manually intervene only when the robot has reached the end of the available training area, without modifying the joint configuration or orientation that the robot is in.

Reward The reward function is the sum of five terms. An upright reward is computed from the base frame up vector $\hat{z}^T$, terms for matching the standing pose are computed from the joint angles of the hips, shoulders, and knees, and a forward velocity term is computed from the projected forward velocity ${}^{\mathcal{B}}v^x$ and the total velocity ${}^{\mathcal{B}}v$. Each of the five terms is active while its preceding terms are satisfied to at least 0.7 and otherwise set to 0:

$$\begin{aligned} r^{\text{upr}} &\doteq (\hat{z}^T[0,0,1]\text{-}1)/2 \quad r^{\text{hip}} \doteq 1\text{-}\tfrac{1}{4}\|\, q^{\text{hip}} + 0.2\,\|_1 \quad r^{\text{shoulder}} \doteq 1\text{-}\tfrac{1}{4}\|\, q^{\text{shoulder}} + 0.2\,\|_1 \\ r^{\text{knee}} &\doteq 1\text{-}\tfrac{1}{4}\|\, q^{\text{knee}}\text{-}1.0\,\|_1 \quad r^{\text{velocity}} \doteq 5\big(\big(\max(0, {}^{\mathcal{B}}v_x)/\|{}^{\mathcal{B}}v\|_2\big)\,\text{clip}({}^{\mathcal{B}}v_x/0.3, \text{-}1, 1) + 1\big) \end{aligned} \tag{5}$$

Results As shown in Figure 4, after one hour of training, Dreamer learns to consistently learns flips the robot over from its back, stand up, and walk forward. Within 5 minutes of training, the robot learns to roll off its back and land on its feet. 20 minutes later, it learns how stand up on its feet. About an hour into training, the robot learns a pronking gait to walk forward at the desired velocity. After 1 hour of training, we tested the robustness of the learned recovery strategies by repeatedly knocking the robot off of its feet with a large pole, shown in Figure 8. Within 10 minutes, the robot learned to adapt and withstand pushes. We compare to SAC (Haarnoja et al., 2018) on this task and observe that SAC successfully learns to flip over from its back but is unable to stand up and walk forward.

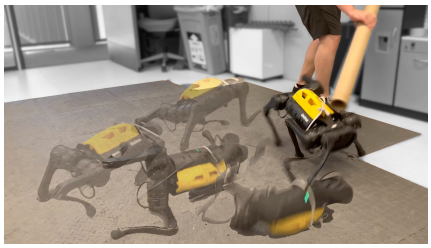
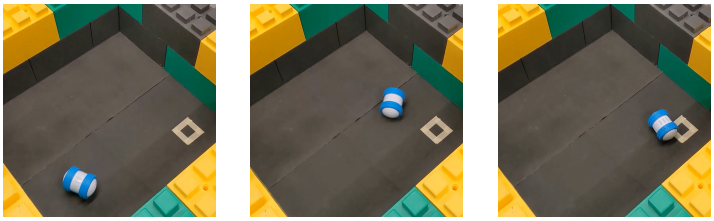


Figure 8: Within 10 minutes of pushing and perturbing the learned walking behavior, the robot adapts to withstanding the pushes or quickly rolling over and back on its feet.

3.2 UR5 Multi-Object Visual Pick and Place

Setup This task, representative of robotic tasks in warehouse and logistics environments, requires the robot manipulator to transport items from one bin into the other. Figure 5 highlights a successful pick and place cycle of this task. The difficulty of this task comes from the complex visual dynamics, as the robot can interact with multiple objects (3 for our experiments). In addition, lack of ground



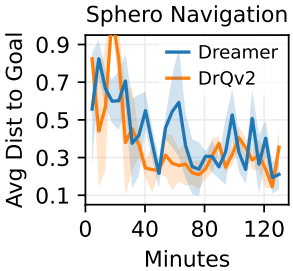


Figure 7: **Sphero Navigation** This task requires the Sphero Ollie robot to navigate two a fixed goal location through continuous actions and given a top-down RGB image as the only sensory input. The task requires the robot to localize itself from pixels without proprioceptive inputs, to infer its orientation from the sequence of past images because, which is ambiguous from a single image, and to control the robot from under-actuated motors that require building up momentum over time. Dreamer learns a successful policy on this task in under 2 hours.

truth object positions restricts the robot to sparse rewards and requires the robot to locate objects from raw pixel observations. We employ the UR5 from Universal Robotics, a high performance, collaborative robot, which we control at 2 Hz. The robot is provided with a sparse reward of 1 for every successful grasp on the object, -1 for releasing the object in the current bin, and 10 for placing the object in the correct bin. The robot is provided with a discrete action space, controlling delta Cartesian positions of the end effector position and toggling of the gripper state. The robot's observations consist of proprioceptive states (joint angles, gripper position, and end effector Cartesian pose) and a 3rd person RGB image of the scene.

Results Dreamer achieves an average pick rate of 2.5 objects per minute within 8 hours. The robot initially struggles to learn as the reward signal is very sparse, but begins to gradually improve after 2 hours of training. The robot first learns to localize the objects and toggles the gripper when near an object. Over time, the grasping becomes precise, and the robot develops strategies to push objects out of corners. We establish a human baseline by collecting 20 minutes of expert demonstrations from three separate demonstrators controlling the UR5 with a joystick. Figure 5 shows the learning curves of Dreamer compared to Rainbow DQN and a human baseline. Rainbow DQN converges to the suboptimal behavior of grasping and immediately dropping objects in the same bin. In contrast, Dreamer approaches human-level performance after 8 hours. We hypothesize that Rainbow DQN fails because of the small amount of data available when training directly in the real world.

3.3 XArm Visual Pick and Place

Setup This task, similar to that of the UR5, requires the XArm to move a single item back and forth between 2 bins, which is shown in Figure 6. The Ufactory XArm 5 is an accessible low cost 7 DOF manipulator, which we control at approximately 0.5 Hz. A thread connects the the object to the gripper. While this increases the likelyhood of the object from getting stuck in corners, it also increases the complexity of the dynamics. The sparse reward, discrete action space, and observation space are the same as with the UR5 except for the addition of depth observations.

Results Dreamer learns a policy that enables the XArm to achieve an average pick rate of 3.1 objects per minute in 10 hours of time, which is comparable to human performance on this task. Figure 6 shows the learning curves of Dreamer compared to the Rainbow baseline, where we observe a similar result as with the UR5. We found that Dreamer can adapt to changing lighting conditions effectively. A policy trained over night begins to lose performance as the morning light is cast onto the environment in the morning, but recovers performance after a few hours more of training.

3.4 Sphero Navigation

Setup Additionally we test Dreamer on a simple visual navigation task, requiring a wheeled robot to maneuver to a goal location using only raw images as input. We use the Sphero Ollie robot, a cylindrical robot with two controllable motors, which the robot controls with continuous

torque commands at 2 Hz. Because the robot is symmetric and the robot only has access to image observations, the robot must infer its heading from the history of observations. The robot is provided with a dense L2 reward equal to the negative L2 distance. As the goal is fixed, after 100 environment steps, we end the episode and randomize the robots position through a sequence of high power random motor actions.

Results After training Dreamer on the Sphero goal-conditioned navigation task for 2 hours, we find that the robot consistently navigates to the goal in a short amount of time and stay near the goal for the remainder of the episode. As shown in Figure 7, Dreamer achieves an average distance to the goal of roughly 0.15, which is defined as the average of the euclidean distance to the goal divided by the episode length (100 steps) and normalized by the arena length (80cm). Dreamer matches the performance of DrQv2, a specialized model-free algorithm that optimizes for learning on raw images for control tasks. Our results match the findings of Yarats et al. (2021) on simulated environments.

4 Related Work

In addition to the complex algorithmic challenges present in RL algorithms, deploying these algorithms on real world hardware provides additional systems challenges (Zhu et al., 2020). In locomotion, one common approach is to leverage large amounts of simulated data under domain and dynamics randomization, then deploying in the real world. (Lee et al., 2020; Siekmann et al., 2021; Escontrela et al., 2022; Miki et al., 2022; Rudin et al., 2021; Peng et al., 2018; Smith et al., 2021). One class of methods in manipulation attempts to scale the number of robots and experience a learner collects through a fleet of robots or massive offline datasets (Kalashnikov et al., 2018; 2021; Ebert et al., 2021; Dasari et al., 2019). Another common approach is to leverage expert demonstrations to make learning more sample efficient (James and Davison, 2021; James et al., 2021). These methods generally require some amount of domain specific knowledge of the robot or human involvement in the learning process, limiting the generality.

Model based RL, due to its higher sample efficiency over model free methods, is a common class of algorithms applied to learning on real world robots (Deisenroth et al., 2013). A model based algorithm first learns a dynamics model, which can then be used to plan actions, or be used as a simulator to learn a model free policy, (Nagabandi et al., 2019; Hafner et al., 2018; Yang et al., 2019; Hafner et al., 2019; 2020). One approach learns an action conditioned video prediction model (Finn and Levine, 2017; Ebert et al., 2018; Finn et al., 2016), but can suffer from computational inefficiency and drifting of the visual predictions. Few works have demonstrated a high performing model based algorithm that can generalize to multiple different robot mythologies without algorithmic changes.

5 Discussion

In this work, we demonstrate that modern world models provide a strong approach for training end-to-end RL policies directly in the real world. We apply Dreamer to four tasks across four robotic platforms, teaching a quadruped robot to stand up and walk from scratch in one hour, teaching a robot arm to accomplish pick and place by fusing pixel images and joint angles, and teaching a wheeled robot to learn a navigation purely from pixels.

Limitations While Dreamer shows promising results by learning directly on hardware, learning on hardware over many hours creates wear on robots that may require human intervention or repair. Additionally, more work is required to explore the limits of Dreamer and our baselines by training for a longer time. Finally, we see tackling more challenging tasks, potentially by combining the benefits of fast real world learning with those of simulators, as an impactful future research direction.

References

D. Hafner, T. Lillicrap, J. Ba, and M. Norouzi. Dream to control: Learning behaviors by latent imagination. *arXiv preprint arXiv:1912.01603*, 2019.

D. Hafner, T. Lillicrap, M. Norouzi, and J. Ba. Mastering atari with discrete world models. *arXiv preprint arXiv:2010.02193*, 2020.

Y. Gal, R. McAllister, and C. E. Rasmussen. Improving pilco with bayesian neural network dynamics models. In *Data-Efficient Machine Learning workshop, ICML*, 2016.

F. Ebert, C. Finn, S. Dasari, A. Xie, A. Lee, and S. Levine. Visual foresight: Model-based deep reinforcement learning for vision-based robotic control. *arXiv preprint arXiv:1812.00568*, 2018.

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*, pages 8583–8592. PMLR, 2020.

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*, 34: 28954–28967, 2021.

D. Hafner, T. Lillicrap, I. Fischer, R. Villegas, D. Ha, H. Lee, and J. Davidson. Learning latent dynamics for planning from pixels. *arXiv preprint arXiv:1811.04551*, 2018.

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

D. J. Rezende, S. Mohamed, and D. Wierstra. Stochastic backpropagation and approximate inference in deep generative models. *arXiv preprint arXiv:1401.4082*, 2014.

V. Makoviychuk, L. Wawrzyniak, Y. Guo, M. Lu, K. Storey, M. Macklin, D. Hoeller, N. Rudin, A. Allshire, A. Handa, et al. Isaac gym: High performance gpu-based physics simulation for robot learning. *arXiv preprint arXiv:2108.10470*, 2021.

R. S. Sutton and A. G. Barto. *Reinforcement learning: An introduction*. MIT press, 2018.

R. J. Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3-4):229–256, 1992.

M. Henaff, A. Canziani, and Y. LeCun. Model-predictive policy learning with uncertainty regularization for driving in dense traffic. *arXiv preprint arXiv:1901.02705*, 2019.

D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

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*, 518(7540):529, 2015.

M. Hessel, J. Modayil, H. Van Hasselt, T. Schaul, G. Ostrovski, W. Dabney, D. Horgan, B. Piot, M. Azar, and D. Silver. Rainbow: Combining improvements in deep reinforcement learning. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. *arXiv preprint arXiv:1801.01290*, 2018a.

T. Haarnoja, A. Zhou, K. Hartikainen, G. Tucker, S. Ha, J. Tan, V. Kumar, H. Zhu, A. Gupta, P. Abbeel, et al. Soft actor-critic algorithms and applications. *arXiv preprint arXiv:1812.05905*, 2018b.

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. *arXiv preprint arXiv:1911.08265*, 2019.

D. Yarats, R. Fergus, A. Lazaric, and L. Pinto. Mastering visual continuous control: Improved data-augmented reinforcement learning. *arXiv preprint arXiv:2107.09645*, 2021.

T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor, 2018. URL https://arxiv.org/abs/1801.01290.

H. Zhu, J. Yu, A. Gupta, D. Shah, K. Hartikainen, A. Singh, V. Kumar, and S. Levine. The ingredients of real-world robotic reinforcement learning, 2020.

J. Lee, J. Hwangbo, L. Wellhausen, V. Koltun, and M. Hutter. Learning quadrupedal locomotion over challenging terrain. *Science Robotics*, 5(47), oct 2020. doi:10.1126/scirobotics.abc5986. URL https://doi.org/10.1126%2Fscirobotics.abc5986.

J. Siekmann, K. Green, J. Warila, A. Fern, and J. Hurst. Blind bipedal stair traversal via sim-to-real reinforcement learning, 2021.

A. Escontrela, X. B. Peng, W. Yu, T. Zhang, A. Iscen, K. Goldberg, and P. Abbeel. Adversarial motion priors make good substitutes for complex reward functions, 2022.

T. Miki, J. Lee, J. Hwangbo, L. Wellhausen, V. Koltun, and M. Hutter. Learning robust perceptive locomotion for quadrupedal robots in the wild. *Science Robotics*, 7(62), jan 2022. doi:10.1126/scirobotics.abk2822.

N. Rudin, D. Hoeller, P. Reist, and M. Hutter. Learning to walk in minutes using massively parallel deep reinforcement learning, 2021.

X. B. Peng, M. Andrychowicz, W. Zaremba, and P. Abbeel. Sim-to-real transfer of robotic control with dynamics randomization. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1–8, May 2018. doi:10.1109/ICRA.2018.8460528.

L. Smith, J. C. Kew, X. B. Peng, S. Ha, J. Tan, and S. Levine. Legged robots that keep on learning: Fine-tuning locomotion policies in the real world, 2021.

D. Kalashnikov, A. Irpan, P. Pastor, J. Ibarz, A. Herzog, E. Jang, D. Quillen, E. Holly, M. Kalakrishnan, V. Vanhoucke, and S. Levine. Qt-opt: Scalable deep reinforcement learning for vision-based robotic manipulation, 2018.

D. Kalashnikov, J. Varley, Y. Chebotar, B. Swanson, R. Jonschkowski, C. Finn, S. Levine, and K. Hausman. Mt-opt: Continuous multi-task robotic reinforcement learning at scale, 2021.

F. Ebert, Y. Yang, K. Schmeckpeper, B. Bucher, G. Georgakis, K. Daniilidis, C. Finn, and S. Levine. Bridge data: Boosting generalization of robotic skills with cross-domain datasets, 2021.

S. Dasari, F. Ebert, S. Tian, S. Nair, B. Bucher, K. Schmeckpeper, S. Singh, S. Levine, and C. Finn. Robonet: Large-scale multi-robot learning, 2019.

S. James and A. J. Davison. Q-attention: Enabling efficient learning for vision-based robotic manipulation, 2021.

S. James, K. Wada, T. Laidlow, and A. J. Davison. Coarse-to-fine q-attention: Efficient learning for visual robotic manipulation via discretisation, 2021.

M. P. Deisenroth, G. Neumann, and J. Peters. 2013.

A. Nagabandi, K. Konoglie, S. Levine, and V. Kumar. Deep dynamics models for learning dexterous manipulation, 2019.

Y. Yang, K. Caluwaerts, A. Iscen, T. Zhang, J. Tan, and V. Sindhwani. Data efficient reinforcement learning for legged robots, 2019.

C. Finn and S. Levine. Deep visual foresight for planning robot motion. In *Robotics and Automation (ICRA), 2017 IEEE International Conference on*, pages 2786–2793. IEEE, 2017.

C. Finn, I. Goodfellow, and S. Levine. Unsupervised learning for physical interaction through video prediction. In *Advances in neural information processing systems*, pages 64–72, 2016.