Latent Policy Steering with Embodiment-Agnostic Pretrained World Models

Yiqi Wang
Carnegie Mellon University
Pittsburgh, PA 15213
yiqiw2@andrew.cmu.edu

Mrinal Verghese
Carnegie Mellon University
Pittsburgh, PA 15213
mverghes@andrew.cmu.edu

Jeff Schneider
Carnegie Mellon University
Pittsburgh, PA 15213
jeff4@andrew.cmu.edu

Abstract

Learning visuomotor policies via imitation has proven effective across a wide range of robotic domains. However, the performance of these policies is heavily dependent on the number of training demonstrations, which requires expensive data collection in the real world. In this work, we aim to reduce data collection efforts when learning visuomotor robot policies by leveraging existing or cost-effective data from a wide range of embodiments, such as public robot datasets and the datasets of humans playing with objects (human data from play). Our approach leverages two key insights. First, we use optic flow as an embodiment-agnostic action representation to train a World Model (WM) across multi-embodiment datasets, and finetune it on a small amount of robot data from the target embodiment. Second, we develop a method, Latent Policy Steering (LPS), to improve the output of a behavior-cloned policy by searching in the latent space of the WM for better action sequences. In real world experiments, we observe significant improvements in the performance of policies trained with a small amount of data (over 50% relative improvement with 30 demonstrations and over 20% relative improvement with 50 demonstrations) by combining the policy with a WM pretrained on two thousand episodes sampled from the existing Open X-embodiment dataset across different robots or a cost-effective human dataset from play.

1 Introduction

Imitation learning via Behavior Cloning (BC) is a widely adopted paradigm to acquire visuomotor policies for robots [4, 6, 35]. To achieve high task success, sufficient demonstrations must be collected, which is a time-consuming process. Furthermore, the collected data is often specific to a robot, a task, or an environment, and the data collection process may need to be repeated for different embodiments or different environments. Due to the recent progress in collecting large datasets [38] across different robots and environments, progress has been made to build generalist robot policies [39, 36] via cross-embodiment training. However, these models sometimes do not generalize to new tasks with satisfactory results. To fine-tune these models for better performance, a considerable amount of data is required, given the large model size used to learn from large datasets [4, 35, 3]. For example, $\pi 0$ [3] requires 5-10 extra hours of fine-tuning data to achieve high success rates on new tasks.

According to historical Machine Learning research, a model trained on large and diverse datasets across multiple tasks will produce general representations that are transferable to new tasks with

Preprint. Under review.

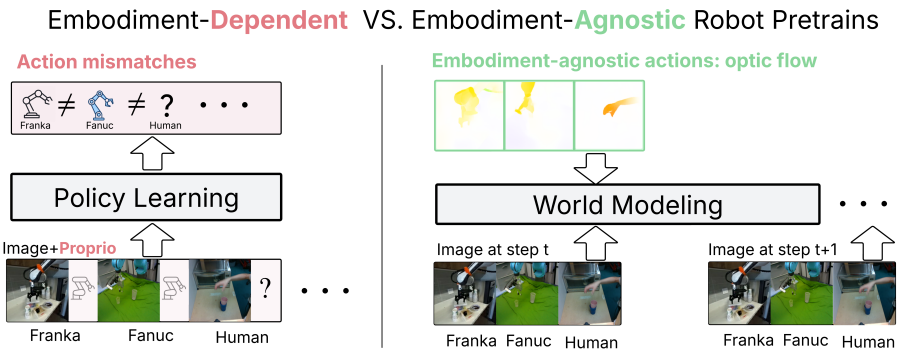

Figure 1: **Pretrained World Models (WM) with optic flows as an embodiment-agnostic action representation.** Using an embodiment-agnostic action space allows us to integrate data from multiple types of robots and even humans. Instead of doing direct policy learning, which is heavily dependent on action representations, we learn a world model to leverage diverse data sources.

only a small amount of data, known as pretraining [14, 16]. However, pretraining a robot model is challenging: each robot in the dataset has embodiment-specific information (proprioception, actions). Thus, a pretraining dataset with different embodiments makes the pretrained representation heavily depends on the embodiments included in the dataset and may generalize poorly to new embodiments.

To tackle this unique challenge, we choose to pretrain a robot World Model (WM) with an embodiment-agnostic action representation. Since a WM models task dynamics by predicting future states given the current state and the action, it is less dependent on proprioception and more embodiment-independent compared to a policy, illustrated in Figure 1. As such, it is better suited to pretraining from multiple robot embodiments. Since a WM depends on actions to model the task dynamics, we further remove dependency on embodiment from a WM by replacing robot actions with optic flow, which captures visual action in terms of motion. Since motions in visual space look similar across different embodiments, optic flow can serve as an embodiment-agnostic action representation for pretraining the WM across embodiments. During inference, such a WM can be combined with a policy using a technique called policy steering [31, 41, 40] to improve the performance of the policy.

We propose a technique called Latent Policy Steering (LPS) to improve a policy with the WM during inference, motivated by prior work on using a WM for planning [29, 46]. In that work, the authors simulate the action plan sampled from the policy with the WM by predicting future states and comparing them to a goal image in the latent space. Unfortunately, that technique becomes less effective when the task is long-horizon and the desired outcome (goal image) is beyond the reach of the policy and the WM. Our proposed technique is effective regardless of the task horizon by making one key observation: since every demonstration collected for Behavior Cloning is an expert demonstration that accomplishes the task, every state in the demonstration becomes a goal state. Thus, one can distill the state comparisons conducted during inference into a state-based value function on the WM. During inference, the value function steers a policy toward success by visiting states close to the dataset distribution.

Our contributions are the following:

- We pretrain a World Model leveraging optic flow as an embodiment-agnostic action representation, and fine-tune the World Model on a small dataset with robot actions.
- We propose Latent Policy Steering (LPS) during inference to improve the policy performance with a World Model. Based on the key observation that all states in the BC dataset are goal states, a value function is trained to steer the policy towards states closer to the goal states (every state from the dataset) rather than deviating from them.
- We demonstrate that the proposed method significantly improves a policy's performance both in simulation and the real world. This shows the effectiveness of leveraging a World Model for small-data scenarios with abundant task-relevant data from other embodiments.

2 Related works

2.1 Robot Learning from Diverse Data

Collecting large amounts of robot data for a specific task can be challenging and time-consuming. A common strategy to overcome this limitation is to leverage data from other sources, including online robot datasets with multiple types of robots and even human videos. Prior works [30, 17, 26, 33] train a visual encoder using both human and robot video data to avoid learning the policy network from scratch. However, the pretrained encoder only provides visual representations instead of directly learning to make decisions. Thus, to learn full policies across multiple robot embodiments, recent work has also looked at learning a single network across multiple robot embodiments by using a modular structure with a common trunk and various heads for different action spaces [39, 36]. Another common approach is the vision-language-action model (VLA) approach, where a pretrained VLM is fine-tuned with robot data to act as a robot policy [19, 5, 3, 15]. However, due to their large size, both these approaches can be expensive to fine-tune to improve performance on a specific task, and sometimes suffer from slow inference. We compare our approach against a fine-tuned HPT [39] as a representative example of these types of models.

Human video is also abundant, but can be challenging to use for robot policies as it lacks specific action information. Some works have attempted to overcome this by learning reusable priors or affordances from human video that can then accelerate robot learning [29, 2, 1]. However, these approaches tend to make specific assumptions about how humans interact with the world, and may not generalize to tasks that don't fit these assumptions.

2.2 Policy Steering & Planning with World Models

Recurrent-state-space-based world models (RSSM) have become an increasingly common approach to model environment dynamics and transition functions in robotics. Popularized by Hafner et. al [12, 13], these models consist of an encoder, which maps visual observations to a latent state representation, a latent transition function which predicts future latent states, and a decoder which is used primarily during training to propagate gradients. Once trained, these world models can be utilized in a variety of ways. One approach is to learn actor and critic models from the same training data to generate action sequences, roll them out over time, and evaluate them with the critic [13]. Another approach is to encode a goal image in the latent space and then use the similarity between planned states and the goal image as a value function to select the best sequence of actions. These action sequences are commonly optimized using gradient-free optimizers like the Cross-Entropy Method (CEM), but gradient-based optimization can also be used [13, 46, 29].

These approaches are closely related to policy steering, where a value function and optionally a world model are used to refine the output of a base policy [31, 41]. Policy steering has been shown to compensate for failure modes in the base policy [31] or to select actions from the base policy that obeys safety constraints [41]. Using a world model allows the robot to steer its policy based not just on the next action, but on a projected series of actions. We build on the ideas from both planning with World Models and Policy Steering by learning a value function to steer actions from a behavior cloned policy towards states within the data distribution, and closer to the task goal.

2.3 Offline & Inverse Reinforcement Learning

In this work, we propose a value function that favors states in the distribution of the training data. This approach is closely related to the pessimism used in offline RL [9, 43, 18, 18, 21] and the concept of Inverse RL [32]. In offline RL, a model (e.g., value functions, dynamics models) only has access to the state-action pairs in an offline dataset. Therefore, predictions for state-action pairs outside of the training data distribution could be unreliable. For example, a value function could be overly optimistic when queried with out-of-distribution state-action pairs, leading to extrapolation errors [9].

2.4 Optic Flow as a Representation in Robotics

Both 2D optic flow and 3D point flow have commonly been used as intermediate representations in robotics. Multiple works have used 3D flow to represent object affordances [8, 45, 44]. These representations transfer well from simulation to real hardware and can be learned from human videos.

Other works have predicted 2D optic flow as an action representation [23, 10] or used optic flow to measure similarity between actions across episodes [23, 37]. Rather than using optic flow as a predicted action representation, in this work, we use optic flow as a unified input action representation to our world model to encode actions across diverse embodiments.

3 Preliminaries

We consider robot control in a Partially Observable Markov Decision Process (POMDP) setting, with a latent state space learned by a neural network. A POMDP can be described by a tuple $(\Omega, \mathcal{X}, \mathcal{S}, \mathcal{A}, \mathcal{R}, \mathcal{P})$, which consists of observations $x \in \mathcal{X}$, conditional observation probabilities $\Omega(x'|s, a)$, states $s \in \mathcal{S}$, actions $a \in \mathcal{A}$, reward function $r = \mathcal{R}(s, a)$, and transition function $P(s'|s, a)$. At the timestep $t \in [1, 2, ..., T]$, the observation, state, action and reward are denoted as o_t, s_t, a_t, r_t. Given discount factor γ, the expected future discounted reward is defined as: $E[\sum_{t=0}^{\infty} \gamma^t r_t]$. A dataset consists of sequences of (x_t, a_t, r_t) where r_t is a binary reward indicating task successes. Given a horizon length h, policy steering seeks the best action plan $\mathbf{a}_t^* = [a_t^*, a_{t+1}^*, ..., a_{t+h}^*]$ such that: $\mathbf{a}_t^* = \text{argmax}_{\ a \in \mathcal{A}}\ \mathbb{E}_{s' \sim P(\cdot|s,a)} \mathcal{R}(s, a)$.

We assume a robot has access to a small dataset of demonstrations for a task on its embodiment (referred to as the target embodiment) and a larger dataset of demonstrations involving similar objects or for a similar task on non-target embodiments, such as other robots or humans.

4 Methods

In this section, we first explain our motivation for choosing optic flow as an embodiment-agnostic action representation for pretraining a WM. Then, we describe the proposed technique called Latent Policy Steering (LPS) to leverage a WM to improve the performance of a policy during inference.

4.1 Flow-as-Action: Embodiment-Agnostic World Modeling

A robot WM that can be easily adapted to a new embodiment should be independent of embodiment-specific information, such as proprioception and actions. In order for a WM to model task dynamics, it should use an embodiment-agnostic action representation rather than a raw robot action as input. Unlike prior work, such as HPT [39], which learns generalizable representations on the fly with end-to-end training, we propose to replace the raw action from different embodiments with an existing embodiment-agnostic action representation which is easily computed with the off-the-shelf tools: optic flow. One of our key observations is that different robots' visual motions have similar patterns when they execute similar skills (e.g, pick up an object) as shown in Figure 2. Therefore, we leverage optic flow as an embodiment-agnostic action representation for WM's pretraining.

We precompute optic flow for each episode in the pretraining dataset. A Convolution Neural Network (CNN) with a Multi-Layer Perceptron (MLP) is used to encode the optic flow and project the encoding to a vector with a similar dimension as a robot action (e.g., 7 for end-effector pose and gripper states).

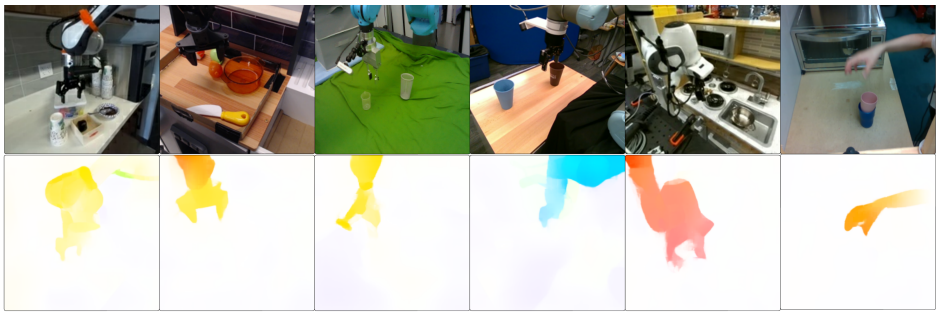

Figure 2: **Optic flow as an embodiment-agnostic action representation.** We observe that motions captured by optic flow across embodiments are similar in visual space. By using optic flow as an action representation, we remove WM's dependency on specific embodiments, allowing the pretrained model to efficiently leverage data from multiple embodiments.

During pretraining, this encoder is optimized with the WM fully end-to-end. When finetuning the WM for a target embodiment, the optic flow encoder is discarded, and the WM takes raw robot actions without any extra encoding.

4.2 Latent Policy Steering with a world model

We propose Latent Policy Steering (LPS) to improve the policy's performance by conducting a look-ahead search for the best action plan in the WM's latent space with a value function, summarized in Algorithm 1. One naive way to do policy steering is to choose the action that leads to the state with the highest reward. Since every demonstration in the dataset is successful, this can be done by labeling the last three steps with a reward of 1, and other steps with 0, similar to Kumar et. al [22]. Then, one could train a value function $\mathcal{V}(s)$ based on the WM to estimate the long-term discounted rewards for policy steering. However, in a small dataset, successful demonstrations only cover a limited set of states, and a policy may not visit the same set of states during inference due to distribution shifts or compounding errors. As a result, such a value function is not robust to outliers during inference.

Training LPS Our key observation is that every state in the BC dataset is a goal state, since every demonstration is a successful demonstration. LPS steers a policy to visit these "goal" states in the dataset and avoid actions that deviate from the dataset. This is achieved by simulating and comparing the expert states visited by the actions from the dataset and non-expert states visited by the predicted actions by querying the policy (lines 7-10, Algorithm 1). All non-expert states that deviate from the corresponding expert states are penalized by converting the cosine similarity between $s_{t:t+h}$ and $s'_{t:t+h}$ into a negative reward (line 10, Algorithm 1). We train a value function with the objective called Lambda-Return [34], to encode the following: 1) the value corresponds to successful task completion based on the binary reward, and 2) the value corresponds to states that deviate from the dataset based on the extra rewards. No gradients from the value function backpropagate to the WM.

We query the behavior cloned diffusion policy during the value function's training. As such $\mathcal{V}(s)$ is trained on the states that are likely to appear during inference. This allows LPS to be robust to outliers during inference, and its effectiveness is proven in the ablation study.

Algorithm 1 finetuning a WM for Latent Policy Steering.

1: **Requires:** Robot dataset $D = [(x_{1:T}, a_{1:T}, r_{1:T})_n | n = [1, 2, ..., N]]$ with observations, actions, and binary rewards, pretrained world model $\mathcal{W}$ and pretrained policy π.
2: **Initialize:** value function $\mathcal{V}$, horizon h, discount γ, and λ for return computation.
3: # finetune the $\mathcal{W}$ with robot actions and optimize the value function $\mathcal{V}$ for policy steering.
4: **while** not converged **do**
5: Samples $(x_{t:t+h}, a_{t:t+h}, r_{t:t+h})$ from D.
6: Optimize the $\mathcal{W}$ with $(x_{t:t+h}, a_{t:t+h})$. ▷ finetunes the WM with robot actions.
7: Sample predicted action plan: $a'_{t:t+h} \sim \pi(x_t)$.
8: $s_t = \mathcal{W}_{\text{frozen}}.\text{encode}(x_t)$ ▷ observations to latent states.
9: $s_{t:t+h}, s'_{t:t+h} = \mathcal{W}_{\text{frozen}}(s_t, a_{t:t+h}), W_{\text{frozen}}(s_t, a'_{t:t+h})$ ▷ unroll the $\mathcal{W}$ with actions.
10: $r'_{t:t+h} = r_{t:t+h} + (\text{cosine}(s_{t:t+h}, s'_{t:t+h}) - 1)/2$ ▷ penalize deviation from the dataset.
11: $R_\lambda, R'_\lambda = \text{lambda\_return}(r_{t:t+h}, \gamma, \lambda), \text{lambda\_return}(r'_{t:t+h}, \gamma, \lambda)$
12: $R_{\text{all}}, S_{\text{all}} \leftarrow \text{concat}(R_\lambda, R'_\lambda), \text{concat}(s_{t:t+h}, s'_{t:t+h})$
13: Optimize the value function $\mathcal{V}$ with the loss: $\mathcal{L}(R_{\text{all}}, V(S_{\text{all}}))$.
14: **end while**

Inference with LPS A large batch of action plans is sampled from the BC policy during inference. After encoding the current observation, the WM predicts the corresponding future states for each plan and computes the state values. A value is assigned to each plan based on a weighted average of state values with a discount factor, by assigning heavier weights to future states. The action plan with the highest value is executed in open-loop. In practice, we found that this discounted weighted averaging of values works better than using the last state value per plan to represent the plan-level value.

Table 1: Latent Policy Steering with WM in Robomimic

Task\Settings	BC	IQL	WM-goal	WM-V(s) (Ours)
Lift	99.1 ± 0.4	$\mathbf{99.3 \pm 0.3}$	99.1 ± 0.6	$\mathbf{99.3 \pm 0.9}$
Can	79.8 ± 1.6	77.3 ± 0.3	84.2 ± 3.6	$\mathbf{87.3 \pm 3.9}$
Square	45.6 ± 1.1	49.2 ± 9.4	50.3 ± 5.3	$\mathbf{52.9 \pm 9.8}$
Transport	30.4 ± 32.9	25.6 ± 23.6	26.4 ± 36.3	$\mathbf{34.6 \pm 21.6}$
Average	63.7%	62.6%	65.0%	**68.5%**

We report the mean success rate across 3 seeds with variance. Our proposed idea WM-V(s) outperforms all baselines on average across 4 tasks.

5 Experiments

The proposed method is evaluated thoroughly both in a simulated benchmark and a real-world setting. In the sections below, we first evaluate our method with other baselines in Robomimic [28] to understand how much Latent Policy Steering (LPS) with a WM improves the policy performance. Then, we perform two ablation studies in Robomimic [28] to understand how well LPS responds to different action plan horizons, and to evaluate our design decisions. Finally, we compare LPS with pre-trained WMs and other baselines in the simulation as well as in the real world.

Simulation settings: For each task, we use either 30, 50, or 100 demonstrations from a Franka robot. We choose the following tasks from the Robomimic [28] benchmark for evaluations. **Lift**: lift a block. **Can**: pick up a can and place it into a bin with other bins for distractions. **Square** : pick a square un-threaded nut and place it onto a fixed peg with precision. **Transport**: A challenging 2-arm long-horizon task involves 1) 1 arm uncovers a box and hands a hammer to another arm, 2) the second arm empties the destination box by removing a red block, and 3) the second arm receives the hammer from the first arm and places it into a box.

Baselines: **BC**: a behavior cloning policy implemented as diffusion policy [6], with an action prediction horizon of 16. During inference, 16 actions are executed in an open-loop manner. **IQL**: a state-action value function trained by predicting discounted binary rewards using offline Reinforcement Learning [20]. During inference, it will score action plans proposed by the BC baseline and select the action plan with the highest score, similar to [31]. **WM-goal**: similar to [29, 46], the WM predicts the future states visited by the action plans sampled from the BC policy, and the best plan is determined by the similarity between the last latent state and the goal image encoding. The WM is implemented as in DreamerV3 [13]. Goal images represent the last visual observation of the demonstration that is closest to the initial observation during evaluation. **WM-V(s)** (Ours): This is the proposed method using a WM implemented by DreamerV3 [13], trained with Franka demonstrations. During inference, it simulates the states visited by the action plan sampled from the BC baseline and executes the best plan according to the score from $V(s_t)$. To obtain the best performance, we use a weighted average score across the future states to compute the plan-level score with a discount factor of 0.9. Both the **WM-goal** and **WM-V(s)** operate on images from a single fixed camera.

Evaluation protocol in simulation: We report the average success rate across three seeds with variance. For each seed, we run evaluations across 10 environments in parallel, until a maximum total interaction limit is reached. The resulting evaluation episodes per seed are usually over 150 episodes. Any non-fixed object is re-initialized with a different location and orientation each episode. Episode success is determined by the ground truth binary rewards given by the environment.

5.1 Does LPS improve the policy's performance with sufficient data?

We evaluate LPS with other baselines to understand how effective LPS is with a WM given sufficient data (100 demonstrations).

Results: We found the LPS with a WM consistently improves the policy performance across four tasks, and outperforms all baselines (Table 1). We find IQL cannot steer the base policy as effectively, likely due to the distribution shifts. The set of state-action pairs visited by the policy during inference is not the same set used to train the IQL value function. Although our proposed method also trains a

value function, the value function is trained on both the states from the datasets and the states likely to be visited by the policy during inference. Thus, it is more robust to the distribution shift during inference. Since most robotics tasks have a task horizon much greater than the WM's horizon (e.g., 300 steps for a task vs. 16 for WM), we also find that a goal image from the end of the demonstration is ineffective for policy steering during most timesteps.

5.2 How does Latent Policy Steering respond to different horizons?

We study the effectiveness of LPS given different horizon lengths: h=[2, 4, 8, 12, 16, 20, 24].

Results: We found the LPS with WM performs better than BC at horizons 4,8,16,20, and performs worse than BC for a horizon of 24 (Figure 3). Since the extra rewards that capture deviations from the dataset are based on a scaled state similarity (line 10, Algorithm 1), the reward scale will change greatly if the states are very dissimilar. Since the BC with horizon=24 makes accurately predicting actions challenging, we observe that the states visited by the BC are generally further away from the expert states compared to the shorter horizons. This leads to noisy rewards during training the value function, and a noisy value function which is less helpful during policy steering.

5.3 How do the designed rewards affect the performance of Latent Policy Steering?

We create two variants of LPS to understand which designs are crucial for LPS's performance. **WM-V(s)-binary**: removes the non-expert states $s'_{t:t+h}$, rewards that captures deviations from the dataset $r'_{t:t+h}$ from the proposed LPS (i.e., $R_{\text{all}}, S_{\text{all}} = R_\lambda, s_{t:t+h}$ at line12, Algorithm 1). The value function is only trained with the binary rewards and states from the dataset. **WM-V(s)-bootstrap**: removes the reward that captures deviations from the dataset $r'_{t:t+h}$ from the proposed LPS but still has the non-expert states $s'_{t:t+h}$ (i.e., $r' = r$ from line 10, Algorithm 1) labeled with binary rewards. The value function is trained with the binary rewards, states from the dataset, and simulated states from the WM.

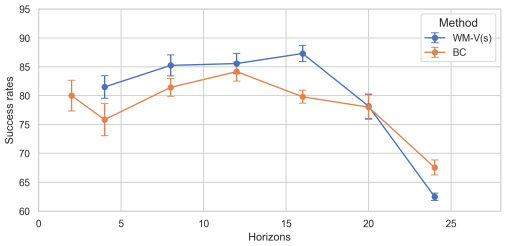

Figure 3: **Latent Policy Steering with different horizons.** We found the proposed LPS can effectively improve policy performance from a shorter horizon to a longer horizon up to 16. When the horizon becomes very long (e.g., 20, 24), the reward based on state similarity becomes noisy, making the value function less useful during inference.

Results: Both variants perform much worse than WM-V(s), suggesting that both components in the LPS value function are necessary for good performance (Table 2). WM-V(s)-binary doesn't train the value function V(s) with non-expert states visited by the policy, and loses robustness to distribution shifts and compounding errors during inference. While WM-V(s)-bootstrap does train the value function on non-expert states visited by the policy, similar to WM-V(s), it naively labels the non-expert with binary rewards as the expert states (changing line 10 with $r' = r$, Algorithm 1). Training the value function with extra states labeled with unrealistic rewards (assuming they're states that lead to success) makes the value function over-optimistic and becomes less useful during inference.

Table 2: Latent Policy Steering with ablations on rewards

Task\Settings	BC	WM-V(s)-binary	WM-V(s)-bootstrap	WM-V(s)
Lift	99.1 ± 0.4	96.0 ± 4.1	98.7 ± 0.3	**99.3 ± 0.9**
Can	79.8 ± 1.6	82.2 ± 4.8	80.2 ± 4.3	**87.3 ± 3.9**
Square	45.6 ± 1.1	52.0 ± 7.1	48.5 ± 18.8	**52.9 ± 9.8**
Transport	30.4 ± 32.9	20.4 ± 1.4	22.0 ± 2.1	**34.6 ± 21.6**
Average	63.7%	62.6%	62.3%	**68.5%**

We report the mean success rate over 3 seeds with variance. Taking away reward capture behaviors that deviate from the dataset (WM-V(s)-binary/bootstrap) significantly reduces the performance of the proposed method WM-V(s).

Table 3: Latent Policy Steering with pretrained WM in Robomimic

	30 Franka demonstrations			50 Franka demonstsrations		
Task\Settings	BC	WM-V(s)	WM-V(s)-pretrain	BC	WM-V(s)	WM-V(s)-pretrain
Lift	**25.6 ± 3.6**	18.0 ± 5.6	24.7 ± 1.4	82.0 ± 6.2	52.7 ± 5.0	**84.4 ± 10.8**
Can	64.9 ± 10.2	**74.7 ± 3.1**	71.1 ± 1.0	76.7 ± 2.1	80.2 ± 4.9	**84.9 ± 3.6**
Square	19.4 ± 2.5	20.6 ± 3.6	**24.2 ± 5.6**	44.8 ± 6.5	48.3 ± 19.7	**52.4 ± 7.2**
Average	36.6%	37.8%	**40.0%**	67.8%	60.0%	**73.9%**

We report the mean success rate across 3 seeds with variance. Given low-data scenarios (30 or 50 demonstrations), the proposed method can greatly benefit from a WM pretrained with optic flow.

5.4 Does the pretrained WM improve the LPS's performance in low-data scenarios?

5.4.1 Simulation experiment

For each task, we use either 30 or 50 demonstrations from the Franka robot data. The rest of the settings are the same as in Section 5.1. A pretraining dataset with 400 demonstrations is collected via teleoperation across four different robots: UR5e, Sawyer, IIWA, and Kinova3 from Robosuite [47]. Robot actions are replaced by the optic flow computed via GMFlow [42].

Compared to WM-V(s), WM-V(s)-pretrain is first pretrained with optic flows given the multi-embodiment dataset across 4 robots. The optic flow encoder is discarded during fine-tuning and replaced with robot actions from Franka demonstrations.

Results: Given a small number of demonstrations (30 or 50), LPS can greatly benefit from a pretrained WM in a low-data scenario (Table 3). The Lift task, for instance, has demonstrations that are at least 50% shorter than those from other tasks. Thus, 30 or 50 demonstrations have an insufficient amount of transitions to train a WM, and WM-V(s) performs much worse than the BC baseline for the Lift Task. However, with the help of a pretrained WM, the WM-V(s)-pretrain performs much better than WM-V(s) without pretraining.

5.4.2 Real world experiment

Settings: Each task has 50 teleoperated demonstrations using the Franka robot data with a wrist and front camera. We use the FastUMI Fingertips [24]. The OpenX pretrain dataset includes 2000 demonstrations sampled from the Open X-Embodiment dataset [38] across four different robots, involving skills and objects relevant to the tasks. Another pretrain dataset is collected by a human manually playing with objects without a particular goal (known as data-from-play [25]). Different from prior work [25], our human-data-from-play is very cheap to collect since humans are experts in object manipulation. In total, we collect video of a human interacting with the real world environment for 1 hour and trim it into segments. For both pretrain datasets, optic flow is computed via GMFlow [42] to serve as the action representation.

Figure 4: **Real world experimental setup.** The image shows the robot, camera, and objects.

We evaluate the method with the following tasks: **Put-radish-in-pot**: The robot picks up a toy radish and places it in a pot. **Stacking cups**: The robot inserts the pink cup into the larger blue cup. **Open-oven-put-pot-in-close-oven**: The robot opens the oven door, places a pot inside, and closes the oven. The pot will have a radish in it, so the robot has to adjust the grasp to avoid grasping the radish in the pot. We also use a slippery pot to make sure the task is challenging. An overview of the real-world experiment is shown in Figure 4.

Baselines: **HPT**: A behavior cloning policy with cross-embodiment pretraining on large-scale datasets [39]. The transformer trunk is frozen, and the embodiment-specific stem and prediction

Table 4: Latent Policy Steering with the pretrained WM in the real world

	30 Franka demonstrations					50 Franka demonstrations				
	From scratch		pretrain-and-finetune			From scratch		pretrain-and-finetune		
Task \ Settings	BC	WM-V(s)	HPT-large	WM-V(s)-OpenX	WM-V(s)-human	BC	WM-V(s)	HPT-large	WM-V(s)-OpenX	WM-V(s)-human
put-radish-in-pot	14/20	15/20	5/20	16/20	**17/20**	16/20	18/20	10/20	**20/20**	**20/20**
Stack-cups	10/20	14/20	2/20	14/20	**16/20**	14/20	14/20	3/20	15/20	**17/20**
Open-oven-put-pot-in-close-oven	2/20	6/20	0/20	**9/20**	8/20	8/20	11/20	0/20	**12/20**	11/20
Average	43.3%	58.3%	11.7%	65.0%	**68.3%**	63.3%	71.7%	21.7%	78.3%	**80.0%**

We report the success trials over 20 trials in the real world with the same random seed used for model training. Across 3 tasks, the proposed method with the pretrained WM (WM-V(s)-openx/human) outperforms baselines with or without pretraining.

head are finetuned on the Franka demonstrations. We choose a diffusion-based prediction head, similar to the BC baseline above. The action prediction horizon is also 16. **WM-V(s)**: the proposed method similar to Section 5.1. The WM is trained on Franka demonstrations with robot actions. No pretraining is used to initialize the WM. **WM-V(s)-OpenX**: A WM is first pretrained with optic flow given the OpenX pretrain dataset across four robots. The optic flow encoder is replaced by robot actions during finetuning with the Franka demonstrations. **WM-V(s)-human**: A WM is first pretrained with optic flow, given a human dataset from 1 hour of play. The optic flow encoder is replaced by robot actions during finetuning with the Franka demonstrations.

Evaluation protocols: The non-fixed object (i.e., radish, pink cup, pot) is randomly initialized in a rectangle area on a table. The middle 80% of the desk area is used to collect data, while the rest 20% desk area is reserved for evaluations. We report the success rate of 20 trials.

Results: We observe significant improvement from BC to LPS with a WM pretrained with OpenX data and human data from play, shown in Table 4. Given that the scene in the real world is more complicated than the scene in the simulation [27], we expect that pretraining can improve LPS's performance (WM-V(s)-OpenX/human) compared to a WM training from scratch (WM-V(s)). It is surprising that WM pretrained with human data from play performs better than the WM pretrained with OpenX robot data. We believe the reason is likely due to the distribution gaps between human-data-from-play to task data being smaller than OpenX data to task data, because the human-data-from-play is collected in the same desk environment used for robot data collection.

The WM-V(s)-OpenX/human both outperform the pretrained HPT-large, finetuned on the dataset. HPT-large is pretrained as an embodiment-dependent policy by encoding proprioception and predicting actions and its head and stem have to be fine-tuned for a new embodiment. In a low-data scenario, it performs poorly and tends to overfit the dataset. A WM pretrained with optic flow as embodiment-agnostic action representation adapts more easily to the target embodiment. In addition, a policy such as HPT [39] can suffer from compounding errors during inference, while LPS with a WM makes a policy less vulnerable to compounding errors during inference by selecting the plan that is successful and staying close to the data set through the value function.

6 Discussion & Limitations

We pretrain an embodiment-agnostic World Model (WM) on existing data sources such as public multi-embodiment robot datasets [38] and cost-effective human data from play to improve visuomotor policies with a small amount of real-world data. Thanks to the optic flow as an embodiment-agnostic action representation, the pretrained WM has minimal dependency on specific embodiments and can be easily fine-tuned to a target embodiment for deployment with a small amount of data. Through real-world evaluations, our proposed method improves a policy's relative performance during inference over 50% when using 30 demonstrations and 20% when using 50 demonstrations, demonstrating its effectiveness over prior work that pretrains a more embodiment-dependent policy. The code for pretraining and finetuning the WM for LPS will be made publicly available.

For the dataset used to pretrain the WM, we assumed that all visual observations were filmed through a single fixed camera such that the corresponding optic flow captures meaningful visual action of the agent. Given videos filmed with unfixed cameras, optic flow can contain other signals and may capture less visual motion related to the agent. Given large-scale datasets with non-static views, such as Epic-Kitchen[7] or Ego4d [11] involve humans interacting with diverse objects in various scenes, a robot model could greatly benefit from these data collected in the real world. Thus, in future work, we hope to explore more scalable action representations that are embodiment-agnostic and compatible with large-scale unstructured data in the wild.

References

[1] Shikhar Bahl, Abhinav Gupta, and Deepak Pathak. *Human-to-Robot Imitation in the Wild*. arXiv:2207.09450 [cs, eess]. July 2022. DOI: 10.48550/arXiv.2207.09450. URL: http://arxiv.org/abs/2207.09450 (visited on 06/04/2024).

[2] Shikhar Bahl et al. *Affordances from Human Videos as a Versatile Representation for Robotics*. arXiv:2304.08488 [cs]. Apr. 2023. DOI: 10.48550/arXiv.2304.08488. URL: http://arxiv.org/abs/2304.08488 (visited on 06/04/2024).

[3] Kevin Black et al. "pi0: A vision-language-action flow model for general robot control, 2024". In: *URL https://arxiv. org/abs/2410.24164* ().

[4] Anthony Brohan et al. "Rt-1: Robotics transformer for real-world control at scale". In: *arXiv preprint arXiv:2212.06817* (2022).

[5] Anthony Brohan et al. "Rt-2: Vision-language-action models transfer web knowledge to robotic control". In: *arXiv preprint arXiv:2307.15818* (2023).

[6] Cheng Chi et al. "Diffusion policy: Visuomotor policy learning via action diffusion". In: *The International Journal of Robotics Research* (2023), p. 02783649241273668.

[7] Dima Damen et al. "Scaling egocentric vision: The epic-kitchens dataset". In: *Proceedings of the European conference on computer vision (ECCV)*. 2018, pp. 720–736.

[8] Ben Eisner, Harry Zhang, and David Held. *FlowBot3D: Learning 3D Articulation Flow to Manipulate Articulated Objects*. arXiv:2205.04382 [cs]. May 2024. DOI: 10.48550/arXiv.2205.04382. URL: http://arxiv.org/abs/2205.04382 (visited on 05/13/2025).

[9] Scott Fujimoto, David Meger, and Doina Precup. "Off-policy deep reinforcement learning without exploration". In: *International conference on machine learning*. PMLR. 2019, pp. 2052–2062.

[10] Ankit Goyal et al. *IFOR: Iterative Flow Minimization for Robotic Object Rearrangement*. arXiv:2202.00732 [cs]. Feb. 2022. DOI: 10.48550/arXiv.2202.00732. URL: http://arxiv.org/abs/2202.00732 (visited on 09/13/2024).

[11] Kristen Grauman 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*. 2022, pp. 18995–19012.

[12] Danijar Hafner et al. "Learning latent dynamics for planning from pixels". In: *International conference on machine learning*. PMLR. 2019, pp. 2555–2565.

[13] Danijar Hafner et al. "Mastering diverse domains through world models". In: *arXiv preprint arXiv:2301.04104* (2023).

[14] Tom Henighan et al. "Scaling laws for autoregressive generative modeling". In: *arXiv preprint arXiv:2010.14701* (2020).

[15] Physical Intelligence et al. "pi0.5: a Vision-Language-Action Model with Open-World Generalization". In: *arXiv preprint arXiv:2504.16054* (2025).

[16] Jared Kaplan et al. "Scaling laws for neural language models". In: *arXiv preprint arXiv:2001.08361* (2020).

[17] Siddharth Karamcheti et al. *Language-Driven Representation Learning for Robotics*. arXiv:2302.12766 [cs]. Feb. 2023. DOI: 10.48550/arXiv.2302.12766. URL: http://arxiv.org/abs/2302.12766 (visited on 03/04/2025).

[18] Rahul Kidambi et al. "Morel: Model-based offline reinforcement learning". In: *Advances in neural information processing systems* 33 (2020), pp. 21810–21823.

[19] Moo Jin Kim et al. *OpenVLA: An Open-Source Vision-Language-Action Model*. arXiv:2406.09246 [cs]. Sept. 2024. DOI: 10.48550/arXiv.2406.09246. URL: http://arxiv.org/abs/2406.09246 (visited on 05/13/2025).

[20] Ilya Kostrikov, Ashvin Nair, and Sergey Levine. "Offline reinforcement learning with implicit q-learning". In: *arXiv preprint arXiv:2110.06169* (2021).

[21] Aviral Kumar et al. "Conservative q-learning for offline reinforcement learning". In: *Advances in neural information processing systems* 33 (2020), pp. 1179–1191.

[22] Aviral Kumar et al. "Pre-training for robots: Offline rl enables learning new tasks from a handful of trials". In: *arXiv preprint arXiv:2210.05178* (2022).

[23] Li-Heng Lin et al. *FlowRetrieval: Flow-Guided Data Retrieval for Few-Shot Imitation Learning*. arXiv:2408.16944 [cs]. Aug. 2024. DOI: 10.48550/arXiv.2408.16944. URL: http://arxiv.org/abs/2408.16944 (visited on 09/13/2024).

[24] Kehui Liu et al. "FastUMI: A Scalable and Hardware-Independent Universal Manipulation Interface with Dataset". In: *arXiv e-prints* (2024), arXiv–2409.

[25] Corey Lynch et al. "Learning latent plans from play". In: *Conference on robot learning*. Pmlr. 2020, pp. 1113–1132.

[26] Arjun Majumdar et al. *Where are we in the search for an Artificial Visual Cortex for Embodied Intelligence?* arXiv:2303.18240 [cs]. Feb. 2024. DOI: 10.48550/arXiv.2303.18240. URL: http://arxiv.org/abs/2303.18240 (visited on 09/13/2024).

[27] Ajay Mandlekar et al. "What Matters in Learning from Offline Human Demonstrations for Robot Manipulation". In: *arXiv preprint arXiv:2108.03298*. 2021.

[28] Ajay Mandlekar et al. "What matters in learning from offline human demonstrations for robot manipulation". In: *arXiv preprint arXiv:2108.03298* (2021).

[29] Russell Mendonca, Shikhar Bahl, and Deepak Pathak. "Structured world models from human videos". In: *arXiv preprint arXiv:2308.10901* (2023).

[30] Suraj Nair et al. "R3m: A universal visual representation for robot manipulation". In: *arXiv preprint arXiv:2203.12601* (2022).

[31] Mitsuhiko Nakamoto et al. "Steering your generalists: Improving robotic foundation models via value guidance". In: *arXiv preprint arXiv:2410.13816* (2024).

[32] Andrew Y Ng, Stuart Russell, et al. "Algorithms for inverse reinforcement learning." In: *Icml*. Vol. 1. 2. 2000, p. 2.

[33] Ilija Radosavovic et al. "Real-world robot learning with masked visual pre-training". In: *Conference on Robot Learning*. PMLR. 2023, pp. 416–426.

[34] John Schulman et al. "High-dimensional continuous control using generalized advantage estimation". In: *arXiv preprint arXiv:1506.02438* (2015).

[35] Nur Muhammad Shafiullah et al. "Behavior transformers: Cloning k modes with one stone". In: *Advances in neural information processing systems* 35 (2022), pp. 22955–22968.

[36] Octo Model Team et al. "Octo: An open-source generalist robot policy". In: *arXiv preprint arXiv:2405.12213* (2024).

[37] Mrinal Verghese and Christopher Atkeson. *Skills Made to Order: Efficient Acquisition of Robot Cooking Skills Guided by Multiple Forms of Internet Data*. arXiv:2409.15172 [cs]. Sept. 2024. DOI: 10.48550/arXiv.2409.15172. URL: http://arxiv.org/abs/2409.15172 (visited on 05/13/2025).

[38] Quan Vuong et al. "Open x-embodiment: Robotic learning datasets and rt-x models". In: *Towards Generalist Robots: Learning Paradigms for Scalable Skill Acquisition@ CoRL2023*. 2023.

[39] Lirui Wang et al. "Scaling proprioceptive-visual learning with heterogeneous pre-trained transformers". In: *Advances in Neural Information Processing Systems* 37 (2024), pp. 124420–124450.

[40] Yanwei Wang et al. "Inference-Time Policy Steering through Human Interactions". In: *arXiv preprint arXiv:2411.16627* (2024).

[41] Yilin Wu et al. "From Foresight to Forethought: VLM-In-the-Loop Policy Steering via Latent Alignment". In: *arXiv preprint arXiv:2502.01828* (2025).

[42] Haofei Xu et al. "GMFlow: Learning Optical Flow via Global Matching". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2022, pp. 8121–8130.

[43] Tianhe Yu et al. “Mopo: Model-based offline policy optimization”. In: *Advances in Neural Information Processing Systems* 33 (2020), pp. 14129–14142.

[44] Chengbo Yuan et al. *General Flow as Foundation Affordance for Scalable Robot Learning*. arXiv:2401.11439 [cs]. Sept. 2024. DOI: 10.48550/arXiv.2401.11439. URL: http://arxiv.org/abs/2401.11439 (visited on 05/13/2025).

[45] Harry Zhang, Ben Eisner, and David Held. *FlowBot++: Learning Generalized Articulated Objects Manipulation via Articulation Projection*. arXiv:2306.12893 [cs]. May 2024. DOI: 10.48550/arXiv.2306.12893. URL: http://arxiv.org/abs/2306.12893 (visited on 05/13/2025).

[46] Gaoyue Zhou et al. “Dino-wm: World models on pre-trained visual features enable zero-shot planning”. In: *arXiv preprint arXiv:2411.04983* (2024).

[47] Yuke Zhu et al. “robosuite: A Modular Simulation Framework and Benchmark for Robot Learning”. In: *arXiv preprint arXiv:2009.12293*. 2020.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The claims made in the abstract and introduction are described carefully with text and algorithms in the Sections 4.1 and 4.2. The claims made in the abstract and introduction matches with the experiment results both in simulation and in the real world. Detailed experiment designs and results can be found in the Section 5.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: The limitations and the potential future direction is described in the Section 6.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [NA]

Justification: We have empirical ablation experiments to understand the proposed method. We don't have any theory assumptions or proofs.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: Experiment settings and evaluation protocols are described in the Section 5. Detail instructions and code are in the supplementary materials for reproducibility.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: Code is provided in the supplementary materials, and will be shared via GitHub after publications. All data collected via teleoperation in the simulation and the real world will be shared.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

Answer: [Yes]

Justification: The evaluation details can be found in the Section 5. Training details can be found in the supplementaries.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [Yes]

Justification: For all the experiments in the simulation, the average success rate with variance across 3 seeds is reported, which each seed has at least a hundred of episodes. In the real world, we do not report variance across multiple seeds since the real world environment' randomness cannot be controlled by a seed. We use a consistent numpy and torch seed during training and evaluations for the model during real world evaluations.

Guidelines:

- The answer NA means that the paper does not include experiments.

- The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: We share the computation resources in the supplementary result.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: We strongly follow the code of ethics in this work.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [NA]

    Justification: This is not applicable since this work introduce a new algorithm for pretraining a model and improving its inference performance. This work has no direct societal impacts associated with it.

Guidelines:

- The answer NA means that there is no societal impact of the work performed.
- If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: The released model is trained on public data with public licenses and do no contain any sensitive or risky content.

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: The model is trained on public data with public licenses. For the released code and model, they are free to use by most audience for the purpose of advancing the field. The code and the released models will have MIT license.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.

- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: The provided code are well documented and detailed instruction for used the pretrained model is provided with the code.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: This work does not include crowdsourcing experiments with human subjects. The human video data from play and the robot dataset are collected by the authors.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: No human subjects with crowsourcing experiments are involved in this work.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

    Answer: [NA]

    Justification: The core method development in this work does not involve LLMs as any important, original, or non-standard components.

    Guidelines:

    - The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy (`https://neurips.cc/Conferences/2025/LLM`) for what should or should not be described.