2023 IEEE International Conference on Robotics and Automation (ICRA 2023)
May 29 - June 2, 2023. London, UK

Efficient Preference-Based Reinforcement Learning Using Learned Dynamics Models

Yi Liu[1], Gaurav Datta[1], Ellen Novoseller[2], Daniel S. Brown[3]

***Abstract*— Preference-based reinforcement learning (PbRL) can enable robots to learn to perform tasks based on an individual's preferences without requiring a hand-crafted reward function. However, existing approaches either assume access to a high-fidelity simulator or analytic model or take a model-free approach that requires extensive, possibly unsafe online environment interactions. In this paper, we study the benefits and challenges of using a *learned dynamics model* when performing PbRL. In particular, we provide evidence that a learned dynamics model offers the following benefits when performing PbRL: (1) preference elicitation and policy optimization require significantly fewer environment interactions than model-free PbRL, (2) diverse preference queries can be synthesized safely and efficiently as a byproduct of standard model-based RL, and (3) reward pre-training based on suboptimal demonstrations can be performed without any environmental interaction. Our paper provides empirical evidence that learned dynamics models enable robots to learn customized policies based on user preferences in ways that are safer and more sample efficient than prior preference learning approaches. Supplementary materials and code are available at `https://sites.google.com/berkeley.edu/mop-rl`.**

I. INTRODUCTION

Developing assistive and collaborative robots that adapt to a variety of end-users requires customizing robot behaviors without manually tuning cost functions. One common approach is for robots to learn from human demonstrations [1, 29, 42]. However, providing demonstrations is not always possible and when demonstrations are suboptimal existing methods often overfit to the suboptimalities [16, 23, 24, 37].

Recent years have seen progress in preference-based reinforcement learning (PbRL)—an approach that, rather than relying on demonstrations, queries the user for pairwise preferences over trajectories [55]. However, existing PbRL approaches either use model-free PbRL and assume access to a perfect simulator [14, 30, 35] or or access to an analytic dynamics model [47]. In many problems, however, a good dynamics model is not available and must be inferred from data, as done in model-based RL [15]. Prior work on PbRL has not addressed the challenges and potential benefits of using a learned dynamics model to perform PbRL. Due to the importance of model-based RL in robotics domains, we propose to study Model-based Preference-based RL (MoP-RL). To our knowledge, this work is the first to study the benefits of learning a dynamics model in preference learning for realistic domains such as robotics. Central to our paper is the following idea:

Combining techniques from model-based RL with human preference learning offers several advantages: 1) safer and more sample efficient reward learning and policy optimization, 2) diverse trajectories useful for preference learning, which are a free byproduct of model-based RL, and 3) reward pre-training with automatically generated preferences over demonstrations without any environment interaction.

Safe and sample efficient learning. Combining model-based RL with PbRL allows robots to use a learned dynamics model to hypothesize different behaviors and even show these planned behaviors to a human without executing them in the environment, while still learning customizable behaviors. This enables safe and efficient reward function learning and policy optimization by limiting the number of environmental interactions. Though learning a dynamics model may require a large amount of data, we may amortize this cost by learning multiple reward functions for different users via the same learned dynamics model.

Diverse trajectories for preference learning. A learned dynamics model can also be used to simulate diverse trajectories so that preference queries can be created without any environment interaction. While such query generation has safety and efficiency benefits, since it requires no environment interaction, one must still decide which trajectories to show the human. We leverage the fact that, by design, many model-based RL approaches already generate a diverse set of trajectories. Given the learned dynamics model, we use model predictive control (MPC) [44] based on the cross entropy method (CEM) [7] to optimize trajectories. This common MPC approach uses a cost function (learned from preferences in our case) to iteratively refine an action sequence [15, 18, 21, 28, 38, 53]. Our insight is that this allows us to generate diverse preference queries without any extra computation—trajectories generated by successive CEM iterations provide a diverse set of trajectories that can be reused for preference learning. Because successive CEM iterations converge to the model's belief of the human's preferences, obtaining preference queries over successive CEM iterations also allows the supervisor to quickly correct the learned reward function if it leads to undesired behavior.

Reward pre-training from suboptimal demonstrations. Prior work has proposed Disturbance-based Reward Extrapolation (D-REX) [11] as a way to automatically produce ranked trajectories by injecting noise into a policy learned via behavioral cloning. While this approach can lead to well-shaped reward functions and enable better-than-demonstrator performance, it requires rolling out noisy trajectories in the environment, which can be unsafe and slow. However, with a

[1]UC Berkeley, [2]Army Research Lab, [3]University of Utah

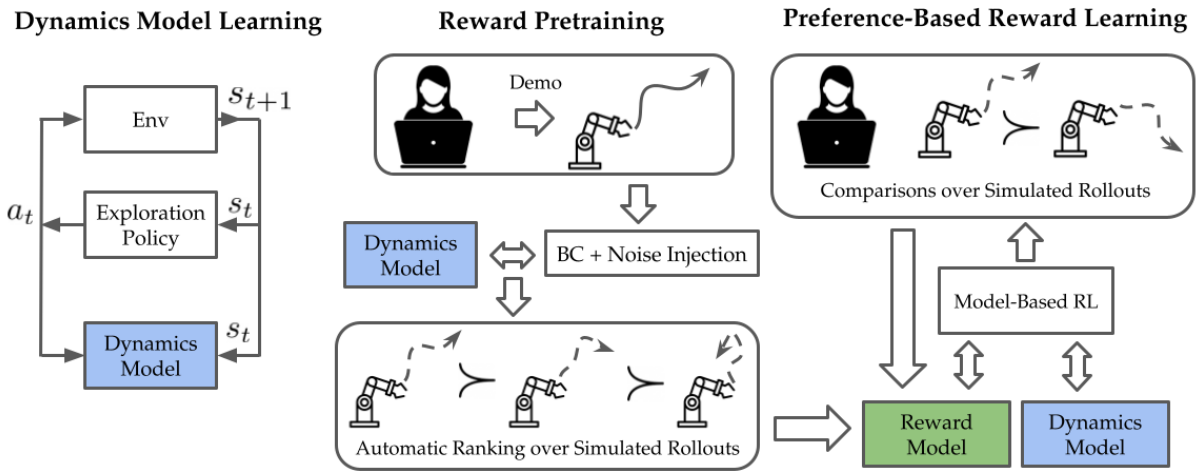

Fig. 1: **Model-Based Preference-Based RL (MoP-RL):** We leverage intrinsic exploration (left) to learn a dynamics model that enables efficient reward pre-training (center) by running behavioral cloning (BC) on a small number of demonstrations and using noise injection to simulate a variety of worse trajectories using rollouts in the learned dynamics model. We fine-tune the learned reward function via active preference queries to a human (right), where the trajectories for queries are simulated using the learned dynamics model and produced as a byproduct of running model-based RL with the learned reward function. MoP-RL uses the learned dynamics model to enable safe and sample efficient learning of human preferences.

learned dynamics model, we can perform such noisy rollouts within the learned model. We study this approach and show that we can leverage a small number of suboptimal demonstrations to achieve improvements in preference learning using model-based reward pre-training without requiring any environmental interactions.

In summary, our work makes the following contributions:

1) The first analysis of preference-based RL with a learned dynamics model (see Figure 1).
2) A novel preference-generation approach that uses the samples produced as a byproduct of model-based RL to generate diverse and informative preference-queries without requiring physical rollouts in the environment.
3) An approach for reward pre-training together with MoP-RL, which enables combining suboptimal demonstrations and pairwise preferences in a model-based setting.
4) Experiments suggesting that our approach can learn to perform complex tasks from preferences with fewer environmental interactions than prior approaches and can scale to high-dimensional visual control tasks.

II. RELATED WORK

Model-based learning. There has been much interest in learning dynamics models [26, 34, 39, 49] and applying these learned models to RL [4, 15, 36, 41, 51, 52, 59–61, 63]. Past work has indicated that model-based RL can achieve higher sample efficiency than model-free methods [32, 62]. Outside of RL, learned dynamics models have also been used successfully in imitation learning methods that learn from expert demonstrations. For instance, forward and inverse dynamics models have been shown to be useful when learning from demonstrations consisting of state-action pairs [2, 3, 22, 58], as well as when learning from observations without access to the demonstrator's actions [19, 33, 50]. ReQueST [43] actively collects human reward labels over model-based trajectory rollouts and then performs model-based RL using a learned reward function. However, our work differs in the following important ways: (1) ReQueST only performs model-based RL on the learned rewards after completion of learning; in contrast, MoP-RL leverages the reward learned *so far* to synthesize human feedback queries using informed trajectories that improve throughout learning. (2) ReQueST requires a human to provide a reward label to every state in each queried trajectory, resulting in a significantly higher human burden than MoP-RL, which requires only pairwise preferences over entire trajectories.

Preference-based RL. It is often easier for a person to qualitatively rank two or more behaviors than to demonstrate a good behavior. Learning from pairwise preferences over trajectories is a common approach to learning reward functions and corresponding RL policies. However, despite the widespread interest in preference-based RL, most prior work either takes a model-free approach [9, 14, 30, 35, 57] or assumes that the system dynamics are perfectly known [25, 31, 47, 54, 56, 64]. By contrast, we seek to study preference-based RL when using a learned dynamics model. Prior works on model-free preference-based RL demonstrate the effectiveness of pairwise preference learning for a range of RL tasks [**tien2022study**, 10, 14, 35]. In particular, PEBBLE [35] achieves state-of-the-art performance of PbRL in simulated robot locomotion and manipulation tasks. However, model-free RL tends to require more environment interaction than model-based RL methods [32, 62], and we are not aware of prior work that studies PbRL with dynamics model learning. Shin et al. [48] study PbRL for offline RL which avoids online interactions, but requires a large dataset of prior trajectories. Novoseller et al. [40] learn a dynamics model

and a reward predictor from human preferences in tabular environments, whereas our experiments focus on continuous, higher-dimensional environments. Our experiments suggest that MoP-RL achieves better policy performance than model-free PbRL with significantly fewer environment interactions.

Our work uses noise injection into the learned model for reward pretraining. Recent work by Chen et al. [13] provides evidence of inductive bias when pre-training using noise injection and a preference-learning objective, as done in our paper. However, Chen et al.'s approach requires solving a sequence of optimization problems and running online IRL, and thus requires a large amount of computation time and many on-policy rollouts [13]. We leave it to future work to investigate alternatives to D-REX [11] for reward pre-training without environmental interaction.

III. PROBLEM DEFINITION

Problem formulation and notation. We formulate the problem as a Markov Decision Process (MDP), $\mathcal{M} = (\mathcal{S}, \mathcal{A}, P, r)$, where $\mathcal{S}$ is the state space, $\mathcal{A}$ is the action space, $P : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to [0, 1]$ specifies the state transition probabilities, and r is the human's reward function, which, importantly, we assume is unobserved by the learning agent. A robot *trajectory* takes the form $\tau = (s_0, a_0, s_1, a_1, \ldots, s_T)$ for some episode length T. The robot does not observe numerical rewards. Rather, we assume access to a human user who can answer pairwise preference queries of the form, "Do you prefer trajectory A or B?" as well as possibly provide a small set $\mathcal{D}_{\text{demo}}$ of suboptimal demonstrations, where $\mathcal{D}_{\text{demo}}$ consists of trajectories $\tau^E = (s_0^E, a_0^E, \ldots, s_T^E) \in \mathcal{D}_{\text{demo}}$. The pairwise preference $\tau_A \succ \tau_B$ indicates that the user states a preference for trajectory τ_A over τ_B. We assume that the human's preferences are based on a true but unobserved reward function, $r : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$, which quantifies the utility of each state-action pair to the human. The total underlying reward of a trajectory $\tau = (s_0, a_0, s_1, a_1, \ldots, s_T)$ is denoted via $J(\tau) := \sum_{t=0}^{T-1} r(s_t, a_t)$. A *policy* $\pi : \mathcal{S} \times \mathcal{A} \to [0, 1]$ maps states to distributions over actions. We aim to learn a policy π that matches user preferences by maximizing expected performance under the human's true reward function r. Our only access to r is through the demonstrations and pairwise preferences.

Assumptions. We assume access to a human who can give preferences over trajectories based on their internal utility function as described above. As part of this assumption, we assume that we can visualize trajectories for the human to rank. Note that we only assume access to a method to show the human a visualization of states using the learned dynamics model—we do not assume to know or have a model of the true system dynamics.

Performance metrics. We aim to identify the optimal policy π^* that maximizes the human's total underlying reward in expectation: $\pi^* = \text{argmax}_\pi \mathbb{E}_{\tau\sim\pi}[J(\tau)] = \text{argmax}_\pi \mathbb{E}_{\tau\sim\pi}\left[\sum_{(s,a)\in\tau} r(s, a)\right]$. Furthermore, we aim to discover π^* while minimizing the total agent-environment interaction and the total number of human interactions.

IV. METHODS

A. Learning Rewards from Preferences

To obtain a well-performing policy π, we learn a predictor $\hat{r}_\theta : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ of the user's reward r, parameterized by θ. We model the probability $\hat{P}$ that the user prefers trajectory τ_1 to τ_2 via the Bradley-Terry model [10, 14]: $\hat{P}(\tau_1 \succ \tau_2) = \frac{\exp(\hat{J}_\theta(\tau_1))}{\exp(\hat{J}_\theta(\tau_1))+\exp(\hat{J}_\theta(\tau_2))}$, where $\hat{J}_\theta(\tau) := \sum_{s,a\in\tau} \hat{r}_\theta(s, a)$. Given a collection of user preferences $\mathcal{D}_{\text{pref}}$, we learn θ by minimizing the following cross-entropy loss function [10, 11, 14]:

$$\mathcal{L}_{\text{pref}}(\theta) = -\sum_{\tau_i \succ \tau_j \in \mathcal{D}_{\text{pref}}} \log\left[\frac{\exp\left(\hat{J}_\theta(\tau_i)\right)}{\exp\left(\hat{J}_\theta(\tau_i)\right) + \exp\left(\hat{J}_\theta(\tau_j)\right)}\right]. \tag{1}$$

With image observations, rather than directly predicting rewards via $\hat{r}_\theta(s, a)$, we use an LSTM to generate reward predictions that integrate over a sequence of observations. The LSTM takes in a state sequence, and its final output is fed into the feedforward network $\hat{r}_\theta$ to predict the reward.

B. Learning a Dynamics Model

Given a dataset of observed state-action transitions, $\mathcal{D}_{\text{dyn}} = \{(s_i, a_i, s_{i+1})\}_{i=1}^M$, we learn a transition dynamics model $\hat{f}_\phi : \mathcal{S} \times \mathcal{A} \to \mathcal{S}$, parametrized by ϕ. We learn ϕ by minimizing the following loss: $\mathcal{L}_{\text{dyn}}(\phi) := \sum_{(s,a,s_{\text{next}})\in\mathcal{D}_{\text{dyn}}} (s_{\text{next}} - \hat{f}_\phi(s, a))^2 + \lambda||\phi||^2$, where our forward dynamics network predicts the change in state, $\hat{f}_\phi^{\text{diff}}(s, a) := \hat{f}_\phi(s, a) - s$, and λ is a weight decay hyperparameter. We pre-process the data by normalizing it to have zero mean and unit variance along each dimension. When the observations are images, we adapt Stochastic Video Generation with a Learned Prior (SVG-LP) [17] to train a visual dynamics model.

This work leverages two methods for collecting the dynamics training dataset $\mathcal{D}_{\text{dyn}}$. The first is random agent-environment interaction, in which the agent samples its action space uniformly randomly over a series of trajectories beginning in different start states. However, in many domains, random agent-environment interaction may yield insufficient exploration. To collect diverse data for learning a dynamics model, we leverage random network distillation (RND) [12], a powerful approach for exploration in deep RL that provides reward bonuses based on the error of a neural network predicting the output of a randomly-initialized network. We gather a dataset of state-action transitions observed while training a Soft Actor Critic [27] policy using the RND bonus as the sole reward signal. In addition to using RND for dynamics pre-training, we integrate RND into our online learning pipeline as discussed in Section IV-D.

C. Pre-training with Suboptimal Demonstrations

Brown et al. [11] propose the D-REX algorithm, in which demonstration trajectories are automatically ranked based on

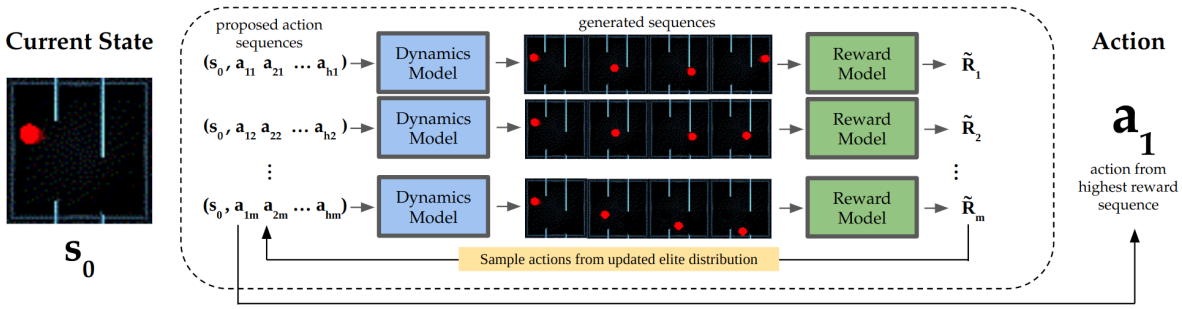

Fig. 2: **Cross Entropy Method:** For an initial state s_0, we sample a set of action sequences and predict the corresponding state trajectory for each one using the dynamics model $\hat{f}_\phi$. We estimate rewards for these trajectories as a linear combination of the reward model $\hat{r}_\theta$'s prediction and an RND bonus. The m_e action sequences with the highest predicted rewards form the elites; these are used to update the CEM distribution and sample the next population of action sequences. In the final CEM iteration, the mean of the action distribution of elites is selected for execution in the environment.

the degree of noise used to generate them. We adapt D-REX to the model-based setting. Given a set of human demonstrations $\mathcal{D}_{\text{demo}}$, we train a behavior cloning policy $\pi_{BC} : \mathcal{S} \to \mathcal{A}$ on each state-action pair in $\mathcal{D}_{\text{demo}}$. Importantly, we roll out this policy inside the *learned dynamics model*. Simulated rollouts are generated with ϵ-greedy noise, such that the agent takes a uniformly random action with probability ϵ and follows π_{BC} otherwise. For trajectories τ_i and τ_j starting from the same initial state and generated with noise levels ϵ_i and ϵ_j, $\epsilon_i > \epsilon_j$, we automatically rank $\tau_j \succ \tau_i$. We also prefer any trajectory $\tau \in \mathcal{D}_{\text{demo}}$ over any trajectory generated by rolling out π_{BC}.

D. Model-Based RL

We adopt a model-based RL approach inspired by Chua et al. [15], leveraging the cross-entropy method (CEM) [45, 46] to plan with respect to learned reward and dynamics models (see Fig. 2). CEM maintains a distribution $\mathcal{N}(\boldsymbol{\mu}_i, \Sigma_i)$ in each CEM iteration i, from which the i^{th} population of m action sequences is sampled: $\{a_{1j}^{(i)}, \dots, a_{hj}^{(i)} \mid j = 1, \dots, m\}$, with planning horizon h. We then use the learned dynamics and rewards, $\hat{f}_\phi$ and $\hat{r}_\theta$, to estimate the expected total reward $\tilde{R}_j^{(i)}$ of each action sequence, conditioned on a fixed start state $s_0 \in \mathcal{S}$: $\hat{s}_{1j}^{(i)} = \hat{f}_\phi(s_0, a_{1j}^{(i)})$, $\hat{s}_{2j}^{(i)} = \hat{f}_\phi(\hat{s}_{1j}^{(i)}, a_{2j}^{(i)})$, etc., and $\tilde{R}_j^{(i)} = \sum_{t=0}^{h-1} \hat{r}_\theta(\hat{s}_{tj}^{(i)}, a_{(t+1)j}^{(i)})$, where $\hat{s}_{0j}^{(i)} := s_0$. The m population members are then ranked according to their estimated rewards $\tilde{R}_j^{(i)}$, and the top $m_e < m$ population members are termed the *elites*. We take the mean and variance of the elites to refine the CEM distribution, obtaining $\mathcal{N}(\boldsymbol{\mu}_{i+1}, \Sigma_{i+1})$. To choose actions in the environment, we employ model predictive control (MPC), as in [15, 36]: we execute the first k actions of the action sequence given by $\boldsymbol{\mu}_{\text{final}}$, the mean of the action distribution calculated in the final CEM iteration. Our experiments utilize $k = 1$.

We additionally leverage RND [12] to improve exploration. RND provides a prediction error signal $\hat{g}_\rho(s)$ at each state $s \in \mathcal{S}$, parameterized by $\boldsymbol{\rho}$. Intuitively, areas with higher error $\hat{g}_\rho(s)$ have been explored less, and so $\hat{g}_\rho(s)$ functions as an exploration bonus. During CEM, we augment the expected reward: $\tilde{R}_j^{(i)} = \sum_{t=0}^{h-1} [\hat{r}_\theta(\hat{s}_{tj}^{(i)}, a_{(t+1)j}^{(i)}) + \lambda_{RND} \hat{g}_\rho(\hat{s}_{tj}^{(i)})]$, where λ_{RND} is a hyperparameter.

E. Active Model-Based Preference Query Generation

To efficiently generate pairwise preference queries without additional environment interaction, we form queries using trajectories from the CEM process, during which the learned dynamics model $\hat{f}_\phi$ is used to roll out candidate action sequences. In selecting pairs of CEM trajectories to elicit preferences, we consider two main questions: (1) How can we generate a diverse pool $\mathcal{D}_{\text{traj}}$ of trajectories, e.g., so that the human is not forced to give preferences between overly-similar or otherwise limited choices? (2) Given a pool $\mathcal{D}_{\text{traj}}$ of candidate trajectories, how can we select which trajectory pairs are best to show to the human?

To address (1), we include trajectories from all CEM iterations in $\mathcal{D}_{\text{traj}}$. This ensures that $\mathcal{D}_{\text{traj}}$ contains diverse data, including from non-elite trajectories. Notably, constructing preference queries using only the final CEM iteration could force the human to compare overly-similar trajectories, leading to unreliable preference labels. At a randomly-selected time $t_{keep} \in \{1, \dots, T\}$ in each episode, we save $m_{\text{keep}} < m$ trajectories from each CEM iteration in $\mathcal{D}_{\text{traj}}$; these are randomly selected from the elites in the final CEM iteration and from among non-elites in previous iterations. We randomly choose a new t_{keep} in each episode so that across episodes, $\mathcal{D}_{\text{pref}}$ contains pairs of trajectories beginning in different states. For (2), we maximize the information gain [5, 6] to identify trajectory pairs in $\mathcal{D}_{\text{traj}}$ that are expected to be most informative about the human's underlying reward r. We identify the N trajectory pairs $\{\tau_1^q, \tau_2^q \mid \tau_1^q, \tau_2^q \in \mathcal{D}_{\text{traj}}, q = 1, \dots, N\}$ with the highest mutual information $I(r; \mathbb{I}_{[\tau_1^q \succ \tau_2^q]} \mid \mathcal{D}_{\text{pref}})$ between their preference outcome and the unknown reward r [5, 6]:

$$I(r; \mathbb{I}_{[\tau_1 \succ \tau_2]} \mid \mathcal{D}_{\text{pref}}) = H(\mathbb{I}_{[\tau_1 \succ \tau_2]} \mid \mathcal{D}_{\text{pref}}) - \mathbb{E}_{r \sim p(r \mid \mathcal{D}_{\text{pref}})}[H(\mathbb{I}_{[\tau_1 \succ \tau_2]} \mid r, \mathcal{D}_{\text{pref}})], \tag{2}$$

where $\mathbb{I}_{[\tau_1 \succ \tau_2]} \in \{0, 1\}$ indicates the outcome of a preference query between τ_1 and τ_2, $\mathcal{D}_{\text{pref}}$ is the pairwise preference

Algorithm 1: Model-based Preference-based Reinforcement Learning (MoP-RL)

Initialize parameters θ, ϕ of reward and dynamics networks $\hat{r}_\theta$ and $\hat{f}_\phi$, respectively
Collect dataset $\mathcal{D}_{\text{dyn}}$ of transitions in environment using exploration strategy (random or RND)
Optimize $\mathcal{L}_{\text{dyn}}$ in (IV-B) with respect to ϕ
Initialize $\mathcal{D}_{\text{pref}} \leftarrow \emptyset$
Collect dataset $\mathcal{D}_{\text{demo}}$ of human demonstrations
Initialize $\hat{r}_\theta$ via model-based D-REX using $\mathcal{D}_{\text{demo}}$ and using $\hat{f}_\phi$ to roll out trajectories
for *each episode* k **do**
  Initialize $\mathcal{D}_{\text{traj}} \leftarrow \emptyset$
  $t_{\text{keep}} \sim \mathcal{U}\{1, \ldots, T\}$
  for *each timestep* $t \in \{1, \ldots, T\}$ **do**
    $s_t \leftarrow$ current state
    Select optimal action sequence $(a_t, \ldots, a_{t+h-1})$ using CEM with $\hat{f}_\phi$, $\hat{r}_\theta$, and $\hat{g}_\rho$
    Execute a_t in the environment
    if $t = t_{\text{keep}}$ **then**
      $\mathcal{D}_{\text{traj}} \leftarrow m_{\text{keep}}$ trajectories from each CEM iteration
  $\{\tau_1^q, \tau_2^q \mid q = 1, \ldots, N\} \leftarrow$ top N trajectory pairs in $\mathcal{D}_{\text{traj}}$ according to info-gain (2)
  for $q = 1, \ldots, N$ **do**
    Query human for preference y
    $\mathcal{D}_{\text{pref}} \leftarrow \mathcal{D}_{\text{pref}} \cup \{(\tau_1^q, \tau_2^q, y)\}$
  if k *% reward update frequency = 0* **then**
    Optimize $\mathcal{L}_{\text{pref}}$ in (1) with respect to θ
  if k *% RND update frequency = 0* **then**
    Update RND network g_ρ on visited states

dataset (so far), H is information entropy, and we use an ensemble of reward networks to estimate the posterior $p(r \mid \mathcal{D}_{\text{pref}})$. Algorithm 1 details the entire MoP-RL algorithm.

V. EXPERIMENT RESULTS

A. Experiment Domains

Our experiments consider four simulation domains detailed below. For all domains, preferences are provided by a synthetic labeler that does not make mistakes. To simulate the fact that humans have difficulty answering some queries [6], the labeler skips preference queries where the reward difference falls below a threshold. Additional experiment details are provided in the supplementary website.

Maze-LowDim. The agent must navigate a 2-dimensional maze to reach a goal location. The agent's state is given by its (x, y) coordinates in the maze. The agent's action is a vector $(f_x, f_y) \in \mathbb{R}^2$, $-f_{\max} \leq f_x, f_y \leq f_{\max}$, representing a force applied in each coordinate direction. The synthetic labeler provides preferences using a hand-specified underlying cost function, which gives higher preference first for reaching the far right wall and then the top-right corner.

Maze-Image. This environment uses the same maze, agent dynamics, and synthetic reward as Maze-LowDim; however, the agent observes an image of the environment (as in Fig. 2) at each step.

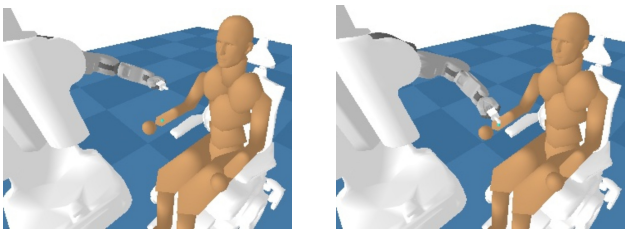

Fig. 3: Assistive Gym start (left) and goal (right) state.

Assistive-Gym. We utilize a simplified version of the itch scratching task in the Assistive Gym [20] simulation environment (Fig. 3), in which a robot manipulator must position itself on an itch on a person's arm and apply a target amount of force. We use a reduced state space containing 1) the vector difference between the robot end effector and itch location and 2) the amount of force applied by the end effector. The synthetic labeler prefers trajectories that bring the end effector close to the itch and apply an amount of force below a threshold to the person's arm.

Hopper. In the OpenAI Gym [8] Hopper environment, we aim to train the agent to perform a backflip. The state space $\mathcal{S} \subset \mathbb{R}^{11}$ consists of the angular positions and velocities of the Hopper's 3 joints and the top of the robot, as well as the height of the hopper and the velocities of the x-coordinate and z-coordinate of the top. The action space $\mathcal{A} \subset \mathbb{R}^3$ consists of a torque applied to each of the 3 joints. Pairwise preferences over trajectories are given by a synthetic labeler with a hand-engineered reward function designed by Christiano et al. [14].

B. Performance Metrics and Algorithms Compared

Our experiments compare the performance of MoP-RL to PEBBLE [35], a state-of-the-art model-free PbRL algorithm. For each simulation domain, all algorithms learn from the same number of pairwise preferences. Performance is evaluated via the following metrics:

1) **Success rate** (Maze-LowDim, Maze-Image): percentage of times that the agent satisfies a goal condition.
2) **Average reward** (Assistive-Gym): ground truth reward over an entire episode, averaged over several rollouts.

In each environment, MoP-RL first trains a dynamics model. In Maze-LowDim, we collect dynamics training data via random environment interaction, while in the other domains—in which exploration is more challenging—we collect dynamics data via RND. Then, MoP-RL performs interactive reward learning, possibly with reward pre-training from demonstrations. The reward model $\hat{r}_\theta$ is a function of state and action in Assistive-Gym and Hopper, and of state only in both maze environments. PEBBLE performs an initial unsupervised exploration phase prior to reward learning. MoP-RL used 1000 trajectories to learn the dynamics model in Maze-LowDim, 2000 in Maze-Image, 2500 for Assistive-Gym, and 5000 for Hopper. PEBBLE used 150 trajectories

Fig. 4: MoP-RL trains a Hopper to perform a backflip via preference queries over learned dynamics.

Algorithm	Training Rollouts	Success Rate
PEBBLE	50	6.3%
PEBBLE	100	81.3%
PEBBLE	200	100%
MoP-RL (w/o demos)	18	0.0%
MoP-RL (w/o demos)	24	56.3%
MoP-RL (w/o demos)	30	100%
MoP-RL (w/ 2 demos)	18	100%
MoP-RL (w/ 2 demos)	24	100%
MoP-RL (w/ 2 demos)	30	100%

TABLE I: **Performance on Maze-LowDim Environment.** We report the numbers of unsupervised and training rollouts and the success rate (over 16 rollouts) for each method when trained with 60 preference queries.

Algorithm	Unsupervised Rollouts	Training Rollouts	Success Rate
PEBBLE	2000	3300	0.0%
PEBBLE	5000	3300	0.0%
MoP-RL (w/o demos)	2000	800	25.0%
MoP-RL (w/ 10 demos)	2000	800	68.8%

TABLE II: **Performance on Maze-Image Environment.** We report the numbers of unsupervised and training rollouts and the success rate (over 16 rollouts) for each method compared. Both methods were trained with 1000 preference queries.

in the unsupervised step in Maze-LowDim, 2000 or 5000 in Maze-Image (see Table II), and 150 in Assistive-Gym.

C. Results

We hypothesize that MoP-RL will require fewer environmental interactions in comparison to model-free RL, while achieving similar or better performance. We compare MoP-RL and PEBBLE [35] in the Maze-LowDim, Maze-Image, and Assistive-Gym Itch Scratching tasks.

Maze-LowDim. Table I shows results for the Maze-LowDim environment. We see that PEBBLE requires a large number of rollouts to learn the reward function accurately enough to successfully navigate the maze. By contrast, MoP-RL is much more sample efficient. We also observe that pre-training the reward network using two demonstrations significantly speeds up online preference learning, requiring fewer trajectories to successfully complete the task.

Maze-Image. Table II shows results for the Maze-Image environment. We see that MoP-RL outperforms PEBBLE, even when PEBBLE is given significantly more unsupervised access to the environment. Prior work [35] only evaluates PEBBLE on low-dimensional reward learning tasks. We adapted the author's implementation of PEBBLE to allow an image-based reward function and use the same reward function architecture for both PEBBLE and MoP-RL. However, our results demonstrate that PEBBLE struggles in high-dimensional visual domains. Furthermore, we see that pre-training with 10 suboptimal demonstrations significantly improves reward learning without any additional environment interactions. Pretraining leads to improved task success by first learning a rough estimate of the reward function and then fine-tuning via model-based preference queries.

Algorithm	Training Rollouts	Average Reward
PEBBLE	100	-46.1 $\pm$ 1.36
MoP-RL	8	-26.9 $\pm$ 6.82

TABLE III: **Performance on Assistive-Gym Itch Scratching Environment.** We report the numbers of unsupervised and training rollouts and the average reward (mean $\pm$ standard error over 4 rollouts) for each method compared. Both methods were trained with 80 preference queries.

Assistive Gym. Table III shows the results for the itching task from Assistive Gym. MoP-RL also achieves a higher task reward than PEBBLE in this environment.

The above experiments demonstrate that MoP-RL requires significantly fewer environment interaction steps than PEBBLE to learn a reward function. MoP-RL also enables dynamics model pre-training to be performed separately from preference learning. This is important since, unlike a learned reward function, learned dynamics can be re-used to safely and efficiently learn the preferences of multiple users.

Hopper Backflip. Inspired by previous model-free PbRL results [14], we demonstrate that MoP-RL can train the OpenAI Gym Hopper to perform a backflip via preference queries over a learned dynamics model. An example learned backflip, displayed in Figure 4, suggests that MoP-RL can learn to perform novel behaviors for which designing a hand-crafted reward function is difficult.

VI. CONCLUSION

This work introduces MoP-RL, a model-based approach to preference-based RL. We demonstrate that dynamics modeling is uniquely suited for preference-based learning, since a learned dynamics model can be used to generate pairwise preference queries without environment interaction and to greatly reduce the amount of environmental interaction needed to optimize a policy from the learned reward function. Furthermore, MoP-RL combines multiple human interaction modalities by integrating preference learning with model-based reward pre-training from demonstrations for improved performance. Limitations include that we synthetically generated all pairwise preferences and that we do not consider noisy preference data. Future work includes evaluating MoP-RL in a user study, investigating safety-critical domains, and exploring applications of MoP-RL, in environment where interactions are expensive and where different human users have varying robot interaction preferences.

REFERENCES

[1] B. D. Argall, S. Chernova, M. Veloso, and B. Browning, "A survey of robot learning from demonstration," *Robotics and autonomous systems*, vol. 57, no. 5, pp. 469–483, 2009.

[2] N. Baram, O. Anschel, I. Caspi, and S. Mannor, "End-to-end differentiable adversarial imitation learning," in *International Conference on Machine Learning*, PMLR, 2017, pp. 390–399.

[3] N. Baram, O. Anschel, and S. Mannor, "Model-based adversarial imitation learning," *arXiv preprint arXiv:1612.02179*, 2016.

[4] S. Bechtle, Y. Lin, A. Rai, L. Righetti, and F. Meier, "Curious iLQR: Resolving uncertainty in model-based RL," in *Conference on Robot Learning*, PMLR, 2020, pp. 162–171.

[5] E. Biyik, N. Huynh, M. Kochenderfer, and D. Sadigh, "Active preference-based Gaussian process regression for reward learning," in *Robotics: Science and Systems*, 2020.

[6] E. Biyik, M. Palan, N. C. Landolfi, D. P. Losey, D. Sadigh, *et al.*, "Asking easy questions: A user-friendly approach to active reward learning," in *Conference on Robot Learning*, PMLR, 2020, pp. 1177–1190.

[7] Z. I. Botev, D. P. Kroese, R. Y. Rubinstein, and P. L'Ecuyer, "The cross-entropy method for optimization," in *Handbook of statistics*, vol. 31, Elsevier, 2013, pp. 35–59.

[8] G. Brockman *et al.*, *Openai gym*, 2016. eprint: arXiv:1606.01540.

[9] D. Brown, R. Coleman, R. Srinivasan, and S. Niekum, "Safe imitation learning via fast Bayesian reward inference from preferences," in *International Conference on Machine Learning*, PMLR, 2020, pp. 1165–1177.

[10] D. Brown, W. Goo, P. Nagarajan, and S. Niekum, "Extrapolating beyond suboptimal demonstrations via inverse reinforcement learning from observations," in *International Conference on Machine Learning*, PMLR, 2019, pp. 783–792.

[11] D. S. Brown, W. Goo, and S. Niekum, "Better-than-demonstrator imitation learning via automatically-ranked demonstrations," in *Conference on robot learning*, PMLR, 2020, pp. 330–359.

[12] Y. Burda, H. Edwards, A. Storkey, and O. Klimov, "Exploration by random network distillation," in *Seventh International Conference on Learning Representations*, 2019, pp. 1–17.

[13] L. Chen, R. Paleja, and M. Gombolay, "Learning from suboptimal demonstration via self-supervised reward regression," in *Conference on robot learning*, PMLR, 2021, pp. 1262–1277.

[14] P. F. Christiano, J. Leike, T. B. Brown, M. Martic, S. Legg, and D. Amodei, "Deep reinforcement learning from human preferences," in *NIPS*, 2017.

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

[16] C. Chuck, M. Laskey, S. Krishnan, R. Joshi, R. Fox, and K. Goldberg, "Statistical data cleaning for deep learning of automation tasks from demonstrations," in *2017 13th IEEE Conference on Automation Science and Engineering (CASE)*, IEEE, 2017, pp. 1142–1149.

[17] E. Denton and R. Fergus, "Stochastic video generation with a learned prior," in *International conference on machine learning*, PMLR, 2018, pp. 1174–1183.

[18] 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.

[19] A. Edwards, H. Sahni, Y. Schroecker, and C. Isbell, "Imitating latent policies from observation," in *International conference on machine learning*, PMLR, 2019, pp. 1755–1763.

[20] Z. Erickson, V. Gangaram, A. Kapusta, C. K. Liu, and C. C. Kemp, "Assistive gym: A physics simulation framework for assistive robotics," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, 2020, pp. 10 169–10 176.

[21] C. Finn and S. Levine, "Deep visual foresight for planning robot motion," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, 2017, pp. 2786–2793.

[22] C. Finn, S. Levine, and P. Abbeel, "Guided cost learning: Deep inverse optimal control via policy optimization," in *International conference on machine learning*, PMLR, 2016, pp. 49–58.

[23] G. R. Ghosal, M. Zurek, D. S. Brown, and A. D. Dragan, "The effect of modeling human rationality level on learning rewards from multiple feedback types," in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2023.

[24] N. Gopalan, N. Moorman, M. Natarajan, and M. Gombolay, "Negative result for learning from demonstration: Challenges for end-users teaching robots with task and motion planning abstractions," in *Robotics: Science and Systems (RSS)*, 2022.

[25] A. Gritsenko and D. Berenson, "Learning cost functions for motion planning from human preferences," in *Proceedings of the IROS 2014 Workshop on Machine Learning in Planning and Control of Robot Motion*, vol. 1, 2014, pp. 48–6.

[26] S. Grünewälder, G. Lever, L. Baldassarre, M. Pontil, and A. Gretton, "Modelling transition dynamics in MDPs with RKHS embeddings," in *Proceedings of the 29th International Conference on International Conference on Machine Learning*, 2012, pp. 1603–1610.

[27] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine, "Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor," in *International conference on machine learning*, PMLR, 2018, pp. 1861–1870.

[28] D. Hafner *et al.*, "Learning latent dynamics for planning from pixels," in *International conference on machine learning*, PMLR, 2019, pp. 2555–2565.

[29] A. Hussein, M. M. Gaber, E. Elyan, and C. Jayne, "Imitation learning: A survey of learning methods," *ACM Computing Surveys (CSUR)*, vol. 50, no. 2, pp. 1–35, 2017.

[30] B. Ibarz, J. Leike, T. Pohlen, G. Irving, S. Legg, and D. Amodei, "Reward learning from human preferences and demonstrations in Atari," *arXiv preprint arXiv:1811.06521*, 2018.

[31] A. Jain, S. Sharma, T. Joachims, and A. Saxena, "Learning preferences for manipulation tasks from online coactive feedback," *The International Journal of Robotics Research*, vol. 34, no. 10, pp. 1296–1313, 2015.

[32] Ł. Kaiser *et al.*, "Model based reinforcement learning for Atari," in *International Conference on Learning Representations*, 2019.

[33] R. Kidambi, J. Chang, and W. Sun, "MobILE: Model-based imitation learning from observation alone," *Advances in Neural Information Processing Systems*, vol. 34, 2021.

[34] J. Z. Kolter and G. Manek, "Learning stable deep dynamics models," *Advances in neural information processing systems*, vol. 32, 2019.

[35] K. Lee, L. Smith, and P. Abbeel, "PEBBLE: Feedback-efficient interactive reinforcement learning via relabeling experience and unsupervised pre-training," in *International Conference on Machine Learning*, 2021.

[36] K. Lowrey, A. Rajeswaran, S. Kakade, E. Todorov, and I. Mordatch, "Plan online, learn offline: Efficient learning and exploration via model-based control," in *International Conference on Learning Representations*, 2018.

[37] A. Mandlekar *et al.*, "What matters in learning from offline human demonstrations for robot manipulation," in *Conference on Robot Learning*, PMLR, 2022, pp. 1678–1690.

[38] A. Nagabandi, K. Konolige, S. Levine, and V. Kumar, "Deep dynamics models for learning dexterous manipulation," in *Conference on Robot Learning*, PMLR, 2020, pp. 1101–1112.

[39] D. Nguyen-Tuong and J. Peters, "Model learning for robot control: A survey," *Cognitive processing*, vol. 12, no. 4, pp. 319–340, 2011.

[40] E. Novoseller, Y. Wei, Y. Sui, Y. Yue, and J. Burdick, "Dueling posterior sampling for preference-based reinforcement learning," in *Conference on Uncertainty in Artificial Intelligence*, PMLR, 2020, pp. 1029–1038.

[41] M. Okada and T. Taniguchi, "Variational inference MPC for Bayesian model-based reinforcement learning," in *Conference on Robot Learning*, PMLR, 2020, pp. 258–272.

[42] H. Ravichandar, A. S. Polydoros, S. Chernova, and A. Billard, "Recent advances in robot learning from demonstration," *Annual review of control, robotics, and autonomous systems*, vol. 3, pp. 297–330, 2020.

[43] S. Reddy, A. Dragan, S. Levine, S. Legg, and J. Leike, "Learning human objectives by evaluating hypothetical behavior," in *International Conference on Machine Learning*, PMLR, 2020, pp. 8020–8029.

[44] J. A. Rossiter, *Model-based predictive control: a practical approach*. CRC press, 2017.

[45] R. Rubinstein and D. Kroese, *The Cross-Entropy Method: A Unified Approach to Combinatorial Optimization, Monte-Carlo Simulation, and Machine Learning*. Springer-Verlag, 2004.

[46] R. Rubinstein, "The Cross-Entropy Method for Combinatorial and Continuous Optimization," *Methodology And Computing In Applied Probability*, 1999.
[47] D. Sadigh, A. D. Dragan, S. Sastry, and S. A. Seshia, "Active preference-based learning of reward functions.," in *Robotics: Science and Systems*, 2017.
[48] D. Shin, A. Dragan, and D. S. Brown, "Benchmarks and algorithms for offline preference-based reward learning," *Transactions on Machine Learning Research*, 2023.
[49] T. G. Thuruthel, E. Falotico, F. Renda, and C. Laschi, "Learning dynamic models for open loop predictive control of soft robotic manipulators," *Bioinspiration & biomimetics*, vol. 12, no. 6, p. 066 003, 2017.
[50] F. Torabi, G. Warnell, and P. Stone, "Behavioral cloning from observation," in *Proceedings of the 27th International Joint Conference on Artificial Intelligence*, 2018, pp. 4950–4957.
[51] R. Veerapaneni *et al.*, "Entity abstraction in visual model-based reinforcement learning," in *Conference on Robot Learning*, PMLR, 2020, pp. 1439–1456.
[52] S. J. Wang, S. Triest, W. Wang, S. Scherer, and A. Johnson, "Rough terrain navigation using divergence constrained model-based reinforcement learning," in *5th Annual Conference on Robot Learning*, 2021.
[53] G. Williams, A. Aldrich, and E. Theodorou, "Model predictive path integral control using covariance variable importance sampling," *arXiv preprint arXiv:1509.01149*, 2015.
[54] A. Wilson, A. Fern, and P. Tadepalli, "A Bayesian approach for policy learning from trajectory preference queries," *Advances in neural information processing systems*, vol. 25, 2012.
[55] C. Wirth, R. Akrour, G. Neumann, J. Fürnkranz, *et al.*, "A survey of preference-based reinforcement learning methods," *Journal of Machine Learning Research*, vol. 18, no. 136, pp. 1–46, 2017.
[56] C. Wirth and J. Fürnkranz, "On learning from game annotations," *IEEE Transactions on Computational Intelligence and AI in Games*, vol. 7, no. 3, pp. 304–316, 2014.
[57] C. Wirth, J. Fürnkranz, and G. Neumann, "Model-free preference-based reinforcement learning," in *Thirtieth AAAI Conference on Artificial Intelligence*, 2016.
[58] A. Wu, A. Piergiovanni, and M. S. Ryoo, "Model-based behavioral cloning with future image similarity learning," in *Conference on Robot Learning*, PMLR, 2020, pp. 1062–1077.
[59] B. Wu, S. Nair, L. Fei-Fei, and C. Finn, "Example-driven model-based reinforcement learning for solving long-horizon visuomotor tasks," in *5th Annual Conference on Robot Learning*, 2021.
[60] Y. Xu, Z. Liu, G. Duan, J. Zhu, X. Bai, and J. Tan, "Look before you leap: Safe model-based reinforcement learning with human intervention," in *Conference on Robot Learning*, PMLR, 2022, pp. 332–341.
[61] T. Yu *et al.*, "MOPO: Model-based offline policy optimization," *Advances in Neural Information Processing Systems*, vol. 33, pp. 14 129–14 142, 2020.
[62] M. Zhang, S. Vikram, L. Smith, P. Abbeel, M. Johnson, and S. Levine, "SOLAR: Deep structured representations for model-based reinforcement learning," in *International Conference on Machine Learning*, PMLR, 2019, pp. 7444–7453.
[63] Y. Zhang, I. Clavera, B. Tsai, and P. Abbeel, "Asynchronous methods for model-based reinforcement learning," in *Conference on Robot Learning*, PMLR, 2020, pp. 1338–1347.
[64] M. Zucker, J. A. Bagnell, C. G. Atkeson, and J. Kuffner, "An optimization approach to rough terrain locomotion," in *2010 IEEE International Conference on Robotics and Automation*, IEEE, 2010, pp. 3589–3595.