

AMOR: Adaptive Character Control through Multi-Objective Reinforcement Learning

LUCAS N. ALEGRE*, Universidade Federal do Rio Grande do Sul, Brazil and Disney Research, Switzerland

AGON SERIFI*, Disney Research, Switzerland

RUBEN GRANDIA, Disney Research, Switzerland

DAVID MÜLLER, Disney Research, Switzerland

ESPEN KNOOP, Disney Research, Switzerland

MORITZ BÄCHER, Disney Research, Switzerland

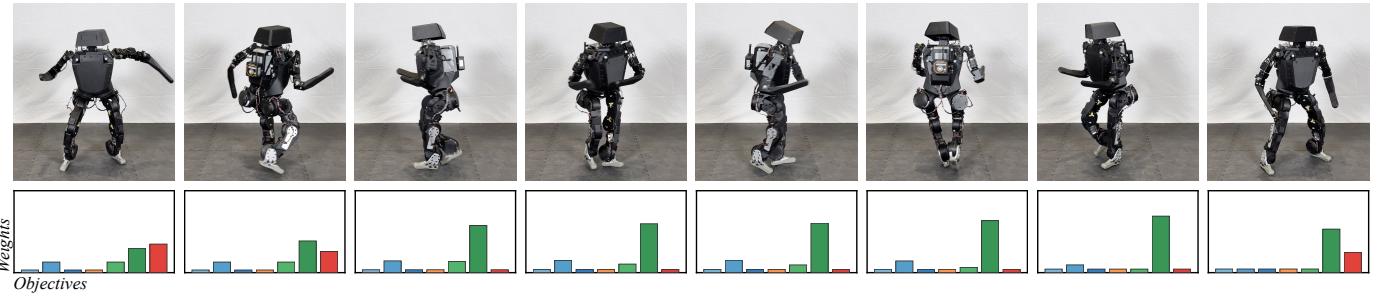


Fig. 1. Our method uses multi-objective reinforcement learning to enable on-the-fly tuning of reward weights post-training, which can be used to transfer challenging motions onto physical robots. The bar plots show the tuned weights of the individual reward terms at different points in time: [upper body joint angles](#), [lower body joint angles](#), [foot joint angles](#), [rigid body poses](#), [root pose](#), [root velocities](#), and [smoothness](#).

Reinforcement learning (RL) has significantly advanced the control of physics-based and robotic characters that track kinematic reference motion. However, methods typically rely on a weighted sum of conflicting reward functions, requiring extensive tuning to achieve a desired behavior. Due to the computational cost of RL, this iterative process is a tedious, time-intensive task. Furthermore, for robotics applications, the weights need to be chosen such that the policy performs well in the real world, despite inevitable sim-to-real gaps. To address these challenges, we propose a multi-objective reinforcement learning framework that trains a single policy conditioned on a set of weights, spanning the Pareto front of reward trade-offs. Within this framework, weights can be selected and tuned after training, significantly speeding up iteration time. We demonstrate how this improved workflow can be used to perform highly dynamic motions with a robot character. Moreover, we explore how weight-conditioned policies can be leveraged in hierarchical settings, using a high-level policy to dynamically select weights according to the current task. We show that the multi-objective policy encodes a diverse spectrum of behaviors, facilitating efficient adaptation to novel tasks.

CCS Concepts: • Computing methodologies → Control methods; Physical simulation; Reinforcement learning; Learning from demonstrations.

Additional Key Words and Phrases: multi-objective reinforcement learning, character control, motion tracking, physics-based characters, robotics

*denotes equal contribution.

Authors' Contact Information: Lucas N. Alegre, Universidade Federal do Rio Grande do Sul, Porto Alegre, RS, Brazil and Disney Research, Zürich, Switzerland, lnalegre@inf.ufrgs.br; Agon Serifi, Disney Research, Zürich, Switzerland, agon.serifi@disneyresearch.com; Ruben Grandia, Disney Research, Zürich, Switzerland, ruben.grandia@disneyresearch.com; David Müller, Disney Research, Zürich, Switzerland, david.mueller@disneyresearch.com; Espen Knoop, Disney Research, Zürich, Switzerland, espen.knoop@disneyresearch.com; Moritz Bächer, Disney Research, Zürich, Switzerland, moritz.baecher@disneyresearch.com.

1 Introduction

Creating motion tracking controllers is a fundamental challenge in physics-based character animation and robotics. The predominant approach trains controllers using reinforcement learning (RL), where weighted sums of carefully-designed reward functions guide the agent towards achieving a desired behavior.

A key challenge is that, due to possibly conflicting rewards (e.g., maximizing accuracy while minimizing energy), choosing the weights is nontrivial. In practice, finding a set of weights that results in the intended behavior often involves a trial-and-error approach. Because standard RL methods require setting these weights prior to training, the required retraining makes this a time-intensive process. Furthermore, in a setting where multiple motions are tracked by a single policy, motions with distinct styles or dynamics may benefit from a different trade-off, which a single fixed set of weights cannot provide.

The requirement for tuning weights is further exacerbated in robotics applications, where a controller is trained in simulation, but expected to perform well in the real world. The sim-to-real gap poses additional, but unknown, requirements on the behavior, which need to be navigated by selecting appropriate weights. For example, smoothness terms typically need to be weighted much higher for satisfactory results on real hardware compared to simulation.

To overcome these limitations, we propose **Adaptive character control through Multi-Objective Reinforcement learning (AMOR)**, a method that leverages multi-objective RL (MORL) to train a control policy conditioned on reward weights. Our method allows users to set the weights after training and directly observe the adapted

behavior, without requiring retraining. Any iterative weight tuning is therefore significantly accelerated.

Being able to adjust weights without retraining opens up exciting opportunities. In this paper, we explore two ways to leverage the capabilities of AMOR. First, we manually tune the weights to achieve the sim-to-real transfer of dynamic motions for a robotic character. Second, we explore the use of AMOR in a hierarchical setting, where a high-level policy (HLP) uses its adaptive capabilities to solve a novel task. We observe that reward trade-offs may not only need to vary across but also within skills. To automate this fine-grained selection of reward weights, the HLP dynamically adjusts the reward trade-offs. For training, we rely on a generator-discriminator approach [Ho and Ermon 2016; Peng et al. 2021; Xu and Karamouzas 2021].

By enabling adjustments of reward trade-offs without retraining, AMOR paves the way towards adaptive physics-based character control. In summary, our contributions are:

- A novel *context-conditioned* MORL problem formulation that enables the extraction of Pareto fronts conditioned on different contexts, with a single policy.
- AMOR, a controller conditioned on reward weights and task context, capable of zero-shot adaptation to desired trade-offs among conflicting objectives.
- A hierarchical policy that leverages AMOR for fine-grained real-time adjustments of reward weights, offering interpretability of implicit rewards as a byproduct.

2 Related Work

Physics-based Character Control. Early work in character control relied on carefully designed cost functions and optimization to synthesize lifelike locomotion and diverse skills [Al Borno et al. 2013; Coros et al. 2010; Hämäläinen et al. 2015; Hodgins et al. 1995; Lee et al. 2010; Mordatch et al. 2010, 2012; Sharon and van de Panne 2005; Sok et al. 2007].

The growing availability of motion capture data gradually shifted the field towards learning-based methods, where controllers learn from human motion rather than relying on hand-crafted optimization [Liu et al. 2015, 2012, 2010; Won et al. 2017]. Especially with the development of efficient RL algorithms [Mnih et al. 2016, 2015; Schulman et al. 2015a,b, 2017], neural network controllers gained traction through their ability to learn continuous motor policies from data [Heess et al. 2017, 2015; Merel et al. 2019; Peng et al. 2018a, 2017; Torabi et al. 2018a], although balancing style, realism, and robustness in reward design remains non-trivial.

While some methods have leveraged insights such as symmetry and energy efficiency in locomotion [Yu et al. 2018], predominant approaches define objectives that explicitly measure pose and velocity tracking accuracy [Bergamin et al. 2019; Peng et al. 2017]. Although this has been shown to scale beyond locomotion [Peng et al. 2018a,b], such methods have often attempted to craft a single reward function for all motions, which has proven challenging and consequently limited their applicability beyond a handful of motions or specific motion styles.

Various ideas have been explored to scale tracking controllers; combining expert policies [Merel et al. 2019, 2017, 2020; Peng et al.

2019; Won et al. 2020], incorporating future frames [Chentanez et al. 2018; Park et al. 2019], incrementally increasing motion complexity [Luo et al. 2023; Wang et al. 2020], model-based RL [Fussell et al. 2021; Yao et al. 2022], leveraging latent spaces [Gehring et al. 2023; Hasenclever et al. 2020; Serifi et al. 2024; Won et al. 2022; Zhu et al. 2023], or the use of advanced transformer-based architectures [Tessler et al. 2024]. Despite this progress, these methods continue to rely on fixed-weight reward design, which remains a critical bottleneck.

Concurrently, another line of research has attempted to sidestep handcrafted objectives by replacing explicit reward functions with learned implicit rewards [Peng et al. 2021; Xu and Karamouzas 2021]. Instead of explicitly computing deviations, these methods utilize a discriminator score, inspired by adversarial learning [Goodfellow et al. 2014; Ho and Ermon 2016; Torabi et al. 2018b], to differentiate between the reference and the controller-produced simulated motion. While they automate aspects of reward construction, they introduce training instabilities like mode collapse and require significant compute. Moreover, they lose the interpretability of explicit rewards and are usually used to design controllers that aim to embed a repertoire of diverse skills into the policy rather than tracking a specific reference motion well. Some limitations were improved by follow-up work [Dou et al. 2023; Tang et al. 2024; Tessler et al. 2023], but challenges remain.

Recently, efforts have been made to bring physics-based controllers to robots [Cheng et al. 2024; He et al. 2024; Serifi et al. 2024], where reward design has proven even more challenging [Gu et al. 2025; Ha et al. 2024; Ibarz et al. 2021]. Potential sim-to-real gaps can result in unexpected trade-offs, requiring an adjustment of reward functions that, in turn, necessitates a retraining of the controller from scratch.

Multi-Objective Reinforcement Learning (MORL). Multi-objective optimization and Pareto front extraction have seen use in interactive design exploration in graphics [Schulz et al. 2018]. In our work, we highlight applications in RL and physics-based character control instead.

MORL [Hayes et al. 2022] emerged as an extension of traditional RL [Sutton and Barto 2018] to address problems involving multiple conflicting objectives. Unlike standard RL, where a single policy is trained, agents are tasked to learn a *set* of policies, each specializing to a different prioritization between the reward functions. Early MORL work targeted the development of the underlying theory and the training of small sets of Pareto-optimal policies in synthetic environments [Rojiers et al. 2013; Van Moffaert and Nowé 2014].

Recent MORL work can be divided into techniques that explicitly maintain a population of policies [Felten et al. 2024; Xu et al. 2020], and methods that learn a single preference-conditioned model that encodes multiple policies [Alegre et al. 2023; Yang et al. 2019]. Our work falls into the second category, which scales better with the number of policies.

A few works have considered constrained MORL approaches for physics-based character motion [Kim et al. 2024; Wang et al. 2020]. In Wang et al. [2020], tasks are formulated using multi-objective reward functions. However, fixed weights and cost thresholds are employed, limiting the versatility of the resulting controllers. More

recent work models tasks by dividing them into pre-specified stages, defined by different rewards and cost functions [Kim et al. 2024]. However, this method requires an expert to manually specify the cost thresholds and reward weights for each pre-specified stage of a task. To employ multiple critics, Xu et al. [2023] use a MORL formulation where different groups of body parts are treated as independent tasks with their own value function. However, the authors only learn a single policy on the Pareto front, assigning fixed weights for all body groups. Our method, in contrast to the aforementioned works, learns trade-offs that cover the entire space of preferences.

Another setting related to MORL is the multi-task setting within the successor features (SFs) framework [Barreto et al. 2019, 2017; Dayan 1993]. In this line of work, the reward function is assumed to be a linear combination of a set of (learned) *features*. Alegre et al. [2022] showed connections between the MORL and SFs settings, demonstrating that ideas from both fields could be combined. While SF-based methods focus on the rapid adaptation to novel tasks, the goal in MORL—and in this paper—is to learn a set of Pareto-optimal solutions that capture all trade-offs between conflicting objectives.

3 AMOR Overview

Fig. 2 shows the structure of AMOR. At its core, it is an RL-based controller that tracks a kinematic reference motion, similarly to related work [Serifi et al. 2024]. The policy is trained to output actions a_t that maximize the reward of a simulated character, given the current state of the character s_t , and a context vector c_t encoding task-relevant information. Specifically, in our context, c_t includes a time-varying kinematic reference, and also a latent-space encoding of a motion window that captures past and future targets.

The problem specification also includes a set of reward terms r_t that capture the policy performance. Particularly, we consider a set of 7 rewards, including joint-space and task-space tracking, velocity tracking, and smoothness. Instead of prescribing a fixed set of reward weights, and giving the policy a scalar reward, we maintain a vector of rewards and condition the policy on the reward weights, so the policy becomes $\underline{\pi}(a_t | s_t, c_t, w)$.

For each training episode and environment, we sample a set of reward weights from a multi-dimensional simplex. Policy updates are then computed using a multi-objective extension of the PPO algorithm, which is also given this same weight vector. Once converged, this results in a policy which can track arbitrary reference motions under arbitrary weightings of different reward terms, allowing for on-the-fly weight tuning post-training by either a user or an algorithm.

4 Multi-Objective Reinforcement Learning

Before discussing AMOR and its HLP extension in more detail, we provide background on multi-objective reinforcement and introduce our algorithmic extensions.

4.1 Background

In standard RL [Sutton and Barto 2018], an agent interacts with its environment by selecting actions a_t based on the current state s_t following a policy $\pi(a_t | s_t)$. This action leads to a transition to a new

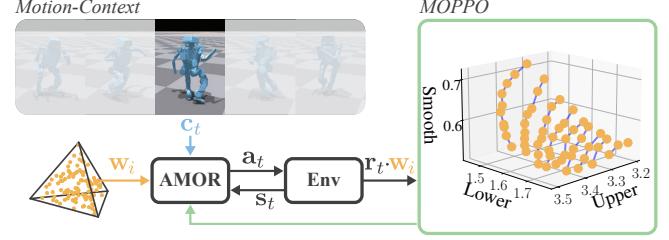


Fig. 2. AMOR Overview. AMOR optimizes for multiple objectives conditioned on state, motion-context, and reward weights, where reward weights are sampled from a multi-dimensional simplex. The environment provides a vector of rewards, which are then used by a multi-objective PPO algorithm together with the weights to update the policy.

state $s_{t+1} \sim p(\cdot | s_t, a_t)$ and yields a scalar reward $r_t = r(s_t, a_t, s_{t+1})$. The agent’s goal is to maximize the expected discounted return,

$$J(\pi) = \mathbb{E}_{d_0} [V^\pi(s_0)] = \mathbb{E}_\pi \left[\sum_{t \geq 0} \gamma^t r_t \mid s_0 \sim d_0 \right], \quad (1)$$

where $\gamma \in [0, 1]$ is the discount factor, $V^\pi(s)$ is the value function, and d_0 an initial state distribution.

MORL [Hayes et al. 2022] extends this framework by providing the agent with a vector-valued reward $r_t(s_t, a_t, s_{t+1}) \in \mathbb{R}^m$, where each element represents a distinct (and potentially conflicting) objective. Instead of a scalar return, each policy is associated with an expected vector return, $J(\pi) = \mathbb{E}_\pi [\sum_{t \geq 0} \gamma^t r_t \mid s_0 \sim d_0]$. In contrast to standard RL, no single optimal solution exists; instead, the goal of an agent is to identify a *Pareto front*, \mathcal{F} . We say a point $J(\pi)$ is *Pareto non-dominated* if and only if there does not exist another point $J(\pi')$ such that $J_i(\pi') \geq J_i(\pi), \forall i$ and $J_i(\pi') > J_i(\pi)$ for at least one $i \in \{1, \dots, m\}$. In the context of continuous robotic control tasks [Felten et al. 2023; Xu et al. 2020], the Pareto Front is typically convex¹ and can be defined in terms of a linear dominance relation

$$\mathcal{F} = \{J(\pi) \mid \exists w \text{ s.t. } J(\pi) \cdot w \geq J(\pi') \cdot w, \forall \pi'\}, \quad (2)$$

where the elements w_i in the reward weight vector $w \in \Delta^m$ form a convex combination, satisfying the requirements $\sum_{i=1}^m w_i = 1$ and $w_i \geq 0$. Intuitively, each w induces a different scalar reward $r_t = r(s_t, a_t, s_{t+1}) \cdot w$. Due to the linearity of the expectation and sum operations, it follows that

$$J(\pi) = \mathbb{E}_\pi \left[\sum_{t \geq 0} \gamma^t r_t \cdot w \right] = \mathbb{E}_\pi \left[\sum_{t \geq 0} \gamma^t r_t \right] \cdot w = J(\pi) \cdot w, \quad (3)$$

with the optimal solution $J^* = \max_\pi J(\pi) \cdot w$. The corresponding policy π^* is therefore optimal for any trade-off w between the m objectives.

4.2 Algorithmic Extensions

We extend the standard MORL framework by conditioning policies, $\pi(a_t | s_t, c_t, w)$, and rewards, $r_t(s_t, a_t, s_{t+1}, c_t)$, on an additional context vector c_t that encodes task-relevant information, making the Pareto front also a function of the context. Given a context, each

¹In the MORL literature, the Pareto front under linear preferences is also known as the *convex coverage set* (CCS)[Rojers et al. 2013].

$\mathbf{w} \in \Delta^m$ then induces a different Pareto-optimal policy within the Pareto front.

To train π , we introduce a multi-objective extension of the Proximal Policy Optimization (PPO) algorithm [Schulman et al. 2017], which we refer to as MOPPO. Instead of learning a scalar value function $V^\pi(\mathbf{s}, \mathbf{c}) = \mathbb{E}_\pi [\sum_{t \geq 0} \gamma^t r_t \mid \mathbf{s}_0 = \mathbf{s}, \mathbf{c}_0 = \mathbf{c}]$, as done in traditional goal-conditioned RL, MOPPO learns a vector-valued function conditioned on the reward weights \mathbf{w} ,

$$V^\pi(\mathbf{s}, \mathbf{c}, \mathbf{w}) = \mathbb{E}_\pi \left[\sum_{t \geq 0} \gamma^t \mathbf{r}_t \mid \mathbf{s}_0 = \mathbf{s}, \mathbf{c}_0 = \mathbf{c} \right], \quad (4)$$

optimizing multiple objectives simultaneously. The multi-objective policy gradient, $\nabla_\pi [\mathbf{J}(\pi) \cdot \mathbf{w}]$, used to update the policy,

$$\mathbb{E}_{d^\pi} \left[\sum_{t \geq 0} (\mathbf{A}^\pi(\mathbf{s}_t, \mathbf{c}_t, \mathbf{a}_t) \cdot \mathbf{w}) \nabla_\pi \log \pi(\mathbf{a}_t | \mathbf{s}_t, \mathbf{c}_t, \mathbf{w}) \right], \quad (5)$$

relies on a vector-valued advantage function $\mathbf{A}^\pi(\mathbf{s}_t, \mathbf{c}_t, \mathbf{a}_t)$. Here, d^π represents the discounted stationary distribution of states induced by π and the environment dynamics, $\mathbf{s}_{t+1}, \mathbf{c}_{t+1} \sim p(\cdot | \mathbf{s}_t, \mathbf{c}_t, \mathbf{a}_t)$. As is common practice in PPO implementations [Andrychowicz et al. 2021], we employ generalized advantage estimation (GAE) and normalize the scalarized advantage, $\mathbf{A}^\pi \cdot \mathbf{w}$, with its mean and standard deviation over each mini-batch. Moreover, we employ the standard PPO clipped loss function, but constructed based on Eq. (5).

To ensure coverage of the space of possible reward weights and the Pareto front, we assign a different randomly-sampled weight vector $\mathbf{w} \sim \Delta^m$ to every episode. As a result, the agent’s replay buffer stores experience transitions of the form $(\mathbf{s}_t, \mathbf{c}_t, \mathbf{a}_t, \mathbf{r}_t, \mathbf{s}_{t+1}, \mathbf{c}_{t+1}, \mathbf{w})$.

To sample weight vectors uniformly from the $(m-1)$ -dimensional simplex Δ^m , we draw from a Dirichlet distribution with parameter $\alpha = 1$.

5 Training of AMOR Policy

To instantiate AMOR for character control tasks, we train a multi-objective policy to track reference motions under multiple, potentially conflicting objectives. This section details how we formulate the tracking problem, construct the motion context, define the multi-objective reward, and incorporate the reward weights into our policy.

Motion Tracking Problem. We consider the standard problem of tracking kinematic motion using a physically simulated character. As in VMP [Serifi et al. 2024], we assume access to a dataset \mathcal{D} of motion clips consisting of a finite sequence of motion frames, $\mathbf{m}_t = (h_t, \theta_t, \mathbf{v}_t, \mathbf{q}_t, \dot{\mathbf{q}}_t, \mathbf{p}_t, \dot{\mathbf{p}}_t)$. Here, h_t is the height of the character’s root relative to the ground, θ_t is the orientation of the root in a 6D representation [Xiang and Li 2020; Zhou et al. 2019], and \mathbf{v}_t is a 6D vector representing the root’s linear and angular velocities. Moreover, \mathbf{q}_t is the angular position of the character’s joints and $\dot{\mathbf{q}}_t$ their angular velocities. Finally, \mathbf{p}_t is a 9D vector that encodes the poses of hands and feet relative to the root (3D position, 6D orientation) and $\dot{\mathbf{p}}_t$ encodes the corresponding linear velocities. To make the motion invariant to the global pose, we normalize frames \mathbf{m}_t by expressing all orientations and velocities with respect to the local heading frame of the root θ_t .

Our goal is to choose actions \mathbf{a}_t to match these reference poses as closely as possible, subject to realistic physics constraints. However, perfect tracking of reference motion in the dataset is generally infeasible. Largely due to an ill-posed retargeting from human mocap to our characters, there are artifacts such as imbalance, inter-penetrations with the ground, or foot sliding that are clearly visible to the naked eye. Thus, certain aspects of a kinematic reference may need to be prioritized to maintain dynamic feasibility, creating conflicting objectives.

Motion Context. To condition the policy on the task, we define the *motion context* $\mathbf{c}_t = (\mathbf{m}_t, \mathbf{z}_t)$, where we add a latent vector \mathbf{z}_t that represents a compressed motion window of frames $\mathbf{M}_t = \{\mathbf{m}_{t-W}, \dots, \mathbf{m}_{t+W}\}$, centered at t and of size $2W + 1$, alongside the current motion frame. For training of this latent representation, we follow VMP [Serifi et al. 2024] and train a Variation Autoencoder (VAE) that maps motion windows to latent codes $\mathbf{z}_t = e(\mathbf{M}_t)$. This latent representation captures local motion patterns around the frame t . We refer to [Serifi et al. 2024] for details on the loss function, the training of the VAE, and the normalization of motion windows.

Multi-Objective Tracking Reward. We use a multi-objective reward vector $\mathbf{r} \in \mathbb{R}^m$ with $m = 7$ motion tracking objectives

$$\mathbf{r}(\mathbf{s}_t, \mathbf{a}_t, \mathbf{s}_{t+1}, \mathbf{c}_t) = [r_t^{\text{up}}, r_t^{\text{lo}}, r_t^{\text{feet}}, r_t^{\text{rbs}}, r_t^{\text{root}}, r_t^{\text{vel}}, r_t^{\text{smooth}}]^\top. \quad (6)$$

The reward r^{up} tracks the character’s upper joint positions and its height, r^{lo} the lower joint positions, r^{feet} the positions of the ankle joints, and r^{rbs} the position and orientation of the end-effectors. Reward r^{root} tracks the orientation of the root, r^{vel} the linear and angular velocities of the joints and root, and r^{smooth} penalizes high action rates and torques to mitigate vibrations and smoothen the motion.

Since the rewards may vary significantly in magnitude, directly summing them up using linear weights could lead to poor problem conditioning. We therefore assign a prior scaling to each reward term, informed by [Serifi et al. 2024].

The individual reward terms and their prior scaling factors are detailed in Tab. 1. All objectives include a constant survival bonus, c^{alive} , that rewards the agent for not reaching a terminal state (e.g., falling to the ground). This is important for training to prevent the policy from terminating as quickly as possible to avoid any negative accumulation of reward.

Weight Conditioning. In contrast to traditional motion tracking policies, AMOR is additionally conditioned on reward weight vectors \mathbf{w} . For each environment and each episode, we sample weights in the simplex Δ^m and keep them fixed while the motion context moves forward in time but also jumps to new motions. After collecting enough samples, gradient steps are performed to update the policy according to the MOPPO objective (Eq. 5).

6 Hierarchical Weight Adjustment

AMOR’s policy, described in the previous section, results in different behavior, dependent on the input weight vector \mathbf{w} . This allows the user to flexibly select the desired trade-offs in a *zero-shot* manner, without having to train different agents from scratch. In this section, we alternatively propose a *high-level policy* (HLP), $\bar{\pi}(\mathbf{w}_t | \mathbf{s}_t, \mathbf{c}_t)$,

Table 1. Multi-Objective Components. The individual terms and scales of each of the seven objectives. Scaling values are reported separately for the humanoid and the robot. $\mathbf{q}^{\text{uplofeet}}$ denote the DoFs for the upper body, lower body, and feet, respectively. $\mathbf{p}^{\text{posrot}}$ represent the positional and rotational parts of the rigid body poses, respectively. $\mathcal{R} : \mathbb{R}^n \rightarrow SO(3) \subset \mathbb{R}^{3 \times 3}$ represents the transformation that maps quaternions or 6D rotation representations to their corresponding rotation matrix in $SO(3)$. $\mathbf{v}^{\text{linang}}$ respectively denote the linear and angular velocity of the root. $\boldsymbol{\tau}$ denotes joint torques and $\ddot{\mathbf{q}}$ joint accelerations. Target values are denoted by (\cdot) .

Objective	Term(s)	Scale	
		Humanoid	Robot
r^{up}	$\ \mathbf{q}^{\text{up}} - \hat{\mathbf{q}}^{\text{up}}\ _2^2$	1.0	7.0
r^{lo}	$\ \mathbf{q}^{\text{lo}} - \hat{\mathbf{q}}^{\text{lo}}\ _2^2$	1.0	7.0
r^{feet}	$\ \mathbf{q}^{\text{feet}} - \hat{\mathbf{q}}^{\text{feet}}\ _2^2$	1.0	7.0
r^{rhs}	$\ \mathbf{p}^{\text{pos}} - \hat{\mathbf{p}}^{\text{pos}}\ _2^2$ $\ \mathcal{R}(\mathbf{p}^{\text{rot}}) - \mathcal{R}(\hat{\mathbf{p}}^{\text{rot}})\ _2^2$ $\ \dot{\mathbf{p}} - \hat{\dot{\mathbf{p}}}\ _2^2$	1.0 1.0 1.0	1.0 1.0 1.0
r^{root}	$\ \mathcal{R}(\theta) - \mathcal{R}(\hat{\theta})\ _2^2$	1.0	1.0
r^{vel}	$\ \mathbf{v}^{\text{lin}} - \hat{\mathbf{v}}^{\text{lin}}\ _2^2$ $\ \mathbf{v}^{\text{ang}} - \hat{\mathbf{v}}^{\text{ang}}\ _2^2$	1.0 1.0	2.0 2.0
r^{smooth}	$-\ \boldsymbol{\tau}\ _2^2$ $-\ \mathbf{a}_t - \mathbf{a}_{t-1}\ _2^2$ $-\ \mathbf{a}_t - 2\mathbf{a}_{t-1} + \mathbf{a}_{t-2}\ _2^2$ $-\ \ddot{\mathbf{q}}\ _2^2$	$1.0 \cdot 10^{-5}$ $1.0 \cdot 10^{-5}$ $1.0 \cdot 10^{-5}$ $1.0 \cdot 10^{-6}$	$1.0 \cdot 10^{-4}$ 1.5 0.45 $2.5 \cdot 10^{-6}$

which dynamically selects weights \mathbf{w}_t during execution, enabling the agent to adapt its behavior to the current context and dynamic state based on a different, high-level reward. This hierarchical approach uses AMOR with frozen parameters as its low-level policy and is illustrated in Fig. 3. We train the HLP using standard PPO, with the addition of a softmax activation function in the actor network’s final layer to ensure it outputs reward weights in the simplex, i.e., $\mathbf{w}_t \in \Delta^m$. We note that training the HLP is significantly faster than training AMOR, as is typically the case in hierarchical methods [Barreto et al. 2019].

Implicit Reward. The HLP is trained on a reward function, which is not required to be part of the original reward terms. To demonstrate this, we use an implicit reward, as proposed by recent work [Peng et al. 2021; Tessler et al. 2023]: a discriminator tries to distinguish simulated motions from the kinematic reference motions, and a reward is computed based on accuracy. The hierarchical weight adjustment method allows the expansion of AMOR to optimize for such new rewards.

We define $\mathbf{O}_t = \{\mathbf{o}_{t-V}, \dots, \mathbf{o}_t\}$ as a window of size V , containing past observations $\mathbf{o}_t = (\theta_t, \mathbf{v}_t, \mathbf{q}_t)$. Let $D(\mathbf{O}_t | \mathbf{z}_t)$ be a discriminator whose goal is to distinguish between dataset transitions $\hat{\mathbf{O}}_t \sim d^M(\hat{\mathbf{O}}_t, \mathbf{z}_t)$ and transitions resulting from following a given policy, $\mathbf{O}_t \sim d^\pi(\mathbf{O}_t, \mathbf{z})$, where $d^M(\hat{\mathbf{O}}_t, \mathbf{z}_t)$ and $d^\pi(\mathbf{O}_t, \mathbf{z}_t)$ are state transition distributions of the reference motion and the policy, respectively. The discriminator-based reward function used to train the high-level policy is then

$$r^D(\mathbf{O}_t, \mathbf{z}_t) = -\log(1 - D(\mathbf{O}_t | \mathbf{z}_t)). \quad (7)$$

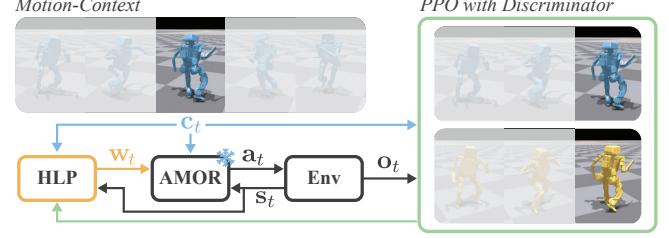


Fig. 3. High-Level Policy Overview. We learn a high-level policy (HLP) that generates reward weights for a pretrained AMOR based on the current motion context. In this stage, a discriminator is trained to distinguish between reference and simulated motions, with its output serving as an implicit reward for the HLP.

Intuitively, this reward function rewards the agent for performing transitions that appear indistinguishable from the real motion transitions in the dataset \mathcal{D} . By maximizing this reward function, $\bar{\pi}$ allows the agent to select the reward weights \mathbf{w}_t that lead to more realistic motion transitions for each \mathbf{s}_t and motion context \mathbf{c}_t .

We follow Tessler et al. [2023] and also condition the discriminator on the latent motion vector \mathbf{z}_t to prevent mode collapse. The discriminator is trained via the loss function

$$L^D = -\mathbb{E}_{\mathbf{M}_t \in \mathcal{D}} \left[L^M + L^\pi + c^{\text{GP}} L^{\text{GP}} \mid \mathbf{z}_t = e(\mathbf{M}_t) \right], \quad (8)$$

with terms

$$\begin{aligned} L^M &= \mathbb{E}_{d^M(\hat{\mathbf{O}}_t, \mathbf{z}_t)} \log D(\hat{\mathbf{O}}_t | \mathbf{z}_t) \\ L^\pi &= \mathbb{E}_{d^\pi(\mathbf{O}_t, \mathbf{z}_t)} \log(1 - D(\mathbf{O}_t | \mathbf{z}_t)) \\ L^{\text{GP}} &= \mathbb{E}_{d^M(\hat{\mathbf{O}}_t, \mathbf{z}_t)} \|\nabla_\phi D(\phi) \mid_{\phi=(\hat{\mathbf{O}}_t, \mathbf{z}_t)}\|^2, \end{aligned}$$

which has shown to minimize the Jensen-Shannon divergence between $d^M(\hat{\mathbf{O}}_t, \mathbf{z}_t)$ and $d^\pi(\mathbf{O}_t, \mathbf{z}_t)$ [Nowozin et al. 2016]. L^{GP} is a gradient penalty, scaled by coefficient c^{GP} , used to penalize nonzero gradients on samples from the dataset. This has been shown to improve training stability [Mescheder et al. 2018].

Note that attempts to directly use the discriminator-based reward in the low-level policy led to mode collapse; the presented two-stage approach seems better suited for navigating the complex landscapes created by such higher-level rewards.

Reward Interpretability. After training the HLP on the high-level reward, we can inspect the weights it selects. This shows which combination of low-level reward terms correspond to a behavior that maximizes the implicit reward. The combination of the HLP and AMOR thus allows us to interpret what the discriminator is looking for at a given state and context. For additional artistic control, a user could also edit the weights returned by the HLP.

7 Evaluation and Results

In this section, we evaluate our method’s capabilities for the problem of motion tracking of physically-based characters and robots. In particular, we aim to validate that (i) AMOR is able to approximate the Pareto front of tracking behavior for different motions with a single policy, allowing for behavior tuning without retraining; (ii) the relative importance between tracking objectives plays

a significant role in the resulting tracking behavior; and (iii) by employing our high-level policy, $\bar{\pi}(w|s_t, c_t)$, we can dynamically prioritize different objectives resulting in more flexible and robust tracking behavior.

7.1 Experimental Setting

Characters. We perform our experiments on a standard humanoid with 36 degrees of freedom (DoFs) and a bipedal robot with 20 DoFs. The characters are torque-controlled using a proportional-derivative (PD) controller with realistic actuator models for the robot [Grandia et al. 2024] and virtual actuators for the humanoid [Serifi et al. 2024].

Dataset. The dataset consists of motion capture data from a simple mocap setup (CMU [2001], 1870 clips, 8.5 h) as well as a smaller high-quality dataset of processed motions (Reallusion [2023], 214 clips, 0.5 h).

RL. We employ multi-layer perceptron (MLP) neural networks with ELU activations [Clevert et al. 2016] to model the policies and value functions. We use 4 layers with 1024 units to model AMOR $\underline{\pi}$ and the critic, and a 3-layer model for the high-level policy $\bar{\pi}$. Both policies operate at 50 Hz. We normalize the observations using a running mean, as typically done when using PPO [Andrychowicz et al. 2021]. Our simulations are conducted using the GPU-accelerated Isaac Gym [Makoviychuk et al. 2021] simulator, running 8192 environment instances in parallel at 250 Hz on a single RTX 4090 GPU. We train AMOR for 300k iterations (approximately 5 days) for each character.

7.2 AMOR

Pareto Front. First, we show a visualization of the Pareto front identified by AMOR’s policy, when given different motions to track. In particular, we evaluate $\tilde{\mathcal{F}} = \{J(\underline{\pi}(\cdot, w)) \mid w \sim \Delta^m\}$ by sampling 8192 weight vectors from Δ^m and averaging over 15 episodic returns. Fig. 4 displays the Pareto fronts obtained by tracking three different motions (Idle, Walking, Dancing) on the humanoid by following $\underline{\pi}$. Because each point $J(\underline{\pi})$ on the Pareto front is a 7-dimensional vector, we depict pairwise comparisons between the unscaled cumulative reward corresponding to each objective. All of the 8192 points are Pareto non-dominated w.r.t. all objectives, and the points with a black border are points that are additionally Pareto non-dominated w.r.t. the two objectives in the corresponding figure panel. We also depict with crosses the mean return obtained by following $\underline{\pi}$ with equal reward weights, $w_i = 1/m$. Although the reward terms share the overall goal of achieving better motion tracking, they are inherently conflicting. For example, for the dancing motion in Fig. 4, we observe that prioritizing smoothness conflicts with lower-body tracking. Precisely tracking more dynamic reference motions, which are generally less feasible, introduces jitter. This trade-off is also visible on the physical system, see Fig. 7. Even seemingly unrelated objectives, such as upper- and lower-body tracking, can conflict because the coordination between the body parts is not physically accurate. We note that different motions induce different Pareto fronts with varying degrees of conflict between objectives. This suggests that there is indeed value in not relying on fixed weights when tracking multiple motions.

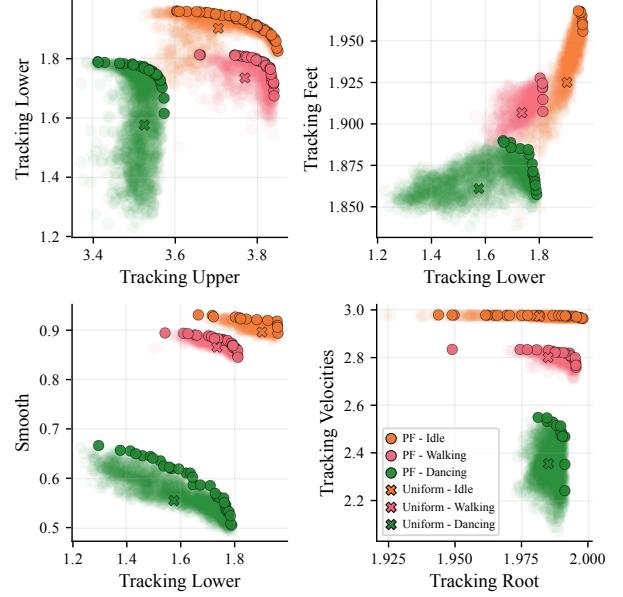
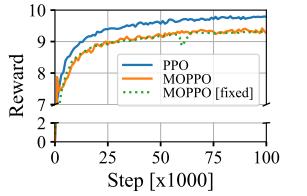


Fig. 4. **Pareto Fronts (PFs).** Visualization of selected PFs generated by tracking three distinct motion types—Idle, Walking, and Dancing—using the humanoid controlled by AMOR’s policy $\underline{\pi}$. x-markers indicate performance under equal weight configuration, corresponding to a fixed-reward policy.



Training Comparison. We compare the total reward obtained during training between AMOR trained with MOPPO and a policy trained with PPO, using the fixed average weights, $w_i = 1/m$. For MOPPO, we evaluate the reward using both randomly sampled weights and the fixed weights used in PPO training. In all cases, we report the sum of equally weighted reward terms. As shown in the inset figure, the training follows a similar trend, with PPO converging faster. This faster convergence is expected, as PPO optimizes for a *single* fixed reward weight combination, while MOPPO must dynamically adapt its behavior to varying weights covering the entire simplex Δ^m , making it a harder learning task. We hypothesize that the gap in total reward between PPO and MOPPO would further reduce after both algorithms fully converge. Notably, MOPPO’s reward evaluated for the average weights is similar to the average reward across the randomized weights.

Behavior Adaptation. To validate that changes in weights indeed lead to a change in behavior, we manually change the weights for the humanoid while performing a dancing motion. The results are best seen in the supplementary video. Fig. 5 highlights keyframes of the experiment. It can be seen that with the increased smoothness term, a smoother motion is obtained at the expense of tracking performance, as would be expected. We further evaluate this observation on the full dataset; Fig. 6 shows the distribution of the unweighted

cumulative reward for smoothness when increasing the corresponding reward weight. The change in the distribution shows that the policy can successfully adapt its behavior to prioritize smoothness.

Robot. We use AMOR on a 20-DoF bipedal robot (Fig. 1), and compare a uniform weight distribution against manually-tuned weights on two challenging examples.

We first consider a stationary dancing motion, and compare the simulated robot to the physical one. As seen in Fig. 7, the real robot exhibits significantly higher jitter in the joint positions and velocities. By increasing the smoothness objective, we can reduce this jitter, and thereby also reduce the sim-to-real gap as shown in the figure. This experiment is highlighted in the supplementary video and best appreciated by focusing on the actuator sound.

In the second example, a double pirouette motion, shown in Fig. 1 and the accompanying video, we demonstrate that tuning the reward weights on the real system enables the robot to perform physically demanding motions beyond the capabilities of the state-of-the-art

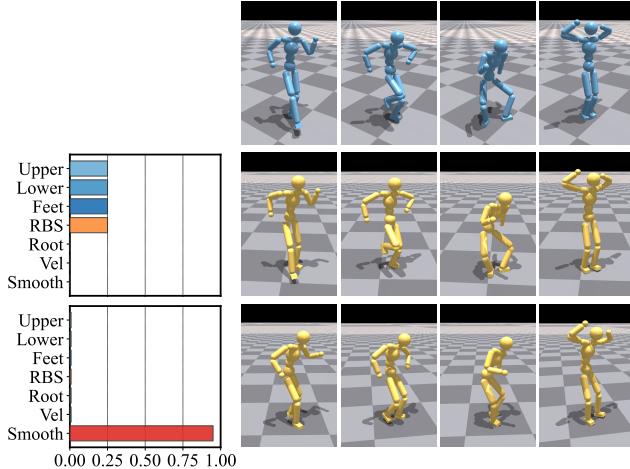


Fig. 5. **Weight Influence.** (Top) Kinematic reference. (Middle) Visual performance when prioritizing tracking reward weights. (Bottom) Visual performance when prioritizing smoothness reward terms.

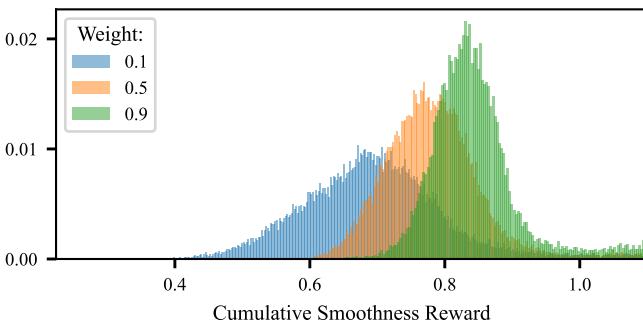


Fig. 6. **Prioritizing Objectives.** Distributions of the unweighted cumulative smoothness reward for three distinct weight values, each evaluated on 32768 episodes.

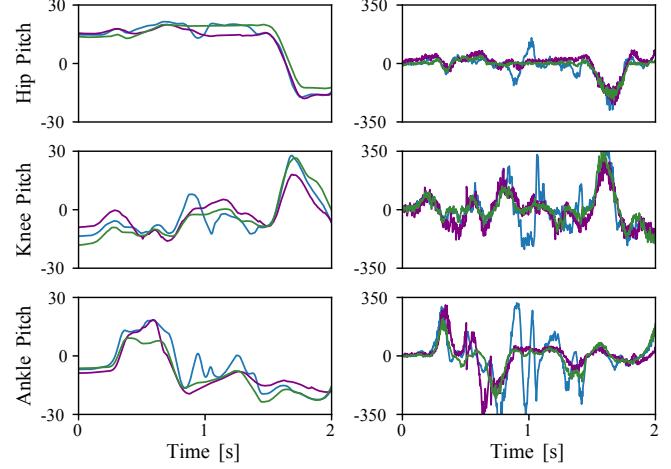


Fig. 7. **Weight Tuning.** The left column shows the measured DoF [$^{\circ}$], and the right column shows the measured DoF velocities [$^{\circ}/\text{s}$] for the three pitch joints of the robot’s right leg during the dance motion in the supplementary video. The purple curve shows the measurements of the simulated robot and the blue curve of the real robot with uniform weight distribution. Through tuning the weights and increasing the smoothness objective, the sim-to-real gap and undesired jitter can be reduced, as shown with the green curve. See also the supporting video.

fixed-weight VMP controller. To achieve this motion, we experimentally determined the impact of each reward term through on-the-fly weight adjustments directly on the robot.

We found that time-varying weights were required, to address particular challenges of different parts of the motion. For example, a high velocity weight was required during the initial part of the motion, to build up enough rotational speed for a stable pirouette. Conversely, a higher smoothing weight was required towards the end of the sequence, for a smoother transition out of the pirouette.

Tuning the weights for this dynamic motion took approximately 1 day of experimentation. Compared to the 5 days it takes to train the RL policy, it is clear that tuning with retraining would be infeasible.

7.3 High-Level Policy

Hierarchical Weight Adjustment. We now evaluate the behavior of the proposed HLP, in particular studying how the weights returned by the HLP vary over time and with the motion being performed. In Fig. 8, we show how the reward weights produced by $\bar{\pi}(w_t | s_t, c_t)$ evolve as the humanoid transitions through walking, spinning, performing an upper-body motion, and standing still.

During walking (red background), the HLP balances the feet and lower body coordination combined with smoothing to feather the steps of the character. The activations of the velocity term are generally short and intermittent; and can for example be seen to align with the start of a spinning motion (yellow window). The motion in the green window features two high punches in the air; this can be seen to correlate with two spikes in the tracking weights (blue curves). During the final part of the motion (purple background) the reference motion remains at rest; this can be seen to activate the velocity weight.

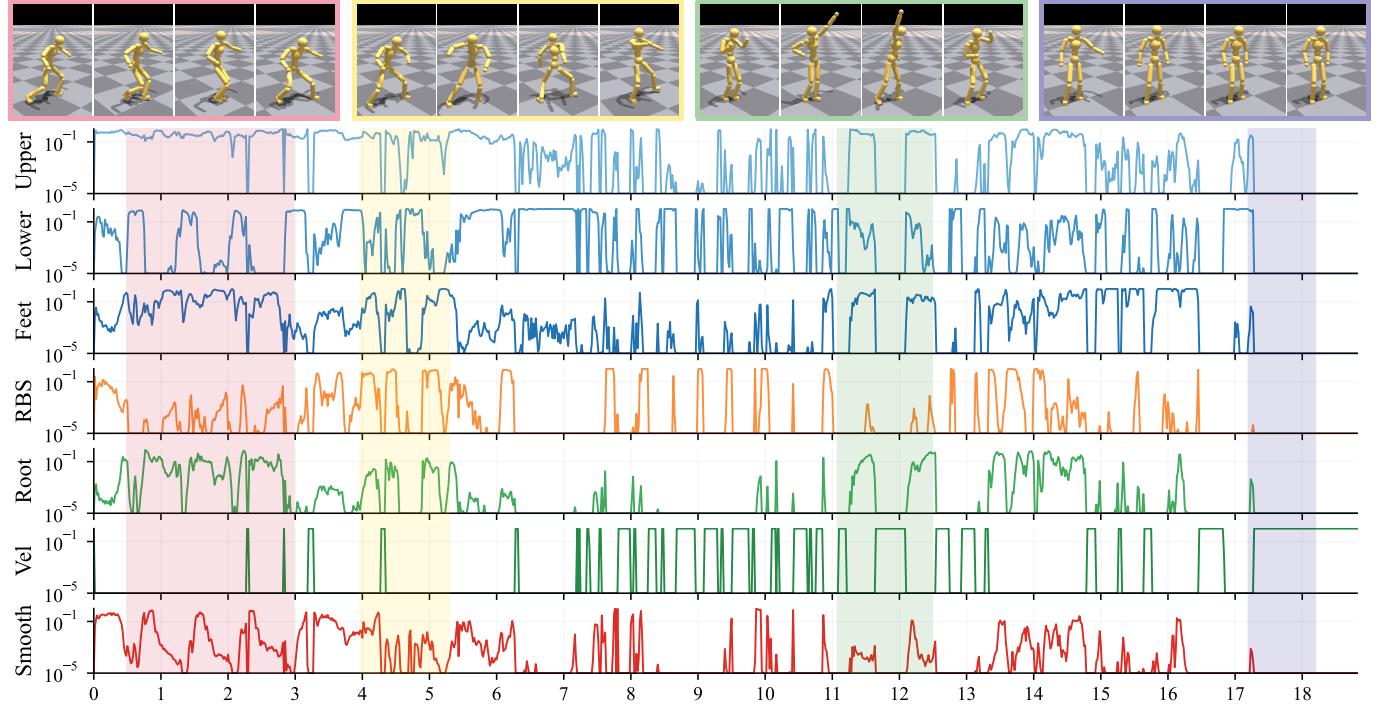


Fig. 8. **Adaptive Weights.** Visualization of the behavior of AMOR’s high-level policy $\bar{\pi}(w_t | s_t, c_t)$ through different types of motion. The resulting reward weights $w_t \in \Delta^m$ are plotted on a logarithmic scale; this better captures their multiplicative effect on the reward.

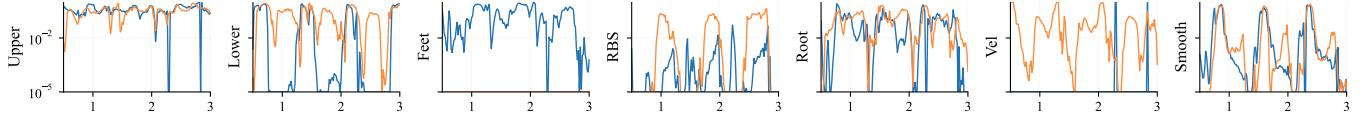


Fig. 9. **Discriminator Window Size Comparison** between 30-frame and 2-frame discriminator rewards on the HLP for the first interval of Fig. 8. The different discriminator window lengths lead to different reward weights.

Interestingly, we observe that changing the window size V of the observations for the discrimination produces different behaviors. Fig. 9 shows a comparison between the 30-frame discriminator and a 2-frame discriminator. While it is difficult to draw conclusions, we can observe that the 2-frame version weighs the velocity term more heavily, while the 30-frame version instead places higher weights on tracking the lower body and feet.

Motion Tracking Performance. We evaluate the HLP by comparing humanoid performance in simulation using HLP’s weight predictions against equally-balanced fixed weights. First, we measure the mean absolute error (MAE) of joint positions relative to a kinematic reference, using 80’000 randomly-sampled 30-second motion episodes. For shorter motions, additional segments are appended to complete the episode. Second, we assess the logits of a discriminator $D(O_t | z_t)$ trained on 32M 30-frame windows O_t collected from simulations with HLP, fixed weights, and dataset transitions. The discriminator, trained to distinguish between simulated and dataset transitions, is evaluated on 8M windows. Note that this discriminator is trained separately for this evaluation, and is not the

Table 2. **Motion Tracking Performance.** Comparison of AMOR’s performance on the simulated humanoid using HLP against equally balanced fixed weights is measured both explicitly, using the mean absolute error (MAE) of the character’s joint pose in degrees, and implicitly, through the logits of a discriminator.

Weights	MAE ↓	Logits ↑
Uniform Weights	10.02	-18.02
HLP Prediction	9.55	-15.48

discriminator used to train the HLP. Results in Tab. 2 show significant improvements in both explicit MAE and implicit assessment through logits when using HLP over fixed weights. This highlights that the hierarchical approach is indeed able to leverage the adaptability of the AMOR policy.

8 Conclusions

We have introduced AMOR, a policy conditioned on context and a linear combination of reward weights trained with MORL. We

have shown that a multi-objective approach scales to the problem complexity found in motion tracking for physics-based characters and robots. By leveraging context conditioning, AMOR can successfully track a wide variety of motions under different reward weights. AMOR allows for on-the-fly adjustment of reward weights after training, unlocking new possibilities, some of which we have explored in this work. While our work targets character control, we expect the adaptive weight tuning enabled by AMOR to be applicable in other domains as well.

We demonstrated the use of AMOR for sim-to-real transfer by tuning rewards to reduce the sim-to-real gap, leveraging on-the-fly reward adjustments and time-varying rewards to push the boundaries of dynamic robot motions.

Moreover, we explored a hierarchical extension to AMOR, by training a HLP which dynamically adjusts reward weights over time to closely match a given motion reference, using a discriminator reward. An interesting direction for future work is to explore alternative higher-level task-based rewards for the HLP.

Another potential application is automated fine-tuning using robot-in-the-loop online reinforcement learning (ORL) [Wu et al. 2022]. Using AMOR as a low-level policy could reduce training time, and mitigate the high failure rates observed early in training.

Interestingly, some performance gap is apparent between regular PPO and MOPPO due to the increased problem complexity. In future work, this could potentially be reduced through large-scale training until convergence, or through alternative MORL techniques, e.g., by employing different strategies for sampling weight vectors during training [Alegre et al. 2023].

References

- Mazen Al Borno, Martin de Lasa, and Aaron Hertzmann. 2013. Trajectory Optimization for Full-Body Movements with Complex Contacts. *IEEE Transactions on Visualization and Computer Graphics* 19, 8 (2013), 1405–1414. doi:10.1109/TVCG.2012.325
- Lucas N. Alegre, Ana L. C. Bazzan, and Bruno C. da Silva. 2022. Optimistic Linear Support and Successor Features as a Basis for Optimal Policy Transfer. In *Proceedings of the 39th International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 162)*, Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvári, Gang Niu, and Sivan Sabato (Eds.). PMLR, 394–413.
- Lucas N. Alegre, Ana L. C. Bazzan, Diederik M. Roijers, Ann Nowé, and Bruno C. da Silva. 2023. Sample-Efficient Multi-Objective Learning via Generalized Policy Improvement Prioritization. In *Proceedings of the 2023 International Conference on Autonomous Agents and Multiagent Systems* (London, United Kingdom) (AAMAS '23). International Foundation for Autonomous Agents and Multiagent Systems, Richland, SC, 2003–2012.
- Marcin Andrychowicz, Anton Raichuk, Piotr Stańczyk, Manu Orsini, Sertan Girgin, Raphaël Marinier, Leonard Hussenot, Matthieu Geist, Olivier Pietquin, Marcin Michalski, Sylvain Gelly, and Olivier Bachem. 2021. What Matters for On-Policy Deep Actor-Critic Methods? A Large-Scale Study. In *Proceedings of the Ninth International Conference on Learning Representations*. <https://openreview.net/forum?id=nlAxjsnIDzg>
- André Barreto, Diana Borsa, Shaobo Hou, Gheorghe Comanici, Eser Akgün, Philippe Hamel, Daniel Toyama, Jonathan Hunt, Shibli Mourad, David Silver, and Doina Precup. 2019. The option keyboard combining skills in reinforcement learning. In *Proceedings of the 33rd International Conference on Neural Information Processing Systems*. Curran Associates Inc., Red Hook, NY, USA, Article 1169, 11 pages.
- André Barreto, Will Dabney, Rémi Munos, Jonathan J Hunt, Tom Schaul, Hado P van Hasselt, and David Silver. 2017. Successor features for transfer in reinforcement learning. *Advances in neural information processing systems* 30 (2017).
- Kevin Bergamin, Simon Clavet, Daniel Holden, and James Richard Forbes. 2019. DReCon: data-driven responsive control of physics-based characters. *ACM Trans. Graph.* 38, 6, Article 206 (Nov. 2019), 11 pages. doi:10.1145/3355089.3356536
- Xuxin Cheng, Yandong Ji, Junming Chen, Ruihan Yang, Ge Yang, and Xiaolong Wang. 2024. Expressive whole-body control for humanoid robots. *arXiv preprint arXiv:2402.16796* (2024).
- Nuttapong Chentanez, Matthias Müller, Miles Macklin, Viktor Makovychuk, and Stefan Jeschke. 2018. Physics-based motion capture imitation with deep reinforcement learning. In *Proceedings of the 11th ACM SIGGRAPH Conference on Motion, Interaction and Games* (Limassol, Cyprus) (MIG '18). Association for Computing Machinery, New York, NY, USA, Article 1, 10 pages. doi:10.1145/3274247.3274506
- Djork-Arné Clevert, Thomas Unterthiner, and Sepp Hochreiter. 2016. Fast and Accurate Deep Network Learning by Exponential Linear Units (ELUs). In *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*, Yoshua Bengio and Yann LeCun (Eds.). <http://arxiv.org/abs/1511.07289>
- CMU. 2001. CMU Graphics Lab Motion Capture Database. <http://mocap.cs.cmu.edu/>
- Stelian Coros, Philippe Beaudoin, and Michiel van de Panne. 2010. Generalized biped walking control. *ACM Trans. Graph.* 29, 4, Article 130 (July 2010), 9 pages. doi:10.1145/1771565.1781156
- Peter Dayan. 1993. Improving Generalization for Temporal Difference Learning: The Successor Representation. *Neural Computation* 5, 4 (1993), 613–624. doi:10.1162/neco.1993.5.4.613
- Zhiyang Dou, Xuelin Chen, Qingnan Fan, Taku Komura, and Wenping Wang. 2023. C-ASE: Learning Conditional Adversarial Skill Embeddings for Physics-based Characters. In *SIGGRAPH Asia 2023 Conference Papers* (Sydney, NSW, Australia) (SA '23). Association for Computing Machinery, New York, NY, USA, Article 2, 11 pages. doi:10.1145/3610548.3618205
- Florian Felten, Lucas N. Alegre, Ann Nowé, Ana L. C. Bazzan, El-Ghazali Talbi, Grégoire Danoy, and Bruno C. da Silva. 2023. A Toolkit for Reliable Benchmarking and Research in Multi-Objective Reinforcement Learning. In *Advances in Neural Information Processing Systems* (New Orleans, USA), Vol. 36.
- Florian Felten, El-Ghazali Talbi, and Grégoire Danoy. 2024. Multi-Objective Reinforcement Learning based on Decomposition: A taxonomy and framework. *Journal of Artificial Intelligence Research* 79 (2024), 679–723.
- Levi Fussell, Kevin Bergamin, and Daniel Holden. 2021. SuperTrack: motion tracking for physically simulated characters using supervised learning. *ACM Trans. Graph.* 40, 6, Article 197 (Dec. 2021), 13 pages. doi:10.1145/3478513.3480527
- Jonas Gehring, Deepak Gopinath, Jungdam Won, Andreas Krause, Gabriel Synnaeve, and Nicolas Usunier. 2023. Leveraging Demonstrations with Latent Space Priors. *Transactions on Machine Learning Research* (2023). <https://openreview.net/forum?id=OzGlu4T4Cz>
- Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative adversarial nets. *Advances in neural information processing systems* 27 (2014).
- Ruben Grandia, Espen Knoop, Michael A Hopkins, Georg Wiedebach, Jared Bishop, Steven Pickles, David Müller, and Moritz Bächer. 2024. Design and control of a bipedal robotic character. In *Proceedings of Robotics: Science and Systems*.
- Zhaoyuan Gu, Junheng Li, Wenlan Shen, Wenhao Yu, Zhaoming Xie, Stephen McCrary, Xianyi Cheng, Abdulaziz Shamsah, Robert Griffin, C Karen Liu, et al. 2025. Humanoid Locomotion and Manipulation: Current Progress and Challenges in Control, Planning, and Learning. *arXiv preprint arXiv:2501.02116* (2025).
- Sehoon Ha, Joonho Lee, Michiel van de Panne, Zhaoming Xie, Wenhao Yu, and Majid Khadiv. 2024. Learning-based legged locomotion; state of the art and future perspectives. *arXiv preprint arXiv:2406.01152* (2024).
- Perttu Hämäläinen, Joon Rajamäki, and C. Karen Liu. 2015. Online control of simulated humanoids using particle belief propagation. *ACM Trans. Graph.* 34, 4, Article 81 (July 2015), 13 pages. doi:10.1145/2767002
- Leonard Hasenclever, Fabio Pardo, Raia Hadsell, Nicolas Heess, and Josh Merel. 2020. CoMic: Complementary Task Learning & Mimicry for Reusable Skills. In *Proceedings of the 37th International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 119)*, Hal Daumé III and Aarti Singh (Eds.). PMLR, 4105–4115. <https://proceedings.mlr.press/v119/hasenclever20a.html>
- Conor F. Hayes, Roxana Rădulescu, Eugenio Bargiacchi, Johan Källström, Matthew Macfarlane, Mathieu Reymond, Timothy Verstraeten, Luisa M. Zintgraf, Richard Dazeley, Fredrik Heintz, Enda Howley, Athirai A. Irissappane, Patrick Mannion, Ann Nowé, Gabriel Ramos, Marcello Restelli, Peter Vamplew, and Diederik M. Roijers. 2022. A practical guide to multi-objective reinforcement learning and planning. *Autonomous Agents and Multi-Agent Systems* 36, 1 (13 Apr 2022), 26. doi:10.1007/s10458-022-09552-y
- Tairan He, Wenli Xiao, Toru Lin, Zhengyi Luo, Zhenjia Xu, Zhenyu Jiang, Changliu Liu, Guanya Shi, Xiaolong Wang, Linxi Fan, and Yuke Zhu. 2024. HOVER: Versatile Neural Whole-Body Controller for Humanoid Robots. *arXiv preprint arXiv:2410.21229* (2024).
- Nicolas Heess, Dhruva Tb, Srinivasan Sriram, Jay Lemmon, Josh Merel, Greg Wayne, Yuval Tassa, Tom Erez, Ziyu Wang, SM Eslami, et al. 2017. Emergence of locomotion behaviours in rich environments. *arXiv preprint arXiv:1707.02286* (2017).
- Nicolas Heess, Gregory Wayne, David Silver, Timothy Lillicrap, Tom Erez, and Yuval Tassa. 2015. Learning Continuous Control Policies by Stochastic Value Gradients. In *Advances in Neural Information Processing Systems*, C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, and R. Garnett (Eds.), Vol. 28. Curran Associates, Inc. https://proceedings.neurips.cc/paper_files/paper/2015/file/

- 148510031349642de5ca0c544f31b2ef-Paper.pdf
- Jonathan Ho and Stefano Ermon. 2016. Generative adversarial imitation learning. *Advances in neural information processing systems* 29 (2016).
- Jessica K. Hodgins, Wayne L. Wooten, David C. Brogan, and James F. O'Brien. 1995. Animating human athletics. In *Proceedings of the 22nd Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH '95)*. Association for Computing Machinery, New York, NY, USA, 71–78. doi:10.1145/218380.218414
- Julian Ibarz, Jie Tan, Chelsea Finn, Mrinal Kalakrishnan, Peter Pastor, and Sergey Levine. 2021. How to train your robot with deep reinforcement learning: lessons we have learned. *The International Journal of Robotics Research* 40, 4-5 (2021), 698–721. doi:10.1177/0278364920987859
- Dohyeong Kim, Hyekjin Kwon, Junseok Kim, Gunmin Lee, and Songhwai Oh. 2024. Stage-Wise Reward Shaping for Acrobatic Robots: A Constrained Multi-Objective Reinforcement Learning Approach. arXiv:2409.15755 [cs.RO] <https://arxiv.org/abs/2409.15755>
- Yoongsang Lee, Sungeun Kim, and Jehee Lee. 2010. Data-driven biped control. In *ACM SIGGRAPH 2010 Papers* (Los Angeles, California) (*SIGGRAPH '10*). Association for Computing Machinery, New York, NY, USA, Article 129, 8 pages. doi:10.1145/183349.1781155
- Libin Liu, KangKang Yin, and Baining Guo. 2015. Improving Sampling-based Motion Control. *Computer Graphics Forum* 34, 2 (2015), 415–423. doi:10.1111/cgf.12571 arXiv:<https://onlinelibrary.wiley.com/doi/pdf/10.1111/cgf.12571>
- Libin Liu, KangKang Yin, Michiel van de Panne, and Baining Guo. 2012. Terrain runner: control, parameterization, composition, and planning for highly dynamic motions. *ACM Transactions on Graphics (TOG)* 31, 6 (2012), 154.
- Libin Liu, KangKang Yin, Michiel van de Panne, Tianjia Shao, and Weiwei Xu. 2010. Sampling-based contact-rich motion control. *ACM Trans. Graph.* 29, 4, Article 128 (July 2010), 10 pages. doi:10.1145/1778765.1778865
- Zhengyi Luo, Jinkun Cao, Alexander W. Winkler, Kris Kitani, and Weipeng Xu. 2023. Perpetual Humanoid Control for Real-time Simulated Avatars. In *International Conference on Computer Vision (ICCV)*.
- Viktor Makoviychuk, Lukasz Wawrzyniak, Yunrong Guo, Michelle Lu, Kier Storey, Miles Macklin, David Hoeller, Nikita Rudin, Arthur Allshire, Ankur Handa, and Gavriel State. 2021. Isaac Gym: High Performance GPU Based Physics Simulation For Robot Learning. In *Proceedings of The Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track*. https://openreview.net/forum?id=fgFBtYgJQX_
- Josh Merel, Leonard Hasenclever, Alexandre Galashov, Arun Ahuja, Vu Pham, Greg Wayne, Yee Whye Teh, and Nicolas Heess. 2019. Neural Probabilistic Motor Primitives for Humanoid Control. In *International Conference on Learning Representations*. <https://openreview.net/forum?id=Bjl6TjRcY7>
- Josh Merel, Yuval Tassa, Dhruva TB, Sriram Srinivasan, Jay Lemmon, Ziyu Wang, Greg Wayne, and Nicolas Heess. 2017. Learning human behaviors from motion capture by adversarial imitation. arXiv:1707.02201 [cs.RO] <https://arxiv.org/abs/1707.02201>
- Josh Merel, Saran Tunyasuvunakool, Arun Ahuja, Yuval Tassa, Leonard Hasenclever, Vu Pham, Tom Erez, Greg Wayne, and Nicolas Heess. 2020. Catch & Carry: reusable neural controllers for vision-guided whole-body tasks. *ACM Trans. Graph.* 39, 4, Article 39 (Aug. 2020), 14 pages. doi:10.1145/3386569.3392474
- Lars Mescheder, Sebastian Nowozin, and Andreas Geiger. 2018. Which Training Methods for GANs do actually Converge?. In *Proceedings of the 35th International Conference on Machine Learning*.
- Volodymyr Mnih, Adria Puigdomenech Badia, Mehdi Mirza, Alex Graves, Timothy Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. 2016. Asynchronous Methods for Deep Reinforcement Learning. In *Proceedings of The 33rd International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 48)*, Maria Florina Balcan and Kilian Q. Weinberger (Eds.). PMLR, New York, New York, USA, 1928–1937. <https://proceedings.mlr.press/v48/mnih16.html>
- Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. 2015. Human-level control through deep reinforcement learning. *nature* 518, 7540 (2015), 529–533.
- Igor Mordatch, Martin de Las, and Aaron Hertzmann. 2010. Robust physics-based locomotion using low-dimensional planning. In *ACM SIGGRAPH 2010 Papers* (Los Angeles, California) (*SIGGRAPH '10*). Association for Computing Machinery, New York, NY, USA, Article 71, 8 pages. doi:10.1145/1833349.1778808
- Igor Mordatch, Emanuel Todorov, and Zoran Popović. 2012. Discovery of complex behaviors through contact-invariant optimization. *ACM Trans. Graph.* 31, 4, Article 43 (July 2012), 8 pages. doi:10.1145/2185520.2185539
- Sebastian Nowozin, Botond Cseke, and Ryota Tomioka. 2016. f-GAN: training generative neural samplers using variational divergence minimization. In *Proceedings of the 30th International Conference on Neural Information Processing Systems* (Barcelona, Spain) (*NIPS'16*). Curran Associates Inc., Red Hook, NY, USA, 271–279.
- Soohwan Park, Hoseok Ryu, Seyoung Lee, Sunmin Lee, and Jehee Lee. 2019. Learning predict-and-simulate policies from unorganized human motion data. *ACM Trans. Graph.* 38, 6, Article 205 (Nov. 2019), 11 pages. doi:10.1145/3355089.3356501
- Xue Bin Peng, Pieter Abbeel, Sergey Levine, and Michiel van de Panne. 2018a. Deep-Mimic: example-guided deep reinforcement learning of physics-based character skills. *ACM Trans. Graph.* 37, 4, Article 143 (July 2018), 14 pages. doi:10.1145/3197517.3201311
- Xue Bin Peng, Glen Berseth, Kangkang Yin, and Michiel Van De Panne. 2017. DeepLoco: dynamic locomotion skills using hierarchical deep reinforcement learning. *ACM Trans. Graph.* 36, 4, Article 41 (July 2017), 13 pages. doi:10.1145/3072959.3073602
- Xue Bin Peng, Michael Chang, Grace Zhang, Pieter Abbeel, and Sergey Levine. 2019. MCP: learning composable hierarchical control with multiplicative compositional policies. Curran Associates Inc., Red Hook, NY, USA.
- Xue Bin Peng, Angjoo Kanazawa, Jitendra Malik, Pieter Abbeel, and Sergey Levine. 2018b. SFV: reinforcement learning of physical skills from videos. *ACM Trans. Graph.* 37, 6, Article 178 (Dec. 2018), 14 pages. doi:10.1145/3272127.3275014
- Xue Bin Peng, Ze Ma, Pieter Abbeel, Sergey Levine, and Angjoo Kanazawa. 2021. AMP: adversarial motion priors for stylized physics-based character control. *ACM Trans. Graph.* 40, 4, Article 144 (Jul 2021), 20 pages. doi:10.1145/3450626.3459670
- Reallusion. 2023. *3D Animation and 2D Cartoons Made Simple*. <https://www.reallusion.com/>
- com https://actorcore.reallusion.com/motion/pack/studio-mocap-sword-and-shield-stunts, https://actorcore.reallusion.com/motion/pack/studio-mocap-sword-and-shield-moves, https://actorcore.reallusion.com/3d-motion/pack/studio-mocap-hero-motion, https://actorcore.reallusion.com/3d-motion/pack/studio-mocap-girl-dance, https://actorcore.reallusion.com/3d-motion/pack/studio-mocap-evolution-of-dance-vol-1, https://actorcore.reallusion.com/3d-motion/pack/studio-mocap-evolution-of-dance-vol-2, https://actorcore.reallusion.com/3d-motion/pack/iclone-motion-pack—street-dance-locking.
- Diederik M. Roijers, Peter Vamplew, Shimon Whiteson, and Richard Dazeley. 2013. A Survey of Multi-Objective Sequential Decision-Making. *J. Artificial Intelligence Research* 48, 1 (Oct. 2013), 67–113.
- John Schulman, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz. 2015a. Trust Region Policy Optimization. In *Proceedings of the 32nd International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 37)*, Francis Bach and David Blei (Eds.). PMLR, Lille, France, 1889–1897. <https://proceedings.mlr.press/v37/schulman15.html>
- John Schulman, Philipp Moritz, Sergey Levine, Michael Jordan, and Pieter Abbeel. 2015b. High-dimensional continuous control using generalized advantage estimation. *arXiv preprint arXiv:1506.02438* (2015).
- John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal Policy Optimization Algorithms. *CoRR* abs/1707.06347 (2017). arXiv:1707.06347 <http://arxiv.org/abs/1707.06347>
- Adriana Schulz, Harrison Wang, Eitan Grinspun, Justin Solomon, and Wojciech Matusik. 2018. Interactive exploration of design trade-offs. *ACM Trans. Graph.* 37, 4, Article 131 (July 2018), 14 pages. doi:10.1145/3197517.3201385
- Agon Serifi, Ruben Grandia, Espen Knoop, Markus Gross, and Moritz Bächer. 2024. VMP: Versatile Motion Priors for Robustly Tracking Motion on Physical Characters. In *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (Montreal, Quebec, Canada) (*SCA '24*). Eurographics Association, Goslar, DEU, 1–11. doi:10.1111/cgf.15175
- D. Sharon and M. van de Panne. 2005. Synthesis of Controllers for Stylized Planar Bipedal Walking. In *Proceedings of the 2005 IEEE International Conference on Robotics and Automation*. 2387–2392. doi:10.1109/ROBOT.2005.1570470
- Kwang Won Sok, Mammyung Kim, and Jehee Lee. 2007. Simulating biped behaviors from human motion data. In *ACM SIGGRAPH 2007 Papers* (San Diego, California) (*SIGGRAPH '07*). Association for Computing Machinery, New York, NY, USA, 107–es. doi:10.1145/1275808.1276511
- Richard S. Sutton and Andrew G. Barto. 2018. *Reinforcement learning: An introduction* (second ed.). The MIT Press, Cambridge, MA, USA.
- Annan Tang, Takuma Hiraoka, Naoki Hiraoka, Fan Shi, Kento Kawahara, Kunio Kojima, Kei Okada, and Masayuki Inaba. 2024. Humanmimic: Learning natural locomotion and transitions for humanoid robot via wasserstein adversarial imitation. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 13107–13114.
- Chen Tessler, Yunrong Guo, Ofir Nabati, Gal Chechik, and Xue Bin Peng. 2024. Masked-Mimic: Unified Physics-Based Character Control Through Masked Motion Inpainting. *ACM Trans. Graph.* 43, 6, Article 209 (Nov. 2024), 21 pages. doi:10.1145/3687951
- Chen Tessler, Yoni Kasten, Yunrong Guo, Shie Mannor, Gal Chechik, and Xue Bin Peng. 2023. CALM: Conditional Adversarial Latent Models for Directable Virtual Characters. In *ACM SIGGRAPH 2023 Conference Proceedings* (Los Angeles, CA, USA) (*SIGGRAPH '23*). Association for Computing Machinery, New York, NY, USA. doi:10.1145/3588432.3591541
- Faraz Torabi, Garrett Warnell, and Peter Stone. 2018a. Behavioral cloning from observation. In *Proceedings of the 27th International Joint Conference on Artificial Intelligence (Stockholm, Sweden) (IJCAI'18)*. AAAI Press, 4950–4957.
- Faraz Torabi, Garrett Warnell, and Peter Stone. 2018b. Generative adversarial imitation from observation. *arXiv preprint arXiv:1807.06158* (2018).
- Kristof Van Moffaert and Ann Nowé. 2014. Multi-Objective Reinforcement Learning Using Sets of Pareto Dominating Policies. *Journal of Machine Learning Research* 15,

- 1 (2014), 3483–3512.
- Tingwu Wang, Yunrong Guo, Maria Shugrina, and Sanja Fidler. 2020. UniCon: Universal Neural Controller For Physics-based Character Motion. arXiv:2011.15119 [cs.GR]
- Jungdam Won, Deepak Gopinath, and Jessica Hodgins. 2020. A Scalable Approach to Control Diverse Behaviors for Physically Simulated Characters. *ACM Trans. Graph.* 39, 4, Article 33 (2020). https://doi.org/10.1145/3386569.3392381
- Jungdam Won, Deepak Gopinath, and Jessica Hodgins. 2022. Physics-based character controllers using conditional VAEs. *ACM Trans. Graph.* 41, 4, Article 96 (July 2022), 12 pages. doi:10.1145/3528223.3530067
- Jungdam Won, Jongho Park, Kwanyu Kim, and Jehee Lee. 2017. How to train your dragon: example-guided control of flapping flight. *ACM Trans. Graph.* 36, 6, Article 198 (Nov. 2017), 13 pages. doi:10.1145/3130800.3130833
- Philipp Wu, Alejandro Escontrela, Danijar Hafner, Ken Goldberg, and Pieter Abbeel. 2022. DayDreamer: World Models for Physical Robot Learning. arXiv:2206.14176 [cs.RO] https://arxiv.org/abs/2206.14176
- Sitao Xiang and Hao Li. 2020. Revisiting the continuity of rotation representations in neural networks. *arXiv preprint arXiv:2006.06234* (2020).
- Jie Xu, Yunsheng Tian, Pingchuan Ma, Daniela Rus, Shinjiro Sueda, and Wojciech Matusik. 2020. Prediction-Guided Multi-Objective Reinforcement Learning for Continuous Robot Control. In *Proceedings of the 37th International Conference on Machine Learning (ICML)*.
- Pei Xu and Ioannis Karamouzas. 2021. A GAN-Like Approach for Physics-Based Imitation Learning and Interactive Character Control. *Proc. ACM Comput. Graph. Interact. Tech.* 4, 3, Article 44 (Sept. 2021), 22 pages. doi:10.1145/3480148
- Pei Xu, Xiumin Shang, Victor Zordan, and Ioannis Karamouzas. 2023. Composite Motion Learning with Task Control. *ACM Trans. Graph.* 42, 4, Article 93 (July 2023), 16 pages. doi:10.1145/3592447
- Runzhe Yang, Xingyuan Sun, and Karthik Narasimhan. 2019. *A generalized algorithm for multi-objective reinforcement learning and policy adaptation*. Curran Associates Inc., Red Hook, NY, USA.
- Heyuan Yao, Zhenhua Song, Baoquan Chen, and Libin Liu. 2022. ControlVAE: Model-Based Learning of Generative Controllers for Physics-Based Characters. *ACM Trans. Graph.* 41, 6, Article 183 (Nov. 2022), 16 pages. doi:10.1145/3550454.3555434
- Wenhao Yu, Greg Turk, and C. Karen Liu. 2018. Learning symmetric and low-energy locomotion. *ACM Trans. Graph.* 37, 4, Article 144 (July 2018), 12 pages. doi:10.1145/3197517.3201397
- Yi Zhou, Connelly Barnes, Jingwan Lu, Jimei Yang, and Hao Li. 2019. On the continuity of rotation representations in neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 5745–5753.
- Qingxu Zhu, He Zhang, Mengting Lan, and Lei Han. 2023. Neural Categorical Priors for Physics-Based Character Control. *ACM Trans. Graph.* 42, 6, Article 178 (Dec. 2023), 16 pages. doi:10.1145/3618397