DOI: 10.1111/cgf.15172
ACM SIGGRAPH / Eurographics Symposium on Computer Animation 2024
M. Skouras and H. Wang
(Guest Editors)

COMPUTER GRAPHICS *forum*
Volume 43 (2024), Number 8

ADAPT: AI-Driven Artefact Purging Technique for IMU Based Motion Capture

P. Schreiner [1,2], R. Netterstrøm[1], H. Yin[2], S. Darkner[2] and K. Erleben[2]

[1]Rokoko Electronics, Denmark
[2]University of Copenhagen, Department of Computer Science, Denmark

Abstract

While IMU based motion capture offers a cost-effective alternative to premium camera-based systems, it often falls short in matching the latter's realism. Common distortions, such as self-penetrating body parts, foot skating, and floating, limit the usability of these systems, particularly for high-end users. To address this, we employed reinforcement learning to train an AI agent that mimics erroneous sample motion. Since our agent operates within a simulated environment, it inherently avoids generating these distortions since it must adhere to the laws of physics. Impressively, the agent manages to mimic the sample motions while preserving their distinctive characteristics. We assessed our method's efficacy across various types of input data, showcasing an ideal blend of artefact-laden IMU-based data with high-grade optical motion capture data. Furthermore, we compared the configuration of observation and action spaces with other implementations, pinpointing the most suitable configuration for our purposes. All our models underwent rigorous evaluation using a spectrum of quantitative metrics complemented by a qualitative review. These evaluations were performed using a benchmark dataset of IMU-based motion data from actors not included in the training data.

CCS Concepts

• ***Computing methodologies*** → ***Motion capture;*** *Physical simulation; Motion processing; Reinforcement learning;*

1. Introduction

IMU-based motion capture is affordable and untethered, however, it lacks the quality of higher end marker based solutions. In this work, we present a physics-based, AI-enhanced method for improving motion data quality. The last decade has seen a rise in the popularity of IMU-based motion capture systems, making this technology common ground. It has proven itself as a reasonable alternative to marker based optical motion capture systems, in scenarios dictated by limited budgets or a need to capture 'in the wild'. Nevertheless, these systems lack the quality of high end marker based solutions. Physically implausible artefacts are common ground in the raw capture data. Examples of artefacts are self-collisions, foot skating, and, floating. For visual examples, see Figure 1.

In this work, we present a method for cleaning motion data while maintaining natural human-like motion qualities. We use a combination of physics simulation and learned AI behavior, through reinforcement learning. Our agent learns to mimic faulty sample motions and generalise this learning to motion data that was not seen during training. The core idea of our approach is that since our agent 'lives' in a physically accurate simulated environment it is not capable of reproducing errors that are in contest with the laws of physics, resulting in visually pleasing and physically plausible output motion. Our method builds on top of recent developments in the state of the art in this field. However, it distinguishes itself by its capability to operate on unseen motion data, stemming from unseen characters. One of our major findings is the importance of the choice of data source when training the agent. We show that the key to a robust and versatile agent is training on a mix of high quality and faulty motion capture data from different sources. Agents trained solely on either high quality or faulty data lack the robustness to generalise to unseen data. Finally, we compare fundamental differences in the configuration of the observation and actuation spaces of our agent for our use case. We demonstrate an optimal configuration and its impact on the agent's capabilities to produce high quality output motion. In summary, the contributions presented in this work are:

- A physics-based framework for cleaning artefacts from motion capture data.
- A method that focuses on generalising to unseen IMU data, from unseen actors. This is in contrast to most other motion-mimicking methods, that draw their sample motions from known distributions.
- We show that mixing faulty and high quality training data is pivotal in generalising our method to unseen animation data.
- We show that the choice of observation and actuation configuration greatly impacts the agent's ability to learn and produce quality output.

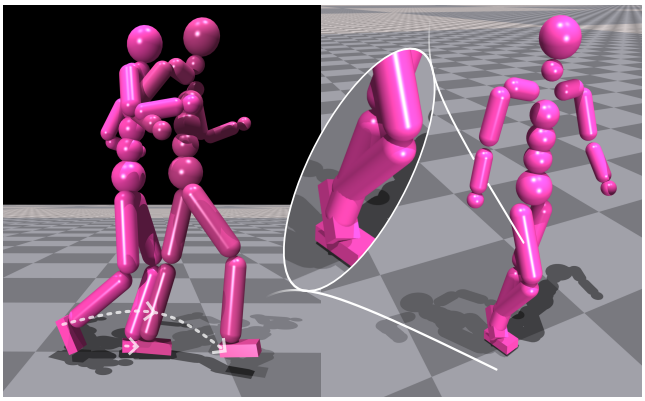

Figure 1: *Examples of typical artefacts as seen in IMU-based motion capture recordings. On the left is an example of two frames from an animation clip where we see foot skating: both the foot in motion (rear foot), and the foot from the supporting leg, are displaced. On the right is an example of self-collision, where one leg is penetrating the other near the ankle.*

2. Related Work

Imperfections are a common trait in real-world data, motion capture data being no exception. Inherent hardware limitations can cause missing or faulty sensor information. In the case of optical motion capture, common sources of error are missing information due to occluded markers or markers going out of view of the cameras. For IMU based motion capture, errors are mostly due to signal noise and bias, limitations in sensor precision, and resolution. These imperfections can cause a range of faults in the resulting animations, such as jittering limbs, self-collisions, foot-skating, and others.

Recognizing the challenges posed by these imperfections in motion capture data, both researchers and industry professionals have continuously sought efficient and effective solutions to assist animators in the data cleaning process. While a significant portion of this cleaning is still manually performed, aided by a range of commercial software suites the quest for full automation remains an unsolved challenge. The following sections delve into some of the innovative approaches in this domain.

2.1. Non-Physics Based Methods

Existing techniques often focus on eliminating these artefacts using straightforward kinematic and/or geometric techniques [PP10]. A shared trait by many of these methods is that they don't take the physics of the character into account, leading to unnatural poses and dynamic behaviour and they rely on post-clean-up techniques using IK solvers [GM85, BRRP97, CK99, LS99, KSG02, GBT06, LL14]. Recent deep learning approaches focus on robust detection of foot planting or global root position and apply IK as a post-clean-up process [SPL*21, MHCH22]. It is known that the IK clean-up is somewhat inferior in quality and that humans are extremely sensitive to even small foot sliding errors [PHO11].

Deep learning approaches have also been adopted to train generative models to directly produce believable dynamic behaviours, possibly taking a pre-clean-up sample in an encoding step for reconstruction. Care is often taken to the model design for effectively capturing the motion data statistics. Examples include MoGlow using latent flow models [HAB20] and its variant encoding skeleton data with graph neural networks [YYKB21]. Generative models demonstrate high performance in synthesising complex human movements of various styles and rich contextual input [VPHB*21, DAS*20, YYB*23]. Physical constraints, however, tend to not be explicitly enforced but with an expectation of extracting them solely from the motion data. As a result, state-of-the-art such as diffusion models are often exploited to represent the behaviours to imitate [TRG*23, YTY*23], in combination with physically-grounded approaches reviewed below.

2.2. Deep Reinforcement Learning (DRL)

Ever since the 1980's physics simulation has been a topic within (interactive) character animation, steadily gaining traction over the past decades. A detailed overview of the early stages of this research was made by [GP12]. More recently DRL for character animation has become an active field of research, as shown by [MHLC*22]. The ability to incorporate physics makes DRL an attractive approach for motion data cleaning. DRL with physics simulations give these methods an inherent sense of realism: they allow for interactivity, and they can be employed online and in real-time. In this class of methods, an agent is trained to reproduce sample motions in a physically simulated environment. These methods can be subdivided into two sub-classes: methods that employ a phase variable to give the agent a notion of timing of the sample motion and those that directly supply the agent with a sample of motion data. Where the former can be used to learn single motions and general behaviour, the latter is more suitable to mimic diverse sets of sample motions closely.

2.2.1. Phase Variable Methods

A number of works successfully mimicked reference motion by relying on the use of a phase variable, providing the policy with information about the timing of the reference motion. With their groundbreaking work, DeepMimic, [PALVdP18] showed that an agent could learn to mimic complex behaviours while a user could set high-level task objectives, such as hitting a target when performing a spin kick. In follow-up work, [PMA*21] used GANs to have an agent learn similar behaviour from unstructured data sets. [MYT*21] improved sample efficiency using a set of constraints called spacetime bounds, effectively limiting the action search space. Inspired by NLP models, [PGH*22] map a set of motions into latent motion embeddings. The authors then train a low-level policy to generate motions from these embeddings, using adversarial imitation learning. Subsequently, they train a high-level policy to complete new tasks by passing latent embeddings to the low-level policy.

In [XMN*22] a differentiable physics simulator is used in combination with a new policy learning algorithm, short horizon actor-critic (SHAC), to improve training time. They showcase a speed up in training humanoid agents compared to the state of the art, reducing training time by a factor of up to 17x. [RYC*23] proposes a method to directly learn a policy by back-propagating through a

differentiable physics simulation, thereby speeding up the learning process and eliminating the need for reinforcement learning altogether.

While phase-variable methods are successful in training an agent to imitate single motions, general motion behaviours and styles, they are less suitable for use in our case where the requirement is to mimic a large variety of specific sample motions. This is due to the lack of information available to the agent during inference about the sample motions.

2.2.2. Increasing the Variety of Learned Skills

A more suitable approach for close reference tracking of diverse motion capture data is to replace the phase variable with a sample of the reference motion in the state information of the agent. The first to implement this idea were [CMM*18], whose agent was trained on a large body of unstructured motion capture data. It was able to mimic a large variety of human motion tasks, even some from unseen data. While impressive in their versatility, the output motion quality does not compare to the phase variable methods earlier discussed. Different works have since tried to improve diversity in the agent's skill set and quality of the output motion. Multiple works focused on making their agent robust for in game use and interactivity. [PRL*19] and [BCHF19] concurrently came with a two layer pipeline. They first generated kinematic motion samples from a user's control input and subsequently mimicked the motion through a RL agent. The latter step improves the generated motion by making it physically plausible, while the first allows for flexible high-level user control, perfect for in-game use. On top of that they enable interaction between the character and their environment. Similarly, [LSCC20] proposes a framework for user control of quadrupeds, using GANs to map high-level user control inputs, such as direction and velocity, to influence factors for primitive motor actions. The controller is then fine-tuned using deep RL to improve robustness. In [MTA*20] the authors encoded latent task intentions from motion trajectories to learn interactive tasks such as catching a ball or carrying an object to a specified target. These works focused on real-time performance, for example for in-game use. Unlike our approach, they rely on a fixed body of underlying data during inference, which stems from the same data source and has been seen during training. This makes them less suitable for tracking arbitrary reference motions, which is a hard requirement for our case.

Other works focused on anatomical features of agents. [WL19] and [LLLL21] used parameterised controllers to accommodate for body shape variations on the fly or changing environmental and motion characteristics. [LPLL19] do motion mimicking with a musculoskeletal model to study the effects of anatomical symptoms and prosthetics on locomotion. While they were effective for their purpose, the complex anatomical model makes the simulations and subsequent policy training computationally heavy and slow. Each of these contributions serves a clear purpose for each of their individual use cases, however they do not intend to solve the problem of generalising to unseen motion data while maintaining output motion quality and they do not address the topic of diverse data modality for training, which we found to be essential for generalisation to unseen IMU data.

More interesting for our objective are those works that explicitly focus on improving skill diversity. To this end, [WGH20] designed a mixture of experts policy but they relied on high quality motion data. One of the few works that do report diverse data sources is [WGSF20]. They introduced constrained multi-objective reward optimisation to avoid domination of individual reward terms, a motion balancer to ensure a uniform distribution of motion classes during training, and adaptive policy variance control to avoid local minima. They demonstrate spectacular results in terms of versatility, robustness, and generalisability, even to unseen motion samples. However, their robustness seems to come at too high of a cost of motion quality for our purpose, specifically for unseen motion data.

Recent studies in the field of virtual reality [WWY22, YLHX22] show impressive motion reconstruction from only 3 sensors and using a reinforcement learning trained policy. Their methods guarantee physically correct motion, potentially bypassing the need for cleanup. However, they rely on combinations of SLAM camera sensors and IMUs, providing relatively high quality global positional information, thus constraining the applicability of the method.

Possibly most relevant to our work, [YSI*22] used motion mimicking to clean up physically implausible artefacts from motion data generated by a diffusion model. The core difference here is that the diffusion model and mimicking policy are both trained and evaluated on the same dataset. The policy subsequently never sees real unseen data during training, but rather samples drawn from the known distribution. In our work we focus on making our method robust towards unseen data, by combining different data modalities and a large observation space.

3. Method

Unique to our approach is the way we focus on generalising to unseen data, from unseen actors. We achieve this by using a dataset that consists of a large number and variety of motions that are recorded with either high-quality optical motion capture setups or lower quality IMU-based motion capture setups. Some IMU-based assets purposefully contain physical implausibilities to make the agent acquainted with that input. This is in contrast to other methods that usually rely on high-quality motion capture data, often from a single source.

Our method follows three general stages. During **data preparation** we first record a dataset of animations using IMU-based motion capture suits. These recordings are post-processed with industry-standard clean-up filters. We then select recordings such that the resulting data set contains a mixture of faulty and clean recordings. Finally, we mix in a set of high-quality recordings from an optical motion capture source. We have compared this approach to using only IMU-based data and found this last step crucial in getting optimal performance on unseen motion data. Section 3.1 describes the data in more detail.

Next, during the **training stage**, our policy is optimised to mimic the presented sample motions. In a large parallel physically simulated environment, we collect a number of rollouts from our policy. For each rollout, we select a random sample asset and initialise

it at a random frame in the animation. Rollouts have a maximum length of 300 frames. After collecting a fixed number of rollouts, we optimise our policy using the RL-games implementation of PPO from [MM21], as PPO is the de facto standard for policy gradient optimisation for this type of problem. We repeat this until the policy converges in terms of reward collection and episode length. A general overview of the motion-mimicking formulation is given in Section 3.2. Choices for our agent's state observation and action spaces are described in Sections 3.2.2 and 3.2.1 respectively. Section 3.2.3 discusses the way we calculate the agent's rewards. Details about our policy and value network configuration are given in Section 3.2.4 and the remaining details on our training and simulation setup are discussed in Section 3.2.5.

In the last stage, our policy is **evaluated on unseen data** using a set of performance metrics that we found descriptive for our context. These metrics are the episode length ratio compared to the ground truth, the mean squared error between the sample and ground truth root trajectories, the survival rate of the agent and the ratio between achieved and maximum achievable reward. To this end, we collect 20 rollouts for each asset in our test data set and evaluate our model in terms of the performance metrics. We repeat this every 400 epochs to find the best policy from a training run. The performance metrics are described in detail in Section 3.2.6.

3.1. A Mixed IMU and Optical Mocap Data Set

For this project, we used both high-quality motion capture data from optical sources and IMU-based motion capture recorded using Rokoko's Smartsuit Pro II. This section describes the details concerning data collection, hardware used, and total numbers. For details about specific training datasets, we refer to Section 4.3.

Optical Data The optical data is sourced from a large, commercially available motion asset library. These assets have been recorded at various motion capture studios, using optical systems such as Optitrack and Vicon. The exact hardware is not known by the authors. All assets are recorded at 120 frames per second (fps) and are down-sampled to 60 fps. We used a total of 107 assets, containing a total of 80102 frames, in this project, which were exclusively used during training and not for testing.

IMU Data The IMU-based motion data was recorded on different occasions using Rokoko's Glove Ready Smartsuit Pro II. These suits contain 17 IMU sensors distributed over the body. The sensors record bone orientations, which are fitted to a body model through Rokoko's studio software to produce animations of a humanoid character. Since these sensors exclusively record orientations, the animations initially lack a sense of global placement. Through post-processing filters provided by the studio software, the animations are cleaned and augmented with global position estimates. The resulting animations still contain artefacts, such as self-collisions, foot-skating, and jittering limbs.

In this work, we use a total of 118 such recordings, recorded using 13 actors. The actors had various body shapes, dimensions, and genders. Of these recordings, 13 assets were exclusively used as test data. To ensure that the test result is representative of both cleaning abilities as well as the agent's ability to mimic motion, the test assets were carefully selected to contain both fairly clean and vivid animations as well as animations with typical flaws. The actors used to record the test assets were kept out of the training data.

The IMU-based animations were initially recorded at 200 fps and subsequently exported to FBX files, sampled at 60 fps during training and inference.

3.2. Motion Mimicking for Artefact Clean Up

We aim to fix physical implausibilities in the mocap data, such as self-collisions, foot skating, floating, ground penetrations, and jitter. To this end, we train an AI-based agent that is capable of mimicking a given sample motion. The agent constitutes a physically plausible model of a humanoid and 'lives' in an environment, which is a physically simulated approximation of the real world. Hence the agent must adhere to common laws of physics while mimicking sample motions, prohibiting it from copying the above artefacts from sample motions.

The agent's joints are actuated by joint torques computed from PD controllers within the simulation environment. A policy $\pi(\mathbf{a}|\mathbf{s})$, constitutes the probability of an action $\mathbf{a}$ given the agent's current state $\mathbf{s}$. The actions in this setting are joint angle targets for the PD controllers. After applying actions $\mathbf{a}$, the environment computes the agent's new state and a reward, r, based on how well the agent performed. See Figure 2 for a schematic overview of this process.

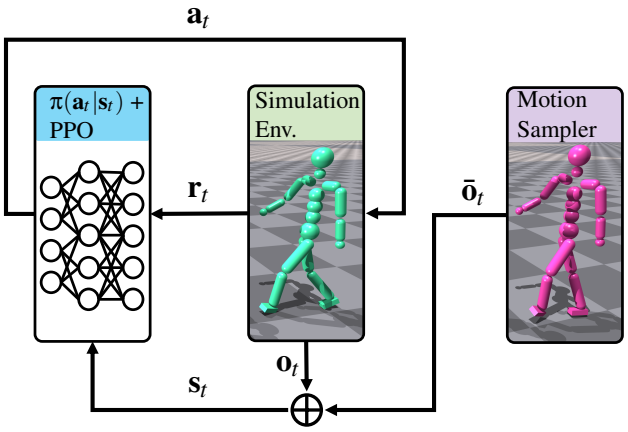

Figure 2: *Schematic of the policy evaluation loop used to create rollouts from policy $\pi(\mathbf{a}_t|\mathbf{s}_t)$. The simulation and motion sampler both generate parts of the state vector $\mathbf{s}_t$. The policy computes the most likely action given the state, which is applied to the character by the simulation environment. This results in a new state and reward. After a batch of rollouts is collected, the policy is updated using the collected rewards. We use PPO to optimise the policy.*

3.2.1. State Observation

We use a skeleton containing 20 bodies, connected by 19 joints. The elbows and knees are modeled as 1-DoF hinge joints. All other joints are modeled as 3 hinge joints connected in series to achieve a 3-DoF rotational joint. The total number of DoF is $13 \times 3\text{DoF} +$

$4 \times 1\text{DoF} = 43\text{DoF}$. Hand motion is not actuated, and therefore their joints are not counted for the total number of DoF. The pelvic bone is assigned to be the character's root.

The agent's policy gets information about its own state in the form of a state vector. Based on this it computes the actions for the next step. As discussed in Section 2, one key feature is that we want the policy to use the information on the desired state, as well as its own state, in order to enhance its ability to mimic arbitrary reference motions. Therefore we include information about the simulated character's motion and the reference motion to the state vector.

In Table 1 we present our state vector $\mathbf{s}_t \in \mathbb{R}^{273}$ at time step t and its components.

Table 1: *The agent's state observation vector. The state observation vector is composed of a number of key observations on the agent's state, stacked as a 1D vector of 273 elements.*

Symbol	Dim.	Description
$\mathbf{v}_{com}$	$\mathbb{R}^3$	Centre of Mass (CoM) velocity of the agent
$\bar{\mathbf{v}}_{com}$	$\mathbb{R}^3$	CoM velocity of the reference motion
$\Delta\mathbf{v}_{com}$	$\mathbb{R}^3$	$\mathbf{v}_{com} - \bar{\mathbf{v}}_{com}$
$\bar{\mathbf{v}}_{hor}$	$\mathbb{R}^2$	The reference motion's velocity in the horizontal plane
$\Delta\mathbf{v}_{hor}$	$\mathbb{R}^2$	$\mathbf{v}_{hor} - \bar{\mathbf{v}}_{hor}$
$\mathbf{p}_a$	$\mathbb{R}^{60}$	Positions of all 20 rigid bodies of the agent with respect to its CoM
$\mathbf{v}_a$	$\mathbb{R}^{60}$	Global velocities of all 20 rigid bodies of the agent
$\Delta\mathbf{p}_a$	$\mathbb{R}^{60}$	$\mathbf{p}_a - \bar{\mathbf{p}}_a$
$\Delta\mathbf{v}_a$	$\mathbb{R}^{60}$	$\mathbf{v}_a - \bar{\mathbf{v}}_a$
$\mathbf{a}_{t-1}$	$\mathbb{R}^{20}$	Smoothed actions from the previous time step (see section 3.2.2)
$\mathbf{s}_t$	$\mathbb{R}^{273}$	$\{\mathbf{v}_{com}, \bar{\mathbf{v}}_{com}, \Delta\mathbf{v}_{com}, \bar{\mathbf{v}}_{hor}, \Delta\mathbf{v}_{hor}, \mathbf{p}_a, \mathbf{v}_a, \Delta\mathbf{p}_a, \Delta\mathbf{v}_a, \mathbf{a}_{t-1}\}$

All velocity quantities are expressed in the global reference frame. The rigid body origin positions, $\mathbf{p}_a$, are expressed in the frame whose origin is attached at the agent's centre of mass (CoM) and whose axes are parallel to the global frame. To compute the difference between the agent's and the reference motion's rigid body positions, we compute the reference motion's rigid body positions $\bar{\mathbf{p}}_a$ in a similar frame attached to the reference motion's CoM. We then simply calculate the difference between the two vectors: $\Delta\mathbf{p}_a = \mathbf{p}_a - \bar{\mathbf{p}}_a$.

The last entry of the state vector is the smoothed actions from the previous time step. This is done in accordance with [BCHF19] to give the policy information about the smoothing process.

3.2.2. Action Space

In this section, we give a brief overview of our action space. For our implementation, we took inspiration from the work of [BCHF19]. We follow their implementation of the action space closely but with a few tweaks.

Our agent's joints are actuated by torques computed by PD controllers. All computations are on a per-frame basis, but in our notation, we omit the frame number for clarity.

$$\tau_d = k_p e_d + k_d \dot{e}_d \,, \tag{1}$$

$$e_d \equiv \theta_d - \tilde{\theta}_d \,. \tag{2}$$

Here θ_d is the current angle of degree of freedom d and $\tilde{\theta}_d$ is the target angle for that degree of freedom. The target angles are computed using the reference motion and a correction term from our policy:

$$\tilde{\theta}_d \equiv \bar{\theta}_d + \alpha_d a_d \,. \tag{3}$$

Here $\bar{\theta}_d$ is the angle of degree of freedom d from the reference motion at the current frame, which serves as a feed-forward open-loop action. Our policy computes the closed-loop action a_d based on the current state of the agent. Finally, α_d is a fixed binary operator that can be either 0 or 1, thus excluding certain degrees of freedom from closed-loop actuation. We set $\alpha_d = 1$ for the degrees of freedom of a set of key joints:

$$\{\textit{Right shoulder, Right hip, Right knee (1D), Right ankle, Left shoulder, Left hip, Left knee (1D), Left Ankle}\} \,. \tag{4}$$

In contrast to [BCHF19], we found that actuating any of the spinal elements in a closed-loop occasionally causes unnatural spinal wobbling and therefore we excluded all spinal elements from the closed-loop actuation.

In order to avoid high-frequency oscillations in the control signal, we further follow the example from [BCHF19] and perform a smoothing operation on our closed-loop action signals:

$$\mathbf{a}_t \leftarrow \beta\, \mathbf{a}_{raw,t} + (1-\beta)\, \mathbf{a}_{t-1} \,. \tag{5}$$

Here $\mathbf{a}_t$ are the smoothed actions at time t, $\mathbf{a}_{raw,t}$ are the actions generated by the policy, and β is a smoothing factor. For details and a justification for the value of this parameter, we refer to the original work. We use $\beta = 0.2$ as reported by the authors for all experiments in this work.

3.2.3. Rewards

Our agent is rewarded based on the similarity between the simulated character state and the sample motion state. We use a compound reward based on the similarity between the simulated and the sample's joint angles, positions of bone landmarks, and bone landmark velocities. The reward scaling and threshold parameters discussed further in this section are listed in Table 2.

Table 2: *Reward scaling parameters and threshold parameters. The weights were empirically chosen to ensure different reward terms contribute equally to the total reward.*

Parameter	α_{local}	α_p	α_v	$\epsilon_{discount}$
Value	2.5	1.0	0.1	0.025

Local Pose Reward We calculate a pose reward based on the sum of the local joint angle errors for the J joints of the skeleton, using:

$$r_{local} \equiv \exp\left(-\alpha_{local}\frac{1}{J}\sum_{j=0}^{J}\angle(\mathbf{q}_j^{-1}\bar{\mathbf{q}}_j)\right). \quad (6)$$

Here $\mathbf{q}_j$ represents the rotation of the j-th joint in a local frame attached to the joint's origin and whose axes are fixed to the joint's parent body. The $\bar{}$ indicates ground truth. The parameter α_{local} is an empirically chosen weighing factor for the local joint angle reward.

Position Reward We use the same position and velocity rewards as [BCHF19]:

$$r_p \equiv \exp\left(-\alpha_p\frac{1}{J}\sum_{j=1}^{J}\sum_{k=1}^{6}\parallel \mathbf{p}_{jk}-\bar{\mathbf{p}}_{jk}\parallel_2\right). \quad (7)$$

Here $\mathbf{p}_{jk}$ and $\bar{\mathbf{p}}_{jk}$ are the positions, with respect to the root bone of the simulation and sample skeletons respectively, of face centre k of a unit cube mounted to the origin of bone j. The weighing factor α_p is empirically chosen.

Velocity Reward The velocity reward is computed analogously to the position reward as:

$$r_v \equiv \exp\left(-\alpha_v\frac{1}{J}\sum_{j=1}^{J}\sum_{k=1}^{6}\parallel \mathbf{v}_{jk}-\bar{\mathbf{v}}_{jk}\parallel_2\right). \quad (8)$$

Finally, the entire reward is discounted based on the fall factor from [BCHF19]. This serves as a correction on the reward, in case the simulated character falls behind on the sample motion.

$$e_{\text{discount}} \equiv \text{clamp}(1.3-1.4\parallel \mathbf{p}_{head}-\bar{\mathbf{p}}_{head}\parallel_2, 0, 1). \quad (9)$$

The resulting reward function is:

$$r \equiv e_{\text{discount}}(r_{local}+r_p+r_v). \quad (10)$$

Finally, we terminate the episode in case the agent either falls behind beyond a given threshold, $e_{discount} < \epsilon_{discount}$ or when the agent has fallen. Falling is detected as 3 or more spinal elements being in contact with the ground plane at the same time. The termination threshold is set to $\epsilon_{discount} = 0.025$.

3.2.4. Policy and Value Networks

Our policy and value functions are approximated using neural networks, both with an identical architecture. We follow the approach of [YSI*22] and use multi-layer perceptrons (MLPs) with 3 fully connected layers of size [1024, 1024, 512]. At each layer output, we use *tanh* activation functions as suggested by [BCHF19]. The policy network estimates the mean of a normal distribution. During training, the agent's actions are sampled from this distribution, using a fixed variance. This variance determines to a high degree the amount of exploration an agent performs. We found empirically that a value of $\sigma^2 = 0.03$ gave a good balance between exploration and stable training. Table 3 gives an overview of these and other hyperparameters and simulation parameters.

3.2.5. Training & Simulation

We trained all our agents using NVIDIA's Isaac Gym. This choice was driven by Isaac Gym's ability to do massive, GPU accelerated parallel simulations, reducing training times to a minimum. As a reinforcement learning algorithm, we use PPO ([SWD*17]), as it is the de facto standard for these types of tasks as can be seen in [BCHF19, PALVdP18, PMA*21]. We use the PPO implementation from RL-games by [MM21], which comes shipped with Isaac Gym. The training was performed on an NVIDIA RTX A5500 with 24GB of memory. For more information about our training and simulation configuration, we refer to Table 3.

Table 3: *Training and simulation parameters for our motion mimicking agent's policy optimisation.*

Paramter	Value
Simulation time step	0.0166 s (60 Hz)
# of substeps	2
Control frequency	30 Hz
Episode length	300
Network layers	[1024, 1024, 512]
Activation	tanh
Learning rate	$5 \cdot 10^{-5}$
Distribution variance σ^2	0.03
PPO clipping ϵ_{clip}	0.2
Batch size	65536
Minibatch size	16384
Mini epochs	6
Discount factor γ	0.99
GAE τ	0.95
Episode horizon	16
Number of environments	4096

3.2.6. Performance Metrics

To determine performance on unseen motion data, we use four metrics discussed in this section. The metrics are effective on the condition that the ground truth assets all have the same length, as only then does it make sense to calculate statistics on episode length ratio, root displacement errors, survival rate, and maximum achievable rewards.

We discuss the metrics here as they are computed per rollout. For the evaluation of policies, we then calculate mean and standard deviations for each metric, after evaluating them on a test dataset of unseen motion assets.

Episode Length Ratio The episode length ratio is defined as the length of rollout i, divided by the length of the ground truth motion asset (gt):

$$ELR_i \equiv \frac{\text{length}(rollout_i)}{\text{length}(gt_i)}. \quad (11)$$

This metric provides information about the ability of a policy to mimic sample motions well enough to not terminate early. If the metric is one, then the rollout was not terminated early; if the metric is zero, it does not mimic a single pose.

Table 4: *The results of our method, in bold, compared for models trained on different training data and configurations of the observation and action spaces. The models are compared based on the ratio of the episode lengths and the original asset's lengths, the fraction of episodes that reached at least 90% of the original asset length, the MSE of the root trajectories per frame in m, and the ratio of the obtained and maximum obtainable reward.*

Experiment	Episode length ratio μ_{el}	Episode length ratio σ_{el}	SR_{90}	Root MSE / frame μ_{mse}	Root MSE / frame σ_{mse}	Max reward ratio μ_r	Max reward ratio σ_r
IMU + Optical (50 assets, 39270 frames)	**0.969**	**0.081**	0.889	$\mathbf{5.118 \cdot 10^{-05}}$	$\mathbf{4.789 \cdot 10^{-05}}$	**0.400**	**0.114**
IMU large data (107 assets, 80102 frames)	0.961	0.120	**0.907**	$6.386 \cdot 10^{-05}$	$5.639 \cdot 10^{-05}$	0.371	0.111
IMU (33 assets, 28925 frames)	0.889	0.191	0.757	$6.949 \cdot 10^{-05}$	$5.882 \cdot 10^{-05}$	0.386	0.121
Optical, large data (103 assets, 68392 frames)	0.782	0.248	0.469	$1.424 \cdot 10^{-04}$	$1.107 \cdot 10^{-04}$	0.257	0.070
Optical (17 assets, 10345 frames)	0.468	0.212	0.017	$2.132 \cdot 10^{-04}$	$1.325 \cdot 10^{-04}$	0.329	0.095
Configuration Drecon [BCHF19]	0.942	0.141	0.885	$5.863 \cdot 10^{-05}$	$5.598 \cdot 10^{-05}$	0.372	0.110
Conf. full state & action space	0.924	0.168	0.839	$6.251 \cdot 10^{-05}$	$7.980 \cdot 10^{-05}$	0.273	0.099

Root Displacement Error This metric measures the mean squared error (MSE) between the root trajectories of rollout i and its ground truth motion asset:

$$RDE_i \equiv \frac{1}{N} \sum_{n=0}^{N-1} \| \mathbf{p}_{i,n} - \bar{\mathbf{p}}_{i,n} \|_2^2 \,. \tag{12}$$

We use N for the number of frames of the rollout. The RDE provides information about the accuracy with which the policy tracks the ground truth motion trajectory.

Survival Rate .9 The survival rate measures the probability of a policy successfully mimicking a given asset from the start until a given percentage of the ground truth length. The ".9" means the probability of an agent surviving 90% of the given ground truth asset. If the metric is one, the agent survives 90% or more of the given asset.

$$SR_i \equiv \begin{cases} 1 & \text{if } len(rollout_i) \geq len(gt) \cdot 0.9 \\ 0 & \text{else} \end{cases} \,. \tag{13}$$

Max Reward Ratio The max reward ratio measures the fraction of the maximum cumulative reward an agent achieves over rollout i:

$$MRR_i \equiv \frac{R_i}{R_{max,i}} \,. \tag{14}$$

The cumulative reward is calculated as:

$$R_i = \sum_{n=0}^{N-1} r_n \,. \tag{15}$$

For a rollout with length N and using Equation 10. For the maximum cumulative reward, $R_{max,i}$ we use $N = length(gt_i)$ and $e_{discount} = r_{local} = r_p = r_v = 1.0$. Note that due to using exponential rewards and the choice of weights, achieving the maximum reward is rather unlikely and in practice the score for this metric falls well below 1.0.

4. Results

In this work, we hypothesise that a humanoid agent, trained using reinforcement learning, is capable of fixing common physically implausible motion artefacts in unseen IMU-based motion capture data. We seek to find answers to questions, such as what data mixture is optimal for training, and which configurations of the state and action spaces result in the most robust agent. In this section, we discuss the experiments and their results that provide answers to these questions.

4.1. Unseen Data Study

To determine how well our method generalises to unseen data, we tested our agent's performance on a set of motion data that was not present during training. The test data consists of 13 IMU-based recordings from four different actors, with different body types and genders. The recordings were split into 27 segments of 300 frames long. This was done to mitigate the effects of failure due to specifically difficult movements in long recordings, which would otherwise render large parts of the test data inaccessible. This was justified by the fact that we wanted to test the agent for overall capabilities and not so much for surviving for the longest possible amount of time.

The first row of Table 4 shows the performance in terms of our metrics. For each of the 27 segments, 20 rollouts were collected all starting at slightly different timings. The metrics were calculated on a total of 540 rollouts. The standard deviations in the table give a measure of how much the mean values differed between rollouts.

Figure 3 is a visual comparison of a ground truth asset with heavy self-collisions, and the same asset reproduced by our agent. The agent manages to faithfully keep key features and details of the motion intact, while not exhibiting any self-collisions. Our method also improves the CoM position of the character: in the last frame of the sequence, the ground truth character can be seen to lean unnaturally to a side, causing the projection of the CoM on the floor to lie outside of the base of support of the feet. This is unusual for human motion and would cause the character to risk losing balance in

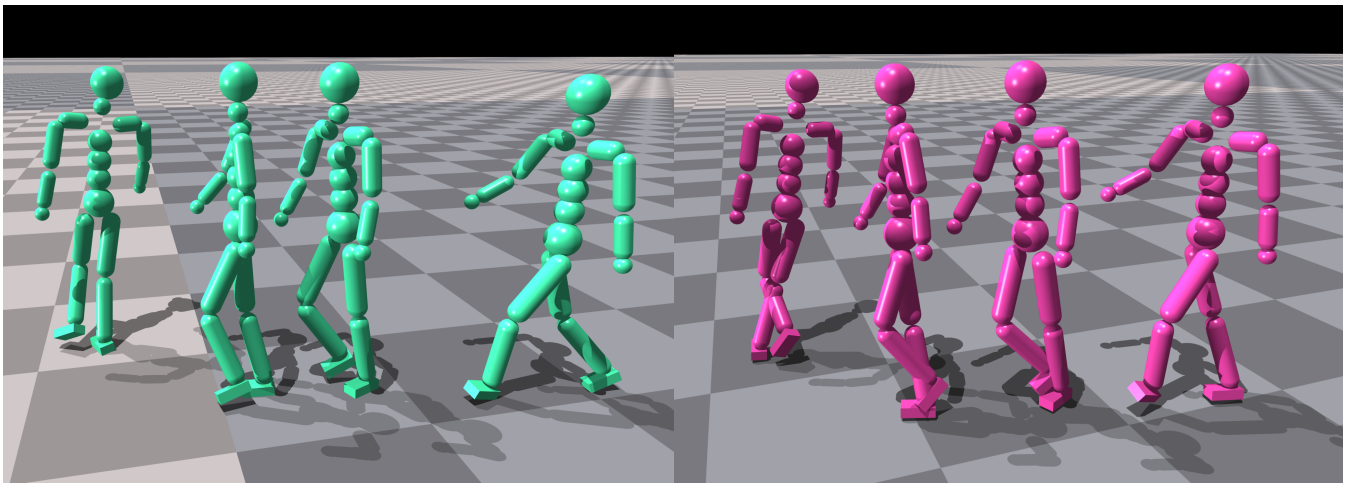

Figure 3: *Collision fixing on a flawed animation. The image shows still frames from an animation before and after processing by our method. The ground truth (pink) displays self-collisions at the legs in 3 of the still frames while the agent (green) does not. The agent also maintains a more natural pose throughout the animation, while the ground truth leans in unnaturally in the last still frame, placing its centre of mass far outside of its foot base.*

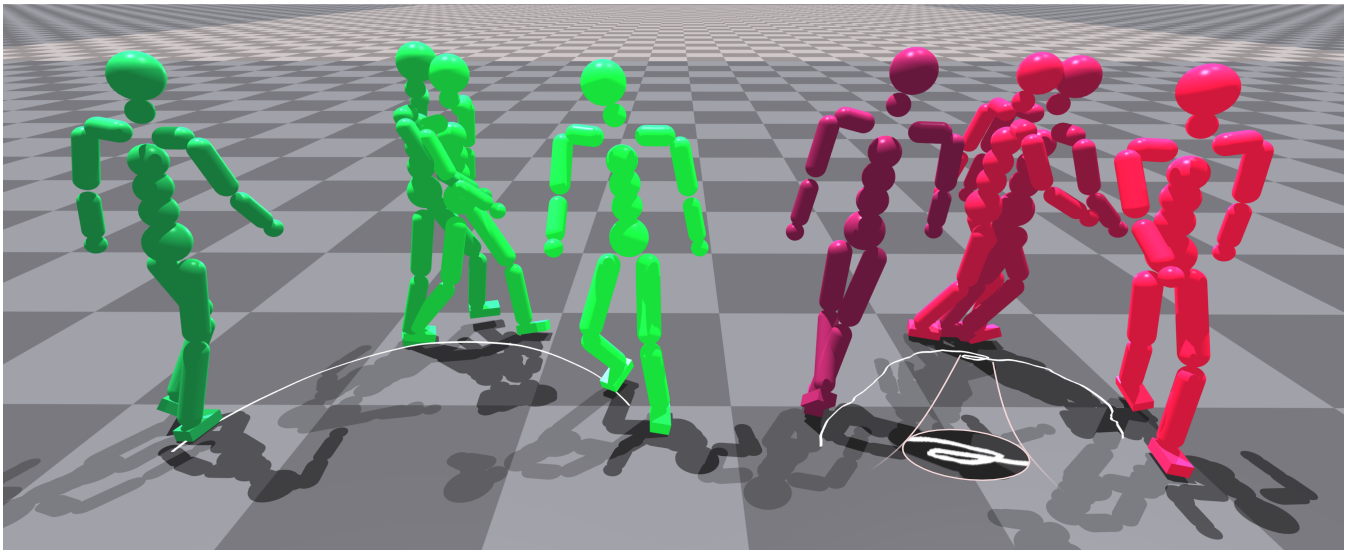

Figure 4: *A visualisation of our agent (green) applied to an animation containing foot skating (pink/red). The color gradient represents the flow of time, where brighter is later in the sequence. The white lines are the character's centers of mass (CoM) projected onto the horizontal plane. Note how in the ground truth the character floats backward between frames 2 and 3, while our agent follows a much smoother CoM trajectory.*

a real-world setting [HGS05]. In contrast, our character has a more natural, upright stance, causing the CoM projection to be within the base of support.

Our agent was also successful in improving the root motion of unseen ground truth animations that were flawed. In figure 4 we demonstrate an example, where the ground truth contained heavy foot skating and a generally unstable CoM trajectory. Our agent maintained a smoother CoM trajectory while eliminating foot skate. For full visualisations of these and other examples, we refer to the supplementary video which contains full side-by-side recordings of our agent with the ground truth.

4.2. Comparison to an Optical Reference

Our method is designed to improve IMU-based motion capture recordings, and specifically to rid them of physical implausibilities. However, to get an idea of how the resulting motions compare to the near ground truth actor motion, we include a comparison to reference motion, captured with both an IMU and an optical motion

capture setup. The full comparison can be viewed in the supplemental video.

Even though the improvement of our result with respect to the input IMU motion is evident in terms of nonphysical artefacts, the optical reference still outperforms in terms of naturalness of motion. This limitation is not unexpected since the agent has no direct incentive to improve the motion beyond removing artefacts. It still follows the input IMU motion closely, thus any imperfections that are not directly breaking the laws of physics will remain present in the results. One such example is the end pose, as shown in figure 5. Our agent (green) ends up in a different and slightly unnatural stance, with the left food behind the right and the toe touching the heel, due to the IMU input (pink) ending in this stance.

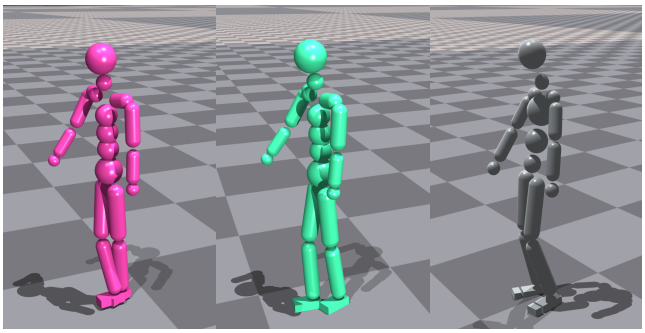

Figure 5: *A comparison between the IMU based input motion (pink, left), our result (green, centre), and an optical reference (gray, right). Note how in our result the self-penetrating feet are fixed, however the stance from the ground truth can not be recovered, as this information is lost in the IMU input.*

4.3. Data Ablation Study

A question we sought to answer was to what extent the type and mixture of the training data impact the performance of the agent. Generally speaking, optical motion capture data is high quality and low on noise while it is confined in recording space. IMU data, on the other hand, is easy to record, untethered, and low cost, but this comes at the expense of data quality and high noise. We trained models on different mixtures and sizes of datasets and evaluated them on our performance metrics.

The following 5 datasets were compared:

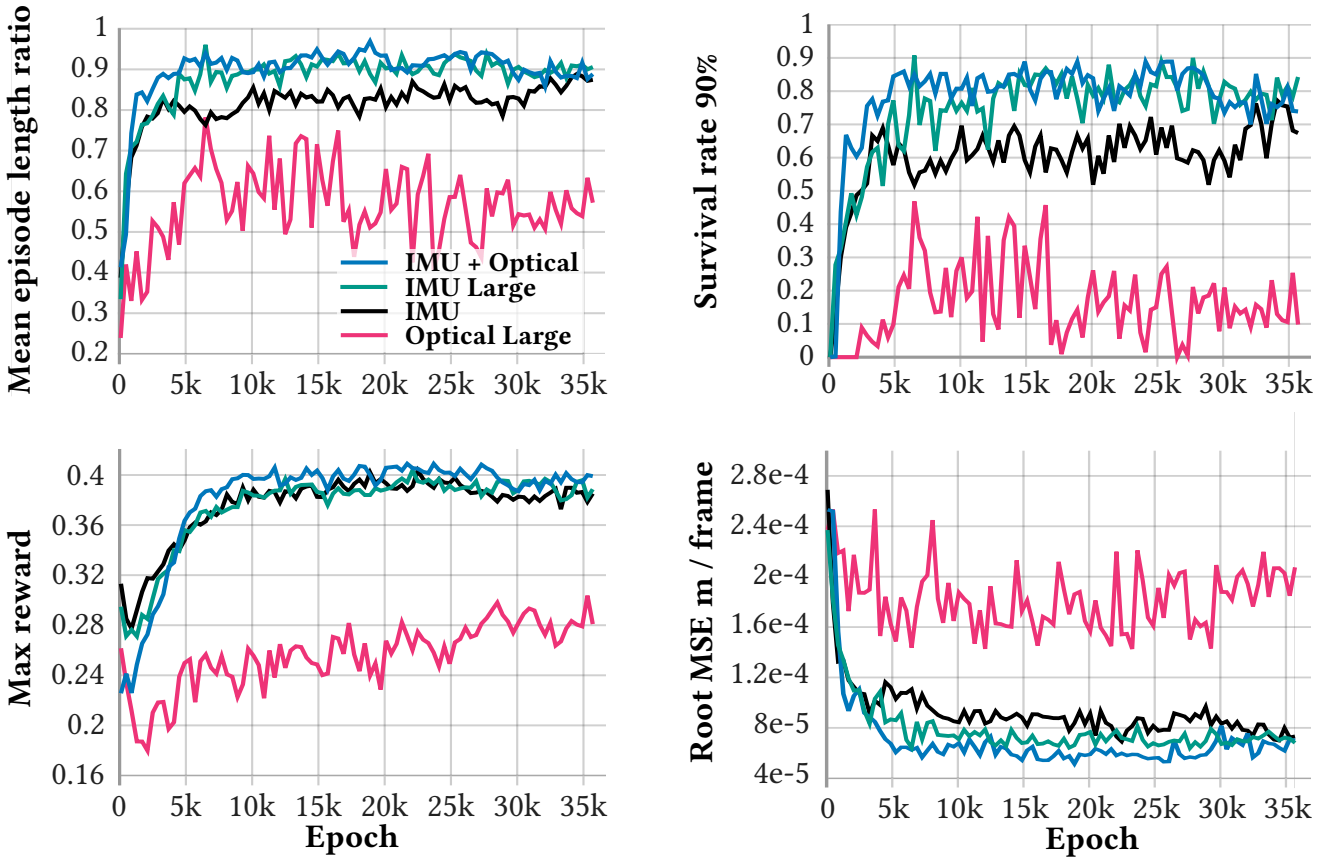

Figure 6: *Effects of training data modality in terms of performance metrics. The model trained using the IMU+Optical dataset outperforms other dataset configurations on all metrics, except the 90% survival rate. The model trained exclusively on optical data scores lowest on all metrics.*

1. **IMU:** This dataset contained 33 motions recorded with an IMU based motion capture suit, totaling 28925 frames at 60 frames per second. Some assets contained artefacts common to IMU setups, such as jittery limbs, self-penetration and foot skate.
2. **Optical** This dataset contains 17 high-quality motion assets recorded with high-end optical motion capture hardware. It contained a total of 10345 frames at 60 frames per second.
3. **IMU + Optical:** This dataset is the combination of the above two datasets (50 motion assets, 39270 frames at 60 frames per second).
4. **IMU Large:** To rule that the IMU+Optical agent outperformed the other agents on sheer data size, we also trained the single source agents on larger datasets. The IMU Large dataset contained 107 assets, with a total of 80102 frames of motion data.
5. **Optical Large:** This dataset contained 107 high-quality optical based motion assets with a total of 68392 frames.

Each agent was trained for approximately 35000 epochs. To get an indication of the agent's general performance the agents were evaluated with our test metrics on a dataset of unseen IMU-based motion assets. The actors recorded in this dataset were not present in the training data. The results for the best scoring epoch for each model can be found in table 4. The results for all epochs are plotted in figure 6, except for the agent trained on small optical data, as it did not manage to reproduce most of the motions in the test dataset.

The agent trained on a mixture of IMU and optical-based data outperformed the other agents, including those trained on larger bodies of motion data. It held the highest score for episode length ratio, max reward ratio, and root MSE per frame, while the IMU Large agent scored slightly better on the 90% survival rate metric.

This led us to conclude that introducing different types of data to the training dataset is beneficial and preferable over simply increasing the dataset size. Our combined IMU and optical motion capture dataset was about half the size of the IMU Large dataset, yet still performed better. This superior performance is rooted in two key factors: On the one hand, introducing different data types makes the agent more robust to different types of input data, which is crucial when evaluating unseen data. On the other hand, the high-quality motion data aids the agent in compensating for flaws in the IMU data.

The agent trained solely on optical motion data exhibited poor performance. The likely reason is a substantial change in the input data distribution across different motion capture sources. Consequently, training on one source and inferring on another is ineffective.

4.4. Configuration Ablation Study

In order to find the most robust agent for inference on unseen data, we compared different state and action configurations for our agent. In [BCHF19] the authors showed how a reduced state and action space was beneficial for their case (for simplicity called the Drecon configuration here). However, we argue that a configuration with a similar action space but with state information on all bones/joints is beneficial for robustness against unseen data. We therefore tested their proposed configuration against the configuration described in the method section of this work. Finally, we tested whether using the full action space would be beneficial for our case. Figure 7 shows six frames of an animation comparing all three configurations as well as the ground truth. While our agent (the green character) faithfully follows the reference motion, the Drecon configuration falls (from frame 4) after a sharp turn. The agent with full state and action space manages to perform the full motion, but introduces blandness to the motion, as witnessed, for example, by the lesser pronounced arm motion in frames 2, 4, 5 and 6.

Both the Drecon configuration and the full action space configuration were tested against our performance metrics, using the same test dataset as in the previous comparisons. The metric scores of the best epochs for this comparison can be found in the lower section of Table 4. The metrics over all training epochs are plotted in Figure 8.

Our agent trained with full state information and reduced action space (top row, IMU + Optical) performed best on all metrics,

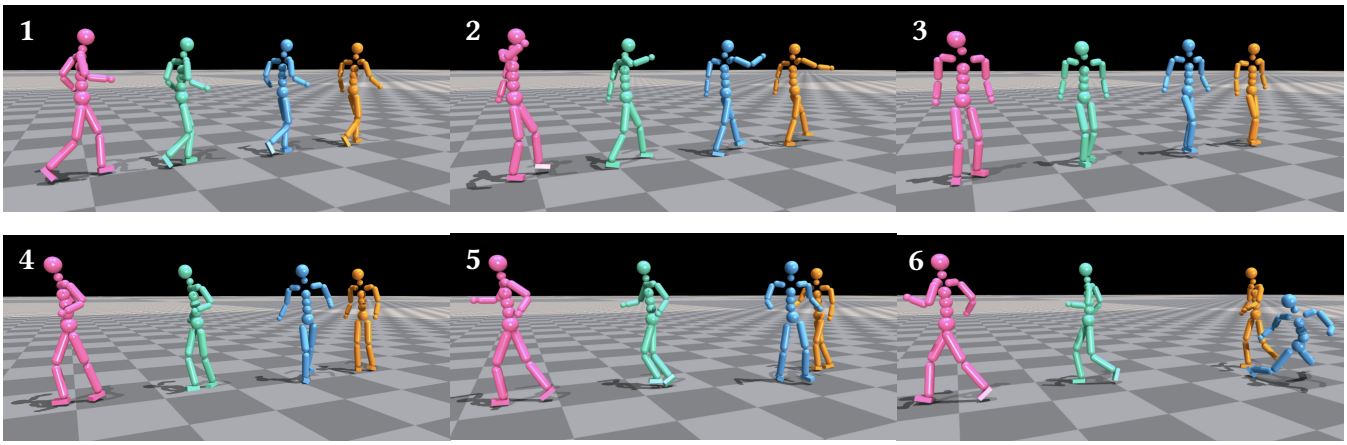

Figure 7: *A comparison of performance between configurations: The pink character is the ground truth, green is our configuration, blue has reduced state and action spaces, and orange has full state and action spaces. Note how the blue character falls in frames 5 and 6, while the orange character adds an increased blandness to the motion (frames 4 and 5).*

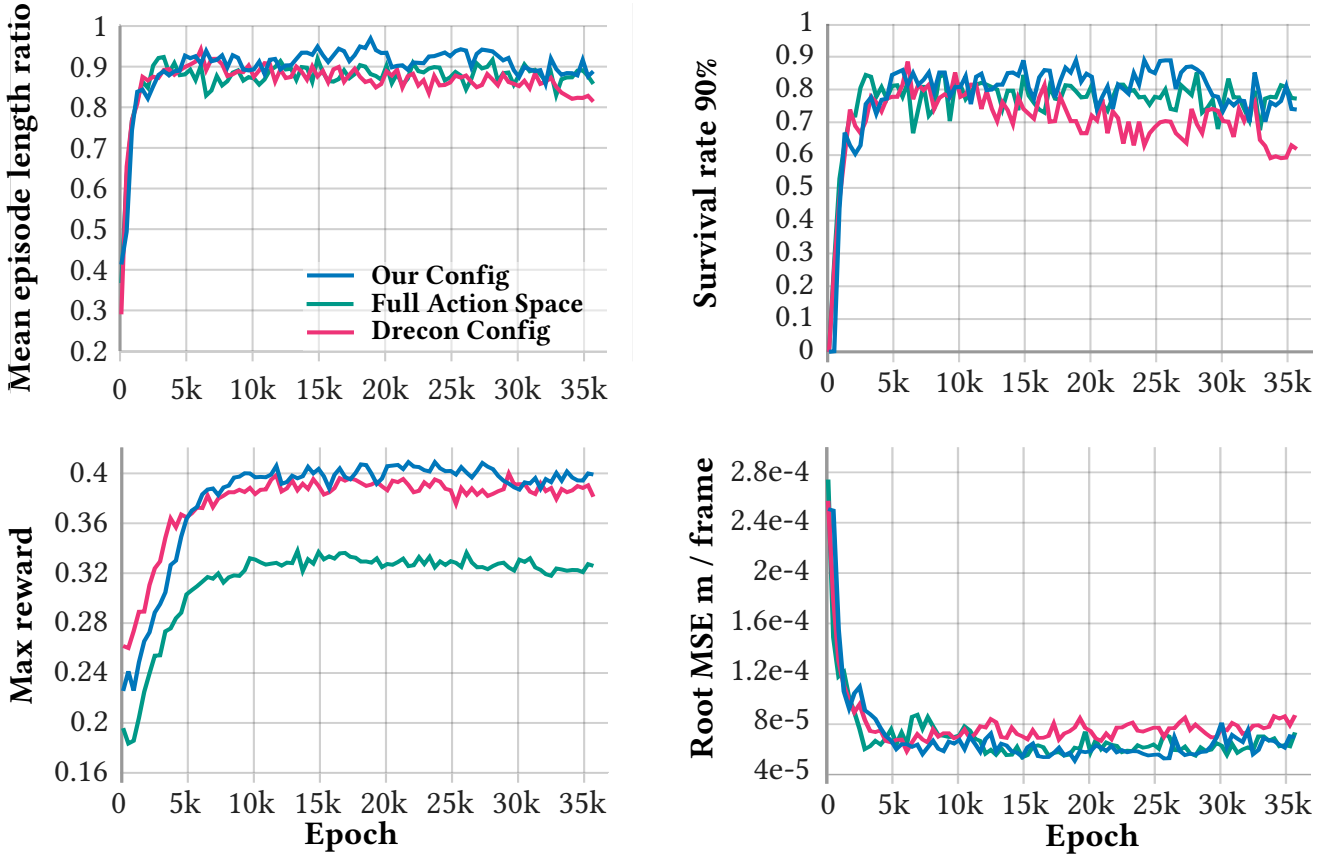

Figure 8: *Effects of state and action space configurations in terms of performance metrics. The model trained using our full state and reduced action space outperforms other configurations on all metrics, closely followed by the Drecon configuration from [BCHF19].*

closely followed by the Drecon configuration. We attribute this advantage to the agent having more information to go on than the Drecon agent, adding to its robustness against unseen data.

We confirmed the findings from [BCHF19] that using the full action space is not beneficial and that it does not contribute to the case of unseen data. This is due to the problem becoming unnecessarily complex for the policy, leading to reduced motion quality, as witnessed by the lower max reward ratio. This is further confirmed visually in the supplementary video.

5. Discussion and Conclusion

We demonstrated how motion mimicking using reinforcement learning can be effectively deployed to purge faulty motion data from physically implausible artefacts. Our agent was evaluated on a test dataset of IMU based motion capture recording from multiple actors, that were not included in the training data. It could reproduce sample motions faithfully and in their entirety in almost 90% of the cases while fixing artefacts such as self-collisions, footskating, and jittery limbs. Moreover, it produced more realistic centre of mass trajectories.

For training of our agent we used a mixture of high and low quality motion data from different sources. This mixture benefited the performance of our agent, making it more robust to unseen data. Even when the homogeneous dataset was larger then the mixed dataset, training with the mixed dataset yielded better performance. Training exclusively on high quality motion data did not produce an agent that was capable of generalising to unseen IMU based data.

Finally, we proposed a configuration of the observation and action spaces that improved our agent's performance on unseen data. The agent had access to information about all its joints, while it could only actuate a set of key joints. We compared this configuration to other configurations, one where both action and observation spaces were reduced, and one where the agent had access to the full observation and action space. Our configuration showed the best results during validation on unseen data.

6. Limitations

A limitation to our work is that we expect our agent to solve two contradicting problems: on the one hand the agent is incentivised to mimic motion closely through joint angle and position rewards, while on the other we want it to deviate from close mimicking when there is unnatural phenomena such as self-collisions. This nuance is not explicitly represented in the reward setup, occasionally causing our agent to try and mimic artefacts as close as possible without

breaking the laws of physics. This causes undesirable behaviours in the output motion, such as stumbling or intermediate steps. As a direction for future work, one proposed path to solving this contradiction is combining conventional rewards with a reward from a discriminator network, like the one proposed in [PMA*21], to reward the agent on a global style, rather than exactly mimicking the reference motion.

We believe that this will also improve the overall quality of the resulting motion. As simply adding high quality motions to the training data is not necessarily enough to force high quality motion output. As our comparison to optical reference motion showed, our method is largely depending on the input data from the IMU based motion capture hardware, even when that input motion is of low quality in terms of naturalness of motion. The result is that any flaws in the input motion that are not breaking any laws of physics are mimicked by our agent.

As a final remark, we observe that our agent fails in about 10% of the cases. This is likely due to out of distribution motions for which the agent has not learned actions, or sudden sharp movements causing our agent to trip. A way to improve the agent's robustness could be to include a longer forward time horizon to the agent's observation space. Currently the agent sees a single step ahead, multiple steps ahead could enable the agent to foresee and prepare for future events, thus increasing its robustness.

Acknowledgements

This project was funded in part by Innovationsfonden Danmark (grant no. 9065-00085A). We would like to express our gratitude to Rokoko Electronics for providing us with access to data, hardware and computational resources needed for this work.

References

[BCHF19] BERGAMIN K., CLAVET S., HOLDEN D., FORBES J. R.: Drecon: data-driven responsive control of physics-based characters. *ACM Transactions On Graphics (TOG) 38*, 6 (2019), 1–11. 3, 5, 6, 7, 10, 11

[BRRP97] BODENHEIMER B., ROSE C., ROSENTHAL S., PELLA J.: The process of motion capture: Dealing with the data. In *Computer Animation and Simulation '97* (Vienna, 1997), Thalmann D., van de Panne M., (Eds.), Springer Vienna, pp. 3–18. 2

[CK99] CHOI K.-J., KO H.-S.: On-line motion retargetting. In *Proceedings. Seventh Pacific Conference on Computer Graphics and Applications (Cat. No.PR00293)* (1999), pp. 32–42. doi:10.1109/PCCGA.1999.803346. 2

[CMM*18] CHENTANEZ N., MÜLLER M., MACKLIN M., MAKOVIYCHUK V., JESCHKE S.: Physics-based motion capture imitation with deep reinforcement learning. In *Proceedings of the 11th ACM SIGGRAPH Conference on Motion, Interaction and Games* (New York, NY, USA, 2018), Association for Computing Machinery, pp. 1–10. 3

[DAS*20] DONG Y., ARISTIDOU A., SHAMIR A., MAHLER M., JAIN E.: Adult2child: Motion style transfer using cyclegans. In *Proceedings of the 13th ACM SIGGRAPH Conference on Motion, Interaction and Games* (New York, NY, USA, 2020), MIG '20, Association for Computing Machinery. doi:10.1145/3424636.3426909. 2

[GBT06] GLARDON P., BOULIC R., THALMANN D.: Robust on-line adaptive footplant detection and enforcement for locomotion. *Vis. Comput. 22*, 3 (mar 2006), 194–209. doi:10.1007/s00371-006-0376-9. 2

[GM85] GIRARD M., MACIEJEWSKI A. A.: Computational modeling for the computer animation of legged figures. In *Proceedings of the 12th Annual Conference on Computer Graphics and Interactive Techniques* (New York, NY, USA, 1985), SIGGRAPH '85, Association for Computing Machinery, p. 263–270. doi:10.1145/325334.325244. 2

[GP12] GEIJTENBEEK T., PRONOST N.: Interactive character animation using simulated physics: A state-of-the-art review. In *Computer graphics forum* (2012), vol. 31, Wiley Online Library, pp. 2492–2515. 2

[HAB20] HENTER G. E., ALEXANDERSON S., BESKOW J.: Moglow: Probabilistic and controllable motion synthesis using normalising flows. *ACM Trans. Graph. 39*, 6 (nov 2020). 2

[HGS05] HOF A. L., GAZENDAM M., SINKE W.: The condition for dynamic stability. *Journal of biomechanics 38*, 1 (2005), 1–8. 8

[KSG02] KOVAR L., SCHREINER J., GLEICHER M.: Footskate cleanup for motion capture editing. In *Proceedings of the 2002 ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (New York, NY, USA, 2002), SCA '02, Association for Computing Machinery, p. 97–104. doi:10.1145/545261.545277. 2

[LL14] LU J., LIU X.: Foot plant detection for motion capture data by curve saliency. In *Fifth International Conference on Computing, Communications and Networking Technologies (ICCCNT)* (2014), pp. 1–6. doi:10.1109/ICCCNT.2014.6963001. 2

[LLLL21] LEE S., LEE S., LEE Y., LEE J.: Learning a family of motor skills from a single motion clip. *ACM Transactions on Graphics (TOG) 40*, 4 (2021), 1–13. 3

[LPLL19] LEE S., PARK M., LEE K., LEE J.: Scalable muscle-actuated human simulation and control. *ACM Transactions On Graphics (TOG) 38*, 4 (2019), 1–13. 3

[LS99] LEE J., SHIN S. Y.: A hierarchical approach to interactive motion editing for human-like figures. In *Proceedings of the 26th Annual Conference on Computer Graphics and Interactive Techniques* (USA, 1999), SIGGRAPH '99, ACM Press/Addison-Wesley Publishing Co., p. 39–48. doi:10.1145/311535.311539. 2

[LSCC20] LUO Y.-S., SOESENO J. H., CHEN T. P.-C., CHEN W.-C.: Carl: Controllable agent with reinforcement learning for quadruped locomotion. *ACM Transactions on Graphics (TOG) 39*, 4 (2020), 38–1. 3

[MHCH22] MOUROT L., HOYET L., CLERC F. L., HELLIER P.: Underpressure: Deep learning for foot contact detection, ground reaction force estimation and footskate cleanup. *Computer Graphics Forum 41*, 8 (2022), 195–206. doi:10.1111/cgf.14635. 2

[MHLC*22] MOUROT L., HOYET L., LE CLERC F., SCHNITZLER F., HELLIER P.: A survey on deep learning for skeleton-based human animation. In *Computer Graphics Forum* (2022), vol. 41, Wiley Online Library, pp. 122–157. 2

[MM21] MAKOVIICHUK D., MAKOVIYCHUK V.: rl-games: A high-performance framework for reinforcement learning. https://github.com/Denys88/rl_games, May 2021. 4, 6

[MTA*20] MEREL J., TUNYASUVUNAKOOL S., AHUJA A., TASSA Y., HASENCLEVER L., PHAM V., EREZ T., WAYNE G., HEESS N.: Catch & carry: reusable neural controllers for vision-guided whole-body tasks. *ACM Transactions on Graphics (TOG) 39*, 4 (2020), 39–1. 3

[MYT*21] MA L.-K., YANG Z., TONG X., GUO B., YIN K.: Learning and exploring motor skills with spacetime bounds. In *Computer Graphics Forum* (2021), vol. 40, Wiley Online Library, pp. 251–263. 2

[PALVdP18] PENG X. B., ABBEEL P., LEVINE S., VAN DE PANNE M.: Deepmimic: Example-guided deep reinforcement learning of physics-based character skills. *ACM Transactions On Graphics (TOG) 37*, 4 (2018), 1–14. 2, 6

[PGH*22] PENG X. B., GUO Y., HALPER L., LEVINE S., FIDLER S.: Ase: Large-scale reusable adversarial skill embeddings for physically simulated characters. *ACM Transactions On Graphics (TOG) 41*, 4 (2022), 1–17. 2

[PHO11] PRAŽÁK M., HOYET L., O'SULLIVAN C.: Perceptual evaluation of footskate cleanup. In *Proceedings of the 2011 ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (2011), pp. 287–294. 2

[PMA*21] PENG X. B., MA Z., ABBEEL P., LEVINE S., KANAZAWA A.: Amp: Adversarial motion priors for stylized physics-based character control. *ACM Transactions on Graphics (TOG) 40*, 4 (2021), 1–20. 2, 6, 12

[PP10] PEJSA T., PANDZIC I.: State of the art in example-based motion synthesis for virtual characters in interactive applications. *Computer Graphics Forum 29*, 1 (2010), 202–226. doi:10.1111/j.1467-8659.2009.01591.x. 2

[PRL*19] PARK S., RYU H., LEE S., LEE S., LEE J.: Learning predict-and-simulate policies from unorganized human motion data. *ACM Transactions on Graphics (TOG) 38*, 6 (2019), 1–11. 3

[RYC*23] REN J., YU C., CHEN S., MA X., PAN L., LIU Z.: Diffmimic: Efficient motion mimicking with differentiable physics. *arXiv preprint arXiv:2304.03274* (2023). 2

[SPL*21] SCHREINER P., PEREPICHKA M., LEWIS H., DARKNER S., KRY P. G., ERLEBEN K., ZORDAN V. B.: Global position prediction for interactive motion capture. *Proc. ACM Comput. Graph. Interact. Tech. 4*, 3 (sep 2021). doi:10.1145/3479985. 2

[SWD*17] SCHULMAN J., WOLSKI F., DHARIWAL P., RADFORD A., KLIMOV O.: Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347* (2017). 6

[TRG*23] TEVET G., RAAB S., GORDON B., SHAFIR Y., COHEN-OR D., BERMANO A. H.: Human motion diffusion model. In *The Eleventh International Conference on Learning Representations (ICLR)* (2023). 2

[VPHB*21] VALLE-PÉREZ G., HENTER G. E., BESKOW J., HOLZAPFEL A., OUDEYER P.-Y., ALEXANDERSON S.: Transflower: Probabilistic autoregressive dance generation with multimodal attention. *ACM Trans. Graph. 40*, 6 (dec 2021). doi:10.1145/3478513.3480570. 2

[WGH20] WON J., GOPINATH D., HODGINS J.: A scalable approach to control diverse behaviors for physically simulated characters. *ACM Transactions on Graphics (TOG) 39*, 4 (2020), 33–1. 3

[WGSF20] WANG T., GUO Y., SHUGRINA M., FIDLER S.: Unicon: Universal neural controller for physics-based character motion. *arXiv preprint arXiv:2011.15119* (2020). 3

[WL19] WON J., LEE J.: Learning body shape variation in physics-based characters. *ACM Transactions on Graphics (TOG) 38*, 6 (2019), 1–12. 3

[WWY22] WINKLER A., WON J., YE Y.: Questsim: Human motion tracking from sparse sensors with simulated avatars. In *SIGGRAPH Asia 2022 Conference Papers* (2022), pp. 1–8. 3

[XMN*22] XU J., MAKOVIYCHUK V., NARANG Y., RAMOS F., MATUSIK W., GARG A., MACKLIN M.: Accelerated policy learning with parallel differentiable simulation. *arXiv preprint arXiv:2204.07137* (2022). 2

[YLHX22] YE Y., LIU L., HU L., XIA S.: Neural3points: Learning to generate physically realistic full-body motion for virtual reality users. In *Computer Graphics Forum* (2022), vol. 41, Wiley Online Library, pp. 183–194. 3

[YSI*22] YUAN Y., SONG J., IQBAL U., VAHDAT A., KAUTZ J.: Physdiff: Physics-guided human motion diffusion model. *arXiv preprint arXiv:2212.02500* (2022). 3, 6

[YTY*23] YIN W., TU R., YIN H., KRAGIC D., KJELLSTRÖM H., BJÖRKMAN M.: Controllable motion synthesis and reconstruction with autoregressive diffusion models. In *2023 32th IEEE International Conference on Robot & Human Interactive Communication (RO-MAN)* (2023). 2

[YYB*23] YIN W., YIN H., BARAKA K., KRAGIC D., BJÖRKMAN M.: Dance style transfer with cross-modal transformer. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)* (January 2023), pp. 5058–5067. 2

[YYKB21] YIN W., YIN H., KRAGIC D., BJÖRKMAN M.: Graph-based normalizing flow for human motion generation and reconstruction. In *2021 30th IEEE International Conference on Robot & Human Interactive Communication (RO-MAN)* (2021), pp. 641–648. doi:10.1109/RO-MAN50785.2021.9515316. 2