Discovering Fatigued Movements for Virtual Character Animation

Noshaba Cheema  
noshaba.cheema@dfki.de  
DFKI, Max Planck Institute for Informatics, Saarland Informatics Campus  
Germany

Rui Xu  
rui.xu@dfki.de  
DFKI, Saarland Informatics Campus  
Germany

Nam Hee Kim  
namhee.kim@aalto.fi  
Aalto University  
Finland

Perttu Hämäläinen  
perttu.hamalainen@aalto.fi  
Aalto University  
Finland

Vladislav Golyanik  
golyanik@mpi-inf.mpg.de  
Max Planck Institute for Informatics, Saarland Informatics Campus  
Germany

Marc Habermann  
mhaberma@mpi-inf.mpg.de  
Max Planck Institute for Informatics, Saarland Informatics Campus  
Germany

Christian Theobalt  
theobalt@mpi-inf.mpg.de  
Max Planck Institute for Informatics, Saarland Informatics Campus  
Germany

Philipp Slusallek  
philipp.slusallek@dfki.de  
DFKI, Saarland Informatics Campus  
Germany

ABSTRACT
Virtual character animation and movement synthesis have advanced rapidly during recent years, especially through a combination of extensive motion capture datasets and machine learning. A remaining challenge is interactively simulating characters that fatigue when performing extended motions, which is indispensable for the realism of generated animations. However, capturing such movements is problematic, as performing movements like backflips with fatigued variations up to exhaustion raises capture cost and risk of injury. Surprisingly, little research has been done on faithful fatigue modeling. To address this, we propose a deep reinforcement learning-based approach, which—for the first time in literature—generates control policies for full-body physically simulated agents aware of cumulative fatigue. For this, we first leverage Generative Adversarial Imitation Learning (GAIL) to learn an expert policy for the skill; Second, we learn a fatigue policy by limiting the generated constant torque bounds based on endurance time to non-linear, state- and time-dependent limits in the joint-actuation space using a Three-Compartment Controller (3CC) model. Our results demonstrate that agents can adapt to different fatigue and rest rates interactively, and discover realistic recovery strategies without the need for any captured data of fatigued movement.

CCS CONCEPTS
• Computing methodologies → Animation; Physical simulation; Procedural animation; Adversarial learning; Motion capture.

KEYWORDS
character animation, physics-based animation, reinforcement learning, adversarial learning, cumulative fatigue modeling, biomechanics

ACM Reference Format:

1 INTRODUCTION
For many applications, ranging from games and films to robotics, synthesizing life-like and realistic character animation and control is a crucial yet challenging element. A relatively unexplored yet important area in this direction is the adequate simulation of physiological changes over time. In particular, this is true for cumulative fatigue, i.e., humans getting tired over time when performing strenuous motions. Simulating this effect plausibly aids to the overall realism in interactive applications, such as sports simulators, where athletes start to change their execution patterns due to muscle fatigue, and in games, when virtual characters run out of resources. To achieve this, current games use comparably simple heuristics and pre-defined kinematic animations in state-machines such as motion graphs [Kovar et al. 2002; Lee et al. 2002] to model character fatigue. However, many such methods do not incorporate
within one policy, letting characters fatigue faster or slower than their baseline. Previous work targeting character animation can be roughly categorized into data-driven kinematic-based methods (KM) [Bütter and Clavet 2015; Holden et al. 2017; Levine et al. 2012; Min and Chai 2012; Starke et al. 2020] and physics-based simulation, paired with Deep Reinforcement Learning (DRL) [Baker et al. 2019; Jiang et al. 2019; Pathak et al. 2017; Peng et al. 2021; Yin et al. 2021]. KMs provide an easy means to make use of the inherent realism that comes from captured or hand-animated motion data, lacking the ability to generalize to novel behaviours. Meanwhile, characters animated via KMs lack the ability to react to dynamic stimuli such as external perturbations, unless the prior motion capture incorporates the vast amount of interaction and perturbation scenarios.

On the other hand, DRL methods provide a promising direction in this field, as even sparse rewards or space constraints already allow for the automatic generation of interactive movements and novel emergent behaviours without the need for explicitly capturing variations of data that fulfill the given constraints. However, what kind of rewards or constraints constitute for realistic movements and behaviours remains a fundamental challenge in animation research. Interestingly, none of the previous works in either of the two categories has focused on modeling character fatigue in a biologically plausible fashion while overall animation realism would benefit from such an approach. Only a few works in the biomechanically-based simulation literature [Cheema et al. 2020; Komura et al. 2000] have looked into this problem. However, previous works have only been applied to carefully hand-crafted single limb settings, as they require expensive musculoskeletal models [Komura et al. 2000] or careful reward modeling and hyper-parameter tuning [Cheema et al. 2020] and are not able to demonstrate the emergence of fatigue and recovery behaviours to the same extent as our work.

In this paper, we propose a novel fatigue-aware policy generation framework for physically simulated characters, which allows for the emergence of realistic fatigue and recovery effects. For this we explore a two-step approach: First, we pre-train the policy to allow for a viable number of stable behaviours to emerge, and a well-behaved initial state for fatigue transfer-learning. Second, based on the learned stable behaviours, we refine the policy by modifying the emerging torque limits to nonlinear, state- and time-dependent limits using a Three-Compartment Controller (3CC) model [Xia and Frey Law 2008], which we adapt from the biomechanics literature in a new context not envisioned by the original paper. To show the effectiveness of our framework, we integrate our fatigue module into a state-of-the-art generative adversarial imitation learning (GAIL) framework [Ho and Ermon 2016; Peng et al. 2021; Torabi et al. 2018].

In summary, the main contributions of this work are as follows:

- **Novel Functionality.** The first approach for emergent fatigue and recovery behaviours in full-body Reinforcement Learning for 3D virtual character animation.
- **Efficient Torque-based Fatigue-System for Animation.** The first full-body fatigue system based solely on joint actuation torques for interactive character simulation.
- **Interactive Fatigue and Rest Control.** The use of a Three-Compartment Controller (3CC) fatigue model to limit joint actuation torques based on observed fatigue and residual strength capacity, which allows for interactive control of different fitness levels within one policy.

Our results demonstrate several emergent fatigue behaviours during repetitive athletic tasks, such as arms bending during cartwheels and decreasing kick and jump height after athletic martial arts kicks (Fig. 1), as well as waiting behaviours to recover from the fatigue to be able to continue. Our agents learn unseen motion patterns while resting in order to most effectively recover from the experienced fatigue. For example, our agent learns to effectively compensate the momentum after a backflip or a cartwheel while ending up in a motion state where motor units start to recover. As the risk of injury could prohibit motion capture of fatigued yet complex movements, our method brings an added benefit of bypassing this constraint.

## 2 RELATED WORK

This section reviews physics-based approaches, deep reinforcement learning, musculoskeletal methods and techniques supporting fatigue.

**Note** that as fatigue modeling relies on the physical forces, purely kinematic methods cannot be applied to our problem unless one explicitly captures fatigued motions.

**Physics-based Methods.** Physics-based methods allow movement generation with physical realism and environmental interaction; they give direct insights on the forces required and being applied to the character for a given task by leveraging a more general knowledge of the physical equations of motion [Geijtenbeek et al. 2013; Raibert and Hodgins 1991; Wampler et al. 2014]. A fundamental challenge in physics-based approaches is the design of controllers for simulated characters. Task-specific controllers (e.g. for locomotion), achieved significant success [Coros et al. 2010; Felis and Mombaur 2016; Geijtenbeek et al. 2013; Lee et al. 2010; Ye and Liu 2010a; Yin et al. 2007]. However, such manually designed controllers remain hard to generalize to diverse movements and tasks. With the wide availability of motion capture data, tracking-based controllers have become a popular research domain [Hämäläinen et al. 2015; Tassa et al. 2018].
et al. 2012; Wampler et al. 2014; Ye and Liu 2010b], though they remain limited in motion quality and long-term planning. More recently, deep reinforcement learning-based methods have become a promising research direction to account for long-term planning as well as emergent and reactive behaviour – three aspects necessary for emergent fatigue in behaviour over multiple repetitions.

**Deep Reinforcement Learning.** Deep reinforcement learning (DRL) has been successfully applied to physics-based characters animation [Liu and Hodgins 2017; Peng et al. 2016; Teh et al. 2017]. Here, policy gradient methods emerged for continuous control problems [Schulman et al. 2015, 2017; Sutton et al. 1998]. Imitation learning addresses the challenge of designing task-specific reward functions by learning a policy from examples by explicitly tracking the sequence of target poses in the motion clip [Liu and Hodgins 2018; Peng et al. 2018]. While this technique can imitate a single motion clip, it becomes difficult to scale without including high-level motion planners [Bergamin et al. 2019; Lee et al. 2021b,a; Park et al. 2019; Won et al. 2020, 2021; Zhang et al. 2023] or pose-based control using model-based RL [Fussell et al. 2021; Yao et al. 2022] or behavioural cloning [Won et al. 2022], which is prone to drift if small amounts of demonstrations are available [Ross et al. 2011]. Recently, methods based on generative adversarial imitation learning (GAIL) have shown to be an appealing alternative [Bae et al. 2023; Hassan et al. 2023; Lee et al. 2022; Peng et al. 2022, 2021; Xu and Karamouzas 2021], where an adversarial discriminator is trained to serve as an objective function for training a control policy to imitate the demonstrations. We make use of this technique to learn highly diverse and athletic movements. However, while these methods are able to generate diverse and natural looking movements, they lack bio-mechanical insights which are important for movement realism.

**Musculoskeletal Methods and Biomechanical Cumulative Fatigue.** Several works [Ackermann and van den Bogert 2012; Anderson and Pandy 2001; Geyer and Herr 2010; Ijspeert et al. 2007; Maffrey et al. 2008; Taga 1995; Thelen et al. 2003] developed musculoskeletal models that use biomimetic muscles and tendons to simulate a variety of human and animal motions. Controlling a muscle-based virtual character was also explored in computer animation, from upper-limb [Lee et al. 2018, 2009; Lee and Terzopoulos 2006; Sueda et al. 2008; Tsang et al. 2005], to lower-limb [Park et al. 2022; Wang et al. 2012], and full-body movements [Geijtenbeek et al. 2013; Jiang et al. 2019; Lee et al. 2019, 2014; Wang et al. 2012]. Such methods are computationally expensive, especially for interactive applications such as games. Jiang et al. [2019] convert an optimal control problem in the muscle actuation space to an equal problem in the joint-actuation space. The generated torque limits do not take accumulated fatigue variation over time into account. Musculoskeletal approaches to predicting muscle fatigue are based on detailed muscle activation patterns [Ding et al. 2000; Giat et al. 1996, 1993]. These approaches incorporate fatigue as a modifier of relatively complex muscle models. While they can provide realistic predictions of forces for isolated muscles, they are cumbersome for joint or whole body applications. In contrast, Liu et al. [2002] proposed a computationally efficient motor unit (MU)-based fatigue model, using three muscle activation states to estimate perceived biomechanical fatigue: resting, activated and fatigued. Improving upon this model, Xia and Frey Law [2008] introduced a Three-Compartment Controller (3CC) model for dynamic load conditions, eliminating the need for explicit modeling of muscle actuators.

**Cumulative Fatigue Modeling in Simulated Characters.** To the best of our knowledge little work has been done in this area. Kider Jr et al. [2011] captured extensive amounts of motion capture and biosignal data, including EKG, BVP, GSR, respiration, and skin temperature, to estimate fatigue of human characters. However, capturing data for all variances is time-consuming and expensive. Komura et al. [2000] make use of a musculoskeletal model, which re-targets existing motion clips to fatigued animations automatically using the musculoskeletal fatigue model Giat et al. [1996, 1993] for lower-body movements. While they achieve variance over time based on biomechanically accurate fatigue assumptions, their method needs an expensive musculoskeletal model and cannot account for any emergent recovery behaviours. Cheema et al. [2020] use a Three-Compartment-Controller (3CC-r) model [Looft et al. 2018; Xia and Frey Law 2008] in a fatigue-related reward function, which does not require expensive modeling and simulation of muscle-tendons. They predict ergonomic differences of user interface configurations with a single arm model in a pointing task. However, they do not consider acrobatic full-body movements. Additionally, none of the mentioned works indicate the emergence of rest behaviours to the same extent as our method and have only been applied to carefully crafted single limb settings with limited movements.

### 3 PRELIMINARIES - 3CC MODEL

We first review how cumulative fatigue can be modeled efficiently using only joint actuation torques with a 3CC-model [Looft et al. 2018; Xia and Frey Law 2008], which has been used for ergonomic assessment of endurance times and fatigue in biomechanics [Frey-Law et al. 2012b] and HCI [Cheema et al. 2020; Jang et al. 2017].

**Motor Units.** The 3CC model assumes motor units (MUs) to be in one of three possible states (compartments): 1) active – MUs contributing to the task 2) fatigued – fatigued MUs without activation 3) resting – inactive MUs not required for the task.

These are usually expressed as a percentage of maximum voluntary contraction (%MVC), which can practically be expressed as percentage of maximum voluntary force (%MVF) or torque (%MVT). Rested MUs (M₀) become activated (M₁) once a target load (TL) needs to be held. Active MUs are then directly contributing to the task. Once an MU is activated, its force decays over time and becomes fatigued (M₂). An initial non-fatigued state starts out with M₀ = 100% and M₁ = M₂ = 0%. The following system of equations describes the change of rate over time for each compartment:

\[
\frac{dM_0}{dt} = C(t) - F \cdot M_0 \tag{1a}
\]

\[
\frac{dM_1}{dt} = -C(t) + R_e \cdot M_F \tag{1b}
\]

\[
\frac{dM_2}{dt} = F \cdot M_A - R_e \cdot M_F \tag{1c}
\]

Here, \( R_e \) is defined as

\[
R_e = \begin{cases} 
    r \cdot R & \text{if } M_A \geq TL \\
    R & \text{else,}
\end{cases}
\]
where $F$ and $R$ denote the fatigue and recovery coefficients, and $r$ as an additional rest recovery multiplier for intermittent tasks [Looft et al. 2018]. A change in $F$ denotes the change of rate in fatique, whereas a change in $R$ indicates the overall rate of recovery, as well as an upper bound for maximal fatigue and loads that can be held indefinitely. For example, $R = F \cdot 0.2$ indicates that 20% MVC can be held indefinitely, which would be in accordance to empirical studies [Rohmert 1960]. In this case the limit of $M_F$ would be at 80%. An increase of $r$ indicates an increased recovery rate during tasks with intermittent rest periods when $r > 1$. $C(t)$ in Eq. (1a) and (1b) is a bounded proportional controller, which produces the force required for the target load ($TL$) by controlling the size of $M_A$ and $M_R$.

**Motor Activation-Deactivation Drive $C(t)$.** To obtain behaviours matching muscle physiology - e.g., active MUs decaying over time - control theory is applied. Therefore, $C(t)$ is introduced as a muscle activation-deactivation drive between rested and active MUs:

$$C(t) = \begin{cases} L_R \cdot (TL - M_A) & \text{if } M_A \geq TL \\ L_D \cdot (TL - M_A) & \text{if } M_A < TL \text{ and } M_R > TL - M_A \\ L_D \cdot M_R & \text{if } M_A < TL \text{ and } M_R \leq TL - M_A. \end{cases}$$  

(3)

The three cases can be described in the following way:

- **Case 1:** If there are more active motor units $M_A$ than required for the target load $TL$, then $M_A$ decays and $M_R$ increases in Eq. (1), which makes the muscle go into a recovery state. In this case $C(t)$ becomes negative.
- **Case 2:** When there are not enough active motor units $M_A$ compared to the required target $TL$ but the difference is smaller than the available rested MUs $M_R$, then rested MUs $M_R$ become active MUs $M_A$. In this case $C(t)$ is positive and greater or equal than $F \cdot M_A$ in Eq. (1a).
- **Case 3:** When there are not enough active motor units $M_A$ compared to the required target $TL$ and not enough rested MUs $M_R$ to compensate the discrepancy, the muscle starts to fatigue and the target load cannot be held any longer. In this case, $C(t)$ is positive but smaller than $F \cdot M_A$ in Eq. (1a). Here, $M_A$ decays and $M_R$ becomes (near) zero. $L_D$ and $L_R$ are muscle force development and relaxation factors, which describe the sensitivity towards the target load. Since the time course of either is negligible compared to the time course of fatigue (e.g. varying $L_D/L_R$ from 2 to 50, or a change of 2500%, only alters the endurance time by 10%), Xia and Frey Law [2008] set the same arbitrary value to each $L_D = L_R = 10$.

**Residual Capacity.** Residual Capacity (RC) describes the remaining motor strength capabilities or stamina due to fatigue in percent. 0% indicates no strength reserve, and 100% indicates full non-fatigued strength:

$$RC(t) = M_A + M_R = 100\% - M_F.$$  

(4)

While the 3CC model is a great analysis tool for fatigue and endurance time estimation [Cheema et al. 2020; Frey Law and Avin 2010; Jang et al. 2017], it does not directly lend itself to create full-body character animations modeling the described fatigue effects in Sec. 1. To approach this challenge, we leverage Generative Adversarial Imitation Learning (GAIL) [Ho and Ermon 2016; Peng et al. 2022, 2021; Torabi et al. 2018] and constrain the action space based on the computed Residual Capacity of the 3CC model, which we describe in more detail in the following section.

### 4 Fatigue Modeling for Character Animation

Our method takes as input a motion clip $M$ of the full-body skeleton of the humanoid character represented by a sequence of poses $\hat{q}_t$. This motion clip does not contain varying fatigue levels. Nonetheless, our goal is to generate an animation that mimics the original behaviour while the character fatigues over time and learns plausible recovery strategies. At the technical core, we explore a two-step approach that effectively blends ideas from biomechanical cumulative fatigue modeling, biologically-inspired torque limit constraining, and Deep Reinforcement Learning to enable the emergence of realistic symptoms of fatigue in character animation (see Fig. 2).

We first pre-train the policies on the reference motions to estimate the maximum constant torque bounds $T_{\text{max}}$ across the tasks. The actions $a_t := (u_t, \beta_t)$ at time $t$ from the policy $\pi$ specify target positions $u_t$ for PD-controllers positioned at each of the character’s joints and a stiffness and damping multiplier $\beta_t$, similar to [Yuan et al. 2021], which we query at the policy frequency. Modulating stiffness and damping introduces the possibility for the character to relax and tense its whole body, which is appropriate for the context of fatigue modeling, as fatigued virtual characters using
proportion-derivative (PD) controllers with fixed stiffness/damping parameters may choose overly stiff/conservative motions instead of relaxed motions. The output torques are then applied to the character physics simulation (Sec. 4.1). Once an expert policy has been learned, we use transfer learning to learn a policy, which is able to adapt to fatigue by constraining the torque-bounds over time based on RC computed by the 3CC fatigue modules resulting in $T_{\text{max}}$ (Sec. 4.1). This forces the policy to handle lower torque levels and discover fatigue and rest behaviors in attempt to fulfill the task based on the given constraints. Importantly, the model can be trained on a single ($F, R, r$)-triplet and adapt to novel triplets during inference, which makes training more efficient. Once trained, the agents exhibit unseen fatigued movement patterns and unseen rest recovery strategies emerge to overcome the loss of strength.

### 4.1 Imitation Objective and Torque-Estimation

In this section, we describe the pre-training. We start with a general formulation of a DRL problem, then continue with the imitation objective, and finally describe the constant torque-bound estimation.

**RL Problem Formulation.** At each time step $t$, the agent observes a state $s_t$ based on its environment observations and samples an action $a_t$ from a policy $\pi(a_t|s_t)$ in accordance to the observed state, which leads to a new state $s_{t+1}$ and a reward $r_t = r(s_t, a_t, s_{t+1})$. The agent’s objective is to maximize its expected discounted return $J(\pi)$ [Sutton et al. 1998]

$$J(\pi) = \mathbb{E}_{p(\tau|\pi)} \left[ \sum_{t=0}^{T-1} \gamma^t r_t \right],$$

(5)

where $p(\tau|\pi) = p(s_0) \prod_{t=1}^{T-1} p(s_t|s_{t-1}, a_t)\pi(a_t|s_t)$ represents the likelihood of the trajectory $\tau = \{(s_t, a_t, r_t)\}_{t=0}^{T-1}, s_T\}$ under a policy $\pi$. Here, $p(s_0)$ denotes the initial state distribution, $T$ is the time horizon of a trajectory, and $\gamma \in [0, 1]$ is the discount factor. To design a reward objective, which can imitate diverse athletic movements, we leverage Generative Adversarial Imitation Learning (GAIL) [Ho and Ermon 2016; Peng et al. 2022, 2021].

**Imitation Objective.** In GAIL, the objective to imitate a given task modeled as a discriminator $D(s, a)$, which is trained to predict whether a given state $s$ and action $a$ is sampled from the demonstrations $M$ or generated from the policy $\pi$ [Ho and Ermon 2016]. This formulation of GAIL requires access to the demonstrator’s actions, which, however, are not given when only motion clips are provided as demonstrations. Similar to Torabi et al. [2018], we train the discriminator on state transitions $D(s, s')$ instead of state-action pairs $D(s, a)\$ to overcome this limitation:

$$\arg\min_D \mathbb{E}_{p(\pi|s')} \left[ \log \left( D(s, s') \right) \right] - \mathbb{E}_{p(\pi|s')} \left[ \log \left( 1 - D(s, s') \right) \right],$$

(6)

where $p(\pi|s')$ and $p(\pi|s')$ denote the likelihoods of observing a state transition from state $s$ to $s'$ in the dataset $\mathcal{M}$, and following the policy $\pi$, respectively. Additionally, we incorporate the gradient penalty regularizers [Peng et al. 2021]. The discriminator is then trained using the following objective

$$\arg\min_D \mathbb{E}_{p(\pi|s')} \left[ \log \left( D(s, s') \right) \right] - \mathbb{E}_{p(\pi|s')} \left[ \log \left( 1 - D(s, s') \right) \right] + \lambda \mathbb{E}_{D} \left[ \mathbb{E}_{\varphi|\pi} \left[ \mathbb{E}_{\zeta} \left[ \left\| \nabla_{\phi} D(\varphi) |_{\zeta=(s,d')} \right\|^2 \right] \right] \right],$$

(7)

where $\lambda$ is a gradient penalty coefficient. Akin to Peng et al. [2022], we use the imitation objective for an adversarial imitation policy by defining the reward objective in Eq. (5) as

$$r_t = -\log(1 - D(\Phi(s_t), \Phi(s_{t+1}))),$$

(8)

where $\Phi(s_t)$ denotes a feature map based on the state space $s_t$.

**Torque Estimation.** We use the following PD-controller formulation to estimate the joint torque bounds and actuation torques [Featherstone 2014; Yuan et al. 2021] at every control query:

$$T_d = \beta_d k_d (u_d - \hat{u}_d) - \beta_d k_d \hat{u}_d$$

(9)

where $u_d$ denotes the target orientation and $\beta_d$ a stiffness and damping modulation parameter given from the policy’s action $a_d := (u_d, \beta_d)$. $\hat{u}_d$ denotes the current orientation of the DoF $d$ and $\dot{u}_d$ the current velocity. $k_p$ and $k_d$ specify constant stiffness and damping parameters. The pre-training has the nice side-effect that we can automatically estimate maximum torques as

$$T_{\text{max}d} := \max(T_d, T_{\text{max}d})$$

(10)

with $T_{\text{max}d} = 0$, which avoids manual or grid search for this hyperparameter. Additionally, we consider physiological symmetries by ensuring that two symmetric joints, e.g. left and right elbow, have the same value by computing the minimum between the two, as lower energy movements tend to look more natural [Yu et al. 2018]. In addition, we found we are able to greatly reduce outliers associated to the loss of strength for potential $T_{\text{max}d}$ candidates. We note that our method works for hand-crafted torque limits, as well as limits which depend on a single and multiple motions. In these cases, just the rate of fatigue and recovery for an $(F, R, r)$ setting changes.

**Transfer Learning: Fatigue-based Torque Limits.** Inspired by Jiang et al. [2019], we limit constant torque-bounds to nonlinear state-dependent limits in the joint actuation space. While their method allows for bio-mechanically enhanced torque-based actuation, it does not allow for variance over time from cumulative fatigue. Thus, we multiply the residual capacity for each 3CC-model with the maximum torque bounds found in the previous stage and then use transfer learning to make the policy adapt to the loss of strength as explained next.

**Fatigued Torque Bounds.** We assume the target load $TL$ to be the incoming joint actuator torques of the respective PD-controller of each DoF, which the character requires to reach the target position. Each DoF $d$ is modeled by a 3CC model as

$$T_{d}(T_d) = \frac{T_d}{T_{\text{max}d}} \cdot 100\%$$

(11)

being the ratio of the incoming actuator torque $T_d$ computed by the respective PD-controller and the constant maximum torque-bound found in the previous step representing the percentage of maximum voluntary contraction %MVC. $M_{\text{dA}}, M_{\text{dP}}$, and $M_{\text{dP}}$ are computed in accordance to the incoming target load per DoF using Eq. (1)–(3). To estimate the fatigued torque bounds $T_{\text{max}d}$, we leverage the
residual capacity $RC$ as a time-varying multiplier to the previously found torque bound limits:

$$\tilde{T}_{\text{max},d} = RC_{d} \cdot T_{\text{max},d},$$

(12)

where $RC_{d} = 100\% - M_{P_{d}}$ (see Sec. 3). The final fatigued torque $\tilde{T}_{d}$ applied to the environment for each DoF is computed by clipping the incoming joint actuator torque $T_{d}$ within $RC_{d} \cdot [-T_{\text{max},d}, T_{\text{max},d}]$ with $T_{d}$ being defined as

$$\tilde{T}_{d} \in [-T_{\text{max},d}, T_{\text{max},d}] = [-RC_{d} \cdot T_{\text{max},d}, RC_{d} \cdot T_{\text{max},d}].$$

(13)

Transfer Learning. Simply reducing the joint actuator torques of an agent with a policy trained with full torque-bounds will let the agent fall as the policy never learned to deal with loss of strength. Thus, we apply transfer learning, where the policy is trained on the time-varying torque outputs based on the Residual Capacity. While $R$ and $F$ values are joint specific [Frey-Law et al. 2012b; Loof et al. 2018] accounting for varying endurance times [Frey Law and Avin 2010], we use one $F$, $R$, and $r$ for the whole character as a design choice for usability and simple interactivity. We note that despite training on one $(F, R, r)$ triplet our policy is able to adapt to new pairs during inference. This is due to the fact that a single $(F, R, r)$-triplet alone already corresponds to multiple levels of Residual Capacity over time – making it easy for the agent to adapt to another combination even though the change of rate of $RC$ may differ. As such our method can be viewed as a framework for generating a range of motor skills from a single motion clip [Lee et al. 2021a], where the skills are parameterized by fatigue.

5 MODEL REPRESENTATION

States and Actions. We evaluate our framework using the 28 DoF humanoid character from Peng et al. [2021], as well as their state-space $s_t$ representation in the IsaacGym implementation with the inclusion of the character’s local root rotation [Peng et al. 2022], as well as the fatigued motor units $M_{P}$ in the observation space, totalling a state space of 133 dimensions. The character’s local coordinate frame is defined in as in [Peng et al. 2022, 2021]. Each action $a_t := (\mathbf{u}_t, \beta_t)$ specifies the target rotations $\mathbf{u}_t$ and stiffness/damping multiplier $\beta_t$ for the PD-controllers at each of the character’s joints. This results in a 29D action space – one action per DoF, as well as the stiffness/damping multiplier. $M_{P}$ is randomized during expert training to make the policy agnostic to it before it adapts to it in the fine-tuning stage.

Network Architecture. The policy $\pi(a_t|s_t)$ is represented as a neural network, with the action distribution modeled as a Gaussian, where the state-dependent mean $\mu(s_t)$ and the diagonal covariance matrix $\Sigma$ are specified by the network output $\pi(a_t|s_t) = \mathcal{N}(\mu(s_t), \Sigma)$. The mean is specified by an MLP consisting of two fully-connected hidden layers of 1024 and 512 Rectified Linear Units (ReLU) [Nair and Hinton 2010], followed by a linear output layer. The values of the covariance matrix $\Sigma = \text{diag}(\sigma_1, \sigma_2, \ldots)$ are manually specified [Peng et al. 2021] and fixed over the course of training. The value function $V(s_t)$ and the discriminator $D(s_t, s_{t+1})$ are modeled by separate networks with a similar architecture as the policy.

6 EVALUATION AND RESULTS

We evaluate our method on five diverse movement skills – backflip, cartwheel, hopping and locomotion from the CMU dataset [CMU] provided in the Isaac Gym [Makoviychuk et al. 2021] environment, as well as the 360 tornado kick from the SFU dataset [SFU]. The experiments below evaluate the following aspects of our method: First, compared to constant torque limits, our state and time-varying torque limits push the policy network towards movement strategies and patterns not present in the input data $M$ to overcome the loss of strength. Second, the found recovery strategies resemble human-like strategies for resting. All experiments are carried out using the high-performance GPU-based physics simulator Isaac Gym [Makoviychuk et al. 2021]. During training 4096 environments are simulated in parallel on a single NVIDIA V100 GPU with a simulation frequency of 120Hz, while the policy operates at 30 Hz. All neural networks are trained using PyTorch [Paszke et al. 2019]. Gradient updates are performed via Proximal Policy Optimization [Schulman et al. 2017] with a learning rate of $5 \times 10^{-5}$. We use an episode length of 300 during pre-training and an episode length of 1000 during fatigue transfer to learn the accumulation of fatigue. The gradient penalty coefficient $w_{pg}$ in Eq. (6) is set to 0.2 for all but locomotion, for which it is set to 5 [Makoviychuk et al. 2021]. Additional hyper-parameter settings and implementation details can be found in the supplementary document. We use the “Humanoid AMP”[Peng et al. 2021] character provided in Isaac Gym with 28 internal DoFs and its corresponding rigid body and joint properties. Stiffness and damping parameters were set to custom values in accordance to the realistic proportions of a real-life human male. $M_{A}$, $M_{F}$ and $M_{R}$ are randomized at every environment reset.

6.1 Fatigue Training

We first note that simply taking a pre-trained policy and adjusting the torque limits during inference only lets the character fall into a termination state and not learn any realistic recovery strategies because the character never learned to deal with less torque as can be seen in Fig. 3. Thus, we employ a simple transfer learning procedure where we train the expert policies for several iterations until stable behaviours arise. We disable fatigue behaviour during this pre-training phase. As the observation still contains $M_{F}$ values for the expert policy training, we ensure that the expert policies are agnostic to the $M_{F}$ values by randomizing the $M_{F}$ values at every environment step during this phase, as a strategy for domain randomization [Tobin et al. 2017]. More specifically, we train these expert policies for 2000 iterations for running and 4000 for others. For transfer learning, we apply 2000 iterations of additional training for each motion, using the corresponding expert policy as the warm-starting point. We input $F = 1, R = 0.01, r = 1$ for all transfer learning iterations. The reference torque estimates for computing $TL$ values are given by the expert policies. During the transfer learning phase, we randomize the fatigue state at uniform at each reset of the training episode, as to capture as much variability of the fatigue state as possible while observing the input $(F, R, r)$-triplet. We found that we can train on a single $(F, R, r)$-triplet but test on a variety of combinations (Fig. 1 and 11) if fast enough gradual loss of strength due to fatigue can be observed during training time, as well as some form of increase in strength during recovery periods.
6.2 Fatigue Movements and Recovery Strategies

The loss of strength modeled by our method leads to new movement patterns for the rest recovery strategies, as well as the divergence from the input motion the model was trained on. We found the following behaviours, which compensate for the loss of strength: Waiting by standing or doing a couple of steps as observed in the cartwheel (Fig. 5), tornado kick (Fig. 1) and backflip (Fig. 11). Fig. 1, 4, 5 and 11 show how the agent regains strength during such rest periods; a change of performance, e.g. decrease of height of jumps, especially observable in the hopping and tornado kick motions (Fig. 12); a change of motion style, e.g. with increased tucking and knee-bending behaviours in dynamic motions, such as backflip (Fig. 10); and compensation of forces with movements requiring a lot of momentum such as the cartwheel or backflip by trembling or requiring more suspension (Fig. 9). Additionally, a reduction of number of repetitions (Fig. 13) and speed (Fig. 14) can be observed. Fig. 4 shows a comparison between the original 3CC model resulting from the cartwheel motion (left), as well as our modification for animation (right). Note how the fatigue $M_F$ decreases during the rest period between 6 and 19 seconds (left), and increases with each cartwheel. A cartwheel is indicated by the three spikes in $T_L$ and $M_A$ in the beginning, as well as the spike at 20 seconds – in correspondence to the 4 cartwheels in Fig. 5. For animation, we make use of the residual capacity $RC = 100\% - M_F$ as a strength multiplier for the constant torque bounds. The applied torques (dashed blue line) become similar to $M_A$ when being cut off during fatigue and stay equal to $T_L$ during non-fatiguing periods.

360 Tornado Kick. During fatigue the character learns to kick with lower foot height and a reduced distance between the legs, which can be best observed in Fig. 1 and Fig. 12. As the motion requires a lot of strength and flexibility in the leg region, the agent learns to recover by lowering the jump, as well as waiting between jumps. The most affected joints by fatigue are the kicking knee, as well as the arm joints required to gain the momentum for the jump (Fig. 7).

Backflip. The backflips become lower when fatigued with increase in tucking (Fig. 10) and compensation of forces (Fig. 9 bottom). After rest, the agent is able to do backflips with better forms again but cannot do as many anymore before it needs to rest again (Fig. 11).

Cartwheel. We find that the character learns to stand or walk in order to rest from the fatigue after a repetition of cartwheels (Fig. 5). Furthermore, with each repetition, before the character is able to rest fully, the leg height becomes lower.

Hopping. The most apparent changes in the hopping motion during fatigue are the jump height/length and frequency (Fig. 12 top).

Locomotion. The most apparent change over time is the reduction of speed as well as the change of stride length (Fig. 14).

6.3 Learning Diverse Fitness Levels in One Policy

With the fatigue model driving the behaviour of the policy via the fluctuation of $M_F$, the policy is capable of handling a variety of fatigue states it encounters during deployment. Here, we highlight a key advantage of using the 3CC fatigue model by demonstrating the capability to model different fitness levels using the same character and simply manipulating the parameters of the 3CC model at deployment. More specifically, the policy outputs a full spectrum from a high-stamina to a low-stamina character behaviours as a response to intuitive parameter adjustments at runtime. Figures 1, 8 and 11 juxtapose three different scenarios rendered from deploying the same policy in the same initial state: a non-fatigable character for qualitative baseline (top), a high-stamina character (middle), and a low-stamina character. We further emphasize how varying the parameters over time can be used for the 3CC for more fine-grained control. Simply put, we achieve the capability to capture the diversity of fitness levels with the same reference motion, character specification, and policy, which is similar to a mixture of experts policy where experts corresponding to different fitness levels emerge depending on the $(F, R)$ input without any need for fatigued reference motions. Additionally, we show that our method can also be used to analyze which joints are being most affected and to what degree by fatigue (Fig. 7).

6.4 Ablations and Comparisons

We ablate our method with and without the torque coefficient $\beta$ (Fig. 6) showing that it improves upon motion smoothness when limiting torques. Not observing fatigue leads additionally to low fidelity motions as can be seen in the supplementary video. To validate our method we show that our model is able to switch from a running clip to a walking clip without motion blending solely based on the fatigue (supp. video). We further compare our method against two baselines: 1) Against a GAIL-baseline based on AMP [Peng et al. 2021]; 2) Against the reward-based fatigue model by Cheema et al. [2020]. For the former we use the implementation in Isaac Gym [Makoviychuk et al. 2021], whereas for the latter we fine-tune our pre-trained model with their fatigue-based reward without any torque limitation.

GAIL without Fatigue Control. We observe in Tab. 1 that our model is able to synthesize novel fatigued behaviours and recovery strategies not present in dataset, whereas Peng et al. [2022, 2021] are solely able to imitate existing motion capture disregarding any variance over time due to cumulative fatigue (see also supplemental video). In contrast to that our method allows for interactive control of fatigue and varying degrees of fitness levels over time by changing $(F, R)$ values during inference (see Fig. 1). Previous methods [Peng et al. 2022, 2021] are not able to provide such interactive control and varying degrees of fitness levels in one policy.

Fatigue-based Reward. The closest method [Cheema et al. 2020] to our’s uses a reward based on the difference between $M_A$ and $M_F$.
TL. To compare our torque-limit-based method to their reward-based method, we use our reward for fatigue fine-tuning of our GAIL-baseline. The reward is added to $r_f$ in Eq. (8). The results can be observed in Fig. 15. We show that despite their method being based on the 3CC model and resulting in accurate fatigue analysis, it is not able to synthesize correct fatigued behaviours over time, especially when combined with a GAIL-based reward. Fig. 15 (forth subplot from the top) shows that while $RC$ is high, the agent does a lot of fatigued kicks with lower heights. These however become increasingly higher as the $RC$ lowers, since then the agent obtains most of its reward from the imitation reward. The agent furthermore then seems to exploit the imitation learning policy by doing higher jumps than that are in the dataset since the agent can “rest” during the fall of the increased air time. Using the opposite ported from the BMWi grant KAI (19A21022C). Nam Hee Kim was supported from the EIC Pathfinder grant ACKNOWLEDGMENTS character animation, particularly in terms of discovering automatic pathways for biomechanical simulations and physics-based target goals or orientations. We believe this work opens many ex-

7 CONCLUSION We presented the first full-body approach for physics-based 3D human motion synthesis with fatigue. Our experiments demonstrate the emergence of realistic fatigued movements and recovery behaviours for interactive athletic animations that are difficult to produce with previous techniques (especially without musculoskeletal simulation). Our torque-based fatigue simulation system can be efficiently used for real-time interactive 3D virtual character animation, opening up new possibilities for games and animation tools. Additionally, we show in the supplementary material that due to the morphology-agnostic nature of the 3CC-model our method can be applied to characters of any physiology, as long as a measure of $\%MVC$ as a function of either torque or force can be provided. It can further be easily extended with additional task-rewards such as target goals or orientations. We believe this work opens many exciting pathways for biomechanical simulations and physics-based character animation, particularly in terms of discovering automatic emergent behaviours for intelligent agents.

Laura Morales, Seth Izen and Tobias Cheema for their additional support with the supplementary video and figures.

N. Cheema et al.

Noshaba Cheema was supported from the EIC Pathfinder grant Carousel+ (101017779) and an IMPRS Fellowship. Rui Xu was supported from the BMWi grant KAI (19A21022C). Nam Hee Kim received support from the FCAI Open Application Grant and an NSERC Postgraduate Scholarship (567794-2022). The authors from MPII were supported by the ERC Consolidator Grant 4DReLy (770784). We thank Erik Herrmann and Han Du for their retar-
gaining framework and for providing invaluable feedback, along-side Jaakko Lehtinen. We also wish to express our appreciation to Ahmed ‘The Music Jnn’ Hadrich for the sound design, as well as to Laura Morales, Seth Izen and Tobias Cheema for their additional support with the supplementary video and figures.


Figure 3: Pre-trained policy with fatigued torque limits during inference without transfer learning results in task failure, since the policy has not learned to deal with using lower torques and forces.

Figure 4: Results of the 3CC model (left) and its adaptation (right) for the cartwheel. The graphs correspond to the cartwheel motion depicted in Fig. 5. The gray areas indicate a successful cartwheel, whereas the areas between 6s and 19s, as well as after 20s correspond to emerging waiting behaviors to rest from the tiring actions.

Figure 5: Recovery behaviour of a cartwheel motion. After three cartwheels the agent becomes tired, rests, and does another cartwheel when enough stamina is regained.

Figure 6: Ablation without stiffness and damping multiplier (top) and with (bottom, ours) for the shoulder joint during the backflip motion. While the agent is standing and supposed to rest, the shoulder starts to jitter without our torque coefficient $\beta$, while with it the motion is smooth and still (bottom).

Figure 7: Most affected joints from fatigue. From left to right: Cartwheel, Locomotion, 360 Tornado Kick, Hopping, and Backflip.

Figure 8: Movement patterns (top) and $\%$MVC (bottom) for a prominent DoF. The models show the following $(F, R, r)$ settings (from top to bottom): Tornado Kick – $(0, 0, 0), (2.0, 0.05, 1.0), (1.0, 0.01, 1.0)$; Cartwheel – $(0, 0, 0), (1.0, 0.01, 1.0), (3.0, 0.02, 1.0)$ The graphs at the bottom show the results for our modification of the 3CC model for animation.

Figure 9: Compensation of forces. The character compensates forces for movements requiring a lot of momentum such as the cartwheel or backflip by trembling or requiring more suspension.

Figure 10: Change of motion style. Left: Non-fatigued backflip; Right: Fatigued backflip. The height of the fatigued backflip is lower and the body is less stretched compared to the non-fatigued backflip.
Figure 11: Backflip at different (F, R, r)-settings during inference for our model. Top: Non-fatigued backflip at (0, 0, 0). Middle: Moderately fatigued backflips at (1.0, 0.06, 1.0). Bottom: High fatigued backflips at (2.0, 0.07, 1.0).

Figure 12: Change of motion performance. Hopping (top): The distance between each hop decreases with the accumulation of fatigue. Tornado kick (bottom): The range of the kick and the height of the jump both decrease with the accumulation of fatigue. The character then stands and waits for a significantly long time to recover before performing another kick.

Figure 13: Reduction of number of repetitions. The number of repetitions indicated by the number of loops drops between fatigued and non-fatigued motions during the same amount of time. From top to bottom: non-fatigued backflips, fatigued backflips, non-fatigued cartwheels, fatigued cartwheels.

Figure 14: Change of speed. The character becomes slower as it fatigues.

Figure 15: Reward-based fatigue transfer learning [Cheema et al. 2020]. The grey areas indicate a successful tornado kick. From top to bottom: %MVC for the foot height for the fatigued training setting (1, 0.01, 1) for our method, as well as the corresponding movement pattern. %MVC for Cheema et al. [2020], as well as the corresponding movement pattern and the fatigue reward (bottom). Note how the movement patterns and applied torque levels do not correctly correspond to the level of fatigue.
We conduct a goal-oriented simulation task in which a character moves toward a goal while experiencing growing fatigue. To accomplish this, we employed two distinct environments: 1) The humanoid character utilizing the GAIL-based pre-trained expert, 2) a four-legged spider, specifically the Isaac Gym ‘Ant’ with 8 degrees of freedom (DOFs), representing a character with a distinct morphology that is challenging to replicate through motion capture.

**GAIL-Humanoid.** We expand upon the state space $s_t$ outlined in Section 5 of the main document by incorporating a 2D-direction vector representing the character’s intended heading. The reward function in Eq. 8 is then augmented by adding the following heading reward:

$$r_{dir} = \left( \frac{\Gamma_t}{\|\Gamma_t\|} \cdot \hat{\Gamma} \right) + 1 \cdot 0.5$$

With $\Gamma_t$ describing the current local root x,y-orientation and $\hat{\Gamma}$ the target direction. $\hat{\Gamma}$ is randomized at every environment reset. During inference we set $\hat{\Gamma}$ every 10s, which can be seen in Fig.16.

**Four-legged Spider.** For this experiment we extend the goal-oriented IsaacGym ‘Ant’-environment with the fatigue-based torque-limits detailed in Section 4.2, and the addition of $M_F$ into state space $s_t$. Given that we are dealing with a fictional character, we presume a maximum torque limit of 100 Nm for each DOF. The goal of the Ant is to walk towards a specific target. Cumulative fatigue leads to movements that look more natural for the Ant, where it walks slowly on all fours. This can be observed in last two sub-figures in the second row of Fig.17. The original implementation leads instead to speeding and jumping on its hind legs. The results can be better viewed in the supplementary video. In Fig.18 we further show the average speed difference between the two across 64 environments.

### A GOAL-ORIENTED TASKS

While our method for the first time demonstrates plausible fatigue behaviours and emerging resting strategies in the context of full-body character animation, it still exhibits some limitations. Our proposed method is based on the assumption that perceived fatigue can be directly deduced from biomechanical information. However, in reality perceived fatigue can be attributed to a multitude of various factors, such as physiologic and psychological changes or environmental factors, with fatigue and rest being perceived differently from different individuals [Borg 1982; Frey-Law et al. 2012a; Hincapié-Ramos et al. 2014; Jang et al. 2017; Xia and Frey Law 2008]. In addition, the 3CC model has primarily been only validated on simple isometric tasks. Furthermore, while our method is able to generate unseen motions, they may still result in motions of lower quality compared to methods that completely rely on imitation learning, since we try to generate motions outside of the training distribution. One could improve this by including additional methods to explore the action space further, such as intrinsic motivation or curiosity [Pathak et al. 2017; Yin et al. 2021]. For more naturalness, one could also simulate the motor units $MUs$ in the 3CC model as muscle tendons. However, a key-advantage of the 3CC model for animation is that the $MUs$ can be used as a percentage of maximum voluntary contraction defined as percentage of force or torque, which has also been shown to produce accurate fatigue measures in bio-mechanics literature [Frey Law and Avin 2010; Frey-Law et al. 2012a]. Beyond the general accuracy of fatigue modeling, future directions could include object- or agent-agent interactions [Bae et al. 2023; Hassan et al. 2023; Zhang et al. 2023], as well as the general exploring of motion modeling outside data-distributions similar to Lee et al. [2021a].

Despite these limitations, we believe this work will help pave the way towards developing widely reusable control models for physics-based character animation and biomechanical simulations. As such, our work is of importance for biomechanics and animation researchers and practitioners alike, as it can be employed in various applications such as ergonomics analysis and physical skill training in VR/AR environments, as well as fatigue and stamina animation. In this regard, Digital Human Modeling (DHM) [Demirel and Duffy 2007; Maurya et al. 2019] has been widely used in industrial simulation of workers to perform tasks. In order to achieve biomechanically correct ergonomics assessment, it is necessary...
Figure 19: **Left:** Ragdoll simulation model. **Right:** Re-targeted visualization model. Our framework is used to learn fatigued movements for a 28 degrees-of-freedom humanoid character. Pink tints indicate fatigued areas (left). The color of the rigid body corresponds to the fatigue of its parent joint (e.g. lower leg corresponds to the knee). The health bar indicates the current average residual capacity (right).

Figure 20: UI for fatigue modeling. Fatigue synthesis and analysis application based on our method with sliders for setting various \((F, R, r)\)-combinations during inference, and indication of fatigue levels for each joint.

to include an accurate fatigue model of simulated workers. In the scenario of virtual physical skill training, such as virtual sports training, an accurate fatigue model can prevent the users from performing unhealthy movements or getting hurt [Fieraru et al. 2021]. Moreover, our interactive fatigue and rest controls make it possible to easily adapt our method to different fatigue settings during inference in real-time, allowing our approach to be employed in interactive applications such as games or animation tools as can be seen in Fig. 20 in this document and Fig. 1 in the main document. The morphology-agnostic nature of the 3CC-model further allows for the extension to characters of different physiology, and easy extension towards goal-oriented learning.

### C ADDITIONAL IMPLEMENTATION DETAILS

For the benefit of the broader reinforcement learning and movement synthesis community, we identify and outline implementation issues and our workarounds used to address them. To leverage the massive GPU parallelization offered by NVIDIA Isaac simulator, we elected to use the carefully tuned default simulation parameters for the HumanoidAMP environment. This entails the relatively sparse simulation frequency at 120 Hz and policy query frequency at 30 Hz while using Isaac’s native position control mode, which implements stable PD control into the inner loop of the Featherstone articulation program. The inner loop remains inaccessible as Isaac is at a closed-source preview stage as of this writing.

Given the limited integration between the constraint solver and the simulator, we were unable to directly obtain accurate actuation readings from the simulator, specifically the torque values in Nm.

Figure 21: Overview of the Three-Compartment Controller (3CC) model. \(M_A\), \(M_R\) and \(M_F\) denote the percentage of active, rested and fatigued MUs, respectively. \(F\) and \(R\) denote the fatigue and recovery coefficients, and \(C(t)\) the muscle activation-deactivation drive.

Figure 22: The behaviour of the 3CC model at 50% MVC up until 26s with different fatigue \(F\), rest \(R\) and rest recovery \(r\) rates. Note how the target load \(T_L\) cannot be held any longer due to fatigue at \(\sim 3s\) for \(F = 1\) (top two figures), and \(\sim 12s\) for \(F = 0.1\) (bottom two figures) as \(M_A\) decays and is less than \(T_L\). A change in \(F\) denotes the rate of fatigue. \(R\) denotes the limit of fatigued MUs \(M_F\): \(R = F \cdot 0.2\) for example, indicates that 20% MVC can be held indefinitely (top right and bottom two figures), as well as overall change of rest. A change in \(r\) indicates faster recovery during rest periods, i.e. when \(T_L = 0\).
Algorithm 1 Residual Capacity (RC) per DoF

1: $F$ ← set the fatigue rate
2: $R$ ← set the recovery rate
3: $r$ ← set the rest recovery multiplier
4: $R_p$ ← $R$ set the rest recovery rate
5: $L_D$ ← set the muscle force development factor
6: $L_R$ ← set the muscle force relaxation factor
7: $Δt$ ← interval time between two consecutive simulation steps
8: $T_{max}$ ← set the maximum torque
9: function 3CC($T$)
10: if at the beginning of an episode then
11: if training then
12: $M_R$ ← $Uniform.sample(0, 100)$
13: $M_A$ ← $Uniform.sample(0, 100 - M_R)$
14: $M_F$ ← $100 - M_R - M_A$
15: else
16: $M_R$ ← $100$
17: $M_A$ ← $0$
18: $M_F$ ← $0$
19: end if
20: end if
21: $\tilde{T}$ ← $clip(T, -T_{max}, T_{max})$ ➞ Eq. (11)
22: $TL$ ← $\frac{100}{M_A}$
23: if $M_A < TL$ and $M_R > TL - M_A$ then
24: $C_t$ ← $L_D$ · $(TL - M_A)$ ➞ Eq. (3)
25: end if
26: if $M_A < TL$ and $M_R \leq TL - M_A$ then
27: $C_t$ ← $L_D$ · $M_R$ ➞ Eq. (3)
28: end if
29: if $M_A ≥ TL$ then
30: $C_t$ ← $L_D$ · $(M_A)$ ➞ Eq. (3)
31: $R_p$ ← $r$ · $R$
32: end if
33: $M_R$ ← $M_A + Δt$ · $(C_t - F · M_A)$ ➞ Eq. (1a)
34: $M_R$ ← $M_R + Δt$ · $(C_t - R_p · M_F)$ ➞ Eq. (1b)
35: $M_F$ ← $M_F + Δt$ · $(F · M_A - R_p · M_F)$ ➞ Eq. (1c)
36: $RC$ ← $(100 - M_F)/100$ ➞ Eq. (4)
37: return $RC$
38: end function

Algorithm 2 Fatigue Transfer Learning

1: input $M$ : reference motion
2: input $T_{max}$: maximum torque
3: $D$ ← initialize discriminator with pre-trained model
4: $π$ ← initialize policy with pre-trained model
5: $V$ ← initialize value function with pre-trained model
6: $B$ ← $∅$ initialize replay buffer
7: while not done do
8: for trajectory $i = 1 \ldots m$ do
9: $t^i$ ← $0$ initialize trajectory
10: $s_0$ ← sample start state from pre-defined distribution
11: for time step $[t = 0, \ldots , T - 1]$ do
12: $a_t$ ← sample action (target joint position) from $π$
13: $T ← PDControl(a_t)$ ➞ Eq. (9)
14: for each DoF $d$ do
15: $R_c[d] ← 3CC(T_d)$ (see Alg. 1)
16: $\tilde{T}_{max,d} ← R_c[d] · T_{max}$ ➞ Eq. (12)
17: $\tilde{T}_d ← clip(T_d, -\tilde{T}_{max,d}, \tilde{T}_{max})$ ➞ Eq. (13)
18: end for
19: $s_{t+1} ←$ get next state by applying $\tilde{T}$
20: $r_t ← -log(1 - D(\hat{Φ}(s_t), \hat{Φ}(s_{t+1})))$ ➞ Eq. (8)
21: record $r_t$ in $t^i$
22: end for
23: store $t^i$ in $B$
24: end for
25: update $π$, $V$ and $D$ using data from $B$ and $M$
26: end while

Table 2: Custom stiffness and damping parameters used for the humanoid character.

<table>
<thead>
<tr>
<th>Joint</th>
<th>Stiffness</th>
<th>Damping</th>
</tr>
</thead>
<tbody>
<tr>
<td>Abdomen</td>
<td>120</td>
<td>12</td>
</tr>
<tr>
<td>Neck</td>
<td>90</td>
<td>9</td>
</tr>
<tr>
<td>Shoulders</td>
<td>100</td>
<td>10</td>
</tr>
<tr>
<td>Elbows</td>
<td>110</td>
<td>11</td>
</tr>
<tr>
<td>Hips</td>
<td>320</td>
<td>32</td>
</tr>
<tr>
<td>Knees</td>
<td>370</td>
<td>37</td>
</tr>
<tr>
<td>Ankles</td>
<td>120</td>
<td>12</td>
</tr>
</tbody>
</table>

stiffness and damping parameters modulated as appropriate for each controlled degree of freedom. We are confident that this approximation is reasonable, as the 3CC model used in our research is based on ratios instead of absolute units or scales. For the $k_p$ and $k_d$ parameters at each controlled DoF, we used stiffness and damping parameters modulated as appropriate. We deem this approximation reasonable as the 3CC model is based on ratios instead of absolute units or scales. Recall that the modulated stiffness and damping parameter values are written as $\tilde{k}_{p,t} = \hat{β}_t k_p$ and $\tilde{k}_{d,t} = \hat{β}_t k_d$, respectively. As such, the intended torque $\tilde{T}$ and the clipped torque $\tilde{T}_{max}$ were estimated as

$$\tilde{T} = \hat{β}_p \cdot (\hat{θ}_t - \hat{θ}_f) - \hat{k}_{d,t} \cdot \hat{θ}_f$$

where $\hat{θ}_f$ is the PD target angles corresponding to the policy action $a_t$, $\hat{θ}_t$ is the current DoF angles, and $\hat{θ}_f$ is the current DoF angular velocities, all at timestep $t$. To implement the torque limiting described in the main paper, in case the estimated intended torque $\tilde{T}$ is above the effective ceiling torque $\tilde{T}_{max}$, we compute the ratio between them, which is then multiplied to the intended torque to implement the clipping, i.e.,

$$\tilde{T} = \begin{cases} \tilde{T} \cdot \frac{\tilde{T}_{max}}{\tilde{T}} & \text{otherwise.} \end{cases}$$

We set the simulator’s joint stiffness and damping values to $\tilde{k}_{p,t}$ and $\tilde{k}_{d,t}$, respectively.
Table 3 reports additional hyperparameters for training our approach such as learning rate and batch size. Table 4 reports the found maximum torque bounds. We report bounds for each action and joint. Note that those bounds are found automatically during our proposed pre-training.

Table 3: Additional PPO-Hyperparameter Settings.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Horizon Length</td>
<td>16</td>
</tr>
<tr>
<td>Minibatch Size</td>
<td>4096</td>
</tr>
<tr>
<td>Batch Size</td>
<td>512</td>
</tr>
<tr>
<td>KL Coeff.</td>
<td>0.008</td>
</tr>
<tr>
<td>γ Discount</td>
<td>0.99</td>
</tr>
<tr>
<td>GAE(γ)</td>
<td>0.95</td>
</tr>
<tr>
<td>TD(γ)</td>
<td>0.95</td>
</tr>
<tr>
<td>Adam stepsize</td>
<td>$5 \times 10^{-5}$</td>
</tr>
</tbody>
</table>

Table 4: Found maximum constant torque bounds for each task during pre-training. Note that $z$ denotes the up-axis. Symmetric joints present on the left and right side of the body use the same values. Bold values highlight the maximum value across motions, hence the denominators of $TL$ value calculation.

<table>
<thead>
<tr>
<th>DoF</th>
<th>Run</th>
<th>Hop</th>
<th>Cartwheel</th>
<th>Backflip</th>
<th>Kick</th>
</tr>
</thead>
<tbody>
<tr>
<td>Abdomen$_x$</td>
<td>45</td>
<td>67</td>
<td>163</td>
<td>370.27</td>
<td>290</td>
</tr>
<tr>
<td>Abdomen$_y$</td>
<td>37</td>
<td>197</td>
<td>234</td>
<td>382.34</td>
<td>181</td>
</tr>
<tr>
<td>Abdomen$_z$</td>
<td>57</td>
<td>98</td>
<td>155</td>
<td>240.88</td>
<td>234</td>
</tr>
<tr>
<td>Neck$_x$</td>
<td>14</td>
<td>26</td>
<td>59</td>
<td>21</td>
<td>78.61</td>
</tr>
<tr>
<td>Neck$_y$</td>
<td>27</td>
<td>160.34</td>
<td>133</td>
<td>77</td>
<td>69</td>
</tr>
<tr>
<td>Neck$_z$</td>
<td>15</td>
<td>16</td>
<td>62</td>
<td>63.66</td>
<td>47</td>
</tr>
<tr>
<td>Shoulder$_x$</td>
<td>20</td>
<td>64</td>
<td>149</td>
<td>177.67</td>
<td>143</td>
</tr>
<tr>
<td>Shoulder$_y$</td>
<td>35</td>
<td>98</td>
<td>90</td>
<td>362.03</td>
<td>171</td>
</tr>
<tr>
<td>Shoulder$_z$</td>
<td>13</td>
<td>56</td>
<td>116</td>
<td>306.93</td>
<td>173</td>
</tr>
<tr>
<td>Elbow</td>
<td>22</td>
<td>107</td>
<td>80</td>
<td>91</td>
<td>136.1</td>
</tr>
<tr>
<td>Hip$_x$</td>
<td>150</td>
<td>394</td>
<td>580</td>
<td>143</td>
<td>598.84</td>
</tr>
<tr>
<td>Hip$_y$</td>
<td>203</td>
<td>421</td>
<td>796.42</td>
<td>388</td>
<td>691</td>
</tr>
<tr>
<td>Hip$_z$</td>
<td>79</td>
<td>101</td>
<td>247</td>
<td>135</td>
<td>373.14</td>
</tr>
<tr>
<td>Knee</td>
<td>421</td>
<td>741</td>
<td>496</td>
<td>340</td>
<td>809.59</td>
</tr>
<tr>
<td>Ankle$_x$</td>
<td>23</td>
<td>92</td>
<td>99</td>
<td>73</td>
<td>138.58</td>
</tr>
<tr>
<td>Ankle$_y$</td>
<td>228</td>
<td>207</td>
<td>120</td>
<td>407</td>
<td>451.8</td>
</tr>
<tr>
<td>Ankle$_z$</td>
<td>38</td>
<td>70</td>
<td>115</td>
<td>90</td>
<td>201.1</td>
</tr>
</tbody>
</table>