Reinforcement Learning
for
Muscle-Driven Systems

Dissertation

der Mathematisch-Naturwissenschaftlichen Fakultät

der Eberhard Karls Universität Tübingen

zur Erlangung des Grades eines

Doktors der Naturwissenschaften

(Dr. rer. nat.)

vorgelegt von

Pierre Schumacher

aus Luxemburg/Luxemburg

Tübingen

2025

Gedruckt mit Genehmigung der Mathematisch-Naturwissenschaftlichen Fakultät der Eberhard Karls Universität Tübingen.

Tag der mündlichen Qualifikation:	21.03.2025
Dekan:	Prof. Dr. Thilo Stehle
1. Berichterstatter/-in:	Prof. Dr. Daniel Häufle
2. Berichterstatter/-in:	Prof. Dr. Martin Giese

"*If I had asked people what they wanted, they would have said faster horses.*"

Henry Ford

Abstract

The motor skills performed by human beings every day are still unmatched by artificial systems, despite the rapid advances of control and motion generation methods driven by deep learning. The fields of biomechanics and computational motor control investigate these capabilities with a variety of different methods, among which predictive simulation has been a successful paradigm, allowing researchers to gather insight into human motor control as well as develop new methods for rehabilitation. While existing control methods achieve impressive results by using simplified models or by neglecting aspects such as feedback, environment uncertainty and biologically plausible sensory inputs, a scalable control algorithm capable of generating closed-loop policies for high-dimensional musculoskeletal systems under a variety of conditions is still missing.

One likely candidate to produce such a controller is Reinforcement Learning (RL). RL has been able to create robust controllers in the robot learning domain, performing diverse behaviors in simulation and the real-world by leveraging advances in algorithms and simulation engines. Nevertheless, the high dimensionality and complex dynamics of musculoskeletal systems have limited its adoption in biomechanics. From the perspective of bio-inspired robotics, although biological muscles exhibit properties that likely enhance movement control, it remains unclear which features are essential and which are only evolutionary artifacts, detrimental to movement generation. This dissertation addresses these challenges through novel algorithmic developments and computational frameworks that enable scalable RL for complex biomechanical systems, while also investigating muscular actuation properties in robotic systems across both simulation and real-world settings.

In two computational studies, we could first develop a RL control algorithm capable of generating behaviors for high-dimensional musculoskeletal systems, before extending it with bio-inspired reward terms, making the resulting controller act close-to-natural, while being incredibly robust. We found that overactuation is a key problem in musculoskeletal systems, which exarcerbates exploration issues in RL. By leveraging a technique from the domain of self-organizing systems, we could propose a controller capable of generating effective exploration for arbitrary muscle-driven systems, while performing well in a wide range of environments. Our proposed reward terms for natural walking are adaptive and produce close-to-natural motion in 4 different models without the need for parameter tuning. Additionally, all proposed methods have been shown to work across different simulators of varying computational speed and biomechanical fidelity, showcasing the versatility of our approach.

In another study, we investigated the contributions of muscles to control in a robotic setting. By emulating a simplified muscle actuator in simulation across a wide range of tasks and models, we could investigate the influence of bio-inspired properties on control aspects such as robustness and data efficiency. Our investigations reveal that while idealized torque actuators can achieve optimal performance under precise conditions, muscle-like properties offer superior robustness and learning efficiency across varied tasks and environments. By not relying on prescribed trajectories, but only abstract reward terms, we gave the optimization enough freedom such that the embodiment could have a measurable effect on the learned behaviors. In an extension

to this result, we applied a similar muscle actuator to a simulated quadruped robot. It was observed that the muscular actuation could, in addition to producing more robust behaviors, also bias the learned behaviors towards being more natural and smooth. By building an experimental pipeline where a muscle actuator, identical to the simulated one, could be emulated on real robotic hardware, we were able to perform sim-to-real transfer with a learned RL policy and achieve walking with a real quadruped robot. After performing a parameter optimization of the actuator where feasibility on the hardware was ensured through a newly derived damping rule, it could be seen that the bias induced through the muscle actuator led to behaviors transferring better from simulation to the real system.

Our results show that RL is a viable candidate to create closed-loop controllers that produce robust and close-to-natural behaviors in high-dimensional musculoskeletal systems under a variety of conditions. We believe his to be a first step to enable new advances in biomechanics, motor control and rehabilitation. This thesis also highlights the potential benefits of muscle actuation for robotic systems and the use of emulated actuators to enable the simultaneous optimization of robot morphology and control architecture on real hardware.

Zusammenfassung

Die motorischen Fähigkeiten, die Menschen jeden Tag ausüben, werden von künstlichen Systemen immer noch nicht erreicht, trotz der rasanten Fortschritte bei Steuerungs- und Bewegungserzeugungsmethoden, die durch Deep Learning vorangetrieben werden. Die Bereiche Biomechanik und computergestützte Motorsteuerung untersuchen diese Fähigkeiten mit einer Vielzahl verschiedener Methoden, darunter die prädiktive Simulation als erfolgreiches Paradigma, die es Forschern ermöglicht, Einblicke in die menschliche Motorsteuerung zu gewinnen und neue Methoden für die Rehabilitation zu entwickeln. Während bestehende Kontrollmethoden beeindruckende Ergebnisse erzielen, indem sie vereinfachte Modelle verwenden oder Aspekte wie Feedback, Umgebungsunsicherheit und biologisch plausible sensorische Eingaben vernachlässigen, fehlt immer noch ein skalierbarer Kontrollalgorithmus, der in der Lage ist, Regelkreise für hochdimensionale Muskel-Skelett-Systeme unter verschiedenen Bedingungen zu generieren.

Ein wahrscheinlicher Kandidat für die Entwicklung eines solchen Controllers ist Reinforcement Learning (RL). RL war in der Lage, robuste Steuerungen im Bereich des Roboterlernens zu entwickeln, die durch die Nutzung von Fortschritten bei Algorithmen und Simulationsmaschinen unterschiedliche Verhaltensweisen in der Simulation und in der realen Welt ausführen. Dennoch haben die hohe Dimensionalität und die komplexe Dynamik des Muskel-Skelett-Systems seine Anwendung in der Biomechanik eingeschränkt. Aus Sicht der bioinspirierten Robotik bleibt unklar, welche Merkmale wesentlich sind und welche nur evolutionäre Artefakte sind, die der Bewegungserzeugung abträglich sind, obwohl biologische Muskeln Eigenschaften aufweisen, die wahrscheinlich die Bewegungskontrolle verbessern. Diese Dissertation befasst sich mit diesen Herausforderungen durch neuartige algorithmische Entwicklungen und Rechenrahmen, die skalierbares RL für komplexe biomechanische Systeme ermöglichen und gleichzeitig Muskelbetätigungseigenschaften in Robotersystemen sowohl in Simulationen als auch in realen Umgebungen untersuchen.

In zwei Computerstudien konnten wir zunächst einen RL-Kontrollalgorithmus entwickeln, der in der Lage ist, Verhaltensweisen für hochdimensionale Muskel-Skelett-Systeme zu erzeugen, und ihn dann um bioinspirierte Belohnungsbedingungen erweitern, sodass der resultierende Controller nahezu natürlich agiert und gleichzeitig unglaublich robust ist. Wir haben herausgefunden, dass Überbetätigung ein zentrales Problem im Bewegungsapparat ist, was die Explorationsprobleme bei RL verschärft. Durch die Nutzung einer Technik aus dem Bereich selbstorganisierender Systeme könnten wir einen Controller vorschlagen, der in der Lage ist, eine effektive Erkundung für beliebige muskelbetriebene Systeme zu generieren und gleichzeitig in einer Vielzahl von Umgebungen gute Leistungen zu erbringen. Unsere vorgeschlagenen Belohnungsbedingungen für natürliches Gehen sind adaptiv und erzeugen eine nahezu natürliche Bewegung in 4 verschiedenen Modellen, ohne dass eine Parameteranpassung erforderlich ist. Darüber hinaus wurde gezeigt, dass alle vorgeschlagenen Methoden auf verschiedenen Simulatoren mit unterschiedlicher Rechengeschwindigkeit und biomechanischer Genauigkeit funktionieren, was die Vielseitigkeit unseres Ansatzes unterstreicht.

In einer anderen Studie untersuchten wir den Beitrag der Muskeln zur Kontrolle in

einer Roboterumgebung. Durch die Simulation eines vereinfachten Muskelaktors in einer Vielzahl von Aufgaben und Modellen konnten wir den Einfluss bioinspirierter Eigenschaften auf Kontrollaspekte wie Robustheit und Dateneffizienz untersuchen. Unsere Untersuchungen zeigen, dass idealisierte Drehmomentaktoren zwar unter präzisen Bedingungen eine optimale Leistung erzielen können, muskelähnliche Eigenschaften jedoch eine überlegene Robustheit und Lerneffizienz bei verschiedenen Aufgaben und Umgebungen bieten. Indem wir uns nicht auf vorgeschriebene Trajektorien, sondern nur auf abstrakte Belohnungsterme verließen, gaben wir der Optimierung genügend Freiheit, sodass die Verkörperung einen messbaren Effekt auf die erlernten Verhaltensweisen haben konnte. Als Erweiterung dieses Ergebnisses haben wir einen ähnlichen Muskelaktuator auf einen simulierten vierbeinigen Roboter angewendet. Es wurde beobachtet, dass die Muskelbetätigung nicht nur robustere Verhaltensweisen hervorrufen kann, sondern auch dazu führen kann, dass die erlernten Verhaltensweisen natürlicher und geschmeidiger werden. Durch den Aufbau einer experimentellen Pipeline, in der ein Muskelaktuator, der fast identisch mit dem simulierten ist, auf echter Roboterhardware emuliert werden konnte, konnten wir mit einer erlernten RL-Policy einen Sim-zu-Real-Transfer durchführen und das Gehen mit einem echten vierbeinigen Roboter erreichen. Nach der Durchführung einer Parameteroptimierung des Aktuators, bei der die Umsetzbarkeit auf der Hardware durch eine neu abgeleitete Dämpfungsregel sichergestellt wurde, konnte festgestellt werden, dass die durch den Muskelaktuator induzierten Bias dazu führte, dass sich Verhaltensweisen besser von der Simulation auf das reale System übertragen ließen.

Unsere Ergebnisse zeigen, dass RL ein geeigneter Kandidat für die Herstellung von Regelkreisen ist, die unter verschiedenen Bedingungen robuste und naturnahe Verhaltensweisen in hochdimensionalen Muskel-Skelett-Systemen hervorrufen. Wir glauben, dass dies ein erster Schritt ist, um neue Fortschritte in der Biomechanik, Motorsteuerung und Rehabilitation zu ermöglichen. Diese Arbeit beleuchtet auch die potenziellen Vorteile der Muskelbetätigung für Robotersysteme und die Verwendung emulierter Aktuatoren, um die gleichzeitige Optimierung der Robotermorphologie und der Steuerungsarchitektur auf realer Hardware zu ermöglichen.

Acknowledgements

Very often I would tell anyone around me that I had a wonderful PhD experience. I had two amazing research groups, two great advisors, made good progress and got to go to conferences in exotic places. It should be clear to anyone in academia that a nurturing environment and progress go hand in hand: I wouldn't have been able to work hard and be creative without the support of my two amazing research groups. This journey was much harder than I sometimes tried to make it look.

At the AL-group, I was held together by Cansu Sancaktar, Marco Bagatella, Andreas Geist, Nuria Armengol, Albane Ruaud, Pavel Kolev, Andrii Zadaianchuk, Anselm Paulus, Iris Andrussow, Mikel Zhobro, Nico Gürtler and Marin Vlastelica. I deeply appreciated our coffee talks, late night deadline frenzies and ski trips. From the Hertie crowd, I want to thank Matthew Araz, Fabio Izzi, Christian Niethammer, Jhon Charaja and Junya Inoue. You were always there when I needed you the most. Thank you for creating the warmest and kindest group atmosphere I could imagine.

A very special thanks goes to my advisors, Daniel Häufle and Georg Martius. I lack the words to describe how much I owe you. The people that know how much I like to talk should appreciate the strength of this statement. Anytime I had a meeting with Georg and Daniel in one room, I left it feeling clearer and calmer than when I entered, an experience I wish every PhD student could have. I also thank my thesis committee for offering their valuable time to me.

I also thank my co-authors, Dieter Büchler, Jan Schneider, Syn Schmitt, Thomas Geijtenbeek, Vittorio Caggiano, Vikash Kumar, Nadine Badie, Firas Al-Hafez. It wasn't always easy and there was more than one heated discussion, but it was fun, productive and I think everyone grew in the process. A special thank you goes to Isabell Wochner for being the best co-first-author one could wish for.

I want to thank the friends I made in Pittsburgh. Lalitha Ayyappan, Srikanth Ramamurthy, Ayesha Bhatia, Suvarsha Rai, Hamza Amrani and Suman Hazra. You made Pittsburgh an experience I will cherish for the rest of my life.

Lastly, I want to thank my family, my friends and Pirkko. Without you, I don't think I would have been able to finish this journey. Thank you for staying by me in my worst moments.

Contents

List of Figures

List of Abbreviations

RL Reinforcement Learning

DOF Degrees-of-Freedom

MPC Model-Predictive-Control

MSK musculoskeletal

DEP Differential Extrinsic Plasticity

ML Machine Learning

OC Optimal Control

CPG Central Pattern Generator

Chapter 1

Introduction

1.1 Understanding motor control: Why does it matter?

Despite the increasing pace of developments in robotics and Machine Learning (ML), current controllers still struggle to replicate the robustness, diversity, and complexity of human movement. Human beings perform incredible acts of agility and precision under a variety of environmental conditions, constrained by imprecise, delayed, and incomplete sensory inputs, as well as highly non-linear and non-stationary actuation mechanisms. The study of biomechanics is crucial as it provides insights into the fundamental principles governing human movement, which are essential for advancing both theoretical and applied sciences. Research in human motor control has significantly contributed to the development of medical assessments (Laßmann et al., 2022) and treatments, rehabilitation strategies (Rokhmanova et al., 2024), and the design of assistive devices (Scherb, Wartzack, and Miehling, 2023), enhancing the quality of life for individuals with movement impairments. Furthermore, understanding the intricate interplay between neural control and musculoskeletal dynamics can lead to the creation of more sophisticated and adaptive robotic systems (Shafiee, Bellegarda, and Ijspeert, 2024). By leveraging biomechanical principles, researchers can develop controllers that mimic the efficiency and adaptability of human motor control, potentially revolutionizing fields such as prosthetics, sports science, and ergonomics.

Many motor control studies rely on predictive forward simulations—control signals are found either through direct optimization or by finding controllers which drive a physics-based musculoskeletal (MSK) model to perform behaviors which align with realistic human motion. In contrast to inverse tracking in simulation, which synthesizes appropriate control signals to track a given recorded trajectory, predictive simulation allows researchers to explore the potential behavioral impact induced by changes in muscle physiology or neural-control pathways, rehabilitation procedures, or different designs of assistive devices (Falisse et al., 2019; Nikoo and Uchida, 2022). The successful application of forward simulation has been enabled by different types of control algorithms: Direct trajectory optimization, or Optimal Control (OC), is capable of controlling high-dimensional muscle-driven systems, but struggles to incorporate feedback or environmental uncertainties (De Groote and Falisse, 2021). Model-Predictive-Control (MPC) and reflex-based controllers inherently consider closed-loop control, either through direct sensory-motor loops (Song and Geyer, 2013) or through receding horizon predictions (Sahara et al., 2024), but have not been

extended to systems with hundreds of muscles or behaviors that go strongly beyond simple periodic motions. This gap has potentially contributed to the tendency of the community to study restricted motions in laboratory experiments, complicating the use of predictive simulation to investigate realistic behaviors in uncertain conditions in the real world. A flexible closed-loop control method, capable of synthesizing the richness of human behavior under a variety of external conditions while handling high-dimensional MSK-models, is still missing.

Recently, the explosion of technical advances in the fields of machine learning and robotics have enabled the creation of algorithms of an unprecedented complexity, capable of making predictions on extremely high-dimensional systems (Jumper et al., 2021) and creating controllers for robots acting in the real world (He et al., 2024). The use of these techniques not only allows for a potential increase in precision and accuracy of existing methods in rehabilitation and motor control research (Tschugnall et al., 2021; Gozlan et al., 2024; Cotton, 2024), but might enable a wave of progress based on computational advances which will change the field. Policies optimized through Reinforcement Learning (RL) can be deployed on real quadrupedal and bipedal robots, having been trained entirely in simulation, even though the used models are far from accurately predicting the real world. Faster simulators in combination with algorithmic advances allow these controllers to work under unseen conditions, which has transformed the field of robot learning (Rudin et al., 2021; Nasiriany et al., 2024). RL has also been found to produce more robust policies in real-world systems (Kaufmann et al., 2023), to be capable of distilling large amount of experience (Luo et al., 2024b) and to yield flexible real-time controllers (Margolis et al., 2022). Finally, RL is aligned with the way humans learn motor control, as it is a learning mechanism used to drive adaptation in the brain (Botvinick et al., 2020; Vassiliadis et al., 2021).

Despite these achievements, RL methods remain difficult to train: they suffer from convergence issues, need extensive synthetic data from simulators and often require researchers to craft specific exploration techniques for them to perform well. Consequently, the transfer to applications in motor control and biomechanics has been lacking. Popular simulators like OpenSim (Delp et al., 2007) have mainly been developed for the application of inverse kinematics and dynamics, suffering from speed and stability issues in predictive forward dynamics simulation which is essential for RL to perform well. A community effort driven by NeurIPS challenges (Song et al., 2021) has been made to bridge the gap between computational motor control and RL, but the resulting controllers have still been limited to low-dimensional systems and simple periodic behaviors. Even then, training times for these algorithms often exceeded weeks of time, making it infeasible to progress quickly.

This thesis aims to advance RL methods for MSK systems to enable the use of high-dimensional models for the generation of complex human behavior under a wide range of conditions. By making use of computational advances in simulation software and by investigating the root causes for the lack of transfer from robotics to biomechanics, we develop new RL algorithms which can be reliably used to produce policies uniting the strengths of conventional control methods and remaining flexible enough to be applied to different models and tasks. These methods constitute a first step for researchers to study the full richness of human behavior under varied real world conditions in

predictive simulation, potentially enabling advances in rehabilitation and assistive devices as well as unlocking insights into human motor control.

1.2 The musculoskeletal control problem

Formulated in its current form by Bernstein (1967), the motor equivalence problem, or degrees-of-freedom problem, states that there are many ways a human or animal can perform a certain movement. Consider the task of reaching a certain point with your hand. On a kinematic level, there is an infinite number of variations of elbow and shoulder movement which achieve the same final hand position, not to mention the different trajectories that could have been chosen. Additionally, humans and animals possess a redundancy in their actuation mechanisms. In contrast to most humanoid robots, which often have 1 actuator per Degrees-of-Freedom (DOF), muscle-driven systems are strongly overactuated, with around 244 DOF (Morecki, Ekiel, and Fidelus, 1984) and around 630 skeletal muscles (Prilutsky and Zatsiorsky, 2002). This overactuation introduces additional flexibility in task execution, which might have secondary effects on robustness (Berret et al., 2024) and metabolic consumption (Piche et al., 2022), but not on the primary goal.

In addition to this fundamental problem, muscle actuators also have strong non-linear properties, which makes the application of classical control algorithms infeasible. A widely used computational model for the approximation of biological muscle is the Hill-type muscle model. It modulates muscle force through non-linear force-velocity and force-length relationships, includes a muscle activity low-pass filter mimicking the diffusion of calcium ions in biological muscle and contains an elastic tendon, capable of stretching independently of the muscle fiber (Zajac, 1989; Haeufle et al., 2014).

In face of these difficulties, various mechanisms have been proposed. Some studies approximate human behavior with torque control (Jiang et al., 2019), which is infeasible if the exact forces and dynamics of internal muscle components are required. Driess et al. (2018) leverage equilibrium-point control to linearize muscle dynamics at quasi-static points and learn an inverse model for a robotic arm with artificial muscles, sacrificing the capability of learning fast dynamic movements. Assumptions on the control architecture, used in reflex- (Geyer and Herr, 2010; Song and Geyer, 2013) or Central Pattern Generator (CPG)-based (Ryu and Kuo, 2021) controllers, can produce natural behaviors without muscle model simplifications, but are usually restricted to simple models, similar to MPC approaches (Takagi et al., 2022). Finally, trajectory optimization is able to synthesize behaviors for high-dimensional systems without simplifications, but does not easily incorporate feedback or environment uncertainties (Nitschke et al., 2020).

RL has the potential to create flexible and robust feedback controllers for high-dimensional systems without making simplifying assumptions on the muscle models or the control problem. Nevertheless, this potential has not been realized yet. Many studies only used RL with low-dimensional models (Tieck et al., 2018; Weng, Hashemi, and Arami, 2021a), or assumed stiff tendons in order to define a direct correspondence between muscle activity and joint torques. This allowed them to decompose

the problem into a separate joint-space and muscle-excitation optimization (Lee et al., 2019; Luo et al., 2021).

An interesting approach by Denizdurduran, Markram, and Gewaltig (2022) first obtains kinematic trajectories with torque-based MPC, before using RL-based imitation learning to track the proposed trajectories, lifting the need for experimental data. Nevertheless, this technique ignores the possibility of muscle actuation inducing a different optimal behavior space than torque actuation, as the policies are tracking the torque-trajectory with muscle actuators. As our experiments in contributions C and D show, muscles can bias the learned behaviors to a more natural manifold. Finally, muscle synergies can be extracted from human-data to constrain learning to a simpler control space (Al Borno, Hicks, and Delp, 2020). Subsequent learning in the synergy space simplifies the original problem, but the choice of data strongly affects the possible downstream behaviors. So far, most studies applied to MSK-systems have only learned behaviors identical to the original data.

In contribution A, we propose a new control algorithm for MSK systems. We show that the main issue for RL algorithms in this application is the overactuation of the body, induced by the large number of muscles with respect to the DOF. Differential Extrinsic Plasticity (DEP), a method from the field of self-organization, is used to generate embodiment-specific exploration data which improves the RL-agents performance in a wide range of tasks. DEP requires no prior knowledge about the control system or the muscle architecture, as it excites the system to produce joint angle motion. Only the one-to-one mapping between muscle sensor and excitation signal is needed, which is trivial to obtain in a simulator. This combination of DEP with off-policy RL algorithms, allows us to investigate very high-dimensional muscle-driven systems and propose a flexible and performant control strategy based on RL which does not make assumptions on the muscle model.

Nevertheless, the behaviors produced solely with an external task reward often lack certain markers commonly found in natural movement. The muscles are often used to 100% activity, co-activation levels are high and some extremities can be unnaturally contorted. Such artifacts are ofter remedied in the motor control community by introducing specific cost functions, whose associated optimal behavior matches recorded experimental data. Predictive reaching simulations can rely on muscle activity and jerk minimization (Kistemaker, Wong, and Gribble, 2014), while specifying a target location to be reached. Studies focusing on locomotion might use a combination of effort costs with force-based functions and costs mimicking self-preservation incentives in animals and humans (Veerkamp et al., 2021).

Constraining behaviors in this way to a more natural manifold has worked well for OC and reflex-based controllers, but it has been difficult to transfer these insights to RL. Working approaches have only managed success on low-dimensional models and required large computational resources or complex and sensitive cost curricula (Weng, Hashemi, and Arami, 2021a). In contribution B, we propose to use a reward function rooted in self-preservation, control smoothness and effort minimization. Crucially, our approach uses an adaptive effort term, which slowly reduces the used muscle activity of our controller in a flexible way, without the need to retune the reward function for any of the 4 used models. With this contribution, we are able to investigate the

influence of a biologically-inspired cost function, in combination with evolutionary priors embedded as MSK-modelling choices on learned behaviors with RL, while forcing good fulfillment of the auxiliary rewards.

1.3 The role of embodiment in robotics

Robotic systems traditionally rely on rigid components linked by moveable joints actuated by motors. While a certain diversity of actuation mechanisms is present in industry, the field of robot learning has primarily consolidated on electric motors, with other mechanisms, such as elastic, hydraulic or pneumatic actuators (Pratt and Williamson, 1995; Tondu and Zagal, 2006), only being used in niche applications. The low-cost, high performance and somewhat "neutral" actuation properties make electric motors favorable for robot learning, as they are easy to model in simulation and can use their maximum torque output almost over the entire operating range: They have enabled affordable open-source robots to be developed (Grimminger et al., 2020; Ramos et al., 2021). A wide range of studies on robot learning have used such devices successfully (Li et al., 2023; Chane-Sane et al., 2024) to investigate new control algorithms.

Despite these successes, electric motors face many limitations. Control is highly dependent on execution frequency, with RL agents often operating between 50 and 200 Hz (Chen et al., 2023). Any situation requiring faster reaction speed is usually mitigated with PD-controllers tracking the desired position, which is often not precise enough for fine-grained behaviors like object manipulation (Allshire et al., 2022). Additionally, behaviors often need to be shaped through a large number of reward terms tuned by experts, on top of experimental human data which is retargeted to the robot and then tracked (He et al., 2024). In contrast, animals can perform robust and agile locomotion tasks while using sensor and actuation mechanisms with delays that are up to 100x higher than the ones typically found in robots (Gordon et al., 2020). Many studies have started to investigate the benefits of integrating muscle-like properties into robotic hardware, by implementing tuneable springs (Mo et al., 2023) or pneumatic artificial muscles (Büchler et al., 2022). Unfortunately, the design of hardware is a slow process and new actuation architectures require the development of control algorithms capable of handling the overactuation and non-linear properties often found in bio-inspired hardware.

A promising approach to study the benefits of muscle actuation in robot learning, without incurring the difficulties associated with full artificial muscle architectures, is to separate the different components of biological muscle and investigate their effects in simplified simulations. We have, for example, seen a bias towards human-like movement with the right cost function and a high-dimensional MSK-model in the previous section, but it remains unclear *which* components of our biological actuation mechanism are actually beneficial, and which might complicate control. More precise insights about motor control can be gained with *comparative* experiments. By investigating simple and complex muscle models across different levels of abstraction in predictive simulations, researchers can more precisely probe and disentangle the contributions of the musculoskeletal system to movement control and assess which components might be valuable for other applications like bio-inspired robotics.

Considering idealized torque-control as the most "neutral" actuator, very early studies (Soest and Bobbert, 1993) have already shown stabilizing properties of muscle actuation in low-dimensional systems driven by OC. Similar observations have been made by other theoretical (Wagner and Blickhan, 1999; Eriten and Dankowicz, 2009) and computational works (Stollenmaier, Ilg, and Haeufle, 2020; Haeufle, Grimmer, and Seyfarth, 2010b). While it is easy to define abstract muscle-inspired properties in simulation, hardware developments in real robots are slow, making it difficult to prototype actuator architectures. Additionally, these studies used simplistic controllers which limits the complexity of the learned movements and their transferability to the real world. To enable the use of muscle-like properties in robots and study their effects on learned behaviors, it requires a control method capable of generating complex movements that can be deployed in the real world, while having a flexible actuator architecture capable of mimicking muscles.

Recently, RL has been used in some studies to investigate the effect of the control space on the learning performance, as RL is a method capable of creating feedback controllers across morphologies, therefore constituting a fair comparison. Martín-Martín et al. (2019) found that variable-impedance control with RL can outperform position, velocity and force control in manipulation tasks. Peng and Panne (2017) compared several action spaces in an imitation learning setting, one of them being simulated muscles. While being an interesting comparison, these studies did not perform predictive simulations, i.e. learning from scratch, but learned to track existing trajectories. This puts the muscle-like actuator at a clear disadvantage, as its force output is highly dependent on the current muscle length and velocity, while torque- and position- actuators have a wider operating range over which close to maximum force can be used. In contrast, our contributions C and D show how the investigation of muscles properties in robot learning with underspecified tasks and by learning under a wide range of conditions results in different conclusions.

There have also been efforts to build robots with elastic and muscle-like actuators in hardware. Some of these approaches use RL directly with complex artificial muscles, introducing confounding effects which make it difficult to disentangle the potential benefits of muscles. Hwangbo et al. (2019) have shown that sim-to-real transfer can work with serial elastic actuators, provided that the actuator is approximated by a learned model in the simulator. Büchler et al. (2022) used learned controllers on artificial muscle hardware, employing hybrid simulation approaches which can make use of available simulator data efficiently, but use the hardware when necessary. Other works implement emulated muscles in hardware (Serhan, Nasr, and Henaff, 2006; Seyfarth, Kalveram, and Geyer, 2007; Batts, Song, and Geyer, 2015), with simple rule-based controllers. While allowing to disentangle muscle properties and investigate their effect on learned behaviors, these studies rely on simple rule-based controllers which do not exhibit the required complexity to let the morphology meaningfully affect the result. All the mentioned approaches also require some optimization directly on the hardware.

Our contribution to this topic is two-fold. In contribution C, we first investigate the feasibility of using muscle-like actuators for various tasks with RL, OC and MPC in simulation. This study uses two different simulators, rooted in the fields of biomechanics and robotics, respectively, with muscle models of varying complexity.

By using flexible and powerful optimization techniques, we can probe the potential benefits of abstract muscle properties on the resulting movements. After performing these experiments on locomotion and manipulation tasks in 2D and 3D, we can make strong statements about the robustness and behavioral bias that muscle-like actuators encode.

Going further in contribution D, we again leverage RL to train complex and adaptive controllers directly in simulation, but incorporate a simplified muscle model within a GPU-based simulator, achieving significantly reduced training times while retaining the flexibility to define abstract actuator properties. By making use of GPU-parallelization and domain randomization, we can increase the robustness of our controllers, allowing us to finally bridge the gap between simulation and hardware with a real-time emulation interface that replicates the simulated actuator on the physical system. We then identify damping as the most important parameter for stability of the emulated actuator and derive a damping rule. This allows us to set stable damping values for a wide range of parametrizations of the muscle. This framework enables us to study the impact of muscle-like embodiments on learned behaviors and offers the potential to simplify reward design for sim-to-real transfer in quadrupedal robots.

Unlike previous methods, our RL-based controllers scale to complex behaviors and can be trained robustly in simulation, ensuring reliable performance on real-world hardware. Additionally, the framework allows dynamic adjustments to the actuator structure during training, offering flexibility for exploring various designs. This approach not only facilitates the investigation of muscle-inspired actuators but also serves as a versatile testbed for rapid actuator development and iteration, free from the constraints of predefined, simplistic controllers. Ultimately, it has the potential to allow for the joint development of hardware and controller, allowing for the emergence of completely new actuator types that do not have a corresponding classical control approach yet.

1.4 Outline

Chapter 1 contains a general introduction to the problem space of biomechanics, while detailing some of the difficulties that the control of musculoskeletal systems entails. It ends on a brief discussion on the choice of actuator in robot learning. In chapter 2, we detail the technical background required to understand the contents of this thesis. A special focus is put on the difference between the used simulation engines, common control approaches and their difficulties. We also give some background on DEP and dynamical systems, as well as actuator modeling and issues that might arise when working with software emulation and hardware. Chapter 3 lists the publications that were used in this thesis, contains a summary for each of them and details my contributions to each manuscript. Finally, chapter 4 summarizes the research in this thesis and discusses common difficulties in the field as well providing an outline on how our contributions might be used in the future.

Chapter 2

Methods and literature background

2.1 Simulation engines

Simulation engines are central to this thesis and the current computational biomechanics and robot learning landscape. We denote as simulation engine a software that enables us to numerically compute an approximate solution to a physical system. Historically, such modeling softwares have aimed to be as accurate as possible: the strongly prevalent choice for biomechanics being OpenSim (Seth et al., 2011), an open-source biomechanical modeling and simulation software. This paradigm is shifting, as modern deep learning methods can benefit from large amounts of (synthetic) data, even if it is not a perfect reflection of the real world (Rudin et al., 2021; Makoviychuk et al., 2021; Sferrazza et al., 2024).

This section describes the mathematical assumption of different simulation softwares that have been used in the works underlying this thesis. We used HyFyDy (Geijtenbeek, 2021), a musculoskeletal modeling software closely related to OpenSim, and MuJoCo (Todorov, Erez, and Tassa, 2012), a widely used robotics simulator with MSK components. Both engines were chosen over the gold-standard OpenSim (Delp et al., 2007), which is thousands of times slower than MuJoCo and HyFyDy, as modern optimization methods require an immense amount of data.

2.1.1 Multi-Body-Physics

All the considered simulation engines are capable of simulating multi-body physics, i.e. the temporal evolution of rigid bodies and their associated DOF under contacts and joint constraints. This type of simulation is generally expressed in terms of the following equation, for which a solution is numerically obtained:

$$M(q)\ddot{q} + C(q, \dot{q}) = F, \tag{2.1}$$

where M is the inertia matrix of the system, q is the vector of joint angles, C is the matrix of coriolis, centrifugal and gravitational forces and F contains all internal and external forces, such as actuators and contacts. HyFyDy uses global coordinates, such that each body in the world can move in 6D (linear and rotational DOF), while OpenSim and MuJoCo use reduced coordinates, where the root body is given 6 DOF and all other bodies can move with respect to the root body as moveable joints. All

other forces, such as soft joint limits, tissue with spring and damper properties and motor/muscle forces are modeled either as constraints or as actuators.

Joint limits In contrast to OpenSim, which uses linear stiffness and constant damping at the joint limit, HyFyDy models joint limits with a displacement dependent damping force, given by:

$$\tau_j = -k_\alpha q_j (1 + k_\omega \dot{q}_j) \tag{2.2}$$

where τ_j is the limit force in DOF j, k_α is the displacement stiffness coefficient, q_j is the displacement angle over/under the threshold (zero in case that no limit is violated), k_ω is the damping coefficient and $\dot{q}$ is the angular velocity. This results in a linear increase of counter-force and the damping prefactor when the joint limit is exceeded. In contrast, OpenSim uses a linear increase in counter-force and an instantaneous jump of the damping from zero to $k_\omega \dot{q}_j$.

MuJoCo uses the same constraint model for joint limits as any other constraints. Constraint-free accelerations are modified with:

$$\ddot{q}_1 + d\,(b\,\dot{q} + k\,r) = (1 - d)\,\ddot{q}_0, \tag{2.3}$$

where $\ddot{q}_0$ is the acceleration of a DOF before the correction, $\ddot{q}_1$ after the correction, b is a constraint damping constant, k is a constraint stiffness constant, r is the violation distance in configuration space and d determines how strongly the constraint is enforced. For the example of a hinge joint limit, the lower range limit is defined by: $r(q) = q - q_{\min}$, while $d(q)$ is a polynomial spline controlling how quickly the constraint becomes active.

Contact models HyFyDy uses the Hunt-Crossley contact model (Hunt and Crossley, 1975) with a modified friction computation, similar to OpenSim. This model provides more realistic damping than the Kevin-Voigt-Model (Goldsmith and Frasier, 1960), which is a simple dissipative spring-damper contact model. The HC-contact model is only defined for contacts between analytical shapes, such as planes and spheres. Any foot geometry is approximated by a set of contact spheres attached to the feet (Weng, Hashemi, and Arami, 2022).

MuJoCo uses a custom contact model, based on its general constraint solver. Instead of assuming infinitely stiff contacts, as in other robotics simulators, MuJoCo computes a soft-contact constraint, which is then driven by the constraint solver through a convex optimization. While the engine is tuned for robotic contact (rigid body) by default, the contact constraints can be adapted to portray soft and dissipative contacts. Friction is computed by an elliptical or pyramidal friction cone model. The contact solver can compute contact forces for arbitrary convex meshes, but most models use a combination of ellipsoid and spherical primitive contact objects at the feet.

2.1.2 Muscle model

All the used simulators use variations of Hill-type muscle model. These models mainly group larger numbers of muscles into subgroups for which the dynamics of the muscle tendon, the muscle fiber and the activation dynamics are computed.

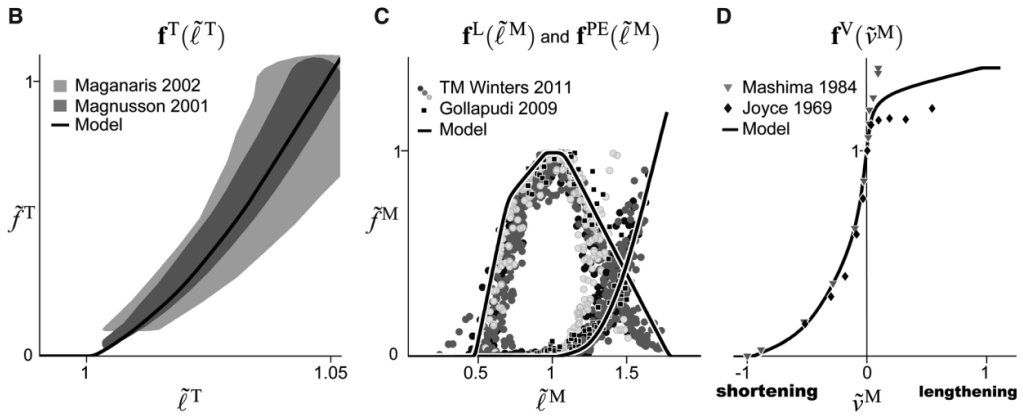


FIGURE 2.1: The active force-length, passive force-length, force-velocity and tendon force curves used in the Millard model. Illustration from Millard et al. (2013).

HyFyDy While HyFyDy supports several muscle models, the recommended one is the Millard equilibrium model (Millard et al., 2013). We will first describe this model, before detailing how it differs from other formulations.

Hill-type muscles are approximated by defining a muscle fiber, mainly characterized by its length, velocity and pennation angle, and a muscle tendon, which is connected to the fiber and to bone. Bones are generally assumed to be rigid and inelastic by most simulators.

The muscle fiber has length and velocity dependent active force profiles and exhibits a passive force when stretched. The active force in the Millard model is given by:

$$f^{\mathrm{CE}}(\tilde{l}_{\mathrm{CE}}, \tilde{v}_{\mathrm{CE}}, a) = f^{opt}\left(f^{\mathrm{L}}(\tilde{l}_{\mathrm{CE}})f^{\mathrm{V}}(\tilde{v}_{\mathrm{CE}})a + f^{\mathrm{PE}}(\tilde{l}_{\mathrm{CE}})\right), \tag{2.4}$$

where $\tilde{l}_{\mathrm{CE}} = l_{\mathrm{CE}}/l_{\mathrm{CE}}^{\mathrm{opt}}$ is the fiber length scaled by its optimal isometric length, $\tilde{v}_{\mathrm{CE}}/v^{\mathrm{max}_{\mathrm{CE}}}$ is the fiber velocity scaled by its maximum value, f is the total force of the muscle fiber, f^{PE} is the force of the parallel elastic element, f^L and f^V are the force-length and force-velocity relationships and a is the muscle activity. Assuming negligible muscle mass and inertia, and an elastic tendon, the following equilibrium equation holds for a given muscle-tendon-unit state:

$$f^{\mathrm{CE}}(\tilde{l}_{\mathrm{CE}}, \tilde{v}_{\mathrm{CE}}, a)\cos\alpha - f^{\mathrm{T}}(\tilde{l}_{\mathrm{T}}) = 0, \tag{2.5}$$

where $\tilde{l_{\mathrm{T}}}$ is the slack-length normalized muscle tendon, f^T the nonlinear tendon force and α is the pennation angle with which the muscle attaches to the fiber. Unfortunately, the solution to this equilibrium equation has several singularities, requiring additional constraints to reduce the “stiffness" of the equation.

A set of assumptions that can be made lead to the damped equilibrium model:

- Pennation angles are constrained to $\alpha < 90°$.
- An additional damping element is introduced.

- Muscle fiber length is constrained to have a minimum.

This leads to the following:

$$f^{\text{opt}}\left(f^{\text{L}}(\tilde{l}_{\text{CE}})f^{\text{V}}(\tilde{v}_{\text{CE}})a + f^{\text{PE}}(\tilde{l}_{\text{CE}}) + \beta\tilde{v}_{\text{CE}}\right)\cos\alpha - f^{\text{opt}}f^{\text{T}}(\tilde{l}_T) = 0. \tag{2.6}$$

Considering forward dynamics simulation, this equation can be solved for $\tilde{v}^{\text{CE}}$ to compute a forward step. This requires the application of a root finding algorithm at each iteration.

In HyFyDy, two changes are made to the Millard equilibrium model. First, the muscle length and force curves are replaced by polynomial functions, second, the root finder is replaced by an explicit formulation. Although the exact details are not publicly available, this change presumably greatly speeds up the simulation, while retaining results close to the original model. Nonetheless, HyFyDy requires a variable step-size ODE solver to ensure stability.

MuJoCo While MuJoCo is primarily a rigid body simulator for robotics, the engine contains a simplified muscle model that is used by a range of studies on musculoskeletal systems (Collings et al., 2022; Barbera et al., 2021; Caggiano et al., 2022). The primary difference to the HyFyDy Hill-type muscle model is the absence of pennation angle and the assumption of a rigid tendon. This assumption greatly stabilizes the simulation and allows much higher computational speeds. While a rigid tendon leads to inaccurate predictions for certain tendons in certain conditions, the impact of this inaccuracy greatly depends on the specific application. The introduced error is negligible when the optimal muscle fiber length is smaller than the tendon slack length (Millard et al., 2013; Rajagopal et al., 2016a).

Consequently, the muscle fiber force is also given by (2.4), but the equilibrium equation (2.5) does no longer apply, as the tendon cannot stretch. This choice along with the general MuJoCo framework, allows the use of a constant step-size integrator while retaining stability. The final force equation is:

$$\tau = f^{L}(\tilde{l}_{\text{CE}})\, f^{V}(v_{\text{CE}})\, a + f^{\text{PE}}(\tilde{l}_{\text{CE}}) \tag{2.7}$$

where f^L and f^V are phenomenological force-length and force-velocity functions which are similar, but distinct from the OpenSim ones. They do not portray a specific organism, but have been modeled in order to exhibit general animal-like properties.

2.1.3 Activation dynamics

Muscle fibers cannot contract or relax arbitrarily fast. When a motor neuron connecting to a muscle fiber is innervated, calcium ion concentrations within the cell gradually increase over 5-50 ms until reaching the target contraction level, depending on the type of muscle. This process is usually modeled by a first order ODE, with different time constants for activation and deactivation of the muscle.

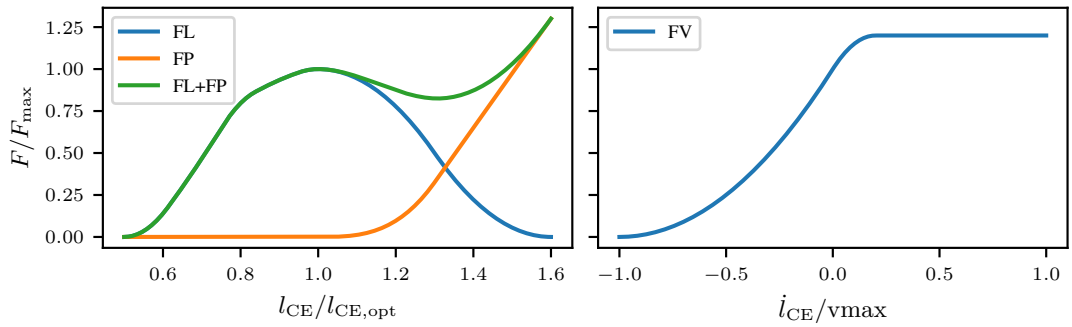


FIGURE 2.2: Force-length (FL) and force-velocity (FV) relationships and passive force (FP) used in MuJoCo. We use the same phenomenological functions in our own MuJoCo muscle model. While FL and FV get scaled by the current muscle activity a, FP does not (see Eq. 2.7). Illustration from Wochner et al. (2022).

HyFyDy In HyFyDy, muscle activation is modeled by:

$$\frac{\partial a(t)}{\partial t} = (u_t - a(t))(c_1 u_t + c_2), \tag{2.8}$$

where h is the simulation timestep, u_t is the control signal at time t and c_i are activation and deactivation time constants. By default, $c_1 = 75$ and $c_2 = 25$.

MuJoCo In MuJoCo, these dynamics are modeled identically to OpenSim (Winters, 1995; Thelen, 2003):

$$\frac{\partial a(t)}{\partial t} = \frac{1}{\tau(u(t), a(t))}(u(t) - a(t)), \tag{2.9}$$

where the control signal is clamped to the range [0, 1]. The time constant τ is determined by the control and states as follows:

$$\tau(u(t), a(t)) = \begin{cases} \tau_{\text{act}} \cdot (0.5 + 1.5 \cdot a), & \text{if } u - a > 0 \\ \frac{\tau_{\text{deact}}}{0.5 + 1.5 \cdot a} & \text{if } u - a \leq 0, \end{cases} \tag{2.10}$$

The default values for the time constants are $\tau_{\text{act}} = 0.01$ and $\tau_{\text{deact}} = 0.04$, following the formulation by Zajac (1989).

2.2 Control for MSK systems

Controllers generate control signals which move a system towards a desired trajectory or goal state. While control theory is a vast field that has proposed countless solutions to simple and complex systems, computational motor control is particularly hard to perform.

2.2.1 What makes it difficult?

Optimality and Robustness In order to break the redundancy of the control problem, a large amount of motor control literature has focused on the case of *optimal* control. Under this paradigm, a cost-function assigns a numerical value to a given system trajectory. Optimization approaches are then used to find the control strategy which optimizes the cost under the constraint that either a desired trajectory, or given start and end points are approached. Numerous experiments with human subjects have shown close correspondence between recorded data and trajectories obtained by optimizing for control signals under cost functions such as cost-of-transport (Umberger, Gerritsen, and Martin, 2003; Uchida et al., 2016), control effort (Haeufle et al., 2020a), smoothness (Flash and Hogan, 1985) or maximum joint torques.

There has, however, been strong evidence that the human motor system is **not** producing optimal behaviors, but rather tries to achieve “good-enough" control. The idea is that through trial-and-error in many different situations, a repertoire of motor solutions is built. Starting from this representation, a human can quickly synthesize a movement for any of a wide range of different goals, even though the solution might not be perfect. Considering the large number of constraints that the human motor system has to act under, it seems unlikely that a single optimal solution can quickly be found for any situation that arises (Loeb, 2012). Additionally, there is evidence that habitual control dominates over optimality when motor changes are introduced, at least on the timescale of hours (Rugy, Loeb, and Carroll, 2012).

Imperfect signals Human proprioception is inherently limited. Visual, proprioceptive and vestibular signals are delayed (Keyser et al., 2023) and incomplete. While muscle spindles measure muscle fiber length and velocity, the full muscle-tendon-unit contains the elastic tendon, which is only measured by a Golgi-tendon force sensor. The neuro-musculoskeletal system has to merge, process and augment the available information in order to accurately move in the world (Kistemaker et al., 2013; Kasuga et al., 2022). Any holistic control approach will have to generate a similar multimodal understanding in order to function.

2.2.2 Common control approaches

We summarize some of the most common control approaches pertinent to this thesis, and how the deal with the mentioned complexities of human motor control.

Optimal control OC is a long standing paradigm in the field of robotics and computational motor control. The goal of various OC methods is to find a control trajectory, given here by $\pi_t(x_0, ..., x_t)$, and resulting states x_t that minimize a cost function $l(x_t, u_t, t)$, given an temporally advancing environment $x_{t+1} = f(x_t, u_t, t)$:

$$\begin{aligned} \min_{\pi_t} J &= \min_{\pi_t} \sum_{t=0}^{T} \ell(x_t, u_t, t), \\ \text{subject to} \quad x_{t+1} &= f(x_t, u_t, t), \\ u_t &= \pi_t(x_0, \ldots, x_t). \end{aligned}$$

There are a range of open-source frameworks to handle this open-loop optimization problem (Dembia et al., 2021; Michaud et al., 2022), which find optimal solutions over the entire problem horizon.

Model predictive control MPC constitutes a closed-loop variant of OC which takes into account sensory feedback received over a short future horizon, which is then incorporated into the control optimization. This has the advantage of being able to account for external perturbations and model uncertainties, especially under learned models. While modern approaches are capable of real-time MPC over short horizons, the resulting controllers only act on information inside the planning window, as every movement is planned from scratch, ultimately leading to suboptimal performance in certain scenarios (Kaufmann et al., 2023).

Rule-based control Rule-based controllers (Geyer and Herr, 2010; Song and Geyer, 2013) are particularly attractive as they can not only reproduce periodic human behavior and realistic reactions to perturbations (Song and Geyer, 2017), they also mimic human control structures like length, velocity and force reflexes. While they are based on hand-crafted equations, the free parameters of these rules are often optimized with biologically-inspired cost functions in mind, making them a subset of OC. Their disadvantage lies in the difficulty to design controllers for very high-dimensional systems, behaviors that go beyond periodic motions and limited robustness to external perturbations. Nonetheless, recent works have extended their use to larger motion (Bunz, Pawusch, and Schmitt, 2024) and model (Caggiano et al., 2024) complexity.

Reinforcement learning RL is a closed-loop control method that aims to optimize a reward function, similar to MPC and cost functions. The crucial difference between both methods lies in the fact that RL can be used to directly optimize a neural network by approximating Q-functions: These learned value functions can incorporate information about future states and distill vast amounts of data, potentially accounting for much longer horizons than MPC. Formally, the RL-methods used in this thesis are optimizing the expected discounted sum of rewards:

$$J(\pi) = \mathop{\mathbb{E}}_{\tau\sim\pi}\left[\sum_{t=0}^{\infty}\gamma^t r(s_t, a_t)\right], \tag{2.11}$$

where $\gamma \in [0, 1]$ is the discount factor. While using RL for single-task optimization is often requires more computational resources to achieve a comparable solution than approaches like trajectory optimization and MPC, it offers distinct advantages. RL controllers can be trained under a large variety of different tasks, environments, and initial conditions. This allows the used networks to learn form more general underlying representations, potentially resulting in a controller which generalizes well to new situations and is incredibly robust. It has been shown that controllers trained across many tasks can more easily be fine-tuned for new tasks than an expert that excels on a single task (Caggiano, Dasari, and Kumar, 2023). This connects back to the notion of creating habitual representations which are “good-enough” and

can easily be adapted to new tasks with little training (Loeb, 2012; Rugy, Loeb, and Carroll, 2012).

Additionally, RL agents can be trained under strongly limited sensor modalities. Robotic systems often have to deal with incomplete and constrained sensors that do not allow for accurate knowledge of the outside world. By embedding understanding over the recent sensor history (Kumar et al., 2021), and by leveraging the long-horizon credit assignment of RL value-functions, RL controllers can infer implicit information about the world in order to compensate for delayed sensors and partially-observable environments.

2.2.3 What makes RL hard?

RL brings several advantages over other optimization-based control methods, but comes with caveats. RL agents are notoriously difficult to train, partially owed to the presence of the deadly triad: function-approximation, bootstrapping and off-policy learning (Sutton and Barto, 2018).

Consequently, several issues which are present but manageable with OC methods become extremely complex do deal with:

Exploration RL agents need to explore their environment in order to find good actions (Amin et al., 2021). Considering the explosion of dimensionality with MSK-models, it is clearly difficult for RL to work well.

Data RL algorithms need a tremendous amount of data which is difficult to obtain with traditional MSK-simulators (Rudin et al., 2021).

Reward Design The reward function for RL controllers are notoriously difficult to tune, making it necessary to design new function leading to the desired optimal behavior, while not getting stuck in a local optimum (Ibrahim et al., 2024).

2.2.4 DEP and self-organization

The concept of self-organization in dynamical systems describes how complex behaviors can arise from the interplay of simple behavioral rules with an external embodiment. A common example is the organization of a fish swarm, where local behavior rules create emergent dynamics of the swarm. Another example are neurons in the human brain, which also follow certain abstract behavior rules. One such rule is of particular importance, as explaining a wide range of neural phenomena (Keysers and Gazzola, 2014) as well as being relevant to developments in modern ML (Ramsauer et al., 2021): Hebbian learning describes how a set of neurons that fires at the same time also increases their connectivity: what fires together, wires together.

Unfortunately, Hebbian learning only adapts neuron connections in response to external stimuli, without taking into account an action that is performed in the world.

In order to arrive at a formulation that incorporates a sensorimotor loop—a connected cycle of an agent receiving sensor information, computing an action and executing it in the world—let C be a controller matrix that computes control signals y based on an input x, illustrated in Fig. 2.3 on the left.

More precisely:

$$y = \tanh(\kappa C x + h), \tag{2.12}$$

where κ is an amplification constant, h is a time-dependent bias, y is the output action, x is the state and C is the learned control matrix. In this scenario, a Hebbian learning-based update rule for the controller matrix takes the form:

$$\dot{C}_{ij} \propto x_i y_j, \tag{2.13}$$

where $\dot{C}$ denotes the change in C. The controller is in this case updated proportionally to the product of inputs and outputs. Hebbian learning rules could in principle be used without using C as a controller, just by repeatedly presenting appropriate inputs and output signals, thereby modeling correlations in the presented data. In contrast, a differential Hebbian learning rule, based on input and output derivatives, would be:

$$\dot{C}_{ij} \propto \dot{x}_i \dot{y}_j, \tag{2.14}$$

connecting the change in C to proportional changes in x and y. This changes the concept from the well known "what fires together wires together" to "chaining together what changes together". Further, DEP introduces several key changes to induce a more dynamic nature which lead to the differential extrinsic plasticity learning rule. Firstly, there is a causal connection between a *future* timestep under an inverse prediction model $f(\dot{x}_t) = y_{t-\Delta t}$. This connection shapes the change in C:

$$\dot{C}_{ij} \propto f(\dot{x}_t) \dot{y}^T_{t-\Delta t}. \tag{2.15}$$

The model f predicts which quantities in the sensory input were affected by control actions taken shortly before. In the original formulation, f was assumed to only predict local effects, leaving an error $\delta\dot{y}$ caused by extrinsic effects. This emphasizes the role of f to capture either only direct and embodiment-specific dependencies, connecting actuators to sensors through causal relationships. Note that the quantities x_t and y_t are now vectors with time subscripts.

Crucially, this learning rule takes into account temporal effects through the dependency on t+1. In other words, the correlation matrix between velocities at timestep t and $t + 1$ drives the update of C. A common choice in prior work is a linear mapping $M : \mathbb{R}^d \to \mathbb{R}^n$ which approximates the connection between sensors and actuators.

This choice of M can be further simplified by assuming the identity matrix. We emphasize that the identify matrix still contains valuable information, as in the form

$$M\dot{x} = \dot{y} \tag{2.16}$$

it contains information about which sensor is connected to which actuator. While this can be assumed to be a one-to-one mapping in some cases, this is not a given. M can also be learned or deliberately set to induce certain behaviors in the model.

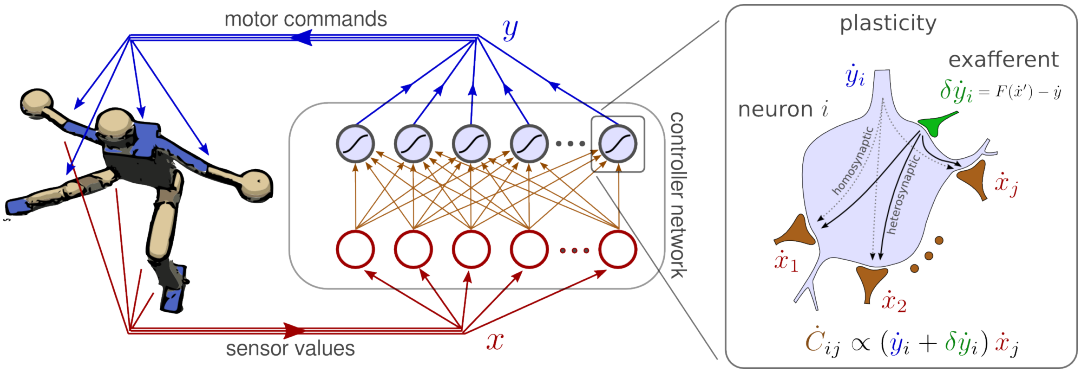


FIGURE 2.3: An illustration of DEP, figure from (Der and Martius, 2015). Left: The controller used in the sensorimotor loop. Sensor signals x_t enter the controller matrix C and produce control signals y_t. Right: An exemplification of the update rule used to adapt the controller. The change in the controller $\dot{C}$ is driven by the correlation between the change of the sensor signal $\dot{x}$ and the change in the output command $\dot{y}$ and the modeling error $\delta\dot{y}$. As we use the identity matrix as our prediction model, the update simplifies to (2.15). Previous studies have manually set the prediction model or learned it from data.

2.3 Robot Learning

While the field of learning for robotics is incredibly vast, this thesis mainly focuses on the use of bio-inspired actuators. The application of data-driven learning to muscle-like actuators has been the focus of a small number of studies with real hardware (Büchler et al., 2022; Ma et al., 2023; Volchko et al., 2022), but the development of actuators and control algorithms has mostly been separated into two distinct communities. We detail the relevant actuators and low-level control mechanisms relevant to this thesis, before focusing on the difficulties and solutions when transferring simulation-trained controllers to a real robots.

2.3.1 Actuation

Actuation is defined as the ability of a robotic system to convert energy into motion. The robotic systems we consider in this thesis are either bipedal or quadrupedal, meaning that they move with two or four legs, respectively. To effect this movement, each limb of the robot is split into links which are connected with moveable joints. These joints are moved by actuators, allowing the robot to act in the world. There are, however, many types of actuators, some of which are distinguished by their hardware structure and some by low-level software that changes how the force output of the actuator depends on the given control input.

Let us consider an actuator to be a function $f_{\text{act}}(u) = \tau : \mathbb{R}^c \to \mathbb{R}^m$ that maps a given control signal to an output torque. In general, this mapping does not need to be instantaneous, as actuators can have delayed responses, the mapping does not need to be linear, as the produced forces often have certain limits or depend on the robot state, and the mapping does not need to conserve the dimensionality of the control signal: A robotic system is said to be over- or underactuated if the dimensionality of the control signal is larger or smaller than the number of degrees of freedom.

Torque Control The simplest actuator we could consider is an idealized torque actuator:

$$f_{\text{ideal}}(u(t)) = f_{\max}\, u(t) = \tau, \tag{2.17}$$

where we consider $u \in [-1, 1]$, for which τ varies in $[-f_{\max}, f_{\max}]$. In this case, $c = m$, as there is one actuator per DOF.

PD Control A widespread low-level controller is the PD controller. In contrast to the torque actuator, its input represents not a percentage of the maximal torques, but a desired position and velocity to be achieved. Formally, it is given by:

$$f_{\text{PD}}\left(u(t), q(t), \dot{q}(t); P, D\right) = P\ (\bar{q}(t) - q(t)) + D\ (\dot{\bar{q}}(t) - \dot{q}(t)) = \tau, \tag{2.18}$$

where q is the current sensor value of the joint angle, $\dot{q}$ its velocity, $\bar{q}$ marks target values and P and D are parameters determining the controller's stiffness and damping properties. In the above equation we assumed that the control signal encodes a desired position and velocity $u = (\bar{q}, \dot{\bar{q}})$. Even in this simple case, we have $c = 2\,m$, meaning that the control signal is twice as large as the number of DOF.

Muscle Control Realistic muscle-driven hardware is often realized by using artificial McKibben muscles (Tondu and Zagal, 2006). Driven by inflatable rubber cylinders filled with air or liquid, these actuators possess many features of real muscles, such as a force-length and force velocity dependence and time-delayed force modulation (Gong and Yu, 2022). These properties in addition to strong hysteresis effects make them difficult to control and the development of new architectures is inherently slow.

We propose to study a simplified muscle model in simulation to train a controller which can be transferred to a robot. The robot in turn uses a low-level controller that emulates the muscle model, displaying abstract muscle properties in real-time. For this goal, we simplified the MuJoCo muscle model from Sec. 2.1.2 further by assuming no tendon, which allows us to represent the muscle length as a linear function of the joint angle (Kistemaker, Van Soest, and Bobbert, 2006; Bayer et al., 2017):

$$l_{\text{CE},i} = m_i\, \phi_j + l_{\text{ref},i} \tag{2.19}$$

$$\dot{l}_{\text{CE},i} = m_i\, \dot{\phi}_j, \tag{2.20}$$

where ϕ_j is the joint angle, m_i and $l_{\text{ref},i}$ are computed from user-defined parameters, and $i \in \{1, 2\}$, as we assume two antagonistic muscles per joint. The modeling of activation dynamics is identical to the MuJoCo model.

By choosing this simple relationship between joint angle of the robot and muscle length, we obtain a model which can be efficiently computed on real hardware while still encoding fundamental properties of muscle actuators.

2.3.2 To the real hardware

Deploying controllers trained in simulation on real hardware is a long-standing challenge of robotics. While rigid-body robotics simulators have increased in accuracy

and computational performance over the years, many hard to model effects remain. Some studies rely on system identification, namely precisely assessing the hardware and replicating it as closely as possible in simulation, but non-stationary effects can be hard to parametrize. Elastic components, contacts and some types of friction are notoriously difficult to model in simulation.

A more powerful paradigm has been to use large-scale domain randomization (Zhao, Queralta, and Westerlund, 2020). By combining powerful GPU-based simulators with deep RL controllers which can absorb a vast amount of experience, it is possible to perform physics randomization and sensor noise injection on an unprecedented scale. Randomization across thousands of environments allows the agent to perform robustly across a wide distribution of conditions, improving generalization to the real world.

The following components are used in this thesis to facilitate sim-to-real transfer:

- We add stochastic offsets to the mass, friction and force values of the robot.
- Normal-distributed noise is added to all sensors of the robot.
- A force is applied to the center of mass of the robot at random intervals.
- Small normal-distributed offsets are added to the initial pose and the initial muscle activity of the robot.
- We regularize the policy actions to not use excessive torques, protecting the robot from breaking.

We additionally use a command curriculum, where the robot initially receives a very small target velocity to achieve, whose range is slowly increased over training.

Indeed it is not enough to just randomly vary the conditions of the simulation, the policy has to also encounter a wide distribution of states during training in order to achieve the necessary robustness for sim-to-real transfer. Finally, training with a zero-velocity command in simulation also allows us to safely setup the robot in a standing position in the real world and to then slowly increase the commanded velocity.

2.3.3 Implementing a real-time actuator

Real-time software emulation of an actuator is a versatile concept that allows researchers to quickly iterate on different control architectures. Unfortunately, it also brings specific challenges that practitioners should be aware of. Most of the specific points mentioned in this section are relevant to all robot learning applications, but I will detail the specific relevance to emulated actuators. The key difference to known low-level controllers like PD or impedance controllers is that during actuator prototyping it is not clear what the exact result should be. While a PD-controller can ideally be expected to attain a given position, it is not known which joint angle a muscle-like actuator should obtain for a given combination of muscle excitation signals.

Tuning Our muscle-actuator possesses a large number of free parameters. While we can use simulation to optimize for *performant* parameters with respect to an external

task, we cannot guarantee that the obtained controller is *feasible* on the real hardware, as the learner might exploit simulator inaccuracies. One solution is to define a feasible parameter region within which a performance-based optimization of hyperparameters can be performed, as we did in contribution D.

Control frequency Control frequency is an often overlooked detail in simulation and hardware. While some people tune the simulator timestep for a stable simulation, this value also affects the RL episode horizon, which mostly works well with episodes between 200 and 1000 in length. Additionally, the control frequency also has a strong effect on the performance of low-level PD-controllers, with damping becoming unstable at slow frequencies (Tan, Liu, and Turk, 2011). While most off-the-shelf controllers are highly optimized in order to run at hundreds of Hz, it is much more difficult to ensure fast execution with a software actuator. Every piece of the hardware pipeline, from the on-board computer to the motor boards to the type of employed sensors can add latency, ultimately destabilizing the actuator. The effect of the control frequency on damping is illustrated in Fig. 2.4.

Joint friction Many simulators employ linear or smoothly increasing joint friction and damping. As real joints often exhibit static and dynamic friction, small and precise movements in the simulator often do not transfer well to the hardware.

We tackle these issues in contribution D by starting with simplistic control signals, e.g. reaching a single stable joint angle. This simple and known desired result allows us to establish a set of stable parameters which is at least feasible, before further optimizing it towards more complex behaviors.

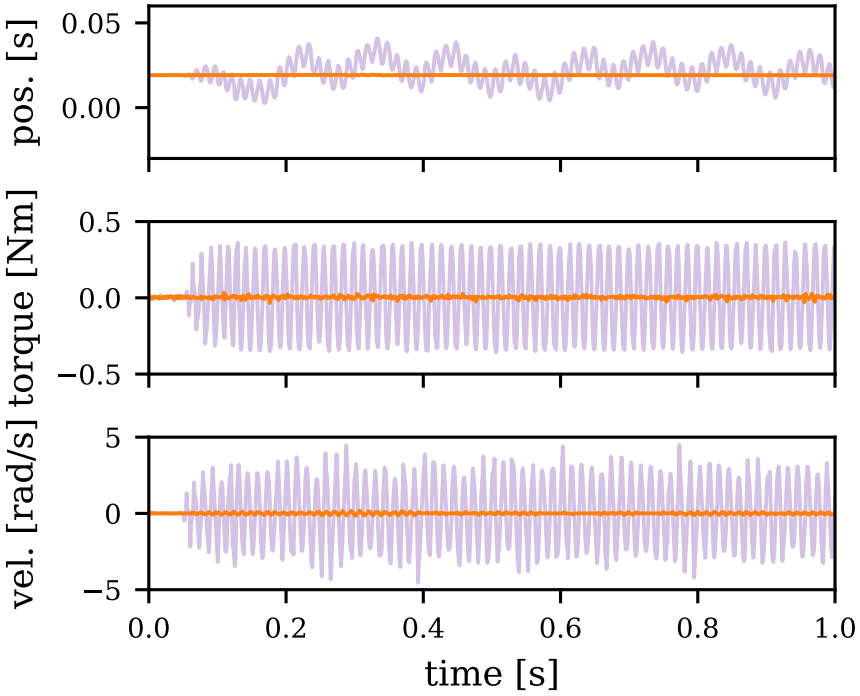


FIGURE 2.4: An emulated muscle actuator is implemented on a robot and a constant control signals is applied. We compare two different values of a muscle damping parameter β. For $\beta = 0.66$ (purple), a damping value that is infeasible wth the hardware given the attained control frequency of our setup, strong oscillations can be observed. For a damping value of $\beta = 0.36$ (orange), given by our derived damping rule, the system can produce adequate damping to stabilize the actuator. This example illustrated how emulated actuators can only be given properties that are achievable with the hardware constraints. Figure from Schumacher et al. (2024).

Chapter 3

Publications

3.1 Contribution 1 — Schumacher et al. (2023)

Title: DEP-RL: Embodied Exploration for Reinforcement Learning in Overactuated and Musculoskeletal Systems

Authors: Pierre Schumacher, Daniel F. B. Haeufle, Dieter Büchler, Syn Schmitt and Georg Martius

Journal/Conference: International Conference on Learning Representations

Year: 2023

Link: https://openreview.net/forum?id=C-xa_D3oTj6

Summary: We isolated the influence of overactuation on exploration to be one of the key factors preventing good performance with RL algorithms when applied to high-dimensional musculoskeletal models. We propose DEP, a technique from the domain of dynamical systems, as an exploration technique that can produce effective joint level exploration for muscle-driven systems in an efficient way. The combination of DEP with existing RL methods allows us to control several human and animal musculoskeletal models and produce the fastest achieved running velocity on an ostrich model with 120 muscles.

Contribution: Under the supervision of Daniel F. B. Haeufle and Georg Martius, I conceptualized the project and took primary responsibility for designing and implementing the computer simulations. In collaboration with Georg Martius and Daniel F. B. Haeufle, I implemented DEP in a modern Python codebase and designed the combination with RL. I carried out all simulations, performed data analysis, interpreted the results and designed the figures. I was responsible for writing the majority of the manuscript, as well as creating all supplementary electronic materials, while reflecting on text sections together with my co-authors. Additionally, I managed the

manuscript submission process, addressed reviewers' feedback, coordinated project meetings, facilitated discussions with co-authors, and oversaw overall project planning.

3.2 Contribution 2 — Schumacher et al. (2025)

Title: Emergence of natural and robust bipedal walking by learning from biologically plausible objectives

Authors: Pierre Schumacher, Thomas Geijtenbeek, Vittorio Caggiano, Vikash Kumar, Syn Schmitt, Georg Martius and Daniel F. B. Haeufle

Journal/Conference: iScience

Year: 2025

Link: https://doi.org/10.1016/j.isci.2025.112203

Summary: Prior work on human motion generation in predictive simulaiton with RL suffered from unnatural markers in the produced behaviors. We extended our previous approach by identifying likely cost terms contributing to human motion generation. Our pipeline optimized all reward terms, except for muscle effort, with the goal of maximizing a validation metric obtained from experimental data. We identified the muscle effort to be a critical factor in RL optimization. In contrast to other studies relying on a series of hand-crafted learning curricula, we propose an adaptive muscle effort reward term, enabling the successful optimization of close-to-natural motion with 4 human musculoskeletal models across 2 simulation engines.

Contribution: I conceptualized the manuscript under the supervision of Daniel F. B, Haeufle and Georg Martius, while confering regularly with my co-authors. I took primary responsibility for designing and implementing the computer simulations and optimization algorithms as well as the reward function design. Thomas Geijtenbeek provided helpful advice with respect to the simulation software and implemented the simulated obstacles for this study. I conferred with Vittorio Caggiano and Vikash Kumar for results pertaining to the MuJoCo simulation. I carried out all simulations, performed data analysis, interpreted the results and designed the figures. Thomas Geijtenbeek lightly edited some figures. Additionally, I managed the manuscript submission process, addressed reviewers' feedback, coordinated project meetings, facilitated discussions with co-authors, and oversaw overall project planning.

3.3 Contribution 3 — Wochner and Schumacher et al. (2022)

Title: Learning with Muscles: Benefits for Data-Efficiency and Robustness in Anthropomorphic Tasks

Authors: Isabell Wochner*, Pierre Schumacher*, Georg Martius, Dieter Büchler, Syn Schmitt and Daniel F. B. Haeufle

Journal/Conference: Conference on Robot Learning

Year: 2022

Link: https://openreview.net/forum?id=Xo3eOibXCQ8

Summary: We identified the beneficial properties of muscle actuators as being a potential improvement for current learning approaches for robotics. In this study, we created a large-scale investigation of different learning methods, optimal control, model predictive control and reinforcement learning on a biomechanical and a rigid body simulator, both using muscle-like actuators of different levels of complexity. It was found that across simulators and learning approaches, muscle actuators perform better in terms of data efficiency, hyperparameter sensitivity and robustness than controllers optimized with torque- or position controllers. Torque controllers were only more effective in restrictive maximum force tasks or when learning approaches were specifically optimized for them. This highlights the use of muscles as strong general purpose actuators for unstructured environments.

Contribution: Isabell Wochner and I conceptualized the manuscript under the supervision of Daniel F. B Haeufle, Georg Martius and Syn Schmitt. I designed, implemented and performed all experiments with MuJoCo and RL. I performed the data analysis and figure creation as well as the supplemental material for all RL experiments, while giving feedback for all other sections of the manuscript. Isabell Wochner and I were jointly responsible for the manuscript edits, addressing reviewer's feedback and coordinating project meetings, while facilitating discussions with co-authors. Isabell Wochner provided the first draft of the manuscript structure and managed the manuscript submission process.

3.4 Contribution 4 — Schumacher et al. (2024)

Title: Learning to Control Emulated Muscles in Real Robots: A Software Test Bed for Bio-Inspired Actuators in Hardware

Authors: Pierre Schumacher, Lorenz Krause, Jan Schneider, Dieter Büchler, Georg Martius and Daniel F. B. Haeufle

Journal/Conference: IEEE International Conference on Biomedical Robotics and Biomechatronics

Year: 2024

Link: https://ieeexplore.ieee.org/document/10719699

summary: We extended previous work on using simplified muscle models for learning on robotic systems to real hardware. The implementation of the muscle model in a GPU-parallelized simulator was followed by an implementation of a real-time emulated muscle actuator on a real robot. This structure enabled us not only to provide evidence for previously found benefits of muscle actuation in a simulated quadruped robot in running and hopping tasks, but also to transfer this behaviors to a real robot through sim-to-real transfer. This showcases the potential of muscle-like hardware in real robots, while also highlighting software emulation as a powerful approach to test new actuator architectures with RL controllers.

Contribution: I conceptualized the manuscript under the supervision of Daniel F. B. Haeufle and Georg Martius, while guiding Lorenz Krause in the initial exploration of the study. I implemented the first code versions of the muscle model and performed the parameter optimization for all simulation experiments. Lorenz Krause performed investigative simulation experiments in parallel under my supervision which provided helpful for the study. Lorenz Krause implemented the low-level controllers that were run on the robot hardware under my supervision and guidance as well as managing the hardware experiments and recording the robot pictures and videos for the manuscript. I contributed significantly to all parts of the text during the draft and review phases and coordinated the manuscript submission process.

Chapter 4

Discussion and future work

Summary and impact of this thesis The aim of this thesis was to develop new control algorithms that can produce human-like motion with high-dimensional musculoskeletal models, as well as to extract useful insights from human motor control for application in robotics. These contributions were meant to provide easy-to-use building blocks for the community, to enable motor control researchers to make use of advancements in RL control methods that can be leveraged to gain insights into motor control and develop new methods for rehabilitation. It was also shown that the transfer of abstract muscle properties to robotics in the form of a simplified muscle-like actuator lead to several benefits over torque and position control, when combined with learning-based methods. Finally, it could be seen that it is possible to transfer this model to robotic hardware and that the findings observed in simulation also hold in the real world, if the feasibility of the parametric muscle morphology is ensured.

Contribution A, proposes a control algorithm which can reliably solve tasks involving the control of MSK-models of varying complexity, without the need to retune hyperparameters. After an investigation on simplified environments, overactuation is identified as being one of the main reasons for the lacking performance found in common RL methods, even when other muscle properties are not present. It is then shown that DEP, a method from the field of self-organization, can effectively produce exploration noise in overactuated systems. DEP is combined with RL as a embodiment-specific exploration technique which can produce strong joint-level exploration of the MSK-system with minimal fine-tuning. This shows that DEP is able to effectively generate exploration data that is **specific** to the model after only seconds of interaction and with the only prior knowledge being the one-to-one correspondence of muscle sensors to muscle excitation signals. Critically, DEP makes use of the data stream from the RL-agent to update its controller matrix, even when not being applied to the system. And finally, DEP is parallelized such that we have one controller matrix per parallel environment process, greatly augmenting the induced data diversity.

Although the approach used in contribution A is performant, in the sense that it maximizes an external task reward, and robust, in the sense that it reaches high success rates under unseen perturbations, the obtained behaviors lack markers that are commonly associated with natural behavior. In contribution B, DEP-RL is extended with an adaptive reward mechanism, which achieves energy minimization in different tasks and 4 different human models across 2 simulation engines. After identifying a series of promising auxiliary reward terms which can potentially bias the behavior

toward being more natural, and a gait-cycle averaged metric capable of assessing gait quality by a single scalar, we built a pipeline capable of optimizing the auxiliary reward terms for maximal gait quality. It is revealed that muscle effort in particular can be minimized much better if a reward schedule is implemented, which slowly increases the reward prefactor over training. Although such a schedule usually has to be carefully tuned for each model (Weng, Hashemi, and Arami, 2021b), the proposed adaptive effort reward takes into account the current performance of the agent in order to tune it online. The adaptation mechanism is flexible enough to work with 4 different models, ranging from 2D and 18 muscles to 3D and 90 muscles, when combined with our previously developed DEP-RL algorithm.

With these works, it is possible to generate **robust** controllers for high-dimensional MSK-systems. The developed control schemes achieve their impact without the need to adapt parameters or tune a reward across different models, which makes them a strong starting point for further studies into biomechanics and human motor control. Additionally, the ability of RL controllers to absorb a vast amount of experience in combination with DEP, a chaotic self-exploration method, is shown to produce incredibly robust controllers in a wide range of different tasks. The combination of evolutionary priors contained in the simulator assumptions and the definition of the anatomical skeleton, an adaptive bio-inspired reward function and DEP-RL can result in a controller capable of generating close-to-natural motion, while being incredibly robust to external perturbations. We are not aware of other control techniques that achieve this level of generalization in this domain.

In contribution C, we have demonstrated the potential of using bio-inspired actuator properties for robotics. While similar investigations have been performed in the past, this thesis showcases the effect of muscle-like actuators on complex *learned* behaviors when the task is under-specified, allowing the morphology to affect the movement. The used reward and cost functions were only loosely defining the task, by specifying a desired reaching target for an arm or an upward velocity for a jumping model. This leaves enough freedom in the final motion for the morphology of the robot to have an effect, which is in contrast to previous studies using precise tracking rewards. The improvements in robustness, hyperparameter sensitivity and data efficiency were shown across different optimization methods, namely OC, MPC and RL in reaching and hopping tasks that were learned on different models in two different simulators, MuJoCo and Demoa (Schmitt, 2022).

In contribution D, we have extended the previous simulation results to real hardware. Instead of designing an artificial muscle in hardware, we proposed to use software controllers in order to emulated muscle-like properties on a real robot. By using RL-controllers trained in a GPU-parallelized simulator in combination with sim-to-real techniques, we could quickly iterate and evaluate the behaviors induced by the muscle morphology.

The use of emulation circumvents the usual sim-to-real gap between the simulation and the real world when using non-linear actuators. It poses a different issue, however, which is the potential infeasibility of the simulated controller on the hardware. We solved this issue by identifying a mismatch between the applicable update frequency of the actuator and the maximal damping induced by the low-level controller as the

main reason for the mismatch. By proposing a damping rule, setting the maximum achievable damping of the muscle actuator to the equivalently achievable value of a pure damping controller $\tau = -D\dot{q}$, we are able to freely optimize the muscle morphology for task performance while keeping the controller feasible.

By investigating a quadruped robot in simulated running and hopping and real-world running tasks, we could reproduce previously observed increases in robustness of the muscle morphology. One pertinent issue in sim-to-real learning is that an under-specified task often results in behaviors which work well in simulation but will not transfer well to the real hardware, exploiting small movements and friction mismatches to the real world. This issue is usually mitigated by heavy reward engineering. Our muscle morphology, however, induced a behavioral bias in the learned movements which are naturally more transferable, even though only the task reward was optimized for.

These results are strong evidence for muscle actuators being well suited for a wide range of general-purpose tasks. They lead to quick performance increases in early training and perform well under a wide distribution of conditions. Nevertheless, torque actuators could be shown to achieve the best maximum performance in all tasks, when precisely tuned and trained until convergence. This is expected, as the idealized torque actuator can in principle achieve any possible torque configuration, while the muscle is restricted to a smaller manifold of behaviors.

When to use RL? This thesis has primarily investigated the application of RL to control generation with musculoskeletal models. The question arises, when should you use RL and when should you not?

Trajectory optimization (OC) approaches have been able to push the complexity of simulated MSK-models (Nitschke et al., 2020), but integrating feedback and environmental uncertainties into the mix requires further algorithmic advances (Koelewijn and van den Bogert, 2020) as OC is claimed to be inherently "unsuited" for feedback control (De Groote and Falisse, 2021). In contrast, MPC and reflex-based approaches naturally incorporate closed-loop feedback and environmental uncertainties, but have generally been applied to low-dimensional models and behaviors (Takagi et al., 2022; Wochner et al., 2020). RL can be used to train controllers for high-dimensional systems and can be used to distill a large amount of variations into the controller: Multi-task learning leads to more fundamental behaviors that generalize better and randomized perturbations and input noise lead to more robust controllers that can transfer to real systems. Most RL network architectures can also be used in real-time applications, once training has finished. Finally, RL can be used for controllers using partially-observable inputs and can make use of curriculum-based learning strategies, modeling the continual adaptation of behaviors (Abel et al., 2023).

RL does, however, not solve all problems. The flexibility of the optimization might allow for more complex behaviors and models to be used, but comes at the cost of interpretability and plausibility of the neuro-musculular control architecture. It has not been possible so far to combine RL with the full complexity of mono- and multi-synaptic reflex arcs, CPGs and appropriate sensory delays. Reflex-based controllers are a better approach for researchers interested in the exact components of behavior

generation, be it for general research or to gain insights into the mechanisms behind certain movement pathologies (Bruel et al., 2022). MPC and OC methods usually allow for better convergence with fewer data samples than RL. If the required constraints on muscle model fidelity do not allow the use of a very fast simulator, it might be infeasible to tune and train RL agents, especially if local optima pose a problem. We have found that RL often requires the use of surrogate reward functions, making it difficult to transfer existing knowledge on optimal cost criteria from the literature. Existing inverse dynamics approaches also converge much faster in tracking simulations, making the training of RL from scratch ill-advised when investigating singular recorded trajectories.

Should I make use of experimental data for behavior generation? Although the approaches introduced in this thesis are not explicitly making use of experimental data in the optimization, it would be incorrect to say that they do not make use of human experimental knowledge at all. All the used MSK-models contain model information derived from anatomical and behavioral experiments, even seemingly abstract reward terms were found in studies with human participants and the act of tuning a method until close-to-human behaviors can be observed implicitly embeds knowledge about human movement into the controller architecture. It is impossible to not make use of data at all.

There are, nevertheless, nuances. A reward function specifying precise movements by kinematically tracking a trajectory on the level of joint angles and velocities would easily reproduce human motion. The tracking method would likely not optimize the same metrics as human beings optimize, which can make it brittle with respect to neuro-muscular or environmental changes. Focusing on more abstract metrics, such as muscle effort, pain avoidance or limiting excessive forces, potentially allows for more movement diversity in training, producing for example a variety of walking, running and jumping behaviors, and to be exposed to different environmental and system changes, e.g. variations of the muscle strength or gains in reflex pathways.

The focus on this type of predictive simulation has allowed researchers to investigate the effect that various system changes have on behavior. Clemente, De Groote, and Dick (2024) investigate how size and body morphology in animals affect considerations of energy-efficient locomotion and standing posture for quadrupedal animals ranging from mice to elephants. In other predictive studies, researchers have quantified the influence of physiological changes on gait (Ong et al., 2019), the effects of assistive devices (Price, Umberger, and Sup, 2019) or balance strategies to external perturbations (Jones et al., 2024). These experiments are often infeasible or unethical to directly perform with humans and might open the way to advances in rehabilitation and pathology assessment. This thesis has primarily focused on forward simulation over inverse analysis, in order to provide tools that allow researchers to answer these questions with flexible closed-loop control policies and filling the performance gap to non-tracking RL-performance in the literature.

While predictive simulation without the use of tightly specified experimental data continues to be a valuable area of investigation, a new paradigm is surging in popularity. ML and RL methods become increasingly capable of distilling large-scale

datasets into single controllers or musculoskeletal models. While we have so far distinguished between tracking singular trajectories and optimizing behaviors from abstract reward terms, large-scale imitation and representation learning methods are capable of learning from tens of thousands of motions and condensing them into a single controller (Luo et al., 2023).

By leveraging the common representations that are formed in this controller, Luo et al. (2024b) were capable of extracting an action decoder which implicitly contains knowledge from the biggest publicly available motion capture dataset (Mahmood et al., 2019). If a new controller is now trained on an under-specified task, instead of directly predicting joint torques or muscle excitation signals, it can predict latent embedding which the action decoder projects into physical actions. As the decoder was trained on large-scale human data, the emerging behaviors are strongly biased towards motions occurring in the dataset. These new techniques have allowed flexible controllers to emerge which can be trained for predictive simulation while acting on distilled experiences of motion capture data, circumventing the need to specify an abstract reward term. Put in other words: Instead of designing objectives which are likely maximized by human beings, human behavior is implicitly embedded into representations or learned reward functions (Peng et al., 2021; Tirinzoni et al., 2024), resulting in universal motion controllers capable of producing diverse human motions under a wide range of different conditions.

It is exceedingly hard to be completely unbiased by experimental knowledge of human behavior. In our work, we have seen that a reduced reliance on direct tracking data can reveal subtle influences of the embodiment on learned behaviors, be it a high-dimensional MSK-model for which natural-like behaviors can be learned, or a simpler muscle model on a robot, which biases behaviors towards naturalness and improves sim-to-real transfer. Tracking single trajectories often offers better convergence properties (Lee et al., 2019) and allows for more complex behaviors than predictive simulation, revealing hard-to-obtain information about body-internal biological components, while also being a useful tool to create controllers for assistive devices (Luo et al., 2021; Luo et al., 2024a). Finally, the paradigm of large-scale motion data distillation for behavior generation might be an alternative that combines experimental data with predictive behavior synthesis, even though it has not yet been extended to MSK-systems and poses significant technical challenges.

Do muscles induce a beneficial bias for behavioral generation? Unfortunately, this question cannot be fully answered by this thesis. We have, however, observed in the presented works that controllers trained for muscle morphologies are generally robust and have a tendency to be biased behaviors towards naturalness. In the following we distinguish between tracking as a tightly specified task, allowing only for a particular target kinematic trajectory to achieve maximal task fulfillment, and under-specified tasks, where the target behavior is only loosely defined by a target velocity or height. The optimal solution space for under-specified tasks is less restricted than for tracking tasks, leaving more space for either the morphology or the optimization algorithm to affect the final trajectory.

The results presented in this thesis validate findings in previous studies. (Soest and Bobbert, 1993) compared muscle to torque actuators in an under-specified open-loop jumping task and found increased robustness with muscles; Increased stability has also been noted by other studies (Wagner and Blickhan, 1999; Eriten and Dankowicz, 2009; Yasuhiro Sugimoto and Tsuchiya, 2009; Haeufle, Grimmer, and Seyfarth, 2010a). Benefits in under-specified tasks were additionally found for morphological computation (Ghazi-Zahedi et al., 2016) and control effort (Haeufle et al., 2020b).

For human data tracking tasks, which we consider to be tightly specified, previous studies came to different conclusions. Barbera et al. (2021) proposed a MSK ostrich model, with which they learned tracking with torque and muscle actuation. It was found that the task is substantially easier with torque actuators. Peng and Panne (2017) compared different actuation principles—position-, torque- and muscle-control—in a tracking task and also found that the muscle actuator performed worse than the alternatives. Other studies using RL with tracking tasks claim that the control problem is easier if the dynamic tracking of skeletal DOF is optimized separately from the muscle excitation signal (Lee et al., 2019; Denizdurduran, Markram, and Gewaltig, 2022; Luo et al., 2024a).

We interpret these results in the following way. An idealized torque controller commanded by a sufficiently powerful optimization algorithm, has the largest possible space of *achievable* behaviors of any actuation type. Nevertheless, such a controller might, depending on the task conditions, either not find this optimal solution, be sensitive to parameter settings, environment uncertainties or external perturbations. It might also be difficult to define a sufficiently precise reward function in some applications, e.g. natural walking in diverse terrains, where it is difficult to collect a sufficient amount of data to numerically find a reward function or specify it by hand.

In contrast, muscle actuators pose functional dependencies and restrictions on the possible joint torque that can be generated at a given time. This evidently restricts the space of achievable behaviors by any controller. Nevertheless, muscle actuators seem to sometimes restrict behaviors to a subspace which is more natural, more robust to environment uncertainties and perturbations, and less sensitive with respect to learning parameters.

These improvements come, however, with a cost. Many control algorithms are not equipped to reliably optimize behaviors for muscle-driven systems. The high-dimensionality and non-linear properties of muscles are often difficult to handle by reflex-controllers, OC or MPC. Although RL excels at handling high-dimensional systems, complex behaviors, and environmental uncertainties, our experiments have shown that overactuation compounds the exploration challenges inherent to RL methods. This exacerbates the already documented tendency of RL-based optimization to converge to local optima more frequently than traditional control approaches (Gaudet, Linares, and Furfaro, 2018). Finally, muscle simulation can be computationally expensive, while artificial hardware muscles are difficult to design and induce even more control difficulties such as hysteresis and friction effects (Tri et al., 2010).

In the robotics applications discussed in this thesis, we opted to simplify the muscle models by omitting tendons and simulating only two muscles per DOF. This decision significantly eased the RL optimization process. Despite this simplification, we found

that torque actuators still outperformed muscles in scenarios where the environment dynamics were precisely known, the controller was highly optimized for the task, or when an overly restrictive maximum force behavior was required. Finally, muscle actuators introduce a larger number of free parameters than impedance or position controllers.

In conclusion, by evaluating the literature and our own experiments, we find that muscle actuators show strong potential in under-specified tasks. They can lead to more robust solutions and an overall easier optimization procedure, while shaping the learned behaviors to be more natural and generalize more likely to the real world. These benefits can, however, not be realized when the used control algorithm cannot handle the strong overactuation of the muscle system. Additionally, parametrizing and prototyping muscle actuators can be difficult, as there are no clear tuning procedures like for more established controllers (Ziegler and Nichols, 2022). This difficulty is especially relevant when designing actuators for the real world.

In contribution D, we found that software emulation can mitigate some of these issues, by allowing optimization and prototyping to move to simulation, before deploying it on the hardware. The best performance is likely to be achieved with specialized hardware or low-level controllers integrated directly into the robot firmware, allowing for much higher control frequencies and stability than a flexible software interface. Finally, specific benefits such as ultra-short delay reactions or energy efficiency are unlikely to be ever be realized in software actuators. See Fig. 2.4, which shows the effect of control frequency on stability for a software emulated muscle. Real hardware with spring-like properties can react almost instantaneously and store and release energy, which is not possible with purely electric actuators (Mo et al., 2023).

Which simulator should I use? All models are wrong, but some are useful. The choice of simulator should not be an ideological question, but a factual one. The right model is the one that reproduces the phenomenon we want to study to a useful degree, while being as computationally or theoretically simple as possible. Following these guidelines ultimately allows researchers to simulate the model of interest while leveraging fast iteration times and scalability (Uhlrich et al., 2023).

MuJoCo simplifies the Hill-type muscle model with the assumption of a rigid tendon. While not relevant for all muscles in the body and not for all behaviors (Millard et al., 2013; Rajagopal et al., 2016a), many studies have found strong contributions by elastic muscle-tendon dynamics in agile locomotion tasks (Alexander, 2002) as well as relevance to energy conservation in steady walking (Ishikawa et al., 2005). If a researcher is interested in exact muscle physiology, such as specific maximum tendon forces under extreme conditions, e.g. car crashes (Nölle et al., 2024), or highly dynamic sports (Bulat et al., 2019), MuJoCo is unlikely to be the right choice of simulation software.

In contrast, the model assumptions made by HyFyDy supposedly stay close to OpenSim, which is considered the gold-standard in the field. These simulators have been validated by a large amount of experimental studies and are widely trusted by the community. Nevertheless, the Hill-type muscle model has also been shown to not be numerically stable or accurate in many scenarios (Yeo et al., 2023; Millard et al.,

2024) and the used human models are a result of an amalgamation of studies with adult cadavers and healthy humans (Rajagopal et al., 2016b), being averaged across a number of anatomical experiments. Motor control studies focusing on specific sub-populations, such as overweight individuals, specific pathologies, children or ethnic groups with distinct anatomical differences are lacking.

MuJoCo, being primarily a robotics simulator, benefits of other technical features: fast execution speeds, GPU-parallelizability, stability under extreme conditions and contact models capable of simulating mesh-based contacts. This allows the community to focus more on research with robotic components like active prosthetics and exoskeletons, where controllers capable of sim-to-real transfer require large amounts of data.

In this thesis, we did not advance simulation methods, but we developed control methods and studied the influence of morphology on behaviors. As such, we needed to propose control algorithms which function under different environment conditions, while still being able to optimize the task at hand. In pursuing this goal, we have mainly used and developed algorithms which are simulator agnostic. DEP-RL and the extension to natural motion through an adaptive reward term work without further modification across HyFyDy, MuJoCo and OpenSim, while PPO, another RL algorithm we used, was applied to MuJoCo and IsaacGym (Makoviychuk et al., 2021).

Throughout these experiments, we have observed that robotic simulators, such as MuJoCo and IsaacGym, tend to create behaviors that are less smooth than the musculoskeletal simulators. Behaviors trained in HyFyDy and OpenSim have largely exhibited less jerk and achieve a closer match to experimental human data with identical reward functions as used in MuJoCo. While it is likely that the biomechanical modeling assumptions in HyFyDy, especially the elastic tendon, produce a closer match to human muscle dynamics than MuJoCo, we could not validate this assumption so far with respect to RL. Experiments with rigid tendons in HyFyDy, as well as softer and more strongly damped contact and joint limit parameters in MuJoCo could not account for the observed differences. It is likely that the more natural behaviors in HyFyDy are not only caused by the higher fidelity muscle models but also by favorably tuned parameters and possibly other simulation engine choices. Lastly, the MyoLeg model in MuJoCo is based on a different underlying OpenSim model which contains features such as moveable patellas with routed tendons, making control possibly more difficult.

Summarizing, the following recommendation can be made:

- Use the simulator that models the component you care about. Robotics simulators might not be well suited for the modeling of injury thresholds, but might excel when investigating human-robot interaction.
- Go for the fastest simulator that you can use. Increases in computational efficiency have changed how controllers can be trained and fast iteration times are conductive to research advances. A flexible method can be applied to a slow but more precise simulator after it has left the prototyping stage.
- Use an open-source simulation engine whenever possible. Not only are they free, they allow you to exactly understand what is going on in your experiment.

- If possible, try to make sure your model can actually perform the movement you want it to perform. An imitation learning algorithm will never converge if there is a mismatch between model and data on a kinematic or a dynamic level.

The quest for better and faster simulation engines has opened up a host of possibilities for new algorithms and applications. It is, however, the responsibility of the research community to question and validate the underlying modeling assumptions, and of the individual researcher to chose the right software for their application. Lastly, by focusing efforts on flexible methods and learning-driven techniques, such as RL, it is possible to develop simulator-agnostic algorithms which work under a range of different conditions.

Outlook This thesis has shown that RL methods can scale to high-dimensional MSK-models and produce highly-complex as well as close-to-natural and robust behaviors while working reliably across different simulators. These control algorithms have been developed with ease-of-use in mind: they perform well across different tasks and models without the need to tune hyperparameters. As such, they constitute a strong starting point for motor control and biomechanics researchers intending to leverage RL for their own studies.

Our work paves the way for researchers to push predictive human simulations beyond laboratory conditions and simple periodic behaviors. Our control approaches allow the investigation of human compensation strategies or adaptations to muscular pathologies with complex 3D models under varied conditions, such as irregular terrain, without relying on the tracking of experimental data (Hellera et al., 2001) or on simplified models (Song and Geyer, 2017; Lassmann et al., 2023).

While this achievement was made possible by relying on abstract, biologically-inspired reward terms, the potential of large-scale imitation learning and subsequent controller distillation has only recently begun to be explored (Tirinzoni et al., 2024). Extending these techniques to MSK-systems might allow researchers to push the behavioral complexity even further, while still being able to generate trajectories under unseen system and environmental changes. Indeed, computer animation approaches have already started to incorporate muscle-components with universal motion generators (Park et al., 2022; Feng, Xu, and Liu, 2023), but the utility for biomechanics and motor control research is still missing due to strong simplifications of the used models. Contributions made in this thesis, allowing general-purpose control without muscle model simplifications, might close this gap.

As a flexible closed-loop control method, RL enables the investigation of behavior generation under biologically realistic inputs, such as partial observability and delays. MPC and OC have been partially extended to consider uncertainties, but require an intricate mathematical framework and precise definitions of them. RL's adaptability allows it to learn with respect to a wide range of conditions, even when factors are hard to model. Reflex controllers can do a bit of this, but they are constrained by their limited behavioral and model complexity and have a narrower region of robustness.

For assistive devices, RL already allows the creation of virtual humans which can be used to train controllers for real-world exoskeletons entirely in simulation (Luo et al.,

2021; Luo et al., 2024a). This is a major breakthrough, as it enables more powerful and versatile controllers and reduces the need for human experiments during development. In addition, RL methods have shown to be able to tune existing prosthetic devices online in order to personalize parameters for a specific test subject. Nevertheless, these studies rely on simplified muscle models and control assumptions, potentially limiting the predictive capabilities of the simulated behaviors. Our contributions might allow researchers to further push the complexity and diversity of learned behaviors with virtual humans without relying on precise tracking simulation and without reducing biomechanical fidelity.

Considering our contributions for robot actuation, we have shown and validated the benefits of muscle actuation in simulation and on a real robot in contributions C and D. While existing controllers deployed on robot hardware using artificial muscles often struggle with good performance, due to a mix of possibly beneficial and detrimental qualities, this thesis shows that muscle actuators can also improve learning and robustness with RL. These results were, however, obtained only on quadruped hardware and are highly dependent on the specific task and model parameters. We believe future research should investigate extensions to our approach by applying it to bipedal robots and potentially including elastic tendons, which have been shown to be beneficial for human locomotion (McNeill Alexander, 2002; Zelik et al., 2014). Muscle-tendon actuators could potentially improve the performance of bipedal robots, which are currently relying on less force sensitive position control, thereby neglecting the finer control of their ankle joints. Ankle joints are often not present in quadruped robots and have been largely ignored by the learning community.

Considering actuator emulation in the more general sense, we believe that the interdependence between actuator development and control research in robot learning presents a significant bottleneck. Researchers typically specialize in either hardware development or control strategies, leading to scenarios where either novel hardware is combined with tried-and-tested control methods (Azocar et al., 2020; Lee, Liu, and Huang, 2024), or advances in control are tested with widely available and affordable hardware (Bellegarda et al., 2022). This segregation limits opportunities for co-optimization, slowing down advancements in both fields. Existing studies that investigate morphology adaptation at the same time as controllers are often done entirely in simulation (Deimel et al., 2017; Wang et al., 2023). While providing useful insight into robot design, these studies ignore real world effects that are not well modelled in simulators, such as friction, contacts and elastic components. Our proposed software emulation and learning framework addresses this challenge by enabling simultaneous prototyping and optimization of hardware and control systems.

Appendix A

DEP-RL: Embodied Exploration for Reinforcement Learning in Overactuated and Musculoskeletal Systems

DEP-RL: EMBODIED EXPLORATION FOR REINFORCEMENT LEARNING IN OVERACTUATED AND MUSCULOSKELETAL SYSTEMS

Pierre Schumacher[1,2] **Daniel F.B. Haeufle**[2,3] **Dieter Büchler**[1] **Syn Schmitt**[3] **Georg Martius**[1]
[1]Max Planck Institute for Intelligent Systems, Tübingen, Germany
[2]Hertie-Institute for Clinical Brain Research, Tübingen, Germany
[3]Institute for Modelling and Simulation of Biomechanical Systems, Stuttgart, Germany

ABSTRACT

Muscle-actuated organisms are capable of learning an unparalleled diversity of dexterous movements despite their vast amount of muscles. Reinforcement learning (RL) on large musculoskeletal models, however, has not been able to show similar performance. We conjecture that ineffective exploration in large overactuated action spaces is a key problem. This is supported by our finding that common exploration noise strategies are inadequate in synthetic examples of overactuated systems. We identify differential extrinsic plasticity (DEP), a method from the domain of self-organization, as being able to induce state-space covering exploration within seconds of interaction. By integrating DEP into RL, we achieve fast learning of reaching and locomotion in musculoskeletal systems, outperforming current approaches in all considered tasks in sample efficiency and robustness.[1]

Figure 1: **We achieve robust control on a series of overactuated environments**. Left to right: torquearm, arm26, humanreacher, ostrich-foraging, ostrich-run, human-run, human-hop

1 INTRODUCTION

It is remarkable how biological organisms effectively learn to achieve robust and adaptive behavior despite their largely overactuated setting—with many more muscles than degrees of freedom. As Reinforcement Learning (RL) is arguably a biological strategy (Niv, 2009), it could be a valuable tool to understand **how** such behavior can be achieved, however, the performance of current RL algorithms has been severely lacking so far (Song et al., 2021).

One pertinent issue since the conception of RL is how to efficiently explore the state space (Sutton & Barto, 2018). Techniques like ϵ-greedy or zero-mean uncorrelated Gaussian noise have dominated most applications due to their simplicity and effectiveness. While some work has focused on exploration based on **temporally** correlated noise (Uhlenbeck & Ornstein, 1930; Pinneri et al., 2020), learning tasks from scratch which require correlation **across** actions have seen much less attention. We therefore investigate different exploration noise paradigms on systems with largely overactuated action spaces.

The problem we aim to solve is the generation of motion through numerous redundant muscles. The natural antagonistic actuator arrangement requires a correlated stimulation of agonist and antagonist muscles to avoid canceling of forces and to enable substantial motion. Additionally, torques generated by short muscle twitches are often not sufficient to induce adequate motions on the joint level due to

[1]See https://sites.google.com/view/dep-rl for videos and code.

chemical low-pass filter properties (Rockenfeller et al., 2015). Lastly, the sheer number of muscles in complex architectures (humans have more than 600 skeletal muscles) constitutes a combinatorial explosion unseen in most RL tasks. Altogether, these properties render sparse reward tasks extremely difficult and create local optima in weakly constrained tasks with dense rewards (Song et al., 2021).

Consequently, many applications of RL to musculoskeletal systems have only been tractable under substantial simplifications. Most studies investigate low-dimensional systems (Tahami et al., 2014; Crowder et al., 2021) or simplify the control problem by only considering a few muscles (Joos et al., 2020; Fischer et al., 2021). Others, first extract muscle synergies (Diamond & Holland, 2014), a concept closely related to motion primitives, or learn a torque-stimulation mapping (Luo et al., 2021) before deploying RL methods. In contrast to those works, we propose a novel method to learn control of high-dimensional and largely overactuated systems on the muscle stimulation level. Most importantly, we avoid simplifications that reduce the effective number of actions or facilitate the learning problem, such as shaped reward functions or learning from demonstrations.

In this setting, we study selected exploration noise techniques and identify differential extrinsic plasticity (DEP) (Der & Martius, 2015) to be capable of producing effective exploration for muscle-driven systems. While originally introduced in the domain of self-organization, we show that DEP creates strongly correlated stimulation patterns tuned to the particular embodiment of the system at hand. It is able to recruit muscle groups effecting large joint-space motions in only seconds of interaction and with minimal prior knowledge. In contrast to other approaches which employ explicit information about the particular muscle geometry at hand, e.g. knowledge about structural control layers or hand-designed correlation matrices (Driess et al., 2018; Walter et al., 2021), we only introduce prior information on which muscle length is contracted by which control signal in the form of an identity matrix. We first empirically demonstrate DEP's properties in comparison to popular exploration noise processes before we integrate it into RL methods. The resulting DEP-RL controller is able to outperform current approaches on unsolved reaching (Fischer et al., 2021) and running tasks (Barbera et al., 2021) involving up to 120 muscles.

Contribution (1) We show that overactuated systems require noise correlated across actions for effective exploration. (2) We identify the DEP (Der & Martius, 2015) controller, known from the field of self-organizing behavior, to generate more effective exploration than other commonly used noise processes. This holds for a synthetic overactuated system and for muscle-driven control—our application area of interest. (3) We introduce repeatedly alternating between the RL policy and DEP within an episode as an efficient learning strategy. (4) We demonstrate that DEP-RL is more robust in three locomotion tasks under out-of-distribution (OOD) perturbations.

To our knowledge, we are the first to control the 7 degrees of freedom (DoF) human arm model (Saul et al., 2015) with RL on a muscle stimulation level—that is with 50 individually controllable muscle actuators. We also achieve the highest ever measured top speed on the simulated ostrich (Barbera et al., 2021) with 120 muscles using RL without reward shaping, curriculum learning, or expert demonstrations.

2 Related works

Muscle control with RL Many works that apply RL to muscular control tasks investigate low-dimensional setups (Tieck et al., 2018; Tahami et al., 2014; Crowder et al., 2021) or manually group muscles (Joos et al., 2020) to simplify learning. Fischer et al. (2021) use the same 7 DoF arm as we do, but simplify control by directly specifying joint torques $a \in \mathbb{R}^7$ and only add activation dynamics and motor noise. Most complex architectures are either controlled by trajectory optimization approaches (Al Borno et al., 2020) or make use of motion capture data (Lee et al., 2019). Barbera et al. (2021) also only achieved a realistic gait with the use of demonstrations from real ostriches; learning from scratch resulted in a slow policy moving in small jumps. Some studies achieved motions on real muscular robots(Driess et al., 2018; Buchler et al., 2016), but were limited to simple morphologies and small numbers of muscles.

NeurIPS challenges Multiple challenges on musculoskeletal control (Kidziński et al., 2018; Kidziński et al., 2019; Song et al., 2021) using OpenSim (Delp et al., 2007) have been held. The top-ten submissions from Kidziński et al. (2018) resorted to complex ensemble architectures and made use of parameter- or OU-noise. High-scoring solutions in (Kidziński et al., 2019) were commonly using explicit reward shaping, demonstrations, or complex training curricula with selected checkpoints,

all of which required extensive hand-crafting. In contrast, our RL agent uses a standard two-layer architecture, no curriculum, no reward shaping, and no demonstrations.

Large action spaces Some studies (Farquhar et al., 2020; Synnaeve et al., 2019) tackle large action spaces by growing them iteratively from smaller versions. This would, however, require a priori knowledge of which muscle groups correspond to each other—which DEP learns by itself. Tavakoli et al. (2021) present a hypergraph-based architecture that scales in principle to continuous action spaces. Again, as it is not clear *which* muscles should be grouped, the number of possible hypergraphs is intractable. Other works (Dulac-Arnold et al., 2016; Wang et al., 2016) deal with large virtual action spaces, but only for discrete actions. Finally, studies on action priors(Biza et al., 2021; Singh et al., 2021) learn initially from expert data to bootstrap policy learning. Our method scales to large continuous action spaces and does not require demonstrations.

Action space reduction with muscles Several works use architectures reducing the control dimensionality before deploying RL methods. Luo et al. (2021) learn a muscle-coordination network that maps desired joint torques to muscle excitations to act on PD controllers. Jiang et al. (2019) establish a correction network that imposes output limits on torque actuators, enabling them to reproduce muscle behavior. However, these methods require specific data collection strategies and prior knowledge on joint ranges, forces or desired behaviors, before applying RL. While they simplify learning and enable large-scale control, our approach is simpler in execution, does not require knowledge about the specific muscular architecture and works with simulators of varying detail, including *elastic* tendons. Some works use PCA or other techniques to extract synergies and learn on them (Al Borno et al., 2020; Zhao et al., 2022), but they either require human data or expert demonstrations. Our approach can readily be applied to a large range of systems with only minimal tuning.

3 BACKGROUND

Reinforcement learning We consider discounted episodic Markov Decision Processes (MDP) $\mathcal{M} = (\mathcal{S}, \mathcal{A}, r, p, p_0, \gamma)$, where $\mathcal{A}$ is the action space, $\mathcal{S}$ the state space, $r : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is the reward function, $p(s'|s, a)$ the transition probability, $p_0(s)$ the initial state distribution and γ is the discount factor. The objective is to learn a policy $\pi(a|s)$ that maximizes the expected discounted return $J = \mathbb{E}_\pi \sum_t \gamma^t r_t$, where r_t are observed when rolling out π. For goal-reaching tasks, we define a goal space $\mathcal{G}$ with $r(s, a, g)$ being 0 when s reached the goal g and -1 everywhere else. The policy is then also conditioned on the goal $\pi^g(a|s, g)$ following Schaul et al. (2015).

Muscle modeling In all our MuJoCo experiments, we use the MuJoCo internal muscle model, which approximates muscle characteristics in a simplified way that is computationally efficient but still reproduces many significant muscle properties. One limitation consists in the non-elasticity of the tendon. This reduces computation time immensely, but also renders the task fully observable, as muscle length feedback now uniquely maps to the joint configuration. Experiments with a planar humanoid were conducted in HyFyDy (Geijtenbeek, 2019; 2021), a fast biomechanics simulator that contains elastic tendons and achieves similar accuracy to the widely used OpenSim software. See Suppl. A.4 for details on the MuJoCo muscle model.

Limitations of uncorrelated noise in (vastly) overactuated systems The defining property of an overactuated system is that the number of actuators exceeds the number of degrees of freedom. Let us consider a hypothetical 1 DoF system, where we replace the single actuator with maximum force F^M by n actuators, each contributing maximally with force F^M/n. The force for a particular actuator is then $F_i = (F^M/n)\, f_i$, with $f_i \in [-1, 1]$, while the total force is $F = \sum_i^n F_i = \sum_i^n \hat{F}_i/n$. We define $\hat{F}_i = F^M\, f_i$ to make the dependence on n clear. What happens if we apply random noise to such a system? The variance of the resulting torque is:

$$\mathrm{Var}\left(\sum_{i=1}^{n} \frac{\hat{F}_i}{n}\right) = \frac{1}{n^2} \sum_{i=1}^{n} \mathrm{Var}\left(\hat{F}_i\right) + \frac{1}{n^2} \sum_{i \neq j} \mathrm{Cov}(\hat{F}_i, \hat{F}_j). \tag{1}$$

For i.i.d. noise with fixed variance $\mathrm{Var}(\hat{F}_i) = \sigma^2$, as typically used in RL, the first term decreases with $1/n$ and the second term is zero. Thus, for large n the effective variance will approach zero. The logical solution would be to increase the variance of each actuator, but as no realistic actuator can output an arbitrarily large force, the maximum achievable variance is bounded. Clearly, a vanishing effective variance cannot result in adequate exploration. For correlated noise, the second term in

Eq. 1 has $n^2 - n$ terms and decays with $1 - 1/n$, so it can avoid vanishing and might be used to increase the effective variance.

4 METHODS

4.1 DIFFERENTIAL EXTRINSIC PLASTICITY (DEP)

Hebbian learning (HL) rules are widely adopted in neuroscience. They change the strength of a connection between neurons proportional to their mutual activity. However, when applied in a control network that connects sensors to actuators, all activity is generated by the network itself and thus the effect of the environment is largely ignored. In contrast, differential extrinsic plasticity (DEP), a learning rule capable of self-organizing complex systems and generating coherent behaviors (Der & Martius, 2015), takes the environment into the loop. Consider a controller:

$$a_t = \tanh(\kappa C s_t + h_t), \tag{2}$$

with the action $a_t \in \mathbb{R}^m$, the state $s_t \in \mathbb{R}^n$, a learned control matrix $C \in \mathbb{R}^{m \times n}$, a time-dependent bias $h_t \in \mathbb{R}^n$ and an amplification constant $\kappa \in \mathbb{R}$. The DEP state is only constituted of sensors related to the actuators, in contrast to the more general RL state. We only consider the initialization $C_{ij} = 0$. To be able to react to the environment, an update rule is required for the control matrix C. A Hebbian rule proposes $\dot{C}_{ij} \propto a_{i,t}\, s_{j,t}$, while a differential Hebbian rule might be $\dot{C}_{ij} \propto \dot{a}_{i,t}\, \dot{s}_{j,t}$. The differential rule changes C according to changes in the sensors and actions, but there is no connection to the consequences of actions, as only information of the *current* time step is used.

DEP proposes several changes: (1) There is a causal connection to the *future* time step $\dot{C} \propto f(\dot{s}_t)\dot{s}_{t-\Delta t}^\top$, where $f(\dot{s}_t) = a_{t-\Delta t}$ is an inverse prediction model and Δt controls the time scale of learned behaviors. (2) A normalization scheme for C adjusts relative magnitudes to retain strong activity in all regions of the state space $\tilde{C}_{ij} = C_{ij}/(||C_{ij}||_i + \epsilon)$, with $\epsilon \ll 1$. (3) The time-dependent bias term $\dot{h}_t \propto -a_t$ prevents overly strong actions from keeping the system at the joint limits with $\dot{s}_t = 0$. This learning rule becomes $\tau\dot{C} = f(\dot{s}_t)\dot{s}_{t-\Delta t}^\top - C$: the first term drives changes while the second one is a first-order decay term. The factor τ tunes the time scale of changing the controller parameters C. The normalization rescales C at each time step. The velocities of any variable $\dot{x}_t$ are approximated by $\dot{x}_t = x_t - x_{t-1}$.

The inverse prediction model f relates observed changes in s to corresponding changes in actions a that could have led to those changes. In the case where each actuator directly influences one sensor through the consequences of its action, the simple identity model can be used, similar to prior work (Der & Martius, 2015; Pinneri & Martius, 2018): $f(\dot{s}_t) \coloneqq \dot{s}_t$. This yields the rule:

$$\tau\dot{C} = \dot{s}_t\dot{s}_{t-\Delta t}^\top - C. \tag{3}$$

As a direct consequence, the update of C is driven by the **velocity correlation matrix** of the current and the previous state. This means that velocity correlations between state-dimensions that happen across time, for instance, due to physical coupling, are amplified by strengthening of corresponding connections between states and actions in Eq. 2. The choice of f implicitly embeds knowledge of which sensor is connected to which actuator and is assuming an approximately linear relationship. This knowledge is easily available for technical actuators. In muscle-driven systems, such a relationship holds for sensors measuring muscle length with $f(\dot{s}_t) \coloneqq -\dot{s}_t$, as muscles contract with increasing excitations. If there are more sensors per actuator, f can either be a rectangular matrix or a more complex model that links proprioceptive sensors to actuators.

Pinneri & Martius (2018) has shown that in low-dimensional systems DEP can converge to different stationary behaviors. Which behavior is reached is highly sensitive to small perturbations. We observe that with high-dimensional and chaotic environments, the stochasticity injected by the randomization after episode resets is sufficient to prevent a behavioral deprivation. In practice, we use DEP in combination with an RL policy, such that the interplay will force DEP out of specific limit cycles, similar to the interventions of the human operator in Martius et al. (2016). Intuitively, DEP is able to quickly coerce a system into highly correlated motions by "chaining together what changes together". More details can be found in Suppl. B.3 and in Der & Martius (2015), while Suppl. C.7 contains an application of DEP to a simple 1D-system elucidating the principal mechanism.

4.2 INTEGRATING DEP AS EXPLORATION IN REINFORCEMENT LEARNING (DEP-RL)

Integrating the rapidly changing policy of DEP into RL algorithms requires some considerations. DEP is an independent network and follows its own gradient-free update rule, which cannot be easily integrated into gradient-based RL training. Summing actions of DEP and the RL policy, as with conventional exploration noise, leads in our experience to chaotic behavior, hindering learning (Suppl. C.2). Therefore, we consider a different strategy: The DEP policy is taking over control completely for a certain time interval, similar to intra-episode exploration (Pislar et al., 2022); RL policy π and exploration policy Eq. 2 are alternating in controlling the system. While we tested many different integrations (Suppl. B.3 and C.2), we here use the following components:

Intra-episode exploration DEP and RL alternate within each episode according to a stochastic switching procedure. After each policy sampling, DEP takes over with probability p_{switch}. Actions are then computed using DEP for a fixed time horizon H_{DEP}, before we alternate back to the policy. In this way, the policy may already approach a goal before DEP creates unsupervised exploration.

Initial exploration As DEP is creating exploration that excites the system into various modes, we suggest running an unsupervised pre-training phase with exclusive DEP control. The data collected during this phase is used to pre-fill the buffer for bootstrapping the learning progress in off-policy RL.

5 EXPERIMENTS

We first conduct synthetic experiments that investigate the exploration capabilities of colored noise (Timmer & König, 1995; Pinneri et al., 2020) (see Suppl. A.2 and A.3), Ornstein-Uhlenbeck (OU) noise (Uhlenbeck & Ornstein, 1930), and DEP by measuring state-space coverage in torque- and muscle-variants of a synthetically overactuated planar arm. OU-noise, in particular, was a common choice in previous muscular control challenges (Song et al., 2021; Kidziński et al., 2018; Kidziński et al., 2019). Afterwards, DEP-RL is applied to the same setup to assure that the previous findings are relevant for learning. Finally, we apply DEP-RL to challenging reaching and locomotion tasks, among which humanreacher and ostrich-run have not been solved with adequate performance so far. Even though DEP-RL could be combined with any off-policy RL algorithm, we choose MPO (Abdolmaleki et al., 2018) as our learning algorithm and will refer to the integration as DEP-MPO from now on. See Suppl. B.1 for details. We use 10 random seeds for each experiment if not stated otherwise.

5.1 ENVIRONMENTS

Ostrich-run and -foraging are taken from Barbera et al. (2021), while all of its variants and all other tasks, including specifications of state spaces and reward functions, are constructed by us from existing geometrical models. We use SCONE (Geijtenbeek, 2019; 2021) for the bipedal human tasks and MuJoCo (Ikkala & Hämäläinen, 2022; Todorov et al., 2012) for the arm-reaching tasks. See Suppl. B.2 for details.

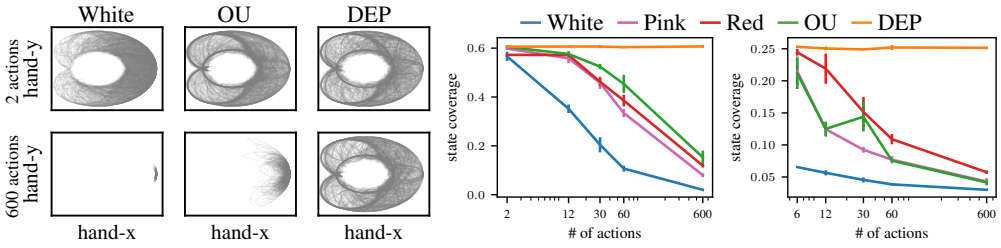
torquearm A 2-DoF planar arm that moves in a 2D plane and is actuated by 2 torque generators.
arm26 torquearm driven by 6 muscles. The agent has to reach random goals with sparse rewards.
humanreacher A 7-DoF arm that moves in full 3D. It is actuated by 50 muscles. The agent has to reach goals that randomly appear in front of it at "face"-height. The reward function is sparse.
ostrich-run A bipedal ostrich needs to run as fast as possible in a horizontal line and is only provided a weakly-constraining reward in form of the velocity of its center of mass, projected onto the x-axis. Only provided with this generic reward and without motion capture data, a learning agent is prone to local optima. The bird possesses 120 individually controllable muscles and moves in full 3D.
ostrich-foraging An ostrich neck and head actuated by 52 muscles need to reach randomly appearing goals with the beak. We changed the reward from the original environment to be sparse.
ostrich-stepdown Variant of ostrich-run which involves a single step several meters from the start. The height is adjustable and the position of the step randomly varies with $\Delta x \sim \mathcal{N}(\Delta x|0, 0.2)$. The task is successful if the ostrich manages to run for ≈ 10 meters past the step. The reward is sparse.
ostrich-slopetrotter The ostrich runs across a series of half-sloped steps. Conventional steps would disadvantage gaits with small foot clearance, while the half-slope allows most gaits to move up the step without getting stuck. The task reward is the achieved distance, given at the end.
human-run An 18-muscle planar human runs in a straight line as fast as possible. The reward is the COM-velocity. The model is the same as in the NeurIPS challenge (Kidziński et al., 2018).

human-hop The reward equals 1 if the human's COM-height exceeds 1.08 m, otherwise it is 0.
human-stepdown The human runs across slopes before encountering a step of varying height.
human-hopstacle The human has to jump across two slanted planes and a large drop without falling.

We observed a critical bug in ostrich-run which we fixed for our experiments. This explains the differing results for the baseline from Barbera et al. (2021) in Sec. 5.4. See Suppl. B.2 for details.

5.2 Exploration with overactuated systems

Primarily, we are interested in efficient control of muscle-driven systems. These systems exhibit a large degree of action redundancy—in addition to a myriad of nonlinear characteristics. Thus, we create an artificial setup allowing us to study exploration of overactuated systems in isolation.



(a) torquearm trajectories (b) torquearm state coverage (c) arm26 state coverage

Figure 2: **Only DEP reaches adequate state-space coverage for all considered action spaces.** (a) Hand trajectories are collected during 50 episodes of 1000 iterations ($\Delta t = 10$ ms) of pure exploration with different noise strategies. We show hand trajectories for the original action space $a \in \mathbb{R}^2$ (top) and the expanded action space $a \in \mathbb{R}^{600}$ (bottom). (b) Endeffector-space coverage for torquearm. (c) Endeffector-space coverage for arm26.

Overactuated torque-driven system In the default case, the robot (Fig. 1: torquearm) can be controlled by specifying desired joint torques $a \in \mathbb{R}^2$. We now artificially introduce an action space multiplier n, such that the control input grows to $2n$. To apply this new control vector to the original system, we average the actions into the original dimensionality, keeping maximal torques identical:

$$a_k = \frac{1}{n} \sum_{j=1}^{n} \hat{a}_{j+n(k-1)}, \tag{4}$$

where $\hat{a}_j$ is the inflated action and $k \in [1, 2]$ for our system. We emphasize that we not only have more actions but also capture important characteristics of an overactuated system, the redundancy. As predicted in Sec. 3, we observe that redundant actuators decrease the effective exploration when using uncorrelated Gaussian (white) noise (compare 2 vs. 600 actions in Fig. 2 (a, b)). Also, for correlated pink and red colored noise or OU-noise, the exploration decays with an increasing number of actions. Only DEP covers the full endeffector space for all setups. See Suppl. A.1 for details on the used coverage metric and Suppl. C.1 for more visualizations.

Overactuated muscle-driven system Consider now a system with individually controlled muscle actuators (Fig. 1: arm26). This architecture is already overactuated as multiple muscles are connected to each joint, the two biarticular muscles (green) even span two joints at the same time. In addition, we apply Eq. 4 to create virtual redundant action spaces. In Figure 2 (c), we see that most noise processes perform even worse than in the previous example, even though the number of actions is identical. Only DEP again reaches full endeffector-space coverage for any investigated number of actions. These results suggest that the *heterogeneous* redundant structure and activation dynamics of muscle-actuators require exploration strategies correlated across time and across actions to induce meaningful endeffector-space coverage, which DEP can generate.

We emphasize that all noise strategies for experiments in Sec. 5.2 were individually optimized to produce maximum sample-entropy for **each** system and for **each** action space, while DEP was tuned only **once** to maximize the joint-space entropy of the humanreacher environment. We additionally observe that all strategies consequently produce outputs close to the boundaries of the action space, known as bang-bang control, maximizing the variance of the resulting joint torque (Sec. 3).

RL with muscles While the previous results demonstrate that the exploration issue is significant, it is imperative to demonstrate the effect in a learning scenario. We therefore train an RL agent for differently sized action spaces and compare its performance to our DEP-MPO. Figure 3 shows that the performance of the MPO agent strongly decreases with the number of available actions, while the DEP-MPO agent performs well even with 600 available actions, which is the approximate number of muscles in the human body. DEP-MPO strongly benefits from the improved exploration of DEP. We repeated the experiment with sparse rewards and HER in Suppl. C.6, the results are almost identical.

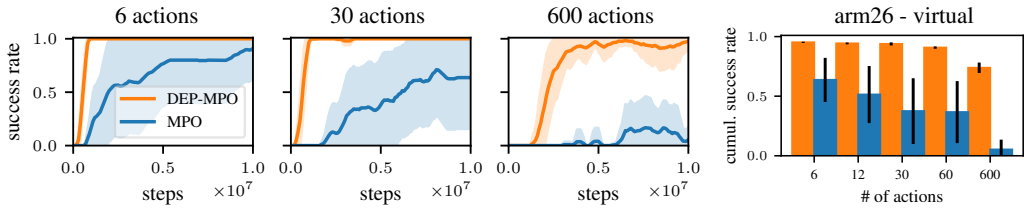


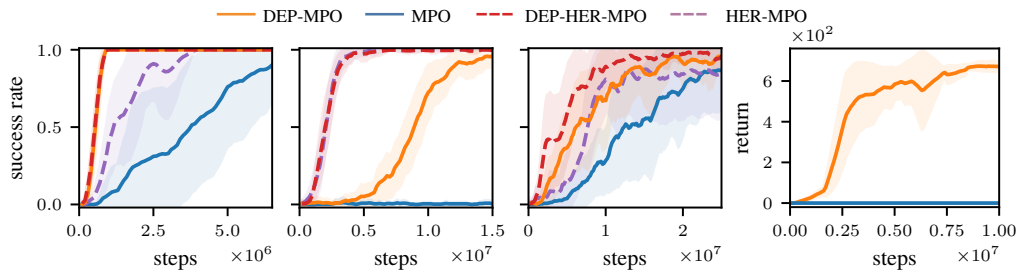
Figure 3: **DEP-MPO outperforms MPO in sparse point-reaching for arm26 with all virtual action spaces.** Left: Learning performance decays with a growing number of actions for MPO, DEP-MPO is largely unaffected. Right: Training-averaged success rates for different action spaces.

5.3 Sparse reward tasks with up to 52 actuators

We now apply our algorithm to realistic control scenarios with large numbers of muscles. As many sparse reward goal-reaching tasks are considered in this section, we choose Hindsight Experience Replay (HER) (Andrychowicz et al., 2017; Crowder et al., 2021) as a natural baseline. Generally, DEP-MPO performs much better than vanilla MPO (Fig. 4). For the more challenging environments, the combination of DEP with HER yields the best results.

DEP-MPO also solves the human-hop task, which is not goal-conditioned. As the reward is only given for exceeding a threshold COM-height, MPO **never** encounters a non-zero reward.

To the best of our knowledge, we are the first to control the 7-DoF human arm with RL on a muscle stimulation level—that is with 50 individually controllable muscle actuators. In contrast, (Fischer et al., 2021) only added activation dynamics and motor noise to 7 torque-controlled joints $a \in \mathbb{R}^7$.



(a) arm26 (b) ostrich-foraging (c) humanreacher (d) human-hop

Figure 4: **Training performance for all sparse tasks.** (a) DEP-MPO and DEP-HER-MPO quickly solve the task. (b) While DEP-HER-MPO performs on par with HER-MPO, DEP-MPO allows the agent to solve the task in contrast to MPO. (c) DEP-MPO achieves better data-efficiency over MPO, while the best agent is DEP-HER-MPO. We conjecture that HER enables more effective use of the unsupervised data. (d) DEP-MPO finds sparse rewards even in the absence of goal-conditioning, which makes the application of HER infeasible. MPO does not encounter any non-zero reward.

5.4 Application to bipedal locomotion

While the results above show that DEP-RL performs well on several reaching tasks where a complete state-space coverage is desirable, it is unclear how it handles locomotion. Useful running gaits only occupy a minuscule volume in the behavioral space and unstructured exploration leads to the agent falling down very often. To test this, we apply DEP-RL to challenging locomotion tasks involving an 18-muscle human and a 120-muscle bipedal ostrich. As Barbera et al. (2021) also attempted to train

a running gait from scratch, we choose their implementation of TD4 as a baseline for ostrich-run, while we provide MPO and DEP-MPO agents. See Suppl. C.4 for additional baselines.

High velocity running When applying DEP-MPO to the human-run task (Fig. 5, left), we observe similar initial learning between MPO and DEP-MPO. At $\approx 10^6$ steps, DEP-MPO policies suddenly increase in performance after a plateau. This coincides with the switch to an alternating gait, which is faster than an asymmetric gait. At the end of training, 5 out of 5 random seeds achieve an alternating gait with symmetric leg extensions, while MPO only achieves this for 1 out of 5 seeds (Fig. 5, right).

In ostrich-run, we observe faster initial learning for DEP-MPO (Fig. 6, left). There is, however, a drop in performance after which the results equalize with MPO. Looking at the best **rollout** of the 10 evaluation episodes for each point in the curve (Fig. 6), we observe that some DEP-MPO trials perform much better than the average. If we consequently record the velocity of the fastest policy checkpoint from all runs for each method, we observe that DEP-MPO achieves the largest **ever** measured top speed (Fig. 7, right). The DEP-MPO policy is also characterized by overall stronger leg extension and symmetric alternation of feet positions. See Suppl. C.5 for visualizations.

In contrast, *extensive* hyperparameter tuning did not lead to better asymptotic performance for MPO (Suppl. B.6), nor did other exploration noise types (Suppl. C.4).

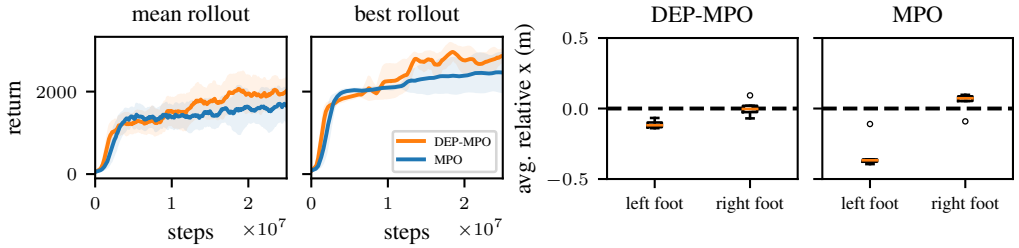


Figure 5: **DEP improves performance in the presence of local optima, as seen in human-run.** Left: DEP-MPO initially performs identically to MPO, before a sudden increase in performance can be observed for all trials. Right: The final gaits learned by DEP-MPO possess high symmetry, as measured by the averaged relative pelvis-deviation of the feet. Only 1 out of 5 random seeds for MPO achieves an alternating gait, which coincides with larger velocities and task returns.

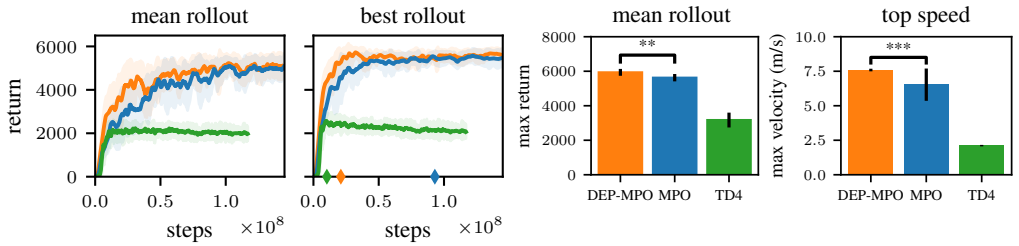


Figure 6: **Learning curves and maximal returns for ostrich-run.** Left: TD4 learns fast at first, but only achieves a suboptimal "hopping" gait. DEP-MPO outperforms the final MPO performance initially, but decays during late training. Even though the returns seem close, DEP-MPO achieves the fastest **ever** recorded top speed on the ostrich. Top speeds are averaged over 50 test episodes, using the fastest checkpoint for each method, marked by diamond markers. Statistical significance is marked with (**) for $p < 0.01$ and (***) for $p < 0.001$ using a student t-test. TD4 return is significantly lower than both other strategies (***, not shown for clarity of the figure).

Robustness against perturbations To investigate the obtained policies, we evaluate their robustness to out-of-distribution (OOD) variations in the environment. The policies are trained on a flat ground (Sec. 5.4) and then frozen, which we call in-distribution (ID). Afterwards, various OOD perturbations in the form of a large step of varying heights, a series of sloped steps or two inclined planes are introduced and the performance is measured to probe the robustness of the learned policies.

DEP-MPO yields the most robust controller against stepdown and sloped-step perturbations for all considered tasks (Fig. 7). As MPO is unable to achieve a good behavior without DEP for human-

Table 1: Training averaged performance metrics for the considered tasks. (S) marks success rates for reaching tasks, while (R) marks returns for all other tasks. Tasks for which HER is not applicable are marked with n.a. We perform student-t tests between the best DEP-augmented policy and the best non-DEP policy with significance levels: (*): $p < 0.05$; (**): $p < 0.01$; (***): $p < 0.001$

	DEP-MPO	DEP-HER-MPO	MPO	HER-MPO
arm26 (S)	$\mathbf{0.95 \pm 0.01}$ (***)	0.93 ± 0.01	0.62 ± 0.23	0.85 ± 0.09
ostrich-foraging (S)	0.42 ± 0.09	$\mathbf{0.90 \pm 0.03}$ (***)	0.00 ± 0.01	0.88 ± 0.03
humanreacher (S)	0.72 ± 0.16	$\mathbf{0.82 \pm 0.12}$ (*)	0.50 ± 0.23	0.65 ± 0.24
human-hop (R)	$\mathbf{472 \pm 85}$ (***)	n.a.	0 ± 0	n.a.
ostrich-run (R)	$\mathbf{4432 \pm 702}$ (**)	n.a.	4064 ± 732	n.a.
human-run (R)	1601 ± 320	n.a.	1395 ± 325	n.a.

hopstacle, we compare the performance of DEP-MPO with (OOD) and without (ID) perturbations. Interestingly, DEP-MPO is very robust, except for one random seed. This policy learned not to hop, but to move one of its legs above its shoulders, increasing its COM-height. Although hard to achieve, this behavior is sensitive to perturbations. Final policy checkpoints are used for all experiments.

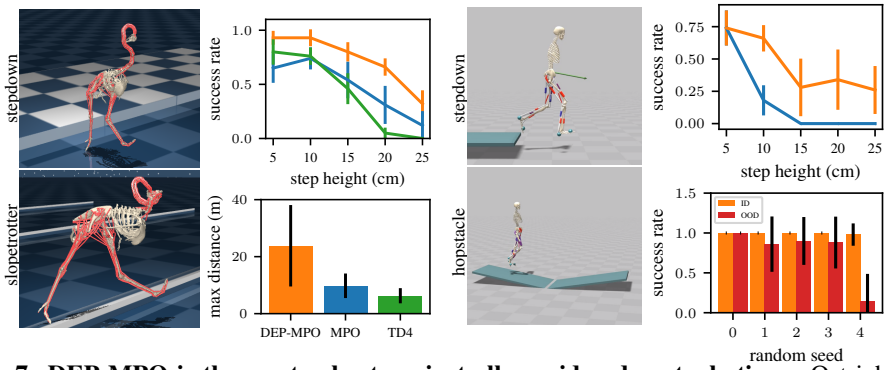


Figure 7: **DEP-MPO is the most robust against all considered perturbations.** Ostrich: DEP-MPO performs best under stepdown perturbations for varying step heights. The starting distance of the step was randomly varied. For the slopetrotter task, the average achieved distance is largest for DEP-MPO, although with considerable variability. Human: DEP-MPO also performs better for stepdown perturbations. For human-hopstacle, 4 out of 5 seeds of DEP-MPO achieve robust hopping. The remaining seed found a non-hopping solution achieving good returns that is not robust.

To our knowledge, we are the first to produce a robust running gait of this speed with RL applied to a system with 120 muscles, and we achieve this without reward shaping, curriculum learning or expert demonstrations. Without demonstrations, Barbera et al. (2021) only achieved a "hopping" behavior.

6 Conclusion

We have shown that common exploration noise strategies perform inadequately on overactuated systems using synthetic examples. We identified DEP, a controller from the domain of self-organization, of being able to induce state-space covering exploration in this scenario. We then proposed a way to integrate DEP into RL to apply DEP-RL to unsolved reaching (Fischer et al., 2021) and locomotion (Barbera et al., 2021) tasks. Even though we do not use motion capture data or training curricula, we were able to outperform all baselines. With this, we provide ample evidence that exploration is a key issue in the application of RL to muscular control tasks.

Despite the promising results, there are several limitations to the present work. The muscle simulation in MuJoCo is simplified compared to OpenSim and other software. While we provided results in the more biomechanically realistic simulator HyFyDy, the resulting motions are not consistent with human motor control yet. In order for the community to benefit from the present study, further work integrating natural cost terms or other incentives for natural motion is required. Additionally, the integration of DEP and RL, while performing very well in the investigated tasks, might not be feasible for every application. Thus, a more principled coupling between the DEP mechanism and an RL policy is an interesting future direction.

7 REPRODUCIBILITY STATEMENT

We provide extensive experimental details in Suppl. B.1, descriptions of all the used environments in Suppl. B.2 and all used hyperparameters together with optimization graphs in Suppl. B.6. The used RL algorithms are available from the TonicRL package (Pardo, 2020). The ostrich environment (Barbera, 2022) and the human-run environments are publicly available. The latter was simulated using a default model in SCONE (Geijtenbeek, 2019), an open-source biomechanics simulation software. We employed a recent version which includes a Python API, available at: `https://github.com/tgeijten/scone-core`. Additionally, we made use of the commercial SCONE plug-in HyFyDy (Geijtenbeek, 2021), which uses the same muscle and contact model as vanilla SCONE, but is significantly faster. We further include all environments and variations as well as the used learning algorithms in the submitted source code. All remaining environments are simulated in MuJoCo, which is freely available. A curated code repository will be published. We also mention hardware requirements in Suppl. B.5.

8 ACKNOWLEDGEMENTS

The authors thank Daniel Höglinger for help in prior work, Andrii Zadaianchuck, Arash Tavakoli and Sebastian Blaes for helpful discussions and Marin Vlastelica, Marco Bagatella and Pavel Kolev for their help reviewing the manuscript. A special thanks goes to Thomas Geijtenbeek for providing the scone Python interface. The authors thank the International Max Planck Research School for Intelligent Systems (IMPRS-IS) for supporting Pierre Schumacher. Georg Martius is a member of the Machine Learning Cluster of Excellence, EXC number 2064/1 – Project number 390727645. This work was supported by the Cyber Valley Research Fund (CyVy-RF-2020-11 to DH and GM). We acknowledge the support from the German Federal Ministry of Education and Research (BMBF) through the Tübingen AI Center (FKZ: 01IS18039B)

REFERENCES

Abbas Abdolmaleki, Jost Tobias Springenberg, Yuval Tassa, Remi Munos, Nicolas Heess, and Martin Riedmiller. Maximum a posteriori policy optimisation. In *International Conference on Learning Representations*, 2018. URL `https://openreview.net/forum?id=S1ANxQW0b`.

Mazen Al Borno, Jennifer L. Hicks, and Scott L. Delp. The effects of motor modularity on performance, learning and generalizability in upper-extremity reaching: A computational analysis. *Journal of The Royal Society Interface*, 17(167):20200011, 2020. doi: 10.1098/rsif.2020.0011.

Marcin Andrychowicz, Filip Wolski, Alex Ray, Jonas Schneider, Rachel Fong, Peter Welinder, Bob McGrew, Josh Tobin, OpenAI Pieter Abbeel, and Wojciech Zaremba. Hindsight experience replay. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL `https://proceedings.neurips.cc/paper/2017/file/453fadbd8a1a3af50a9df4df899537b5-Paper.pdf`.

Vittorio La Barbera. OstrichRL, 2022. URL `https://github.com/vittorione94/ostrichrl`.

Vittorio La Barbera, Fabio Pardo, Yuval Tassa, Monica Daley, Christopher Richards, Petar Kormushev, and John Hutchinson. OstrichRL: A musculoskeletal ostrich simulation to study bio-mechanical locomotion. In *Deep RL Workshop NeurIPS 2021*, 2021. URL `https://openreview.net/forum?id=7KzszSyQP0D`.

Ondrej Biza, Dian Wang, Robert Platt, Jan-Willem van de Meent, and Lawson L.S. Wong. Action priors for large action spaces in robotics. In *Proceedings of the 20th International Conference on Autonomous Agents and MultiAgent Systems*, AAMAS '21, pp. 205–213, Richland, SC, 2021. International Foundation for Autonomous Agents and Multiagent Systems. ISBN 9781450383073.

Dieter Buchler, Heiko Ott, and Jan Peters. A lightweight robotic arm with pneumatic muscles for robot learning. In *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 4086–4092. IEEE, 2016. ISBN 978-1-4673-8026-3. doi: 10.1109/ICRA.2016.7487599.

Douglas C. Crowder, Jessica Abreu, and Robert F. Kirsch. Hindsight Experience Replay Improves Reinforcement Learning for Control of a MIMO Musculoskeletal Model of the Human Arm. *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, pp. 1016–1025, 2021. doi: 10.1109/TNSRE.2021.3081056.

Scott L. Delp, Frank C. Anderson, Allison S. Arnold, Peter Loan, Ayman Habib, Chand T. John, Eran Guendelman, and Darryl G. Thelen. Opensim: Open-source software to create and analyze dynamic simulations of movement. *IEEE Trans. Biomed. Engineering*, 54(11):1940–1950, 2007. URL http://dblp.uni-trier.de/db/journals/tbe/tbe54.html#DelpAALHJGT07.

Ralf Der and Georg Martius. Novel plasticity rule can explain the development of sensorimotor intelligence. *Proceedings of the National Academy of Sciences*, 112(45):E6224–E6232, 2015. doi: 10.1073/pnas.1508400112. URL https://www.pnas.org/doi/abs/10.1073/pnas.1508400112.

A. Diamond and O. E. Holland. Reaching control of a full-torso, modelled musculoskeletal robot using muscle synergies emergent under reinforcement learning. *Bioinspiration & Biomimetics*, pp. 016015, 2014. doi: 10.1088/1748-3182/9/1/016015.

Danny Driess, Heiko Zimmermann, Simon Wolfen, Dan Suissa, Daniel Haeufle, Daniel Hennes, Marc Toussaint, and Syn Schmitt. Learning to Control Redundant Musculoskeletal Systems with Neural Networks and SQP: Exploiting Muscle Properties. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 6461–6468. IEEE, 2018. ISBN 978-1-5386-3081-5. doi: 10.1109/ICRA.2018.8463160.

Gabriel Dulac-Arnold, Richard Evans, Hado van Hasselt, Peter Sunehag, Timothy Lillicrap, Jonathan Hunt, Timothy Mann, Theophane Weber, Thomas Degris, and Ben Coppin. Deep Reinforcement Learning in Large Discrete Action Spaces, 2016. URL http://arxiv.org/abs/1512.07679.

Gregory Farquhar, Laura Gustafson, Zeming Lin, Shimon Whiteson, Nicolas Usunier, and Gabriel Synnaeve. Growing action spaces. In Hal Daumé III and Aarti Singh (eds.), *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pp. 3040–3051. PMLR, 7 2020. URL https://proceedings.mlr.press/v119/farquhar20a.html.

Florian Fischer, Miroslav Bachinski, Markus Klar, Arthur Fleig, and Jörg Müller. Reinforcement learning control of a biomechanical model of the upper extremity. *Scientific Reports*, 11(1):14445, Jul 2021. ISSN 2045-2322. doi: 10.1038/s41598-021-93760-1. URL https://doi.org/10.1038/s41598-021-93760-1.

Thomas Geijtenbeek. Scone: Open source software for predictive simulation of biological motion. *Journal of Open Source Software*, 4(38):1421, 2019. doi: 10.21105/joss.01421. URL https://doi.org/10.21105/joss.01421.

Thomas Geijtenbeek. The Hyfydy simulation software, 11 2021. URL https://hyfydy.com. https://hyfydy.com.

Elena Glassman and Russ Tedrake. A quadratic regulator-based heuristic for rapidly exploring state space. In *2010 IEEE International Conference on Robotics and Automation*, pp. 5021–5028, 2010. doi: 10.1109/ROBOT.2010.5509718.

D F B Haeufle, M Günther, A Bayer, and * Schmitt, S. Hill-type muscle model with serial damping and eccentric force-velocity relation. *Journal of Biomechanics*, pp. 1531–6, 2014. doi: 10.1016/j.jbiomech.2014.02.009.

Jakob Hollenstein, Auddy Sayantan, Matteo Saveriano, Erwan Renaudo, and Justus Piater. How do Offline Measures for Exploration in Reinforcement Learning behave? In *Knowledge Based Reinforcement Learning Workshop at IJCAI-PRICAI 2020, Yokohama, Japan* , 1 2021. URL https://iis.uibk.ac.at/public/papers/Hollenstein-2020-KBRL.pdf.

Aleksi Ikkala and Perttu Hämäläinen. Converting biomechanical models from opensim to mujoco. In Diego Torricelli, Metin Akay, and Jose L. Pons (eds.), *Converging Clinical and Engineering Research on Neurorehabilitation IV*, pp. 277–281, Cham, 2022. Springer International Publishing. ISBN 978-3-030-70316-5.

Yifeng Jiang, Tom Van Wouwe, Friedl De Groote, and C. Karen Liu. Synthesis of biologically realistic human motion using joint torque actuation. *ACM Trans. Graph.*, 38(4), jul 2019. ISSN 0730-0301. doi: 10.1145/3306346.3322966. URL https://doi.org/10.1145/3306346.3322966.

Emanuel Joos, Fabien Péan, and Orcun Goksel. Reinforcement learning of musculoskeletal control from functional simulations. In Anne L. Martel, Purang Abolmaesumi, Danail Stoyanov, Diana Mateus, Maria A. Zuluaga, S. Kevin Zhou, Daniel Racoceanu, and Leo Joskowicz (eds.), *Medical Image Computing and Computer Assisted Intervention – MICCAI 2020*, pp. 135–145, Cham, 2020. Springer International Publishing. ISBN 978-3-030-59716-0.

Łukasz Kidziński, Sharada Prasanna Mohanty, Carmichael F. Ong, Zhewei Huang, Shuchang Zhou, Anton Pechenko, Adam Stelmaszczyk, Piotr Jarosik, Mikhail Pavlov, Sergey Kolesnikov, Sergey Plis, Zhibo Chen, Zhizheng Zhang, Jiale Chen, Jun Shi, Zhuobin Zheng, Chun Yuan, Zhihui Lin, Henryk Michalewski, Piotr Milos, Blazej Osinski, Andrew Melnik, Malte Schilling, Helge Ritter, Sean F. Carroll, Jennifer Hicks, Sergey Levine, Marcel Salathé, and Scott Delp. Learning to run challenge solutions: Adapting reinforcement learning methods for neuromusculoskeletal environments. In Sergio Escalera and Markus Weimer (eds.), *The NIPS '17 Competition: Building Intelligent Systems*, pp. 121–153, Cham, 2018. Springer International Publishing. ISBN 978-3-319-94042-7.

Łukasz Kidziński, Carmichael Ong, Sharada Prasanna Mohanty, Jennifer Hicks, Sean F. Carroll, Bo Zhou, Hongsheng Zeng, Fan Wang, Rongzhong Lian, Hao Tian, Wojciech Jaśkowski, Garrett Andersen, Odd Rune Lykkebø, Nihat Engin Toklu, Pranav Shyam, Rupesh Kumar Srivastava, Sergey Kolesnikov, Oleksii Hrinchuk, Anton Pechenko, Mattias Ljungström, Zhen Wang, Xu Hu, Zehong Hu, Minghui Qiu, Jun Huang, Aleksei Shpilman, Ivan Sosin, Oleg Svidchenko, Aleksandra Malysheva, Daniel Kudenko, Lance Rane, Aditya Bhatt, Zhengfei Wang, Penghui Qi, Zeyang Yu, Peng Peng, Quan Yuan, Wenxin Li, Yunsheng Tian, Ruihan Yang, Pingchuan Ma, Shauharda Khadka, Somdeb Majumdar, Zach Dwiel, Yinyin Liu, Evren Tumer, Jeremy Watson, Marcel Salathé, Sergey Levine, and Scott Delp. Artificial Intelligence for Prosthetics - challenge solutions. 2019.

Dinant A. Kistemaker, A. J. K. Van Soest, J. D. Wong, I. Kurtzer, and P. L. Gribble. Control of position and movement is simplified by combined muscle spindle and Golgi tendon organ feedback. *Journal of Neurophysiology*, pp. 1126–1139, 2013. doi: 10.1152/jn.00751.2012.

Seunghwan Lee, Moonseok Park, Kyoungmin Lee, and Jehee Lee. Scalable muscle-actuated human simulation and control. *ACM Transactions on Graphics*, pp. 1–13, 2019. doi: 10.1145/3306346.3322972.

Shuzhen Luo, Ghaith Androwis, Sergei Adamovich, Erick Nunez, Hao Su, and Xianlian Zhou. Robust walking control of a lower limb rehabilitation exoskeleton coupled with a musculoskeletal model via deep reinforcement learning, 2021. URL https://doi.org/10.21203/rs.3.rs-1212542/v1.

Georg Martius, Rafael Hostettler, Alois Knoll, and Ralf Der. Compliant control for soft robots: Emergent behavior of a tendon driven anthropomorphic arm. In *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 767–773, 2016. doi: 10.1109/IROS.2016.7759138.

Andrew William Moore. Efficient memory-based learning for robot control. Technical report, University of Cambridge, 1990.

Yael Niv. Reinforcement learning in the brain. *Journal of Mathematical Psychology*, pp. 139–154, 2009. doi: 10.1016/j.jmp.2008.12.005.

Fabio Pardo. Tonic: A deep reinforcement learning library for fast prototyping and benchmarking. https://github.com/fabiopardo/tonic, 2020.

Cristina Pinneri and Georg Martius. Systematic self-exploration of behaviors for robots in a dynamical systems framework. In *Proc. Artificial Life XI*, pp. 319–326. MIT Press, Cambridge, MA, 2018. doi: 10.1162/isal_a_00062. URL https://www.mitpressjournals.org/doi/abs/10.1162/isal_a_00062.

Cristina Pinneri, Shambhuraj Sawant, Sebastian Blaes, Jan Achterhold, Joerg Stueckler, Michal Rolinek, and Georg Martius. Sample-efficient cross-entropy method for real-time planning. In *Conference on Robot Learning (CoRL) 2020*, 2020. URL https://corlconf.github.io/corl2020/paper_217/.

Miruna Pislar, David Szepesvari, Georg Ostrovski, Diana L Borsa, and Tom Schaul. When should agents explore? In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=dEwfxt14bca.

Christopher T. Richards and Enrico A. Eberhard. In vitro virtual reality: an anatomically explicit musculoskeletal simulation powered by in vitro muscle using closed-loop tissue–software interaction. *Journal of Experimental Biology*, 223(10), 05 2020. ISSN 0022-0949. doi: 10.1242/jeb.210054. URL https://doi.org/10.1242/jeb.210054. jeb210054.

Robert Rockenfeller, Michael Günther, Syn Schmitt, and * Götz, Thomas. Comparative sensitivity analysis of muscle activation dynamics. *Computational and Mathematical Methods in Medicine*, pp. 1–16, 2015. doi: 10.1155/2015/585409.

Jonas Rubenson, Denham B. Heliams, David G. Lloyd, and Paul A. Fournier. Gait selection in the ostrich: Mechanical and metabolic characteristics of walking and running with and without an aerial phase. *Proceedings of the Royal Society of London. Series B: Biological Sciences*, pp. 1091–1099, 2004. doi: 10.1098/rspb.2004.2702.

Katherine R. Saul, Xiao Hu, Craig M. Goehler, Meghan E. Vidt, Melissa Daly, Anca Velisar, and Wendy M. Murray. Benchmarking of dynamic simulation predictions in two software platforms using an upper limb musculoskeletal model. *Computer Methods in Biomechanics and Biomedical Engineering*, pp. 1445–1458, 2015. doi: 10.1080/10255842.2014.916698.

Tom Schaul, Daniel Horgan, Karol Gregor, and David Silver. Universal value function approximators. In Francis Bach and David Blei (eds.), *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pp. 1312–1320, Lille, France, 7 2015. PMLR. URL https://proceedings.mlr.press/v37/schaul15.html.

T Siebert and * Rode, C. Computational modeling of muscle biomechanics. In Zhongmin Jin (ed.), *Computational Modelling of Biomechanics and Biotribology in the Musculoskeletal System*, pp. 173–204. Woodhead Publishing, Elsevier, 1 edition, 2014. ISBN 978-0-85709-661-6. doi: 10.1533/9780857096739.2.173.

Avi Singh, Huihan Liu, Gaoyue Zhou, Albert Yu, Nicholas Rhinehart, and Sergey Levine. Parrot: Data-driven behavioral priors for reinforcement learning. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=Ysuv-WOFeKR.

Seungmoon Song, Łukasz Kidziński, Xue Bin Peng, Carmichael Ong, Jennifer Hicks, Sergey Levine, Christopher G. Atkeson, and Scott L. Delp. Deep reinforcement learning for modeling human locomotion control in neuromechanical simulation. *Journal of NeuroEngineering and Rehabilitation*, 18(1):126, Aug 2021. ISSN 1743-0003. doi: 10.1186/s12984-021-00919-y. URL https://doi.org/10.1186/s12984-021-00919-y.

Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. A Bradford Book, 2018. ISBN 0-262-03924-9.

Gabriel Synnaeve, Jonas Gehring, Zeming Lin, Daniel Haziza, Nicolas Usunier, Danielle Rothermel, Vegard Mella, Da Ju, Nicolas Carion, Laura Gustafson, and Daniel Gant. Growing Up Together: Structured Exploration for Large Action Spaces. 2019. URL https://openreview.net/forum?id=HylZ5grKvB.

Ehsan Tahami, Amir Jafari, and Ali Fallah. Learning to Control the Three-Link Musculoskeletal Arm Using Actor-Critic Reinforcement Learning Algorithm during Reaching Movement. *Biomedical Engineering: Applications, Basis and Communications*, pp. 1450064, 2014. doi: 10.4015/S1016237214500641.

Arash Tavakoli, Mehdi Fatemi, and Petar Kormushev. Learning to represent action values as a hypergraph on the action vertices. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=Xv_s64FiXTv.

Juan Camilo Vasquez Tieck, Marin Vlastelica Pogančić, Jacques Kaiser, Arne Roennau, Marc-Oliver Gewaltig, and Rüdiger Dillmann. Learning continuous muscle control for a multi-joint arm by extending proximal policy optimization with a liquid state machine. In Věra Kůrková, Yannis Manolopoulos, Barbara Hammer, Lazaros Iliadis, and Ilias Maglogiannis (eds.), *Artificial Neural Networks and Machine Learning – ICANN 2018*, pp. 211–221, Cham, 2018. Springer International Publishing. ISBN 978-3-030-01418-6.

J. Timmer and M. König. On generating power law noise. *A&A*, 300:707–710, 1995.

Emanuel Todorov, Tom Erez, and Yuval Tassa. MuJoCo: A physics engine for model-based control. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 5026–5033, 2012. doi: 10.1109/IROS.2012.6386109.

G. E. Uhlenbeck and L. S. Ornstein. On the theory of the brownian motion. *Phys. Rev.*, 36:823–841, 9 1930. doi: 10.1103/PhysRev.36.823. URL https://link.aps.org/doi/10.1103/PhysRev.36.823.

J.M. Wakeling, C. Tijs, N. Konow, and * A.A. Biewener. Modeling muscle function using experimentally determined subject-specific muscle properties. *Journal of Biomechanics*, pp. 110242, 2021. doi: 10.1016/j.jbiomech.2021.110242.

Johannes R. Walter, Michael Günther, Daniel F. B. Haeufle, and Syn Schmitt. A geometry- and muscle-based control architecture for synthesising biological movement. *Biological Cybernetics*, pp. 7–37, 2021. doi: 10.1007/s00422-020-00856-4.

Ziyu Wang, Tom Schaul, Matteo Hessel, Hado Van Hasselt, Marc Lanctot, and Nando De Freitas. Dueling network architectures for deep reinforcement learning. In *Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48*, ICML'16, pp. 1995–2003. JMLR.org, 2016.

Kunkun Zhao, Haiying Wen, Zhisheng Zhang, Manfredo Atzori, Henning Müller, Zhongqu Xie, and Alessandro Scano. Evaluation of Methods for the Extraction of Spatial Muscle Synergies. *Frontiers in Neuroscience*, 16, 2022. ISSN 1662-453X. URL https://www.frontiersin.org/articles/10.3389/fnins.2022.732156.

Supplementary Material

Videos of the observed exploration patterns and learned policies are available here[2]. A curated code repository will be published upon acceptance.

A Theoretical background

A.1 State-space coverage

Following (Glassman & Tedrake, 2010; Hollenstein et al., 2021), a possible measure for joint-space coverage in low-dimensional spaces can be obtained by projecting all recorded joint values $q_1^k, q_2^k, ..., q_T^k$ with $k \in \{1, 2\}$ into a discrete grid. For minimum and maximum joint values a and b, and a grid of size N^2, let there be a discretized vector of values $x_k = a + k\,\Delta x$, with $\Delta x = (b-a)/N$ and $k \in \{1, ..., N-1\}$. We can then compute a matrix S_{ij} such that:

$$S_{ij} = \begin{cases} 1, & \text{if } \exists t \text{ such that } x_i < q_t^0 < x_{i+1} \text{ and } x_j < q_t^1 < x_{j+1} \\ 0, & \text{otherwise} \end{cases} \tag{5}$$

The coverage is then given by:

$$\tilde{S} = \frac{\sum_{i,j} S_{ij}}{N^2}, \tag{6}$$

which is the number of visited grid points divided by the total grid size. In practice, the metric will not reach 100% as the arm cannot reach every point in space due to its geometry.

A.2 Ornstein-Uhlenbeck noise

The OU-process (Uhlenbeck & Ornstein, 1930) is a stochastic process that produces temporally correlated signals. It is defined by:

$$x_{t+1} = x_t + \theta(\mu - x_t)\Delta t + \sigma\,\omega_t, \tag{7}$$

where $\omega_t \sim \mathcal{N}(\cdot|0, 1)$ is a noise term sampled from a standard Gaussian distribution that drives the process, θ controls the strength of the drift term, σ the strength of the stochastic term and μ is the mean. For our experiments, we set $x_0 = \mu = 0$ s.t. θ and σ remain as tunable parameters. In practice, actions computed by the OU-process might exceed the allowed range $a \in [-1, 1]$ for large σ; actions are subsequently clipped to the minimum and maximum values.

A.3 Colored noise

The color of random noise is defined by the frequency dependency of its power spectral density (PSD):

$$\text{PSD}(f) \propto \frac{1}{f^\beta}, \tag{8}$$

where β is the frequency exponent of the power-law, sometimes colloquially referred to as the color of the noise. For uncorrelated, or white, noise $\beta = 0$ and the PSD is constant. In general, larger values of β lead to noise signals with slower frequency contributions. While colors such as white ($\beta = 0$), pink ($\beta = 1$) and red ($\beta = 2$) were investigated in Sec. 5.2, we allowed any value $\beta \geq 0$ for the optimization in Sec. C.4. In practice, we use an identical implementation of colored noise to (Pinneri et al., 2020), which is based on an efficient Fourier transformation. The tuneable parameters are the color of the noise $\beta \geq 0$ and a scaling parameter $\sigma \geq 0$ which is multiplied by the noise values. Actions exceeding the allowed ranges are clipped, identical to the previous section.

A.4 Muscle modeling

The force production in biological muscles is quite complex and state-dependent (Wakeling et al., 2021; Siebert & Rode, 2014; Haeufle et al., 2014). In contrast to most robotic actuators, the force depends non-linearly on muscle length, velocity, and stimulation. The muscle also has low-pass filter characteristics, making the control problem hard for classical approaches—in addition to the typical redundancy of having more muscles than DoF. While these properties are all reproduced in the

[2] https://sites.google.com/view/dep-rl

MuJoCo internal muscle model, which has been used for other muscle-based studies (Barbera et al., 2021; Fischer et al., 2021; Ikkala & Hämäläinen, 2022; Richards & Eberhard, 2020), MuJoCo uses certain simplifications. It employs phenomenological force-length and force-velocity relationships which are modeled as simple functions. An additional choice, is that the tendon that connects muscles and bones is inelastic. While in more realistic systems the tendon length might vary independently of the muscle length, in MuJoCo we have:

$$l_{\text{total}} = l_{\text{muscle}} + \underbrace{l_{\text{tendon}}}_{\text{constant}}, \tag{9}$$

where l_{muscle} is the length of the muscle fiber and l_{tendon} the length of the tendon. Knowledge of the muscle fiber lengths l_{muscle} consequently allows the unique determination of l_{total} and to infer the current joint configuration. While this choice is sensible considering the immense data requirements of RL, it simplifies the control problem compared to realistic biological agents which must infer l_{total} from other proprioceptive signals, which certain studies suggest to be possible in biological systems (Kistemaker et al., 2013).

We also point out that the chemical muscle activation dynamics in MuJoCo induce a low-pass filter on the applied control signals, such that temporally uncorrelated actions might not cause significant motion. The muscle activity $a_{\text{m}}(t)$ is governed by the dynamics equation:

$$\dot{a}_{\text{m}}(t) = \frac{a(t) - a_{\text{m}}(t)}{\tau(a_{\text{m}}(t), a(t))}, \tag{10}$$

where a is the action as computed by the RL policy and τ is an action and activity dependent time scale, given by:

$$\tau(a_{\text{m}}(t), a(t)) = \begin{cases} \tau_{\text{act}}(0.5 + 1.5\, a_{\text{m}}(t)) & \text{if } a(t) > a_{\text{m}}(t) \\ \tau_{\text{deact}}/(0.5 + 1.5\, a_{\text{m}}(t)) & \text{if } a(t) \leq a_{\text{m}}(t) \end{cases}. \tag{11}$$

The constants τ_{act} and τ_{deact} are set to 0.01 and 0.04 by default, such that activity increases faster than it decreases.

See Barbera et al. (2021) and the MuJoCo documentation for more details on the muscle model and its parametrization.

Real biological systems also suffer significant delays for sensors and actuators, as well as being restricted to certain input modalities. These constraints are currently not modeled in MuJoCo.

B Experimental details

B.1 General details

For the state coverage measures (Fig. 11 and 12), we recorded 50 episodes of 1000 iterations each. The environment was reset after every episode. The state coverage metric was computed over 5 episodes at a time, after which we reset the internal state of DEP.

All experiments involving training (Fig. 3, 4 and 6) were averaged over 10 random seeds. Each point in the learning curves corresponds to 10 evaluation episodes without exploration that were recorded at regular intervals during training.

For the maximum speed measurements, the fastest checkpoint out of all runs in Fig. 6 was chosen for each method. We then executed 50 test episodes without exploration and recorded the fasted velocity within each episode.

For the robustness evaluations, the last training checkpoint of each run in Fig. 6 is chosen, as the robustness of the policies generally increases with training time in our experiments. We then record 100 episodes each with ostrich-stepdown and ostrich-slopetrotter perturbations, without any exploration. For the former, a binary success is recorded if the ostrich is able to pass the step and run for 10 additional meters afterwards. For the latter, we record the average traveled distance, as a large number of obstacles prevents most rollouts from successfully running past all of them.

For the gait visualizations (Fig. 15 and 16), the same policies as for the speed measurements are used, as they exhibit the most natural gaits. We then record a single episode and visualize the last 5 seconds to ensure a converged pattern.

B.2 ENVIRONMENTS

All tasks except OstrichRL (Barbera et al., 2021) and the human environments Geijtenbeek (2019; 2021) were constructed from existing geometrical models in MuJoCo (Ikkala & Hämäläinen, 2022; Todorov et al., 2012) from which we created RL environments. We additionally created variants of ostrich-run involving perturbations, i.e. ostrich-stepdown, and ostrich-slopetrotter.

torquearm A 2-DoF arm that moves in a 2D plane and is actuated by 2 torque generators. It is not used for RL, but as a comparative tool. Its geometry is identical to arm26, but different joint positions are reachable as it is not restricted by the geometry of the muscles. We manually restrict the joint ranges to $q_t^i \in [-120, 120]$ (degrees) to prevent self-collisions.

arm26 A 2-DoF planar arm driven by 6 muscles. The model was adapted from the original one in (Todorov et al., 2012), we modified the maximum muscle forces and shifted the gravity such that the arm fully extends ("down" on Fig. 1). The agent has to reach goals that are 5 cm in radius. They randomly appear in the upper right corner in a 35 cm by 15 cm rectangular area. The arm motion is restricted compared to torquearm by the passive stretch of the muscle fiber. The reward is given by:

$$r(s, s') = \begin{cases} 10, & \text{if } d(s) < 0.05 \\ -1, & \text{otherwise,} \end{cases} \tag{12}$$

where $d(s)$ is the Euclidean distance between the hand position and the goal. The episode terminates if the goal is reached, i.e. $d(s) < 0.05$. The negative reward incentivizes the agent to reach the goal as quickly as possible. Exploration in this task is difficult not only because of the overactuation, but also because the activation dynamics of the muscles require temporal correlation for effective state coverage. An episode lasts for 300 iterations, with $\Delta t = 10$ ms.

humanreacher A 7-DoF arm that moves in full 3D. It is actuated by 50 muscles. The agent has to reach goals of 4 cm that randomly appear in front of it at "face"-height in a 15 cm by 30 cm by 25 cm rectangular volume. The reward function and termination condition are identical to arm26, except for the goal radius. In addition to the issues detailed in the previous paragraph, singular muscles are not strong enough to effect every joint motion. For example, pulling the arm above the shoulder requires several muscles to be stimulated at the same time, while opposing, antagonistic, muscles should not be active. The muscular geometry is also strongly asymmetric. An exploration strategy has to compute the right correlation across connected muscle groups, and across time, for each motion. The joint limits and the bone geometry create cul-de-sac states, e.g. at some point the agent might not be able to extend the elbow further to reach a goal, it has to move back and change the pose. The initial pose of the arm is fully extended and points downwards. We randomly vary the joint pose by $q^i_{\text{init}} + \mathcal{N}(0, 0.01)$ and each joint velocity by $\dot{q}^i_{\text{init}} + \mathcal{N}(0, 0.03)$ after each episode reset. This helps the RL agent and HER to make progress on the task as it causes the arm to slightly self-explore. As DEP is fully deterministic, it also prevents it from generating the same control signals during each episode in the initial unsupervised exploration phase. An episode lasts for 300 iterations, with $\Delta t = 10$ ms.

ostrich-foraging This task is unchanged from (Barbera et al., 2021), except for the rewards which we modified to be sparse, identical to arm26. The termination condition is also identical. An ostrich neck and head actuated by 52 muscles need to reach randomly appearing goals with the beak. The goals appear in a uniform sphere around the beak, but only goals with goal-beak distances $d(s) \in [0.6, 0.8]$ are allowed. The goals have a radius of 5 cm. The initial pose is an upright neck position (see Fig. 1), but following the original task (Barbera et al., 2021) the pose is **not** randomized after episode resets, the last pose of the previous episode is simply kept as the first pose of the new episode. It is thus very unlikely for an agent with inadequate exploration to ever encounter a single goal. The neck itself is very flexible and offers almost no easily reachable cul-de-sac states, which we conjecture to explain the good performance of HER-MPO in Fig. 4. An episode lasts for 400 iterations, with $\Delta t = 25$ ms.

ostrich-run The bipedal ostrich, from (Barbera et al., 2021), needs to run as fast as possible in a horizontal line and is only provided a weakly-constraining reward in form of its velocity. Only provided with this generic reward and without motion capture data, a learning agent is prone to local

optima. The bird possesses 120 individually controllable muscles and moves in full 3D without any external constraints. The reward is given by:

$$r(s, s') = v_x^{\mathrm{COM}}(s), \tag{13}$$

where $v_x^{\mathrm{COM}}(s)$ is the velocity of the center of mass projected to the x-axis. An ideal policy will consequently run in a perfectly straight line as fast as possible. The episode terminates if the head of the ostrich is below 0.9 m, the pelvis is below 0.6 m or the torso angle exceeds $-0.8 < \theta_{\mathrm{torso}}(s) < 0.8$ (radians). The leg positions are slightly randomized at the end of each episode, which lasts for a maximum horizon of 1000 iterations with $\Delta t = 25$ ms.

We point out that the author's implementation of ostrich-run (Barbera, 2022) has set a default stiffness to all joints in the simulation. While this generally ensures the stability of the model, that only applies to joints that connect different parts of the system. In this case, the stiffness was also set for the root joints of the ostrich, essentially creating a spring that weakly pulls it back to the starting position. As the absolute x-position is withheld from the agent to create a periodic state input, the non-observability of the spring force destabilizes learning. We therefore explicitly set the stiffness of all root positional and rotational joints to 0. This explains why our TD4 baseline reaches significantly higher scores than in the work by (Barbera et al., 2021). Our measured maximum return for TD4 lies at ≈ 4044, while the reported returns without the change did not seem to exceed 2000.

ostrich-stepdown A step is added to the original ostrich-run task. The height of the step is adjustable and its position randomly varies with $\Delta x \sim \mathcal{N}(\cdot|0, 0.2)$. The ostrich is initially on top of the step and has to run across the drop in height without falling over. The task is successful if the ostrich manages to run for ≈ 10 meters past the step. The episode is terminated if the x-position exceeds 10 m, the torso angle exceeds $-0.8 < \theta_{\mathrm{torso}}(s) < 0.8$ rad, the head is below 0.5 m or the head is below the pelvis height. We relaxed the termination conditions to allow for suboptimal configurations that are used to bring the ostrich back into a running pose.

ostrich-slopetrotter A series of half-sloped steps is added to the original ostrich-run task. The obstacles are sloped on the incoming side while there is a perpendicular drop similar to a conventional step on the outgoing side. Rectangular stairs would disadvantage gaits with small foot clearance, while the half-slope allows most gaits to move up the step without getting stuck. There are seven obstacles spaced at 5 m intervals in total. The episode terminates if the x-position exceeds 50 m, the remaining termination conditions are identical to ostrich-stepdown. The obstacles are wide enough to prevent slightly diagonal running gaits from simply avoiding the obstacles. The task reward is the achieved distance, given at the end.

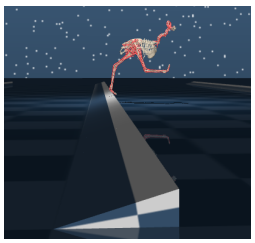


Figure 8: Sloped step for ostrich-slopetrotter.

human-run This task uses the planar human model from the NeurIPS competition (Kidziński et al., 2018) simulated in HyFyDy instead of OpenSim. HyFyDy uses the same muscle and contact models as OpenSim, but is significantly faster. The model has 18 leg muscles and no arms. The task reward is the COM-velocity in x-direction, identical to the ostrich. The episode length is 1000. The initial position is slightly randomized from a standing position. The episode terminates if the COM-height falls below 0.5 m.

human-hop In this task, the reward of human-run is changed to be sparse. The agent receives a reward of 1 if its COM-height exceeds 1.08 m and 0 otherwise. Periodic hopping will maximize this reward. The task is particularly challenging as there is no goal-conditioning to improve exploration in early training. The agent has to figure out a single hop from scratch. The initial state is slightly randomized from a squatting position.

human-stepdown The human-run task is modified by including a parcours of varying slope with a large drop at the end. We record a success if the agent is able to navigate the entire parcours without falling to the ground.

human-hopstacle The human-hop task is modified by including two inclined slopes. The task is marked as a success if the agent is able to periodically hop for 1000 time steps without falling to the

Table 2: Number of joints, state and action dimensions for all considered tasks. The ostrich-run variants ostrich-stepdown and ostrich-slopetrotter share identical state and action spaces, as the policies are not retrained. The planar reaching tasks torquearm and arm26 are used with virtual action spaces. For a given action multiplier n, we also multiply all muscle-related state data by n, i.e. muscle lengths, velocities, forces and activity.

	torquearm	**arm26**	**humanreacher**	**ostrich-foraging**	**ostrich-run**	**human-run**
# of joints	2	2	21	36	56	9
action dimension	2...600	6...600	50	52	120	18
state dimension	16	34	248	289	596	194
episode length	300	300	300	400	1000	1000
time step	10 ms	10 ms	10 ms	25 ms	25 ms	10 ms

ground. Due to the arrangement of the slopes, the agent will either hop inside the funnel, or jump out to the sides, where it will experience a steep drop.

Additional information regarding state and action sizes are summarized in Table 2. The observations are summarized in Table 3.

B.3 DEP IMPLEMENTATION

We use a window of the recent history to adapt the DEP controller during learning. DEP requires as input 1 proprioceptive sensor per actuator. We use joint angles for the torque-driven example in Fig. 11, while all muscle-driven tasks use the sum of muscle lengths and muscle forces, normalized with recorded data to lie in $[-1, 1]$:

$$s_{\mathrm{DEP}} = \tilde{l}_{\mathrm{muscle}} + c\,\tilde{f}_{\mathrm{muscle}}, \tag{14}$$

where l_{muscle} is the length of the muscle fibre, f_{muscle} the force acting on it and c is a scaling constant. Note that $s_{\mathrm{DEP}} \in \mathbb{R}^m$ and $a \in \mathbb{R}^n$ with $m = n$. Even though the rest of the state information is discarded, DEP computes action patterns that achieve correlated sensor changes. When alternating between DEP and the policy in DEP-RL, we also feed the current input to DEP and perform training updates. We observed performance benefits in locomotion tasks as DEP's output is strongly influenced by the recent gait dynamics induced by the policy. DEP is implemented to compute a batch of actions for a batch of parallel environments such that there is a separate history-dependent controller for each

Table 3: State information for all environments. The variants ostrich-stepdown and ostrich-slopetrotter use identical observations to ostrich-run. This allows the evaluation of the robustness of the trained policies against OOD perturbations.

environment	**observations**
torquearm	joint positions, joint velocities, actuator positions, actuator velocities, actuator forces, goal position, hand position
arm26	joint positions, joint velocities, muscle lengths, muscle velocities, muscle forces, muscle activity, goal position, hand position
arm750	joint positions, joint velocities, muscle lengths, muscle velocities, muscle forces, muscle activity, goal position, hand position
ostrich-foraging	joint positions, joint velocities, muscle activity, muscle forces, muscle lengths, muscle velocities, beak position, goal position, the vector from beak position to the goal position
ostrich-run	head height, pelvis height, feet height, joint positions (without x), joint velocities, muscle activity, muscle forces, muscle lengths, muscle velocities, COM-x-velocity
human-run	joint positions (without x), joint velocities, muscle lengths, muscle velocities, muscle forces, muscle activity, y-position of all bodies, orientation of all bodies, angular velocity of all bodies, linear velocity of all bodies, COM-x-velocity, torso angle, COM-y-position

Table 4: DEP hyperparameters for the learned policies. The test episode value signifies that an episode without DEP is recorded every N episodes. The value for arm-reaching was so large that it was effectively never used. The force scale value is used to scale the force input and the muscle length input.

(a) Arm-reaching settings.

	Parameter	Value
DEP	κ	1000
	τ	80
	buffer size	600
	bias rate	0.00002
	s4avg	6
	time dist (Δt)	60
integration	p_{switch}	0.01
	H_{DEP}	20
	test episode	n.a.
	force scale	0.0003

(b) Ostrich settings.

	Parameter	Value
DEP	κ	20
	τ	8
	buffer size	90
	bias rate	0.03
	s4avg	1
	time dist (Δt)	5
integration	p_{switch}	0.0004
	H_{DEP}	4
	test episode	3
	force scale	0.0003

(c) Human-run settings.

	Parameter	Value
DEP	κ	1896
	τ	26
	buffer size	200
	bias rate	0.004154
	s4avg	0
	time dist (Δt)	4
integration	p_{switch}	0.01
	H_{DEP}	10
	test episode	n.a.
	force scale	0.000054749

(d) Human-hop settings.

	Parameter	Value
DEP	κ	1288
	τ	35
	buffer size	200
	bias rate	0.0926
	s4avg	0
	time dist (Δt)	6
integration	p_{switch}	0.005
	H_{DEP}	30
	test episode	n.a.
	force scale	0.000547

environment. As the control matrix is very small, e.g. $C \in \mathbb{R}^{120\times120}$ even in ostrich-run, the most computationally intensive environment, this can be done with minimal overhead.

B.4 RL IMPLEMENTATION

Our RL algorithms are implemented with a slightly modified version of TonicRL (Pardo, 2020).

B.5 HARDWARE

Training of each DEP-MPO agent for ostrich-run, the most computationally intensive environment, was executed on an NVIDIA V100 GPU and 20 CPU cores. Training for 10^8 iterations requires about 48 hours in real-time. Note that in general, we do train for 30 iterations for every 1000 environment interactions, which speeds up training with regard to the reported learning steps. See Sec. B.6 for details.

B.6 HYPERPARAMETERS

We first detail the optimization choices made in the main part, before we give the specific hyperparameters that were chosen.

Optimization for ostrich-run We performed extensive hyperparameter optimization for the ostrich-run task with baseline MPO, but could not achieve a better final performance than default MPO parameters. The best performing set is identical in performance to the best run with default parameters

Table 5: RL parameters for MPO and TD4. The TD4 parameters are identical to (Barbera et al., 2021). Non-reported values are left to their default setting in TonicRL (Pardo, 2020).

(a) MPO settings.

Parameter	Value
buffer size	1e6
batch size	256
steps before batches	3e5
steps between batches	1000
number of batches	30
n-step return	3
n parallel	20
n sequential	10

(b) TD4 settings.

Parameter	Value
buffer size	1e6
batch size	100
steps before batches	5e4
steps between batches	50
number of batches	50
n-step return	1
learning rate	1e-4
TD3 action noise scale	0.25
n parallel	15
n sequential	8
exploration	OU
Action noise scale	0.25
Warm up random steps	1e4

Table 6: Baseline parameters. For HER, 80% of the time a relabelled transition is added in addition to the original one.

(a) OU-noise settings.

	Parameter	Value
humanreacher	θ-drift	0.004
	σ-scale	0.02
ostrich-run	θ-drift	0.1
	σ-scale	0.07

(b) Colored noise settings.

	Parameter	Value
humanreacher	β-color	0.04
	σ-scale	0.1
ostrich-run	β-color	0.008
	σ-scale	0.3

(c) Hindsight experience replay settings.

Parameter	Value
strategy	final
% hindsight	80

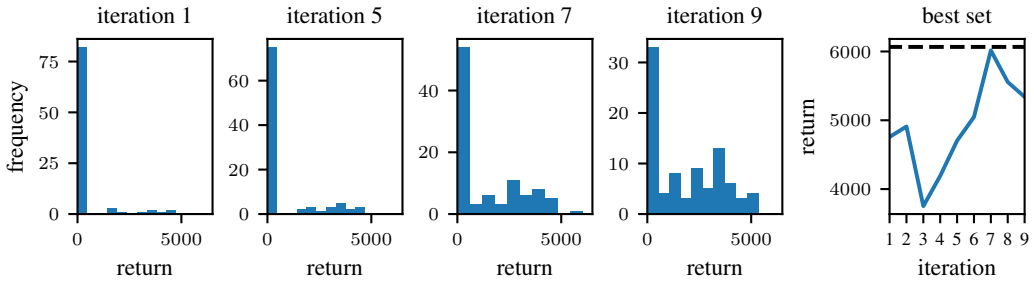


Figure 9: **Hyperparameter optimization for ostrich-run.** We performed 9 iterations of meta-optimization for MPO with 100 sets of parameters in each round, for a total of 900 different combinations. The final best set did not outperform the best run with the default parameters of MPO. The rightmost figure shows the best performing set for each iteration. The best return achieved over 10 evaluation episodes by MPO with default parameters out of 10 seeds is shown with a dashed black line.

in Fig. 6. In total, we computed 9 iterations of meta-optimization with 100 different sets of parameters in each round. The evolution of the performance histograms is shown in Fig. 9.

Exploration experiments For the experiments in Sec. 5.2, all noise strategies, with the exception of DEP, were tuned in a grid search to maximize the end effector state space coverage for each task and for each action space separately. DEP was tuned once to maximize a sample joint-pace entropy measure of the humanreacher task; its hyperparameters were then kept constant for **all** arm-reaching tasks in **all** sections of our study.

DEP-RL We identify three groups of tunable parameters for DEP-RL: the RL agent parameters (Table 5), the DEP parameters (Table 4), and the parameters controlling the integration of DEP and the policy. We initially optimized only for the parameters of DEP and the integration. When we afterwards ran an optimization procedure for all sets of parameters at the same time, we could not outperform our previous results. A pure MPO parameter search did also not yield better performance, such that we kept the parameters of MPO identical to the default parameters in the TonicRL library, except for minor changes regarding parallelization and batch sizes. The DEP parameters for DEP-RL in the reaching tasks were kept identical to the previous paragraph, while we heuristically chose the integration parameters. We, therefore, had 1 set of values for all arm-reaching tasks in the entire study. The DEP and the integration parameters for ostrich-run were optimized for performance, we kept them identical for ostrich-foraging.

Baselines The additional baselines for humanreacher and ostrich-run, see Suppl. C.4, were tuned individually for each task to maximize performance. TD4 is used with identical parameters to (Barbera et al., 2021). The values are detailed in Table 6.

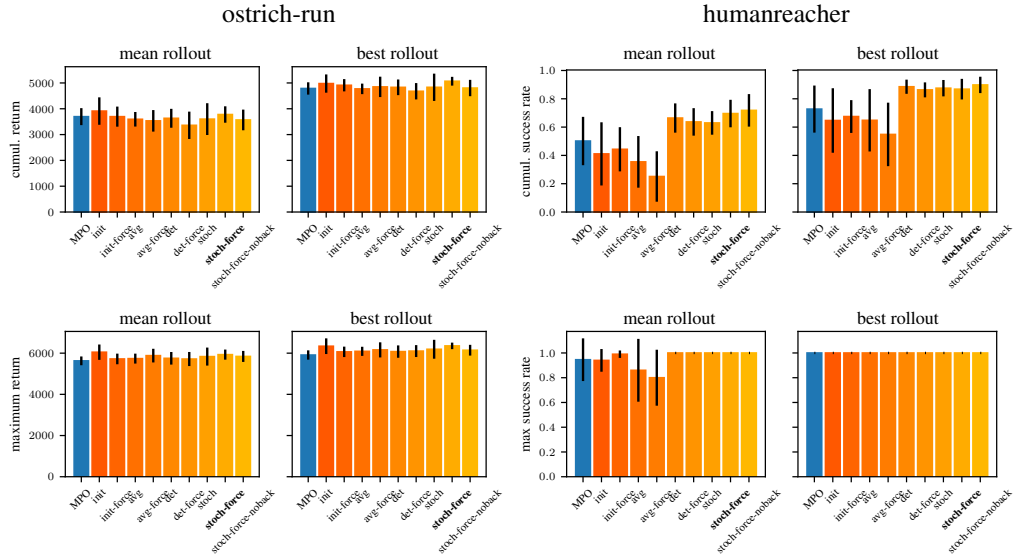


Figure 10: **Ablation experiments for DEP-RL.** The stoch-force-variant (bold) was used for all experiments in the main part. All ablations were trained for 5×10^7 iterations and averaged over 10 random seeds.

C Additional Experiments

C.1 State-Coverage

We show additional visualizations of the trajectories generated by different noise processes on torquearm and arm26 in Fig. 11 and Fig. 12 respectively.

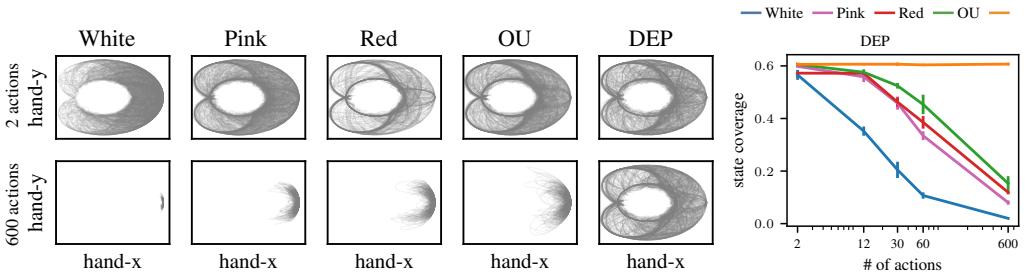


Figure 11: **Only DEP reaches adequate state-space coverage for all considered action spaces in torquearm.** Hand trajectories collected during 50 episodes of 1000 iterations ($\Delta t = 10$ ms) of pure exploration with different noise strategies. Left: Hand trajectories for the original action space $a \in \mathbb{R}^2$ (top) and expanded action space $a \in \mathbb{R}^{600}$ (bottom). Right: Endeffector-space coverage.

C.2 Ablations

We experiment with several implementations of DEP-RL and show cumulative and maximum performances for a selection of them in Fig. 10. The ablations are:

init DEP is only used for initial unsupervised exploration. The collected data is used to pre-fill the replay buffer. This component is active in all other ablations.
avg DEP actions and policy actions are combined in a weighted average. The DEP weight is much smaller than the policy weight.
det DEP and the policy control the system in alternation. They are deterministically switched s.t. DEP acts for H_{DEP} iterations and the RL policy for H_{RL} iterations. The current state is used to train

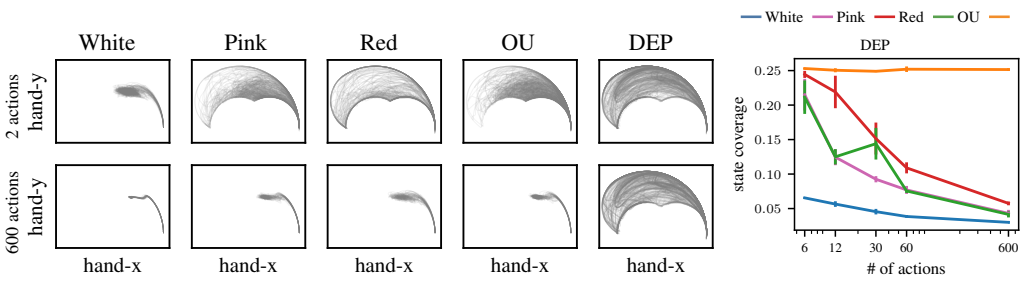


Figure 12: **Only DEP reaches adequate state coverage for all considered action spaces in arm26.** Hand trajectories collected during 50 episodes of 1000 iterations ($\Delta t = 10$ ms) of pure exploration with different noise strategies. Left: Hand trajectories for the original action space $a \in \mathbb{R}^6$ (top) and expanded action space $a \in \mathbb{R}^{600}$ (bottom). Right: endeffector-space coverage.

and adapt the DEP agent, even if the RL action is used for the environment. DEP actions are also added to the replay buffer of the RL agent.

stoch Identical to the previous ablation, but the alternation is stochastic s.t. there is a probablitiy p_{switch} that DEP takes over for H_{DEP} iterations.

We additionally introduce force-variants of all these ablations where the state input of DEP is not only composed of the muscle lengths, but also the forces acting on the muscles. We observed that this causes DEP to seek out states that produce more force variations, which are generally interesting for locomotion. Lengths and forces are normalized from recorded data and then added together with a certain weighting, in order to not change the number of input dimensions of DEP.

Lastly, we show the performance for a *noback*-variant, where DEP is not learning in the background while the RL agent is taking over control. Even though the init-variant achieves a fast gait for locomotion, we chose the **stoch-force**-variant for all the results in the main section, as it achieves good performance on all tasks. All ablations were averaged over ten random seeds.

C.3 Action correlation matrix

We recorded action patterns generated from different noise strategies applied to ostrich-run. Even though we only recorded 50 s of data, and DEP was learning from **scratch**, strong correlations and anti-correlations across muscle groups can be observed in Fig. 13. We deactivate episode terminations in order to observe the full bandwidth of motion generated by DEP. For this particular task, the ostrich was lying on the ground while moving the legs back and forth in an alternating pattern. Uniform, colored and OU noise are unable to produce significant correlations across actions.

C.4 Additional baselines

We combine MPO with colored (β-MPO) and OU-noise (OU-MPO) by summing them to the action computed by the baseline MPO policy. We then apply these new algorithms to humanreacher and ostrich-run, as they constitute challenging reaching and locomotion tasks. We also tested an implementation of DEP-TD4 on ostrich-run. The base agents were identical for all baselines, while the OU and the colored noise were optimized to achieve the best performance, see Suppl. B.6. It can be seen in Fig. 14 (left) that DEP-MPO achieves the largest returns in ostrich-run, while OU-MPO intermittently outperforms vanilla MPO. Similarly, OU-MPO and β-MPO perform better than MPO in the humanreacher task, as seen in Fig. 14 (right), but DEP-MPO achieves the best performance.

C.5 Ostrich gait visualization

We show the achieved foot movements and footstep patterns of the ostrich for the different algorithms in Fig. 15. The leg deviation is strongest for DEP-MPO, while it also achieves the most regular foot pattern. This suggests that DEP improves exploration, as it allows for policies that utilize the embodiment of the agent to a greater extent, while also achieving larger running velocities. MPO manages less leg extension, while the TD4 gait is irregular and asymmetric. The step lengths of $\approx$ 1m achieved by DEP-MPO are also quite close to real ostriches (Rubenson et al., 2004), while MPO and TD4 only achieve small step lengths of $\approx$ 0.5m and $\approx$ 0.3m respectively. We provide

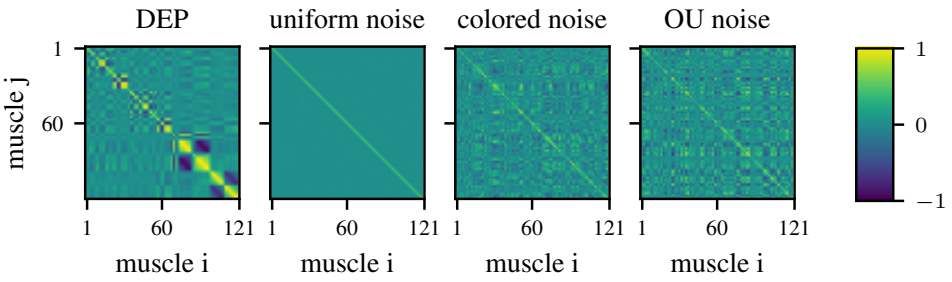


Figure 13: **Action correlation matrix for different exploration strategies.** We recorded 50 s of data (1000 transitions) from ostrich-run and computed the correlation matrix from the action trajectories. Note that even though DEP was initialized with $C_{ij} = 0$, strong correlation and anti-correlation patterns can be observed for antagonistically opposed muscle groups. Colored and OU noise do not exhibit strong correlations across actions, as they are designed to produce temporally correlated signals.

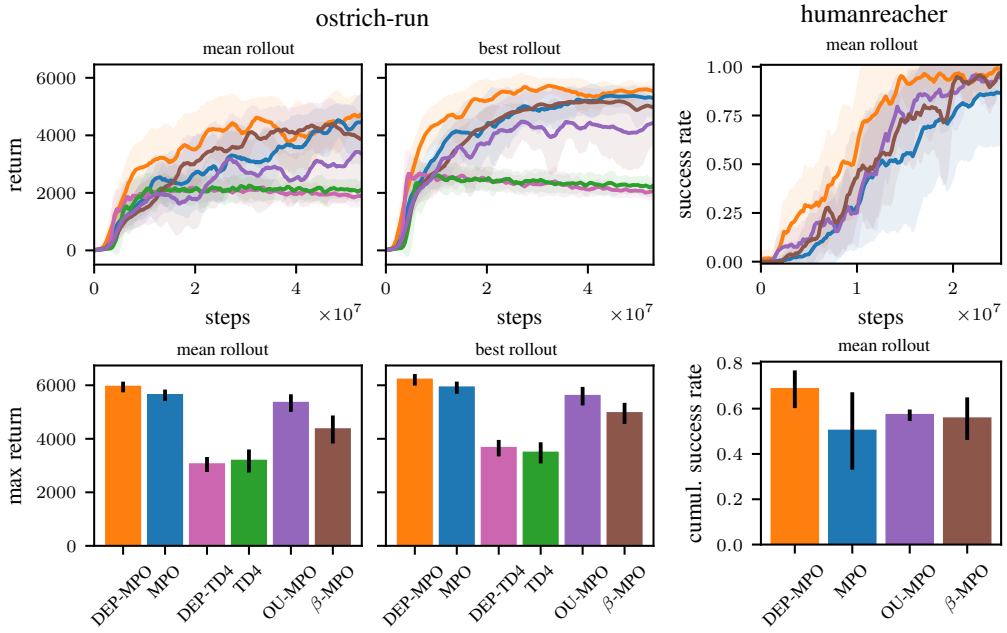


Figure 14: Left: Additional baselines for ostrich-run. We provide OU and β-MPO agents by summing them to the action computed by MPO as with regular exploration noise. Right: Identical baselines for humanreacher. OU and colored noise processes were optimized for the present tasks, while the base MPO agent was identical for **all** experiments in this figure.

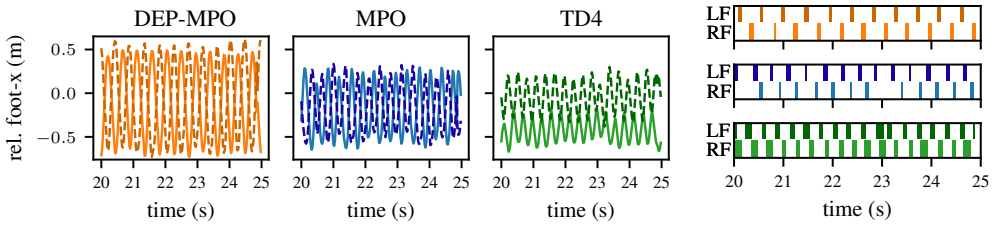


Figure 15: **DEP-MPO achieves the most widespread and symmetric gait.** Left: The relative sagital foot position w.r.t. the torso visualizes the leg extension during locomotion. DEP-MPO creates a symmetric gait with $\approx$ 1m step length. For MPO the step length is much shorter. TD4 has a completely shifted gait, the left foot is often in front of the right foot. Right: Foot contact pattern for all gaits. The shaded areas mark the time during which the respective foot (LF: left foot or RF: right foot) is in contact with the ground. Visualized are the last 5 seconds of an evaluation episode to ensure a converged pattern.

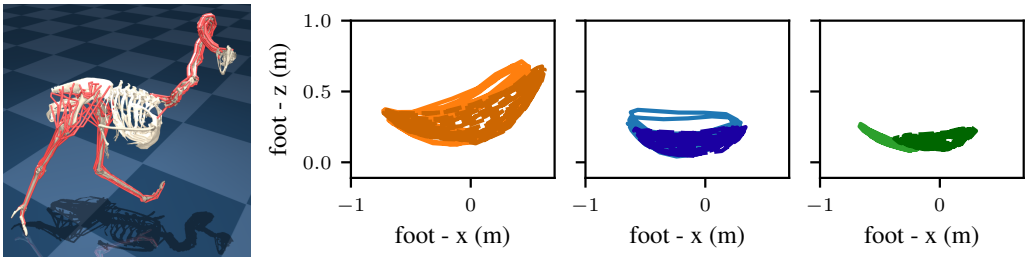


Figure 16: Foot gait patterns for ostrich-run during the final 5 s of an episode. While DEP-MPO portrays a slight asymmetry in the z-direction, MPO and TD4 are noticeably less symmetric.

additional visualizations of the relative x and z trajectories of the feet during locomotion for each algorithm in Fig. 16.

C.6 Additional evaluation with dense and sparse rewards in large virtual action spaces

We present in Fig. 17 the results for experiments with virtual action spaces, as shown in Fig. 3, but here in addition with HER and for the case of sparse and dense rewards. For dense rewards (top), an increasing number of actions requires more environment steps for vanilla MPO to solve the task, while the performance collapses for 600 actions. DEP-MPO reaches good performance for each considered action space. MPO performs similarly in the sparse task, albeit a larger variation across runs can be observed. While HER elicits faster learning, it still requires significantly more environment steps for 6 and 120 actions than DEP-MPO and does not achieve good performance for 600 actions. DEP-MPO and HER-DEP-MPO quickly solve all tasks.

C.7 Description of a one-dimensional system

In this section, we give an outline of how DEP works in a theoretical scenario. We will first describe a simplified DEP rule and detail how its dynamics might excite the mountain car system (Moore, 1990) to cover the state. We will then apply the simplified rule on the mountain car environment and present the results, see Fig. 19.

Mountain car The original DEP controller is described by:

$$a_t = \tanh(Cs_t + h_t), \tag{15}$$

with the state $s_t \in \mathbb{R}^n$, a time-dependent bias $h_t \in \mathbb{R}^m$, the action $a_t \in \mathbb{R}^m$ and the learned control matrix $C \in \mathbb{R}^{m \times n}$. The update rule is now defined as:

$$\tau \dot{C} = f(\dot{s}_t) s_{t-1}^\top - C, \tag{16}$$

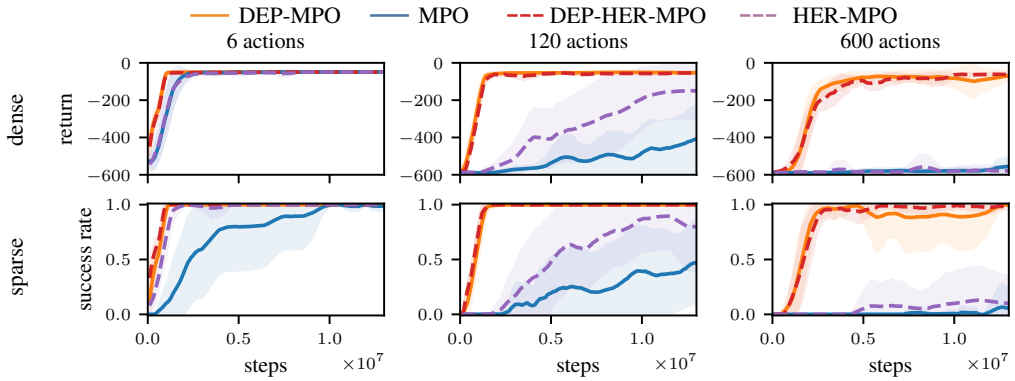


Figure 17: **DEP-MPO outperforms MPO in sparse and dense reward point-reaching for arm26 with all virtual action spaces.** Top: Learning performance for dense rewards. DEP-MPO strongly outperforms MPO, no significant movement learning could be detected for MPO with 600 actions. Bottom: Success rates for sparse reward reaching. While HER seems to increase the performance of the MPO baseline in most cases, the success rate only increases marginally for 600 actions, even after almost 1.5×10^7 steps. DEP-MPO solves all tasks with or without the addition of HER.

where $f(\cdot) : \mathbb{R}^m \to \mathbb{R}^n$ is an inverse model, relating future changes in the state to the change in action that caused them. First, we state the assumptions for this section:

1. We do not consider the normalization scheme for C.
2. We choose $f(\dot{s}_t) = \dot{s}_t$
3. A bias is not considered $h_t = 0$
4. The nonlinearity is approximated by the first term of a Taylor expansion $\tanh(x) \approx x$.
5. We consider C to instantaneously fulfill the update rule, without considering update dynamics.

Combining all the assumptions, we obtain:

$$C = \dot{s}_t \dot{s}_{t-1}^\top. \tag{17}$$

We will now consider a simple system with 1 sensor: the continuous action mountain car.

The original environment defines the RL state $s_t^{\text{RL}} = (x_t, \dot{x}_t)$. From this we extract the sensor information for DEP $s_t = x_t$, with $s \in \mathbb{R}$. In this 1-sensor formulation, the velocity correlation matrix in Eq. 17 is a scalar, as there is only 1 sensor. Consequently, also the C matrix is scalar. The resulting control equation is:

$$a_t = C s_t = \dot{s}_t \dot{s}_{t-1} s_t. \tag{18}$$

Let us assume an initial state of the car slightly to the right side of the valley, as in Fig. 18, with a positive initial velocity. States $s > 0$ are positions to the right of the valley bottom, while $s < 0$ to the left. The initial velocity will cause the car to move up the mountain. After some time, the velocity correlation will be $\dot{s}_t \dot{s}_{t-1} > 0$ with $s > 0$, leading to $C s_t = a_t > 0$. Logically, the car then starts pushing to the right, reinforcing the movement pattern and trying to increase its velocity. However, the task is set up such that the force is insufficient to directly go up the mountain. It will thus change the movement direction at some point and reverse due to gravity.

After changing direction, $\dot{s}_t \dot{s}_{t-1}$ will reverse sign as the *previous* velocity still points to the right, while the *current* velocity points to the left. Thus, $\dot{s}_t \dot{s}_{t-1} < 0$ with $s > 0$, and $C s_t = a_t < 0$. The car will consequently try to push into the negative direction, accelerating downwards. If the past sensor derivative $\dot{s}_{t-1}$ is defined as only one time step away, this trend will immediately reverse and the car will decelerate.

If, however, $\dot{s}_t$ is chosen to be not one time step apart from $\dot{s}_{t-1}$, but $\Delta t \in \mathbb{N}$ steps, then it will take several time steps until the sign reversal of $\dot{s}_t \dot{s}_{t-\Delta t}$ happens. In this intermittent regime, $\dot{s}_t \dot{s}_{t-\Delta t} < 0$ with $s > 0$, and $C s_t = a_t < 0$, pushing the car to the left, until the velocity correlation changes sign again.

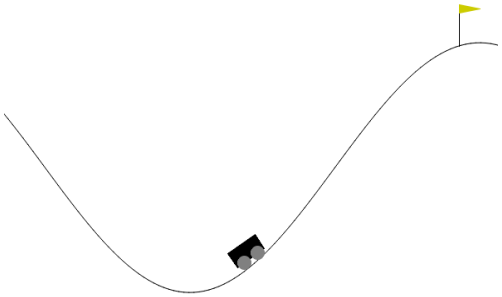


Figure 18: Continuous-action mountain car environment. To solve the task, the car must be brought to the top of the right hill. However, the motor is too weak for a direct approach. The solution involves pushing the car from left to right and vice-versa with the correct frequency to gather momentum and climb up the mountain.

If Δt is appropriately chosen, however, by the time the reversal happens, the car will have moved into the negative state region $s < 0$. In this new region, $\dot{s}_t \dot{s}_{t-\Delta t} > 0$ with $s < 0$, and $C s_t = a_t < 0$, which pushes the car further to the left and up the mountain, until the phenomenon repeats.

We offer simulation results of the described dynamics in Fig. 19. For this simulated example, we consider again the nonlinearity of $\tanh$ (Eq. 15), as it allows us to multiply C by a large constant and still satisfy the action limits of the environment. This is a necessary step as we omitted the normalization scheme for C. We thus consider $C = \tanh(\kappa \dot{s}_t \dot{s}_{t-\Delta t})$, with $\kappa \gg 1$.

To demonstrate the influence of Δt, we plot the results for two examples (Fig. 19). For $\Delta t = 27$, the system trajectory at time step ≈ 95 shows a reversal point. Before this reversal point, the position s is positive and so is the correlation $\dot{s}_t \dot{s}_{t-\Delta t}$, which leads to positive actions. Due to gravity and the weak motor, however, the car starts to move into the opposite direction. As there was a velocity sign change, we have $\dot{s}_t \dot{s}_{t-\Delta t} < 0$ while the position s_t is still positive, which yields a negative action: The car is accelerating downwards and builds up momentum, which, after a few repetitions, allows it to explore the environment.

The values of Δt for this example were chosen to yield good visualization. We observe full exploration of the mountain car system for $\Delta t \in \{5, ..., 28\}$.

This might seem like an overly simplified example, but it shows how DEP can increase the variance in a sensor value, even if a large number of actuators is associated to it. Empirical evidence additionally demonstrates that in high-dimensional systems, if all other DEP components are considered, DEP becomes less sensitive to the exact system dynamics and parameter specifications, generalizing easily to muscle-driven systems with over 120 muscles. Note that as the mountain car task contains an harmonic potential well, ubiquitous in many physical systems, the analysis might hold for a wide range of models, even considering elastic muscles with nonlinear spring elements.

C.8 Sensitivity to changes in p_{switch}

We performed an ablation study over the parameter p_{switch}, that controls the probability of switching from the RL policy to the DEP controller. Figure 20 shows that the training averaged success rates for DEP-MPO are much less sensitive to this hyperparameter for the humanreacher task than the average returns for ostrich-run. We conjecture that locomotion is inherently more unstable and that higher DEP probabilities cause the agent to fall down often, which hurts learning performance.

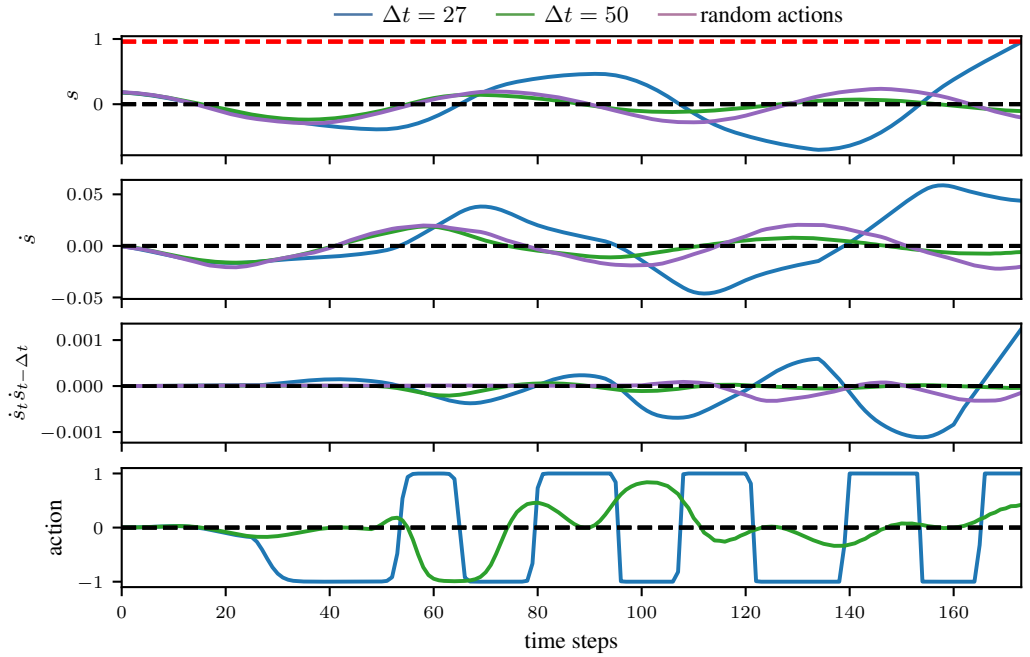


Figure 19: A simplified DEP rule in the mountain car environment (Moore, 1990). The red line marks the position threshold at which the task is solved. The black lines mark the zero point. Negative values of $\dot{s}_t \dot{s}_{t-\Delta t}$ mark intermittent regimes where the controller output reverses, eventually bringing the system into coherent motion. For $\Delta t = 27$, the reversal of $\dot{s}_t \dot{s}_{t-\Delta t}$ happens on a time scale that is able to excite the system, while the setting $\Delta t = 50$ is not able to induce sufficient exploration. The random actions are drawn from a standard Gaussian $\mathcal{N}(0, 1)$. We do not show the proposed actions for the Gaussian to keep the figure readable. The position s for the setting $\Delta t = 27$ clearly oscillates with larger and larger amplitudes over time, increasing the effective variance of the sensor value. These two values of Δt were chosen to yield good visualizations, we observe full exploration of the mountain car system for values $\Delta t \in \{5, ..., 28\}$.

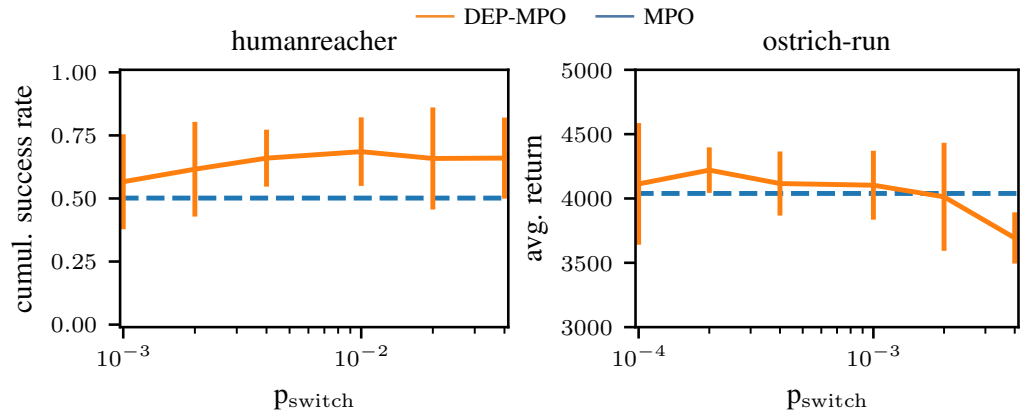


Figure 20: textbfAblation over the DEP probability with DEP-MPO for two tasks. Left: Humanreacher performance is overall robust for different settings of p_{switch}, while the benefit of DEP disappears for very small values. Right: Ostrich-run is more sensitive to the parameter, as locomotion is generally more unstable. Large DEP probabilities cause the agent to fall down very often. The horizontal line marks the average MPO performance without DEP. All values are averaged over 5 seeds. Note the log-axis.

Appendix B

Natural and Robust Walking using Reinforcement Learning without Demonstrations in High-Dimensional Musculoskeletal Models

Natural and Robust Walking using Reinforcement Learning without Demonstrations in High-Dimensional Musculoskeletal Models

Pierre Schumacher[1,2], Thomas Geijtenbeek[3], Vittorio Caggiano[4], Vikash Kumar[4], Syn Schmitt[5], Georg Martius[1,6], Daniel F. B. Haeufle[2,7]

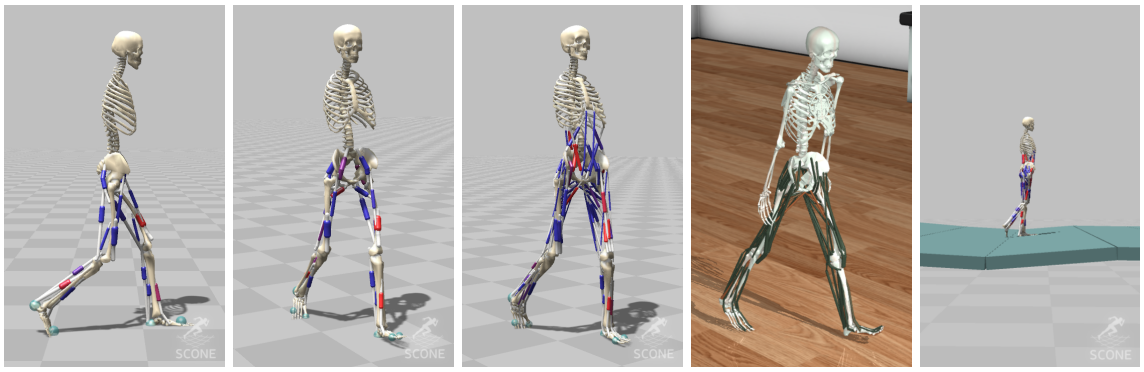


Fig. 1: **We achieve robust and energy-efficient natural walking with RL on a series of human models** Left to right: *H0918*, *H1622*, *H2190*, *MyoLeg* and an uneven terrain environment. Videos: https://sites.google.com/view/naturalwalkingrl

***Abstract*—Humans excel at robust bipedal walking in complex natural environments. In each step, they adequately tune the interaction of biomechanical muscle dynamics and neuronal signals to be robust against uncertainties in ground conditions. However, it is still not fully understood how the nervous system resolves the musculoskeletal redundancy to solve the multi-objective control problem considering stability, robustness, and energy efficiency. In computer simulations, energy minimization has been shown to be a successful optimization target, reproducing natural walking with trajectory optimization or reflex-based control methods. However, these methods focus on particular motions at a time and the resulting controllers are limited when compensating for perturbations. In robotics, reinforcement learning (RL) methods recently achieved highly stable (and efficient) locomotion on quadruped systems, but the generation of human-like walking with bipedal biomechanical models has required extensive use of expert data sets. This strong reliance on demonstrations often results in brittle policies and limits the application to new behaviors, especially considering the potential variety of movements for high-dimensional musculoskeletal models in 3D. Achieving natural locomotion with RL without sacrificing its incredible robustness might pave the way for a novel approach to studying human walking in complex natural environments.**

***Index Terms*—biomechanics, motor control, reinforcement learning, human walking**

[1]Max-Planck Institute for Intelligent Systems, Tübingen, Germany
[2]Hertie Institute for Clinical Brain Research and Center for Integrative Neuroscience, Tübingen, Germany
[3]Goatstream, The Netherlands
[4]Meta AI, New York, USA
[5]Institute for Modelling and Simulation of Biomechanical Systems, University of Stuttgart, Germany
[6]Department of Computer Science, University of Tübingen, Germany
[7]Institute of Computer Engineering, Heidelberg University, Germany

I. Introduction

The aim of this study is to demonstrate that RL methods can generate robust controllers for musculoskeletal models. The novelty of our work is that we do not try to achieve human-like behavior with RL by relying on kinematic data, but only on biologically plausible objectives in combination with realistic biomechanical constraints embedded into simulation engines. These evolutionary priors have the potential to be general enough to allow for the reproduction of natural gait, similar to the achievements of reflex control policies, but with the potential for generating diverse and robust behaviors under many different conditions.

We specifically propose a reward function restricted to metrics that are considered plausible objectives for biological organisms, while using experimental human data only to modify the relative importance of the different metrics, similar to [1]–[3]. This goes beyond previous works applying RL to biomechanical models which either study low-dimensional systems [4], make use of expert data [5] during training or learn unrealistic movements [6], [7]. We propose a reward function based only on walking speed, joint pain, and muscle effort, achieving periodic gaits that resemble human walking kinematics and ground reaction forces (GRF) closer than comparable RL approaches [8]–[11]. Furthermore, the learning approach generated walking in 4 different models and 2 simulation engines of differing biomechanical complexity and accuracy with an identical training protocol and without changing the reward function.

The simpler 2D and 3D models are comparable in complexity and almost reach the naturalism of existing optimal control- and reflex-based frameworks [12]–[14]. While their 80 or 90 muscle

counterparts are substantially more challenging to control, our approach still achieved gaits with kinematics and GRFs similar to experimental human data, albeit with more artifacts, potentially related to biomechanical modeling accuracy. Achieving gaits in these complex models is a step towards applications in rehabilitation, neuroscience, and computer graphics requiring high-dimensional models in complex environments. Striking is the robustness of the learned controllers exhibiting diverse stabilization strategies when faced with dynamic perturbations to an extent unseen in previous reflex-based controllers [12], [15]–[18]. As the used reward terms are considered plausible objectives for biological organisms, the general approach may also be applicable to different movements. Therefore, we believe that this approach is a useful starting point for the community showing that RL **is** a viable candidate to investigate the highly robust nature of complex human movements.

II. Results

Our framework is built upon the recently published DEP-RL [7] approach to learning feedback controllers for musculoskeletal systems. DEP-RL has been shown to achieve robust locomotion in several tasks, including running with a high-dimensional (120 muscles) bipedal ostrich model, by proposing a novel exploration scheme for overactuated systems. The learned behaviors, however, still exhibited unnatural artifacts, such as large co-contraction levels and excessive bending of several joints.

Here, we extend that work by introducing an adaptive reward function that accounts for biologically plausible incentives. These incentives result in gaits that resembling human walking much closer. Furthermore, the reward function is general enough to generate gaits across several models with up to 90 muscles in two and three dimensions and in simulators of differing biomechanical modeling accuracy **without** changes in the weighting of the incentives in the reward function. Only the network size was decreased for the low-dimensional models to benefit from the computational speed up.

A. Reward function

Building on previous work on gait optimization [14], we found that a natural gait can be achieved with RL by using objectives that incentivize:

1) learning to maintain a given speed without falling down,
2) minimizing effort, and
3) minimizing pain.

Thus, our reward function contains three main terms:

$$r = r_{\text{vel}} - c_{\text{effort}} - c_{\text{pain}}. \tag{1}$$

The first term specifies the external task the agent should solve. As we want the agent to move at a walking pace while keeping its balance, we chose the following objective:

$$r_{\text{vel}} = \begin{cases} \exp\left[-(v - v_{\text{target}})^2\right] & \text{if } v < v_{\text{target}} \\ 1 \quad \text{otherwise,} \end{cases} \tag{2}$$

where v is the center-of-mass velocity and the target velocity v_{target} is chosen to be $1.2\,\text{m/s}$, which is close to the average energetically optimal human walking speed [19]. The velocity reward is constant above the target velocity to improve the optimization of the auxiliary cost terms, inspired by a recent study on reward shaping in robotics [20].

Important for achieving natural human walking is the use of minimal muscle effort, as the literature suggests that energy efficiency is a key component of human locomotion [21], [22]:

$$c_{\text{effort}} = \alpha(t)\, a^3 + w_1\, (u - u_{\text{prev}})^2 + w_2\, N_{\text{active}} \tag{3}$$

where the first term penalizes muscle activity a [23], the second term incentivizes smoothness of muscle excitations u, and the third term N_{active} incentivizes a small number of active muscles (penalizing activity exceeding a certain value).

From a technical standpoint, it proved challenging to effectively minimize muscle activity. Using a strong cost scale that leads to energy-efficient walking later in training, causes a performance collapse when enabled from the start. We, therefore, chose an approach rooted in constrained optimization [24]. We propose an adaptation mechanism for the weighting parameter $\alpha(t)$, increasing the weight only when the agent performs well in the main task (r_{vel}) and decreasing it when this constraint is violated. Concretely, we measure the performance by the task return. The details are provided in Alg. 1, we marked the constrained optimization in blue.

This adaptive learning mechanism can be applied to each model and removes the need for hand-tuning of schedules. A change in reward function over time could, however, destabilize learning, as previously collected environment transitions are not reflective of the current effort cost anymore [25]. We, therefore, monitor the performance of the policy in the current environment, while the effort cost is only applied the moment when data is sampled from the replay buffer. This relabeling of previously collected data ensures that our off-policy algorithm can make efficient use of the full replay buffer.

Algorithm 1 Effort weight adaptation.

Require: threshold θ, smoothing β, change in adaptation rate $\Delta\alpha$, decay term $\lambda \in [0, 1]$
$r_{\text{mean}} \leftarrow 0$, $\alpha_t \leftarrow 0$, $\text{s}_{\text{mean}} \leftarrow 0$
while True **do**
  $r \leftarrow$ train_episode() ▷ return from episode
  $r_{\text{mean}} \leftarrow \beta\, r_{\text{mean}} + (1 - \beta)\, r$
  if $r_{\text{mean}} > \theta$ **and** $\text{s}_{\text{mean}} < 0.5$ **then**
    $\Delta\alpha \leftarrow \lambda \cdot \Delta\alpha$ ▷ performance newly high
    ▷ slow down adaptation
  else if $r_{\text{mean}} > \theta$ **and** $\text{s}_{\text{mean}} > 0.5$ **then**
    $\alpha_{t+1} \leftarrow \alpha_t + \Delta\alpha$ ▷ performance high for long
  else
    $\alpha_{t+1} \leftarrow \alpha_t - \Delta\alpha$ ▷ performance too low
  end if
  $c_{\text{target}} \leftarrow \begin{cases} 1 & \textbf{if } r_{\text{mean}} > \theta \\ 0 & \textbf{otherwise} \end{cases}$
  $c_{\text{mean}} \leftarrow \beta c_{\text{mean}} + (1 - \beta)\, c_{\text{target}}$
end while

The third term c_{pain} is necessary to prevent unnatural optima. One striking example is the over-use of mechanical forces of

TABLE I: All used models. Trunk means that the trunk and the pelvis of the model can move separately. Toes means that the toes and the rest of the foot can move separately. The designation 3D marks models that can walk in full 3D, as opposed to planar movements.

Model	# DOFs	# muscles	3D	trunk	toes	engine
H0918	9	18	✗	✗	✗	Hyfydy
H1622	16	22	✓	✗	✗	Hyfydy
H2190	21	90	✓	✓	✗	Hyfydy
MyoLeg	20	80	✓	✗	✓	MuJoCo

the joint limits (e.g. massive knee over-extension) to keep a straight leg while minimizing muscle activity. As this is clearly unnatural behavior, we include objectives that account for the notion of pain:

$$c_{\text{pain}} = w_3 \sum_i \tau_i^{\text{lim}} + w_4 \sum_j F_j^{\text{GRF}}, \tag{4}$$

where τ_i^{lim} is the torque with which the joint angle limit of joint i is violated (joint-limit pain) and F_j^{GRF} is the vertical ground reaction force (GRF) for foot j (joint-loading pain). We only penalize GRFs if they exceed 1.2 times the model's body weight [26], [27], such that all pain cost terms vanish close to the natural gait and do not further bias the solution.

We tuned the cost term weights ω_i for $i \in \{1, ..., 4\}$ by first separating the kinematic data into gait cycles for each leg, starting and ending when the respective foot touches the ground. The resulting data is then averaged over all gait cycles recorded from both legs. The average trajectory is finally compared to its equivalent obtained from experimental human data. The experimental match, defined as the fraction of the gait cycle for which the average simulated trajectory overlaps within the standard deviation of experimental data, serves as an optimization metric for our cost terms. We note that the coefficients are identical across all joints and muscles, and stress that no human data was used **during** the learning process, but only to find weighting coefficients. This procedure is similar to [1], with the difference that we search for values that work across a range of models, instead of optimally for one model.

Finally, we initialize the models with a randomized initial state that starts with one elevated leg, while we also clip all muscle excitations to lie between 0 and 0.5 to further reduce muscle effort and mitigate asymmetries caused by the initial state distribution.

B. Models

With the reward function and the RL approach described above, we are able to learn robust control policies for several models of human walking, with varying complexity, and across two different simulation engines with different levels of biomechanical accuracy (see Fig. 1):

H0918 A planar Hyfydy model with 9 degrees-of-freedom (DOFs) and 18 muscles, based on [28].

H1622 A 3D Hyfydy model with 16 DOFs and 22 muscles, based on [28].

H2190 A 3D Hyfydy model with 21 DOFs and 90 muscles, and articulation between the otherwise rigid pelvis and torso, based on [28]–[30].

MyoLeg A 3D MuJoCo model with 20 DOFs and 80 muscles, based on [29]. As for the *H0918* and *H1622* models, the pelvis and torso are one rigid body part, while each foot contains articulated toes (all five toes are joined into one body segment). See Tab. I for a summary of the models.

C. Simulation engine

The simulation engines used for each model are indicated in the description and are either: *a)* Hyfydy [31], which was used via the SCONE Python API [27], or *b)* MuJoCo, which was used via the MyoSuite [32] environment. We chose these two engines, to highlight the versatility of our approach but also to bridge two communities: biomechanics and RL.

Hyfydy is an engine built for biomechanical accuracy. It is closely related to the well-established OpenSim [33] framework, matching its level of detail in muscle and deformation-based contact-force models while providing increased computational performance. MuJoCo is a fast simulation framework widely used in the robotics and RL community. It also offers a simplified muscle model with rigid tendons and resolves contact forces using the convex Gauss Principle. The MyoSuite [32] builds on this framework, allowing for the development of high-dimensional muscle-control models which have recently gained a lot of interest from the RL community [11], [34], [35]. Both engines achieve the required computational speed to train control policies for these high-dimensional models in under a day. See Suppl. A for more technical details.

D. Learned behaviors

We first show that with our framework, we can train agents across 4 different models to produce walking gaits with the same training approach and reward function. In Figure 2 we compare the resulting gait kinematics against experimental data, included in the SCONE software [27], [36]. Kinematics are shown for 5 rollouts of the most human-like policy checkpoint that was achieved over the entire training run over 10 random seeds, averaged over all gait cycles of both legs in a 10 s walk[1].

The results for the planar *H0918* and the 3D *H1622* model look very similar to the experimental data, even though the ankle kinematics differ slightly. While the agents achieve the most human-like gaits here, the models are also of limited complexity and applicability, compared to the high-dimensional systems, *H2190* and *MyoLeg*. As seen in Tab. II and Fig. 2, our approach still achieves periodic gaits resembling human kinematics with the difficult-to-control 80 and 90 muscle models, even though they contain more artifacts. The *H2190*-agent exhibits less knee flexion and the *MyoLeg*-agent lacks the double-peaked GRF structure; it also exhibits differences in the hip kinematics. Overall, the behavior of the *H2190* model appears more natural than the one produced with the *MyoLeg* model, see also the discussion in Sec. III and the supplementary videos.

Nevertheless, Tab. II shows that RL gaits not only approximate human walking but are also robust and energy-efficient across all models, without changes in the reward function and only minimal changes in the hyperparameters of the RL method.

[1]For videos, see: https://sites.google.com/view/naturalwalkingrl

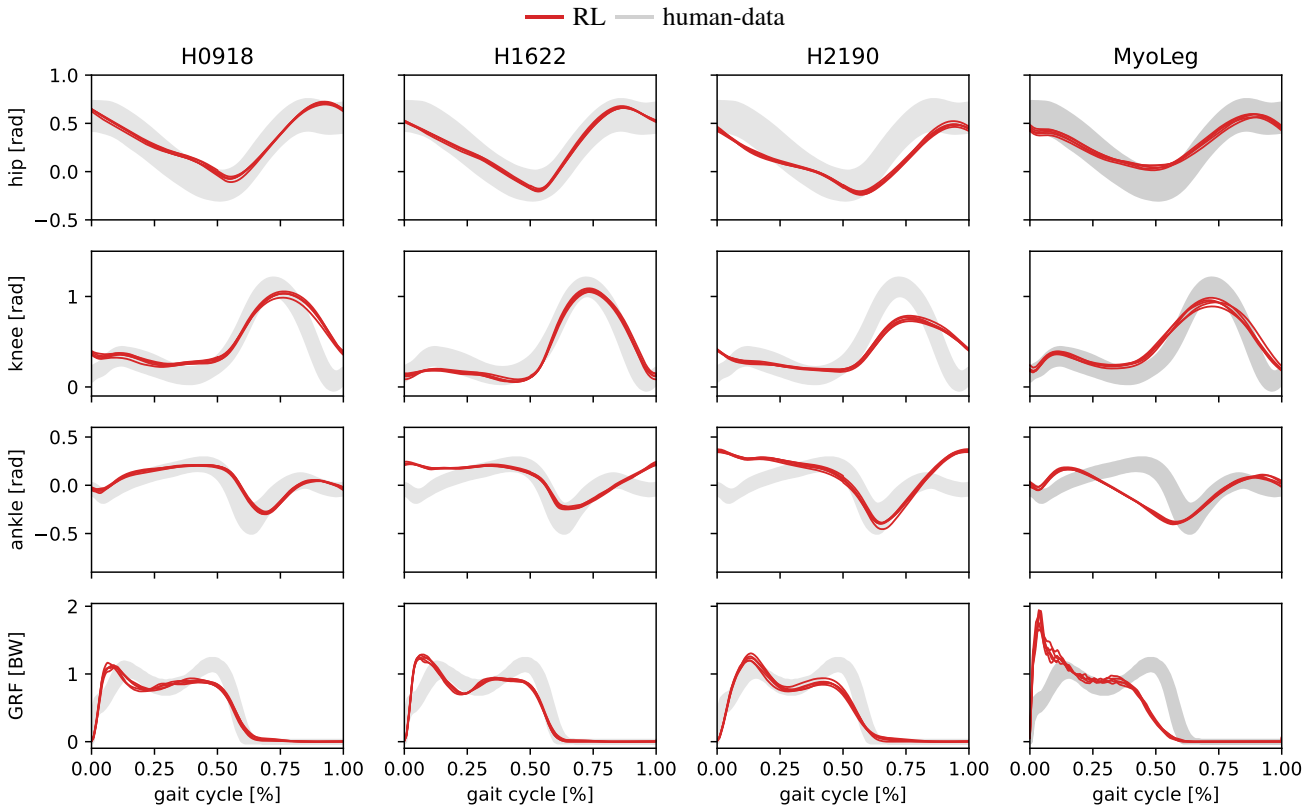


Fig. 2: **Gait kinematics for RL agents for all models** Shown are the hip, knee, ankle and GRF values averaged over 5 rollouts of 10 s walking on flat ground. We excluded rollouts that did not achieve the whole episode length to clearly highlight the achieved kinematics. While there are slight discrepancies between experimental data (grey) and the RL behaviors (red), especially for the high-dimensional models, the proposed reward function provides a strong starting point for researchers aiming to create robust and natural controllers for high-dimensional musculoskeletal systems. Also, see the videos on the website.

We provide the training curves and additional metrics for 10 random seeds in Suppl. B.

In order to probe the robustness of our controllers, we perform roll-outs on uneven terrain, which was **not** seen during training. The entire training procedure was performed with flat ground. The generated terrain contains 10 tiles of 1 m length with random slopes of $\pm 5^\circ$ and is fixed for all evaluations. The behavior of the planar *H0918*-model is compared against a popular reflex-based controller as an illustrative example, adapted from [12], [27] and included with the SCONE software. We were only able to use this simple reflex-based controller with the *H0918* model, as it did not produce stable gaits with the other models. We train 5 reflex-based controllers with different initializations until convergence, while we use the most natural RL policy for each model and perform 20 roll-outs with randomized initial states to test the robustness. We chose this approach as reflex-based controllers are sensitive to the initial simulation state; different roll-outs would be almost identical if we would have to use similar starting states.

While both approaches adequately match human kinematics with low energy consumption in the planar case, the reflex-based controller produces more natural gaits. However, when exposed to uneven terrain, the RL agent achieves an average distance of 10.42 m, which shows that it is much more robust than the reflex controller with an average distance of 2.46 m, see Tab. II. Both controllers also induce similar average muscle activities over the gait cycle, with the RL agent inducing less smooth activity, shown in Fig. 3.

With the same framework, we were also able to train agents to learn maximum speed running, by simply using the achieved velocity as the velocity reward in our reward function. Additionally, the action clipping and effort costs were omitted, as energy consumption is less critical for short maximum performance tasks. See Suppl. C for these results.

As a showcase of the extreme robustness of the RL agents, we generated a difficult drawbridge-terrain task with moveable environment elements that present dynamic perturbations, see Fig. 6b. We test the robustness of *H1622* and *H2190* RL controllers in this scenario, even though they were only ever trained on **flat** ground, and observe remarkable stability across the task. We report the data in Tab. III and in the videos.

Note that we tried several alternatives to our approach which yielded worse results. We performed experiments with different reward terms such as a constant instead of an adaptive effort term, with metabolic energy costs [8] or with a cost of transport [37], [38] reward. Even though these terms sometimes lead to small muscle activity during execution, the kinematics were further away from human data. We conjecture that energy

TABLE II: The table shows the average cubic muscle activity (effort), the percentage match with human experimental data (exp. match), and the average distance walked on the rough terrain. Note that the exp. match metric measures the percentage of the gait cycle during which the trajectory perfectly lies inside the standard deviation of the experimental data. Even relatively natural gaits can still achieve a low metric if the angles are slightly shifted.

controller	system	avg. effort	experimental match	avg. distance [m]
reflex	H0918	$0.041 \pm 3 \times 10^{-3}$	0.68 ± 0.08	2.46 ± 0.98
RL	H0918	$0.013 \pm 3 \times 10^{-4}$	0.67 ± 0.03	10.42 ± 0.94
RL	H1622	$0.015 \pm 2 \times 10^{-3}$	0.73 ± 0.01	5.6 ± 0.99
RL	H2190	$0.017 \pm 1 \times 10^{-5}$	0.50 ± 0.01	10.59 ± 2.51
RL	MyoLeg	$0.013 \pm 2 \times 10^{-4}$	0.43 ± 0.05	n.a.

minimization is not enough of an incentive for human-like gait if the learning algorithm is as flexible as an RL agent. See also Fig. 8 for ablations of our reward function.

Larger effort term exponents, penalization of contacts between limbs or angle-based joint limit violation costs did not lead to better behavior. The prescription of hip movement at a certain frequency (step clock), keeping certain joint angles in pre-specified positions or minimizing torso rotation helped to achieve stable gaits, but prevented effort minimization and did not lead to natural kinematics.

III. Discussion

As the human biomechanical system is highly redundant, there are many possible solutions to walking at a defined speed. There exists strong evidence that natural human walking is in part driven by energy-efficiency [39]. Optimal control approaches have shown that natural walking kinematics can be achieved if energy optimality is considered in the cost function.[2]

However, in most RL approaches, energy consumption is either ignored, or only static action regularization is used, which affects learning but does not yield truly efficient behavior. By introducing a single reward term schedule that adapts the weighting of the energy term in the reward function depending on the current performance, we achieved energy-efficient gaits with more natural kinematics also in RL. Moreover, the adaptation algorithm (Alg. 1) and all other reward terms and their weighting coefficients are general enough to work—without any changes—across 2D and 3D models with different numbers of muscles and even different levels of biomechanical modeling accuracy.

This is a significant step towards finding a general reward function and framework to generate natural and robust movements with RL in muscle-driven systems. Other RL frameworks that do achieve natural muscle utilization either consider low-dimensional systems [9] or strongly rely on motion capture data [41] to render the learning problem feasible. Our approach works **without** the use of motion capture data during training and with few and very general reward terms and therefore may generalize better to other movements.

[2]Some also suggest that muscle fatigue could be the driving factor to explain the experimentally observed kinematic patterns [40].

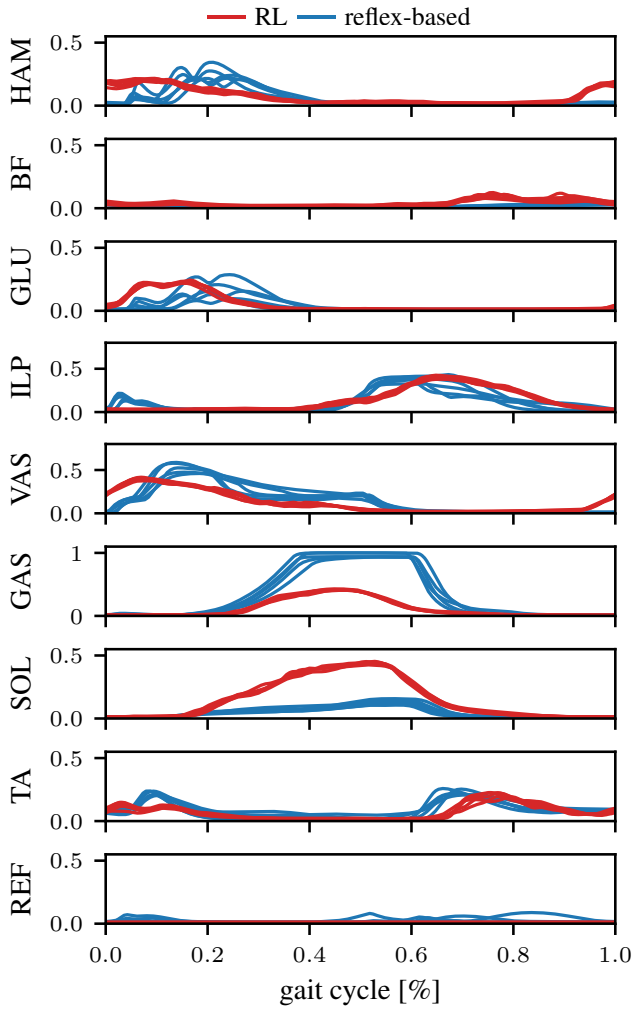


Fig. 3: **Muscle activation for RL agent and reflex-based controller.** We compare muscle activities for two controller types for natural walking with the *H0918* model. The activity for the RL agent has been clipped to 0.5. We use 5 roll-outs of the most natural RL policy and 5 reflex-based controllers that were optimized until convergence. The initial state for the RL agent is randomized, which would cause collapse with the reflex controllers, as they are sensitive to the initial state.

In our opinion, the only comparable work is by Weng et al. [4]. They achieved human-level kinematics on a planar human model with 18 muscles, by crafting a multi-stage learning curriculum affecting the weighting of seven reward terms. As this learning curriculum contains model-specific reward terms and adaptation procedures, we speculate that it would have to be hand-tuned for different models.

While our approach achieved higher robustness than reflex-based controllers and kinematics closer to natural walking than previous demonstration-free RL approaches, several discrepancies to natural walking remain, see Fig. 2 and supplementary videos. The low-dimensional models (*H0918* and *H1622*) in general do not present proper ankle rolling, while the high-

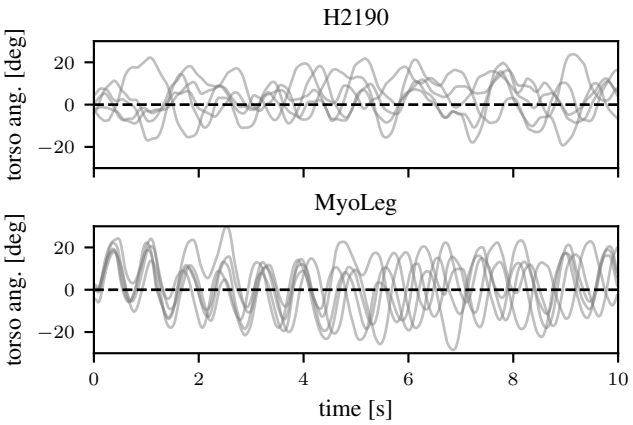


Fig. 4: **Torso oscillations during walking.** We show the torso angle with the vertical axis for 5 rollouts of 10 s for the *H2190*- and the *MyoLeg*-models. The *MyoLeg* presents stronger lateral oscillations. The dashed line shows a straight torso posture.

dimensional models (*H2190* and *MyoLeg*) exhibit less passive leg-swing in the swing phase of the gait.

The behavior of the *MyoLeg* model deviates stronger from human data than the *H2190*, although they are similar in terms of complexity. This is most prominent in the ankle kinematics and the lack of double-peak structure in the GRFs in the *MyoLeg* model. We also observed a tendency for unnatural lateral torso oscillations with the *MyoLeg* model, see Fig. 4 and the videos.

These differences in behaviors could be related to the model parametrization, as the *MyoLeg* uses a different muscle geometry from *H2190* and includes mesh-based contact dynamics, which might increase learning difficulty. Alternatively, the more elaborate biomechanical features in Hyfydy, such as elastic tendons [42], non-linear foot-ground contact mechanics [43], variable pennation angles [44] or error-controlled integration, could account for the increased realism of the behaviors with the Hyfydy models. See Suppl. A for more details on the simulation engines.

Research on the contribution of biomechanical structures to the emergence of natural movement [45]–[48] suggest that, in addition to the learning method and reward function, the biomechanical structures and modeling choices may play a crucial role in the accurate reproduction of human gait. This seems a plausible explanation for the increased realism in the Hyfydy models, as previous observations in predictive simulations suggest that e.g. an elastic tendon is beneficial for natural gait [8], [12], [14]. We regard this as one interesting area of future research, which could help us better understand the fundamentals of the interaction between biomechanics and neuronal control in human locomotion.

In conclusion, we achieved highly robust walking approaching human-like kinematics and ground reaction forces. While a better degree of accuracy was achieved in simpler models, we provide first promising results for difficult-to-control 80 and 90 muscle models which are of high interest for applications in rehabilitation, neuroscience, and computer graphics. Learning with the proposed reward function and RL framework allows for these results across several models of differing complexity and biomechanical modeling accuracy with only minimal changes in the hyperparameters of the method. We hope that this inspires researchers from both the biomechanics and the RL community to further improve on our approach and to develop tools to unravel the fundamentals of the generation of complex, robust, and energy-efficient human movement.

ACKNOWLEDGMENT

Pierre Schumacher was supported by the International Max Planck Research School for Intelligent Systems (IMPRS-IS). This work was supported by the Cyber Valley Research Fund (CyVy-RF-2020-11 to DH and GM).

REFERENCES

[1] B. Berret, E. Chiovetto, F. Nori, and T. Pozzo, "Evidence for composite cost functions in arm movement planning: An inverse optimal control approach," *PLOS Computational Biology*, vol. 7, no. 10, pp. 1–18, 10 2011. [Online]. Available: https://doi.org/10.1371/journal.pcbi.1002183

[2] X. B. Peng, Z. Ma, P. Abbeel, S. Levine, and A. Kanazawa, "Amp: Adversarial motion priors for stylized physics-based character control," *ACM Trans. Graph.*, vol. 40, no. 4, Jul. 2021. [Online]. Available: http://doi.acm.org/10.1145/3450626.3459670

[3] J. Weng, E. Hashemi, and A. Arami, "Human gait cost function varies with walking speed: An inverse optimal control study," *IEEE Robotics and Automation Letters*, vol. 8, no. 8, pp. 4777–4784, 2023.

[4] ——, "Natural walking with musculoskeletal models using deep reinforcement learning," *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 4156–4162, 2021.

[5] J. Park, S. Min, P. S. Chang, J. Lee, M. S. Park, and J. Lee, "Generative gaitnet," in *ACM SIGGRAPH 2022 Conference Proceedings*, ser. SIGGRAPH '22. New York, NY, USA: Association for Computing Machinery, 2022. [Online]. Available: https://doi.org/10.1145/3528233.3530717

[6] J. Xu, M. Macklin, V. Makoviychuk, Y. Narang, A. Garg, F. Ramos, and W. Matusik, "Accelerated policy learning with parallel differentiable simulation," in *International Conference on Learning Representations*, 2022. [Online]. Available: https://openreview.net/forum?id=ZSKRQMvttc

[7] P. Schumacher, D. Haeufle, D. Büchler, S. Schmitt, and G. Martius, "DEP-RL: Embodied exploration for reinforcement learning in overactuated and musculoskeletal systems," in *The Eleventh International Conference on Learning Representations*, 2023. [Online]. Available: https://openreview.net/forum?id=C-xa_D3oTj6

[8] J. Wang, S. Hamner, S. Delp, and V. Koltun, "Optimizing Locomotion Controllers Using Biologically-Based Actuators and Objectives," *ACM Trans. on Graphics*, vol. 31, no. 4, p. 25, 2012.

[9] Ł. Kidziński, S. P. Mohanty, C. F. Ong, Z. Huang, S. Zhou, A. Pechenko, A. Stelmaszczyk, P. Jarosik, M. Pavlov, S. Kolesnikov, S. Plis, Z. Chen, Z. Zhang, J. Chen, J. Shi, Z. Zheng, C. Yuan, Z. Lin, H. Michalewski, P. Milos, B. Osinski, A. Melnik, M. Schilling, H. Ritter, S. F. Carroll, J. Hicks, S. Levine, M. Salathé, and S. Delp, "Learning to run challenge solutions: Adapting reinforcement learning methods for neuromusculoskeletal environments," in *The NIPS '17 Competition: Building Intelligent Systems*, S. Escalera and M. Weimer, Eds. Cham: Springer International Publishing, 2018, pp. 121–153.

[10] S. Song, Ł. Kidziński, X. B. Peng, C. Ong, J. Hicks, S. Levine, C. G. Atkeson, and S. L. Delp, "Deep reinforcement learning for modeling human locomotion control in neuromechanical simulation," *Journal of NeuroEngineering and Rehabilitation*, vol. 18, no. 1, p. 126, Aug 2021. [Online]. Available: https://doi.org/10.1186/s12984-021-00919-y

[11] C. Berg, V. Caggiano, and V. Kumar, "Sar: Generalization of physiological agility and dexterity via synergistic action representation," 2023.

[12] H. Geyer and H. Herr, "A muscle-reflex model that encodes principles of legged mechanics produces human walking dynamics and muscle activities," *IEEE Trans Neural Syst Rehabil Eng*, vol. 18, no. 3, pp. 263–273, Jun 2010.

[13] S. Song and H. Geyer, "Generalization of a muscle-reflex control model to 3D walking," in *2013 35th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*. IEEE, Jul. 2013.

[14] T. Geijtenbeek, M. van de Panne, and A. F. van der Stappen, "Flexible muscle-based locomotion for bipedal creatures," *ACM Transactions on Graphics*, vol. 32, no. 6, 2013.

[15] S. Song and H. Geyer, "Evaluation of a Neuromechanical Walking Control Model Using Disturbance Experiments," *Frontiers in Computational Neuroscience*, vol. 11, no. 3, 2017.

[16] D. F. B. Haeufle, B. Schmortte, H. Geyer, R. Müller, and . Schmitt, Syn, "The Benefit of Combining Neuronal Feedback and Feed-Forward Control for Robustness in Step Down Perturbations of Simulated Human Walking Depends on the Muscle Function," *Frontiers in Computational Neuroscience*, vol. 12, no. 80, 2018.

[17] R. Ramadan, H. Geyer, J. Jeka, G. Schöner, and H. Reimann, "A neuromuscular model of human locomotion combines spinal reflex circuits with voluntary movements," *Scientific Reports*, vol. 12, no. 1, may 2022.

[18] L. Schreff, D. F. B. Haeufle, J. Vielemeyer, and R. Müller, "Evaluating anticipatory control strategies for their capability to cope with step-down perturbations in computer simulations of human walking," *Scientific Reports*, vol. 12, no. 1, jun 2022.

[19] B. J. Mohler, W. B. Thompson, S. H. Creem-Regehr, H. L. Pick, Jr, and W. H. Warren, Jr, "Visual flow influences gait transition speed and preferred walking speed," *Exp. Brain Res.*, vol. 181, no. 2, pp. 221–228, Aug. 2007.

[20] N. Rudin, D. Hoeller, M. Bjelonic, and M. Hutter, "Advanced skills by learning locomotion and local navigation end-to-end," in *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2022, pp. 2497–2503.

[21] D. Abe, Y. Fukuoka, and M. Horiuchi, "Economical speed and energetically optimal transition speed evaluated by gross and net oxygen cost of transport at different gradients," *PLOS ONE*, vol. 10, no. 9, pp. 1–14, 09 2015. [Online]. Available: https://doi.org/10.1371/journal.pone.0138154

[22] P. C. Raffalt, M. K. Guul, A. N. Nielsen, S. Puthusserypady, and T. Alkjær, "Economy, movement dynamics, and muscle activity of human walking at different speeds," *Scientific Reports*, vol. 7, no. 1, p. 43986, Mar 2017. [Online]. Available: https://doi.org/10.1038/srep43986

[23] M. Ackermann and A. J. van den Bogert, "Optimality principles for model-based prediction of human gait," *Journal of Biomechanics*, vol. 43, no. 6, pp. 1055–1060, 2010. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0021929009007210

[24] T. Zahavy, Y. Schroecker, F. Behbahani, K. Baumli, S. Flennerhag, S. Hou, and S. Singh, "Discovering policies with DOMiNO: Diversity optimization maintaining near optimality," in *The Eleventh International Conference on Learning Representations*, 2023. [Online]. Available: https://openreview.net/forum?id=kjkdzBW3b8p

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

[26] J. NILSSON and A. THORSTENSSON, "Ground reaction forces at different speeds of human walking and running," *Acta Physiologica Scandinavica*, vol. 136, no. 2, pp. 217–227, 1989. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1111/j.1748-1716.1989.tb08655.x

[27] T. Geijtenbeek, "SCONE: Open Source Software for Predictive Simulation of Biological Motion," *Journal of Open Source Software*, vol. 4, no. 38, p. 1421, 2019. [Online]. Available: https://scone.software

[28] S. L. Delp, J. P. Loan, M. G. Hoy, and F. E. Zajac, "An interactive Graphics-Based Model of the Lower Extremity to Study Orthopaedic Surgical Procedures," pp. 757 – 767, 1990.

[29] A. Rajagopal, C. Dembia, M. DeMers, D. Delp, J. Hicks, and S. Delp, "Full body musculoskeletal model for muscle-driven simulation of human gait," *IEEE Transactions on Biomedical Engineering*, vol. 63, no. 10, pp. 2068–2079, 2016.

[30] M. Christophy, N. A. Faruk Senan, J. C. Lotz, and O. M. O'Reilly, "A musculoskeletal model for the lumbar spine." *Biomechanics and modeling in mechanobiology*, vol. 11, no. 1-2, pp. 19–34, jan 2012.

[31] T. Geijtenbeek, "The Hyfydy simulation software," 11 2021. [Online]. Available: https://hyfydy.com

[32] V. Caggiano, H. Wang, G. Durandau, M. Sartori, and V. Kumar, "Myosuite – a contact-rich simulation suite for musculoskeletal motor control," https://github.com/facebookresearch/myosuite, 2022. [Online]. Available: https://sites.google.com/view/myosuite

[33] A. Seth *et al.*, "Opensim: Simulating musculoskeletal dynamics and neuromuscular control to study human and animal movement," *PLoS computational biology*, vol. 14, no. 7, p. e1006223, 2018.

[34] V. Caggiano, S. Dasari, and V. Kumar. MyoDex: A Generalizable Prior for Dexterous Manipulation. [Online]. Available: https://openreview.net/forum?id=iYBTiYzN0A

[35] A. S. Chiappa, A. M. Vargas, A. Z. Huang, and A. Mathis, "Latent exploration for reinforcement learning," 2023.

[36] G. Bovi, M. Rabuffetti, P. Mazzoleni, and M. Ferrarin, "A multiple-task gait analysis approach: Kinematic, kinetic and EMG reference data for healthy young and adult subjects," *Gait & Posture*, vol. 33, no. 1, pp. 6–13, jan 2011.

[37] K. Veerkamp, N. Waterval, T. Geijtenbeek, C. Carty, D. Lloyd, J. Harlaar, and M. van der Krogt, "Evaluating cost function criteria in predicting healthy gait," *Journal of Biomechanics*, vol. 123, p. 110530, 2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0021929021003110

[38] A. Mastrogeorgiou, A. Papatheodorou, K. Koutsoukis, and E. Papadopoulos, "Learning energy-efficient trotting for legged robots," in *Robotics in Natural Settings*. Cham: Springer International Publishing, 2023, pp. 204–215.

[39] J. Selinger, S. O'Connor, J. Wong, and J. Donelan, "Humans can continuously optimize energetic cost during walking," *Current Biology*, vol. 25, no. 18, pp. 2452–2456, 2015. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0960982215009586

[40] M. Ackermann and A. J. van den Bogert, "Optimality principles for model-based prediction of human gait," *Journal of Biomechanics*, vol. 43, no. 6, pp. 1055–1060, apr 2010.

[41] S. Lee, M. Park, K. Lee, and J. Lee, "Scalable muscle-actuated human simulation and control," *ACM Trans. Graph.*, vol. 38, no. 4, jul 2019. [Online]. Available: https://doi.org/10.1145/3306346.3322972

[42] M. Ishikawa, P. V. Komi, M. J. Grey, V. Lepola, and G.-P. Bruggemann, "Muscle-tendon interaction and elastic energy usage in human walking," *Journal of Applied Physiology*, vol. 99, no. 2, pp. 603–608, 2005, pMID: 15845776. [Online]. Available: https://doi.org/10.1152/japplphysiol.00189.2005

[43] L. Saraiva, M. Rodrigues da Silva, F. Marques, M. Tavares da Silva, and P. Flores, "A review on foot-ground contact modeling strategies for human motion analysis," *Mechanism and Machine Theory*, vol. 177, p. 105046, 2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0094114X22002932

[44] R. Sopher, A. Amis, D. Davies, and J. Jeffers, "The influence of muscle pennation angle and cross-sectional area on contact forces in the ankle joint," *The Journal of Strain Analysis for Engineering Design*, vol. 52, 09 2016.

[45] K. G. Gerritsen, A. J. van den Bogert, M. Hulliger, and . Zernicke, Ronald F, "Intrinsic Muscle Properties Facilitate Locomotor Control - A Computer Simulation Study," *Motor Control*, vol. 2, no. 3, pp. 206–220, jul 1998.

[46] D. F. Haeufle, S. Grimmer, and . Seyfarth, A., "The role of intrinsic muscle properties for stable hopping - stability is achieved by the force-velocity relation," *Bioinspiration & Biomimetics*, vol. 5, no. 1, p. 016004, 2010.

[47] C. T. John, F. C. Anderson, J. S. Higginson, and S. L. Delp, "Stabilisation of walking by intrinsic muscle properties revealed in a three-dimensional muscle-driven simulation." *Computer methods in biomechanics and biomedical engineering*, vol. 16, no. 4, pp. 451–62, apr 2013.

[48] I. Wochner, P. Schumacher, G. Martius, D. Büchler, S. Schmitt, and D. Haeufle, "Learning with muscles: Benefits for data-efficiency and robustness in anthropomorphic tasks," in *Proceedings of The 6th Conference on Robot Learning*, ser. Proceedings of Machine Learning Research, K. Liu, D. Kulic, and J. Ichnowski, Eds., vol. 205. PMLR, 14–18 Dec 2023, pp. 1178–1188.

[49] M. Millard, T. Uchida, A. Seth, and S. L. Delp, "Flexing computational muscle: modeling and simulation of musculotendon dynamics." *Journal of biomechanical engineering*, vol. 135, no. 2, p. 021005, feb 2013.

[50] K. H. Hunt and F. R. E. Crossley, "Coefficient of Restitution Interpreted as Damping in Vibroimpact," *Journal of Applied Mechanics*, vol. 42, no. 2, p. 440, jun 1975.

[51] M. a. Sherman, A. Seth, and S. L. Delp, "Simbody: multibody dynamics for biomedical research," *Procedia IUTAM*, vol. 2, pp. 241–261, jan 2011.

[52] S. R. Hamner and S. L. Delp, "Muscle contributions to fore-aft and vertical body mass center accelerations over a range of running speeds," *Journal of Biomechanics*, vol. 46, no. 4, pp. 780–787, 2013. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0021929012006768

[53] F. Pardo, "Tonic: A deep reinforcement learning library for fast prototyping and benchmarking," *arXiv preprint arXiv:2011.07537*, 2020.

APPENDIX

A. Simulation Engines

The Hyfydy and MuJoCo simulation engines differ in these key areas:

Musculotendon dynamics The muscle model in Hyfydy is based on Millard at el. [49] and includes elastic tendons, muscle pennation, and muscle fiber damping. The MuJoCo muscle model is based on a simplified Hill-type model, parameterized to match existing OpenSim models [32], and supports only rigid tendons and does not include variable pennation angles.

Contact forces Hyfydy uses the Hunt-Crossly [50] contact model with non-linear damping to generate contact forces, with a friction cone based on dynamic, static, and viscous friction coefficients [51]. MuJoCo contacts are rigid, with a friction pyramid instead of a cone, and without separate coefficients for dynamic and viscous friction.

Contact geometry The MuJoCo model uses a convex mesh for foot geometry, while in the Hyfydy models the foot geometry is approximated using three contact spheres.

Integration step size Hyfydy uses an error-controlled integrator with variable step size, while MuJoCo uses a fixed step size and no error control. The average simulation step size in Hyfydy is around 0.00014s (7000 Hz) for the H2190 model, compared to the fixed MyoSuite step size of 0.001s (1000 Hz) for the *MyoLeg* model.

B. Training Curves

Here we present more detailed results about the training evolution in Fig. 5. We plot the experimental match percentage between the collected gait-cycle averaged data and experimental human data, the muscle-averaged effort, the training returns, and the weight that the effort-reward term has over training. This weight is adapted over time and depends on the agent's performance. It increases slower for the complex models and saturates at smaller values. It can also be seen that the returns for the *MyoLeg* are generally smaller than for the other models. We observed that there was more variance over training and over different seeds for the *MyoLeg*-agent, leading to much smaller averaged returns. It was still possible to find a training checkpoint that achieved robust, close-to human-like walking for this model.

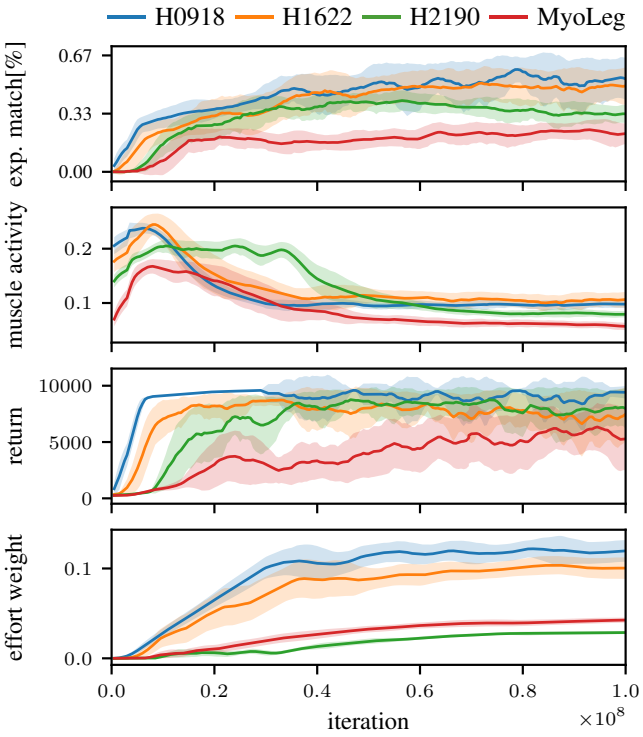


Fig. 5: **Training curves for the walking task.** We present the evolution of the match with experimental data (exp. match), the averaged muscle activity, the task performance, and the effort cost weight. The cost weight increases more slowly for the more complex models, showing its adaptive nature.

TABLE III: Maximum running velocity for different models, expressed in $^{m}/_{s}$ and total achieved distance in the rough terrain environment. We only show the maximum speed over 20 roll-outs for each model to show the largest velocity that we were able to achieve. For the rough terrain, we also record 20 roll-outs for each agent. We do not perform this experiment for the *H0918* model, as the 3D nature of the terrain is not applicable to it, and we do not apply it to the *MyoLeg* model, as the terrain has not been implemented in the MuJoCo simulator.

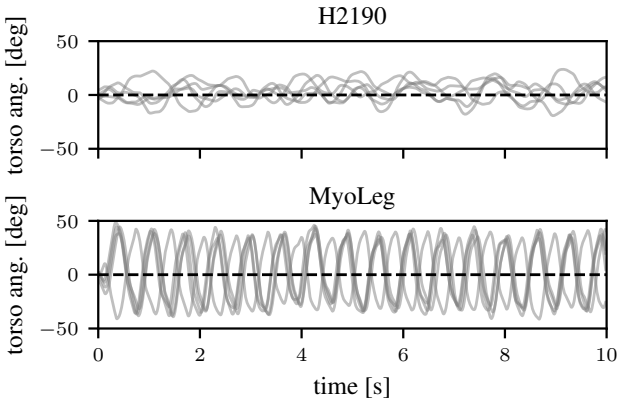
system	H0918	H1622	H2190	MyoLeg
max. velocity	5.38	5.04	6.49	5.44
achieved distance	n.a.	9.87 ± 4.27	10.45 ± 4.77	n.a.

C. Running

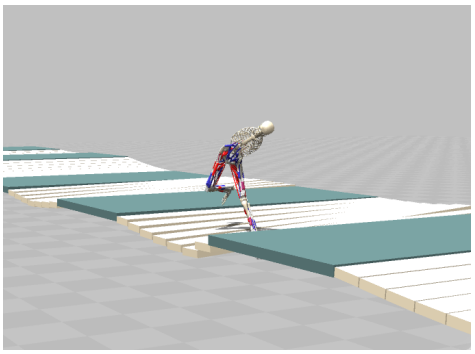
We performed maximum-speed running experiments with every model. While most reward terms remained identical to the natural walking case, we replace the external task reward by the velocity of the center of mass $r_{\text{vel}} = v$ and removed energetic constraints such as the muscle excitation clipping and the effort cost term. The gait-cycle- and leg-averaged kinematics are shown in Fig. C. As this task is a maximum performance movement, we have equalized the forces between the Hyfydy- and MuJoCo-based models, as the Hyfydy-models in the main experiments are generally based on experimental data with weaker maximum isometric muscle forces [28]. Note that we added a negative reward for self-collision forces for the running tasks, as the agents would often cross their legs and hit them against each other, thereby hopping instead of running.

Even though there remains a stronger discrepancy between the produced kinematics and the experimental data than for walking, the hip movement and GRFs are generally well aligned for the Hyfydy-models. The *MyoLeg*-model presents very strong lateral torso oscillations during running, see also Fig. 6a. In future work, biological objectives such as head stabilization or the inclusion of arms in the model might minimize some of these artifacts. See Tab. III for the maximum running velocities for each model.

We also performed robustness experiments on a challenging obstacle course, see Fig. 6b and supplementary videos.



(a) **Torso oscillations during flat-ground running.** We show the torso angle with the vertical axis for 5 rollouts of 10 s for the *H2190* and the *MyoLeg* models. The *MyoLeg* presents strong lateral oscillations. The dashed line shows a straight torso posture.



(b) **Dynamic terrain for running.** We probe the robustness of our policies trained for the *H1622* and the *H2190* with challenging obstacles. The tiles of the bridge rotate around the central axis and hang downwards, similar to a drawbridge. The agents were trained on **flat** ground and only have access to proprioceptive feedback.

Fig. 6: Torso oscillations for running and dynamic perturbation environment.

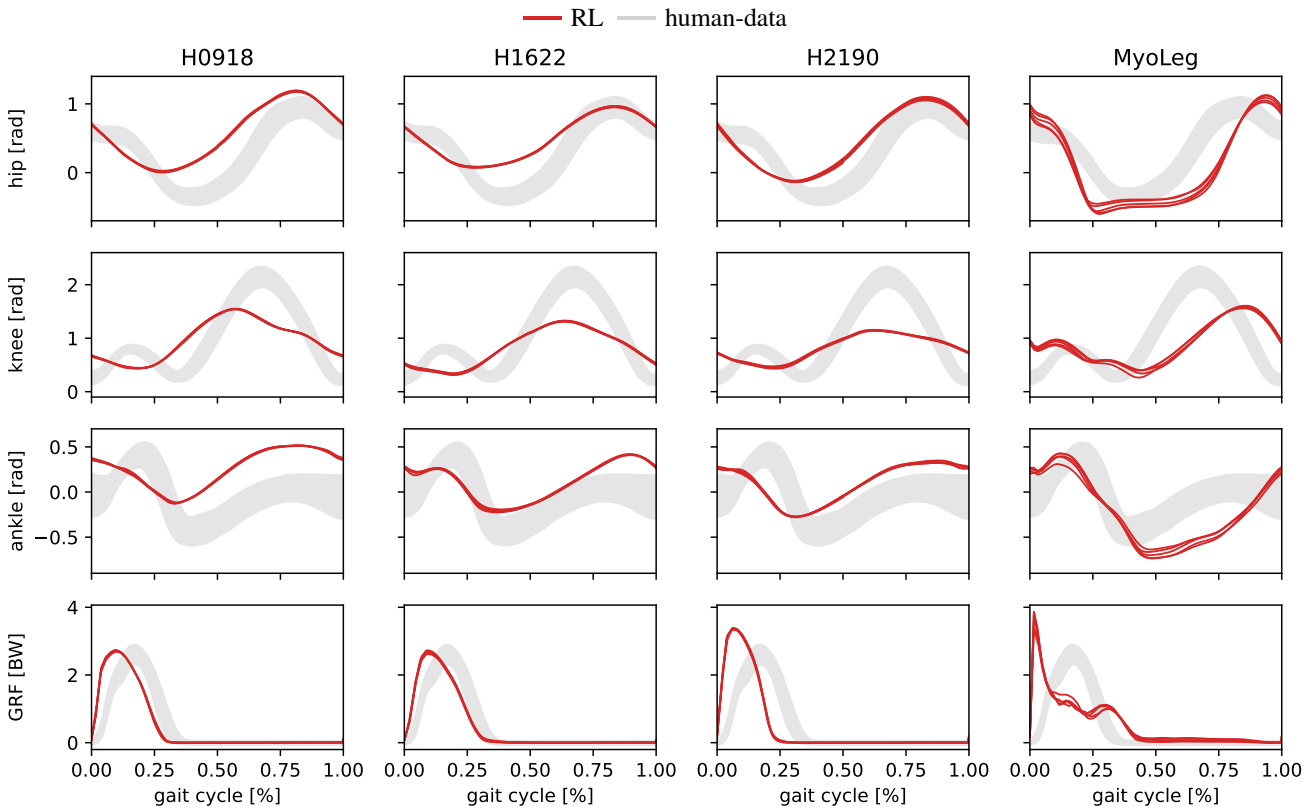


Fig. 7: **Gait-cycle kinematics for running.** The experimental data shows human subjects running at 5 m/s and was extracted from [52].

D. Reward function ablation

We perform ablations on our reward, see Fig. 8. Throughout all considered variations, only the full reward functions leads to gaits resembling human kinematics with low muscle activity across all models.

E. Hyperparameters

Used hyperparameter settings for the RL agent, DEP and the cost function are shown in Tab. IV. Non-reported RL values are left to their default setting in TonicRL [53]. See [7] for an explanation of the DEP-specific terms. The RL parameters were held constant, except for an increase in network capacity for *H2190* and *MyoLeg*.

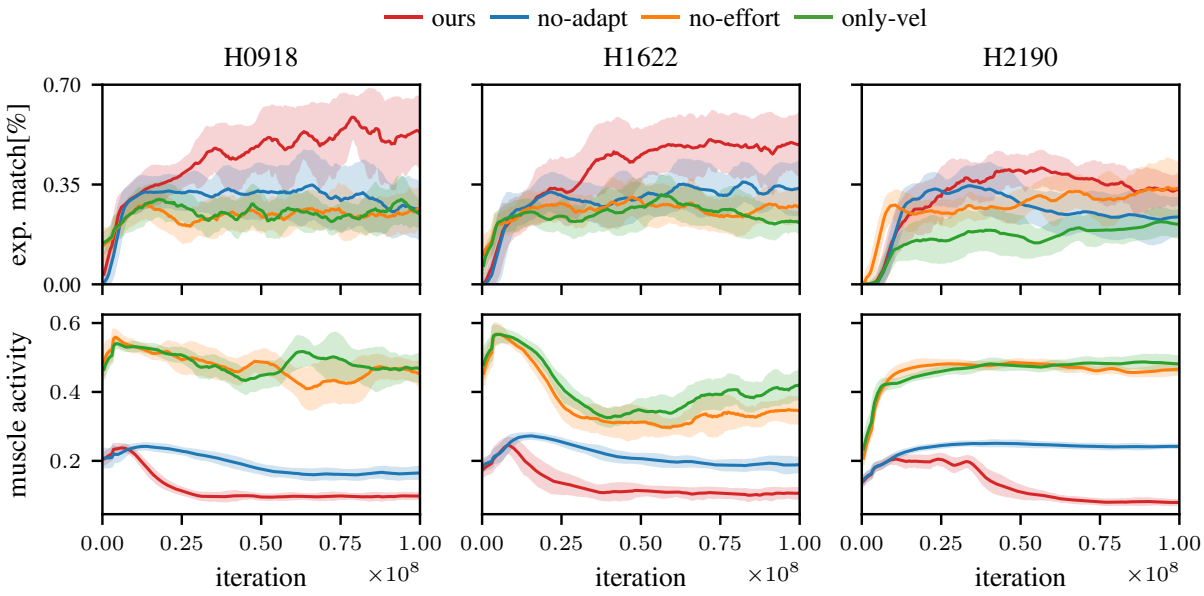


Fig. 8: **Cost function ablations.** We show several ablations of our cost function and plot the average match with experimental human data, as detailed in the main paper, as well as the average muscle activity. A natural gait is generally characterized by a large experimental match as well as minimal muscle activity. Different ablations are shown: The adaptive effort term is zero ($\alpha(t) = 0$): no-adapt. The entire effort cost term is zero ($c_{\text{effort}} = 0$) and we deactivate the action clipping: no-effort. We only reward with the velocity reward term ($c_{\text{effort}} = 0$ & $c_{\text{pain}} = 0$): only-vel. Only the combined cost function achieves a close resemblance to natural gait with low muscle activity. Leaving out the pain-related costs leads to the worst gait trajectories, while a combination of the effort cost terms and the adaptive cost term is needed to achieve the lowest muscle activity.

TABLE IV: Hyperparameters for all algorithms.

(a) DEP settings.

	Parameter	Value
DEP	κ	1200
	τ	40
	buffer size	200
	bias rate	0.002
	s4avg	2
	time dist (Δt)	5
integration	p_{switch}	3.71×10^{-4}
	H_{DEP}	8
	test episode	3
	force scale	n.a.

(b) MPO settings

Parameter	Value
buffer size	1e6
batch size	256
steps before batches	2e5
steps between batches	1000
number of batches	30
n-step return	1
n parallel	20
n sequential	10
hidden layers	2
hidden sizes	256
lr_{actor}	3×10^{-4}
$\text{lr}_{\text{critic}}$	3×10^{-4}
lr_{dual}	1×10^{-2}

(c) MPO setting changes for *H2190* and *MyoLeg*.

Parameter	Value
hidden sizes	1024
lr_{actor}	3.53×10^{-5}
$\text{lr}_{\text{critic}}$	6.08×10^{-5}
lr_{dual}	2.13×10^{-3}

(d) Cost function settings.

Parameter	Value	Meaning
ω_1	0.097	action smoothing
ω_2	1.579	number of active muscles above 15% activity
ω_3	0.131	joint limit torque
ω_4	0.073	GRFs above 1.2 BW
$\Delta\alpha$	9×10^{-4}	change in adaptation rate
θ	1000	performance threshold
β	0.8	running avg. smoothing
λ	0.9	decay term

Appendix C

Learning with Muscles: Benefits for Data-Efficiency and Robustness in Anthropomorphic Tasks

Learning with Muscles: Benefits for Data-Efficiency and Robustness in Anthropomorphic Tasks

Isabell Wochner[*1] **Pierre Schumacher**[*2,3] **Georg Martius**[2]

Dieter Büchler[2] **Syn Schmitt**[†1] **Daniel F.B. Haeufle**[†1,3]

[1]Institute for Modelling and Simulation of Biomechanical Systems, University of Stuttgart, Germany
[2]Max Planck Institute for Intelligent Systems, Tübingen, Germany
[3]Hertie-Institute for Clinical Brain Research, University of Tübingen, Germany

Abstract: Humans are able to outperform robots in terms of robustness, versatility, and learning of new tasks in a wide variety of movements. We hypothesize that highly nonlinear muscle dynamics play a large role in providing inherent stability, which is favorable to learning. While recent advances have been made in applying modern learning techniques to muscle-actuated systems both in simulation as well as in robotics, so far, no detailed analysis has been performed to show the benefits of muscles when learning from scratch. Our study closes this gap and showcases the potential of muscle actuators for core robotics challenges in terms of data-efficiency, hyperparameter sensitivity, and robustness [2].

Keywords: reinforcement learning, model predictive control, actuator morphology

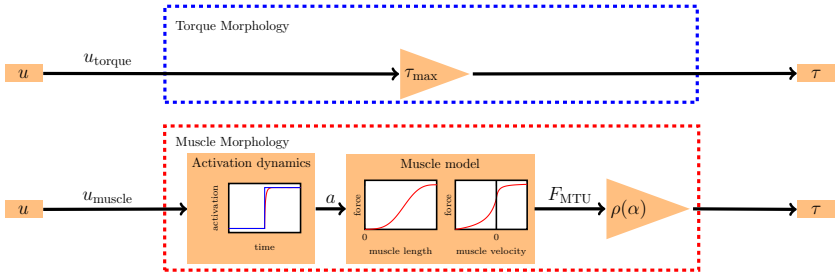


Figure 1: Key differences between torque actuator morphology and muscle actuator morphology.

1 Introduction

Recent developments in new learning methods allow the generation of complex anthropomorphic motions such as grasping, jumping or hopping in robotics. However, current systems still struggle with real-world scenarios beyond the narrow conditions of laboratory experiments. Humans, on the other hand, are capable of quickly adapting to uncertain, complex, and changing environments in a sheer endless variety of tasks. One key difference between biological and robotic systems lies in their actuator morphology: robotic drives are mostly designed to yield a linear relation between control signal and output torque. In contrast, muscles have complex nonlinear characteristics.

It has already been demonstrated, that muscular nonlinearities may provide a benefit for stability and robustness, especially under environmental uncertainties or perturbations [1, 2, 3]. A benefit over linear torque actuator morphology has been observed in computer simulations by exchanging the actuator morphology (similar to Fig. 1) in otherwise identical anthropomorphic tasks like reaching [4]

[*]Equal contribution. † Equal contribution.
[2]See https://sites.google.com/view/learning-with-muscles for code and videos.

6th Conference on Robot Learning (CoRL 2022), Auckland, New Zealand.

or locomotion [5, 6, 7, 8]. Similarly, reduced demand on the information processing capacity has been shown for muscles when compared to torque actuator morphology [9, 10, 11, 12, 13]. This opens the question whether muscular morphology could also be beneficial for robustness and data-efficiency in the process of *learning* movement control.

Recent advances in deep learning facilitated the generation of complex movements like point-reaching [14, 15, 16, 17] and locomotion [18, 19, 20, 21, 17] in simulations with muscular actuator morphology. In the real world, optimization and learning approaches can also find controllers for robotic systems with pneumatic muscles exhibiting somewhat muscle-like actuator morphology [22, 23]. These examples demonstrate that learning and optimization methods *can* control muscle-driven systems and may enable benefits such as safe learning and robustness [23]. However, investigating advantages of nonlinear muscular actuator morphology over linear torque actuator morphology requires a direct comparison of both, which is—to our knowledge—missing in the literature.

While Peng et al. [24] performed a comparative analysis of different actuator morphologies, their work was focused on replicating reference trajectories. In contrast, we learn behaviors without demonstrations, provide extensive hyperparameter ablations and not only employ RL, but also other optimization methods applied to complex 3D models.

The purpose of this study is to test if learning with muscular actuator morphology is more data-efficient and results in more robust performance as compared to torque actuator morphology when learning from scratch. We investigate this in a very broad approach: we employ different learning strategies on multiple anthropomorphic models for multiple variants of reaching and locomotion tasks solved in physics simulators of differing levels of detail. This provides new and comprehensive evidence of the beneficial contribution of muscular morphology to the learning of diverse movements.

2 Morphological difference between torque and muscle actuators

In contrast to idealized torque actuators, where torque is simply proportional to the control signal $u_{\text{torque}} \in [-1, 1]$,

$$\tau = \tau_{\text{max}}\, u_{\text{torque}} \tag{1}$$

muscular force output nonlinearly depends on the muscle control signal u_{muscle}, the muscle length l_{MTU} and contraction velocity $\dot{l}_{\text{MTU}}$. These biologically observed dependencies can be predicted by so-called *Hill-type* muscle models [25]. In a nutshell, the model captures biochemical processes transforming muscle stimulation $u_{\text{muscle}} \in [0, 1]$ to the force-generating calcium ion activity a. This can be modeled by a first-order differential equation of the form [26]

$$\dot{a} = f_a(u_{\text{muscle}} - a) \tag{2}$$

which induces low-pass filter characteristics (Fig. 1). The model further captures the nonlinear *force-length* and *force-velocity relations* [25]. The *force-length relation* is characterized by a positive slope (increasing force with increasing muscle fiber length) in the typical operating range of biological muscle fibres (Fig. 1). The *force-velocity* relation is characterized by decreasing force for faster shortening velocities and increasing force if the muscle is externally stretched against its contraction direction (Fig. 1). A lever arm $\rho(\alpha)$ translates joint angle α into muscle-tendon-unit length l_{MTU} and muscle force into joint torque

$$\tau = \sum_{i=1}^{N} \rho_i(\alpha) f_\tau \left(l_{\text{MTU},i}(\alpha), \dot{l}_{\text{MTU},i}(\dot{\alpha}), a_i \right). \tag{3}$$

for N muscles which span a joint—typically at least two in an antagonistic arrangement.

In practice, we employ two different muscle models: A detailed one with more physiological details, contained in demoa [27], and a simpler model that efficiently adds muscular properties to existing MuJoCo [28] simulations without sacrificing computational speed. See Suppl. A for details.

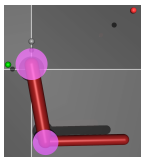
3 Methods

3.1 Learning approaches for movement control

We test if muscle actuator morphology facilitates learning by applying state-of-the-art learning algorithms covering an extensive range of approaches currently used in robotics. The common thread of the selected algorithms lies in their independence of the actuator morphology: this allows us to easily exchange idealized torque actuator morphology with muscle actuator morphology. We choose optimal control, model-predictive control and reinforcement learning as learning approaches.

Table 1: **Overview of all models and tasks**

Model	Task	Control	Environment
ArmMuJoCo	precise reaching	RL	MuJoCo
ArmMuJoCo	fast reaching	RL	MuJoCo
ArmDemoa	smooth point-reaching	opt. control, MPC	demoa
ArmDemoa	hitting ball with high-velocity	opt. control, MPC	demoa
Biped	hopping	RL	MuJoCo
FullBody	squatting	opt. control, MPC	demoa
FullBody	high-jumping	opt. control, MPC	demoa



(a) ArmMuJoCo

(b) ArmDemoa

(c) Biped

(d) FullBody

Figure 2: Models used for the experiments.

Optimal control (OC) The control problem with horizon N can be defined as:

$$\min_{\pi_k} J = \min_{\pi_k} \sum_{k=0}^{N} l(x(k), u(k), k), \qquad \text{subject to } x(k+1) = f(x(k), u(k), k),$$
$$u(k) = \pi_k(x(0), ..., x(k)). \tag{4}$$

where $x(k) \in \mathbb{R}^{n_x}$ denotes the current state at time k, and $u(k) \in \mathbb{R}^{n_u}$ is the applied input at time k. Furthermore, l specifies the cost function, and f denotes the system dynamics. To optimize for the best control policy, we use the covariance matrix adaptation evolution strategy (CMA-ES) [29] in the optimal control case (open-loop strategy). CMA-ES is a derivative-free algorithm and widely used in machine learning. It combines different learning mechanisms for adapting the parameters of a multivariate normal distribution. Note, that we choose the same optimization parameters for both actuator morphologies to allow for a fair comparison even though the number of decision variables n_u is always larger in the muscle-actuated case due to the antagonistic setup.

Model predictive control (MPC) In MPC, we solve the control problem in a receding-horizon fashion with varying prediction horizons and recursively apply only the first element of the predicted optimal control sequence $u(0)$ (closed-loop strategy). We employ a warm start procedure using the CMA-ES optimizer and afterwards start the MPC routine with the local optimizer BOBYQA [30]. The parameters of the optimizers are either varied (see Sec. 4) or given in Suppl. B.

Reinforcement learning (RL) RL allows learning of reusable feedback controllers. Instead of *minimizing* a cost function (see Eq. 4), conventionally the discounted expected reward is *maximized*:

$$\max_{\pi} J = \max_{\pi} \mathbb{E}\left[\sum_{k=0}^{N-1} \gamma^{k-1} r(k)\right] \tag{5}$$

where $r(k)$ is the reward at time k, π is a control policy and $\gamma \in [0, 1]$ is a discount factor such that long-term rewards are weighted less strongly. RL consequently solves a similar problem to MPC, but the resulting controllers are able to act in closed-loop fashion without being given an explicit prediction model. For the point-reaching tasks, we additionally employ goals g characterizing the desired hand position, which then constitutes an additional dependence of the reward function. The aim of the learning process is to learn a controller policy $\pi(u(k)|x(k))$ that takes as input the current sensor values, or state x(k), and outputs a control signal, or action, $u(k)$ such that a task is solved. In practice, we use the RL algorithm MPO [31], implemented in TonicRL [32].

3.2 Models

Arm The Arm model (Fig. 2 a, b) consists of two segments connected with hinge joints (2 joints total) moving against gravity. The ArmMuJoCo [28, 17] model contains four muscles, two for each

joint. In the muscle-actuated case in ArmDemoa [33, 34], six Hill-Type muscles generate forces: two muscles for the shoulder and two for the elbow joint, plus two biarticular muscles acting on both joints. All joints are individually controllable.

Biped We converted the geometrical model of an OpenSim bipedal human without arms [19] for use in MuJoCo. The model, consisting of 7 controllable joints (lower back, hips, knees, ankles) moves in a 2D plane. Each joint is actuated by two antagonistic muscles or one torque actuator.

FullBody The FullBody model [35, 36] consists of two legs and an upper body including arms based on a human skeletal geometry. It consists of 8 controllable joints (ankles, knees, hips, lumbar and cervical spine) in 3D, and 14 movable joints in total including the arms. Each controllable joint was either actuated by two antagonistic muscles (muscle-actuated case) or by one idealized torque actuator (torque-actuated case). For more details, we refer to Suppl. C.

All models and their respective physics differential equations were solved with variable time step (max. time step 0.001s) in demoa and fixed time step (0.005s) in MuJoCo.

3.3 Objectives and rewards

We choose anthropomorphic movement objectives which are highly relevant for robotic applications. We expect that muscular actuator morphology provides benefits for such tasks. All task formulations allow application in muscle and torque actuator morphologies with an identical reward or objective function. For a precise formulation of the used functions and conditions, see Suppl. D.

Smooth point-reaching This task encourages *smooth* point-reaching. Therefore, the objective minimizes the L2-error between the desired and current joint angle while penalizing the angle velocity and jerk to ensure a smooth motion. The desired angle is $90°$ for both the shoulder and the elbow joint, as this requires a large motion.

Precise point-reaching Similar to [13], we incentivize reaching a random hand position in a predetermined rectangle, while minimizing the distance of the end effector to the goal. We specifically add a reward term that gives a much larger reward for precise motions that reach the center of the target area. The episode does not terminate until a time limit of 1000 steps elapses.

Fast point-reaching The same objective as for precise point-reaching is used, but the episode terminates if the target is reached, incentivizing reaching speed over precision.

High-velocity ball serve A ball is dropped in front of the Arm model and the controller learns to hit the ball to achieve maximum ball velocity.

Squatting This squatting objective is taken from [35] and encourages desired hip, knee, and ankle angles for a squatting position.

Maximum height jump The objective for the high-jumping task is taken from [37] and maximizes the position and velocity of the centre of mass of the human body model at the time of lift-off. The model is initialized to start from a squatting position.

Hopping We developed an objective based on the z-axis velocity of the center of mass (COM) of the system that encourages periodic hopping with maximum height. The episode terminates if extreme joint angles are exceeded.

4 Results

In the following, we present three major results for the investigated approaches and environments: (**1**) Muscle-like actuators in general improve data-efficiency compared to torque-actuators. (**2**) The investigated learning and optimization algorithms exhibit greater robustness to hyperparameter variations when applied to muscle-driven systems. (**3**) The motions and controllers obtained from the muscular morphology are more robust against force perturbations that were not present during learning. We average results over 5 and 8 random seeds for OC/MPC and RL respectively.

4.1 Data efficiency: Learning with limited resources

Robotics applications in real-world scenarios often suffer from limited resources, which holds true for training and inference time. Therefore, we investigate the advantages of muscle-like actuator morphology in terms of overall learning efficiency and temporal control resolution.

Advantages of muscular morphology Smooth and precise point-reaching generally require more data with torque-driven systems, as seen in Fig. 3. The performance of the muscle actuator, in contrast to torque morphology, varies very little for different settings of the temporal control resolution c. Precise reaching with RL also results in stable performance with fewer training iterations, and a very

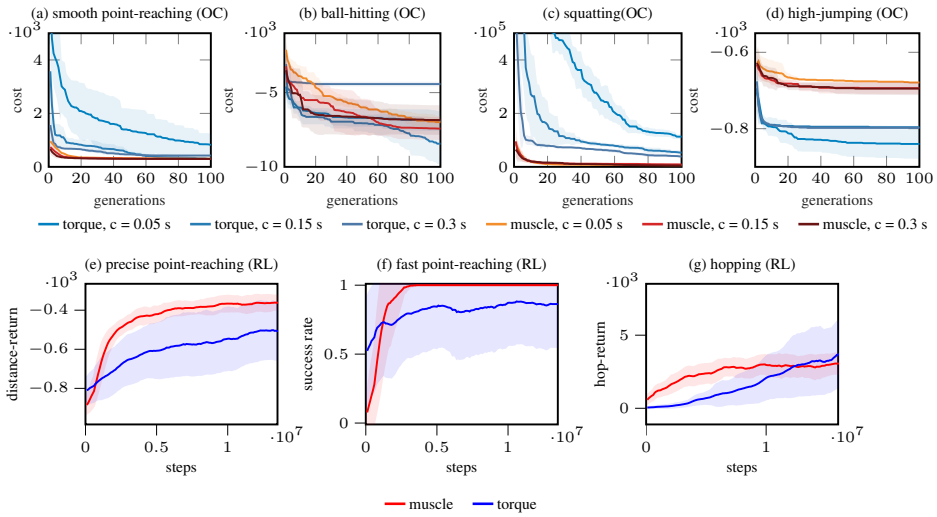


Figure 3: **Cost value or returns for different tasks.** Plotting the mean and standard deviation (shaded area) for 5 (OC/MPC) or 8 (RL) repeated runs for the two actuator morphologies (muscle in red, torque in blue). Additionally, the temporal control resolution c was varied in the OC cases.

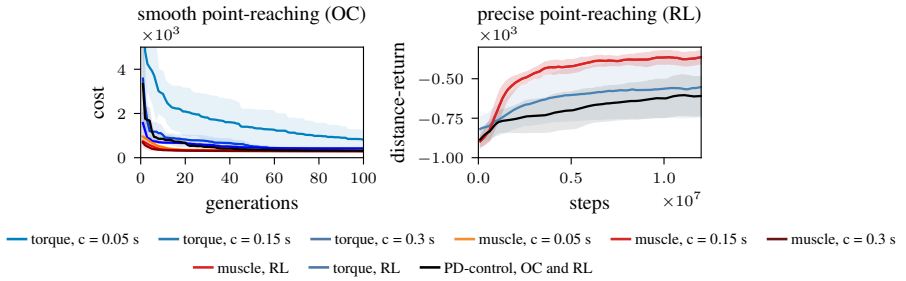


Figure 4: **Cost value or return in comparison with PD-control for torque.** Left: Muscles outperform all other considered morphologies with OC, while PD-control achieves lower cost than torque actuation with large control resolution $c = 0.3$. Right: PD-control does not yield an advantage over torque actuators with RL when applied to the precise point-reaching task.

small standard deviation across runs. Similar findings are seen for the squatting and hopping task, where muscle-actuators achieve better data-efficiency and smaller variation across runs and are able to find a good-enough optimum with fewer iterations.

Advantages of torque morphology In tasks requiring fast and strong motions, without emphasis on stabilization, we find torque actuators to hold certain advantages. In ball hitting and fast reaching, the torque cases show similar or smaller variance, even though both actuators perform well for singular runs. The high-jumping task, where only a strong, swift motion is required to launch the system upwards, is solved much faster in the torque case. We can also observe in the hopping task that, although only after a considerable number of training iterations and exhibiting a large variance, some torque-actuated runs achieve a larger overall return than the best muscle-actuated runs.

We additionally investigated a PD controller for the torque actuator morphology, see Fig. 4. While the PD controller slightly improves the data-efficiency for some cases, for both OC as well as for RL, the muscle actuator outperforms all baselines. See Sec. F.2 for more experiments.

4.2 Robustness to hyperparameter variations

Tuning a growing number of hyperparameters for learning models typically requires considerable time and computational resources. By analysing hyperparameter sensitivity, we test if tuning with torque or muscle actuator morphologies requires less resources.

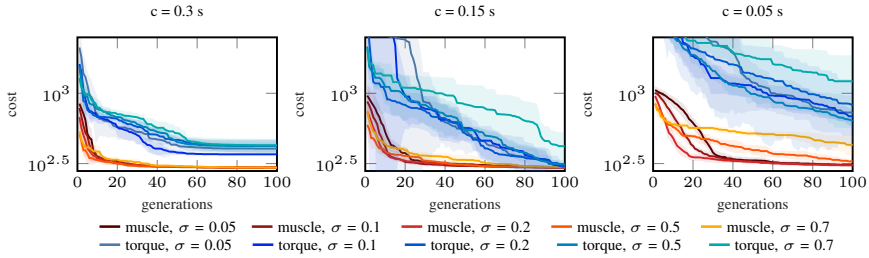


Figure 5: **Muscle morphology is more robust towards hyperparameter variation (σ) in point reaching.** The cost value of the best observation is shown. The mean and standard deviation (shaded area) are plotted for five repeated runs for the two actuator morphologies (muscle in red, torque in blue) with different control resolutions c.

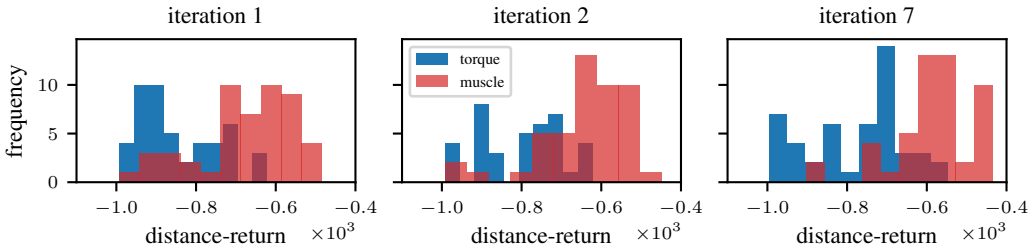


Figure 6: **Muscle actuators need less parameter tuning for good performance.** Hyperparameters are optimized for precise point-reaching following an iterative sampling scheme, each run trains for 2×10^6 iterations. Fifty sets of parameters are sampled randomly from pre-determined distributions, the final performance is evaluated and used to adapt the sampling distributions for the next iteration. We plot the return distributions over the sampled parameters at different iterations.

Figure 5 shows the cost curves for smooth point-reaching for the evolutionary optimization algorithm CMA-ES for different values of σ, which is the principal tuneable parameter for this algorithm. The performance curves vary much more for torque actuators for all considered cases. Furthermore, all muscle-actuated cases find a good-enough optimum with fewer iterations and a smaller variance, independent of the hyperparameter σ and the control resolution c.

The same task was repeated using MPC while varying the main hyperparameter t_{pred}, which represents the prediction horizon in moving horizon strategies. The performance curves and the final cost vary much more for torque actuators (Fig. 7a, note, the cost is plotted logarithmically).

Finally, we performed an extensive hyperparameter optimization for precise point-reaching. For each iteration, 50 sets of parameters are randomly chosen and the final task performance is evaluated after 2×10^6 learning iterations. The sampling distributions for the parameters are then fit to the best performing runs and 50 additional sets are evaluated for the next iteration. We optimize the learning rates of MPO, as well as gradient-clipping thresholds, as these have a strong influence on learning speed and stability. Muscle actuators already outperform torque-actuators in the first iteration, with a greater number of well performing parameter sets (Fig. 6). Almost no low-performing runs remain for iteration 7, while a large torque-performance is only achieved by a small subset of parameter settings. See Suppl. E for more hyperparameter variations.

4.3 Robustness to perturbations

In this section, we probe the robustness of the obtained policies against unknown perturbations. In precise point-reaching, we evaluated the RL reaching policies for two modifications that were not present during training: First, the hand-weight of the model is increased by 1.5 kg (dynamic load), and secondly a free spherical weight is attached to the end effector with a cable (chaotic load). We can see in Fig. 8 that the muscle-based policy does not suffer significant changes in performance, except for a small circular motion (3 cm) around the goal position in the chaotic load case. In contrast, the torque actuator morphology leads to unstable reaching and strong oscillations. Both morphologies seem to handle the dynamic load well. See Suppl. E for more goals.

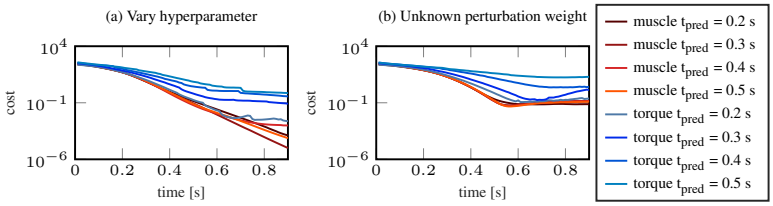


Figure 7: **Muscle morphology is more robust towards variation of hyperparameter t_{pred} and unknown perturbations.** Plotting the development of cost over time for the two actuator morphologies (muscle in red, torque in blue) while varying the hyperparameter t_{pred} denoting the length of the prediction horizon. Left: The unperturbed case. Right: The prediction model is not aware of the added weight to the lower arm.

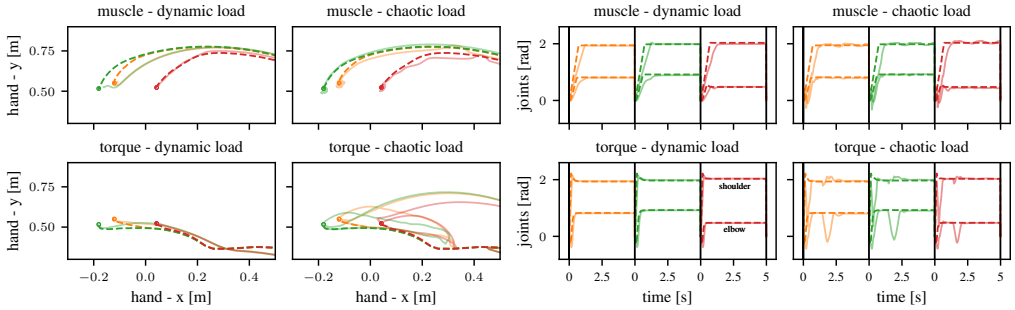


Figure 8: **Trajectories for dynamic (1.5 kg weight) and chaotic (attached ball) load.** Left: Three random goals are exemplarily shown, the respective goal position is marked as a circle. The unperturbed baseline for each goal is shown with a dashed line. Right: Joint trajectories for the same experiment, the unperturbed baselines are shown with dashed lines. Vertical bars mark episode resets.

For the MPC controller, we evaluated robustness by introducing environment changes that are unknown to the prediction model. One example is the lifting of an object with unknown weight, a typical robotics task. When adding 1 kg to the lower arm of the ArmDemoa model (Fig. 7), the performance in both actuator cases is worse than in the unperturbed case (left); the movement is also slower. However, the variance and absolute value of the final cost in the muscle-actuated case are still much lower compared to the torque-actuated case (plotted logarithmically). See Suppl. F.3 for more weight variations.

For periodic hopping with the Biped model, we evaluated trained RL policies with random forces that were drawn from a Gaussian distribution $F \sim \mathcal{N}(\cdot|0, \sigma_F)$ and applied to the hip, knee, and ankle joints with a probability of 0.05 at each time step. We see in Fig. 10 that the torque actuator morphology is stronger affected in relative performance than the muscle morphology. In the robustness investigation with MPC in the FullBody squatting task, a force is applied to the hip joint after the system has reached its desired position. Figure 9 (left) shows that the desired joint angles are much less affected by the perturbation when muscle actuators are controlled. Furthermore, the cost value associated with the movement recovers much slower for torque actuators (Fig. 9 right).

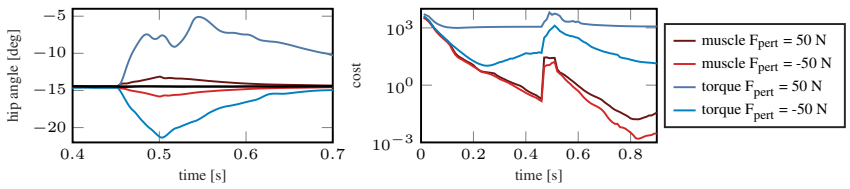


Figure 9: **Hip and cost trajectory for squatting with unknown perturbation forces.** The muscle morphology is shown in red, the torque morphology in blue while varying the perturbation force F_{pert} [N] (applied between $t >= 0.45$ s and $t <= 0.5$ s).

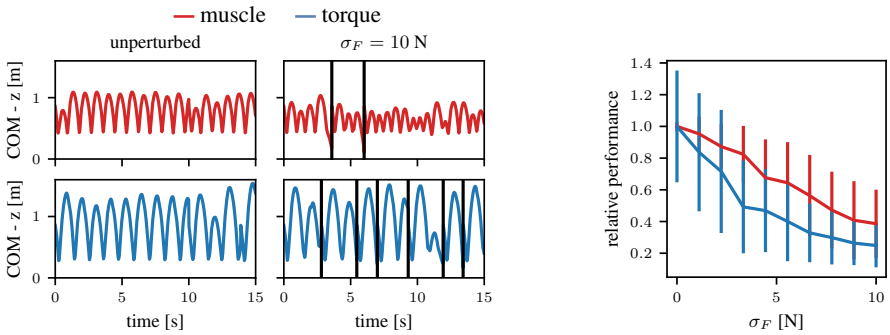


Figure 10: **Muscle actuators lead to more robust hopping.** At each time step, there is a 5 % chance of a random force being applied to the hip, knee, and ankle joints. Left: COM motion over time. Vertical bars mark episode resets due to falls of the Biped. Right: Relative performance for different standard deviations of the random force σ_F. Performance is scaled by the unperturbed mean return.

5 Discussion

We investigated if muscle-like actuators have beneficial effects for modern learning methods in terms of data-efficiency, hyperparameter sensitivity and robustness to perturbations. A multitude of variations across physics simulators, learning algorithms and task domains was considered in order to showcase the potential of the considered morphologies independently of the underlying implementation. We showed that muscles yield benefits in tasks requiring stable motion, even when compared to idealized torque actuators, which can be considered an upper performance bound. Indeed, the used torque actuators are able to instantaneously output any desired force at any point of the trajectory, while muscles only slowly change their output due to activation dynamics and can only produce kinematics-dependent force output. Despite these limitations, the considered learning algorithms learn more efficiently with muscle actuation in all tasks, except for extreme motions where objectives require a strong force application without stability considerations, such as ball-hitting and high-jumping. In bipedal hopping, it was found that muscles result in more efficient learning, even though some torque-runs achieve higher asymptotic performance. Finally, we observe muscle actuation to result in increased robustness to perturbations and hyperparameter variations, which can facilitate learning on real robotic systems that not only present sensor and motor noise, but also prohibit extensive parameter searches.

Outlook for real-world robotics We see two use-cases of our findings: (1) Muscular force-length-velocity and low-pass filter characteristics can be implemented as low-level actuator control for torque-controlled robotic systems (e.g., [38, 39, 40, 41]). This could allow us to exploit the improved data efficiency and robustness observed in our study for RL on a real robotic system. (2) Novel soft robotic actuators, such as artificial muscles [42, 43, 44, 45], promise to revolutionize specific application scenarios of robotics, e.g., wearable rehabilitation devices [46]. While soft actuated systems are hard to control from a classical control theory point of view, our results and other works [24] suggest that RL may even benefit from their properties. In our study, the simplified MuJoCo muscle model is applicable as a low-level controller in the sense of the first use case, while the results with the complex series-elastic muscle model in Demoa highlights the second use-case, making both cases strong arguments to consider RL and muscle properties a promising combination.

Limitations Although we have reported results for a wide variety of algorithms and tasks, we cannot give theoretical statements about the general applicability of our findings. Additionally, some of the tasks we employed were limited in complexity and might also be solvable with classical control algorithms. The MuJoCo muscle model, while computationally efficient, only captures rudimentary properties of biological systems. The demoa implementation, on the other hand, includes visco-elastic, passive tendon characteristics and muscle routing as joint angle-dependent lever arms to account for many physiological details—at substantial additional computational cost. Finally, learning with intermediate control signals given to impedance or position controllers, instead of direct torque commands, might also improve learning performance, while muscle-like properties could have been introduced by learning priors or additional cost terms.

Acknowledgments

We thank Daniel Höglinger for help during the development of the hopping reward function. Furthermore, we like to thank Marc Toussaint, Danny Driess and David Holzmüller for initial discussions regarding the topic of learning with muscles. This work was supported by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) under Germanys Excellence Strategy - EXC 2075 - 390740016 (SimTech). We thank the International Max Planck Research School for Intelligent Systems (IMPRS-IS) for supporting all authors. Georg Martius is a member of the Machine Learning Cluster of Excellence, EXC number 2064/1 Project number 390727645. This work was supported by the Cyber Valley Research Fund (CyVy-RF-2020-11 to DH and GM). We acknowledge the support from the German Federal Ministry of Education and Research (BMBF) through the Tübingen AI Center (FKZ: 01IS18039B).

References

[1] H. Wagner and R. Blickhan. Stabilizing function of skeletal muscles: an analytical investigation. *Journal of Theoretical Biology*, 199(2):163–179, 1999.

[2] M. Eriten and H. Dankowicz. A rigorous dynamical-systems-based analysis of the self-stabilizing influence of muscles. *Journal of Biomechanical Engineering*, 131(1):011011, 2009.

[3] S. Brändle, S. Schmitt, and M. A. Müller. Mathematical Biology A systems-theoretic analysis of low-level human motor control : application to a single-joint arm model. *Journal of Mathematical Biology*, 11 2019. doi:10.1007/s00285-019-01455-z.

[4] K. Stollenmaier, W. Ilg, and D. F. B. Haeufle. Predicting Perturbed Human Arm Movements in a Neuro-Musculoskeletal Model to Investigate the Muscular Force Response. *Frontiers in Bioengineering and Biotechnology*, 8(308), 4 2020. doi:10.3389/fbioe.2020.00308.

[5] A. J. van Soest and M. F. Bobbert. The contribution of muscle properties in the control of explosive movements. *Biological Cybernetics*, 69(3):195–204, 1993.

[6] D. Haeufle, S. Grimmer, and A. Seyfarth. The role of intrinsic muscle properties for stable hopping - stability is achieved by the force-velocity relation. *Bioinspiration & Biomimetics*, 5 (1):016004, 2010. doi:10.1088/1748-3182/5/1/016004.

[7] K. G. Gerritsen, A. J. van den Bogert, M. Hulliger, and R. F. Zernicke. Intrinsic muscle properties facilitate locomotor controla computer simulation study. *Motor control*, 2(3):206–220, 1998.

[8] C. T. John, F. C. Anderson, J. S. Higginson, and S. L. Delp. Stabilisation of walking by intrinsic muscle properties revealed in a three-dimensional muscle-driven simulation. *Computer methods in biomechanics and biomedical engineering*, 16(4):451–462, 2013.

[9] R. J. Full and D. E. Koditschek. Templates and anchors: neuromechanical hypotheses of legged locomotion on land. *Journal of Experimental Biology*, 202(23):3325–3332, 1999.

[10] P. Holmes, R. J. Full, D. Koditschek, and J. Guckenheimer. The dynamics of legged locomotion: Models, analyses, and challenges. *SIAM review*, 48(2):207–304, 2006.

[11] R. Blickhan, A. Seyfarth, H. Geyer, S. Grimmer, H. Wagner, and M. Günther. Intelligence by mechanics. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 365(1850):199–220, 2007.

[12] D. Haeufle, M. Günther, G. Wunner, and S. Schmitt. Quantifying control effort of biological and technical movements: an information-entropy-based approach. *Physical Review E*, 89(1): 012716, 2014.

[13] D. F. Haeufle, I. Wochner, D. Holzmüller, D. Driess, M. Günther, and S. Schmitt. Muscles reduce neuronal information load: quantification of control effort in biological vs. robotic pointing and walking. *Frontiers in Robotics and AI*, 7:77, 2020.

[14] F. Fischer, M. Bachinski, M. Klar, A. Fleig, and J. Müller. Reinforcement learning control of a biomechanical model of the upper extremity. *Scientific Reports*, 11(1):14445, July 2021. doi:10.1038/s41598-021-93760-1.

[15] E. Joos, F. Péan, and O. Goksel. Reinforcement learning of musculoskeletal control from functional simulations. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 135–145. Springer, 2020.

[16] V. Caggiano, H. Wang, G. Durandau, M. Sartori, and V. Kumar. Myosuite – a contact-rich simulation suite for musculoskeletal motor control, 2022. URL https://arxiv.org/abs/2205.13600.

[17] P. Schumacher, D. Häufle, D. Büchler, S. Schmitt, and G. Martius. Dep-rl: Embodied exploration for reinforcement learning in overactuated and musculoskeletal systems, 2022. URL https://arxiv.org/abs/2206.00484.

[18] V. L. Barbera, F. Pardo, Y. Tassa, M. Daley, C. Richards, P. Kormushev, and J. Hutchinson. OstrichRL: A musculoskeletal ostrich simulation to study bio-mechanical locomotion. In *Deep RL Workshop NeurIPS 2021*, 2021. URL https://arxiv.org/abs/2112.06061.

[19] Ł. Kidziński, S. P. Mohanty, C. F. Ong, Z. Huang, S. Zhou, A. Pechenko, A. Stelmaszczyk, P. Jarosik, M. Pavlov, S. Kolesnikov, et al. Learning to Run challenge solutions: Adapting reinforcement learning methods for neuromusculoskeletal environments. In *The NIPS'17 Competition: Building Intelligent Systems*, 2018. URL http://arxiv.org/abs/1804.00361.

[20] Ł. Kidziński, C. Ong, S. P. Mohanty, J. Hicks, S. Carroll, B. Zhou, H. Zeng, F. Wang, R. Lian, H. Tian, W. Jaśkowski, G. Andersen, O. R. Lykkebø, N. E. Toklu, P. Shyam, R. K. Srivastava, S. Kolesnikov, O. Hrinchuk, A. Pechenko, M. Ljungström, Z. Wang, X. Hu, Z. Hu, M. Qiu, J. Huang, A. Shpilman, I. Sosin, O. Svidchenko, A. Malysheva, D. Kudenko, L. Rane, A. Bhatt, Z. Wang, P. Qi, Z. Yu, P. Peng, Q. Yuan, W. Li, Y. Tian, R. Yang, P. Ma, S. Khadka, S. Majumdar, Z. Dwiel, Y. Liu, E. Tumer, J. Watson, M. Salathé, S. Levine, and S. Delp. Artificial intelligence for prosthetics: Challenge solutions. In S. Escalera and R. Herbrich, editors, *The NeurIPS '18 Competition*, pages 69–128, Cham, 2020. Springer International Publishing. ISBN 978-3-030-29135-8.

[21] S. Song, Ł. Kidziński, X. B. Peng, C. Ong, J. Hicks, S. Levine, C. G. Atkeson, and S. L. Delp. Deep reinforcement learning for modeling human locomotion control in neuromechanical simulation. *Journal of NeuroEngineering and Rehabilitation*, Aug 2021.

[22] D. Driess, H. Zimmermann, S. Wolfen, D. Suissa, D. Haeufle, D. Hennes, M. Toussaint, and S. Schmitt. Learning to control redundant musculoskeletal systems with neural networks and sqp: exploiting muscle properties. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6461–6468. IEEE, 2018.

[23] D. Büchler, R. Calandra, and J. Peters. Learning to control highly accelerated ballistic movements on muscular robots. *Robotics and Autonomous Systems*, 2019.

[24] X. B. Peng and M. van de Panne. Learning locomotion skills using deeprl: Does the choice of action space matter? In *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, pages 1–13, 2017.

[25] T. Siebert and C. Rode. Computational modeling of muscle biomechanics. In *Computational Modelling of Biomechanics and Biotribology in the Musculoskeletal System*, chapter 6, pages 173–204. Woodhead Publishing, Elsevier, 1 edition, 2014.

[26] R. Rockenfeller, M. Günther, S. Schmitt, and . Götz, Thomas. Comparative sensitivity analysis of muscle activation dynamics. *Computational and Mathematical Methods in Medicine*, 2015: 1–16, 2015. doi:10.1155/2015/585409.

[27] S. Schmitt. demoa-base: A Biophysics Simulator for Muscle-driven Motion, 2022. URL https://doi.org/10.18419/darus-2550.

[28] E. Todorov, T. Erez, and Y. Tassa. MuJoCo: A physics engine for model-based control. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 5026–5033, Oct. 2012. doi:10.1109/IROS.2012.6386109.

[29] N. Hansen, S. D. Müller, and P. Koumoutsakos. Reducing the time complexity of the derandomized evolution strategy with covariance matrix adaptation (cma-es). *Evolutionary computation*, 11(1):1–18, 2003.

[30] M. J. Powell. The bobyqa algorithm for bound constrained optimization without derivatives. *Cambridge NA Report NA2009/06, University of Cambridge, Cambridge*, 26, 2009.

[31] A. Abdolmaleki, J. T. Springenberg, Y. Tassa, R. Munos, N. Heess, and M. Riedmiller. Maximum a posteriori policy optimisation. In *International Conference on Learning Representations*, 2018. URL https://arxiv.org/abs/1806.06920.

[32] F. Pardo. Tonic: A deep reinforcement learning library for fast prototyping and benchmarking. *arXiv preprint arXiv:2011.07537*, 2020.

[33] I. Wochner and S. Schmitt. arm26: A Human Arm Model, 2022. URL https://doi.org/10.18419/darus-2871.

[34] I. Wochner, D. Driess, H. Zimmermann, D. F. Haeufle, M. Toussaint, and S. Schmitt. Optimality principles in human point-to-manifold reaching accounting for muscle dynamics. *Frontiers in Computational Neuroscience*, 14:38, 2020.

[35] J. R. Walter, M. Günther, D. F. Haeufle, and S. Schmitt. A geometry-and muscle-based control architecture for synthesising biological movement. *Biological Cybernetics*, 115(1):7–37, 2021.

[36] J. R. Walter, I. Wochner, M. Jacob, K. Stollenmaier, P. Lerge, and S. Schmitt. allmin: A Reduced Human All-Body Model, 2022. URL https://doi.org/10.18419/darus-2982.

[37] M. G. Pandy, F. E. Zajac, E. Sim, and W. S. Levine. An optimal control model for maximum-height human jumping. *Journal of Biomechanics*, 23(12):1185–1198, 1990.

[38] F. Garcia-Cordova, A. Guerrero-Gonzalez, J. Pedreno-Molina, and J. Moran. Emulation of the animal muscular actuation system in an experimental platform. In *IEEE International Conference on Systems, Man and Cybernetics. e-Systems and e-Man for Cybernetics in Cyberspace*. IEEE, 2001. doi:10.1109/icsmc.2001.969789.

[39] A. Seyfarth, K. T. Kalveram, and H. Geyer. Simulating Muscle-Reflex Dynamics in a Simple Hopping Robot. In *Proceedings of Fachgespräche Autonome Mobile Systeme*, pages 294–300. Springer, 2007. doi:10.1007/978-3-540-74764-2_45. URL http://link.springer.com/10.1007/978-3-540-74764-2%5F45.

[40] J. Knüsel, A. Crespi, J.-M. Cabelguen, A. J. Ijspeert, and D. Ryczko. Reproducing five motor behaviors in a salamander robot with virtual muscles and a distributed cpg controller regulated by drive signals and proprioceptive feedback. *Frontiers in Neurorobotics*, 14, 2020. ISSN 1662-5218. doi:10.3389/fnbot.2020.604426. URL https://www.frontiersin.org/articles/10.3389/fnbot.2020.604426.

[41] A. Rai, R. Antonova, S. Song, W. Martin, H. Geyer, and C. Atkeson. Bayesian optimization using domain knowledge on the atrias biped. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1771–1778. IEEE, 2018.

[42] I. Boblan, R. Bannasch, A. Schulz, and H. Schwenk. A human-like robot torso zar5 with fluidic muscles: Toward a common platform for embodied ai. In *50 Years of Artificial Intelligence*, pages 347–357. Springer, 2007.

[43] G. K. Klute, J. M. Czerniecki, and B. Hannaford. Mckibben artificial muscles: pneumatic actuators with biomechanical intelligence. In *1999 IEEE/ASME International Conference on Advanced Intelligent Mechatronics (Cat. No. 99TH8399)*, pages 221–226. IEEE, 1999.

[44] B. Vanderborght, A. Albu-Schäffer, A. Bicchi, E. Burdet, D. G. Caldwell, R. Carloni, M. Catalano, O. Eiberger, W. Friedl, G. Ganesh, et al. Variable impedance actuators: A review. *Robotics and autonomous systems*, 61(12):1601–1614, 2013.

[45] S. Wolfen, J. Walter, M. Günther, D. F. Haeufle, and S. Schmitt. Bioinspired pneumatic muscle spring units mimicking the human motion apparatus: benefits for passive motion range and joint stiffness variation in antagonistic setups. In *2018 25th International Conference on Mechatronics and Machine Vision in Practice (M2VIP)*, pages 1–6. IEEE, 2018.

[46] M. Zhu, S. Biswas, S. I. Dinulescu, N. Kastor, E. W. Hawkes, and Y. Visell. Soft, wearable robotics and haptics: Technologies, trends, and emerging applications. *Proceedings of the IEEE*, 110(2):246–272, 2022. doi:10.1109/JPROC.2021.3140049.

Appendix D

Learning to Control Emulated Muscles in Real Robots: Towards Exploiting Bio-Inspired Actuator Morphology

Learning to Control Emulated Muscles in Real Robots: Towards Exploiting Bio-Inspired Actuator Morphology

Pierre Schumacher[1,2], Lorenz Krause[1,2], Jan Schneider[1], Dieter Büchler[1], Georg Martius[*1,3], Daniel Haeufle[*2,4]

***Abstract*— Recent studies have demonstrated the immense potential of exploiting muscle actuator morphology for natural and robust movement – in simulation. A validation on real robotic hardware is yet missing. In this study, we emulate muscle actuator properties on hardware in real-time, taking advantage of modern and affordable electric motors. We demonstrate that our setup can emulate a simplified muscle model on a real robot while being controlled by a learned policy. We improve upon an existing muscle model by deriving a damping rule that ensures that the model is not only performant and stable but also tuneable for the real hardware. Our policies are trained by reinforcement learning entirely in simulation, where we show that previously reported benefits of muscles extend to the case of quadruped locomotion and hopping: the learned policies are more robust and exhibit more regular gaits. Finally, we confirm that the learned policies can be executed on real hardware and show that sim-to-real transfer with real-time emulated muscles on a quadruped robot is possible. These results show that artificial muscles can be highly beneficial actuators for future generations of robust legged robots. Videos: https://sites.google.com/view/emulatedmuscles**

I. Introduction

The development of learning algorithms for robotic locomotion predominantly relies on position control with electric motors. The ideal actuator is considered to impose as few constraints as possible on the system, while additional properties such as regular foot patterns, minimal torque utilization, and robust policies arise due to the inclusion of a series of cost terms and certain training curricula. Nevertheless, there is rising interest in alternative actuation models, promising advantageous properties amenable to stable control and generalizing controllers without the need for tedious cost term tuning. Some of these studies investigate the effect of action spaces on learned controllers in general [1]–[3], while others demonstrate the benefits of a particular actuator class, which is well-known but under-utilized: muscles [4]–[6].

Muscles support the nervous system of animals and humans in robust control as they generate zero-time delay reactions (termed *pre*flexes) to environmental influences [7]–[9]. While some of these advantages have been investigated in simplistic simulated [1], [4], [10] and hardware demonstrators [11], [12], actual real-world experiments remain indispensable to validate the relevance to robotic hardware.

Different artificial muscle architectures are being developed to test the potential of the reported simulation findings, two notable examples being McKibben [13], [14] and HASEL muscles [15]. Although very promising, many years of research are needed to make these systems reliable and to allow reproducible experimentation in robotics. Some approaches aim to speed up the development of control algorithms for artificial muscles by using hybrid sim-hardware setups [16], [17] or model-based algorithms [14], which is not feasible for every task.

With the advent of powerful and affordable direct-drive electric motors it is possible to emulate particular actuators in real time. The concept of emulating muscles on motors has been tested in simulation, even for a bipedal robot [5], [6], but in hardware only with a singular motor on a lab bench [11], [18], [19]. This approach was limited so far, as a PD-controller was required to match the torques predicted by the muscle emulator due to time delays induced by the emulation [18], [19]. Benefits of emulated muscles have not been demonstrated in complex and realistic robotic settings.

Even in those instances where emulated muscles have been simulated, the control algorithms were limited to state machines or reflex-based controllers [5], [6]. The combination with more powerful frameworks such as reinforcement learning (RL), is doubly interesting, as it may allow the generation of diverse movements with emulated muscles, while the importance of the actuation model for RL performance is also well documented [1], [3], [20], including evidence that muscles can benefit learning [4].

The purpose of this study is to demonstrate that emulated muscle properties are beneficial for learning robust walking without the need for excessive reward engineering and can be exploited on real hardware. To achieve this, we first extend simulation results of a previously proposed muscle model [4]. We implement this model in the GPU-based simulator Isaac Gym [21] and show several benefits in quadruped locomotion and hopping tasks — when compared to commonly used actuators. After observing increased robustness and more regular gaits in the learned policies in simulation, we emulate the muscle-actuator in hardware with a high frequency real-time control cycle without an intermediate PD-controller. Our policies are learned with RL entirely in simulation without

[1]Max Planck Institute for Intelligent Systems, Tübingen, Germany.
[2]Hertie Institute for Clinical Brain Research, and Centre for Integrative Neuroscience, University of Tübingen, Germany.
[3]Distributed Intelligence, University of Tübingen, Germany
[4]Institute of Computer Engineering (ZITI), Heidelberg University, Germany
*GM and DH contributed equally to this publication
The authors thank the International Max Planck Research School for Intelligent Systems (IMPRS-IS) for supporting Pierre Schumacher. This work was supported by the Cyber Valley Research Fund (CyVy-RF-2020-11 to DH and GM)

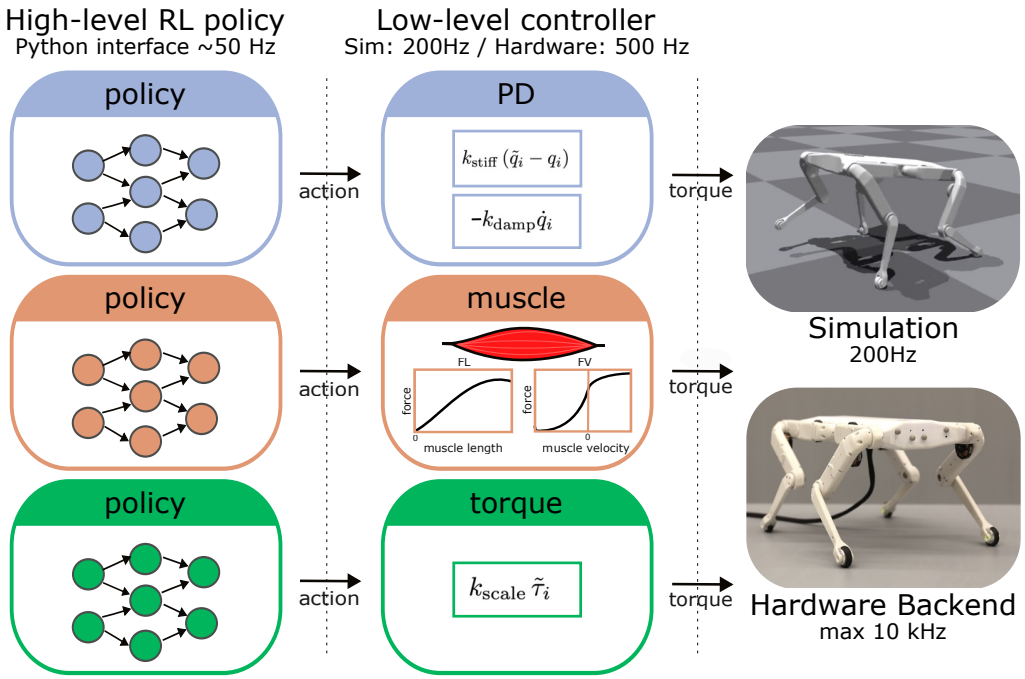


Fig. 1: **Emulated muscles on real robots.** We compare three different low-level actuation controllers which are used by three policies trained in simulation: PD, muscle and torque control. The resulting policies can then be tested on a real robot, by a real-time control module running at ~ 500 Hz, which communicates with the slowly executed RL policy executed with ~ 50 Hz. The robot backend performs torque commands with a maximum of 10 kHz and repeats previous torques between command updates. Importantly, the policy action either represents desired positions, muscle excitations or torques. The simulation has a physics timestep of 5 ms, with 4 physics updates for each controller step.

the need for numerous fine-tuned cost terms and are shown to transfer to the real system. This finding showcases the effectiveness of muscle actuators and hints at the potential for physical robots, either through emulation or novel hardware actuators.

Contributions: (**1**) We investigate different actuator models with RL on a simulated robotic quadruped. (**2**) We extend previous results on muscle actuation with RL from Wochner et al. [4] to a realistic robotic locomotion task. (**3**) We identify a critical parameter of the previously proposed muscle model and derive a theoretically motivated value to facilitate tuning the model for real systems. (**4**) We achieve sim-to-real transfer with a walking policy to a real-time emulated muscle-controller on robotic hardware.

II. Methods

A. Low-level actuation controllers

We compare the performance of RL policies trained with three different low-level controllers: PD-controller, muscle-controller, and direct torque-controller (Fig. 1).

PD-controllers are commonly used in quadruped locomotion studies [22], [23]; our version receives a desired position computed by the policy, while the desired velocity is set to zero, similar to [1]:

$$\tau_i = k_{\text{stiff}}\,(\tilde{q}_i - q_i) - k_{\text{damp}}\dot{q}_i, \tag{1}$$

where τ_i is the computed torque, $\tilde{q}_i$ the desired position, q_i current position, and $\dot{q}_i$ the velocity for joint i.

The **torque-controller** is considered to be ideal in the sense that the torque computed by the policy is directly applied to the joint:

$$\tau_i = k_{\text{scale}}\,\tilde{\tau}_i, \tag{2}$$

where $\tilde{\tau}_i \in [-1, 1]$ is computed by the policy and k_{scale} scales the policy output to the robot's torque range. While it is possible to use torque actuation on quadrupeds with learned policies, it imposes strong control frequency requirements on the implementation [24].

The **muscle-controller** is similar to [4] and emulates length and velocity dependent muscle force characteristics, as well as activation dynamics:

$$\tau_i = f_{\max}\left[\sum_{k=1}^{2}(-1)^{k+1}\text{FL}(l_k)\,\text{FV}(\bar{\dot{l}}_k)\,m_{\text{act},k} + \text{FP}(l_k)\right], \tag{3}$$

where $\text{FL}(\cdot)$ is the force-length and $\text{FV}(\cdot)$ the force-velocity relationship (see "muscle" in Fig. 1), $\text{FP}(\cdot)$ the passive force and m_{act} the muscle activity. The sum is taken over two muscles for each joint, each pulling in opposing directions. The activity m_{act} approaches the policy action $\tilde{a}$ with a low-pass filter:

$$\dot{m}_{\text{act}}(t) = \frac{1}{\Delta t_a}(\tilde{a}(t) - m_{\text{act}}(t)), \tag{4}$$

with the time constant $\Delta t_a = 0.01\,\mathrm{s}$. The scalar $f_{\max}$ was introduced to easily tune the maximum force output. The muscle length is given by:

$$l_k = a_k\, q_i + b_k, \tag{5}$$

with a_k and b_k being part of the parametrization. The muscle velocity is given by the derivative of Eq. 5

$$\bar{\dot{l}}_k = \beta\, \dot{l_k} = \beta\, (a_k\, \dot{q}_i). \tag{6}$$

Importantly, we multiply the muscle velocity by a scaling parameter β in the FV-function (Eq. 6), corresponding to $1/v_{\max}$ in [4], allowing us to tune the maximum damping strength of the muscle. The parameters a and b correspond to m and l_{ref} in [4], where the exact formulation is detailed.

B. Muscle model implementation

For the simulation experiments, we re-implemented the MuJoCo-based muscle model from [4]. In order to preserve Isaac Gym's computational speed [22], we translated the original Cython implementation of the muscle model to a vectorized PyTorch version. Overall, the vanilla PD-controller is only 30% faster than the muscle implementation, even though we simulate 16 muscles for the robot—compared to 8 motors for the PD and torque-controllers. Considering that the training time is mostly below an hour on a consumer-GPU, this computational efficiency is sufficient for the purpose of this study. The environment runs with a timestep of 5 ms for the physics engine and a control time step of 20 ms.

For the hardware experiments, the muscle model is implemented in C++, which communicates with real-time Python functions which were bound via PyBind11 [25]. This module interfaces with the RL policy running at ~ 50 Hz, which is not real-time executed, while the actual muscle-controller is running at ~ 500 Hz. It is important to achieve a large execution frequency on the hardware to reliably emulate the muscle model [4]. This paradigm is similar to most studies using PD-controllers on quadruped robots [24].

C. Muscle parameter tuning

In contrast to the PD-controller [26], there is no clear tuning procedure for a muscle-controller. The velocity scaling β is especially difficult to tune, as it affects the damping of the actuator through the FV-relationship [4] and might not transfer to real hardware due to control frequency limitations, see Sec. III-A. When we tried to set this parameter through an iterative tuning procedure similar to previous studies [27], we either obtained parameters that performed well in simulation but were not stable on the hardware, or they were stable but did not yield performant policies.

We derive an expression for the damping coefficient β, such that under certain conditions, the damping is equivalent to a damping-controller $\tau = -k_{\mathrm{damp}}\dot{q}$. With this approach, we can determine reasonable β values that are achievable on the hardware based on the easier-to-tune damping-controller.

We start by assuming two symmetric muscles connected to a single joint at position $q = 0$. As the FV-relationship is thought to primarily contribute to muscle damping [8], we also assume $\mathrm{FL}(l) = 1$, $\mathrm{FP}(l) = 0$ and constant activity $m_{act,k} = 1$. Equation 3 then simplifies to:

$$\tau = f_{\max}\, \left(\mathrm{FV}(\bar{\dot{l}}_1) - \mathrm{FV}(\bar{\dot{l}}_2)\right) \tag{7}$$

$$= f_{\max}\, \left(\mathrm{FV}(\bar{\dot{l}}_1) - \mathrm{FV}(-\bar{\dot{l}}_1)\right) \tag{8}$$

$$\approx f_{\max}\, (4\, \bar{\dot{l}}_1) \tag{9}$$

where $\bar{\dot{l}}_1 = -\bar{\dot{l}}_2$ in Eq. 6 as we assume symmetric muscles and $a_1 = -a_2$ in Eq. 5, and we have used a Taylor expansion around $\bar{\dot{l}}_1 = 0$. By using the definition of the FV-curve from [4], [28], we obtain

$$\mathrm{FV}(\bar{\dot{l}}) = 1 + 2\,\bar{\dot{l}} + \mathcal{O}(\bar{\dot{l}}^2).$$

By comparing Eq. 9 to a damping controller and using $\bar{\dot{l}}_1 = a_1 \beta \dot{q}$ we get:

$$-k_{\mathrm{damp}}\dot{q} = -f_{\max}\, 4\, \beta\, a_1\, \dot{q} \tag{10}$$

$$\Rightarrow \beta = \frac{k_{\mathrm{damp}}}{4\, a_1 f_{\max}}. \tag{11}$$

The negative sign in Eq. 10 was introduced as muscles are assumed to always pull on the joint. We found that setting $k_{\mathrm{damp}} = 0.1$ is an *achievable* value for a damping controller on the robot hardware. By using Eq. 11 for all muscle experiments, we ensure that the maximum co-contraction level generates *achievable* damping on the hardware. The policy can still reduce the amount of equivalent damping by changing the amount of co-contraction [8].

D. Tasks

We employ two tasks: walking and hopping, which have been proven interesting in previous comparisons of action spaces [4]. Walking requires a periodic and relatively slow joint movement, while periodic hopping is a highly explosive movement that also requires stabilization when landing.

1) Walking: We restrict ourselves to the target velocity tracking reward from [22] and omit all other terms that were used in that study:

$$r_{\mathrm{walk}} = \exp\left(-(v_{\mathrm{target}} - v_x)^2/\sigma\right), \tag{12}$$

where v_{target} is the target x-velocity, v_x is the x-velocity of the base and $\sigma = 0.25$ is a sensitivity factor. The task is reset if the robot base or the upper legs make contact with the ground or if the base height is $h_{\mathrm{base}} < 0.2\,\mathrm{m}$.

2) Periodic hopping: The hopping reward is based on the vertical velocity of the robot base v_z as

$$r_{\mathrm{hop}} = \exp\left(10\, v_{\mathrm{clip}}\right), \tag{13}$$

where $v_{\mathrm{clip}} = \mathrm{clip}(v_z, 0, 10)$. This is the same hopping reward as in [4] with slightly different scaling, accounting for a different velocity range with the SOLO robot. We observed the reward function to not work well if the desired hopping velocity is in a range where the reward function is not sensitive to changes. The episode is reset when the robot base changes in orientation too much from upright orientation, or when a body part different than the feet touches the ground.

Both tasks additionally use action rate regularization:

$$r_{\text{act}} = -w_{\text{act}} \left(a_{t+1} - a_t\right)^2, \tag{14}$$

where a_t is the action at time step t and w_{act} is a weighting coefficient.

The total reward is

$$r = r_{\text{walk/hop}} + r_{\text{act}}. \tag{15}$$

E. Learning high-level policies

In order to train policies, we use the popular RL algorithm PPO [29] with the implementation from [22] and identical hyperparameters as in [23] for the `SoloLeap`-task, shown in Table S6 in their work. Our policy and critic are fully-connected MLPs with 3 hidden dimensions of 128 units each and ELU-activation functions. For simulation experiments, we do not use domain randomization, while sim-to-real transfer requires randomization of initial states, ground friction, body mass, and input noise. See Tab. I for details.

TABLE I: Values for domain randomization (DR) and input noise (IN). All values are sampled uniformly in the given ranges and added to the existing values.

	Parameter	Range
DR	init. joint pos.	$[-1.0, 1.0]$
	init. muscle act.	$[0.5, 1.0]$
	friction	$[0.5, 1.25]$
	joint damping	$[0, 0.09]$
	push_{xy}	$[-1.5, 1.5]$
	mass shift	$[-0.5, 1.2]$
IN	base lin. vel.	$[-0.02, 0.02]$
	base ang. vel.	$[-0.05, 0.05]$
	gravity vector	$[-0.05, 0.05]$
	joint pos.	$[-0.01, 0.01]$
	joint vel.	$[-0.075, 0.075]$
	muscle length	$[-0.01, 0.01]$
	muscle vel.	$[-1.0, 1.0]$
	muscle act.	$[-0.01, 0.01]$
	muscle force	$[-1.0, 1.0]$

III. Results

Each simulation experiment was repeated over 10 random seeds. In experiments where we show a specific rollout of a specific policy, we analyzed all trained seeds for all policies at the end of their training, and selected in each case the one with the best behavior. In general we observe very small variance across seeds, likely due to the large number (4096) of parallel environments in Isaac Gym.

A. Parameter tuning procedure

As our study aims to investigate the potential benefits of muscle-like actuators, we use a methodical approach for parameter tuning. First, only the task-relevant reward is used as objective. We tune the actuator-relevant parameters to achieve maximum performance in a population-based optimization over 10 iterations of 100 trials each. This optimization successfully leads to policies achieving large task rewards. Upon closer inspection, however, we observe them to be very jittery, not likely to generalize to the real hardware.

TABLE II: Optimized parameter values for the used actuators. Note that β is not optimized but set through Eq. 11.

	Parameter	Value
muscle	l_{min}	0.24
	l_{max}	1.53
	fvmax	1.38
	fpmax	1.76
	lce_min	0.74
	lce_max	0.94
	f_{max}	34
	ϕ_{min}	-3.14
	ϕ_{max}	3.14
	τ_{act}	0.01
	β	0.36
PD	k_{stiff}	2
	k_{damp}	0.05
torque	n.a.	n.a.
action rate weight	w_{act}	0.004

We consequently add action-rate regularization, see Eq. 14. The best parameters from the previous step are taken, and a grid search over weighting coefficients for the action rate cost is performed. It is commonly observed that smooth actuation signals are more transferable to real-world robots [30]. The level of action smoothness that works well for sim-to-real transfer is hard to quantify, as such we visually check for the variance in the applied torques, instead of optimizing the regularizer together with the other parameters.

This optimization procedure is executed for the PD and muscle low-level controller for the walking task. The torque actuator has no parameters besides the maximum torque, which is given by the system specification of the robot. We then keep all parameters identical for the hopping task.

B. Walking

While performant and transferable gaits usually require dozens of cost terms which are hand-tuned by experts [22], [23], we investigate the behaviors that arise from a simple and under-specified forward velocity reward combined with an action rate cost. Different actuator morphologies might embed certain movement priors into the learning algorithm which do not require a complex tuning procedure.

The task reward over training improves faster for the PD- and torque-controllers than for the muscle-controllers, see Fig. 3. While all three policies perform well according to this metric, we also take a look at the learned gaits in Fig. 4.

The touch patterns of the feet with the ground (Fig. 4b) show that the muscle-agent learns the most regular gait. The PD-agent tends to not elevate the feet from the ground, as can be seen from the gait illustration (Fig. 4c) and the joint angle trajectories (Fig. 4a). This behavior relies on very precise movements and friction coefficients in the simulation, which would likely not generalize to the real world. Interestingly, the torque-agent lifts the feet off the ground but tends to use extreme and uncontrolled motions. This showcases that even with an under-specified reward function, a particular embodiment, or low-level controller, can bias the behavior towards being more natural.

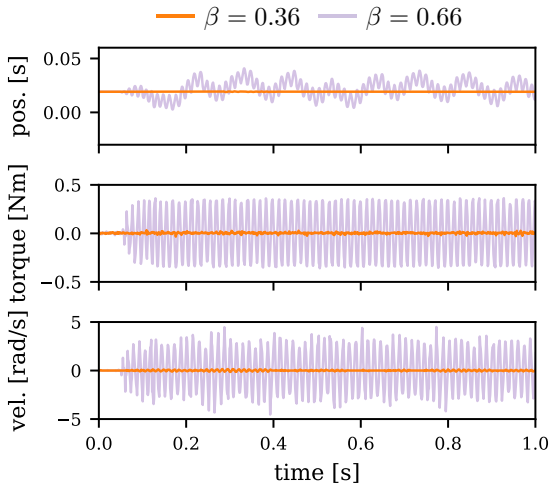


Fig. 2: **Hardware deployment: excessive emulated damping destabilizes the control.** We apply constant muscle excitations of $m_{\text{act}} = 1.0$ on both muscles of a joint of the hardware robot. Ideally, the joint would remain at its initial position, resisting external perturbations with high damping and stiffness. When using a high scaling value $\beta = 0.66$, the emulation controller can not keep up due to its limited control frequency, and the motor starts to oscillate strongly. The value $\beta = 0.36$, given by Eq. 11, leads to stable performance. We observe similar effects with a PD-controller and excessive k_{damp}-values.

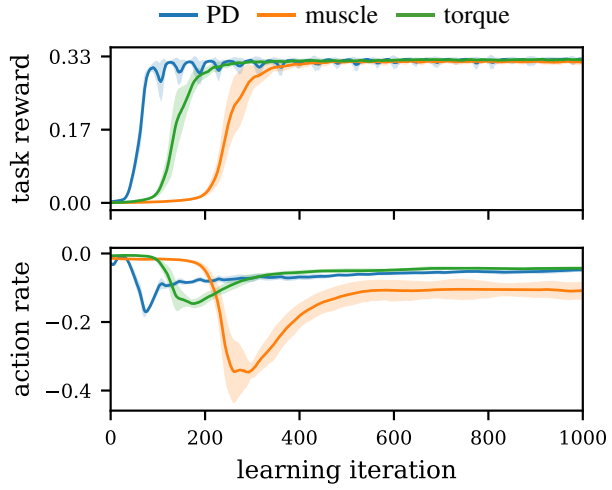


Fig. 3: **Stable locomotion is achieved by all actuators.** Upper: Velocity-tracking task reward over time. The PD-agent learns with the smallest number of iterations, while torque- and muscle-agents achieve a similar maximum performance with more training. Lower: The action rate reward only slowly improves for the muscle-agent, likely because the number of actions is twice as large as for the other actuators (16 vs 8).

C. Periodic hopping

The simulation results for the maximum height periodic hopping task are shown in Fig. 5. The reward function, see Eq. 13, incentivizes maximal upwards velocities for the longest possible time. Even though the PD and torque agents achieve some very high jumps (Fig. 5c), the hopping behavior is straighter and more regular with the muscle actuation, making it easier to control when landing. This leads to large episode lengths (Fig. 5b) for the muscle-agent, as the other controllers can often not recover after landing. These results are slightly different to [4], where the muscle-agent achieved better performance than the alternatives only early in the training, but this difference may be explained by the superior stability of quadrupeds over bipedal robots.

D. Robustness

Wochner et al. [4] demonstrated that muscle-actuators exhibit higher robustness under unseen perturbations. We therefore tested the trained policies under terrain variations used in [22] and recorded the fraction of steps across 100 episodes in which the policies would not fall down, which we define as the success rate. Note that the policies have only been trained on flat terrain. We see that the muscle-controller is generally more robust and has higher success rates than the other controllers (Fig. 6).

E. Sim-to-real transfer

In order to test the applicability of our findings and highlight their relevance for future robot design, we implemented our muscle model on real robotic hardware. First, we demonstrate the sim-to-real transfer of our policies in a task requiring tracking joint angles over time. For this experiment, the robot is suspended in mid-air on a fixed stand (Fig. 7b).

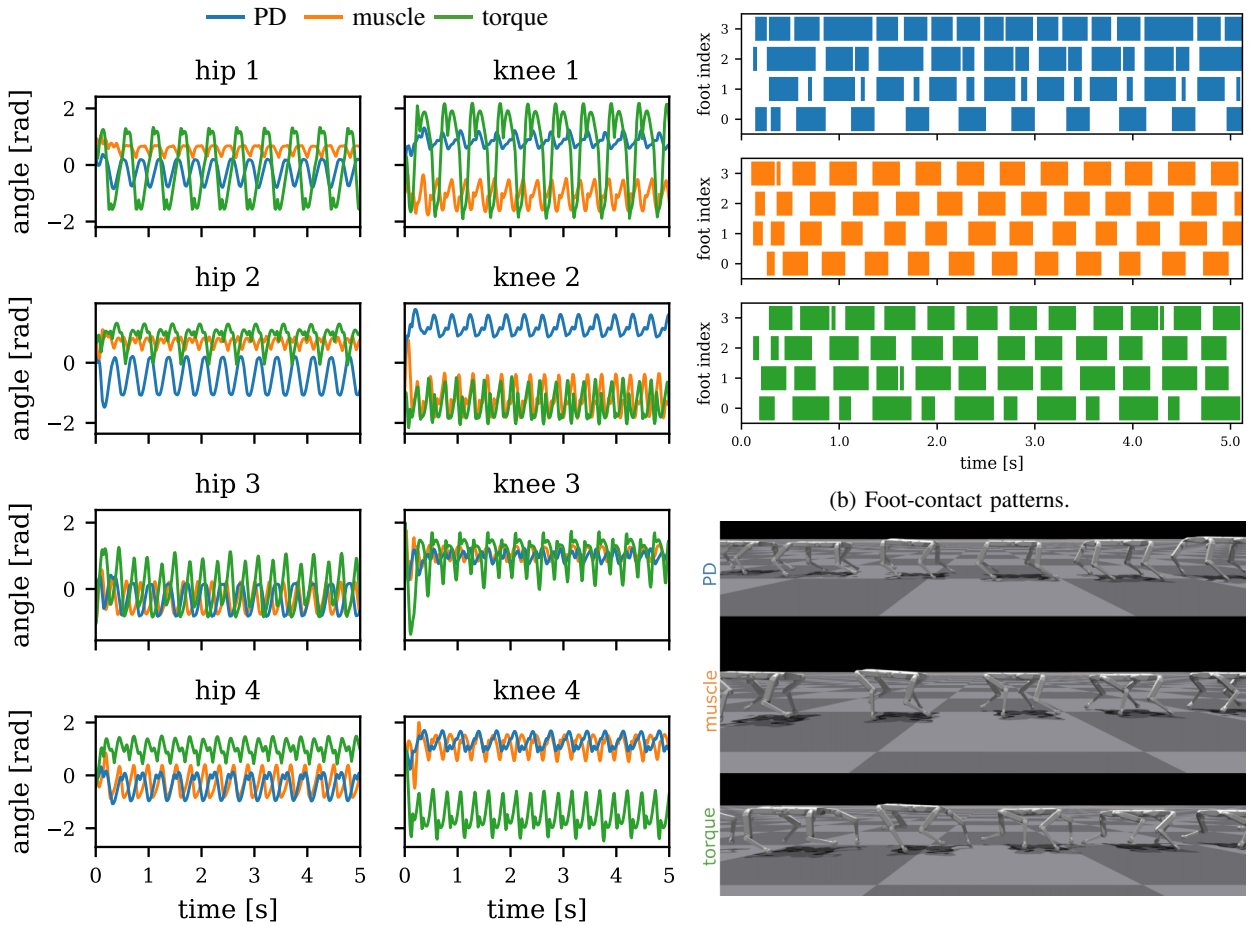
The sim-to-real policies were trained with observation noise and domain randomization, see Tab. I. Finally, we added a small amount of joint damping through a low-level controller with $k_{\text{damp}} = 0.08$ Nms/rad in all hardware experiments to stabilize the system and prevent damage. We also added it in simulation to minimize the sim-to-real gap.

We compare a policy trained with the real-time muscle-controller to a PD-controller for which we give alternating target positions and set the target velocity to zero (Fig. 7a). We use PD gains $k_{\text{stiff}} = 5.0$ and $k_{\text{damp}} = 0.1$, similar to [23]. Figure 7a shows that the muscle-controller is able to track the given joint angles similarly well to the PD-controller.

In a more realistic task, we train a walking policy with the muscle-controller in simulation and can reliably perform a running gait without additional training on the hardware. This successful sim-to-real transfer demonstrates that muscle actuation enables learning realistic and robust gaits without the need for excessive reward engineering. See Fig. 7c and the videos on the project website https://sites.google.com/view/emulatedmuscles.

IV. Conclusion

In this study, we have investigated the benefits of muscle-like actuation for quadruped locomotion tasks. We validated



(a) Joint angles over time.

(b) Foot-contact patterns.

(c) Learned gaits.

Fig. 4: **Muscles learn more regular walking patterns.** Using only the linear velocity and an action rate regularizing term as reward, the muscle-actuator learns the most regular walking pattern, lifting the feet above the ground while moving. In contrast, the torque-actuator agent walks with strongly extended motions that come close to the angular limits of the robotic limbs. The PD-controller agent drags the feet very closely over the ground, likely not generalizing to real-world hardware.

previous results on stability and robustness of the learned policies, while finding that the emulated actuator morphology also influences the learned gaits in an under-specified locomotion task. Finally, we proposed a tuning procedure that allowed us to validate the muscle model on a real robotic system under performance and stability constraints. We are able to execute policies learned entirely in simulation with little reward engineering on robotic hardware that runs a real-time emulated muscle model. These results showcase the potential of muscle-like actuators when combined with RL.

References

[1] X. B. Peng and M. van de Panne, "Learning locomotion skills using deeprl: Does the choice of action space matter?" in *Proceedings of the ACM SIGGRAPH / Eurographics Symposium on Computer Animation*, ser. SCA '17. New York, NY, USA: ACM, 2017, pp. 12:1–12:13. [Online]. Available: http://doi.acm.org/10.1145/3099564.3099567

[2] R. Martín-Martín, M. A. Lee, R. Gardner, S. Savarese, J. Bohg, and A. Garg, "Variable impedance control in end-effector space: An action space for reinforcement learning in contact-rich tasks," in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2019, pp. 1010–1017.

[3] E. Aljalbout, F. Frank, M. Karl, and P. van der Smagt, "On the role of the action space in robot manipulation learning and sim-to-real transfer," 2023.

[4] I. Wochner, P. Schumacher, G. Martius, D. Büchler, S. Schmitt, and D. Haeufle, "Learning with muscles: Benefits for data-efficiency and robustness in anthropomorphic tasks," in *Proceedings of the 6th Conference on Robot Learning (CoRL)*, ser. Proceedings of Machine Learning Research, vol. 205. PMLR, Dec. 2022, pp. 1178–1188. [Online]. Available: https://proceedings.mlr.press/v205/wochner23a.html

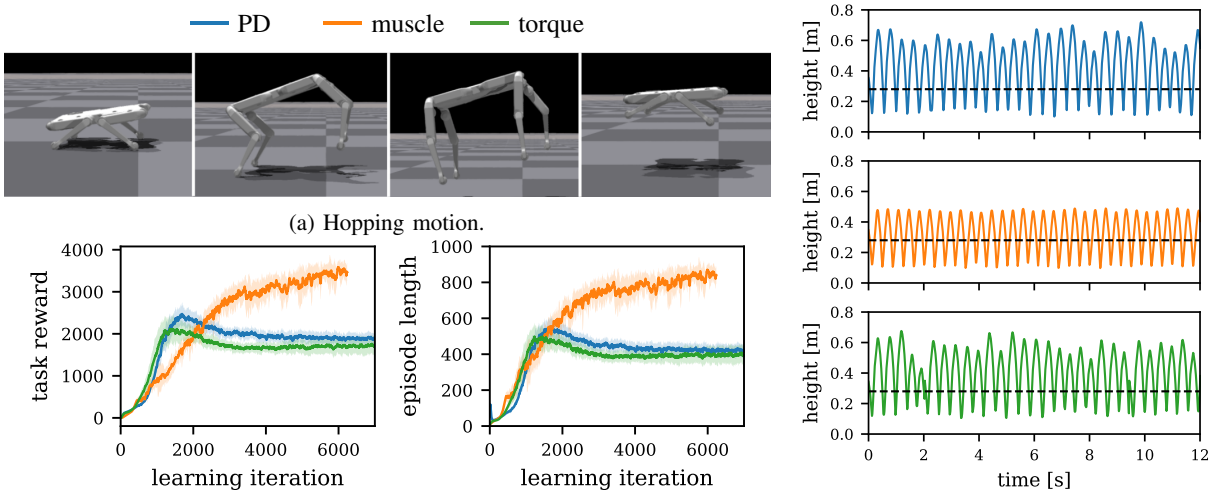
[5] Z. Batts, S. Song, and H. Geyer, "Toward a virtual neuromuscular control for robust walking in bipedal robots," in *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2015, pp. 6318–6323.

[6] A. Rai, R. Antonova, S. Song, W. Martin, H. Geyer, and C. Atkeson, "Bayesian optimization using domain knowledge on the atrias biped," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, 2018, pp. 1771–1778.

[7] A. J. van Soest and M. F. Bobbert, "The contribution of muscle properties in the control of explosive movements," *Biol Cybern*, vol. 69, no. 3, pp. 195–204, 1993.

[8] F. Izzi, A. Mo, S. Schmitt, A. Badri-Spröwitz, and D. F. B. Haeufle, "Muscle prestimulation tunes velocity preflex in simulated perturbed hopping," *Scientific Reports*, vol. 13, no. 1, p. 4559, Mar 2023. [Online]. Available: https://doi.org/10.1038/s41598-023-31179-6

[9] M. Araz, S. Weidner, F. Izzi, A. Badri-Spröwitz, T. Siebert, and D. F. B.

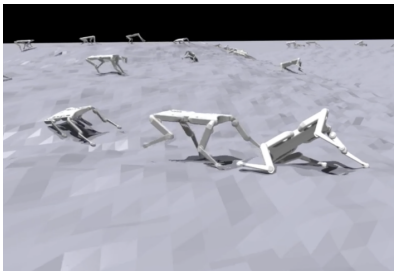


(a) Hopping motion.

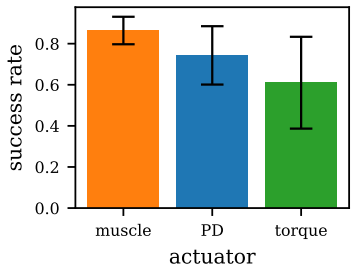
(b) Task performance.

(c) Height of the robot base.

Fig. 5: **Muscles learn more stable hopping.** Similar to results by Wochner et al. [4], we observe that muscle actuators learn more stable periodic hopping than torque or PD actuators. While the torque and especially the PD agents can occasionally achieve larger maximum heights than the muscle-agent, the periodic hopping behavior is less stable and achieves a smaller overall episode return. This task rewards large vertical velocities exponentially, the optimal behavior is therefore periodic jumping with maximum velocity and large displacement. We show the average base height while walking as a black line in (c) as a reference height for the hopping motion.



(a) Difficult terrain.



(b) Success rates for different actuators.

Fig. 6: **The muscle actuator is the most robust.** We rollout the different agents in an unseen terrain. The success rate is highest for the muscle actuator, followed by the PD and then the torque.

Haeufle, "Muscle preflex response to perturbations in locomotion: In vitro experiments and simulations with realistic boundary conditions," *Front Bioeng Biotechnol*, vol. 11, p. 1150170, Apr. 2023.

[10] D. F. B. Haeufle, S. Grimmer, and A. Seyfarth, "The role of intrinsic muscle properties for stable hopping—stability is achieved by the force–velocity relation," *Bioinspiration & Biomimetics*, vol. 5, no. 1, p. 016004, feb 2010. [Online]. Available: https://dx.doi.org/10.1088/1748-3182/5/1/016004

[11] A. Seyfarth, K. T. Kalveram, and . Geyer, Hartmut, "Simulating Muscle-Reflex Dynamics in a Simple Hopping Robot," in *Proceedings of Fachgespräche Autonome Mobile Systeme*. Springer, 2007, pp. 294–300. [Online]. Available: http://link.springer.com/10.1007/978-3-540-74764-2_45

[12] A. Mo, F. Izzi, E. C. Gönen, D. Haeufle, and A. Badri-Spröwitz, "Slack-based tunable damping leads to a trade-off between robustness and efficiency in legged locomotion," *Scientific Reports*, vol. 13, no. 1, feb 2023.

[13] B. Tondu and S. Zagal, "Mckibben artificial muscle can be in accordance with the hill skeletal muscle model," in *The First IEEE/RAS-EMBS International Conference on Biomedical Robotics and Biomechatronics, 2006. BioRob 2006.*, 2006, pp. 714–720.

[14] H. Ma, D. Büchler, B. Schölkopf, and M. Muehlebach, "Reinforcement learning with model-based feedforward inputs for robotic table tennis," *Autonomous Robots*, vol. 47, no. 8, pp. 1387–1403, Dec 2023. [Online]. Available: https://doi.org/10.1007/s10514-023-10140-6

[15] P. Rothemund, N. Kellaris, S. K. Mitchell, E. Acome, and C. Keplinger, "Hasel artificial muscles for a new generation of lifelike robots—recent progress and future opportunities," *Advanced Materials*, vol. 33, no. 19, p. 2003375, 2021. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/adma.202003375

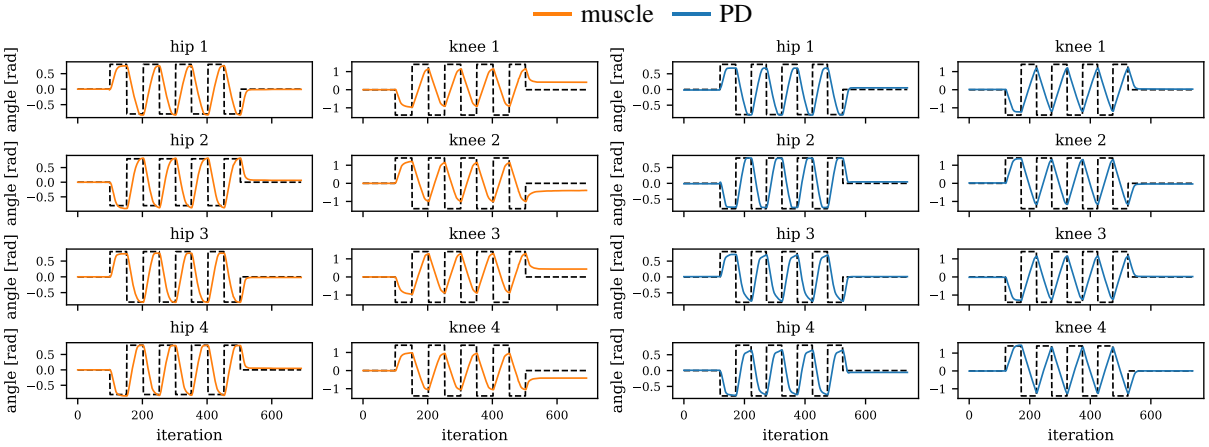
[16] D. Büchler, S. Guist, R. Calandra, V. Berenz, B. Schölkopf, and J. Peters, "Learning to play table tennis from scratch using muscular robots," *IEEE Transactions on Robotics*, vol. 38, no. 6, pp. 3850–3860, 2022.

[17] S. Guist, J. Schneider, A. Dittrich, V. Berenz, B. Schölkopf, and D. Büchler, "Hindsight states: Blending sim and real task elements for efficient reinforcement learning," in *Robotics: Science and Systems XIX*, Jul. 2023. [Online]. Available: https://www.roboticsproceedings.org/rss19/p038.html

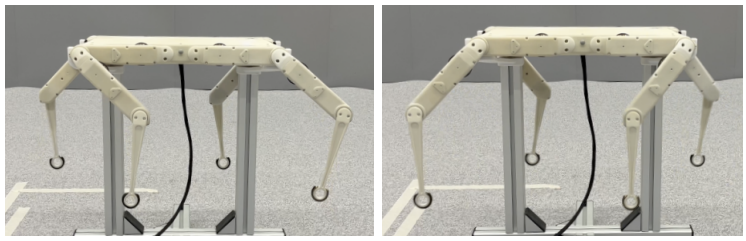
[18] H. Serhan, C. Nasr, and P. Henaff, "Designing a muscle like system based on pid controller and tuned by neural network," in *The 2006 IEEE International Joint Conference on Neural Network Proceedings*, 2006, pp. 4991–4998.

[19] H. Serhan and P. Henaff, "Muscle-like compliance in knee articulations improves biped robot walkings," in *Recent Advances in Robotic Systems*, G. Wang, Ed. Rijeka: IntechOpen, 2016, ch. 3. [Online]. Available: https://doi.org/10.5772/63746

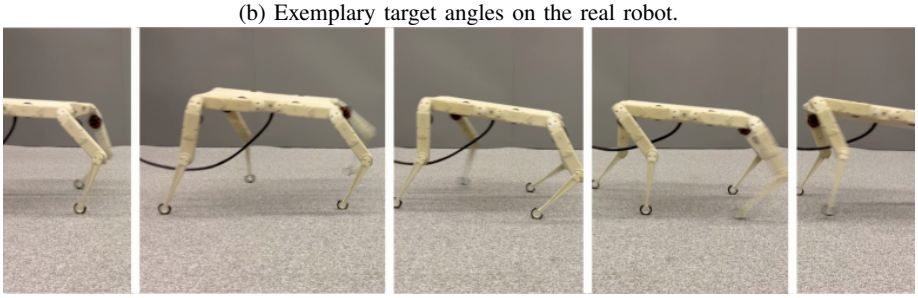
[20] J. Schneider, P. Schumacher, D. Häufle, B. Schölkopf, and D. Büchler,



(a) Joint angle tracking with muscle and PD actuators.



(b) Exemplary target angles on the real robot.



(c) Walking gait of the muscle-policy.

Fig. 7: **Sim-to-real transfer with an emulated muscle actuator.** (a) & (b) We trained policies for joint angle target tracking in simulation under the addition of simulated sensor noise and domain randomization. The muscle-policy is able to track targets with a real-time emulated muscle actuator on real robotic hardware through sim2real transfer. Note that the target angles were given at a fast frequency, making it challenging to track them perfectly in order to achieve a dynamic regime. (c) We train a muscle-driven policy in a walking task in simulation and deploy it on the real robot hardware.

"Investigating the impact of action representations in policy gradient algorithms," 2023.

[21] V. Makoviychuk, L. Wawrzyniak, Y. Guo, M. Lu, K. Storey, M. Macklin, D. Hoeller, N. Rudin, A. Allshire, A. Handa, and G. State, "Isaac gym: High performance gpu-based physics simulation for robot learning," 2021.

[22] N. Rudin, D. Hoeller, P. Reist, and M. Hutter, "Learning to walk in minutes using massively parallel deep reinforcement learning," in *5th Annual Conference on Robot Learning*, 2021. [Online]. Available: https://openreview.net/forum?id=wK2fDDJ5VcF

[23] C. Li, M. Vlastelica, S. Blaes, J. Frey, F. Grimminger, and G. Martius, "Learning agile skills via adversarial imitation of rough partial demonstrations," in *Proceedings of the 6th Conference on Robot Learning (CoRL)*, Dec. 2022. [Online]. Available: https://openreview.net/forum?id=x6INXlnUGro

[24] S. Chen, B. Zhang, M. W. Mueller, A. Rai, and K. Sreenath, "Learning torque control for quadrupedal locomotion," in *2023 IEEE-RAS 22nd International Conference on Humanoid Robots (Humanoids)*, 2023, pp. 1–8.

[25] W. Jakob, J. Rhinelander, and D. Moldovan, "pybind11 – seamless operability between C++11 and Python," 2017, https://github.com/pybind/pybind11.

[26] J. G. Ziegler and N. B. Nichols, "Optimum Settings for Automatic Controllers," *Transactions of the American Society of Mechanical Engineers*, vol. 64, no. 8, pp. 759–765, 12 2022. [Online]. Available: https://doi.org/10.1115/1.4019264

[27] D. Mattern, P. Schumacher, F. M. López, M. C. Raabe, M. R. Ernst, A. Aubret, and J. Triesch, "MIMo: A multi-modal infant model for studying cognitive development," 2023.

[28] E. Todorov, T. Erez, and Y. Tassa, "Mujoco: A physics engine for model-based control," in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2012, pp. 5026–5033.

[29] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," 2017.

[30] S. Mysore, B. Mabsout, R. Mancuso, and K. Saenko, "Regularizing action policies for smooth control with reinforcement learning," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021, pp. 1810–1816.

Bibliography

Abel, David et al. (2023). "A Definition of Continual Reinforcement Learning". In: *Thirty-seventh Conference on Neural Information Processing Systems*. URL: https://openreview.net/forum?id=ZZS9WEWYbD.

Al Borno, Mazen, Jennifer L. Hicks, and Scott L. Delp (2020). "The Effects of Motor Modularity on Performance, Learning and Generalizability in Upper-Extremity Reaching: A Computational Analysis". In: *Journal of The Royal Society Interface* 17.167, p. 20200011. DOI: 10.1098/rsif.2020.0011.

Alexander, R Mcneill (Dec. 2002). "Tendon elasticity and muscle function". en. In: *Comp. Biochem. Physiol. A Mol. Integr. Physiol.* 133.4, pp. 1001–1011.

Allshire, Arthur et al. (2022). "Transferring Dexterous Manipulation from GPU Simulation to a Remote Real-World TriFinger". In: *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 11802–11809. DOI: 10.1109/IROS47612.2022.9981458.

Amin, Susan et al. (2021). *A Survey of Exploration Methods in Reinforcement Learning*. arXiv: 2109.00157 [cs.LG]. URL: https://arxiv.org/abs/2109.00157.

Azocar, Alejandro F. et al. (Oct. 2020). "Design and clinical implementation of an open-source bionic leg". In: *Nature Biomedical Engineering* 4.10, pp. 941–953. ISSN: 2157-846X. DOI: 10.1038/s41551-020-00619-3. URL: https://doi.org/10.1038/s41551-020-00619-3.

Barbera, Vittorio La et al. (2021). "OstrichRL: A Musculoskeletal Ostrich Simulation to Study Bio-mechanical Locomotion". In: *Deep RL Workshop NeurIPS 2021*. URL: https://openreview.net/forum?id=7KzszSyQP0D.

Batts, Zachary, Seungmoon Song, and Hartmut Geyer (2015). "Toward a virtual neuromuscular control for robust walking in bipedal robots". In: *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 6318–6323. DOI: 10.1109/IROS.2015.7354279.

Bayer, A. et al. (2017). "The influence of biophysical muscle properties on simulating fast human arm movements". In: *Computer Methods in Biomechanics and Biomedical Engineering* 20.8, pp. 803–821. ISSN: 1476-8259. DOI: 10.1080/10255842.2017.1293663.

Bellegarda, Guillaume et al. (2022). "Robust High-Speed Running for Quadruped Robots via Deep Reinforcement Learning". In: *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 10364–10370. DOI: 10.1109/IROS47612.2022.9982132.

Bernstein, N.A. (1967). *The Co-ordination and Regulation of Movements*. Pergamon Press. URL: https://books.google.de/books?id=kX5OAQAAIAAJ.

Berret, Bastien et al. (Nov. 2024). "Co-contraction embodies uncertainty: An optimal feedforward strategy for robust motor control". In: *PLOS Computational Biology*

20.11, pp. 1–27. DOI: 10.1371/journal.pcbi.1012598. URL: https://doi.org/10.1371/journal.pcbi.1012598.

Botvinick, Matthew et al. (2020). "Deep Reinforcement Learning and Its Neuroscientific Implications". In: *Neuron* 107.4, pp. 603–616. ISSN: 0896-6273. DOI: https://doi.org/10.1016/j.neuron.2020.06.014. URL: https://www.sciencedirect.com/science/article/pii/S0896627320304682.

Bruel, Alice et al. (2022). "Investigation of neural and biomechanical impairments leading to pathological toe and heel gaits using neuromusculoskeletal modelling". In: *The Journal of Physiology* 600.11, pp. 2691–2712. DOI: https://doi.org/10.1113/JP282609. eprint: https://physoc.onlinelibrary.wiley.com/doi/pdf/10.1113/JP282609. URL: https://physoc.onlinelibrary.wiley.com/doi/abs/10.1113/JP282609.

Büchler, Dieter et al. (2022). "Learning to play table tennis from scratch using muscular robots". In: *IEEE Transactions on Robotics* 38.6, pp. 3850–3860.

Bulat, Muge et al. (2019). "Musculoskeletal Simulation Tools for Understanding Mechanisms of Lower-Limb Sports Injuries". In: *Current Sports Medicine Reports* 18.6. ISSN: 1537-8918. URL: https://journals.lww.com/acsm-csmr/fulltext/2019/06000/musculoskeletal_simulation_tools_for_understanding.6.aspx.

Bunz, Elsa K., Louisa H. Pawusch, and Syn Schmitt (2024). "Optimizing Reflex-Based Neuromusculoskeletal Walking Model on Rough Terrain Reveals Increased Robustness and Key Stabilizing Reflexes". In: *2024 10th IEEE RAS/EMBS International Conference for Biomedical Robotics and Biomechatronics (BioRob)*, pp. 477–482. DOI: 10.1109/BioRob60516.2024.10719737.

Caggiano, Vittorio, Sudeep Dasari, and Vikash Kumar (June 15, 2023). *MyoDex: A Generalizable Prior for Dexterous Manipulation*. URL: https://openreview.net/forum?id=iYBTiYzN0A (visited on 06/27/2023).

Caggiano, Vittorio et al. (2022). *MyoSuite – A contact-rich simulation suite for musculoskeletal motor control*. https://github.com/facebookresearch/myosuite. DOI: 10.48550/ARXIV.2205.13600. URL: https://sites.google.com/view/myosuite.

Caggiano, Vittorio et al. (2024). *MyoChallenge 2023: Towards Human-Level Dexterity and Agility*. URL: https://openreview.net/forum?id=3A84lx1JFh.

Chane-Sane, Elliot et al. (2024). "CaT: Constraints as Terminations for Legged Locomotion Reinforcement Learning". In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*.

Chen, Shuxiao et al. (Dec. 2023). "Learning Torque Control for Quadrupedal Locomotion". In: *IEEE International Conference on Humanoid Robots (Humanoids)*. Austin, TX, pp. 1–8.

Clemente, Christofer J., Friedl De Groote, and Taylor J. M. Dick (Oct. 2024). "Predictive musculoskeletal simulations reveal the mechanistic link between speed, posture and energetics among extant mammals". In: *Nature Communications* 15.1, p. 8594. ISSN: 2041-1723. DOI: 10.1038/s41467-024-52924-z. URL: https://doi.org/10.1038/s41467-024-52924-z.

Collings, A J et al. (Apr. 2022). "Functional analysis of anuran pelvic and thigh anatomy using musculoskeletal modelling of Phlyctimantis maculatus". en. In: *Front. Bioeng. Biotechnol.* 10, p. 806174.

Cotton, R. James (2024). *Differentiable Biomechanics Unlocks Opportunities for Markerless Motion Capture*. arXiv: 2402.17192 [cs.CV]. URL: https://arxiv.org/abs/2402.17192.

De Groote, Friedl and Antoine Falisse (Mar. 2021). "Perspective on musculoskeletal modelling and predictive simulations of human movement to assess the neuromechanics of gait". en. In: *Proc. Biol. Sci.* 288.1946, p. 20202432.

De Groote, Friedl and Antoine Falisse (2021). "Perspective on musculoskeletal modelling and predictive simulations of human movement to assess the neuromechanics of gait". In: *Proceedings of the Royal Society B: Biological Sciences* 288.1946. ISSN: 14712954. DOI: 10.1098/rspb.2020.2432.

Deimel, Raphael et al. (2017). "Automated co-design of soft hand morphology and control strategy for grasping". In: *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 1213–1218. DOI: 10.1109/IROS.2017.8202294.

Delp, Scott L. et al. (2007). "OpenSim: Open-Source Software to Create and Analyze Dynamic Simulations of Movement." In: *IEEE Trans. Biomed. Eng.* 54.11, pp. 1940–1950. URL: http://dblp.uni-trier.de/db/journals/tbe/tbe54.html#DelpAALHJGT07.

Dembia, Christopher L. et al. (Dec. 2021). "OpenSim Moco: Musculoskeletal optimal control". In: *PLOS Computational Biology* 16.12, pp. 1–21. DOI: 10.1371/journal.pcbi.1008493. URL: https://doi.org/10.1371/journal.pcbi.1008493.

Denizdurduran, Berat, Henry Markram, and Marc-Oliver Gewaltig (Dec. 2022). "Optimum trajectory learning in musculoskeletal systems with model predictive control and deep reinforcement learning". In: *Biological Cybernetics* 116.5, pp. 711–726. ISSN: 1432-0770. DOI: 10.1007/s00422-022-00940-x. URL: https://doi.org/10.1007/s00422-022-00940-x.

Der, Ralf and Georg Martius (2015). "Novel plasticity rule can explain the development of sensorimotor intelligence". In: *Proceedings of the National Academy of Sciences* 112.45, E6224–E6232. DOI: 10.1073/pnas.1508400112. eprint: https://www.pnas.org/doi/pdf/10.1073/pnas.1508400112. URL: https://www.pnas.org/doi/abs/10.1073/pnas.1508400112.

Driess, Danny et al. (2018). "Learning to control redundant musculoskeletal systems with neural networks and SQP: exploiting muscle properties". In: *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, pp. 6461–6468.

Eriten, Melih and Harry Dankowicz (2009). "A rigorous dynamical-systems-based analysis of the self-stabilizing influence of muscles". In: *Journal of Biomechanical Engineering* 131.1, p. 011011.

Falisse, Antoine et al. (2019). "Rapid predictive simulations with complex musculoskeletal models suggest that diverse healthy and pathological human gaits can emerge from similar control strategies". In: *Journal of The Royal Society Interface* 16.157, p. 20190402. DOI: 10.1098/rsif.2019.0402. eprint: https://royalsocietypublishing.org/doi/pdf/10.1098/rsif.2019.

0402. URL: https://royalsocietypublishing.org/doi/abs/10.1098/rsif.2019.0402.

Feng, Yusen, Xiyan Xu, and Libin Liu (2023). *MuscleVAE: Model-Based Controllers of Muscle-Actuated Characters*. arXiv: 2312.07340 [cs.GR].

Flash, T and N Hogan (July 1985). "The coordination of arm movements: an experimentally confirmed mathematical model". en. In: *J. Neurosci.* 5.7, pp. 1688–1703.

Gaudet, Brian, Richard Linares, and Roberto Furfaro (2018). "Integrated guidance and control for pinpoint mars landing using reinforcement learning". English (US). In: *AAS/AIAA Astrodynamics Specialist Conference, 2018*. Ed. by Puneet Singla et al. Advances in the Astronautical Sciences. Publisher Copyright: © 2018 Univelt Inc. All rights reserved.; AAS/AIAA Astrodynamics Specialist Conference, 2018 ; Conference date: 19-08-2018 Through 23-08-2018. Univelt Inc., pp. 3135–3154. ISBN: 9780877036579.

Geijtenbeek, Thomas (Nov. 2021). *The Hyfydy Simulation Software*. URL: https://hyfydy.com.

Geyer, H. and H. Herr (June 2010). "A muscle-reflex model that encodes principles of legged mechanics produces human walking dynamics and muscle activities". In: *IEEE Trans Neural Syst Rehabil Eng* 18.3, pp. 263–273.

Ghazi-Zahedi, Keyan et al. (2016). "Evaluating Morphological Computation in Muscle and DC-Motor Driven Models of Hopping Movements". In: *Frontiers in Robotics and AI* 3. ISSN: 2296-9144. DOI: 10.3389/frobt.2016.00042. URL: https://www.frontiersin.org/journals/robotics-and-ai/articles/10.3389/frobt.2016.00042.

Goldsmith, Werner and John T. Frasier (1960). "Impact: the theory and physical behaviour of colliding solids." In: URL: https://api.semanticscholar.org/CorpusID:123157883.

Gong, Daoxiong and Jianjun Yu (2022). "Design and Control of the McKibben Artificial Muscles Actuated Humanoid Manipulator". In: *Rehabilitation of the Human Bone-Muscle System*. Ed. by Adrian Olaru. Rijeka: IntechOpen. Chap. 3. DOI: 10.5772/intechopen.101761. URL: https://doi.org/10.5772/intechopen.101761.

Gordon, Joanne C et al. (June 2020). "Tuning of feedforward control enables stable muscle force-length dynamics after loss of autogenic proprioceptive feedback". In: *eLife* 9. Ed. by K VijayRaghavan, Noah J Cowan, and Lena H Ting, e53908. ISSN: 2050-084X. DOI: 10.7554/eLife.53908. URL: https://doi.org/10.7554/eLife.53908.

Gozlan, Yoni et al. (2024). *OpenCapBench: A Benchmark to Bridge Pose Estimation and Biomechanics*. arXiv: 2406.09788 [cs.CV]. URL: https://arxiv.org/abs/2406.09788.

Grimminger, F. et al. (2020). "An Open Torque-Controlled Modular Robot Architecture for Legged Locomotion Research". In: *IEEE Robotics and Automation Letters* 5.2, pp. 3650–3657. DOI: 10.1109/LRA.2020.2976639.

Haeufle, D F B, S Grimmer, and A Seyfarth (Feb. 2010a). "The role of intrinsic muscle properties for stable hopping—stability is achieved by the force–velocity relation". In: *Bioinspiration & Biomimetics* 5.1, p. 016004. DOI: 10.1088/1748-

3182/5/1/016004. URL: https://dx.doi.org/10.1088/1748-3182/5/1/016004.

Haeufle, Daniel F. B. et al. (2020a). "Muscles Reduce Neuronal Information Load: Quantification of Control Effort in Biological vs. Robotic Pointing and Walking". In: *Frontiers in Robotics and AI* 7. ISSN: 2296-9144. DOI: 10.3389/frobt.2020.00077. URL: https://www.frontiersin.org/journals/robotics-and-ai/articles/10.3389/frobt.2020.00077.

Haeufle, Daniel F.B., S. Grimmer, and * Seyfarth A. (2010b). "The role of intrinsic muscle properties for stable hopping - stability is achieved by the force-velocity relation". In: *Bioinspiration & Biomimetics* 5.1, p. 016004. ISSN: 17483182. DOI: 10.1088/1748-3182/5/1/016004.

Haeufle, Daniel FB et al. (2020b). "Muscles reduce neuronal information load: quantification of control effort in biological vs. robotic pointing and walking". In: *Frontiers in Robotics and AI* 7, p. 77.

Haeufle, DFB et al. (2014). "Hill-type muscle model with serial damping and eccentric force–velocity relation". In: *Journal of biomechanics* 47.6, pp. 1531–1536.

He, Tairan et al. (2024). "Learning human-to-humanoid real-time whole-body teleoperation". In: *arXiv preprint arXiv:2403.04436*.

Hellera, M. O. et al. (2001). "Musculo-skeletal loading conditions at the hip during walking and stair climbing". In: URL: https://api.semanticscholar.org/CorpusID:29836263.

Hunt, K. H. and F. R. E. Crossley (June 1975). "Coefficient of Restitution Interpreted as Damping in Vibroimpact". In: *Journal of Applied Mechanics* 42.2, pp. 440–445. ISSN: 0021-8936. DOI: 10.1115/1.3423596. eprint: https://asmedigitalcollection.asme.org/appliedmechanics/article-pdf/42/2/440/5454660/440_1.pdf. URL: https://doi.org/10.1115/1.3423596.

Hwangbo, Jemin et al. (2019). "Learning agile and dynamic motor skills for legged robots". In: *Science Robotics* 4.26, eaau5872. DOI: 10.1126/scirobotics.aau5872. eprint: https://www.science.org/doi/pdf/10.1126/scirobotics.aau5872. URL: https://www.science.org/doi/abs/10.1126/scirobotics.aau5872.

Ibrahim, Sinan et al. (2024). "Comprehensive Overview of Reward Engineering and Shaping in Advancing Reinforcement Learning Applications". In: *IEEE Access* 12, pp. 175473–175500. DOI: 10.1109/ACCESS.2024.3504735.

Ishikawa, Masaki et al. (Aug. 2005). "Muscle-tendon interaction and elastic energy usage in human walking". en. In: *J. Appl. Physiol.* 99.2, pp. 603–608.

Jiang, Yifeng et al. (July 2019). "Synthesis of biologically realistic human motion using joint torque actuation". In: *ACM Trans. Graph.* 38.4. ISSN: 0730-0301. DOI: 10.1145/3306346.3322966. URL: https://doi.org/10.1145/3306346.3322966.

Jones, Rachel et al. (May 2024). "Delayed center of mass feedback in elderly humans leads to greater muscle co-contraction and altered balance strategy under perturbed balance: A predictive musculoskeletal simulation study". In: *PLOS ONE* 19.5, pp. 1–26. DOI: 10.1371/journal.pone.0296548. URL: https://doi.org/10.1371/journal.pone.0296548.

Jumper, John et al. (Aug. 2021). "Highly accurate protein structure prediction with AlphaFold". In: *Nature* 596.7873, pp. 583–589. ISSN: 1476-4687. DOI: 10.1038/s41586-021-03819-2. URL: https://doi.org/10.1038/s41586-021-03819-2.

Kasuga, Shoko et al. (2022). "Integration of proprioceptive and visual feedback during online control of reaching". In: *Journal of Neurophysiology* 127.2. PMID: 34907796, pp. 354–372. DOI: 10.1152/jn.00639.2020. eprint: https://doi.org/10.1152/jn.00639.2020. URL: https://doi.org/10.1152/jn.00639.2020.

Kaufmann, Elia et al. (Aug. 2023). "Champion-level drone racing using deep reinforcement learning". In: *Nature* 620.7976, pp. 982–987. ISSN: 1476-4687. DOI: 10.1038/s41586-023-06419-4. URL: https://doi.org/10.1038/s41586-023-06419-4.

Keyser, Johannes et al. (June 2023). "Late integration of vision and proprioception during perturbed reaches". en. In: *J. Neurophysiol.* 129.6, pp. 1282–1292.

Keysers, Christian and Valeria Gazzola (Apr. 2014). "Hebbian learning and predictive mirror neurons for actions, sensations and emotions". en. In: *Philos. Trans. R. Soc. Lond. B Biol. Sci.* 369.1644, p. 20130175.

Kistemaker, Dinant a, Arthur J Van Soest, and * Bobbert Maarten F (May 2006). "Is equilibrium point control feasible for fast goal-directed single-joint movements?" In: *Journal of Neurophysiology* 95.5, pp. 2898–912. ISSN: 0022-3077. DOI: 10.1152/jn.00983.2005.

Kistemaker, Dinant A., Jeremy D. Wong, and Paul L. Gribble (2014). "The cost of moving optimally: kinematic path selection". In: *Journal of Neurophysiology* 112.8. PMID: 24944215, pp. 1815–1824. DOI: 10.1152/jn.00291.2014. eprint: https://doi.org/10.1152/jn.00291.2014. URL: https://doi.org/10.1152/jn.00291.2014.

Kistemaker, Dinant A. et al. (2013). "Control of position and movement is simplified by combined muscle spindle and Golgi tendon organ feedback". In: *Journal of Neurophysiology* 109.4. PMID: 23100138, pp. 1126–1139. DOI: 10.1152/jn.00751.2012. eprint: https://doi.org/10.1152/jn.00751.2012. URL: https://doi.org/10.1152/jn.00751.2012.

Koelewijn, Anne D. and Antonie J. van den Bogert (2020). "A solution method for predictive simulations in a stochastic environment". In: *Journal of Biomechanics* 104, p. 109759. ISSN: 0021-9290. DOI: https://doi.org/10.1016/j.jbiomech.2020.109759. URL: https://www.sciencedirect.com/science/article/pii/S0021929020301755.

Kumar, Ashish et al. (2021). "Rma: Rapid motor adaptation for legged robots". In.

Lassmann, Christian et al. (July 2023). "Dysfunctional neuro-muscular mechanisms explain gradual gait changes in prodromal spastic paraplegia". In: *Journal of NeuroEngineering and Rehabilitation* 20.1, p. 90. ISSN: 1743-0003. DOI: 10.1186/s12984-023-01206-8. URL: https://doi.org/10.1186/s12984-023-01206-8.

Laßmann, Christian et al. (2022). "Specific Gait Changes in Prodromal Hereditary Spastic Paraplegia Type 4: preSPG4 Study". In: *Movement Disorders* 37.12, pp. 2417–2426. DOI: https://doi.org/10.1002/mds.29199. eprint: https://movementdisorders.onlinelibrary.wiley.com/doi/

pdf/10.1002/mds.29199. URL: https://movementdisorders.onlinelibrary.wiley.com/doi/abs/10.1002/mds.29199.

Lee, I-Chieh, Ming Liu, and He Huang (2024). "Enhancing User-Prosthesis Integration Through Intelligent Transparency". In: *2024 10th IEEE RAS/EMBS International Conference for Biomedical Robotics and Biomechatronics (BioRob)*, pp. 471–476. DOI: 10.1109/BioRob60516.2024.10719813.

Lee, Seunghwan et al. (July 2019). "Scalable Muscle-Actuated Human Simulation and Control". In: *ACM Trans. Graph.* 38.4. ISSN: 0730-0301. DOI: 10.1145/3306346.3322972. URL: https://doi.org/10.1145/3306346.3322972.

Li, Chenhao et al. (2023). "Learning agile skills via adversarial imitation of rough partial demonstrations". In: *Conference on Robot Learning*. PMLR, pp. 342–352.

Loeb, Gerald E. (Dec. 2012). "Optimal isn't good enough". In: *Biological Cybernetics* 106.11, pp. 757–765. ISSN: 1432-0770. DOI: 10.1007/s00422-012-0514-6. URL: https://doi.org/10.1007/s00422-012-0514-6.

Luo, Shuzhen et al. (Dec. 2021). *Robust Walking Control of a Lower Limb Rehabilitation Exoskeleton Coupled with a Musculoskeletal Model via Deep Reinforcement Learning*. en. preprint. In Review. DOI: 10.21203/rs.3.rs-1212542/v1. URL: https://www.researchsquare.com/article/rs-1212542/v1 (visited on 06/09/2022).

Luo, Shuzhen et al. (June 2024a). "Experiment-free exoskeleton assistance via learning in simulation". en. In: *Nature* 630.8016, pp. 353–359.

Luo, Zhengyi et al. (2023). "Perpetual Humanoid Control for Real-time Simulated Avatars". In: *International Conference on Computer Vision (ICCV)*.

Luo, Zhengyi et al. (2024b). "Universal Humanoid Motion Representations for Physics-Based Control". In: *The Twelfth International Conference on Learning Representations*. URL: https://openreview.net/forum?id=OrOd8PxOO2.

Ma, Hao et al. (Dec. 2023). "Reinforcement learning with model-based feedforward inputs for robotic table tennis". In: *Autonomous Robots* 47.8, pp. 1387–1403. ISSN: 1573-7527. DOI: 10.1007/s10514-023-10140-6. URL: https://doi.org/10.1007/s10514-023-10140-6.

Mahmood, Naureen et al. (Oct. 2019). "AMASS: Archive of Motion Capture as Surface Shapes". In: *International Conference on Computer Vision*, pp. 5442–5451.

Makoviychuk, Viktor et al. (2021). *Isaac Gym: High Performance GPU-Based Physics Simulation For Robot Learning*. arXiv: 2108.10470 [cs.RO].

Margolis, Gabriel et al. (2022). "Rapid Locomotion via Reinforcement Learning". In: *Robotics: Science and Systems*.

Martín-Martín, Roberto et al. (2019). "Variable Impedance Control in End-Effector Space. An Action Space for Reinforcement Learning in Contact Rich Tasks". In: *Proceedings of the International Conference of Intelligent Robots and Systems (IROS)*.

McNeill Alexander, R (2002). "Tendon elasticity and muscle function". In: *Comparative Biochemistry and Physiology Part A: Molecular & Integrative Physiology* 133.4, pp. 1001–1011. ISSN: 1095-6433. DOI: https://doi.org/10.1016/S1095-6433(02)00143-5. URL: https://www.sciencedirect.com/science/article/pii/S1095643302001435.

Michaud, Benjamin et al. (2022). "Bioptim, a python framework for musculoskeletal optimal control in biomechanics". In: *IEEE Transactions on Systems, Man, and Cybernetics: Systems*.

Millard, Matthew et al. (Feb. 2013). "Flexing computational muscle: modeling and simulation of musculotendon dynamics." In: *Journal of biomechanical engineering* 135.2, p. 021005. ISSN: 1528-8951. DOI: 10.1115/1.4023390.

Millard, Matthew et al. (2024). "A benchmark of muscle models to length changes great and small". In: *Journal of the Mechanical Behavior of Biomedical Materials* 160, p. 106740. ISSN: 1751-6161. DOI: https://doi.org/10.1016/j.jmbbm.2024.106740. URL: https://www.sciencedirect.com/science/article/pii/S1751616124003722.

Mo, An et al. (Feb. 2023). "Slack-based tunable damping leads to a trade-off between robustness and efficiency in legged locomotion". In: *Scientific Reports* 13.1. DOI: 10.1038/s41598-023-30318-3.

Morecki, Adam, J. S. Ekiel, and Kazimierz Fidelus (1984). "Cybernetic Systems of Limb Movements in Man, Animals and Robots". In: URL: https://api.semanticscholar.org/CorpusID:60583790.

Nasiriany, Soroush et al. (2024). "RoboCasa: Large-Scale Simulation of Everyday Tasks for Generalist Robots". In: *Robotics: Science and Systems (RSS)*.

Nikoo, Ali and Thomas K. Uchida (2022). "Be Careful What You Wish for: Cost Function Sensitivity in Predictive Simulations for Assistive Device Design". In: *Symmetry* 14.12. ISSN: 2073-8994. DOI: 10.3390/sym14122534. URL: https://www.mdpi.com/2073-8994/14/12/2534.

Nitschke, Marlies et al. (Oct. 2020). "Efficient trajectory optimization for curved running using a 3D musculoskeletal model with implicit dynamics". In: *Scientific Reports* 10.1, p. 17655. ISSN: 2045-2322. DOI: 10.1038/s41598-020-73856-w. URL: https://doi.org/10.1038/s41598-020-73856-w.

Nölle, Lennart V. et al. (Nov. 2024). "Using muscle-tendon load limits to assess unphysiological musculoskeletal model deformation and Hill-type muscle parameter choice". In: *PLOS ONE* 19.11, pp. 1–18. DOI: 10.1371/journal.pone.0302949. URL: https://doi.org/10.1371/journal.pone.0302949.

Ong, Carmichael F. et al. (Oct. 2019). "Predicting gait adaptations due to ankle plantarflexor muscle weakness and contracture using physics-based musculoskeletal simulations". In: *PLOS Computational Biology* 15.10, pp. 1–27. DOI: 10.1371/journal.pcbi.1006993. URL: https://doi.org/10.1371/journal.pcbi.1006993.

Park, Jungnam et al. (2022). "Generative GaitNet". In: *ACM SIGGRAPH 2022 Conference Proceedings*. SIGGRAPH '22. Vancouver, BC, Canada: Association for Computing Machinery. ISBN: 9781450393379. DOI: 10.1145/3528233.3530717. URL: https://doi.org/10.1145/3528233.3530717.

Peng, Xue Bin and Michiel van de Panne (2017). "Learning Locomotion Skills Using DeepRL: Does the Choice of Action Space Matter?" In: *Proceedings of the ACM SIGGRAPH / Eurographics Symposium on Computer Animation*. SCA '17. Los Angeles, California: ACM, 12:1–12:13. ISBN: 978-1-4503-5091-4. DOI: 10.1145/3099564.3099567. URL: http://doi.acm.org/10.1145/3099564.3099567.

Peng, Xue Bin et al. (July 2021). "AMP: Adversarial Motion Priors for Stylized Physics-Based Character Control". In: *ACM Trans. Graph.* 40.4. DOI: 10.1145/3450626.3459670. URL: http://doi.acm.org/10.1145/3450626.3459670.

Piche, Elodie et al. (2022). "Metabolic cost and co-contraction during walking at different speeds in young and old adults". In: *Gait & Posture* 91, pp. 111–116. ISSN: 0966-6362. DOI: https://doi.org/10.1016/j.gaitpost.2021.10.014. URL: https://www.sciencedirect.com/science/article/pii/S096663622100535X.

Pratt, G.A. and M.M. Williamson (1995). "Series elastic actuators". In: *Proceedings 1995 IEEE/RSJ International Conference on Intelligent Robots and Systems. Human Robot Interaction and Cooperative Robots*. Vol. 1, 399–406 vol.1. DOI: 10.1109/IROS.1995.525827.

Price, Mark A., Brian R. Umberger, and Frank C. Sup (2019). "Dynamic optimization of Gait with a Generalized Lower-Limb Prosthesis Model". In: *2019 IEEE 16th International Conference on Rehabilitation Robotics (ICORR)*, pp. 734–739. DOI: 10.1109/ICORR.2019.8779532.

Prilutsky, Boris I and Vladimir M Zatsiorsky (Jan. 2002). "Optimization-based models of muscle coordination". en. In: *Exerc. Sport Sci. Rev.* 30.1, pp. 32–38.

Rajagopal, Apoorva et al. (Oct. 2016a). "Full-body musculoskeletal model for muscle-driven simulation of human gait". In: *IEEE Trans. Biomed. Eng.* 63.10, pp. 2068–2079.

Rajagopal, Apoorva et al. (2016b). "Full body musculoskeletal model for muscle-driven simulation of human gait". In: *IEEE Transactions on Biomedical Engineering* 63.10, pp. 2068–2079. ISSN: 0018-9294. DOI: 10.1109/TBME.2016.2586891. eprint: TBME.2016.2586891.

Ramos, Joao et al. (2021). "HOPPY: An Open-source Kit for Education with Dynamic Legged Robots". In: *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 4312–4318. DOI: 10.1109/IROS51168.2021.9636108.

Ramsauer, Hubert et al. (2021). "Hopfield Networks is All You Need". In: *International Conference on Learning Representations*. URL: https://openreview.net/forum?id=tL89RnzIiCd.

Rokhmanova, Nataliya et al. (2024). "IMU-Based Kinematics Estimation Accuracy Affects Gait Retraining Using Vibrotactile Cues". In: *IEEE Transactions on Neural Systems and Rehabilitation Engineering* 32, pp. 1005–1012. DOI: 10.1109/TNSRE.2024.3365204.

Rudin, Nikita et al. (2021). "Learning to Walk in Minutes Using Massively Parallel Deep Reinforcement Learning". In: *5th Annual Conference on Robot Learning*. URL: https://openreview.net/forum?id=wK2fDDJ5VcF.

Rugy, Aymar de, Gerald E Loeb, and Timothy J Carroll (May 2012). "Muscle coordination is habitual rather than optimal". en. In: *J. Neurosci.* 32.21, pp. 7384–7391.

Ryu, Hansol X. and Arthur D. Kuo (June 2021). "An optimality principle for locomotor central pattern generators". In: *Scientific Reports* 11.1, p. 13140. ISSN: 2045-2322. DOI: 10.1038/s41598-021-91714-1. URL: https://doi.org/10.1038/s41598-021-91714-1.

Sahara, Yuta et al. (2024). "Construction of Musculoskeletal Simulation for Shoulder Complex with Ligaments and Its Validation via Model Predictive Control". In: *2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 327–333. DOI: 10.1109/IROS58592.2024.10802465.

Scherb, David, Sandro Wartzack, and Jörg Miehling (2023). "Modelling the interaction between wearable assistive devices and digital human models—A systematic review". In: *Frontiers in Bioengineering and Biotechnology* 10. ISSN: 2296-4185. DOI: 10.3389/fbioe.2022.1044275. URL: https://www.frontiersin.org/journals/bioengineering-and-biotechnology/articles/10.3389/fbioe.2022.1044275.

Schmitt, Syn (2022). *demoa-base: A Biophysics Simulator for Muscle-driven Motion*. Version V1. DOI: 10.18419/darus-2550. URL: https://doi.org/10.18419/darus-2550.

Schumacher, Pierre et al. (2024). "Learning to Control Emulated Muscles in Real Robots: A Software Test Bed for Bio-Inspired Actuators in Hardware". In: *2024 10th IEEE RAS/EMBS International Conference for Biomedical Robotics and Biomechatronics (BioRob)*, pp. 806–813. DOI: 10.1109/BioRob60516.2024.10719699.

Serhan, H.J., C.G. Nasr, and P. Henaff (2006). "Designing a Muscle like System Based on PID Controller and Tuned by Neural Network". In: *The 2006 IEEE International Joint Conference on Neural Network Proceedings*, pp. 4991–4998. DOI: 10.1109/IJCNN.2006.247203.

Seth, Ajay et al. (2011). "OpenSim: a musculoskeletal modeling and simulation framework for in silico investigations and exchange". In: *Procedia IUTAM* 2. IUTAM Symposium on Human Body Dynamics, pp. 212–232. ISSN: 2210-9838. DOI: https://doi.org/10.1016/j.piutam.2011.04.021. URL: https://www.sciencedirect.com/science/article/pii/S2210983811000228.

Seyfarth, Andre, Karl Theodor Kalveram, and Hartmut Geyer (2007). "Simulating Muscle-Reflex Dynamics in a Simple Hopping Robot". In: *Autonome Mobile Systeme 2007*. Ed. by Karsten Berns and Tobias Luksch. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 294–300. ISBN: 978-3-540-74764-2.

Sferrazza, Carmelo et al. (2024). *HumanoidBench: Simulated Humanoid Benchmark for Whole-Body Locomotion and Manipulation*. arXiv: 2403.10506 [cs.RO]. URL: https://arxiv.org/abs/2403.10506.

Shafiee, Milad, Guillaume Bellegarda, and Auke Ijspeert (2024). "ManyQuadrupeds: Learning a Single Locomotion Policy for Diverse Quadruped Robots". In: *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 3471–3477. DOI: 10.1109/ICRA57147.2024.10610155.

Soest, A. J. van and M. F. Bobbert (1993). "The contribution of muscle properties in the control of explosive movements". In: *Biol Cybern* 69.3, pp. 195–204.

Song, Seungmoon and Hartmut Geyer (July 2013). "Generalization of a muscle-reflex control model to 3D walking". In: *2013 35th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*. Osaka: IEEE.

– (2017). "Evaluation of a Neuromechanical Walking Control Model Using Disturbance Experiments". In: *Frontiers in Computational Neuroscience* 11.3. DOI: 10.3389/fncom.2017.00015.

Song, Seungmoon et al. (Aug. 2021). "Deep reinforcement learning for modeling human locomotion control in neuromechanical simulation". In: *Journal of Neuro-Engineering and Rehabilitation* 18.1. DOI: 10.1186/s12984-021-00919-y.

Stollenmaier, Katrin, Winfried Ilg, and Daniel F. B. Haeufle (Apr. 2020). "Predicting Perturbed Human Arm Movements in a Neuro-Musculoskeletal Model to Investigate the Muscular Force Response". In: *Frontiers in Bioengineering and Biotechnology* 8.308. DOI: 10.3389/fbioe.2020.00308.

Sutton, Richard S. and Andrew G. Barto (2018). *Reinforcement Learning: An Introduction*. Cambridge, MA, USA: A Bradford Book. ISBN: 0262039249.

Takagi, A. et al. (2022). "A model predictive control strategy to regulate movements and interactions". In: *bioRxiv*. DOI: 10.1101/2022.08.24.505193. eprint: https://www.biorxiv.org/content/early/2022/08/26/2022.08.24.505193.full.pdf. URL: https://www.biorxiv.org/content/early/2022/08/26/2022.08.24.505193.

Tan, Jie, Karen Liu, and Greg Turk (2011). "Stable proportional-derivative controllers". In: *IEEE Computer Graphics and Applications* 31.4, pp. 34–44.

Thelen, Darryl G (Feb. 2003). "Adjustment of muscle mechanics model parameters to simulate dynamic contractions in older adults". en. In: *J. Biomech. Eng.* 125.1, pp. 70–77.

Tieck, Juan Camilo Vasquez et al. (2018). "Learning Continuous Muscle Control for a Multi-joint Arm by Extending Proximal Policy Optimization with a Liquid State Machine". In: *Artificial Neural Networks and Machine Learning – ICANN 2018*. Ed. by Věra Kůrková et al. Cham: Springer International Publishing, pp. 211–221. ISBN: 978-3-030-01418-6.

Tirinzoni, Andrea et al. (2024). "Zero-shot Whole-Body Humanoid Control via Behavioral Foundation Models". In.

Todorov, Emanuel, Tom Erez, and Yuval Tassa (2012). "MuJoCo: A physics engine for model-based control". In: *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 5026–5033. DOI: 10.1109/IROS.2012.6386109.

Tondu, B. and S.D. Zagal (2006). "McKibben artificial muscle can be in accordance with the Hill skeletal muscle model". In: *The First IEEE/RAS-EMBS International Conference on Biomedical Robotics and Biomechatronics, 2006. BioRob 2006.* Pp. 714–720. DOI: 10.1109/BIOROB.2006.1639174.

Tri, Vo Minh et al. (2010). "Characterization of Hysteresis in a Pneumatic Muscle Manipulator with Accounting for the Creep Effect". In: *IFAC Proceedings Volumes* 43.21. 4th IFAC Symposium on System Structure and Control, pp. 296–302. ISSN: 1474-6670. DOI: https://doi.org/10.3182/20100915-3-IT-2017.00041. URL: https://www.sciencedirect.com/science/article/pii/S1474667015384093.

Tschuggnall, Michael et al. (2021). "Machine learning approaches to predict rehabilitation success based on clinical and patient-reported outcome measures". In: *Informatics in Medicine Unlocked* 24, p. 100598. ISSN: 2352-9148. DOI: https://doi.org/10.1016/j.imu.2021.100598. URL: https://www.sciencedirect.com/science/article/pii/S2352914821000885.

Uchida, Thomas K et al. (Mar. 2016). "Stretching your energetic budget: How tendon compliance affects the metabolic cost of running". en. In: *PLoS One* 11.3, e0150378.

Uhlrich, Scott D. et al. (2023). "Ten steps to becoming a musculoskeletal simulation expert: A half-century of progress and outlook for the future". In: *Journal of Biomechanics* 154, p. 111623. ISSN: 0021-9290. DOI: https://doi.org/10.1016/j.jbiomech.2023.111623. URL: https://www.sciencedirect.com/science/article/pii/S0021929023001926.

Umberger, Brian R, Karin G M Gerritsen, and Philip E Martin (Apr. 2003). "A model of human muscle energy expenditure". en. In: *Comput. Methods Biomech. Biomed. Engin.* 6.2, pp. 99–111.

Vassiliadis, Pierre et al. (July 2021). "Reward boosts reinforcement-based motor learning". en. In: *iScience* 24.7, p. 102821.

Veerkamp, K. et al. (June 2021). "Evaluating cost function criteria in predicting healthy gait". In: *Journal of Biomechanics* 123, p. 110530. DOI: 10.1016/j.jbiomech.2021.110530.

Volchko, Angella et al. (Apr. 2022). "Model-Based Data-Driven System Identification and Controller Synthesis Framework for Precise Control of SISO and MISO HASEL-Powered Robotic Systems". In: *2022 IEEE 5th International Conference on Soft Robotics (RoboSoft)*, pp. 209–216. DOI: 10.1109/RoboSoft54090.2022.9762220.

Wagner, Heiko and Reinhard Blickhan (1999). "Stabilizing function of skeletal muscles: an analytical investigation". In: *Journal of Theoretical Biology* 199.2, pp. 163–179.

Wang, Tsun-Hsuan et al. (2023). "DiffuseBot: Breeding Soft Robots With Physics-Augmented Generative Diffusion Models". In: *Thirty-seventh Conference on Neural Information Processing Systems*. URL: https://openreview.net/forum?id=1zo4iioUEs.

Weng, Jiacheng, Ehsan Hashemi, and Arash Arami (2021a). "Natural Walking With Musculoskeletal Models Using Deep Reinforcement Learning". In: *IEEE Robotics and Automation Letters* 6.2, pp. 4156–4162. DOI: 10.1109/LRA.2021.3067617.

– (2021b). "Natural Walking With Musculoskeletal Models Using Deep Reinforcement Learning". In: *IEEE Robotics and Automation Letters* 6.2, pp. 4156–4162. DOI: 10.1109/LRA.2021.3067617.

– (2022). "Adaptive Reference Inverse Optimal Control for Natural Walking With Musculoskeletal Models". In: *IEEE Transactions on Neural Systems and Rehabilitation Engineering* 30, pp. 1567–1575. DOI: 10.1109/TNSRE.2022.3180690.

Winters, Jack M. (July 1995). "An improved muscle-reflex actuator for use in large-scale neuromusculoskeletal models". In: *Annals of Biomedical Engineering* 23.4, pp. 359–374. ISSN: 1573-9686. DOI: 10.1007/BF02584437. URL: https://doi.org/10.1007/BF02584437.

Wochner, I. et al. (Dec. 2022). "Learning with Muscles: Benefits for Data-Efficiency and Robustness in Anthropomorphic Tasks". In: *Proceedings of the 6th Conference on Robot Learning (CoRL)*. Vol. 205. Proceedings of Machine Learning Research. PMLR, pp. 1178–1188. URL: https://proceedings.mlr.press/v205/wochner23a.html.

Wochner, Isabell et al. (2020). "Optimality principles in human point-to-manifold reaching accounting for muscle dynamics". In: *Frontiers in Computational Neuroscience* 14, p. 38.

Yasuhiro Sugimoto Shinya Aoi, Naomichi Ogihara and Kazuo Tsuchiya (2009). "Stabilizing Function of the Musculoskeletal System for Periodic Motion". In: *Advanced Robotics* 23.5, pp. 521–534. DOI: 10.1163/156855309X420075. eprint: https://doi.org/10.1163/156855309X420075. URL: https://doi.org/10.1163/156855309X420075.

Yeo, Sang-Hoon et al. (2023). "Numerical instability of Hill-type muscle models". In: *Journal of The Royal Society Interface* 20.199, p. 20220430. DOI: 10.1098/rsif.2022.0430. eprint: https://royalsocietypublishing.org/doi/pdf/10.1098/rsif.2022.0430. URL: https://royalsocietypublishing.org/doi/abs/10.1098/rsif.2022.0430.

Zajac, F E (1989). "Muscle and tendon: properties, models, scaling, and application to biomechanics and motor control". en. In: *Crit. Rev. Biomed. Eng.* 17.4, pp. 359–411.

Zelik, Karl E et al. (Apr. 2014). "The role of series ankle elasticity in bipedal walking". en. In: *J. Theor. Biol.* 346, pp. 75–85.

Zhao, Wenshuai, Jorge Peña Queralta, and Tomi Westerlund (2020). "Sim-to-Real Transfer in Deep Reinforcement Learning for Robotics: a Survey". In: *2020 IEEE Symposium Series on Computational Intelligence (SSCI)*, pp. 737–744. DOI: 10.1109/SSCI47803.2020.9308468.

Ziegler, J. G. and N. B. Nichols (Dec. 2022). "Optimum Settings for Automatic Controllers". In: *Transactions of the American Society of Mechanical Engineers* 64.8, pp. 759–765. ISSN: 0097-6822. DOI: 10.1115/1.4019264. eprint: https://asmedigitalcollection.asme.org/fluidsengineering/article-pdf/64/8/759/6967291/759_1.pdf. URL: https://doi.org/10.1115/1.4019264.