GAIT Project $n + 1$

Lionel Blondé

September 18, 2024

Contents

1 Introduction

The objective of this document is to lay out the various options in have in addressing what we set out to tackle in this $n + 1$ iteration of the GAIT project.

To have a detailed account of *what* the project sets out to tackle, please refer to the proposal, which goes into the motivations and plans in depth. In here, we lay out the variety of options we have to address the raised research questions. This exhibition entails general approaches and methods, but also elaborates on the technical solutions.

The objective is to have a clear picture not only of the research landscape in the area but also to surface the conceptual and technical challenges that are inherent to the

area. While the *area* in question might have been phrased as "gait modeling" in previous iterations, in this $n+1$ iteration, we formulate is as *learning locomotion controllers*.

The task of learning how a control policy that knows how to walk *when embodied physically in the real world* is challenging for a multiplicity of reasons [RXZ+24, CICK+23, YDG+23, KBL+23, SKL22].

We do not aim for embodied deployment. What we want however is to learn controllers that are able to solve complex locomotion tasks *in silico*, *i.e.* in simulation. While we wish to make use of such learned behaviors to derive insights and guide decisions in real world tasks (such as surgical assistance), we do not aim to embody these in live robots.

An important point to make however is that most if not all the works in [RXZ+24, CICK+23, YDG+23, KBL+23, SKL22] make use of a simulated world in order to devise the controllers that they deploy in the real world. In that sense, the research progress carried out during this project could therefore be transferable and readily extended to research endeavors targeted at embodying gait controllers in hardware. The difficulty in the deploying such behavior is in closing the gap between the simulated and real world: the "sim-to-real" gap. Despite not having to deal with this gap due to the lack of deployment need per se, we face the gap when we have to deal with patient data. Indeed, data recorded by physicians (*e.g.* motion capture markers) do not live in the same space as the locomotion simulators at our disposal. Using such data to learn controllers in simulation required an alignment system that either *(i)* embeds the markers in the simulation space, or *(ii)* morphs the simulation space to the world the markers live in. The latter corresponds to the real world. Going for option *(ii)* was the goal of iteration n of the GAIT project. This alignment of the simulator with the patient's morphology was lead by the BIOROB team at EPFL. The actual technical solution changed over the course of the project as team members were coming and going. In the end, the simulation suite that allowed for the flexibily needed to close the gap between patient data and *in silico* world was OpenSim [SSRD11]. It stands as a viable option.

Going back a few steps, consider the task of learning a locomotion controller in simulation. In broad lines, the task requires the researcher practitioner to *(a)* choose or design a *locomotion simulator (b)* choose or design a *learning signal (c)* choose or design a *learning approach (d)* choose or design a *learning algorithm* These axes or choice or design are themselves multi-faceted, and this document will attempt to give the account of the options we have for each. Note, "choose" refers to the selection and use of an already existing solution, while "design" refers to the deliberate creation of a new solution. As it stands, such a system is therefore highly modular. Progress can be made on any combination of the four axes above. How these axes are formulated is flawed in the sense that these axes are not independent from each other. For example, the approach chosen determines to an extent the classes of algorithms, or trait thereof, that one might opt for. Still, it provides a salient language for us to navigate how to tackle the many-sided task.

Due to the interactions between these axes, the remainder will not catalog the four axes one by one, giving suggestions for each independently. Instead, we will consider the various learning approaches that can be considered to solve the task, and expand from there as we go with options. Organizing the exhibition as such hopefully gives a better account of those options compared to a catalog with bi-directional links all over the place.

2 Landscape

By *simulator*, we refer to a piece of software that our learning agent interact with in the course of its learning process. A simulator is what facilitates a *simulation*, which Sheldon M. Ross defines in [Ros22] as the imitation of the operation of a real-world process or system over time. In our case, the process we are interested in is the human gait. Formally, the simulator is modelled as a Markov Decision Process or MDP [Put94]. The latter consists of a tuple containing a state space (the sitations the agent finds itself in), an action space (the decisions the agent might make), a dynamics model that governs the physics of the world in which the agent interacts, and a reward process. MDPs are used in the formalism of problems defined under the reward hypothesis, which sees each task as the maximization of a signal called reward [BMAD22, SSPS21]. Under this umbrella of problem formulations lies reinforcement learning, *abbrv.* RL [SB98]. It is used to formalize problems of sequential decision making under uncertainty, and fits the problem of locomotion control well if we see the gait problem as consisting in making a series of reactive decisions consecutively in the interactive environment that is our simulator. We see immediately that the RL agent is limited by the speed at which the simulator responds, which is why this aspect is so critical. The choice of simulator will depend on the trade-off we want to strike. Be it a prioritization of fidelity or of speed or execution, this choice does not change the task. For the task to be fully defined however, the agent needs a reward or cost response upon interaction with the system. This value should serve as guidance, informing the agent seeking to maximize its cumulative reward —or equivalently minimize its cumulative cost— about how well it is progressing toward the completion of the task. This signal is designed [SLB09, RA98, Rus98, Sim94], and can very easily misshapen, leading to unforeseen effects or even exploits [AC16, EKO$^+$17, PBS22, LKS$^+$22, SHKK22, Sim94] It is natural to think about energy parsimony when it comes to the primary signal optimized by human when walking or running. This has been proved against in the case of horses in the past however [HT81, FT91]. [1] Nevertheless, it has recently been found that minimizing energy consumption leads to the emergence of gaits in legged robots [FKMP21]. A very recent study documents that how agents can overfit to certain reward designs, and emphasizes that one should construct a reward while keeping in mind that it is the reward accumulation that we want the agent to optimize toward [BKS$^+$23].

Simulators designed for control and locomotion are in general released with pseudo-benchmarked reward function. These are the result of tedious hand-engineering, often tainted by the issues raised and studied in [BKS$^+$23]. They are usually aligned with the forward progress of the agent in the sagittal plane [SLM$^+$15]. Another way to design a reward that would incentivize the learning agent to adopt the desired behavior is to leverage the availability of an expert at the task. This expert could be human or artificial, *e.g.* a hard-coded controller. This setting is called imitation learning, or learning from demonstration [BCDS08, AS97, Bag15]. Assuming alignment of expert and agent gaits, a sensible metric to use as cost could be the average displacement between the reference expert gait. Such a cost would be zero when the agent perfectly overlaps with the expert

[1] Particularly interesting quote from [Sim94]: "It is important that the physical simulation be reasonably accurate when optimizing for creatures that can move within it. Any bugs that allow energy leaks from non-conservation, or even round-off errors, will inevitably be discovered and exploited by the evolving creatures. Although this can be a lazy and often amusing approach for debugging a physical modeling system, it is not necessarily the most practical."

behavior. The alignment assumption however becomes however increasingly in valid as we stray from purely synthetic scenarios.

The locomotion controller we want to learn, denoted as π, maps the state space $\mathcal{S}$ onto the action space $\mathcal{A}$. If demonstrations are state-action pairs, the problem of learning such a π reduces to solving a supervised learning problem (dubbed *behavioral cloning*).

Note, learning a model with behavioral cloning (BC) [Pom89, Pom90, RBS07] can be done *offline*, *i.e.* without interacting with the simulator —characterizing *online* systems. BC is flawed in more than one way [RB10, RGB11]. To learn a viable policy, it requires diverse and action-annotated demonstrations in abundance. Since this requirement is seldom met, practitioners often turn to apprenticeship learning (AL), working in lower data regimes, and at least equally importantly, not requiring the expert data to be annotated with controls (actions). Indeed, the controls are rarely readily available (*e.g.* video demonstrations). AL [AN04, RBZ06, KAN08] is traditionally consisting of two stages: *(i)* inverse reinforcement learning (IRL; to derive a cost from the expert behavior), and *(ii)* reinforcement learning (RL; to derive a behavior from the cost being learned). These stages are sometimes done once, one after the other, or in alternation throughout the entire learning process [HGE16].

Trying to recover the *real* reward (the one optimized by the expert and that led to its behavior) has been shown to be an ill-posed problem [ZMBD08]. A relaxation of AL consisting in learning only a *surrogate* cost has proved enough, and has led to state-of-the-art results with generative adversarial imitation learning (GAIL) [HE16]. Being a relaxation of GAIL, it can learned from control-annotated demonstrations, or from observations (states) alone, without actions from the expert. The first setting is called "ILfD", the second "ILfO", for "Imitation Learning from Observations" [MTD+17, LGAL17, RAW+18, FLL18, APB+18, TWS18a, TWS18b]. Adversarial imitation learning is the go-to way to learn a sensible cost (reward) signal, but what about step *(ii)* above —the RL method to use in order to learn a controller from the surrogate reward?

Prime candidates to solve the RL outer loop and that have proved successful in various scenarios including locomotion tasks are *trust-region methods*. The figure-heads of this family of algorithms are TRPO [SLM+15], PPO [SWD+17], MPO [AST+18], V-MPO [FSAS+20], and MDPO [TSEG20].

The latter, which stands for "Mirror Descent Policy Optimization" has had its first surge of use recently in an LLM context [GWW+24]. MDPO is, according to the original paper, simpler and more stable than PPO, which is infamously tedious to tame [EIS+19, HDR+22, MMPG24]. MDPO has been introduced in two variants, one stems from PPO, the other from SAC [HZAL18, HZH+18], a method that falls under an architecture called *actor-critic* [CB95, KT00, SSM01]. While PPO is *on-policy*, SAC (which stands for "Soft Actor-Critic"), is *off-policy*. A method is said to be on-policy when the policy being learned and the one being used are one and the same. When they are distinct, the method is off-policy (and the two policies are called the *behavior* and the *target* policies). We will shortly tackle why off-policy-ness is of great interest when dealing with real-world RL in scenarios where safety is critical and where data scarcity is being faced (*cf.* 3.1).

If one has access to demonstrations in abundance *and* organized in complete sequences (*i.e.* not sub-sampled) then one can turn to *sequence modeling* (*cf.* [GBC16], Chap. 10). Tackling control problem as sequence prediction is not new, but is disregards the fact that control agents, as interactive in nature, must comply with the physics of their environment [SICP20]. Typical technical solutions have involved RNN/LSTM backbones for their ability to propagate information in time across timesteps. It does so by compressing

previous information seen in the trajectory (accumulating over time) into a fixed size hidden state.

Mainstream attention has moved onto a different architecture called the Transformer [VSP+17], which splits sequences in tokens from a fixed vocabulary, and learning how to build sequences by trying to predict the next token given all the previous ones in the sequence. The core mechanism at play is *attention*, and specifically *causal attention*, where the future is masked: the model can not attend to future token in its prediction. This ensures temporal consistency. The two concurrent works that first tackled control problems *(i)* as sequence modeling problems, *(ii)* with transformers, are [CWYA21] and [JLL21]. An vast array of works has spawned from these. Still, they are inherently flawed because they are not trained online, but from an offline dataset. Despite this flaw, they have proved very effective for real-world humanoid locomotion tasks [RZS+24] and real-world robotics tasks [BBC+22, BBC+23]. The obvious caveat is that it is a challenging engineering endeavor. Hybrid solutions have led to the best results so far. These leverage the power of transformers as feature extractors from sequences, but the task is of the RL type, *i.e.* maximize the cumulative reward, or IL type. Humanoid locomotion via RL is the goal of [RXZ+24], with broader robotics tasks via RL being the focus in the Q-Transformer [CVI+23], and via BC (IL) in Behavior Transformer (BeT) [SCAP22]. Diffusion heads have also been used on top of transformer backbones with great practical success, like in the Diffusion Policy [CFD+23]. Diffusion has also been successful for locomotion: Motion Diffusion Model (MDM) [TRG+23].

3 Desiderata

Ultimately what we want is to learn high-performance [2] *in silico* behavior in the shortest time possible, where the *time* is question encompasses *(i)* the wall-clock time it takes for the learning agent to solve the task, and *(ii)* the number of man-hours it takes for a given research idea to be validated or invalidated. Of course, those two are related: the shorter an experiment takes, the sooner the practitioner can iterate on the obtained results and run the next experiments with the adapted code. But those are two different axes on which the practitioner can make progress for the pipeline to be more efficient in time. To have the lowest time possible, our system should involve a simulator that is as fast as possible and a learning algorithm that requires the fewest possible number of interaction with the simulator to learn the optimal behavior for the task.

We will treat those in reverse order, first talking about the latter (a desideratum called *sample efficiency*) in 3.1, before dealing with the former (the speed of execution) in 3.2.

3.1 Sample Efficiency

Since simulators can be complex and therefore slow, it is of primary interest to treat with methods that require few interactions with the simulator, *i.e.* sample-efficient methods.

There are two main avenues for potential major improvements in sample efficiency: off-policy learning (described briefly earlier in 2), and model-based learning. Learning off-policy allows for substantial sample efficiency gains because it is compatible with experience replay. This mechanism consists in keeping a FIFO-queue buffer containing the N most recent transitions, and training the agent's policy (and value) on its contents.

[2] Note, the intricacies of the performance objective depend on the specific task formulation at hand.

It is *off*-policy because the sampling distribution corresponding to getting transitions uniformly from the replay buffer is a mixture of past policy updates, which is (in non-degenerate cases) different from the policy being learned. Those methods are usually less friendly, but the sample-efficiency gains are arguably well worth it in practical scenarios.

Model-based learning consists in learning a model of the environment in addition to learning a policy (and value). It can be abbreviated MBRL for "model-based RL". What is modeled has traditionally been the elements of the MDP, *i.e.* the forward dynamics $f : (s_t, a_t) \mapsto s_{t+1}$, the reward process $r : (s_t, a_t) \mapsto r_t$, and the termination flag (whether the episode is over or not) $d : (s_t, a_t) \mapsto d_t$. This is because such a model can then act as a substitute for the real environment. In that case, the interaction needs are lowered, and the method is more sample efficient (technically). This category of MBRL approaches (replacing the environment with a learned one) was introduced by Sutton with Dyna [Sut91]. As such, we will use the adjective "Dyna-style" to refer to this family of methods. MVE [FWS+18] extends Dyna, and STEVE [BHT+18] extends MVE. In the same vein, MBPO [JFZL19] is closely related to MVE and STEVE.

In addition to Dyna-style methods, there are "SVG-style" methods, where SVG refers to the "Stochastic Value Gradients" work [HWS+15]. SVG builds on "Value Gradients" [FA12], and has been revisited more recently in [ASYW21]. The origin of this approach could be traced back to the appearance of back-propagation-through-time (BPTT) [RHW86, RF87, NW90, Wer90, WP90, JR92, GTH98] It has then been continued in more modern setting by [Sch11, DR11], and very recently in the very successful DreamerV{1-3} methods [HLBN19, HLNB21, HPBL23, LDW+23]. These methods learn a model of the world with neural networks so that the learned approximation of the world can be part of the episode-spanning computational graph created by the RL objective. Effectively, the gradient propagate backward through the trajectories, from the termination to the initialization. For how these methods operate, the family is sometimes said to use "pathwise derivatives", to backpropagate through paths. While SVG uses only paths built from real samples (actually collected in rollouts in simulation), the Model-Augmented Actor-Critic (MAAC) [CFA20] uses both real rollouts and model rollouts (generated from the learned model of the environment). Another method from that family is SuperTrack [FBH21]. The policy is not optimized via RL however, but with IL.

The third approach to MBRL consists in learning a model of the world and deriving a policy by *planning* on the model with a method like MPC (model-predictive control). MPC is an online optimal control method [BH69, Ber00, Kir04]. By virtue of being online, it is adaptable by design. It might look closer to optimal control than RL but more modern MPC-style methods use a learned *termination value or critic* for the learned controller to always be able to gauge the long-term of its plans, as in TD-MPC2 [HSW24].

MPC is particularly appealing when the uncertainty of the model of the world is modeled, since the policy can then make good use of this information, as in PETS [CCML18] and POPLIN [WB20].

For the three ways of conducting MBRL (Dyna-style, SVG-style, MPC-style), it is common to learn models that do not follow the strict signatures of the MDP elements. Instead, a more complex model is often learned, the forward model is derived from it if needed. More complex models allow for more salient and holistic neural representations to be learned, leading to better values and controller down the line. They are called *world models*, which is an appropriately generic name considering how one might differ from one work to the next. Some references to world models include [HS18, HLF+18, HDV+21, MAF23, DZTZ24, AJM+24]. So:

Dyna-style The Dyna-style methods learn a model of the world to use this as an approximation instead of interacting with the simulator, hence *technically* reducing how many times the agent must interact with the simulation to learn.

SVG-style The SVG-style methods learn a *neural* model of the world to use in the objective's computational graph and be part of the optimization problem directly, in the hope that leveraging world gradients will make the controller learn faster.

MPC-style The MPC-style methods learn a model of the world to plan on, and planning is generally less costly in data points than model-free RL methods to derive a policy.

All three methods have their respective merits. The most *neural land* of the three is the SVG-style family. It has had successes (see above) but mostly disappoints for the expected successes it should have based on how paradigm-shifting end-to-end neural pipelines have been in essentially all other sub-domains of machine learning. The culprit is the behavior of the gradients of the world models in control scenarios (let us call those *world gradients*). DreamerV{1-3} (SVG-style) has been extremely successful in *discrete* control, but TD-MPC2 (MPC-style) still yields better results in *continuous* control. This is likely due to the world gradients being tedious to deal with in continuous control tasks. Backpropagation of annoying gradients (too big or too small) has been a hot topic for as long as neural networks have existed —see previous references cited above on BPTT.

A way to potentially get better world gradient is to use a *differentiable simulator*. We will abbreviate those as *diffsims*. Note that in such a case, there is no world model *a priori*. Instead of designing a learning task for a world model to extract useful and salient information from the world by itself —learning just what it *needs* to solve the task, the world gradients the agent gets are derived from hard-coded operations encoded via deep learning frameworks. As such, diffsims are a way to benefit from the sample efficiency of SVG-style model-based methods without the need to learn anything else than the agent's policy (and value). They are not *learned from data* (note, *where* this data comes from is a hurdle of its own in world modeling) but are *crafted* by the seasoned practitioner who injects domain knowledge and well-vetted structural biases in it. The diffsim designer has knowledge about the domain and knows how the design choices impact the interaction time (direct consequence of the integration from dynamics to kinematics). The designer can strike the desired trade-offs between fidelity with the real physics and speed. More sophisticated models can lead to higher fidelity, but are more costly to integrate over.

Works that involve pathwise derivatives in a diffsim include SHAC [XMN+22], AHAC [GSX+24], DiffMimic [RYC+23], PODS [MPH+21], and ILD [CMX22]. While all these are optimized with an RL objective, ILD is optimized with an IL one. Also, PODS requires second order information (about the Hessian) from the diffsim.

Plus, diffsims are hard-coded and therefore less prone to *model hacking* (also refered to as *model exploitation* [RAB12] or *model bias* in PILCO [DR11]). This phenomenon is observed when the policy uses the model of the world in ways that were unintended by the practitioner. It *hacks* or exploits the model to solve the task it was asked to solve, but did it in unforeseen ways at test time. Model bias is mentioned in SVG [HWS+15], ME-TRPO [KCD+18], PIPPS [PRPD18], and the authors of MBPO [JFZL19] write that "errors in model can undermine the advantage from model-based data augmentation". This is typical when dealing with black boxes like neural nets. The phenomenon is close in spirit to reward hacking. Note, model bias is a hurdle for all three families of model-based methods. It is easier to correct in diffsims simply because, since the simulator is

hard-coded and readable, we can *know* what was exploited. This part of the pipeline is a white box. Model exploitation loopholes are especially critical to avoid in SVG-style methods since feedback propagate along trajectories and pollutes everytime upstream if erroneous. Also, locomotion simulators are tedious to turn into diffsims notably because of the *contacts*, which introduce discontinuities in the simulator's response, and therefore Dirac deltas in the diffsim gradients. Design choices with respect to contact design are therefore particularly critical. Harder contacts are better for fidelity and therefore closing the sim-to-real gap, but softer contacts (dampened, surfaces like ground being permeable) remove the risk of introducing spikes in world gradients. Another point of design: in order to back-propagate from the RL objective through the diffsim backward through long trajectories, the reward itself need be a differentiable signal with respect to the quantities of interest, which is not necessarily obvious. In the case of IL however, the displacement error need only the dynamics to be differentiable. Lastly, since diffsims are tedious to engineer, using them is often an exercise in patience.

Some relevant references on diffsims: [DHDW16, HLS+19, FSD+19, HAL+20, QLKL20, GHS+20, GHZ+20, MPH+21, HMC+21, CHR+21, WOL+21, QLKL21, HHD+21, HMN+21, GFP+21, DFFR+21, LHL+22, XMN+22, SSZT22, ALGS+22, CMX22, RYC+23, GSX+24]. These range from the introduction of new simulators, to analyses of gradients types (zeroth versus first order gradients), to new ways to leverage diffsims to a fuller extent.

3.2 Execution Speed

This section is meant to contain items justifying the need for high execution speed if one wants to close the sim-to-real gap in a reasonable amount of time (for RL standards).

In 2019, the OpenAI robotics team —now defunct— leveraged a techique that led them to train a dextrous robotic hand [3] to solve the Rubik's cube puzzle [Ope19]. The endeavor consisted in first *solving* the dextrous in-hand manipulation of the cube, which was reported in [OAB+18], before integrating an off-the-shelf Rubik's cube solver into the system (engineering feat on its own). In order to solve the dextrous in-hand manipulation task, the engineers at OpenAI used a technique called *domain randomization* (DR), originating in works such as [TFR+17, TZC+18]. The technique is akin to the data augmentations carried out in self-supervised learning but these augmentations are done at the rendering engine level. The engineers developed a system that allows fast and customizable rendering of robotics environments. They called the system the OpenAI Remote Rendering Backend (ORRB) [CWW19]. In short, the system allows for the randomization of various aspects of the rendered robotic hand and objects in simulation (variety of textures, colors, etc.). Domain randomization very useful to be able to communicate to the agent what is relevant and what is not. By seeing a variety of textures and colors that do not change the task in nature, we want the model to understand what it should be *invariant* to. From this diversity comes robustness with respect to the unknowns of the real world.

So, if one want to close the gap, showing lots of things to the agent is crucial — *e.g.* via DR, and for that to be feasible, we need fast simulation engine backends.

Performant rendering engine have proved crucial since they allow techniques like DR. The video game industry has invested huge amounts to have performant rendering backends. It is therefore not surprising to witness the use of game engines to train agents via

[3] The "Shadow Dexterous Hand", from `https://www.shadowrobot.com/dexterous-hand-series/`

control approaches. The problem of solving locomotion problems in simulation with control algorithms has not only been tackled by the robotics or biorobotics crowds, but also by the *character animation* crowd, originiting in works such as [RH91, vdPF93, GTH98]. An account of "pre-2012" state-of-the-art works carried out in interactive character animation using simulated physics is given in [GP12]. Later works include: MACE [PBvdP16], DeepLoco [PBYvdP17], Phase-Functioned Neural Network (PFNN) [HKS17], DeepMimic [PALvdP18], UniCon [WGSF20], AMP [PMA+21], SuperTrack [FBH21] [4], DeepPhase [SMK22], AdaptNet [XXA+23]. The goal of character animation is for the rendered video to be believable enough for the viewer, and in some cases controllable — desirable trait for interactive mediums like video games. The goal is *not* to be as strictly faithful to the real physics of the world as possible. As such, the physics engine backends used in character animation are often simplifying how contacts are handled. Indeed, as described earlier, contacts introduce discontinuities in the simulator's response, which makes the integration of the dynamics into kinematics more numerically unstable. When a foot hits or leaves the ground, there is a sudden change in the system's dynamics, causing: *(a)* instability in numerical integration due to rapid force changes, *(b)* friction and slipping complications, which are hard to model accurately, *(c)* impulse forces during impact, requiring careful handling to avoid simulation errors, *(d)* constraint violations, such as feet penetrating the ground or sliding unexpectedly. Dampeners can solve contact problems in locomotion simulators by smoothing the force transitions during contact events. Engines usually leverage such techniques, unless fidelity with real world locomotion is paramount. MuJoCo [TET12] is often considered superior to many other simulation engines for handling contacts due to its advanced and efficient contact modeling. MuJoCo uses a form of *soft contact model* with damping to handle contacts Specifically, MuJoCo employs a compliant contact model [5]. where contacts are modeled using a combination of elastic (spring-like restoring force) and damping (energy-dissipation) forces.

The major hurdle of MuJoCo is that it was developed to be used on CPU hardware. But recent works, including RP1M [ZCS+24], have started using MuJoCo XLA (MJX) [6], which supports simulation in parallel with GPUs. While very high-fidelity biorobotics-inspired simulators such as OpenSim [SSRD11] and MyoSuite [CWD+22] will likely not be ported to GPU, chances are the control community will make the switch from the original MuJoCo backends (CPU) to MJX (GPU). This transition allows for the training of agents in a massively parallel fashion. Nvidia's Isaac Gym [MWG+21] did report tremendous speed-ups due to the parallelization enabled by having the GPU hardware as the core design choice. Nvidia also developed `dflex` [7], a PyTorch-based physics diffsim, which they used in SHAC [XMN+22] and AHAC [GSX+24]. For reasons unknown, `dflex` is not maintained anymore since 2022, despite being in use in their 2024 paper (see appendix of the latter for details).

MJX was built by core contributors to both MuJoCo and Brax [DFFR+21] (diffsim), who will together continue to support both Brax and MJX —according to the official MJX documentation. So, the best candidate when it comes to diffsim seems to be Brax. The only caveat is that it requires the use of JAX deep learning framework. While there exists

[4] Note, SuperTrack is from a game engine studio, uses a world model, as well as pathwise derivatives.

[5] A *compliant contact model* is a method of simulating contacts between objects by allowing small deformations or penetrations at the contact point, instead of treating the objects as perfectly rigid. This approach uses spring-like forces and dampers to model the interaction between the contacting surfaces.

[6] `https://mujoco.readthedocs.io/en/stable/mjx.html`

[7] `https://github.com/NVlabs/DiffRL/tree/main/dflex`

an official PyTorch wrapper in Brax, it does not allow gradients to flow through, making it a non-differentiable GPU-based simulator. Note, Brax takes a fair amount of shortcuts to make its primitives differentiable (shared in the companion paper [DFFR+21]).

When it comes to *musculo-skeletal* models, `dflex` has one. Compared to the ones from OpenSim [SSRD11] and MyoSuite [CWD+22] which are CPU-based and therefore slow, `dflex`'s is GPU-based (fast) *and* differentiable. The major problem being that `dflex` is not maintained anymore. Note, a private conversation with one of the main authors of the AHAC paper revealed that the musculo-skeletal model they have is available from head-to-toe despite being only from hips-to-toe in the papers (SHAC and AHAC). The upper body has simply been commented out in the codebase. The author encouraged use to communicate with them about our project to that they can assist us in using their *next-gen* simulator engine, `warp` [8]. Nvidia left `dflex` development, and has yet to release its successor diffsim based on `warp`. This day might never come. Note, Isaac Gym development has also been abandonned, probably to promote the move to `warp`. To conclude, GPU-based simulators have allowed huge speed-ups in training and therefore development time. The main player in the field is Nvidia —not surprising considering their place in the GPU market, but now is an awkward time where they seem to be switching backend. Using MJX seems to be the smarted move support-wise. To conclude:

Both MJX and Brax (diffsim) are active and led by Google, but are written in JAX. Isaac Gym and `dflex` (diffsim) are inactive and led by Nvidia, but they are in PyTorch.

4 NeurIPS Competitions: Learning to Walk

4.1 From 2017 to 2024

4.1.1 NeurIPS 2017: Learning to Walk

- **Objective**: Train a bipedal agent to walk as far as possible in a simulated environment using reinforcement learning.
- **Challenges**: Realistic walking behaviors in a physics-based MuJoCo simulation.
- **Musculo-skeletal Models**: **No** – The simulation involved direct control of joints (torques) rather than muscles.
- **Outcome**: Highlighted the need for efficient policy learning in basic locomotion tasks.

4.1.2 NeurIPS 2018: AI for Prosthetics

- **Objective**: Control a prosthetic leg to enable walking for a human model with a simulated amputation using **OpenSim**.
- **Challenges**: Human-prosthetic interaction, simulating biomechanics of both the prosthetic and remaining leg.
- **Musculo-skeletal Models**: **Yes** – The simulation used musculo-skeletal models with muscle-based controls, making it more realistic in mimicking human gait.

[8] `https://github.com/NVIDIA/warp`

- **Outcome**: Showed the importance of biomechanical realism in prosthetic simulations.

4.1.3 NeurIPS 2019: Learn to Move – Walk Around

- **Objective**: Agents had to walk and run in a complex 3D environment using **OpenSim**, with adaptive movement across various terrains.
- **Challenges**: Terrain complexity, requiring stability and adaptability.
- **Musculo-skeletal Models**: **Yes** – OpenSim's musculo-skeletal models with muscle-based controls were used, making the agent control more human-like.
- **Outcome**: Advanced the ability to generalize movement behaviors across different environments with more complex biomechanical models.

4.1.4 NeurIPS 2020: Learn to Move – Adaptive Arm

- **Objective**: Focused on upper-limb control and object manipulation using a simulated arm.
- **Challenges**: Adapting to changing goals and constraints in fine motor control.
- **Musculo-skeletal Models**: **Yes** – The competition used musculo-skeletal modeling for the arm, controlling muscles rather than direct torques.
- **Outcome**: Expanded RL applications to include fine motor skills, emphasizing musculo-skeletal dynamics.

4.1.5 NeurIPS 2021: The Walkers

- **Objective**: Control four-legged walkers across challenging terrains.
- **Challenges**: Handling uneven terrain, slopes, and moving obstacles.
- **Musculo-skeletal Models**: **No** – The simulation used joint-based controls (torques) rather than muscle-based controls.
- **Outcome**: Focused on the adaptability of RL agents to dynamic terrains and obstacles without introducing musculo-skeletal complexity.

4.1.6 NeurIPS 2022: The Walkers – Humanoids

- **Objective**: Agents had to control humanoid walkers, adding complexity due to the high-dimensionality of the humanoid model.
- **Challenges**: Stability and balance of a humanoid across complex terrains.
- **Musculo-skeletal Models**: **No** – The controls were based on joint torques, not muscle-based controls.
- **Outcome**: Explored advanced techniques like *hierarchical RL* and *imitation learning* for humanoid locomotion, but without musculo-skeletal models.

4.1.7 NeurIPS 2023: Robust Locomotion

- **Objective**: Develop controllers for robust locomotion across various terrains and dynamic conditions.
- **Challenges**: Emphasis on robust performance in unseen environments.
- **Musculo-skeletal Models**: **No** – The simulation focused on controlling joint torques rather than muscles.
- **Outcome**: Leveraged *domain randomization* and *adversarial training* to improve the generalization of locomotion controllers, but without muscle-based models.

4.1.8 NeurIPS 2024: Physiological Dexterity and Agility in Bionic Humans

- **Objective**: This challenge, called the MyoChallenge[9], focuses on creating controllers for musculo-skeletal systems in bionic human models. The goal is to push the limits of physiological dexterity and agility by simulating fine motor control and robust, dynamic movements.
- **Challenges**:
  - Controlling complex musculo-skeletal systems, including simulating physiological responses.
  - Achieving robust dexterity and agility in both upper- and lower-limb tasks, with a focus on dynamic and adaptive behaviors.
- **Musculo-skeletal Models**: **Yes** – The challenge uses musculo-skeletal models, with muscle-based controls allowing for physiological accuracy in movement.
- **Outcome (Expected)**: This challenge will drive innovations in modeling human-like movement and control, as well as in developing advanced bionic systems that mimic human agility and dexterity.

4.2 Summary *vis-a-vis* Musculo-skeletal Models

- **Included in**: 2018, 2019, 2020 (focused on prosthetics, adaptive walking, and arm control, where simulating realistic muscle actions was crucial).
- **Excluded in**: 2017, 2021, 2022, 2023 (focused more on joint-based or torque control, particularly in robotic and humanoid walking challenges).

5 Concretely

I suggest the following ("H.", "J.", and "L." are Hugues, Joao, and Lionel respectively):

1. H. presents his bioscience background in front of the whole team.
2. H. presents a taylored presentation about biomechanical models in front of this project team only. The contents: Hill models, Raash models, Hunt-Crossley models, Featherstone models, generalized coordinates, etc.

[9] `https://sites.google.com/view/myosuite/myochallenge/myochallenge-2024`

3. H. goes through all the "Learning to Walk" competitions listed out in 4 and walk us through what the winning teams did to get the first place. Podium contestants with notable or out-of-the-ordinary approaches should be discussed well.

4. H. should get familiarize himself with running RL and IL experiments in the LocoMuJoCo [AHZPT23] suite. At the same time, if timeline forces it, L./J. will be conducting experiments to learn controllers in musculo-skeletal simulators. If J. is actively working of diffsims without muscles, then L. will work on the muscle-actuated simulations.

5. The role of L./J. is to assist H. as much as possible in being proficient in handling simulators and understanding RL/IL methods –this assistance is hands-on, since it will take time for H. to get accustomed to this new machinery. Still, someone at ease with Python and biomechanics will likely progress fast, especially with our continued assistance.

6. Concerning *which complex simulators*, we will use: *(a)* MyoLeg, a MuJoCo model, used via the MyoSuite Python API, and *(b)* Hyfydy models used via the SCONE Python API. Both are very fast, although not running on GPU.

7. Whether diffsim and/or GPU simulations will be used in this project depends entirely on the research interests and endeavors or H., J., and L. It is not really a item of primary interest for the project *per se*.

8. H. has some familiarity with PyTorch. L. and J. are proficienty in PyTorch. There have nevertheless evidence that JAX has been instrumental in real-world RL use *where PyTorch failed to deliver in terms of speed*, notatly in [SKL22]. L. will use JAX to revive a previous project that was suffering from long training times; maybe that might be also a good occasion for H. to learn Jax[10].

9. In parallel, L. will be in communication with Eric Heiden from the NVIDIA `warp` team to discuss various things related to their muscle-actuated locomotion diffsim.

6 Extras

A very candidate for locomotion tasks seems to be TD-MPC2 [HSW24] (the *why* was discussed earlier).

A very promising environment: LocoMuJoCo [AHZPT23], which has domain randomization capabilities.

There might be interesting considerations in "Temporal Cycle-Consistency Learning" (TCC) [DAT+19].

Some references of *motion capture* (MoCap): [MTD+17,MHG+19,MTA+20,LBPT+21, BTB+22, LLW+22, WWY22, SC23, RZS+24]

Comparison of the impact of four different action parameterizations (torques, muscle-activations, target joint angles, and target jointangle velocities) in terms of learning time, policy robustness, motion quality, and policy query rates: [PvdP16].

[10]The models and simulators previously mentioned (Hyfydy, MuJoCo, etc.) are outside the computational graph, and therefore agnostic *w.r.t.* deep learning automatic differentiation (AD) frameworks

Interesting tool to scale musculoskeletal models automatically, along with inverse kinematics and inverse dynamics (based on Nimble): [WBR+23]. The algorithm is published as an open source cloud service at `AddBiomechanics.org`. Also under the topic of *system/parameter identification*, [VPT+14] asks the question of whether patient-specific musculoskeletal models are robust to uncertainties in its parameters.

References

[AC16] Dario Amodei and Jack Clark. Faulty Reward Functions in the Wild. `https://openai.com/blog/faulty-reward-functions/`, December 2016. Accessed: –.

[AHZPT23] Firas Al-Hafez, Guoping Zhao, Jan Peters, and Davide Tateo. LocoMuJoCo: A comprehensive imitation Learning benchmark for locomotion. *arXiv [cs.LG]*, November 2023.

[AJM+24] Eloi Alonso, Adam Jelley, Vincent Micheli, Anssi Kanervisto, Amos Storkey, Tim Pearce, and François Fleuret. Diffusion for world modeling: Visual details matter in Atari. *arXiv [cs.LG]*, May 2024.

[ALGS+22] Kelsey R Allen, Tatiana Lopez-Guevara, Kimberly Stachenfeld, Alvaro Sanchez-Gonzalez, Peter Battaglia, Jessica Hamrick, and Tobias Pfaff. Physical Design using Differentiable Learned Simulators. *arXiv [cs.LG]*, February 2022.

[AN04] Pieter Abbeel and Andrew Y Ng. Apprenticeship Learning via Inverse Reinforcement Learning. In *International Conference on Machine Learning (ICML)*, 2004.

[APB+18] Yusuf Aytar, Tobias Pfaff, David Budden, Tom Le Paine, Ziyu Wang, and Nando de Freitas. Playing hard exploration games by watching YouTube. In *Neural Information Processing Systems (NeurIPS)*, 2018.

[AS97] Christopher G Atkeson and Stefan Schaal. Robot Learning From Demonstration. In *International Conference on Machine Learning (ICML)*, volume 97, pages 12–20, 1997.

[AST+18] 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 (ICLR)*, 2018.

[ASYW21] Brandon Amos, Samuel Stanton, Denis Yarats, and Andrew Gordon Wilson. On the Model-Based Stochastic Value Gradient for Continuous Reinforcement Learning. In *Learning for Dynamics and Control*, pages 6–20. PMLR, May 2021.

[Bag15] J Andrew Bagnell. An invitation to imitation. Technical report, Carnegie Mellon, Robotics Institute, Pittsburgh, 2015.

[BBC+22] Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Joseph Dabis, Chelsea Finn, Keerthana Gopalakrishnan, Karol Hausman, Alex Herzog, Jasmine Hsu, Julian Ibarz, Brian Ichter, Alex Irpan, Tomas Jackson, Sally Jesmonth, Nikhil J Joshi, Ryan Julian, Dmitry Kalashnikov, Yuheng Kuang, Isabel Leal, Kuang-Huei Lee, Sergey Levine, Yao Lu, Utsav Malla, Deeksha Manjunath, Igor Mordatch, Ofir Nachum, Carolina Parada, Jodilyn Peralta, Emily Perez, Karl Pertsch, Jornell Quiambao, Kanishka Rao, Michael Ryoo, Grecia Salazar, Pannag Sanketi, Kevin Sayed, Jaspiar Singh, Sumedh Sontakke, Austin Stone, Clayton Tan, Huong Tran, Vincent Vanhoucke, Steve Vega, Quan Vuong, Fei Xia, Ted Xiao, Peng Xu, Sichun Xu, Tianhe Yu, and Brianna Zitkovich. RT-1: Robotics Transformer for Real-World Control at Scale. *arXiv [cs.RO]*, December 2022.

[BBC+23] Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Xi Chen, Krzysztof Choromanski, Tianli Ding, Danny Driess, Avinava Dubey, Chelsea Finn, Pete Florence, Chuyuan Fu, Montse Gonzalez Arenas, Keerthana Gopalakrishnan, Kehang Han, Karol Hausman, Alexander Herzog, Jasmine Hsu, Brian Ichter, Alex Irpan, Nikhil Joshi, Ryan Julian, Dmitry Kalashnikov, Yuheng Kuang, Isabel Leal, Lisa Lee, Tsang-Wei Edward Lee, Sergey Levine, Yao Lu, Henryk Michalewski, Igor Mordatch, Karl Pertsch, Kanishka Rao, Krista Reymann, Michael Ryoo, Grecia Salazar, Pannag Sanketi, Pierre Sermanet, Jaspiar Singh, Anikait Singh, Radu Soricut, Huong Tran, Vincent Vanhoucke, Quan Vuong, Ayzaan Wahid, Stefan Welker, Paul Wohlhart, Jialin Wu, Fei Xia, Ted Xiao, Peng Xu, Sichun Xu, Tianhe Yu, and Brianna Zitkovich. RT-2: Vision-Language-Action Models Transfer Web Knowledge to Robotic Control. Technical report, DeepMind, July 2023.

[BCDS08] Aude Billard, Sylvain Calinon, Rüdiger Dillmann, and Stefan Schaal. Robot Programming by Demonstration. In Siciliano Bruno and Khatib Oussama, editors, *Springer Handbook of Robotics*, pages 1371–1394. Springer Berlin Heidelberg, 2008.

[Ber00] Dimitri P Bertsekas. *Dynamic Programming and Optimal Control*. Athena Scientific, 2000.

[BH69] Arthur Earl Bryson and Yu-Chi Ho. *Applied Optimal Control: Optimization, Estimation, and Control*. Blaisdell Publishing Company, 1969.

[BHT+18] Jacob Buckman, Danijar Hafner, George Tucker, Eugene Brevdo, and Honglak Lee. Sample-Efficient Reinforcement Learning with Stochastic Ensemble Value Expansion. In *Neural Information Processing Systems (NeurIPS)*, 2018.

[BKS+23] Serena Booth, W Bradley Knox, Julie Shah, Scott Niekum, Peter Stone, and Alessandro Allievi. The perils of trial-and-error reward design: Misdesign through overfitting and invalid task specifications. In *Conference on Artificial Intelligence (AAAI)*, volume 37, pages 5920–5929. Association for the Advancement of Artificial Intelligence (AAAI), 2023.

[BMAD22] Michael Bowling, John D Martin, David Abel, and Will Dabney. Settling the Reward Hypothesis. Technical report, DeepMind, December 2022.

[BTB+22] Steven Bohez, Saran Tunyasuvunakool, Philemon Brakel, Fereshteh Sadeghi, Leonard Hasenclever, Yuval Tassa, Emilio Parisotto, Jan Humplik, Tuomas Haarnoja, Roland Hafner, Markus Wulfmeier, Michael Neunert, Ben Moran, Noah Siegel, Andrea Huber, Francesco Romano, Nathan Batchelor, Federico Casarini, Josh Merel, Raia Hadsell, and Nicolas Heess. Imitate and Repurpose: Learning Reusable Robot Movement Skills From Human and Animal Behaviors. Technical report, DeepMind, March 2022.

[CB95] Robert H Crites and Andrew G Barto. An Actor/Critic Algorithm that Equivalent to Q-Learning. In *Neural Information Processing Systems (NeurIPS)*, 1995.

[CCML18] Kurtland Chua, Roberto Calandra, Rowan McAllister, and Sergey Levine. Deep Reinforcement Learning in a Handful of Trials using Probabilistic Dynamics Models. In *Neural Information Processing Systems (NeurIPS)*, 2018.

[CFA20] Ignasi Clavera, Violet Fu, and Pieter Abbeel. Model-Augmented Actor-Critic: Backpropagating through Paths. In *International Conference on Learning Representations (ICLR)*, 2020.

[CFD+23] Cheng Chi, Siyuan Feng, Yilun Du, Zhenjia Xu, Eric Cousineau, Benjamin Burchfiel, and Shuran Song. Diffusion Policy: Visuomotor Policy Learning via Action Diffusion. *arXiv [cs.RO]*, March 2023.

[CHR+21] Samuel Clarke, Negin Heravi, Mark Rau, Ruohan Gao, Jiajun Wu, Doug James, and Jeannette Bohg. DiffImpact: Differentiable Rendering and Identification of Impact Sounds. In *Conference on Robot Learning (CoRL)*, 2021.

[CICK+23] Ken Caluwaerts, Atil Iscen, J Chase Kew, Wenhao Yu, Tingnan Zhang, Daniel Freeman, Kuang-Huei Lee, Lisa Lee, Stefano Saliceti, Vincent Zhuang, Nathan Batchelor, Steven Bohez, Federico Casarini, Jose Enrique Chen, Omar Cortes, Erwin Coumans, Adil Dostmohamed, Gabriel Dulac-Arnold, Alejandro Escontrela, Erik Frey, Roland Hafner, Deepali Jain, Bauyrjan Jyenis, Yuheng Kuang, Edward Lee, Linda Luu, Ofir Nachum, Ken Oslund, Jason Powell, Diego Reyes, Francesco Romano, Feresteh Sadeghi, Ron Sloat, Baruch Tabanpour, Daniel Zheng, Michael Neunert, Raia Hadsell, Nicolas Heess, Francesco Nori, Jeff Seto, Carolina Parada, Vikas Sindhwani, Vincent Vanhoucke, and Jie Tan. Barkour: Benchmarking Animal-level Agility with Quadruped Robots. Technical report, DeepMind, May 2023.

[CMX22] Siwei Chen, Xiao Ma, and Zhongwen Xu. Imitation Learning via Differentiable Physics. *arXiv [cs.LG]*, June 2022.

[CVI+23] Yevgen Chebotar, Quan Vuong, Alex Irpan, Karol Hausman, Fei Xia, Yao Lu, Aviral Kumar, Tianhe Yu, Alexander Herzog, Karl Pertsch, Keerthana

Gopalakrishnan, Julian Ibarz, Ofir Nachum, Sumedh Sontakke, Grecia Salazar, Huong T Tran, Jodilyn Peralta, Clayton Tan, Deeksha Manjunath, Jaspiar Singht, Brianna Zitkovich, Tomas Jackson, Kanishka Rao, Chelsea Finn, and Sergey Levine. Q-Transformer: Scalable Offline Reinforcement Learning via Autoregressive Q-Functions. In *Conference on Robot Learning (CoRL)*, 2023.

[CWD+22] Vittorio Caggiano, Huawei Wang, Guillaume Durandau, Massimo Sartori, and Vikash Kumar. MyoSuite: A Contact-rich Simulation Suite for Musculoskeletal Motor Control. In Roya Firoozi, Negar Mehr, Esen Yel, Rika Antonova, Jeannette Bohg, Mac Schwager, and Mykel Kochenderfer, editors, *Annual Conference on Learning for Dynamics and Control (L4DC)*, volume 168 of *Proceedings of Machine Learning Research*, pages 492–507. PMLR, 2022.

[CWW19] Maciek Chociej, Peter Welinder, and Lilian Weng. ORRB - OpenAI Remote Rendering Backend. Technical report, OpenAI, June 2019.

[CWYA21] Chang Chen, Yi-Fu Wu, Jaesik Yoon, and Sungjin Ahn. TransDreamer: Reinforcement Learning with Transformer World Models. In *NeurIPS Workshop "Deep Reinforcement Learning"*, 2021.

[DAT+19] Debidatta Dwibedi, Yusuf Aytar, Jonathan Tompson, Pierre Sermanet, and Andrew Zisserman. Temporal Cycle-Consistency Learning. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[DFFR+21] C Daniel Freeman, Erik Frey, Anton Raichuk, Sertan Girgin, Igor Mordatch, and Olivier Bachem. Brax – A Differentiable Physics Engine for Large Scale Rigid Body Simulation. *arXiv [cs.RO]*, June 2021.

[DHDW16] Jonas Degrave, Michiel Hermans, Joni Dambre, and Francis Wyffels. A Differentiable Physics Engine for Deep Learning in Robotics. *arXiv [cs.NE]*, November 2016.

[DR11] Marc Deisenroth and Carl E Rasmussen. PILCO: A model-based and data-efficient approach to policy search. In *International Conference on Machine Learning (ICML)*, 2011.

[DZTZ24] Zihan Ding, Amy Zhang, Yuandong Tian, and Qinqing Zheng. Diffusion World Model. *arXiv [cs.LG]*, February 2024.

[EIS+19] Logan Engstrom, Andrew Ilyas, Shibani Santurkar, Dimitris Tsipras, Firdaus Janoos, Larry Rudolph, and Aleksander Madry. Implementation Matters in Deep Policy Gradients: A Case Study on PPO and TRPO. In *International Conference on Learning Representations (ICLR)*, 2019.

[EKO+17] Tom Everitt, Victoria Krakovna, Laurent Orseau, Marcus Hutter, and Shane Legg. Reinforcement Learning with a Corrupted Reward Channel. In *International Joint Conference on Artificial Intelligence (IJCAI)*, 2017.

[FA12] Michael Fairbank and Eduardo Alonso. Value-Gradient Learning. In *International Joint Conference on Neural Networks (IJCNN)*, pages 1–8, June 2012.

[FBH21] Levi Fussell, Kevin Bergamin, and Daniel Holden. SuperTrack: Motion Tracking for Physically Simulated Characters using Supervised Learning. *ACM Trans. Graph.*, 40(6):1–13, December 2021.

[FKMP21] Zipeng Fu, Ashish Kumar, Jitendra Malik, and Deepak Pathak. Minimizing Energy Consumption Leads to the Emergence of Gaits in Legged Robots. In *Conference on Robot Learning (CoRL)*, 2021.

[FLL18] Justin Fu, Katie Luo, and Sergey Levine. Learning Robust Rewards with Adversarial Inverse Reinforcement Learning. In *International Conference on Learning Representations (ICLR)*, 2018.

[FSAS+20] H Francis Song, Abbas Abdolmaleki, Jost Tobias Springenberg, Aidan Clark, Hubert Soyer, Jack W Rae, Seb Noury, Arun Ahuja, Siqi Liu, Dhruva Tirumala, Nicolas Heess, Dan Belov, Martin Riedmiller, and Matthew M Botvinick. V-MPO: On-Policy Maximum a Posteriori Policy Optimization for Discrete and Continuous Control. In *International Conference on Learning Representations (ICLR)*, 2020.

[FSD+19] Antoine Falisse, Gil Serrancolí, Christopher L Dembia, Joris Gillis, and Friedl De Groote. Algorithmic differentiation improves the computational efficiency of OpenSim-based trajectory optimization of human movement. *PLoS ONE*, 14(10):e0217730, October 2019.

[FT91] C T Farley and C R Taylor. A mechanical trigger for the trot-gallop transition in horses. *Science*, 253(5017):306–308, July 1991.

[FWS+18] Vladimir Feinberg, Alvin Wan, Ion Stoica, Michael I Jordan, Joseph E Gonzalez, and Sergey Levine. Model-based value estimation for efficient model-free reinforcement learning. In *International Conference on Machine Learning (ICML)*, February 2018.

[GBC16] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016.

[GFP+21] Nathan Grinsztajn, Johan Ferret, Olivier Pietquin, Philippe Preux, and Matthieu Geist. There Is No Turning Back: A Self-Supervised Approach for Reversibility-Aware Reinforcement Learning. In *Neural Information Processing Systems (NeurIPS)*, 2021.

[GHS+20] Paula Gradu, John Hallman, Daniel Suo, Alex Yu, Naman Agarwal, Udaya Ghai, Karan Singh, Cyril Zhang, Anirudha Majumdar, and Elad Hazan. Deluca – A Differentiable Control Library: Environments, Methods, and Benchmarking. In *NeurIPS Workshop on Differentiable Vision, Graphics, and Physics in Machine Learning*, 2020.

[GHZ+20] Moritz Geilinger, David Hahn, Jonas Zehnder, Moritz Bächer, Bernhard Thomaszewski, and Stelian Coros. ADD: Analytically Differentiable Dynamics for Multi-Body Systems with Frictional Contact. *ACM Trans. Graph.*, 39(6):1–15, November 2020.

[GP12] T Geijtenbeek and N Pronost. Interactive Character Animation Using Simulated Physics: A State-of-the-Art Review. *Computer Graphics Forum*, 31(8):2492–2515, December 2012.

[GSX+24] Ignat Georgiev, Krishnan Srinivasan, Jie Xu, Eric Heiden, and Animesh Garg. Adaptive Horizon Actor-Critic for policy learning in contact-rich differentiable simulation. In *International Conference on Machine Learning (ICML)*, May 2024.

[GTH98] Radek Grzeszczuk, Demetri Terzopoulos, and Geoffrey Hinton. NeuroAnimator: Fast Neural Network Emulation and Control of Physics-Based Models. In *ACM Transactions on Graphics (SIGGRAPH)*, SIGGRAPH '98, pages 9–20, New York, NY, USA, July 1998. Association for Computing Machinery.

[GWW+24] Tom Gunter, Zirui Wang, Chong Wang, Ruoming Pang, Andy Narayanan, Aonan Zhang, Bowen Zhang, Chen Chen, Chung-Cheng Chiu, David Qiu, Deepak Gopinath, Dian Ang Yap, Dong Yin, Feng Nan, Floris Weers, Guoli Yin, Haoshuo Huang, Jianyu Wang, Jiarui Lu, John Peebles, Ke Ye, Mark Lee, Nan Du, Qibin Chen, Quentin Keunebroek, Sam Wiseman, Syd Evans, Tao Lei, Vivek Rathod, Xiang Kong, Xianzhi Du, Yanghao Li, Yongqiang Wang, Yuan Gao, Zaid Ahmed, Zhaoyang Xu, Zhiyun Lu, Al Rashid, Albin Madappally Jose, Alec Doane, Alfredo Bencomo, Allison Vanderby, Andrew Hansen, Ankur Jain, Anupama Mann Anupama, Areeba Kamal, Bugu Wu, Carolina Brum, Charlie Maalouf, Chinguun Erdenebileg, Chris Dulhanty, Dominik Moritz, Doug Kang, Eduardo Jimenez, Evan Ladd, Fangping Shi, Felix Bai, Frank Chu, Fred Hohman, Hadas Kotek, Hannah Gillis Coleman, Jane Li, Jeffrey Bigham, Jeffery Cao, Jeff Lai, Jessica Cheung, Jiulong Shan, Joe Zhou, John Li, Jun Qin, Karanjeet Singh, Karla Vega, Kelvin Zou, Laura Heckman, Lauren Gardiner, Margit Bowler, Maria Cordell, Meng Cao, Nicole Hay, Nilesh Shahdadpuri, Otto Godwin, Pranay Dighe, Pushyami Rachapudi, Ramsey Tantawi, Roman Frigg, Sam Davarnia, Sanskruti Shah, Saptarshi Guha, Sasha Sirovica, Shen Ma, Shuang Ma, Simon Wang, Sulgi Kim, Suma Jayaram, Vaishaal Shankar, Varsha Paidi, Vivek Kumar, Xin Wang, Xin Zheng, Walker Cheng, Yael Shrager, Yang Ye, Yasu Tanaka, Yihao Guo, Yunsong Meng, Zhao Tang Luo, Zhi Ouyang, Alp Aygar, Alvin Wan, Andrew Walkingshaw, Andy Narayanan, Antonie Lin, Arsalan Farooq, Brent Ramerth, Colorado Reed, Chris Bartels, Chris Chaney, David Riazati, Eric Liang Yang, Erin Feldman, Gabriel Hochstrasser, Guillaume Seguin, Irina Belousova, Joris Pelemans, Karen Yang, Keivan Alizadeh Vahid, Liangliang Cao, Mahyar Najibi, Marco Zuliani, Max Horton, Minsik Cho, Nikhil Bhendawade, Patrick Dong, Piotr Maj, Pulkit Agrawal, Qi Shan, Qichen Fu, Regan Poston, Sam Xu, Shuangning Liu, Sushma Rao, Tashweena Heeramun, Thomas Merth, Uday Rayala, Victor Cui, Vivek Rangarajan Sridhar, Wencong Zhang, Wenqi Zhang, Wentao Wu, Xingyu Zhou, Xinwen Liu, Yang Zhao, Yin Xia, Zhile Ren, and Zhongzheng Ren. Apple Intelligence foundation language models. *arXiv [cs.AI]*, July 2024.

[HAL+20] Yuanming Hu, Luke Anderson, Tzu-Mao Li, Qi Sun, Nathan Carr, Jonathan Ragan-Kelley, and Frédo Durand. DiffTaichi: Differentiable Programming for Physical Simulation. In *International Conference on Learning Representations (ICLR)*, 2020.

[HDR+22] Shengyi Huang, Rousslan Fernand Julien Dossa, Antonin Raffin, Anssi Kanervisto, and Weixun Wang. The 37 Implementation Details of Proximal Policy Optimization · The ICLR Blog Track. `https://iclr-blog-track.github.io/2022/03/25/ppo-implementation-details/`, March 2022. Accessed: 2023-8-21.

[HDV+21] Matteo Hessel, Ivo Danihelka, Fabio Viola, Arthur Guez, Simon Schmitt, Laurent Sifre, Theophane Weber, David Silver, and Hado van Hasselt. Muesli: Combining Improvements in Policy Optimization. In *International Conference on Machine Learning (ICML)*, 2021.

[HE16] Jonathan Ho and Stefano Ermon. Generative Adversarial Imitation Learning. In *Neural Information Processing Systems (NIPS)*, 2016.

[HGE16] Jonathan Ho, Jayesh K Gupta, and Stefano Ermon. Model-Free Imitation Learning with Policy Optimization. In *International Conference on Machine Learning (ICML)*, 2016.

[HHD+21] Zhiao Huang, Yuanming Hu, Tao Du, Siyuan Zhou, Hao Su, Joshua B Tenenbaum, and Chuang Gan. PlasticineLab: A Soft-Body Manipulation Benchmark with Differentiable Physics. In *International Conference on Learning Representations (ICLR)*, 2021.

[HKS17] Daniel Holden, Taku Komura, and Jun Saito. Phase-functioned neural networks for character control. *ACM Transactions on Graphics*, 36(4):42, July 2017.

[HLBN19] Danijar Hafner, Timothy Lillicrap, Jimmy Ba, and Mohammad Norouzi. Dream to Control: Learning Behaviors by Latent Imagination. *arXiv [cs.LG]*, December 2019.

[HLF+18] Danijar Hafner, Timothy Lillicrap, Ian Fischer, Ruben Villegas, David Ha, Honglak Lee, and James Davidson. Learning Latent Dynamics for Planning from Pixels. *arXiv [cs.LG]*, November 2018.

[HLNB21] Danijar Hafner, Timothy Lillicrap, Mohammad Norouzi, and Jimmy Ba. Mastering Atari with Discrete World Models. In *International Conference on Learning Representations (ICLR)*, 2021.

[HLS+19] Yuanming Hu, Jiancheng Liu, Andrew Spielberg, Joshua B Tenenbaum, William T Freeman, Jiajun Wu, Daniela Rus, and Wojciech Matusik. ChainQueen: A Real-Time Differentiable Physical Simulator for Soft Robotics. In *International Conference on Robotics and Automation (ICRA)*, 2019.

[HMC+21] Eric Heiden, David Millard, Erwin Coumans, Yizhou Sheng, and Gaurav S Sukhatme. NeuralSim: Augmenting Differentiable Simulators with Neural Networks. In *International Conference on Robotics and Automation (ICRA)*, 2021.

[HMN+21] Eric Heiden, Miles Macklin, Yashraj Narang, Dieter Fox, Animesh Garg, and Fabio Ramos. DiSECt: A Differentiable Simulation Engine for Autonomous Robotic Cutting. In *Robotics: Science and Systems (RSS)*, 2021.

[HPBL23] Danijar Hafner, Jurgis Pasukonis, Jimmy Ba, and Timothy Lillicrap. Mastering Diverse Domains through World Models. *arXiv [cs.AI]*, January 2023.

[HS18] David Ha and Jürgen Schmidhuber. World Models. *arXiv [cs.LG]*, March 2018.

[HSW24] Nicklas Hansen, Hao Su, and Xiaolong Wang. TD-MPC2: Scalable, Robust World Models for Continuous Control. In *International Conference on Learning Representations (ICLR)*, 2024.

[HT81] Donald F Hoyt and C Richard Taylor. Gait and the energetics of locomotion in horses. *Nature*, 292(5820):239–240, July 1981.

[HWS+15] Nicolas Heess, Greg Wayne, David Silver, Timothy Lillicrap, Yuval Tassa, and Tom Erez. Learning Continuous Control Policies by Stochastic Value Gradients. In *Neural Information Processing Systems (NIPS)*, 2015.

[HZAL18] Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft Actor-Critic: Off-Policy Maximum Entropy Deep Reinforcement Learning with a Stochastic Actor. In *International Conference on Machine Learning (ICML)*, 2018.

[HZH+18] Tuomas Haarnoja, Aurick Zhou, Kristian Hartikainen, George Tucker, Sehoon Ha, Jie Tan, Vikash Kumar, Henry Zhu, Abhishek Gupta, Pieter Abbeel, and Sergey Levine. Soft Actor-Critic Algorithms and Applications. *arXiv [cs.LG]*, December 2018.

[JFZL19] Michael Janner, Justin Fu, Marvin Zhang, and Sergey Levine. When to Trust Your Model: Model-Based Policy Optimization. In *Neural Information Processing Systems (NeurIPS)*, 2019.

[JLL21] Michael Janner, Qiyang Li, and Sergey Levine. Offline Reinforcement Learning as One Big Sequence Modeling Problem. In *Neural Information Processing Systems (NeurIPS)*, 2021.

[JR92] Michael I Jordan and David E Rumelhart. Forward models: Supervised learning with a distal teacher. *Cogn. Sci.*, 16(3):307–354, July 1992.

[KAN08] J Z Kolter, Pieter Abbeel, and Andrew Y Ng. Hierarchical Apprenticeship Learning with Application to Quadruped Locomotion. In *Neural Information Processing Systems (NIPS)*, pages 769–776, 2008.

[KBL+23] Elia Kaufmann, Leonard Bauersfeld, Antonio Loquercio, Matthias Müller, Vladlen Koltun, and Davide Scaramuzza. Champion-level drone racing using deep reinforcement learning. *Nature*, 620(7976):982–987, August 2023.

[KCD+18] Thanard Kurutach, Ignasi Clavera, Yan Duan, Aviv Tamar, and Pieter Abbeel. Model-Ensemble Trust-Region Policy Optimization. *arXiv [cs.LG]*, February 2018.

[Kir04] Donald E Kirk. *Optimal Control Theory: An Introduction*. Courier Corporation, January 2004.

[KT00] Vijay R Konda and John N Tsitsiklis. Actor-Critic Algorithms. In *Neural Information Processing Systems (NIPS)*, pages 1008–1014, 2000.

[LBPT+21] 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 *NeurIPS Workshop "Deep Reinforcement Learning"*, 2021.

[LDW+23] Jessy Lin, Yuqing Du, Olivia Watkins, Danijar Hafner, Pieter Abbeel, Dan Klein, and Anca Dragan. Learning to Model the World with Language. *arXiv [cs.CL]*, July 2023.

[LGAL17] Yuxuan Liu, Abhishek Gupta, Pieter Abbeel, and Sergey Levine. Imitation from Observation: Learning to Imitate Behaviors from Raw Video via Context Translation. *arXiv [cs.LG]*, July 2017.

[LHL+22] Xingyu Lin, Zhiao Huang, Yunzhu Li, Joshua B Tenenbaum, David Held, and Chuang Gan. DiffSkill: Skill Abstraction from Differentiable Physics for Deformable Object Manipulations with Tools. In *International Conference on Learning Representations (ICLR)*, 2022.

[LKS+22] Lauro Langosco, Jack Koch, Lee Sharkey, Jacob Pfau, Laurent Orseau, and David Krueger. Goal Misgeneralization in Deep Reinforcement Learning. In *International Conference on Machine Learning (ICML)*, 2022.

[LLW+22] Siqi Liu, Guy Lever, Zhe Wang, Josh Merel, S M Ali Eslami, Daniel Hennes, Wojciech M Czarnecki, Yuval Tassa, Shayegan Omidshafiei, Abbas Abdolmaleki, Noah Y Siegel, Leonard Hasenclever, Luke Marris, Saran Tunyasuvunakool, H Francis Song, Markus Wulfmeier, Paul Muller, Tuomas Haarnoja, Brendan Tracey, Karl Tuyls, Thore Graepel, and Nicolas Heess. From motor control to team play in simulated humanoid football. *Sci Robot*, 7(69):eabo0235, August 2022.

[MAF23] Vincent Micheli, Eloi Alonso, and François Fleuret. Transformers are Sample-Efficient World Models. In *International Conference on Learning Representations (ICLR)*, 2023.

[MHG+19] Josh Merel, Leonard Hasenclever, Alexandre Galashov, Arun Ahuja, Vu Pham, Greg Wayne, Yee Whye Teh, and Nicolas Heess. Neural probabilistic motor primitives for humanoid control. In *International Conference on Learning Representations (ICLR)*, 2019.

[MMPG24] Skander Moalla, Andrea Miele, Razvan Pascanu, and Caglar Gulcehre. No representation, no trust: Connecting representation, collapse, and trust issues in PPO. *arXiv [cs.LG]*, May 2024.

[MPH+21] Miguel Angel Zamora Mora, Momchil Peychev, Sehoon Ha, Martin Vechev, and Stelian Coros. PODS: Policy Optimization via Differentiable Simulation. In Marina Meila and Tong Zhang, editors, *International Conference on Machine Learning (ICML)*, volume 139 of *Proceedings of Machine Learning Research*, pages 7805–7817. PMLR, 2021.

[MTA+20] Josh Merel, Saran Tunyasuvunakool, Arun Ahuja, Yuval Tassa, Leonard Hasenclever, Vu Pham, Tom Erez, Greg Wayne, and Nicolas Heess. Catch & Carry: Reusable Neural Controllers for Vision-Guided Whole-Body Tasks. *ACM Trans. Graph.*, 39(4):39:1–39:12, July 2020.

[MTD+17] Josh Merel, Yuval Tassa, T B Dhruva, Sriram Srinivasan, Jay Lemmon, Ziyu Wang, Greg Wayne, and Nicolas Heess. Learning human behaviors from motion capture by adversarial imitation. *arXiv [cs.RO]*, July 2017.

[MWG+21] Viktor Makoviychuk, Lukasz Wawrzyniak, Yunrong Guo, Michelle Lu, Kier Storey, Miles Macklin, David Hoeller, Nikita Rudin, Arthur Allshire, Ankur Handa, and Gavriel State. Isaac Gym: High Performance GPU-Based Physics Simulation For Robot Learning. Technical report, Nvidia, August 2021.

[NW90] D H Nguyen and B Widrow. Neural networks for self-learning control systems. *IEEE Control Syst. Mag.*, 10(3):18–23, April 1990.

[OAB+18] OpenAI, Marcin Andrychowicz, Bowen Baker, Maciek Chociej, Rafal Jozefowicz, Bob McGrew, Jakub Pachocki, Arthur Petron, Matthias Plappert, Glenn Powell, Alex Ray, Jonas Schneider, Szymon Sidor, Josh Tobin, Peter Welinder, Lilian Weng, and Wojciech Zaremba. Learning dexterous in-hand manipulation. Technical report, OpenAI, August 2018.

[Ope19] OpenAI. Solving Rubik's Cube with a Robot Hand. 2019.

[PALvdP18] Xue Bin Peng, Pieter Abbeel, Sergey Levine, and Michiel van de Panne. DeepMimic: Example-Guided Deep Reinforcement Learning of Physics-Based Character Skills. *arXiv [cs.GR]*, April 2018.

[PBS22] Alexander Pan, Kush Bhatia, and Jacob Steinhardt. The Effects of Reward Misspecification: Mapping and Mitigating Misaligned Models. In *International Conference on Learning Representations (ICLR)*, 2022.

[PBvdP16] Xue Bin Peng, Glen Berseth, and Michiel van de Panne. Terrain-adaptive locomotion skills using deep reinforcement learning. *ACM Trans. Graph.*, 35(4):81, July 2016.

[PBYvdP17] Xue Bin Peng, Glen Berseth, Kangkang Yin, and Michiel van de Panne. DeepLoco: Dynamic Locomotion Skills Using Hierarchical Deep Reinforcement Learning. *ACM Transactions on Graphics (SIGGRAPH)*, 36(4), 2017.

[PMA+21] Xue Bin Peng, Ze Ma, Pieter Abbeel, Sergey Levine, and Angjoo Kanazawa. AMP: Adversarial Motion Priors for Stylized Physics-Based Character Control. *arXiv [cs.GR]*, April 2021.

[Pom89] Dean Pomerleau. ALVINN: An Autonomous Land Vehicle in a Neural Network. In *Neural Information Processing Systems (NIPS)*, pages 305–313, 1989.

[Pom90] Dean Pomerleau. Rapidly Adapting Artificial Neural Networks for Autonomous Navigation. In *Neural Information Processing Systems (NIPS)*, pages 429–435, 1990.

[PRPD18] Paavo Parmas, Carl Edward Rasmussen, Jan Peters, and Kenji Doya. PIPPS: Flexible Model-Based Policy Search Robust to the Curse of Chaos. In *International Conference on Machine Learning (ICML)*, 2018.

[Put94] Martin L Puterman. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. Wiley, 1994.

[PvdP16] Xue Bin Peng and Michiel van de Panne. Learning Locomotion Skills Using DeepRL: Does the Choice of Action Space Matter? *arXiv [cs.LG]*, November 2016.

[QLKL20] Yi-Ling Qiao, Junbang Liang, Vladlen Koltun, and Ming C Lin. Scalable Differentiable Physics for Learning and Control. In *International Conference on Machine Learning (ICML)*, 2020.

[QLKL21] Yi-Ling Qiao, Junbang Liang, Vladlen Koltun, and Ming C Lin. Efficient Differentiable Simulation of Articulated Bodies. In *International Conference on Machine Learning (ICML)*, 2021.

[RA98] Jette Randløv and Preben Alstrøm. Learning to Drive a Bicycle Using Reinforcement Learning and Shaping. In *International Conference on Machine Learning (ICML)*, volume 98, pages 463–471, 1998.

[RAB12] Stephane Ross and J Andrew Bagnell. Agnostic System Identification for Model-Based Reinforcement Learning. In *International Conference on Machine Learning (ICML)*, 2012.

[RAW+18] Scott Reed, Yusuf Aytar, Ziyu Wang, Tom Paine, Aäron van den Oord, Tobias Pfaff, Sergio Gomez, Alexander Novikov, David Budden, and Oriol Vinyals. Visual Imitation With a Minimal Adversary. Technical report, DeepMind, 2018.

[RB10] Stéphane Ross and J Andrew Bagnell. Efficient Reductions for Imitation Learning. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2010.

[RBS07] Nathan Ratliff, J Andrew Bagnell, and Siddhartha S Srinivasa. Imitation learning for locomotion and manipulation. In *IEEE-RAS International Conference on Humanoid Robots*, pages 392–397, November 2007.

[RBZ06] Nathan D Ratliff, J Andrew Bagnell, and Martin A Zinkevich. Maximum Margin Planning. In *International Conference on Machine Learning (ICML)*, pages 729–736, 2006.

[RF87] A J Robinson and F Fallside. The Utility Driven Dynamic Error Propagation Network. Technical Report CUED/F-INFENG/TR.1, University of Cambridge, November 1987.

[RGB11] Stéphane Ross, Geoffrey J Gordon, and J Andrew Bagnell. A Reduction of Imitation Learning and Structured Prediction to No-Regret Online Learning. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2011.

[RH91] Marc H Raibert and Jessica K Hodgins. Animation of dynamic legged locomotion. In *ACM Transactions on Graphics (SIGGRAPH)*, volume 25, pages 349–358, New York, NY, USA, 1991. Association for Computing Machinery.

[RHW86] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning representations by back-propagating errors. *Nature*, 323(6088):533–536, October 1986.

[Ros22] Sheldon M Ross. *Simulation (sixth edition)*. Academic Press, San Diego, CA, 6 edition, December 2022.

[Rus98] Stuart Russell. Learning agents for uncertain environments (extended abstract). In *Conference on Learning Theory (COLT)*, 1998.

[RXZ+24] Ilija Radosavovic, Tete Xiao, Bike Zhang, Trevor Darrell, Jitendra Malik, and Koushil Sreenath. Real-world humanoid locomotion with reinforcement learning. *Sci. Robot.*, 9(89):eadi9579, April 2024.

[RYC+23] Jiawei Ren, Cunjun Yu, Siwei Chen, Xiao Ma, Liang Pan, and Ziwei Liu. DiffMimic: Efficient Motion Mimicking with Differentiable Physics. In *International Conference on Learning Representations (ICLR)*, 2023.

[RZS+24] Ilija Radosavovic, Bike Zhang, Baifeng Shi, Jathushan Rajasegaran, Sarthak Kamat, Trevor Darrell, Koushil Sreenath, and Jitendra Malik. Humanoid locomotion as next token prediction. *arXiv [cs.RO]*, February 2024.

[SB98] Richard S Sutton and Andrew G Barto. *Reinforcement Learning: An Introduction*. MIT Press, 1998.

[SC23] Jiarui Sun and Girish Chowdhary. Towards Globally Consistent Stochastic Human Motion Prediction via Motion Diffusion. *arXiv [cs.CV]*, May 2023.

[SCAP22] Nur Muhammad Mahi Shafiullah, Zichen Jeff Cui, Ariuntuya Altanzaya, and Lerrel Pinto. Behavior Transformers: Cloning k modes with one stone. In *Neural Information Processing Systems (NeurIPS)*, 2022.

[Sch11] Jürgen Schmidhuber. Making the World Differentiable: On Using Self-Supervised Fully Recurrent Neural Networks for Dynamic Reinforcement Learning and Planning in Non-Stationary Environments. 2011.

[SHKK22] Joar Skalse, Nikolaus H R Howe, Dmitrii Krasheninnikov, and David Krueger. Defining and Characterizing Reward Hacking. In *Neural Information Processing Systems (NeurIPS)*, 2022.

[SICP20] Tim Salzmann, Boris Ivanovic, Punarjay Chakravarty, and Marco Pavone. Trajectron++: Dynamically-Feasible Trajectory Forecasting With Heterogeneous Data. *arXiv [cs.RO]*, January 2020.

[Sim94] Karl Sims. Evolving Virtual Creatures. In *ACM Transactions on Graphics (SIGGRAPH)*, SIGGRAPH '94, pages 15–22, New York, NY, USA, 1994. ACM.

[SKL22] Laura Smith, Ilya Kostrikov, and Sergey Levine. A Walk in the Park: Learning to Walk in 20 Minutes With Model-Free Reinforcement Learning. *arXiv [cs.RO]*, August 2022.

[SLB09] Satinder Singh, Richard L Lewis, and Andrew G Barto. Where Do Rewards Come From? 2009.

[SLM+15] John Schulman, Sergey Levine, Philipp Moritz, Michael I Jordan, and Pieter Abbeel. Trust Region Policy Optimization. In *International Conference on Machine Learning (ICML)*, February 2015.

[SMK22] Sebastian Starke, Ian Mason, and Taku Komura. DeepPhase: periodic autoencoders for learning motion phase manifolds. *ACM Trans. Graph.*, 41(4):1–13, July 2022.

[SSM01] Richard S Sutton, Satinder Singh, and David McAllester. Comparing Policy-Gradient Algorithms. 2001.

[SSPS21] David Silver, Satinder Singh, Doina Precup, and Richard S Sutton. Reward is enough. *Artif. Intell.*, 299:103535, October 2021.

[SSRD11] Ajay Seth, Michael Sherman, Jeffrey A Reinbolt, and Scott L Delp. OpenSim: a musculoskeletal modeling and simulation framework for in silico investigations and exchange. *Procedia IUTAM*, 2:212–232, 2011.

[SSZT22] Hyung Ju Suh, Max Simchowitz, Kaiqing Zhang, and Russ Tedrake. Do Differentiable Simulators Give Better Policy Gradients? In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato, editors, *International Conference on Machine Learning (ICML)*, volume 162, pages 20668–20696, 2022.

[Sut91] Richard S Sutton. Dyna, an Integrated Architecture for Learning, Llanning, and Reacting. *SIGART Bull.*, 2(4):160–163, July 1991.

[SWD+17] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Klimov Oleg. Proximal Policy Optimization Algorithms. July 2017.

[TET12] Emanuel Todorov, Tom Erez, and Yuval Tassa. MuJoCo: A physics engine for model-based control. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5026–5033, October 2012.

[TFR+17] J Tobin, R Fong, A Ray, J Schneider, W Zaremba, and P Abbeel. Domain randomization for transferring deep neural networks from simulation to the real world. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 23–30, September 2017.

[TRG+23] Guy Tevet, Sigal Raab, Brian Gordon, Yonatan Shafir, Daniel Cohen-Or, and Amit H Bermano. Human Motion Diffusion Model. In *International Conference on Learning Representations (ICLR)*, 2023.

[TSEG20] Manan Tomar, Lior Shani, Yonathan Efroni, and Mohammad Ghavamzadeh. Mirror descent policy optimization. *arXiv [cs.LG]*, May 2020.

[TWS18a] Faraz Torabi, Garrett Warnell, and Peter Stone. Behavioral Cloning from Observation. In *International Joint Conference on Artificial Intelligence (IJCAI)*, 2018.

[TWS18b] Faraz Torabi, Garrett Warnell, and Peter Stone. Generative Adversarial Imitation from Observation. *arXiv [cs.LG]*, July 2018.

[TZC+18] Jie Tan, Tingnan Zhang, Erwin Coumans, Atil Iscen, Yunfei Bai, Danijar Hafner, Steven Bohez, and Vincent Vanhoucke. Sim-to-Real: Learning Agile Locomotion For Quadruped Robots. *arXiv [cs.RO]*, April 2018.

[vdPF93] Michiel van de Panne and Eugene Fiume. Sensor-Actuator Networks. In *ACM Transactions on Graphics (SIGGRAPH)*, SIGGRAPH '93, pages 335–342, New York, NY, USA, September 1993. Association for Computing Machinery.

[VPT+14] Giordano Valente, Lorenzo Pitto, Debora Testi, Ajay Seth, Scott L Delp, Rita Stagni, Marco Viceconti, and Fulvia Taddei. Are subject-specific musculoskeletal models robust to the uncertainties in parameter identification? *PLoS One*, 9(11):e112625, November 2014.

[VSP+17] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention Is All You Need. In *Neural Information Processing Systems (NIPS)*, 2017.

[WB20] Tingwu Wang and Jimmy Ba. Exploring Model-based Planning with Policy Networks. In *International Conference on Learning Representations (ICLR)*, 2020.

[WBR+23] Keenon Werling, Nicholas A Bianco, Michael Raitor, Jon Stingel, Jennifer L Hicks, Steven H Collins, Scott L Delp, and C Karen Liu. AddBiomechanics: Automating model scaling, inverse kinematics, and inverse dynamics from human motion data through sequential optimization. *bioRxivorg*, September 2023.

[Wer90] Paul J Werbos. Backpropagation Through Time: What It Does and How to Do It. *Proc. IEEE*, 78(10):1550–1560, October 1990.

[WGSF20] Tingwu Wang, Yunrong Guo, Maria Shugrina, and Sanja Fidler. UniCon: Universal Neural Controller For Physics-based Character Motion. *arXiv [cs.GR]*, November 2020.

[WOL+21] Keenon Werling, Dalton Omens, Jeongseok Lee, Ioannis Exarchos, and C Karen Liu. Fast and Feature-Complete Differentiable Physics for Articulated Rigid Bodies with Contact. In *Robotics: Science and Systems (RSS)*, 2021.

[WP90] Ronald J Williams and Jing Peng. An Efficient Gradient-Based Algorithm for On-Line Training of Recurrent Network Trajectories. *Neural Comput.*, 2(4):490–501, December 1990.

[WWY22] Alexander Winkler, Jungdam Won, and Yuting Ye. QuestSim: Human Motion Tracking from Sparse Sensors with Simulated Avatars. *arXiv [cs.CV]*, September 2022.

[XMN+22] Jie Xu, Viktor Makoviychuk, Yashraj Narang, Fabio Ramos, Wojciech Matusik, Animesh Garg, and Miles Macklin. Accelerated Policy Learning with Parallel Differentiable Simulation. In *International Conference on Learning Representations (ICLR)*, 2022.

[XXA+23] Pei Xu, Kaixiang Xie, Sheldon Andrews, Paul G Kry, Michael Neff, Morgan Mcguire, Ioannis Karamouzas, and Victor Zordan. AdaptNet: Policy Adaptation for Physics-Based Character Control. *ACM Trans. Graph.*, 42(6):1–17, December 2023.

[YDG+23] Mengjiao Yang, Yilun Du, Kamyar Ghasemipour, Jonathan Tompson, Dale Schuurmans, and Pieter Abbeel. Learning Interactive Real-World Simulators. *arXiv [cs.AI]*, October 2023.

[ZCS+24] Yi Zhao, Le Chen, Jan Schneider, Quankai Gao, Juho Kannala, Bernhard Schölkopf, Joni Pajarinen, and Dieter Büchler. RP1M: A large-scale motion dataset for piano playing with bi-manual dexterous robot hands. *arXiv [cs.RO]*, August 2024.

[ZMBD08] Brian D Ziebart, Andrew L Maas, J Andrew Bagnell, and Anind K Dey. Maximum Entropy Inverse Reinforcement Learning. In *AAAI Conference on Artificial Intelligence*, pages 1433–1438, 2008.