Learning Symmetric and Low-Energy Locomotion

WENHAO YU, Georgia Institute of Technology
GREG TURK, Georgia Institute of Technology
C.KAREN LIU, Georgia Institute of Technology

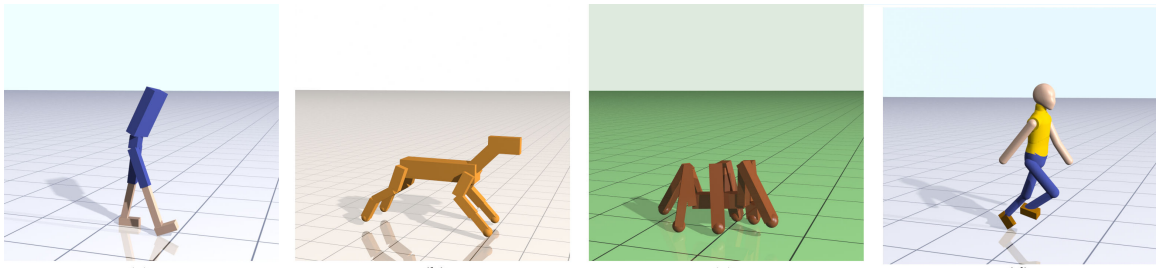

Fig. 1. Locomotion Controller trained for different creatures. (a) Biped walking. (b) Quadruped galloping. (c) Hexapod Walking. (d) Humanoid running.

Learning locomotion skills is a challenging problem. To generate realistic and smooth locomotion, existing methods use motion capture, finite state machines or morphology-specific knowledge to guide the motion generation algorithms. Deep reinforcement learning (DRL) is a promising approach for the automatic creation of locomotion control. Indeed, a standard benchmark for DRL is to automatically create a running controller for a biped character from a simple reward function [Duan et al. 2016]. Although several different DRL algorithms can successfully create a running controller, the resulting motions usually look nothing like a real runner. This paper takes a minimalist learning approach to the locomotion problem, without the use of motion examples, finite state machines, or morphology-specific knowledge. We introduce two modifications to the DRL approach that, when used together, produce locomotion behaviors that are symmetric, low-energy, and much closer to that of a real person. First, we introduce a new term to the loss function (not the reward function) that encourages symmetric actions. Second, we introduce a new curriculum learning method that provides modulated physical assistance to help the character with left/right balance and forward movement. The algorithm automatically computes appropriate assistance to the character and gradually relaxes this assistance, so that eventually the character learns to move entirely without help. Because our method does not make use of motion capture data, it can be applied to a variety of character morphologies. We demonstrate locomotion controllers for the lower half of a biped, a full humanoid, a quadruped, and a hexapod. Our results show that learned policies are able to produce symmetric, low-energy gaits. In addition, speed-appropriate gait patterns emerge without any guidance from motion examples or contact planning.

Authors' addresses: Wenhao Yu, School of Interactive Computing, Georgia Institute of Technology, wyu68@gatech.edu; Greg Turk, School of Interactive Computing, Georgia Institute of Technology, turk@cc.gatech.edu; C.Karen Liu, School of Interactive Computing, Georgia Institute of Technology, karenliu@cc.gatech.edu.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
© 2018 Copyright held by the owner/author(s). Publication rights licensed to Association for Computing Machinery.
0730-0301/2018/8-ART144 $15.00
https://doi.org/10.1145/3197517.3201397

CCS Concepts: • **Computing methodologies** → **Animation**; **Reinforcement Learning**;

Additional Key Words and Phrases: locomotion, reinforcement learning, curriculum learning

ACM Reference Format:
Wenhao Yu, Greg Turk, and C.Karen Liu. 2018. Learning Symmetric and Low-Energy Locomotion. *ACM Trans. Graph.* 37, 4, Article 144 (August 2018), 12 pages. https://doi.org/10.1145/3197517.3201397

1 INTRODUCTION

Creating an animated character that can walk is a fascinating challenge for graphics researchers and animators. Knowledge from biomechanics, physics, robotics, and animation give us ideas for how to coordinate the virtual muscles of a character's body to move it forward while maintaining balance and style. Whether physical or digital, the creators of these characters apply physics principles, borrow ideas from domain experts, use motion data, and undertake arduous trial-and-error to craft lifelike movements that mimic real-world animals and humans. While these characters can be engineered to exhibit locomotion behaviors, a more intriguing question is whether the characters can learn locomotion behaviors on their own, the way a human toddler can.

The recent disruptive development in Deep Reinforcement Learning (DRL) suggests that the answer is yes. Researchers indeed showed that artificial agents can learn some form of locomotion using advanced policy learning methods with a large amount of computation. Even though such agents are able to move from point A to point B without falling, the resulting motion usually exhibits jerky, high-frequency movements. The motion artifacts can be mitigated by introducing motion examples or special objectives in the reward function, but these remedies are somewhat unsatisfying as they sacrifice generality of locomotion principles and return partway to heavily engineered solutions.

This paper takes a minimalist approach to the problem of learning locomotion. Our hypothesis is that natural locomotion will emerge from simple and well-known principles in biomechanics, without

the need of motion examples or morphology-specific considerations. If the agent can successfully learn in this minimal setting, we further hypothesize that our learning method can be generalized to agents with different morphologies and kinematic properties. Our approach is inspired by the recent work that applies reinforcement learning to train control policies represented as neural networks, but aims to address two common problems apparent in the motions produced by the existing methods. First, we observe that the motions appear much more energetic than biological systems. Second, an agent with perfectly symmetrical morphology often produces visibly asymmetrical motion, contradicting the observation in biomechanics literature that gaits are statistically symmetrical [Herzog et al. 1989]. Therefore, we propose a new policy learning method that minimizes energy consumption and encourages gait symmetry to improve the naturalness of locomotion.

While most existing methods in motor skill learning penalize the use of energy in the reward function, the weighting of the penalty is usually relatively low for fear of negatively impacting the learning of the main task (e.g. maintaining balance). As a result, the energy term serves merely as a regularizer and has negligible effect on preventing the agent from using excessive joint torques. We introduce a new curriculum learning method that allows for a high energy penalty while still being able to learn successfully using the existing policy learning algorithms. The curriculum provides modulated physical assistance appropriate to the current skill level of the learner, ensuring continuous progress toward successful locomotion with low energy consumption. Our algorithm automatically computes the assistive forces to help the character with lateral balance and forward movement. The curriculum gradually relaxes the assistance, so that eventually the character learns to move entirely without help.

In addition to energy consumption, gait symmetry offers both stable and adaptive locomotion that reduces the risk of falling [Patterson et al. 2008]. The symmetry of walking trajectories can be measured by established metric used in clinical settings [Nigg et al. 1987]. However, directly using this metric in the reward function leads to two complications for policy gradient methods. First, the requirement of evaluating an entire trajectory introduces a delayed and sparse reward, resulting in a much harder learning problem. Second, the policy, especially at the beginning of learning, might not be able to produce structured, cyclic motion, rendering the symmetry metric ineffective in rewarding the desired behaviors. Our solution departs from the conventional metrics that measure the symmetry of the states. Instead, we measure the *symmetry of actions* produced by the policy. We propose a mirror symmetry loss in the objective function to penalize the paired limbs for learning different control strategies.

Our evaluation shows that the agent can indeed learn locomotion that exhibits symmetry and speed-appropriate gait patterns and consumes relatively low-energy, without the need of motion examples, contact planning, and additional morphology-specific terms in the reward function. We further show that the same reward function with minimal change of weighting and the learning methodology can be applied to a variety of morphologies, such as bipeds, quadrupeds, or hexapods. We test our method against three baselines: learning without the mirror symmetry loss, learning without the curriculum, and learning without either component. The comparisons show that, without the curriculum learning, the trained policies fail to move forward or/and maintain balance. On the other hand, without the mirror symmetry loss, the learning process takes significantly more trials and results in asymmetric locomotion.

2 RELATED WORK

2.1 Physically-based Character Control

Obtaining natural human and animal motor skills has been a long-standing challenge for researchers in computer animation. Existing algorithms in character control usually require breaking the motion into more manageable parts or using motion data in order to generate natural-looking results [Geijtenbeek and Pronost 2012]. One approach is the use of finite state machines (FSMs) [Coros et al. 2010, 2011; de Lasa et al. 2010; Felis and Mombaur 2016; Geijtenbeek et al. 2013; Hodgins et al. 1995; Jain et al. 2009; Wang et al. 2009, 2012; Yin et al. 2007]. Yin *et al.* used FSMs to connect keyframes of the character poses, which is then combined with feedback rules to achieve balanced walking, skipping and running motion for humanoid characters [Yin et al. 2007]. A different application of FSMs can be seen in [de Lasa et al. 2010], where they formulated the locomotion task as a Quadratic Programming (QP) problem and used an FSM model to switch between different objective terms to achieve walking motion. Although this class of techniques can successfully generate plausible character motions, it is usually difficult to generalize them to non-biped morphologies or arbitrary tasks.

An alternative approach to obtain natural character motions is to incorporate motion data such as videos [Wampler et al. 2014] or motion capture data [da Silva et al. 2008; Lee et al. 2010, 2014; Liu et al. 2005; Muico et al. 2009; Sok et al. 2007; Ye and Liu 2010]. Despite the high fidelity motion this approach can generate, the requirement of motion data does not allow its application to arbitrary character morphologies as well as generalization to novel control tasks.

Apart from designing finite state machines or using motion data, reward engineering combined with trajectory optimization has also been frequently applied to generate physically-based character motions [Al Borno et al. 2013; Ha and Liu 2014; Mordatch et al. 2012, 2013; Wampler and Popović 2009]. Al Borno *et al.* demonstrated a variety of humanoid motor skills by breaking a sequence of motion into shorter windows, and for each window a task-specific objective is optimized [Al Borno et al. 2013]. Mordatch *et al.* applied the Contact-Invariant-Optimization algorithm to generate full-body humanoid locomotion with muscle-based lower-body actuations [Mordatch et al. 2013]. Symmetry and periodicity of the motion was explicitly enforced to generate realistic locomotion. Trajectory-optimization based methods provide a general framework for synthesizing character motions. However, a few common drawbacks for this category of methods include: they are usually off-line methods, they are sensitive to large perturbations, and they require explicitly modeling of system dynamics. To overcome the first two issues, Mordatch *et al.* trained a neural network policy using data generated from a trajectory optimization algorithm and demonstrated interactive control of character locomotion with different morphologies [Mordatch et al. 2015]. In this work, we aim to develop an algorithm that

can synthesize plausible locomotion controllers with minimal prior knowledge about the character, the target motion, and the system dynamics.

2.2 Reinforcement Learning

Reinforcement Learning (RL) provides a general framework for modeling and solving control of physics-based characters. In computer animation, policy search methods, a sub-area of RL, have been successfully applied in optimizing control policies for complex motor skills, such as cartwheels [Liu et al. 2016], monkey-vaults [Ha and Liu 2014], bicycle stunts [Tan et al. 2014], skateboarding [Liu and Hodgins 2017] and locomotion [Geijtenbeek et al. 2013; Levine and Koltun 2014; Peng et al. 2015, 2016, 2017; Won et al. 2017]. Two major classes of policy search methods used in the previous work include sampling-based methods [Geijtenbeek et al. 2013; Tan et al. 2014] and gradient-based methods [Levine and Koltun 2014; Liu et al. 2016; Peng et al. 2016, 2017].

A representative example of sampling-based method is CMA-ES, which iterates between evaluating a set of sampled parameters and improving the sampling distribution [Hansen and Ostermeier 1996]. This class of policy search methods is relatively robust to local minima and does not require gradient information. However, the sample number and memory requirement usually scales with the number of model parameters, making it less suitable to optimize models with many parameters such as deep neural networks. To combat this constraint on policy complexity, researchers resort to specialized controller design and a fine-tuned reward structure.

On the other hand, gradient-based methods naturally fit into the stochastic gradient descent framework, an algorithm that has been successfully demonstrated to train deep neural networks with hundreds of millions of parameters [LeCun et al. 2015]. Gradient-based methods like REINFORCE exhibit large variance in the estimated gradient, limiting their application to relatively simple control problems [Sutton et al. 2000]. Recent developments in deep reinforcement learning have seen significant progress on improving the gradient estimation accuracy and stability of training, leading to algorithms such as DDPG [Lillicrap et al. 2015], A3C [Mnih et al. 2016], TRPO [Schulman et al. 2015a], and PPO [Schulman et al. 2017], which can be used to solve complex motor control problems. For example, by combining TRPO with Generalized Advantage Estimation [Schulman et al. 2015b], Schulman *et al.* demonstrated learning of locomotion controllers for a 3D humanoid character. Later, they proposed Proximal Policy Optimization (PPO), which further improved the data efficiency of the algorithm [Schulman et al. 2017]. Despite the impressive feat of tackling the 3D biped locomotion problem from scratch, the resulting motion usually looks jerky and unnatural. Peng *et al.* achieved significantly more natural-looking biped locomotion by combining motion capture data with an actor-critic algorithm [Peng et al. 2017].

2.3 Curriculum Learning

Our approach is also inspired by works in curriculum learning (CL). The general idea behind CL is to present the training data to the learning algorithm in an order of increasing complexity [Bengio et al. 2009]. Researchers in machine learning have shown that CL can improve performance and efficiency of learning problems such as question-answering [Graves et al. 2017], classification [Bengio et al. 2009; Pentina et al. 2015], navigation [Held et al. 2017; Matiisen et al. 2017] and game playing [Narvekar et al. 2016].

Curriculum learning has also been applied to learning motor skills in character animation and robotics. Florensa *et al.* demonstrated successful training of robot manipulation and navigations controllers, where the policies are trained with initial states that are increasingly farther from a goal state [Florensa et al. 2017]. Pinto *et al.* applied CL to improve the efficiency of learning of a robot grasping controller [Pinto and Gupta 2016]. Karpathy et al trained a single legged character to perform acrobatic motion by decomposing the motion into a high-level curriculum with a few sub-tasks and learned each sub-task with a low-level curriculum[Karpathy and Van De Panne 2012]. In learning locomotion controllers, van de Panne *et al.* applied external torques to the character in order to keep it upright during training and demonstrated improved learning performance [Van de Panne and Lamouret 1995]. Similarly, Wu *et al.* used helper forces to assist in optimizing a set of locomotion controllers, which are then used by a foot-step planner to controller the character to walk on uneven terrains [Wu and Popović 2010]. Our method shares the similar idea of using an external controller to assist the character in learning locomotion gaits, but differs from their works in two key aspects. First, in addition to balance assistance, our curriculum learning also provides *propelling assistance* which shows significant improvement over balance assistance alone. Second, our method gradually reduces the strength of the assistance forces without affecting the reward function, while their method explicitly minimizes the assistances force in the optimization which might require objective function tuning. Yin *et al.* applied continuation method to search for a curriculum path that gradually adapt a nominal locomotion controller to different tasks such as stepping over obstacle or walking on ice [Yin et al. 2008]. More recently, Heess *et al.* demonstrated learning of agile locomotion controllers in a complex environment using deep Reinforcement Learning [Heess et al. 2017]. They applied PPO to train the agents on environments with increasing complexity, which provided a environment-centered curriculum. Similarly, we also find that efficiency of curriculum learning can be improved by using a environment-centered curriculum.

3 BACKGROUND: POLICY LEARNING

The locomotion learning process can be modeled as a Markov Decision Process (MDP), defined by a tuple: $(\mathcal{S}, \mathcal{A}, r, \rho_0, P, \gamma)$, where $\mathcal{S}$ is the state space, $\mathcal{A}$ is the action space, $r : \mathcal{S} \times \mathcal{A} \mapsto \mathbb{R}$ is the reward function, ρ_0 is the initial state distribution, $P : \mathcal{S} \times \mathcal{A} \mapsto \mathcal{S}$ is the transition function and γ is the discount factor. Our goal is to solve for the optimal parameters θ of a policy $\pi_\theta : \mathcal{S} \times \mathcal{A} \mapsto \mathbb{R}$, which maximizes the expected long-term reward:

$$\pi_{\theta^*} = \underset{\theta}{\operatorname{argmax}} \ \mathbb{E}_{\mathbf{s}\sim\rho_0}[V^\pi(\mathbf{s})], \tag{1}$$

where the value function of a policy, $V^\pi : \mathcal{S} \mapsto \mathbb{R}$, is defined as the expected long-term reward of following the policy π_θ from some

input state $\mathbf{s}_t$:

$$V^{\pi}(\mathbf{s}_t) = \mathbb{E}_{\mathbf{a}_t \sim \pi, \mathbf{s}_{t+1} \sim P, \ldots}[\sum_{i=0}^{\infty} \gamma^i r(\mathbf{s}_{t+i}, \mathbf{a}_{t+i})]. \quad (2)$$

3.1 Learning Locomotion Policy

In the context of locomotion learning problem, we define a state as $\mathbf{s} = [\mathbf{q}, \dot{\mathbf{q}}, \mathbf{c}, \hat{v}]$, where $\mathbf{q}$ and $\dot{\mathbf{q}}$ are the joint positions and joint velocities. $\mathbf{c}$ is a binary vector with the size equal to the number of end-effectors, indicating the contact state of the end-effectors (1 in contact with the ground and 0 otherwise). $\hat{v}$ is the target velocity of the center of mass in the forward direction. The action $\mathbf{a}$ is simply the joint torques generated by the actuators of the character.

Designing a reward function is one of the most important tasks in solving a MDP. In this work, we use a generic reward function for locomotion similar to those used in RL benchmarks [Duan et al. 2016] [Brockman et al. 2016] [Openai 2017]. It consists of three objectives: move forward, balance, and use minimal actuation.

$$r(\mathbf{s}, \mathbf{a}) = w_v E_v(\mathbf{s}) + E_u(\mathbf{s}) + w_l E_l(\mathbf{s}) + E_a + w_e E_e(\mathbf{a}). \quad (3)$$

The first term of the reward function, $E_v = -|\bar{v}(\mathbf{s}) - \hat{v}|$, encourages the character to move at the desired velocity $\hat{v}$. $\bar{v}(\mathbf{s})$ denotes the average velocity in the most recent 2 seconds. The next three terms are designed to maintain balance. $E_u = -(w_{u_x}|\phi_x(\mathbf{s})| + w_{u_y}|\phi_y(\mathbf{s})| + w_{u_z}|\phi_z(\mathbf{s})|)$ rewards the character for maintaining its torso or head upright, where $\phi(\mathbf{s})$ denotes the orientation of the torso or head. $E_l = -|c_z(\mathbf{s})|$ penalizes deviation from the forward direction, where $c_z(\mathbf{s})$ computes the center of mass (COM) of the character in the frontal axis. E_a is the alive bonus which rewards the character for not being terminated at the current moment. A rollout is terminated when the character fails to keep its COM elevated along the forward direction, or to keep its global orientation upright. Finally, $E_e = -\|\mathbf{a}\|$ penalizes excessive joint torques, ensuring minimal use of energy. Details on the hyper-parameters related to the reward function and the termination conditions are discussed in Section 6.

3.2 Policy Gradient Algorithm

Policy gradient methods have demonstrated success in solving such a high-dimensional, continuous MDP. In this work, we use Proximal Policy Optimization (PPO) [Schulman et al. 2017] to learn the optimal locomotion policy because it provides better data efficiency and learning performance than the alternative learning algorithms. Our method can also be easily applied to other learning algorithms such as Trust Region Policy Optimization (TRPO) [Schulman et al. 2015a] or Deep Deterministic Policy Gradient (DDPG) [Lillicrap et al. 2015].

Like many policy gradient methods, PPO defines an advantage function as $A^{\pi}(\mathbf{s}_t, \mathbf{a}) = Q^{\pi}(\mathbf{s}, \mathbf{a}) - V^{\pi}(\mathbf{s})$, where Q^{π} is the state-action value function that evaluates the return of taking action $\mathbf{a}$ at state $\mathbf{s}$ and following the policy π_θ thereafter. However, PPO minimizes a modified objective function to the original MDP problem:

$$L_{PPO}(\theta) = -\mathbb{E}_{\mathbf{s}_0, \mathbf{a}_0, \mathbf{s}_1, \ldots}[\min(r(\theta)A_t, clip(r(\theta), 1-\epsilon, 1+\epsilon)A_t)], \quad (4)$$

where $r(\theta) = \frac{\pi_\theta(\mathbf{a}|\mathbf{s})}{\pi_{\theta_{old}}(\mathbf{a}|\mathbf{s})}$ is the importance re-sampling term that enables us to use data sampled under an old policy $\pi_{\theta_{old}}$ to estimate expectation for the current policy π_θ. The *min* and the *clip* operators together ensure that π_θ does not change too much from $\pi_{\theta_{old}}$. More details in deriving the objective functions can be found in the original paper on PPO [Schulman et al. 2017].

4 LOCOMOTION CURRICULUM LEARNING

Learning locomotion directly from the principles of minimal energy and gait symmetry is difficult without additional guidance from motion examples or reward function shaping. Indeed, a successful locomotion policy must learn a variety of tasks often with contradictory goals, such as maintaining balance while propelling the body forward, or accelerating the body while conserving energy. One approach to learning such a complex motor skill is to design a curriculum that exposes the learner to a series of tasks with increasing difficulty, eventually leading to the original task.

Our locomotion curriculum learning is inspired by physical learning aids that provide external forces to simplify the motor tasks, such as exoskeletons for gait rehabilitation or training wheels for riding a bicycle. These learning aids create a curriculum to ease the learning process and will be removed when the learner is sufficiently skilled at the original task. To formalize this idea, we view the curriculum as a continuous Euclidean space parameterized by curriculum variables $\mathbf{x} \in \mathbb{R}^n$. The learning begins with the simplest lesson $\mathbf{x}_0$ for the beginner learner, gradually increasing the difficulty toward the original task, which is represented as the origin of the curriculum space (i.e. $\mathbf{x} = \mathbf{0}$). With this notion of a continuous curriculum space, we can then develop a continuous learning method by finding the optimal path from $\mathbf{x}_0$ to the origin in the curriculum space.

Similar to the standard policy gradient method, at each learning iteration, we generate rollouts from the current policy, use the rollout to estimate the gradients of the objective function of policy optimization, and update the policy parameters θ based on the gradient. With curriculum learning, we introduce a virtual assistant to provide assistive forces to the learner during rollout generation. The virtual assistant is updated at each learning iteration such that it provides assistive forces appropriate to the current skill level of the learner.

Two questions remain in our locomotion curriculum learning algorithm. First, what is the most compact set of parameters for the virtual assistant such that locomotion skills can be effectively learned through curriculum? Second, what is the appropriate curriculum schedule, i.e. how much assistive force should we give to the learner at each moment of learning?

4.1 Virtual Assistant

Our virtual assistant provides assistive forces to simplify the two main tasks of locomotion: moving forward and maintaining lateral balance. The lateral balancing force is applied along the frontal axis (left-right) of the learner, preventing it from falling sideway. The propelling force is applied along the sagittal axis, pushing the learner forward to reach the desired velocity. With these two assistive forces, the learner can focus on learning to balance in the sagittal plane as well as keeping the energy consumption low.

Both lateral balancing force and propelling force are produced by a virtual proportional-derivative (PD) controller placed at the pelvis of the learner, as if an invisible spring is attached to the learner to provide support during locomotion. Specifically, the PD controller controls the lateral position and the forward velocity of the learner. The target lateral position is set to 0 for maintaining lateral balance while the target forward velocity is set to the desired velocity $\bar{v}$ for assisting the learner moving forward.

Different levels of assistance from the virtual assistant create different lesson for the learner. We use the stiffness coefficient k_p and the damping coefficient k_d to modulate the strength of the balancing and propelling forces respectively. As such, our curriculum space is parameterized by $\mathbf{x} = (k_p, k_d)$. Any path from $\mathbf{x}_0$ to $(0, 0)$ constitutes a curriculum for the learner.

Our implementation of the virtual assistant is based on the stable proportional-derivative (SPD) controller proposed by Tan *et al.* [2011]. The SPD controller provides a few advantages. First, it does not require any pre-training and can be applied to any character morphology with little tuning. In addition, it provides a smooth assistance in the state space, which facilitates learning. Finally, it is unconditionally stable, allowing us to use large controller gains without introducing instability.

4.2 Curriculum Scheduling

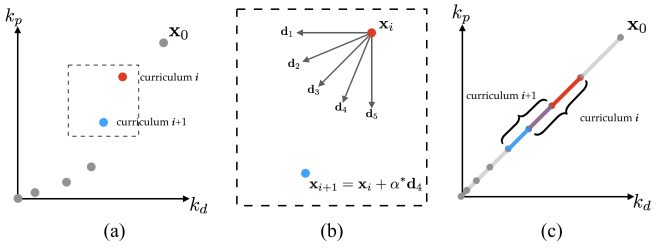

Fig. 2. (a) The learner-centered curriculum determines the lessons adaptively based on the current skill level of the agent, resulting in a piece-wise linear path from $\mathbf{x}_0$ to the origin. (b) In each learning iteration, the learner-centered curriculum finds the next point in the curriculum space using a simple search algorithm that conducts 1D line-searches along five direction. The goal is to find the largest step size, α^*, such that the current policy can still reach 60% of the original return $\bar{R}$. (c) Environment-centered curriculum follows a series of predefined lessons along a linear path from $\mathbf{x}_0$ to the origin. It introduces the learner to a range of lessons in one curriculum learning iteration, resulting in a set of co-linear, overlapping line segments.

The goal of curriculum scheduling is to systematically and gradually reduce the assistance from the initial lesson $\mathbf{x}_0$ and ultimately achieve the final lesson in which the assistive force is completely removed. Designing such a schedule is challenging because an aggressive curriculum that reduces the assistive forces too quickly can fail the learning objectives while a conservative curriculum can lead to inefficient learning.

We propose two approaches to the problem of curriculum scheduling (Figure 2): Learner-centered curriculum and Environment-centered curriculum. The learner-centered curriculum allows the learner to decide the next lesson in the curriculum space, resulting in a piece-wise linear path from $\mathbf{x}_0$ to the origin. The environment-centered curriculum, on the other hand, follows a series of predefined lessons. However, instead of focusing on one lesson at a time, it exposes the learner to a range of lessons in one curriculum learning iteration, resulting in a set of co-linear, overlapping line segments from $\mathbf{x}_0$ to the origin of the curriculum space.

4.2.1 Learner-centered curriculum. The learner-centered curriculum determines the lessons adaptively based on the current skill level of the agent (Algorithm 1). We assume that the initial lesson $\mathbf{x}_0$ is sufficiently simple such that the standard policy learning algorithm can produce a successful policy π, which generates rollouts $\mathcal{B}$ with average return, $\bar{R}$. We then update the lesson to make it more challenging (Algorithm 2) and proceed to the main learning loop. At each curriculum learning iteration, we first update π by running one iteration of the standard policy learning algorithm. If the average of the rollouts from the updated policy can reach h% of the original return $\bar{R}$ ($h = 80$), we update the lesson again using Algorithm 2. The curriculum learning loop terminates when the magnitude of $\mathbf{x}$ is smaller than ϵ ($\epsilon = 5$). We run a final policy learning without any assistance, i.e. $\mathbf{x} = (0, 0)$. At this point, the policy learns this final lesson very quickly.

Given the current lesson $\mathbf{x}_i$, the goal of Algorithm 2 is to find the next point in the curriculum space that is the closest to the origin while the current policy can still retain some level of proficiency. Since it is only a two-dimensional optimization problem and the solution $\mathbf{x}_{i+1}$ lies in $\|\mathbf{x}_{i+1}\| - \|\mathbf{x}_i\| < 0$ and is strictly positive component-wise, we implement a simple search algorithm that conducts 1D line-searches along five directions from $\mathbf{x}_i$: (-1, 0), (-1, -0.5), (-1, -1), (-0.5, -1), (0, -1). For each direction $\mathbf{d}$, the line-search will return the largest step size, α, such that the current policy can still reach l% of the original return $\bar{R}$ ($l = 60$) under the assistance $\mathbf{x}_i + \alpha\mathbf{d}$. Among five line-search results, we choose the largest step size, α^* along the direction $\mathbf{d}^*$ to create the next lesson: $\mathbf{x}_{i+1} = \mathbf{x}_i + \alpha^*\mathbf{d}^*$.

4.2.2 Environment-centered curriculum. Instead of searching for the next lesson, the environment-centered curriculum updates the lessons along a predefined linear path from $\mathbf{x}_0$ to the origin (Algorithm 3). In addition, the learner is trained with a range of lessons $[\mathbf{x}_{begin}, \mathbf{x}_{end}]$ in each curriculum learning iteration. Specifically, the learner will be exposed to an environment in which the virtual assistant starts with $\mathbf{x}_{begin}$ and reduces its strength gradually to $\mathbf{x}_{end}$ at the end of each rollout horizon. The formula of strength reduction from $\mathbf{x}_{begin}$ to $\mathbf{x}_{end}$ can be designed in several ways. We use a simple step function to drop the strength of the virtual assistant by k% every p seconds ($k = 25$ and $p = 3$). Each step in the step function can be considered a learning milestone. Training with a range of milestones in a single rollout prevents the policy from overfitting to a particular virtual assistant, leading to more efficient learning.

In each learning iteration, we first run the standard policy learning for one iteration, with the environment programmed to present the current range of lessons $[\mathbf{x}_{begin}, \mathbf{x}_{end}]$. After the policy is updated, we evaluate the performance of the policy using two conditions. If the policy meets both conditions, we update the range of lessons to $k\% \cdot [\mathbf{x}_{begin}, \mathbf{x}_{end}]$. Note that the updated $\mathbf{x}_{begin}$ is equivalent

Algorithm 1 Learner-Centered Curriculum Learning

$\mathbf{x} = \mathbf{x}_0$
$[\pi, \mathcal{B}] \leftarrow$ PolicyLearning($\mathbf{x}$)
$\bar{R} \leftarrow$ AvgReturn($\mathcal{B}$)
$\mathbf{x} \leftarrow$ UpdateLesson($\mathbf{x}$, $\bar{R}$, π)
while $||\mathbf{x}|| \geq \epsilon$ **do**
    $[\pi, \mathcal{B}] \leftarrow$ OneIterPolicyLearning($\mathbf{x}$)
    $R \leftarrow$ AvgReturn($\mathcal{B}$)
    if $R \geq h\% \cdot \bar{R}$ **then**
        $\mathbf{x} \leftarrow$ UpdateLesson($\mathbf{x}$, $\bar{R}$, π)
$[\pi, \mathcal{B}] \leftarrow$ PolicyLearning($(0, 0)$)
return π

Algorithm 2 Update Lesson

input: $\mathbf{x}, \bar{R}, \pi$
$\mathcal{D} = [(-1, 0), (-1, -0.5), (-1, -1), (-0.5, -1), (0, -1)]$
$\mathbf{x}_{min} = (\infty, \infty)$
for each $\mathbf{d} \in \mathcal{D}$ **do**
    $\alpha^* = \text{argmax}_\alpha \ \alpha$
        s.t. EvalReturn($\mathbf{x} + \alpha\mathbf{d}, \pi$) $> l\% \cdot \bar{R}$
    if $||\mathbf{x} + \alpha^*\mathbf{d}|| < ||\mathbf{x}_{min}||$ **then**
        $\mathbf{x}_{min} = \mathbf{x} + \alpha^*\mathbf{d}$
return $\mathbf{x}_{min}$

to the second milestone of the previous learning iteration, resulting in some overlapping lessons in two consecutive curriculum learning iterations (See Figure 2c). The overlapping lessons are an important aspect of the environment-centered curriculum learning because they allow the character to bootstrap its current skill when learning a new set of predefined lessons. Similar to the learner-centered curriculum, the curriculum learning loop terminates when the magnitude of $\mathbf{x}_{begin}$ is smaller than ϵ ($\epsilon = 5$), and we run a final policy learning without any assistance, i.e. $\mathbf{x}_{begin} = (0, 0)$ and $\mathbf{x}_{end} = (0, 0)$.

The first condition for assessing the progress of learning checks whether the learner is able to reach the second milestone in each rollout. That is, the agent must stay balance for at least $2p$ seconds. Using this condition alone, however, might result in a policy that simply learns to stand still or move minimally in balance. Therefore, we use another condition that requires the average return of the policy to reach a pre-determined threshold, $\bar{R}$, which is $g\%$ of the return from the initial policy trained with full assistance ($g = 70$).

5 MIRROR SYMMETRY LOSS

Symmetry is another important characteristic of a healthy gait. Assessing gait symmetry usually requires at least an observation of a full gait cycle. This requirement poses a challenge to policy learning because the reward cannot be calculated before the end of the gait cycle, leading to a delayed reward function. We propose a new way to encourage gait symmetry by measuring the symmetry of *actions* instead of *states*, avoiding the potential issue of delayed reward.

Imaging a person who is standing in front of a floor mirror with her left hand behind her back. If she uses the right hand to reach

Algorithm 3 Environment-Centered Curriculum Learning

$\mathbf{x}_{begin} = \mathbf{x}_0$
$\mathbf{x}_{end} = (k^2)\% \cdot \mathbf{x}_{begin}$
$[\pi, \mathcal{B}] \leftarrow$ PolicyLearning($\mathbf{x}_{begin}, \mathbf{x}_{end}$)
$\bar{R} \leftarrow g\% \cdot$ AvgReturn($\mathcal{B}$)
while $||\mathbf{x}_{begin}|| \geq \epsilon$ **do**
    $[\pi, \mathcal{B}] \leftarrow$ OneIterPolicyLearning($\mathbf{x}_{begin}, \mathbf{x}_{end}$)
    if BalanceTest($\mathcal{B}$) and AvgReturn($\mathcal{B}$) $> \bar{R}$ **then**
        $\mathbf{x}_{begin} = k\% \cdot \mathbf{x}_{begin}$
        $\mathbf{x}_{end} = k\% \cdot \mathbf{x}_{end}$
$[\pi, \mathcal{B}] \leftarrow$ PolicyLearning($(0, 0), (0, 0)$)
return π

for her hat, what we see in the mirror is a person with her right hand behind her back reaching for a hat using her left hand. Indeed, if the character has a symmetric morphology, the action it takes in some pose during locomotion should be the mirrored version of the action taken when the character is in the mirrored pose. This property can be expressed as:

$$\pi_\theta(\mathbf{s}) = \Psi_a(\pi_\theta(\Psi_o(\mathbf{s}))), \tag{5}$$

where $\Psi_a(\cdot)$ and $\Psi_o(\cdot)$ maps actions and states to their mirrored versions respectively. We overload the notation π_θ to represent the mean action of the stochastic policy. Enforcing Equation 5 as a hard constraint is difficult for standard policy gradient algorithms, but we can formulate a soft constraint and include it in the objective function for policy optimization:

$$L_{sym}(\theta) = \sum_{i=0}^{B} ||\pi_\theta(\mathbf{s}_i) - \Psi_a(\pi_\theta(\Psi_o(\mathbf{s}_i)))||^2, \tag{6}$$

where B is the number of simulation samples per iteration. We use $20,000$ samples in all of our examples. Since Equation 6 is differentiable with respect to the policy parameters θ, it can be combined with the standard reinforcement learning objective and optimized using any gradient-based RL algorithm.

Incorporating the mirror symmetry loss, the final optimization problem for learning the locomotion policy can be defined as:

$$\pi_{\theta^*} = \underset{\theta}{\text{argmin}} \ \ L_{PPO}(\theta) + wL_{sym}(\theta), \tag{7}$$

where w is the weight to balance the importance of the gait symmetry and the expected return of the policy ($w = 4$). Note that the mirror symmetry loss is included in the objective function of the policy optimization, rather than in the reward function of the MDP. This is because L_{sym} explicitly depends on the policy parameters θ, thus adding L_{sym} in the reward function would break the assumption in the Policy Gradient Theorem [Sutton et al. 2000]. That is, changing θ should change the probability of a rollout, not its return. If we included L_{sym} in the reward function, it would change the return of the rollout when θ is changed. Instead, we include L_{sym} in the objective function and calculate its gradient separately from that of the L_{PPO}, which depends on the Policy Gradient Theorem.

6 RESULTS

We evaluate our method on four characters with different morphologies and degrees of freedom. The input to the control policy includes $\mathbf{s} = [\mathbf{q}, \dot{\mathbf{q}}, \mathbf{c}, \hat{v}]$ and the output is the torque generated at each actuated joint, as described in Section 3.1. Because the character is expected to start at zero velocity and accelerate to the target velocity, the policy needs to be trained with a range of velocities from zero to the target. We include $\hat{v}$ in the input of the policy to modulate the initial acceleration; $\hat{v}$ is set to zero at the beginning of the rollout and increases linearly for the first $0.5|\hat{v}|$ seconds, encouraging the character to accelerate at $2m/s^2$. The parameters of the reward functions used in all the experiments are listed in Table 1. We demonstrate the environment-centered curriculum learning for all the examples and selectively use the learner-centered curriculum learning for comparison. The resulting motions can be seen in the supplementary video.

We set the starting point of the curriculum $\mathbf{x}_0$ to $(k_p, k_d) = (2000, 2000)$ in all examples. Note that k_p and k_d are the proportional gains and damping gains in two independent SPD controllers that provide balancing and propelling forces respectively. The damping gain used to compute the balancing force is $0.1k_p$ and the proportional gain used to compute the propelling force is 0.

We use Pydart [Ha 2016], a python binding of the DART library [Liu and Jain 2012] to perform multi-body simulation. We simulate the characters at 500 Hz, and query the control policy every 15 simulation steps, yielding a control frequency of 33 Hz. We use the PPO algorithm implemented in the OpenAI Baselines library [Dhariwal et al. 2017] for training the control policies. The control policies used in all examples are represented by feed-forward neural networks with three fully-connected hidden layers, and each hidden layer consists of 64 units. We fix the sample number to be 20,000 steps per iteration for all examples. The number of iteration required to obtain a successful locomotion controller depends on the complexity of the task, ranging from 500 to 1500, yielding a total sample number between 10 and 30 millions. We perform all training using 8 parallel threads on an Amazon EC2 node with 16 virtual cores and 32G memory. Each training iteration takes 25 − 45s depending on the degrees of freedoms of the character model, leading to a total training time between 4 and 15 hours.

6.1 Locomotion of Different Morphologies

Simplified biped. Bipedal locomotion has been extensively studied in the literature, and it is a familiar form of locomotion to everyone. Thus we start with training a simplified biped character to perform walking and running. The character has 9 links and 21 DOFs, with 1.65m in height and weighs in total 50kg. The results can be seen in Figure 3. As expected, when trained with a low target velocity ($1m/s$), the character exhibits a walking gait. When trained with a high target velocity ($5m/s$), the character uses a running gait indicated by the emergence of a flight phase.

Quadruped. Quadrupeds exhibit a large variety of locomotion gaits, such as pacing, trotting, cantering, and galloping. We applied our approach to a quadruped model as shown in Figure1(b). The model has 13 links and 22 DOFs, with a height of 1.15m and weight of 88.35kg. As quadrupeds can typically move faster than biped, we trained the quadruped to move at 2m/s and 7m/s. The results are shown in Figure 5. The trained policy results in a trotting gait for low target velocity consistently. For high target velocities, the character learns either trotting or galloping, depending on the initial random seed of the policy.

Hexapod. We designed a hexapod creature that has 13 links and 24 DOFs, inspired by the body plan of an insect. We trained the hexapod model to move at 2m/s and 4m/s. As shown in Figure 6, the hexapod learns to use all six legs to move forward at low velocity, while it lifts the front legs and use the middle and hind legs to 'run' forward at higher velocity.

Humanoid. Finally, we trained a locomotion policy for a full humanoid character with a detailed upper body. The character has 13 links and 29 DOFs with 1.75m in height and 76.6kg in weight. We trained the humanoid model to walk at 1.5m/s and run at 5m/s. In addition, we trained the model to walk backward at −1.5m/s. We kept the same reward function parameters between forward and backward walking. Results of the humanoid locomotion can be seen in Figure 4. During walking forward and backward, the character learns to mostly relax its arms without much swinging motion. For running, the character learn to actively swing its arms in order to counteract the angular momentum generated by the leg movements, which stabilizes the torso movements during running.

6.2 Comparison between Learner-centered and Environment-centered Curriculum Learning

We compare the learner-centered and environment-centered curriculum learning algorithms on the simplified biped model. As demonstrated in the supplementary video, both methods can successfully train the character to walk and run at target velocities with symmetric gaits. We further analyze the performance of the two algorithms by comparing how they progress in the curriculum space, as shown in Figure 7. We measure the progress of the curriculum learning with the $l2$ norm of the curriculum parameter $\mathbf{x}$, since the goal is to reach 0 as fast as possible. We can see that environment-centered curriculum learning shows superior data-efficiency by generating a successful policy with about half the data that is required for the learner-centered curriculum learning.

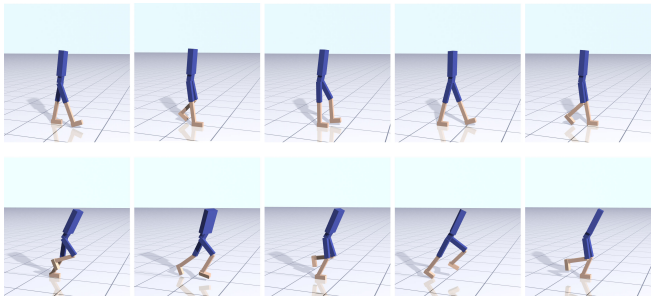

Fig. 3. Simplified biped walking (top) and running (bottom). Results are trained using environment-centered curriculum learning and mirror symmetry loss.

Table 1. Task and reward parameters

Character	$\hat{v}$	w_v	w_{u_x}	w_{u_y}	w_{u_z}	w_l	E_a	w_e
Simplified Biped	0 to $1m/s$	3	1	1	1	3	4	0.4
Simplified Biped	0 to $5m/s$	3	1	1	1	3	7	0.3
Quadruped	0 to $2m/s$	4	0.5	0.5	1	3	4	0.2
Quadruped	0 to $7m/s$	4	0.5	0.5	1	3	11	0.35
Hexapod	0 to $2m/s$	3	1	1	1	3	4	0.2
Hexapod	0 to $4m/s$	3	1	1	1	3	7	0.2
Humanoid	0 to $1.5m/s$	3	1	1.5	1	3	6	0.3
Humanoid	0 to $5m/s$	3	1	1.5	1	3	9	0.15
Humanoid	0 to $-1.5m/s$	3	1	1.5	1	3	6	0.3

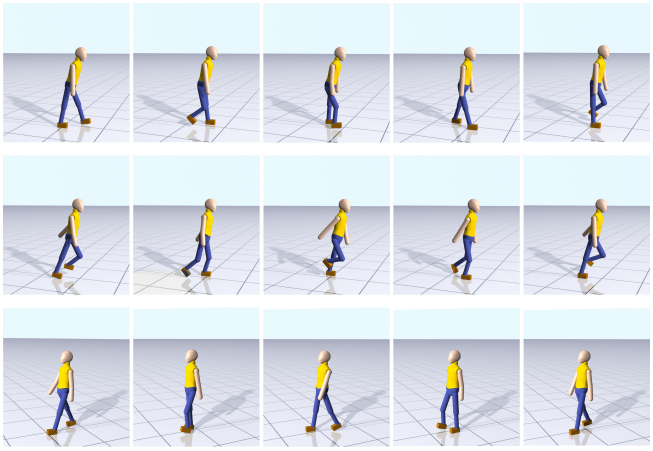

Fig. 4. Humanoid walking (top), running (middle) and backward walking (bottom). Results are trained using environment-centered curriculum learning and mirror symmetry loss.

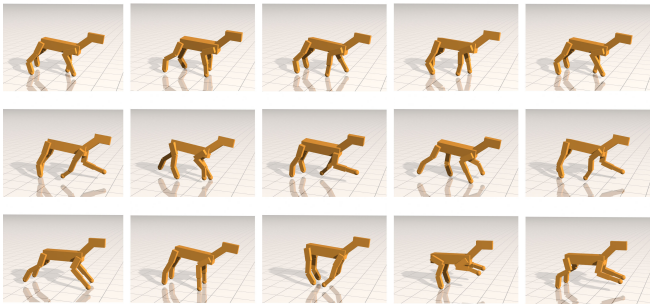

Fig. 5. Dog trotting (top, middle) and galloping (bottom). Results are trained using environment-centered curriculum learning and mirror symmetry loss.

6.3 Comparison with Baseline Methods

To demonstrate the effect of curriculum learning and mirror symmetry loss, we compare our method with environment-centered curriculum learning and mirror symmetry loss (ECL + MSL) to three baseline methods: with environment-based curriculum learning only (ECL), with mirror symmetry loss only (MSL) and using

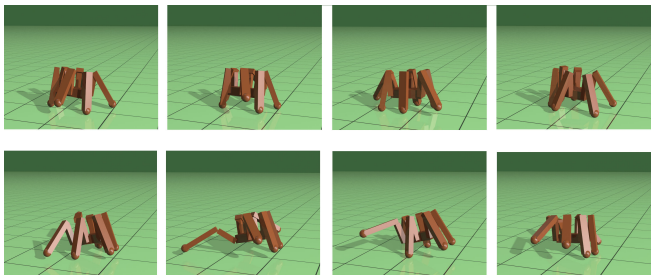

Fig. 6. Hexapod moving at 2m/s (top) and 4m/s (bottom). Results are trained using environment-centered curriculum learning and mirror symmetry loss.

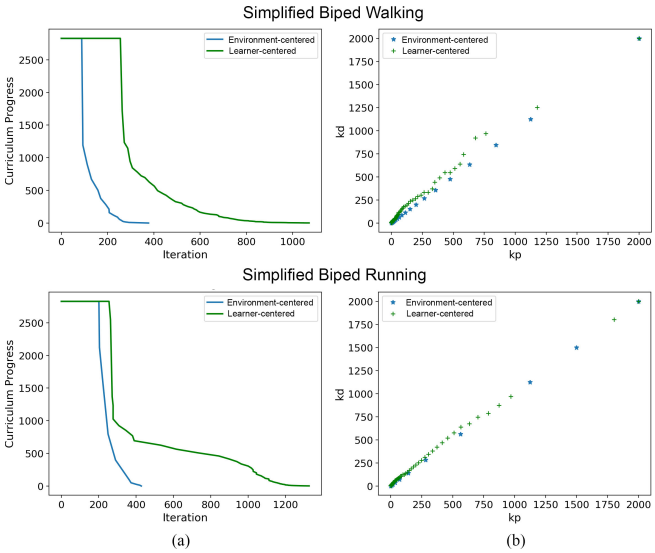

Fig. 7. Comparison between environment-centered and learner-centered curriculum learning for simplified biped tasks. (a) Curriculum progress over iteration numbers. 0 in the y axis means no assistance is provided. (b) Points in the curriculum space visited by the two curriculum update schemes.

vanilla PPO (PPO) with no mirror symmetry loss nor curriculum learning. The baseline methods are trained on the simplified biped

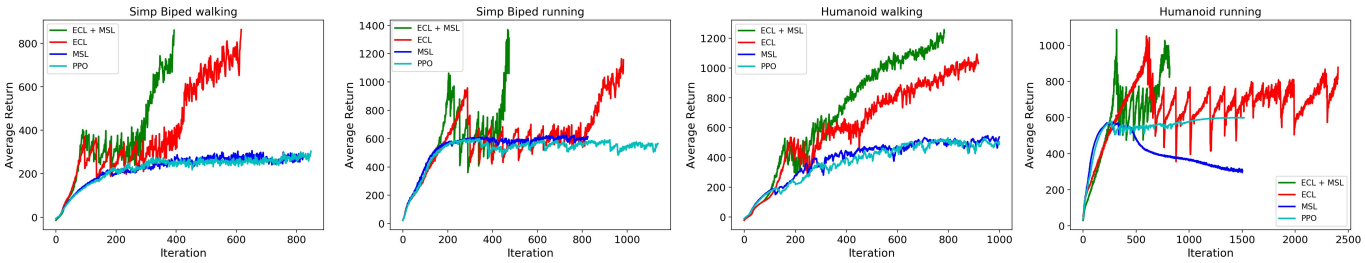

Fig. 8. Learning curves for the proposed algorithm and the baseline methods.

character and the humanoid character for both walking and running. The learning curves for all the tests can be seen in Figure 8. In all four of these tasks, our approach learns faster than all of the baseline methods. Without curriculum learning (i.e. blue and cyan curves), the algorithm typically learns to either fall slowly or stand still (as is shown in the supplementary video). On the other hand, without mirror symmetry loss, the resulting policy usually exhibits asymmetric gaits and the training process is notably slower, which is mostly evident in the running tasks, as shown in Figure 9 and Figure 10.

In addition to the three baseline methods described above, we also trained on the simplified biped walking task using vanilla PPO with a modified reward function, where w_e is reduced to 0.1. This allows the character to use higher torques with little penalty ("PPO high torque" in Table 2). While the character is able to walk forward without falling, the motion appears jerky and uses significantly more torque than our results (see supplementary video).

To compare the policies quantitatively, we report the average actuation magnitude i.e. E_e and use the Symmetry Index metric proposed by Nigg *et al.* to measure the symmetry of the motion [Nigg et al. 1987]:

$$SI(X_L, X_R) = 2\frac{|X_L - X_R|}{X_L + X_R}\%,$$

where X_L and X_R are the average of joint torques produced by the left and right leg respectively. The smaller the value SI is, the more symmetric the gait is. The results can be seen in Table 2. As expected, policies trained with our method uses less joint torque and produces more symmetric gaits.

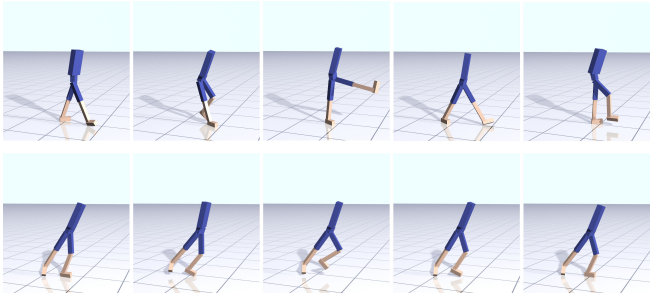

Fig. 9. Simplified biped walking (top) and running (bottom). Results are trained using environment-centered curriculum learning only (no mirror symmetry loss).

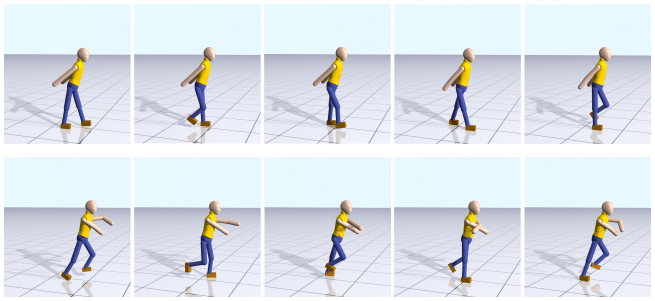

Fig. 10. Humanoid walking (top) and running (bottom). Results are trained using environment-centered curriculum learning only (no mirror symmetry loss).

Table 2. Comparison of trained policies in action magnitude and symmetry. ECL denotes using environment-centered curriculum learning, MSL means mirror symmetry loss and PPO means training with no curriculum learning or mirror symmetry loss. We present results for the successfully trained policies.

Task	Training Setup	E_e	SI
Simplified Biped walk	ECL+MSL	2.01	0.0153
Simplified Biped walk	ECL	2.98	0.1126
Simplified Biped walk	PPO high torque	5.96	0.0416
Simplified Biped run	ECL + MSL	5.57	0.0026
Simplified Biped run	ECL	6.052	0.4982
Humanoid walk	ECL+MSL	6.2	0.0082
Humanoid walk	ECL	7.84	0.0685
Humanoid run	ECL+MSL	17.0976	0.0144
Humanoid run	ECL	18.56	0.0391

6.4 Learning Asymmetric Tasks

One benefit of encouraging symmetric *actions* rather than symmetric *states* is that it allows the motion to appear asymmetric when desired. As shown in Figure 11, we trained a humanoid that is walking while holding a heavy object (10kg) in the right hand. The character uses an asymmetric gait that moves more vigorously on the left side to compensate for the heavy object on the right side. If we chose to enforce symmetry on the states directly, the character would likely use a large amount of torque on the right side to make the poses appear symmetric.

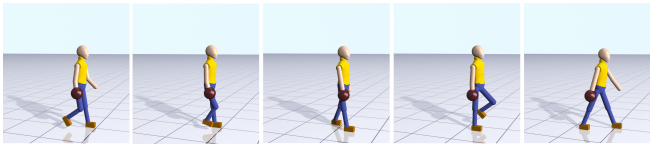

Fig. 11. Humanoid walking holding heavy object in right hand. Results are trained using environment-centered curriculum learning and mirror symmetry loss.

7 DISCUSSION

In this work, we intentionally avoid the use of motion examples to investigate whether learning locomotion from biomechanics principles is a viable approach. In practice, it would be desirable to use our policy as a starting point and improve the motion quality by further training with motion examples or additional reward terms. For example, we took the network of the biped walking policy to warm-start another policy learning session, in which the character learns to walk with knees raising up high (Figure 12 TOP). We also warm-started from the humanoid running policy to obtain a controller using a bigger stride length (Figure 12 BOTTOM). Because the starting policy is already capable of balancing and moving forward, the refinement learning takes only 200 iterations to train.

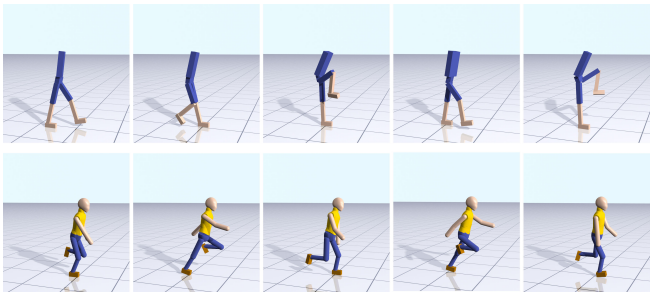

Fig. 12. Biped walking with high knees (TOP) and humanoid running with large stride length (BOTTOM) warm-started from our approach.

Our experiments show that both learner-centered and environment-centered curricula are effective in training locomotion controllers. The learner-centered curriculum is designed to bootstrap learning from the character's existing skill, but the curriculum might end up taking a long and winding path to reach the origin of the curriculum space. On the other hand, the environment-centered curriculum takes a straight path to the origin, but it presents the character with predetermined lessons and ignores the character's current skill, which might result in an overly aggressive curriculum. However, our scheduling algorithm for the environment-centered curriculum ensures that the lessons in two consecutive learning iterations overlap, reducing the chance of the character being thrown into a completely unfamiliar environment and failing immediately. While the learner-centered curriculum requires less hyper parameter tuning, we prefer the environment-centered curriculum for its data efficiency. The main design choice of the environment-centered curriculum lies in the formula of reduction from $\mathbf{x}_{begin}$ to $\mathbf{x}_{end}$. Our arbitrarily designed step function works well in all of our examples, but it may be possible to design a different reduction formula that can lead to an even more data-efficient learning algorithm.

One of the most encouraging results of this work is the emergence of different gaits for different target velocities. Most previous work obtains different gait patterns through contact planning or constraint enforcement [Coros et al. 2010; Ye and Liu 2010]. In contrast, with different target velocity $\hat{v}$, our characters learn speed-appropriate gaits using nearly identical reward functions (Table 1), without additional engineering effort.

In addition to the parameters of the reward function and hyper parameters in the network, the style of the motion also depends on the kinematic and dynamic properties of the character. A comprehensive sensitivity analysis is beyond the scope of this work. However, we notice that the resulting motion is particularly sensitive to the range of joint torques that each actuator is allowed to generate. In this paper, the torque range is set heuristically based on the perceived strength of each joint. Our preliminary results show that different locomotion gaits can result when different torque range settings. For example, we observed hopping behavior when we greatly increased the leg strength of the humanoid character.

There are many different ways to design the virtual assistant, and some are more effective than others. For example, we tried to provide lateral balance using two walls, one on each side of the character. The initial lesson presented a narrow passage between the two walls so that the character could not possibly fall. As the curriculum proceeded, we then widened the gap between the walls. The results showed that the character learned to intentionally lean on the wall to decrease the penalty of falling. As the gap widened, the character leaned even more until the character fell because the walls were too far apart. We have also attempted to provide propelling assistance through the use of a virtual treadmill. This experiment was unsuccessful even when learning the initial lesson $\mathbf{x}_0$. Our speculation is that the treadmill does not provide the necessary assistance for learning how to move forward; the learner still needs to put a significant amount of effort towards matching the speed of moving ground. In other words, it is arguable that learning with assistance ($\mathbf{x}_0$) is just as difficult to learn as without assistance.

8 CONCLUSION

We have demonstrated a reinforcement learning approach for creating low-energy, symmetric, and speed-appropriate locomotion gaits. One element of this approach is to provide virtual assistance to help the character learn to balance and to reach a target speed. The second element is to encourage symmetric behavior through the use of an additional loss term. When used together, these two techniques provide a method of automatically creating locomotion controllers for arbitrary character body plans. We tested our method on the lower half of a biped, a full humanoid, a quadruped, an a hexapod and demonstrated learning of locomotion gaits that resemble the natural motions of humans and animals, without prior knowledge about the motion. Because our method generalizes to other body plans, an animator can create locomotion controllers for characters and creatures for which there is no existing motion data.

Although our characters demonstrate more natural locomotion gaits comparing to existing work in DRL, the quality of the motion is still not on a par with previous work in computer animation that

exploits real-world data. Further investigation on how to incorporate motion capture data, biological-based modeling, and policy refinement (see Figure 12) is needed. In addition, our work is only evaluated on terrestrial locomotion with characters represented by articulated rigid bodies. One possible future direction is to apply the curriculum learning to other types of locomotion, such as swimming, flying, or soft-body locomotion.

ACKNOWLEDGMENTS

We thank Charles C. Kemp, Jie Tan, Ariel Kapusta, Alex Clegg, Zackory Erickson and Henry M. Clever for the helpful discussions. This work was supported by NSF award IIS-1514258 and AWS Cloud Credits for Research.

REFERENCES

Mazen Al Borno, Martin De Lasa, and Aaron Hertzmann. 2013. Trajectory optimization for full-body movements with complex contacts. *IEEE transactions on visualization and computer graphics* 19, 8 (2013), 1405–1414.

Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. 2009. Curriculum learning. In *Proceedings of the 26th annual international conference on machine learning*. ACM, 41–48.

Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. 2016. OpenAI Gym. *CoRR* abs/1606.01540 (2016). arXiv:1606.01540 http://arxiv.org/abs/1606.01540

Stelian Coros, Philippe Beaudoin, and Michiel van de Panne. 2010. Generalized Biped Walking Control. In *ACM SIGGRAPH 2010 Papers (SIGGRAPH '10)*. ACM, New York, NY, USA, Article 130, 9 pages. https://doi.org/10.1145/1833349.1781156

Stelian Coros, Andrej Karpathy, Ben Jones, Lionel Reveret, and Michiel van de Panne. 2011. Locomotion Skills for Simulated Quadrupeds. *ACM Transactions on Graphics* 30, 4 (2011), Article TBD.

Marco da Silva, Yeuhi Abe, and Jovan Popović. 2008. Interactive Simulation of Stylized Human Locomotion. *ACM Trans. Graph.* 27, 3, Article 82 (Aug. 2008), 10 pages. https://doi.org/10.1145/1360612.1360681

Martin de Lasa, Igor Mordatch, and Aaron Hertzmann. 2010. Feature-based Locomotion Controllers. *ACM Trans. Graph.* 29, 4, Article 131 (July 2010), 10 pages. https://doi.org/10.1145/1778765.1781157

Prafulla Dhariwal, Christopher Hesse, Matthias Plappert, Alec Radford, John Schulman, Szymon Sidor, and Yuhuai Wu. 2017. OpenAI Baselines. https://github.com/openai/baselines. (2017).

Yan Duan, Xi Chen, Rein Houthooft, John Schulman, and Pieter Abbeel. 2016. Benchmarking deep reinforcement learning for continuous control. In *International Conference on Machine Learning*. 1329–1338.

Martin L Felis and Katja Mombaur. 2016. Synthesis of full-body 3-D human gait using optimal control methods. In *Robotics and Automation (ICRA), 2016 IEEE International Conference on*. IEEE, 1560–1566.

Carlos Florensa, David Held, Markus Wulfmeier, and Pieter Abbeel. 2017. Reverse curriculum generation for reinforcement learning. *arXiv preprint arXiv:1707.05300* (2017).

T. Geijtenbeek and N. Pronost. 2012. Interactive Character Animation Using Simulated Physics: A State-of-the-Art Review. *Comput. Graph. Forum* 31, 8 (Dec. 2012), 2492–2515. https://doi.org/10.1111/j.1467-8659.2012.03189.x

Thomas Geijtenbeek, Michiel van de Panne, and A. Frank van der Stappen. 2013. Flexible Muscle-based Locomotion for Bipedal Creatures. *ACM Trans. Graph.* 32, 6, Article 206 (Nov. 2013), 11 pages. https://doi.org/10.1145/2508363.2508399

Alex Graves, Marc G Bellemare, Jacob Menick, Remi Munos, and Koray Kavukcuoglu. 2017. Automated Curriculum Learning for Neural Networks. *arXiv preprint arXiv:1704.03003* (2017).

Sehoon Ha. 2016. PyDart2. (2016). https://github.com/sehoonha/pydart2

Sehoon Ha and C Karen Liu. 2014. Iterative training of dynamic skills inspired by human coaching techniques. *ACM Transactions on Graphics* 34, 1 (2014).

Nikolaus Hansen and Andreas Ostermeier. 1996. Adapting arbitrary normal mutation distributions in evolution strategies: The covariance matrix adaptation. In *Evolutionary Computation, 1996., Proceedings of IEEE International Conference on*. IEEE, 312–317.

Nicolas Heess, Srinivasan Sriram, Jay Lemmon, Josh Merel, Greg Wayne, Yuval Tassa, Tom Erez, Ziyu Wang, Ali Eslami, Martin Riedmiller, et al. 2017. Emergence of locomotion behaviours in rich environments. *arXiv preprint arXiv:1707.02286* (2017).

David Held, Xinyang Geng, Carlos Florensa, and Pieter Abbeel. 2017. Automatic Goal Generation for Reinforcement Learning Agents. *arXiv preprint arXiv:1705.06366* (2017).

WALTER Herzog, Benno M Nigg, LYNDA J Read, and EWA Olsson. 1989. Asymmetries in ground reaction force patterns in normal human gait. *Med Sci Sports Exerc* 21, 1 (1989), 110–114.

Jessica K. Hodgins, Wayne L. Wooten, David C. Brogan, and James F. O'Brien. 1995. Animating Human Athletics. In *Proceedings of the 22Nd Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH '95)*. ACM, New York, NY, USA, 71–78. https://doi.org/10.1145/218380.218414

Sumit Jain, Yuting Ye, and C. Karen Liu. 2009. Optimization-Based Interactive Motion Synthesis. *ACM Transaction on Graphics* 28, 1 (2009), 1–10.

Andrej Karpathy and Michiel Van De Panne. 2012. Curriculum learning for motor skills. In *Canadian Conference on Artificial Intelligence*. Springer, 325–330.

Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. 2015. Deep learning. *Nature* 521, 7553 (2015), 436–444.

Yoonsang Lee, Sungeun Kim, and Jehee Lee. 2010. Data-driven biped control. *ACM Transactions on Graphics (TOG)* 29, 4 (2010), 129.

Yoonsang Lee, Moon Seok Park, Taesoo Kwon, and Jehee Lee. 2014. Locomotion Control for Many-muscle Humanoids. *ACM Trans. Graph.* 33, 6, Article 218 (Nov. 2014), 11 pages. https://doi.org/10.1145/2661229.2661233

Sergey Levine and Vladlen Koltun. 2014. Learning Complex Neural Network Policies with Trajectory Optimization. In *ICML '14: Proceedings of the 31st International Conference on Machine Learning*.

Timothy P Lillicrap, Jonathan J Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. 2015. Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971* (2015).

C. Karen Liu, Aaron Hertzmann, and Zoran Popović. 2005. Learning Physics-Based Motion Style with Nonlinear Inverse Optimization. *ACM Transactions on Graphics* 24, 3 (July 2005), 1071–1081.

C. Karen Liu and Sumit Jain. 2012. A Short Tutorial on Multibody Dynamics. *Tech. Rep. GIT-GVU-15-01-1, Georgia Institute of Technology, School of Interactive Computing* (2012). http://dartsim.github.io/

Libin Liu and Jessica Hodgins. 2017. Learning to schedule control fragments for physics-based characters using deep q-learning. *ACM Transactions on Graphics (TOG)* 36, 3 (2017), 29.

Libin Liu, Michiel Van De Panne, and Kangkang Yin. 2016. Guided Learning of Control Graphs for Physics-Based Characters. *ACM Trans. Graph.* 35, 3, Article 29 (May 2016), 14 pages. https://doi.org/10.1145/2893476

Tambet Matiisen, Avital Oliver, Taco Cohen, and John Schulman. 2017. Teacher-Student Curriculum Learning. *arXiv preprint arXiv:1707.00183* (2017).

Volodymyr Mnih, Adria Puigdomenech Badia, Mehdi Mirza, Alex Graves, Timothy Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. 2016. Asynchronous methods for deep reinforcement learning. In *International Conference on Machine Learning*. 1928–1937.

Igor Mordatch, Kendall Lowrey, Galen Andrew, Zoran Popovic, and Emanuel V Todorov. 2015. Interactive control of diverse complex characters with neural networks. In *Advances in Neural Information Processing Systems*. 3132–3140.

Igor Mordatch, Emanuel Todorov, and Zoran Popović. 2012. Discovery of Complex Behaviors Through Contact-invariant Optimization. *ACM Trans. Graph.* 31, 4, Article 43 (July 2012), 8 pages. https://doi.org/10.1145/2185520.2185539

Igor Mordatch, Jack M. Wang, Emanuel Todorov, and Vladlen Koltun. 2013. Animating Human Lower Limbs Using Contact-invariant Optimization. *ACM Trans. Graph.* 32, 6, Article 203 (Nov. 2013), 8 pages. https://doi.org/10.1145/2508363.2508365

Uldarico Muico, Yongjoon Lee, Jovan Popović, and Zoran Popović. 2009. Contact-aware Nonlinear Control of Dynamic Characters. *ACM Trans. Graph.* 28, 3, Article 81 (July 2009), 9 pages. https://doi.org/10.1145/1531326.1531387

Sanmit Narvekar, Jivko Sinapov, Matteo Leonetti, and Peter Stone. 2016. Source task creation for curriculum learning. In *Proceedings of the 2016 International Conference on Autonomous Agents & Multiagent Systems*. International Foundation for Autonomous Agents and Multiagent Systems, 566–574.

B Nigg, R Robinson, and W Herzog. 1987. Use of Force Platform Variables to Ouantify the Effects of Chiropractic Manipulation on Gait Symmetry. *Journal of manipulative and physiological therapeutics* 10, 4 (1987).

Openai. 2017. openai/roboschool. (2017). https://github.com/openai/roboschool

Kara K Patterson, Iwona Parafianowicz, Cynthia J Danells, Valerie Closson, Mary C Verrier, W Richard Staines, Sandra E Black, and William E McIlroy. 2008. Gait asymmetry in community-ambulating stroke survivors. *Archives of physical medicine and rehabilitation* 89, 2 (2008), 304–310.

Xue Bin Peng, Glen Berseth, and Michiel van de Panne. 2015. Dynamic Terrain Traversal Skills Using Reinforcement Learning. *ACM Trans. Graph.* 34, 4, Article 80 (July 2015), 11 pages. https://doi.org/10.1145/2766910

Xue Bin Peng, Glen Berseth, and Michiel van de Panne. 2016. Terrain-adaptive Locomotion Skills Using Deep Reinforcement Learning. *ACM Trans. Graph.* 35, 4, Article 81 (July 2016), 12 pages. https://doi.org/10.1145/2897824.2925881

Xue Bin Peng, Glen Berseth, Kangkang Yin, and Michiel Van De Panne. 2017. DeepLoco: Dynamic Locomotion Skills Using Hierarchical Deep Reinforcement Learning. *ACM Trans. Graph.* 36, 4, Article 41 (July 2017), 13 pages. https://doi.org/10.1145/3072959.3073602

Anastasia Pentina, Viktoriia Sharmanska, and Christoph H Lampert. 2015. Curriculum learning of multiple tasks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 5492–5500.

Lerrel Pinto and Abhinav Gupta. 2016. Supersizing self-supervision: Learning to grasp from 50k tries and 700 robot hours. In *Robotics and Automation (ICRA), 2016 IEEE International Conference on*. IEEE, 3406–3413.

John Schulman, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz. 2015a. Trust region policy optimization. In *Proceedings of the 32nd International Conference on Machine Learning (ICML-15)*. 1889–1897.

John Schulman, Philipp Moritz, Sergey Levine, Michael Jordan, and Pieter Abbeel. 2015b. High-dimensional continuous control using generalized advantage estimation. *arXiv preprint arXiv:1506.02438* (2015).

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347* (2017).

Kwang Won Sok, Manmyung Kim, and Jehee Lee. 2007. Simulating Biped Behaviors from Human Motion Data. In *ACM SIGGRAPH 2007 Papers (SIGGRAPH '07)*. ACM, New York, NY, USA, Article 107. https://doi.org/10.1145/1275808.1276511

Richard S Sutton, David A McAllester, Satinder P Singh, and Yishay Mansour. 2000. Policy gradient methods for reinforcement learning with function approximation. In *Advances in neural information processing systems*. 1057–1063.

Jie Tan, Yuting Gu, C. Karen Liu, and Greg Turk. 2014. Learning Bicycle Stunts. *ACM Trans. Graph.* 33, 4, Article 50 (July 2014), 12 pages. https://doi.org/10.1145/2601097.2601121

Jie Tan, Karen Liu, and Greg Turk. 2011. Stable proportional-derivative controllers. *IEEE Computer Graphics and Applications* 31, 4 (2011), 34–44.

Michiel Van de Panne and Alexis Lamouret. 1995. Guided optimization for balanced locomotion. In *Computer animation and simulation*, Vol. 95. Springer, 165–177.

Kevin Wampler and Zoran Popović. 2009. Optimal Gait and Form for Animal Locomotion. In *ACM SIGGRAPH 2009 Papers (SIGGRAPH '09)*. ACM, New York, NY, USA, Article 60, 8 pages. https://doi.org/10.1145/1576246.1531366

Kevin Wampler, Zoran Popović, and Jovan Popović. 2014. Generalizing Locomotion Style to New Animals with Inverse Optimal Regression. *ACM Trans. Graph.* 33, 4, Article 49 (July 2014), 11 pages. https://doi.org/10.1145/2601097.2601192

Jack M. Wang, David J. Fleet, and Aaron Hertzmann. 2009. Optimizing Walking Controllers. *ACM Trans. Graph.* 28, 5, Article 168 (Dec. 2009), 8 pages. https://doi.org/10.1145/1618452.1618514

Jack M. Wang, Samuel R. Hamner, Scott L. Delp, and Vladlen Koltun. 2012. Optimizing Locomotion Controllers Using Biologically-based Actuators and Objectives. *ACM Trans. Graph.* 31, 4, Article 25 (July 2012), 11 pages. https://doi.org/10.1145/2185520.2185521

Jungdam Won, Jongho Park, Kwanyu Kim, and Jehee Lee. 2017. How to train your dragon: example-guided control of flapping flight. *ACM Transactions on Graphics (TOG)* 36, 6 (2017), 198.

Jia-chi Wu and Zoran Popović. 2010. Terrain-adaptive bipedal locomotion control. *ACM Transactions on Graphics (TOG)* 29, 4 (2010), 72.

Yuting Ye and C. Karen Liu. 2010. Optimal Feedback Control for Character Animation Using an Abstract Model. *ACM Trans. Graph.* 29, 4, Article 74 (July 2010), 9 pages. https://doi.org/10.1145/1778765.1778811

KangKang Yin, Stelian Coros, Philippe Beaudoin, and Michiel van de Panne. 2008. Continuation methods for adapting simulated skills. In *ACM Transactions on Graphics (TOG)*, Vol. 27. ACM, 81.

KangKang Yin, Kevin Loken, and Michiel van de Panne. 2007. SIMBICON: Simple Biped Locomotion Control. *ACM Trans. Graph.* 26, 3, Article 105 (July 2007). https://doi.org/10.1145/1276377.1276509