Efficient Hyperparameter Optimization for Physics-based Character Animation

ZESHI YANG, Simon Fraser University
ZHIQI YIN, Simon Fraser University

Physics-based character animation has seen significant advances in recent years with the adoption of Deep Reinforcement Learning (DRL). However, DRL-based learning methods are usually computationally expensive and their performance crucially depends on the choice of hyperparameters. Tuning hyperparameters for these methods often requires repetitive training of control policies, which is even more computationally prohibitive. In this work, we propose a novel Curriculum-based Multi-Fidelity Bayesian Optimization framework (CMFBO) for efficient hyperparameter optimization of DRL-based character control systems. Using curriculum-based task difficulty as fidelity criterion, our method improves searching efficiency by gradually pruning search space through evaluation on easier motor skill tasks. We evaluate our method on two physics-based character control tasks: character morphology optimization and hyperparameter tuning of DeepMimic. Our algorithm significantly outperforms state-of-the-art hyperparameter optimization methods applicable for physics-based character animation. In particular, we show that hyperparameters optimized through our algorithm result in at least 5x efficiency gain comparing to author-released settings in DeepMimic.

CCS Concepts: • **Computing methodologies** → **Animation**; *Physical simulation*; • **Theory of computation** → *Reinforcement learning*; *Bayesian analysis*.

Additional Key Words and Phrases: Physics-based Character Animation, Bayesian Optimization, Reinforcement Learning, Curriculum Learning

1 INTRODUCTION

Physics-based character animation has made significant progresses recently, especially with the application of Deep Reinforcement Learning (DRL) algorithms [Bergamin et al. 2019; Park et al. 2019; Peng et al. 2018a; Won et al. 2020]. For example, DeepMimic-style neural network controllers are able to synthesize diverse and robust high-quality motor skills [Peng et al. 2018a]. Despite the demonstrated impressive performance, it is usually quite hard for a novice graduate student to reproduce the performance of such systems, without knowing the exact value of each hyperparameter involved. Therefore, more and more authors have released their code for better reproducibility. However, a single change of one hyperparameter may totally ruin the performance, and sometimes even the convergence, of such learning algorithms. How the original authors found the working set of hyperparameters remains as an art and mystery. Improving upon prior work, even with released code, thus remains challenging, as any modification to the training algorithm may require a new set of hyperparameters to work well.

Automatic hyperparameter optimization is thus in great need. Hyperparameters, in the narrow sense, refer to values that are used to control the learning process in machine learning algorithms.

Authors' addresses: Zeshi Yang, Simon Fraser University, zeshiy@sfu.ca; Zhiqi Yin, Simon Fraser University, zhiqiy@sfu.ca.

In contrast, regular parameters are derived or optimized during the training process. In this paper, we refer to all parameters external to a learning algorithm that need to be determined prior to the learning as hyperparameters. For example, morphology parameters of a virtual character in motor learning are also hyperparameters. To date, it is a common practice of the field to manually test and select hyperparameters for various character control algorithms. Behind the scenes, maybe simple grid search or random search algorithms are used for semi-automatically choosing hyperparameters. However, in physics-based character control, better search schemes are needed to handle two key challenges: First, the evaluation of new hyperparameters typically involves re-learning of the controllers from scratch, which is usually an expensive black-box function. Second, the number of hyperparameters can also be large and results in the curse of dimensionality.

Bayesian optimization (BO) is a promising candidate for hyperparameter optimizations for physics-based character animation problems. BO is a sequential design strategy for global optimization of expensive-to-evaluate black-box functions that do not assume any functional forms [Jones et al. 1998; Kandasamy et al. 2017; Srinivas et al. 2010]. Traditional BO only evaluates the expensive black-box objective functions themselves [Jones et al. 1998; Srinivas et al. 2010]. We refer to such algorithms as Single Fidelity BO (SFBO) in this paper. For many physics-based character animation applications, however, SFBO is inefficient due to the extremely high cost of function evaluations that rely on physics-based character simulation and motor learning. Multi-fidelity BO (MFBO) accelerates the optimization by using cheap approximations of the objective functions in early optimization stages [Kandasamy et al. 2017; Klein et al. 2017; Song et al. 2019; Swersky et al. 2013]. Such MFBO algorithms typically employ fewer training iterations on smaller datasets as low-fidelity cheap approximations to the original objective functions, and work well for supervised learning tasks.

For physics-based character animation, however, existing MFBO methods do not work well. We analyze in section 3.2 and 3.3 that training iterations are not a good fidelity criterion for physics-based character control, and multi-fidelity objective does not have desired properties across fidelity dimension required by existing MFBO methods like BOCA [Kandasamy et al. 2017] and FABOLAS [Klein et al. 2017]. Therefore, we propose a novel algorithm CMFBO: Curriculum-based Multi-Fidelity Bayesian Optimization for efficient hyperparameter optimization of DRL-based character control systems. CMFBO employs easier motor skill learning tasks as low-fidelity optimization objectives. Task difficulties are organized and scheduled by a curriculum [Bengio et al. 2009]. For example, a curriculum that gradually increases the training episode length can help the character perform longer and longer skills [Kostrikov 2018; Peng et al. 2018a]; and a curriculum that gradually reduces the hand-of-God assistance forces can help the character learn to locomote [Yu et al. 2018]. Easier tasks are faster to evaluate enabling efficient search space pruning. Control policies learned at easier tasks could be transferred to harder tasks to further reduce evaluation cost. Hyperparameters optimized by CMFBO may significantly outperform those tuned by experts and optimized by state-of-the-art methods. For instance, at least 5x efficiency gain is obtained with our optimized hyperparameters on DeepMimic [Peng et al. 2018a], compared to author-released settings.

To summarize, we

- introduce Bayesian Optimization into physics-based character animation for principled hyperparameter optimization;
- propose an efficient algorithm CMFBO for hyperparameter optimization of challenging motor learning tasks based on deep reinforcement learning;
- systematically compare and evaluate SFBO, state-of-the-art MFBO, and CMFBO on two physics-based character animation problems: morphology optimization, and automatic tuning of DeepMimic training hyperparameters.

2 RELATED WORK

2.1 Physics-based Character Animation

Synthesizing natural human motions is a long-standing challenge in computer graphics. We classify existing methods roughly into three categories. First, manually designed controllers, which usually employ finite state machines (FSMs) and heuristic feedback rules. Human insights and domain knowledge are usually involved to design and tune the parameters [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]. Second, model-based trajectory optimization, where equations of motions are enforced as optimization constraints. Designing of an often sophisticated optimization objective function is generally required along with weights tuning for each term [Hämäläinen et al. 2015; Mordatch et al. 2012; Tassa et al. 2012; Wampler et al. 2014]. Recently, model-free DRL-based control methods have been demonstrated to reproduce high-quality complex and robust motor skills [Bergamin et al. 2019; Heess et al. 2017; Lillicrap et al. 2016; Luo et al. 2020; Merel et al. 2020; Park et al. 2019; Peng et al. 2018a, 2015, 2016, 2017, 2018b; Won et al. 2020; Won and Lee 2019; Yu et al. 2018]. DRL-based methods require designing and tuning of a reward function, as well as related hyperparameters of DRL algorithms themselves. In this paper, We focus on hyperparameter optimization for recent DRL-based methods, although our framework is general enough to be applicable to other categories of methods as well.

2.2 Hyperparameter Optimization

2.2.1 Parameter Optimization in Computer Graphics. Parameter tuning and optimization is a common task in computer graphics, such as weights for SIMBICON-type feedback controllers [Wang et al. 2009; Yin et al. 2007]. Multiple derivative-free optimization algorithms, such as Covariance Matrix Adaptation (CMA) [Hansen 2006; Wang et al. 2009] and Bayesian optimization, have been used to optimize such parameters where the objective functions are black-box functions and no derivative information is available. Some methods further combine BO with user evaluations to achieve desired visual effects, such as bidirectional reflectance distribution function (BRDF) design [Brochu et al. 2007] and smoke animation tuning [Brochu et al. 2010]. More recently, BO is applied to low-dimensional subproblems of various graphics applications [Koyama et al. 2020, 2017], such as color enhancement, and human body geometric modelling [Loper et al. 2015].

We also adopt a Bayesian optimization framework, but for hyperparameter optimization. To the best of our knowledge, the only previous work on hyperparameter optimization in graphics is [Tseng et al. 2019], where parameters of image processing hardware, such as threshold values in the denoising module, are optimized with a non-Bayesian approach. Hereafter we will focus on hyperparameter optimization literature from machine learning and robotics.

2.2.2 Hyperparameter Optimization for DRL. State-of-the-art motor learning methods are usually based on DRL. For example, DeepMimic [Peng et al. 2018a] can produce robust and diverse high-quality controllers by imitating motion captured reference motions. However, its performance crucially depends on the choice of its hyperparameters. Minor changes to one single hyperparameter may result in slow convergence or even training failure. But so far, no publications on physics-based character animation have explicitly discussed the issue of hyperparameter tuning or optimization. In machine learning, hyperparameter optimization for simple benchmark systems, such as a cartpole model in OpenAI gym [Brockman et al. 2016], with standard DRL algorithms, such as A2C [Mnih et al. 2016], has been investigated [Nguyen and Osborne 2019; Nguyen et al. 2020]. However, in physics-based character animation, we study the control of high dimensional character models, which is a much more expensive and challenging task requiring specialized DRL algorithms such as DeepMimic to achieve natural-looking skills.

2.2.3 Morphology Design. Morphology design and optimization is a problem in both computer animation and robotics [Agrawal et al. 2014; Bongard 2011; Geijtenbeek et al. 2013; Ha et al. 2017; Hu et al. 2020; Huang et al. 2020; Liao et al. 2019; Lipson and Pollack 2000; Ma et al. 2021; Park and Asada 1994; Paul and Bongard 2001; Pil and Asada 1996; Sims 1994; Spielberg et al. 2017; Villarreal-Cervantes et al. 2012; Wang et al. 2018; Won and Lee 2019]. Traditional model-based methods require accurate dynamic models, and only work for specific types of control algorithms such as trajectory optimization [Ha et al. 2017; Spielberg et al. 2017]. Some recent morphology design methods work with DRL-based control [Ha 2019; Luck et al. 2020; Schaff et al. 2019]. These methods guide optimization in morphology space based on gradient estimation [Ha 2019; Schaff et al. 2019], or performance prediction of unseen morphology through a learned value network [Luck et al. 2020]. We note that none of these methods address the issue of expensive evaluation per design. With effective search space pruning through cheap evaluation on easier tasks, our method significantly outperforms them in terms of sample efficiency.

2.3 Bayesian Optimization

Bayesian Optimization is a class of methods for expensive black-box function optimization. Since no gradient information is available, functions are optimized purely through evaluations. A Bayesian statistical model, usually a Gaussian Process [Rasmussen 2003], is maintained to estimate value of the objective function along with the uncertainty. An *acquisition function* is then repeatedly applied to query and evaluate the most promising and informative region based on existing estimation.

With the advancement of deep learning, there is a great appeal for automatic hyperparameter optimization of deep learning models. BO demonstrates its potential on such tasks for its promising sample efficiency [Nguyen and Osborne 2019; Snoek et al. 2012, 2015]. Recently many MFBO methods are proposed to further improve efficiency of hyperparameter optimization for supervised learning tasks [Kandasamy et al. 2016, 2017; Klein et al. 2017; Song et al. 2019; Swersky et al. 2013, 2014; Takeno et al. 2019] mainly by utilizing cheap approximations of the objective, such as neural network validation loss on smaller datasets with fewer training iterations.

However, most of these methods cannot scale well to motion control tasks. Due to stochasticity and complexity of DRL training, existing fidelity criterion such as the number of training iterations mislead the optimization a lot. Low-fidelity evaluations through early stopping usually provide false estimation on the relative performance of hyperparameters. Besides the fidelity criterion issue, existing MFBO acquisition functions have strong assumptions on the shape of multi-fidelity objective function along the fidelity dimension. For example, BOCA [Kandasamy et al. 2017] assumes a flat fidelity dimension while FABOLAS [Klein et al. 2017] assumes it to satisfy quadratic form. These simplified assumptions are no longer valid in the complex DRL settings. We detail the analysis and how we resolve the problems with our proposed curriculum-based fidelity and progressive acquisition function in section 3.2 and 3.3.

2.4 Curriculum Learning

Curriculum learning (CL) is a learning method where task difficulty gradually increases during training[Bengio et al. 2009]. Easier tasks are cheaper to accomplish and serve as good initial solutions for more difficult tasks. CL has been widely used in character animation to improve the performance and efficiency of motor learning tasks [Karpathy and Van De Panne 2012; Van de Panne and Lamouret 1995; Wu and Popović 2010; Yin et al. 2008]. [Kostrikov 2018; Peng et al. 2018a] employ a time-based curriculum with increasing episode length to help the character perform longer and longer skills. [Yu et al. 2018] applies hand-of-God assistance forces at robot torso and decreases the force gradually during training to encourage the emergence of natural gaits. More
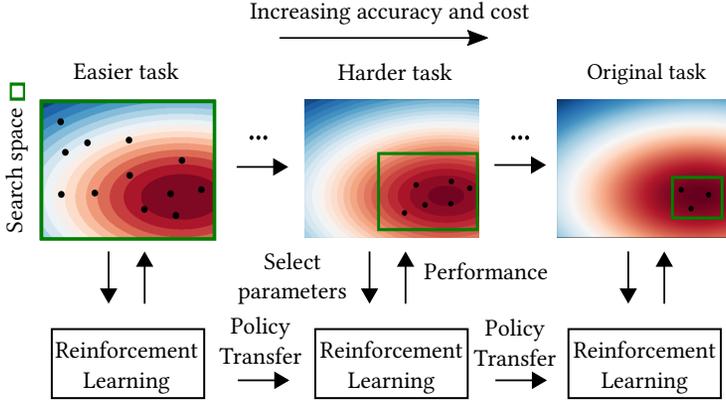
Fig. 1. Conceptual illustration of CMFBO
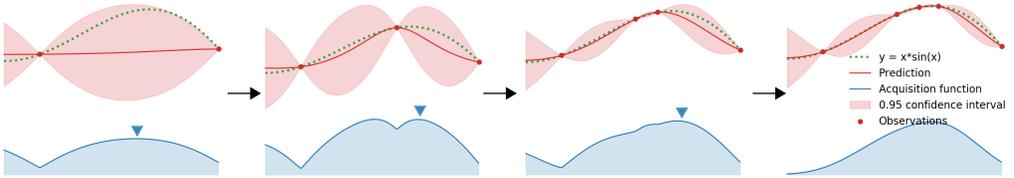
Fig. 2. An example of applying BO to a one-dimensional function. Green dotted curve represents the unknown black-box function. Red points represent the queried points. Red curve along with the rosy shaded region indicate the predicted mean and 95% confidence interval. Blue curve is the acquisition function (It is scaled for the convenience of visualization). Blue downside arrows indicate the maximum of the acquisition function which is the point to be queried next.

recently, [Xie et al. 2020] uses CL to demonstrate successful training of stepping-stone locomotion controllers.

3 METHOD

We formulate hyperparameter optimization in a physics-based animation system as a black-box function optimization problem

$$\arg\max_{x \in \mathcal{A}} f(x) \tag{1}$$

where $\mathcal{A}$ is the hyperparameter space. $f(\cdot) : \mathcal{A} \rightarrow \mathcal{R}$ is the objective function, i.e. performance of control policy given hyperparameter x. The evaluation of $f(\cdot)$ involves DRL training, thus is costly, noisy and cannot be computed in closed forms.

Figure 1 illustrates a conceptual overview of our approach based on MFBO. We first define our multi-fidelity objective function based on a curriculum (section 3.2). During optimization, we use lower-fidelity cheap approximations to locate the promising region of hyperparameters through the proposed progressive acquisition function (section 3.3). Policies learned at easier tasks are transferred to difficult tasks for more efficient evaluation (section 3.4).

3.1 Background

In this section we briefly review the technique of general Bayesian optimization involved in our approach. Given a black-box function $f(x)$, BO finds its maxima through repeated function evaluations. Since evaluating $f(x)$ could be expensive, BO is designed to minimize the number of function evaluations by querying the most promising and informative points. Given a set of current observations $D_t = \{(x_i, y_i)\}_{i=1}^t$, where y_i is a noisy measurement of $f(x_i)$, the *acquisition function* $a(x, D_t) : \mathcal{A} \to \mathbb{R}$ quantifies the utility of an arbitrary point. Maximizing the acquisition function will give us the point most worth trying next.

The acquisition function is designed for finding candidate points with both large values and rich information. Querying a point close to existing ones in D_t is less informative. Gaussian Process Upper Confidence Bound (GP-UCB) [Srinivas et al. 2010] is a popular acquisition function defined as:

$$a(x, D_t) = \mu_t(x) + \beta^{\frac{1}{2}}\sigma_t(x) \tag{2}$$

where $\mu_t(x)$ and $\sigma_t(x)$ are approximated posterior mean value and standard deviation of $f(x)$ respectively. $\mu_t(\cdot)$ favors candidates which are likely to have large values. $\sigma_t(\cdot)$ encourages querying informative points with high uncertainty. β enables a trade-off between exploitation and exploration.

Closed-form estimations for $\mu_t(\cdot)$ and $\sigma_t(\cdot)$ are available through a Gaussian Process (GP) [Rasmussen 2003] surrogate model of the objective function trained on D_t. A GP contains a prior mean function $m(x)$ and a kernel function $k(x, x')$. $m(x)$ encodes our prior belief of the objective function value. Kernel function $k(x, x')$ measures correlations between $f(x)$ and $f(x')$. Given $m(\cdot)$, $k(\cdot, \cdot)$ and existing observations D_t, μ_t and σ_t could be computed as:

$$\mu_t(x) = k(x, X)(K + \eta^2 I)^{-1}Y$$
$$\sigma_t^2(x) = k(x, x) + \eta^2 - k(x, X)(K + \eta^2 I)^{-1}k(X, x) \tag{3}$$

where $Y \in \mathbb{R}^t, Y_i = y_i; X \in \mathbb{R}^{t \times d}, X_i = x_i; K \in \mathbb{R}^{t \times t}, K_{i,j} = k(x_i, x_j); k(x, X) = (k(x, x_1), k(x, x_2), ...k(x, x_t));$ η is the standard deviation of the observation noise.

Many choices of kernel functions exist, including Square Exponential kernel and Matern kernel. In this work we adopt the Square Exponential kernel defined as:

$$k_{SE}(x, x') = \sigma_f^2 exp(-\frac{1}{2}(x - x')^T \Lambda^{-1}(x - x')) \tag{4}$$

$\Lambda = diag(\lambda_1, \lambda_2, ...\lambda t)$, where λ_i is the length scale of the i-th component of inputs. $f(\cdot)$ is flat across the i-th component of the input when λ_i is large.

Figure 2 shows a visual illustration of applying BO to a one-dimensional function. We refer interested readers to the second chapter of [Rasmussen 2003] for more in-depth reviews.

3.2 Curriculum-based Multi-Fidelity Functions

The main challenge of hyperparameter optimization for physics-based character control tasks comes from the fact that performance evaluation function $f(x)$ given hyperparameter x is highly computationally expensive. Even if BO is designed for black-box function optimization with minimum samples, the entire optimization procedure still requires a prohibitive amount of computational resources.

We propose to solve this problem with a multi-fidelity approach on top of Bayesian optimization, where low-fidelity cheap evaluations are utilized for efficient search space pruning. This procedure crucially depends on an accurate fidelity criterion being able to distinguish good hyperparameters from bad ones with lower-fidelity approximations. To the best of our knowledge, number of training iterations is the only fidelity criterion from existing literature applicable to our problem. Though number of training iterations shows its effectiveness for supervised learning tasks [Kandasamy
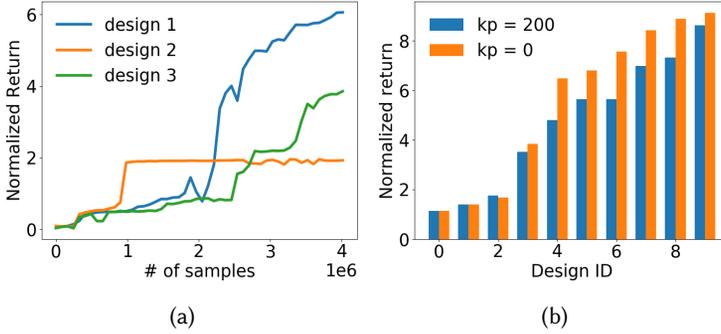
Fig. 3. (a): Learning curves of locomotion controllers with different morphology designs. (b): Performance of ten morphology designs on two difficulty levels. Task difficulty is determined by the strength of hand-of-God assistance forces parameterized by proportional gain *kp* of the assistance stable PD controller.

et al. 2017; Klein et al. 2017; Snoek et al. 2012], performance estimated through early stopping can be deceptive for challenging DRL character control tasks. Early stages for DRL training can be noisy [Fujimoto et al. 2018; Haarnoja et al. 2018], and good hyperparameters do not always guarantee quick convergence [Nguyen et al. 2020]. We illustrate this problem in the context of finding optimal morphology hyperparameters for character locomotion. We show the performance of three morphology designs in figure 3a, where some good morphology designs show worse performance in early training stages.

To mitigate this problem, we propose the use of curriculum-based task difficulty as a new fidelity criterion tailored to physics-based character control settings. Task difficulty can be parameterized by a single continuous scalar variable z. Multi-fidelity objective function is therefore defined to be $f(x, z)$. Note that $f(x, z_{max})$ is the original objective function. $f(x, z)$ is more accurate and requires more computational resources as z gets closer to z_{max}. Similar to single fidelity BO, we use GP to model multi-fidelity function $f(x, z)$. The kernel function generalizes to:

$$k((x, z), (x', z')) = k_{SE}(x, x') \, k_{SE}(z, z') \tag{5}$$

The factorized kernel adopts the multiplication form based on the fact that: if x and z are close to x' and z' respectively, $f(x, z)$ correlates strongly with $f(x', z')$.

We validate the effectiveness of using task difficulty as fidelity criterion through an example shown in figure 3b, where ten morphology hyperparameters are evaluated on two different levels of task difficulty parameterized by the strength of virtual hand-of-God assistance forces. With fidelity criterion defined with task difficulty, relative performance of hyperparameters across different fidelity levels are well preserved. We detail our specific curriculum settings for different tasks in section 4.

3.3 Progressive Acquisition Function

Given a properly designed multi-fidelity objective function, an effective acquisition function is still in need to achieve efficient hyperparameter optimization in physics-based character control tasks. In each iteration, multi-fidelity acquisition function $a((x, z), D_t)$ determines which point x in hyperparameter space is going to be evaluated next, and which fidelity level z the evaluation is going to be performed on. An effective multi-fidelity acquisition function is expected to generate queries such that the original objective can be optimized with much fewer samples by inferring optimality information through low-fidelity cheap approximations.

Existing MFBO methods like BOCA [Kandasamy et al. 2017] and FABOLAS [Klein et al. 2017] require a well-fitted surrogate model for accurate estimation of original objective value from lower-fidelity approximations. Mathematical assumptions on the shape of fidelity dimension are usually necessary to achieve this. For example, BOCA assumes the fidelity dimension is flat while FABOLAS assumes it to satisfy quadratic form. However, due to stochasticity and complexity of DRL training, performance difference of hyperparameters on different fidelity levels can be unpredictable, as shown in figure 3b. A GP trained on such DRL performance data will learn a small length scale, which degrades the accuracy of high fidelity value prediction through a reasonable number of lower-fidelity samples.

We attempt this problem with a different approach. Instead of relying on low fidelity samples to infer high-fidelity function value, we propose to use them simply for locating promising regions. The motivation is based on the key observation that though performance difference of hyperparameters on different fidelity levels can be unpredictable, optimal regions of objective function on different fidelity levels highly overlap, as shown in figure 3b. Search space for a high-fidelity objective can therefore be greatly pruned by simply exploring around optimal regions estimated at lower fidelity levels.

Specifically, we propose a novel progressive acquisition function, which progressively conducts optimization from low fidelity to high fidelity levels. At iteration t, target fidelity z_t is chosen to be the least expensive while most informative one until we reach the highest. This is performed through an iterative procedure. Initializing z_t with z_{min}, we repeatedly increase z_t by a small step α and re-compute best candidate point x_t for new z_t. The procedure terminates when our model shows insufficient knowledge at (x_t, z_t) with uncertainty $\sigma_t(x_t, z_t)$ exceeding threshold ϵ. In our implementation, we choose α to be equal to the length scale of the fidelity kernel l_z, since function values from GP surrogate model strongly correlate along fidelity dimension within l_z proximity. z_t should be increased only when we have sufficient knowledge at (x_t, z_t). ϵ can therefore be set conservatively to a small value negligible comparing to the objective function value range in promising regions. In our experiments we set ϵ to 0.02.

Given a specified fidelity level z_t, the best candidate point x_t is chosen by optimizing a variant of the upper confidence bound (UCB) utility function defined as:

$$a((x, z_t), D_t) = \mu_t(x, z_t) + \beta^{\frac{1}{2}} \sigma_t(x, z_{max}) \qquad (6)$$

where $\mu_t(x, z_t)$ favors high-performing regions and $\sigma_t(x, z_{max})$ favors uncertain regions of the original objective. We set β to $0.2d \log(2t)$ following suggestions from [Kandasamy et al. 2017; Srinivas et al. 2010], where d is the dimension of the hyperparameters and t is the iteration index. The optimization starts from $z_t = z_{min}$ to find initial promising candidates across the hyperparameter space. When $z_t > z_{min}$, optimization of $a((x, z_t), D_t)$ is restricted to be close to known high-utility regions obtained from lower-fidelity levels, which enables efficient search space pruning. This can be achieved by simply applying a gradient-based local optimization method like L-BFGS [Liu and Nocedal 1989] on $a((x, z_t), D_t)$, with solutions of $\arg \max a(\cdot)$ at lower-fidelity levels serving as initial guesses. The optimization of the acquisition function does not have to be perfect, since we only care about promising points instead of optimal ones. Our algorithm works well with our default choices for α, β and ϵ without manual tuning. We summarize this procedure in algorithm 1 for ease of re-implementation.

3.4 Policy Transfer

Progressive acquisition function enables the use of transfer learning to speed up evaluation on most (x_t, z_t) pairs where $z_t > z_{min}$. This comes from the fact that progressive acquisition function queries z_t progressively, and only looks for x_t within a common promising region. Therefore, as

long as $z_t > z_{min}$, a previously queried pair (x_i, z_i) with $i < t$ usually (if not always) exists such that (x_t, z_t) is close to (x_i, z_i). Pre-trained policy on (x_i, z_i) can therefore serve as a warm start for training on (x_t, z_t).

In practice, (x_i, z_i) can be acquired by traversing all previously visited (x, z) pairs to find the closest one to (x_t, z_t) measured by kernel function correlation $k((x_t, z_t), (x_i, z_i))$. Theoretically, negative transfer may happen but we do not observe any in our experiments. We show in our experiments that the use of transfer learning saves a significant number of samples during CMFBO optimization.

Algorithm 2 summarizes the pipeline of CMFBO.

Algorithm 1: Progressive Acquisition Function

Input: Iteration index t, fidelity kernel length scale l_z, observation dataset D_t and uncertainty threshold ϵ

Output: Best candidate pair (x_t, z_t)

$z_t \leftarrow z_{min}$

$x_t \leftarrow \arg\max_x a((x, z_t), D_t)$ using L-BFGS with multiple random start points

while $z_t < z_{max}$ and $\sigma_t(x_t, z_t) < \epsilon$ **do**
  1.$z_t \leftarrow \min(z_t + l_z, z_{max})$;
  2.$x_t \leftarrow \arg\max_x a((x, z_t), D_t)$ using L-BFGS with x_t as an initial solution ;

Return (x_t, z_t)

Algorithm 2: Curriculum-based MFBO

Input: Multi-fidelity function $f(x, z)$ and the maximum allowed computational cost N_{total}

Output: Observed function maximum $y^\star$ and corresponding $x^\star$

$N_{current} \leftarrow 0, D_0 \leftarrow \varnothing$, the Gaussian Process is initialized randomly.

for $t = 1, 2, ...$ **do**
  1.Choose (x_t, z_t) by progressive acquisition function optimization (see algorithm 1);
  2.Evaluate $f(x_t, z_t)$ to get performance y_t and corresponding computational cost N_t;
  3.$D_t \leftarrow D_{t-1} \cup ((x_t, z_t), y_t)$ and update parameters of the Gaussian Process using D_t;
  4.$N_{current} \leftarrow N_{current} + N_t$;
  5.**if** $N_{current} \geq N_{total}$ **then**
    | break;

$t^\star \leftarrow \arg\max_t y_t, y^\star \leftarrow y_{t\star}, x^\star \leftarrow x_{t\star}$;

Return $y^\star$ and $x^\star$

---

4 EXPERIMENTS

We validate CMFBO on two physics-based character control tasks: morphology optimization and hyperparameter optimization of DeepMimic. For each task, we discuss its specific curriculum setting and objective function, followed by experiment results, comparisons and ablation studies.

In our implementation, we use Gaussian Process from GPy[GPy 2012] and L-BFGS [Liu and Nocedal 1989] from Scipy [Virtanen et al. 2020]. All physics simulation is performed on PyBullet[Coumans and Bai 2019]. We report the performance statistics on a desktop with i7-7800x CPU (12 threads). Each experiment is run 3 times independently with different random seeds.

4.1 Morphology Optimization

The locomotion capability of a creature is intimately coupled with its morphology. Elaborately designed morphology of simulated characters encourages the emergence of natural locomotion [Geijtenbeek et al. 2013]. In this experiment, we optimize character morphology for learning fast and low-energy locomotion. Our simulated character has 15 torque-controlled revolute joints. Morphology hyperparameter x is defined to be a 12-dimensional vector that scales the length and radius of character links within a limited range, as shown in table 1. Link mass is uniformly scaled according to its volume. Sagittal symmetry is imposed on legs of the character.

4.1.1 Environment Setup. We designed a DRL-based character control environment tailored to fast locomotion, based on humanoid environments in [Coumans and Bai 2019]. Character state is defined to be orientation and linear velocities of the root link, along with angles and velocities for all joints. Action is normalized joint torques. We adopt a reward function specialized for locomotion control proposed in [Xie et al. 2020], except that task specific and velocity penalty terms are removed to encourage faster locomotion. Readers can refer to [Xie et al. 2020] for more details of the reward function. We further adopt the symmetry loss proposed in [Yu et al. 2018] to encourage symmetric gaits.

We use a feed-forward policy network with three fully connected layers, each with 128 units using tanh activation. The critic network shares the same architecture with the policy. Policy is trained with Proximal Policy Optimization (PPO) [Schulman et al. 2017]. We set discount factor $\gamma = 0.95$ and $\lambda = 0.95$ for both $TD(\lambda)$ and $GAE(\lambda)$ [Schulman et al. 2015]. Learning rates of both policy and critic network are set to 3×10^{-4}. In each training iteration, we sample 4096 state-action tuples with 10 paralleled environments. Batch size for policy update is set to 256.

4.1.2 Curriculum and Multi-fidelity Objective. We construct our multi-fidelity function based on a curriculum learning setting proposed in [Yu et al. 2018], where a stable proportional-derivative (SPD) controller [Tan et al. 2011] is applied to the character root to provide hand-of-God balancing forces. Task difficulty is parameterized by stiffness kp and damping kd coefficients of SPD controller. High gain controllers make the task easier by providing large assistance forces. Given the normalized task difficulty scalar z ranging from 0 to 1, we set $kp(z) = 200 - 200z$. kd is set to be equal to kp as proposed in [Yu et al. 2018]. The policy training starts at $kp(0)$. kp gradually decreases to $kp(z)$ during training according to the curriculum schedule in [Yu et al. 2018]. We refer readers to section 4.2.2 in [Yu et al. 2018] for more details. The multi-fidelity objective is the normalized return defined as:

$$f(x, z) = \frac{\sum_{t=1}^{T} r(s_t, a_t)}{T} \tag{7}$$

where $r(s_t, a_t)$ is the reward function, and T is the episode length fixed to 500 simulation steps (8.25 seconds).

4.1.3 Comparison. We compare our method with several baseline and state-of-the-art algorithms: a single fidelity BO with GP-UCB acquisition [Srinivas et al. 2010], a multi-fidelity method BOCA [Kandasamy et al. 2017] and a recent DRL-based algorithm [Schaff et al. 2019] (Schaff et al. 2019) specialized for morphology optimization. GP-UCB and [Schaff et al. 2019] can be directly applied to the problem. For multi-fidelity method BOCA, We implement it following [Kandasamy et al. 2017] and construct the multi-fidelity function parameterized by both number of training iterations and our proposed curriculum-based task difficulty. Maximum allowed iteration number at fidelity z for BOCA is set to $500 + 1500z$ since most training finishes within 2000 iterations. Maximum allowed simulation steps N_{total} is set to 10^8. The optimization process takes about 18 hours for all methods.

Length scale	Range	Radius scale	Range
Torso	[0.3, 2]	Torso	[0.5, 1.5]
Waist	[0.3, 2]	Waist	[0.5, 1.5]
Pelvis	[0.3, 2]	Pelvis	[0.5, 1.5]
Thigh	[0.3, 2]	Thigh	[0.5, 1.5]
Knee	[0.3, 2]	Knee	[0.5, 1.5]
Foot	[0.3, 2]	Foot	[0.5, 1.5]

Table 1. Morphology hyperparameters.

(a)   (b)
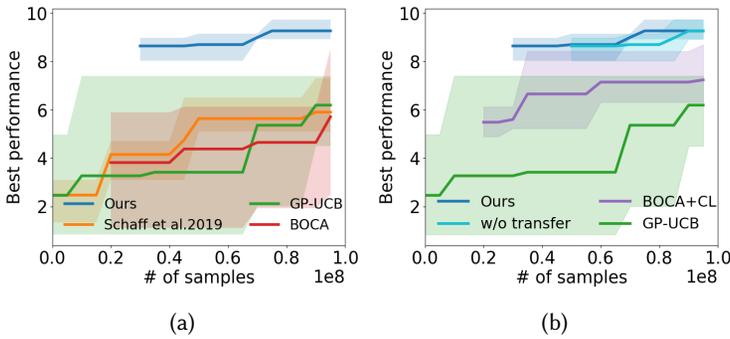
Fig. 4. (a):Results on morphology optimization. We compare best performance over the number of training samples among CMFBO (Ours), GP-UCB, BOCA with number of iteration as fidelity and [Schaff et al. 2019]. (b): Ablation study on morphology optimization. We further compare our method with BOCA with curriculum-based fidelity (BOCA+CL), and CMFBO without policy transfer (w/o transfer).
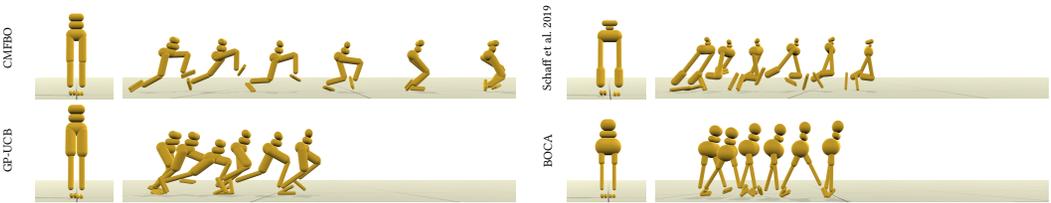
Fig. 5. Visualization of morphology and gaits optimized by different methods.

4.1.4 Results. We plot the best performance over number of total simulation steps in figure 4a for various methods. Note that for multi-fidelity methods, best performance is only recorded when the original objective function is evaluated. Our method finds high-performing morphology hyperparameters much faster than other methods. BOCA performs worse than single fidelity method GP-UCB due to a misleading fidelity criterion defined as number of training iterations, as explained in section 3.2. Figure 5 visualizes the best morphology and the associated gaits for each method obtained from 3 individual runs. Our method learns morphology resembling human and shows fast and stable gaits. We encourage readers to see supplementary videos for more details. Note that though our method learns different morphology hyperparameters at different runs, proper values of essential components key to stable locomotion are consistently learned, like thigh and foot lengths.

Parameters	Range	Default
Batch size	[32,1024]	256
Policy updates per iteration	[1,10]	1
Learning rate of actor network	$[2.5 \times 10^{-6}, 2.5 \times 10^{-4}]$	2.5×10^{-6}
Learning rate of critic network	$[1 \times 10^{-4}, 1 \times 10^{-2}]$	1×10^{-2}
Weight decay	$[5 \times 10^{-4}, 5 \times 10^{-2}]$	5×10^{-4}
PPO clip rate	[0.02, 0.2]	0.2
Maximum of the gradient norm	[1, 100]	100

Table 2. Hyperparameters of DeepMimic

Figure 4b shows results of our ablation study, where we further compare our method against BOCA with our proposed curriculum-based fidelity (BOCA + CL), and our method without policy transfer (w/o transfer). With an effective fidelity criterion defined by curriculum-based task difficulty, BOCA shows a large performance gain. However, our method still outperforms BOCA by a large margin due to effective search space pruning through our progressive acquisition function. The use of transfer learning further improves efficiency of our method, with a saving of roughly 2×10^{7} samples before a high-performing set of hyperparameters is confirmed on the original fidelity.

4.2 DeepMimic Hyperparameter Optimization

DeepMimic is a recent DRL-based framework for diverse motor skill learning. Many recent physics-based character animation systems are built on DeepMimic, such as [Bergamin et al. 2019; Luo et al. 2020; Park et al. 2019; Won et al. 2020; Won and Lee 2019]. As far as we know, this is the first work towards hyperparameter optimization for high-dimensional physics-based character control systems. In our experiments, we focus on the optimization of DRL training hyperparameters listed in table 2. The choices of these hyperparameters are critical for the learning performance where none of our randomly sampled 5 hyperparameters lead to successful policy training. These hyperparameters correlate with each other implicitly and are hard to be manually tuned. Some recent works [Ma et al. 2021; Won and Lee 2019] therefore simply adopt the default DeepMimic hyperparameters directly into their systems with minor modifications. We evaluate our method on two motor skill learning tasks: walk and backflip. Our optimized hyperparameters improve DeepMimic learning efficiency by a large margin. Hyperparameters optimized for walking can be reused for other skills as well resulting in superior performance than original settings.

4.2.1 Curriculum and Multi-fidelity Objective. We construct our multi-fidelity function using a time-based curriculum proposed in [Kostrikov 2018; Peng et al. 2018a]. Given the normalized task difficulty scalar z, task difficulty is given by maximum episode length $T(z) = 30 + 570z$, which ranges from 1 to 20 seconds. Our multi-fidelity objective considers both policy performance and sample efficiency, which is defined as:

$$f(x, z) = \frac{\sum_{t=1}^{t=T(z)} r(s_t, a_t)}{T(z)} e^{-\left(\frac{N_{sample}}{C}\right)}. \tag{8}$$
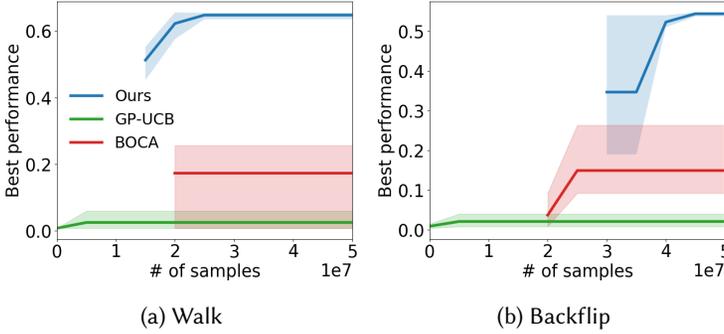
Fig. 6. DeepMimic hyperparameter optimization results. We compare best performance over the number of training samples among CMFBO, GP-UCB and BOCA.
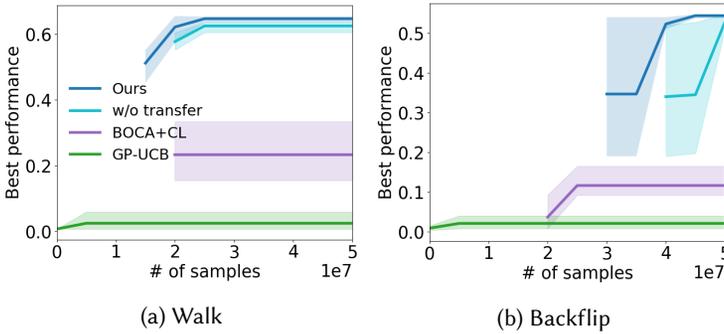
Fig. 7. Ablation studies on DeepMimic hyperparameter optimization.

N_{sample} is the total number of samples consumed during training. Training is terminated when normalized expected return reaches a threshold pre-defined in original DeepMimic, or the normalized expected return does not increase for 500 iterations. C is a scaling factor set to 10^7 and 2×10^7 for walk and backflip respectively, based on the scale of usual samples required by DeepMimic on different tasks. Note that original DeepMimic has a fixed annealing-based schedule for maximum episode length during training. This schedule can be too slow on easy tasks. We switch to an adaptive schedule to avoid wasting samples. Starting from 1 second, episode length is increased by 0.5 seconds every iteration until it reaches $T(z)$, as long as the current fail rate is below 20%.

4.2.2 Comparison. We compare our method against GP-UCB and BOCA in our experiments. For BOCA based on number of iterations as its fidelity criterion, maximum iterations on each fidelity $N(z)$ is set to $500 + 6500z$ and $500 + 11500z$ respectively for walk and backflip. We also show comparison with BOCA using our curriculum-based fidelity in our ablation study. Maximum allowed simulation steps N_{total} is set to 5×10^7 for all methods. The optimization process takes about 16 hours for all methods.

4.2.3 Results. As shown in figure 6, our method consistently outperforms other methods. CMFBO find different hyperparameters with different random seeds, but the performance is consistently good. Table 3 shows the best optimized hyperparameters, which are quite different from the default ones listed in table 2. We select best hyperparameters optimized from each method and test their

Parameters	Walk	Backflip
Batch size	172	48
Policy updates per iteration	8	10
Learning rate of actor network	1.68×10^{-5}	4.6×10^{-5}
Learning rate of critic network	2×10^{-3}	0.01
Weight decay	5×10^{-4}	5×10^{-4}
PPO clip rate	0.18	0.02
Maximum of the gradient norm	47.36	12.30

Table 3. Optimized hyperparameters for walk and backflip.

policy learning efficiency based on 3 individual runs. Results are shown in figure 8. GP-UCB fails to find hyperparameters with which policies could be learned successfully. Since DeepMimic is computationally expensive, only few evaluations can be performed in such single fidelity methods. BOCA performs better than GP-UCB and find hyperparameters as good as the default ones for walk task. It also finds hyperparameters converging 1.7 times faster than default ones for backflip. Comparing to GP-UCB and BOCA, our method finds high-performing hyperparameters highly efficiently. Hyperparameters optimized by CMFBO requires $5 - 6$ times less simulation steps than default settings for successful policy learning. We note that by default DeepMimic takes about 8 and 16 hours respectively to train control policies until convergence for walk and backflip, while our method can find high-performing hyperparameters within 8 and 13 hours respectively for these tasks, even before the training of default DeepMimic finishes. We conduct the same set of ablation studies as in morphology optimization shown in figure 7, where we compare CMFBO against BOCA with curriculum-based fidelity and CMFBO without policy transfer. As expected, each component is essential to achieve the impressive efficiency of CMFBO.

We show that hyperparameters optimized by CMFBO generalize to other tasks as well. Figure 9 illustrates the large performance gain of policy learning on run and cartwheel tasks using hyperparameters optimized by CMFBO on walk comparing with DeepMimic default settings.

CMFBO can also give us insights on the sensitivity of each hyperparameter. After optimization, the surrogate model shows small length scales along dimensions corresponding to policy updates per iteration, learning rate of critic network and weight decay. This implies that the performance is more sensitive to these hyperparameters within the given ranges.

We briefly analyze the computational cost distribution of CMFBO for DeepMimic hyperparameter optimization. The optimization of acquisition function usually finishes within three minutes, which is negligible comparing with the time required for DRL training. We summarize the number of training samples consumed on different fidelity levels in table 4, which indicates that most computational resources are allocated to low fidelity approximations. Easy tasks require less time to train than difficult ones. For instance, in walk task of DeepMimic, training at the easiest and the original difficulty level with our optimized hyperparameters take roughly 20 minutes and 80 minutes respectively.

5 CONCLUSIONS

This paper introduces the hyperparameter optimization problem in physics-based character animation. We present CMFBO, a novel multi-fidelity Bayesian optimization framework to achieve efficient optimization. Motivated by curriculum learning, we propose the use of task difficulty as an effective fidelity criterion, which enables relative performance of different hyperparameters to be accurately estimated even with low-fidelity cheap approximations. We enable efficient search

(a) Walk

(b) Backflip
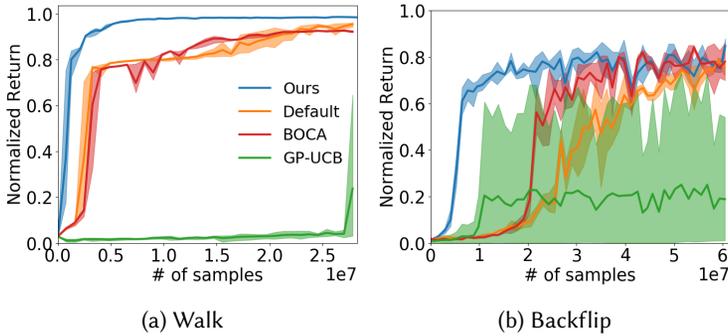
Fig. 8. Learning curves of policies trained with hyperparameters from CMFBO, DeepMimic default settings, BOCA and GP-UCB.

(a) Run

(b) Cartwheel
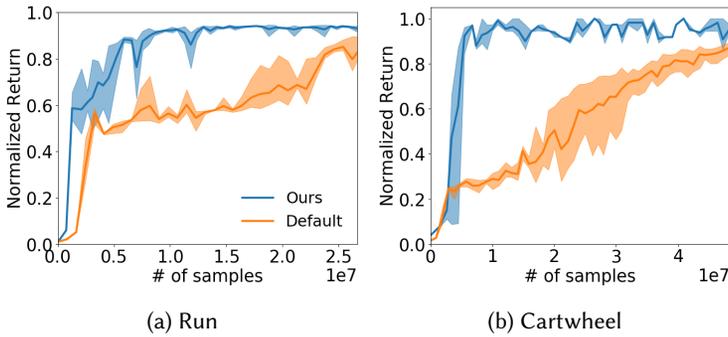
Fig. 9. Learning curves of policy learning for other tasks with hyperparameters optimized by CMFBO on walk task comparing with DeepMimic default settings.

	Walk	Backflip
# of sampled hyperparameters	20	23
# of training samples on original tasks	6.1×10^6	1.23×10^7
# of training samples on low fidelities	4.39×10^7	3.77×10^7

Table 4. The distribution of training samples in CMFBO

space pruning through a progressive acquisition function focusing only on optimal regions. Transfer learning is further adopted to reduce evaluation cost of function queries. Through extensive experiments, we demonstrate that CMFBO is more efficient than state-of-the-art hyperparameter optimization algorithms. In particular, we show that hyperparameters optimized through CMFBO result in at least 5x performance gain comparing to original author-released settings in DeepMimic. We believe that our method could serve as stepping-stone for automatic tuning of physics-based character animation systems and free researchers or engineers from laborious and exhausting tuning works.

There are several interesting future directions worth exploring. First, the search space grows exponentially with the dimension of hyperparameters. We could incorporate some recent machine learning and optimization techniques, like random embedding [Wang et al. 2013], to scale our

method to even higher dimensions requiring much larger computation budgets, such as optimizing the complex routing of muscles for full-body characters. Second, our current method is limited to Euclidean search space. It would be interesting to extend our method to general Non-Euclidean search space such as Riemannian manifolds in [Jaquier et al. 2020]. Last but not least, although our method is demonstrated on simulated characters, it could possibly be extended to real-world automatic robot design with the recent success of 'sim to real' transfer learning [Tan et al. 2018; Yu et al. 2019].

REFERENCES

Shailen Agrawal, Shuo Shen, and Michiel van de Panne. 2014. Diverse motions and character shapes for simulated skills. *IEEE transactions on visualization and computer graphics* 20, 10 (2014), 1345–1355.

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

Kevin Bergamin, Simon Clavet, Daniel Holden, and James Richard Forbes. 2019. DReCon: data-driven responsive control of physics-based characters. *ACM Transactions on Graphics (TOG)* 38, 6 (2019), 1–11.

Josh Bongard. 2011. Morphological change in machines accelerates the evolution of robust behavior. *Proceedings of the National Academy of Sciences* 108, 4 (2011), 1234–1239.

Eric Brochu, Tyson Brochu, and Nando de Freitas. 2010. A Bayesian interactive optimization approach to procedural animation design. In *Proceedings of the 2010 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*. 103–112.

Eric Brochu, Abhijeet Ghosh, and Nando de Freitas. 2007. Preference galleries for material design. *SIGGRAPH Posters* 105, 10.1145 (2007), 1280720–1280834.

Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. 2016. Openai gym. *arXiv preprint arXiv:1606.01540* (2016).

Stelian Coros, Philippe Beaudoin, and Michiel Van de Panne. 2010. Generalized biped walking control. *ACM Transactions on Graphics (TOG)* 29, 4 (2010), 1–9.

Stelian Coros, Andrej Karpathy, Ben Jones, Lionel Reveret, and Michiel Van De Panne. 2011. Locomotion skills for simulated quadrupeds. *ACM Transactions on Graphics (TOG)* 30, 4 (2011), 1–12.

Erwin Coumans and Yunfei Bai. 2016–2019. PyBullet, a Python module for physics simulation for games, robotics and machine learning. http://pybullet.org.

Martin De Lasa, Igor Mordatch, and Aaron Hertzmann. 2010. Feature-based locomotion controllers. *ACM Transactions on Graphics (TOG)* 29, 4 (2010), 1–10.

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

Scott Fujimoto, Herke Van Hoof, and David Meger. 2018. Addressing function approximation error in actor-critic methods. *arXiv preprint arXiv:1802.09477* (2018).

Thomas Geijtenbeek, Michiel Van De Panne, and A Frank Van Der Stappen. 2013. Flexible muscle-based locomotion for bipedal creatures. *ACM Transactions on Graphics (TOG)* 32, 6 (2013), 1–11.

GPy. since 2012. GPy: A Gaussian process framework in python. http://github.com/SheffieldML/GPy.

David Ha. 2019. Reinforcement learning for improving agent design. *Artificial life* 25, 4 (2019), 352–365.

Sehoon Ha, Stelian Coros, Alexander Alspach, Joohyung Kim, and Katsu Yamane. 2017. Joint Optimization of Robot Design and Motion Parameters using the Implicit Function Theorem.. In *Robotics: Science and systems*.

Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. 2018. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. *arXiv preprint arXiv:1801.01290* (2018).

Perttu Hämäläinen, Joose Rajamäki, and C Karen Liu. 2015. Online control of simulated humanoids using particle belief propagation. *ACM Transactions on Graphics (TOG)* 34, 4 (2015), 1–13.

Nikolaus Hansen. 2006. The CMA evolution strategy: a comparing review. In *Towards a new evolutionary computation*. Springer, 75–102.

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

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*. 71–78.

Sha Hu, Zeshi Yang, and Greg Mori. 2020. Neural fidelity warping for efficient robot morphology design. *arXiv preprint arXiv:2012.04195* (2020).

Wenlong Huang, Igor Mordatch, and Deepak Pathak. 2020. One policy to control them all: Shared modular policies for agent-agnostic control. In *International Conference on Machine Learning*. PMLR, 4455–4464.

Sumit Jain, Yuting Ye, and C Karen Liu. 2009. Optimization-based interactive motion synthesis. *ACM Transactions on Graphics (TOG)* 28, 1 (2009), 1–12.

Noémie Jaquier, Leonel Rozo, Sylvain Calinon, and Mathias Bürger. 2020. Bayesian optimization meets Riemannian manifolds in robot learning. In *Conference on Robot Learning*. PMLR, 233–246.

Donald R Jones, Matthias Schonlau, and William J Welch. 1998. Efficient global optimization of expensive black-box functions. *Journal of Global optimization* 13, 4 (1998), 455–492.

Kirthevasan Kandasamy, Gautam Dasarathy, Junier B Oliva, Jeff Schneider, and Barnabás Póczos. 2016. Gaussian process bandit optimisation with multi-fidelity evaluations. In *Advances in Neural Information Processing Systems*. 992–1000.

Kirthevasan Kandasamy, Gautam Dasarathy, Jeff Schneider, and Barnabás Póczos. 2017. Multi-fidelity bayesian optimisation with continuous approximations. *Advances in Neural Information Processing Systems* (2017), 1799–1808.

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

Aaron Klein, Stefan Falkner, Simon Bartels, Philipp Hennig, and Frank Hutter. 2017. Fast bayesian optimization of machine learning hyperparameters on large datasets. In *Artificial Intelligence and Statistics*. PMLR, 528–536.

Ilya Kostrikov. 2018. PyTorch Implementations of Reinforcement Learning Algorithms. https://github.com/ikostrikov/pytorch-a2c-ppo-acktr-gail.

Yuki Koyama, Issei Sato, and Masataka Goto. 2020. Sequential gallery for interactive visual design optimization. *ACM Transactions on Graphics* 39, 4 (Jul 2020). https://doi.org/10.1145/3386569.3392444

Yuki Koyama, Issei Sato, Daisuke Sakamoto, and Takeo Igarashi. 2017. Sequential line search for efficient visual design optimization by crowds. *ACM Transactions on Graphics (TOG)* 36, 4 (2017), 1–11.

Thomas Liao, Grant Wang, Brian Yang, Rene Lee, Kristofer Pister, Sergey Levine, and Roberto Calandra. 2019. Data-efficient learning of morphology and controller for a microrobot. In *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, 2488–2494.

Timothy P Lillicrap, Jonathan J Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. 2016. Continuous control with deep reinforcement learning. *International Conference on Learning Representations* (2016).

Hod Lipson and Jordan B Pollack. 2000. Automatic design and manufacture of robotic lifeforms. *Nature* 406, 6799 (2000), 974–978.

Dong C Liu and Jorge Nocedal. 1989. On the limited memory BFGS method for large scale optimization. *Mathematical programming* 45, 1-3 (1989), 503–528.

Matthew Loper, Naureen Mahmood, Javier Romero, Gerard Pons-Moll, and Michael J Black. 2015. SMPL: A skinned multi-person linear model. *ACM transactions on graphics (TOG)* 34, 6 (2015), 1–16.

Kevin Sebastian Luck, Heni Ben Amor, and Roberto Calandra. 2020. Data-efficient Co-Adaptation of Morphology and Behaviour with Deep Reinforcement Learning. In *Conference on Robot Learning*. PMLR, 854–869.

Ying-Sheng Luo, Jonathan Hans Soeseno, Trista Pei-Chun Chen, and Wei-Chao Chen. 2020. CARL: Controllable Agent with Reinforcement Learning for Quadruped Locomotion. *ACM Transactions on Graphics (Proceedings of SIGGRAPH 2020)* 39, 4 (2020), 10 pages.

Li-Ke Ma, Zeshi Yang, Tong Xin, Baining Guo, and KangKang Yin. 2021. Learning and Exploring Motor Skills with Spacetime Bounds. *Computer Graphics Forum* 40, 2 (2021).

Josh Merel, Saran Tunyasuvunakool, Arun Ahuja, Yuval Tassa, Leonard Hasenclever, Vu Pham, Tom Erez, Greg Wayne, and Nicolas Heess. 2020. Catch & Carry: reusable neural controllers for vision-guided whole-body tasks. *ACM Transactions on Graphics (TOG)* 39, 4 (2020), 39–1.

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, Emanuel Todorov, and Zoran Popović. 2012. Discovery of complex behaviors through contact-invariant optimization. *ACM Transactions on Graphics (TOG)* 31, 4 (2012), 1–8.

Vu Nguyen and Michael A Osborne. 2019. Knowing the what but not the where in Bayesian optimization. *arXiv preprint arXiv:1905.02685* (2019).

Vu Nguyen, Sebastian Schulze, and Michael Osborne. 2020. Bayesian optimization for iterative learning. *Advances in Neural Information Processing Systems* 33 (2020).

Jahng-Hyon Park and Haruhiko Asada. 1994. Concurrent design optimization of mechanical structure and control for high speed robots. (1994).

Soohwan Park, Hoseok Ryu, Seyoung Lee, Sunmin Lee, and Jehee Lee. 2019. Learning predict-and-simulate policies from unorganized human motion data. *ACM Transactions on Graphics (TOG)* 38, 6 (2019), 1–11.

Chandana Paul and Josh C Bongard. 2001. The road less travelled: Morphology in the optimization of biped robot locomotion. In *Proceedings 2001 IEEE/RSJ International Conference on Intelligent Robots and Systems. Expanding the Societal Role of*

Robotics in the the Next Millennium (Cat. No. 01CH37180), Vol. 1. IEEE, 226–232.

Xue Bin Peng, Pieter Abbeel, Sergey Levine, and Michiel van de Panne. 2018a. Deepmimic: Example-guided deep reinforcement learning of physics-based character skills. *ACM Transactions on Graphics (TOG)* 37, 4 (2018), 1–14.

Xue Bin Peng, Glen Berseth, and Michiel Van de Panne. 2015. Dynamic terrain traversal skills using reinforcement learning. *ACM Transactions on Graphics (TOG)* 34, 4 (2015), 1–11.

Xue Bin Peng, Glen Berseth, and Michiel Van de Panne. 2016. Terrain-adaptive locomotion skills using deep reinforcement learning. *ACM Transactions on Graphics (TOG)* 35, 4 (2016), 1–12.

Xue Bin Peng, Glen Berseth, KangKang Yin, and Michiel Van De Panne. 2017. Deeploco: Dynamic locomotion skills using hierarchical deep reinforcement learning. *ACM Transactions on Graphics (TOG)* 36, 4 (2017), 1–13.

Xue Bin Peng, Angjoo Kanazawa, Jitendra Malik, Pieter Abbeel, and Sergey Levine. 2018b. SFV: Reinforcement Learning of Physical Skills from Videos. *ACM Transactions on Graphics (TOG)* 37, 6 (2018).

Anton C Pil and H Haruhiko Asada. 1996. Integrated structure/control design of mechatronic systems using a recursive experimental optimization method. *IEEE/ASME transactions on mechatronics* 1, 3 (1996), 191–203.

Carl Edward Rasmussen. 2003. Gaussian processes in machine learning. In *Summer School on Machine Learning*. Springer, 63–71.

Charles Schaff, David Yunis, Ayan Chakrabarti, and Matthew R Walter. 2019. Jointly learning to construct and control agents using deep reinforcement learning. In *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, 9798–9805.

John Schulman, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz. 2015. Trust region policy optimization. In *International conference on machine learning*. 1889–1897.

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

Karl Sims. 1994. Evolving virtual creatures. In *Proceedings of the 21st annual conference on Computer graphics and interactive techniques*. 15–22.

Jasper Snoek, Hugo Larochelle, and Ryan P Adams. 2012. Practical bayesian optimization of machine learning algorithms. In *Advances in neural information processing systems*. 2951–2959.

Jasper Snoek, Oren Rippel, Kevin Swersky, Ryan Kiros, Nadathur Satish, Narayanan Sundaram, Mostofa Patwary, Mr Prabhat, and Ryan Adams. 2015. Scalable bayesian optimization using deep neural networks. In *International conference on machine learning*. 2171–2180.

Jialin Song, Yuxin Chen, and Yisong Yue. 2019. A general framework for multi-fidelity bayesian optimization with gaussian processes. In *The 22nd International Conference on Artificial Intelligence and Statistics*. 3158–3167.

Andrew Spielberg, Brandon Araki, Cynthia Sung, Russ Tedrake, and Daniela Rus. 2017. Functional co-optimization of articulated robots. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 5035–5042.

Niranjan Srinivas, Andreas Krause, Sham M Kakade, and Matthias Seeger. 2010. Gaussian process optimization in the bandit setting: No regret and experimental design. *Proceedings of the 27th annual international conference on machine learning*.

Kevin Swersky, Jasper Snoek, and Ryan P Adams. 2013. Multi-task bayesian optimization. In *Advances in neural information processing systems*. 2004–2012.

Kevin Swersky, Jasper Snoek, and Ryan Prescott Adams. 2014. Freeze-thaw Bayesian optimization. *arXiv preprint arXiv:1406.3896* (2014).

Shion Takeno, Hitoshi Fukuoka, Yuhki Tsukada, Toshiyuki Koyama, Motoki Shiga, Ichiro Takeuchi, and Masayuki Karasuyama. 2019. Multi-fidelity Bayesian optimization with max-value entropy search. *arXiv preprint arXiv:1901.08275* (2019).

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

Jie Tan, Tingnan Zhang, Erwin Coumans, Atil Iscen, Yunfei Bai, Danijar Hafner, Steven Bohez, and Vincent Vanhoucke. 2018. Sim-to-Real: Learning Agile Locomotion For Quadruped Robots. In *Proceedings of Robotics: Science and Systems*. Pittsburgh, Pennsylvania. https://doi.org/10.15607/RSS.2018.XIV.010

Yuval Tassa, Tom Erez, and Emanuel Todorov. 2012. Synthesis and stabilization of complex behaviors through online trajectory optimization. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 4906–4913.

Ethan Tseng, Felix Yu, Yuting Yang, Fahim Mannan, Karl ST Arnaud, Derek Nowrouzezahrai, Jean-François Lalonde, and Felix Heide. 2019. Hyperparameter optimization in black-box image processing using differentiable proxies. *ACM Trans. Graph.* 38, 4 (2019), 27–1.

Michiel Van de Panne and Alexis Lamouret. 1995. Guided optimization for balanced locomotion. In *Computer Animation and Simulation'95*. Springer, 165–177.

Miguel G Villarreal-Cervantes, Carlos A Cruz-Villar, Jaime Alvarez-Gallegos, and Edgar A Portilla-Flores. 2012. Robust structure-control design approach for mechatronic systems. *IEEE/ASME Transactions on Mechatronics* 18, 5 (2012), 1592–1601.

Pauli Virtanen, Ralf Gommers, Travis E Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski, Pearu
    Peterson, Warren Weckesser, Jonathan Bright, et al. 2020. SciPy 1.0: fundamental algorithms for scientific computing in
    Python. *Nature methods* 17, 3 (2020), 261–272.
Kevin Wampler, Zoran Popović, and Jovan Popović. 2014. Generalizing locomotion style to new animals with inverse
    optimal regression. *ACM Transactions on Graphics (TOG)* 33, 4 (2014), 1–11.
Jack M Wang, David J Fleet, and Aaron Hertzmann. 2009. Optimizing walking controllers. In *ACM SIGGRAPH Asia 2009
    papers*. 1–8.
Jack M Wang, Samuel R Hamner, Scott L Delp, and Vladlen Koltun. 2012. Optimizing locomotion controllers using
    biologically-based actuators and objectives. *ACM Transactions on Graphics (TOG)* 31, 4 (2012), 1–11.
Tingwu Wang, Renjie Liao, Jimmy Ba, and Sanja Fidler. 2018. Nervenet: Learning structured policy with graph neural
    networks. In *International Conference on Learning Representations*.
Ziyu Wang, Masrour Zoghi, Frank Hutter, David Matheson, Nando De Freitas, et al. 2013. Bayesian Optimization in High
    Dimensions via Random Embeddings.. In *IJCAI*. 1778–1784.
Jungdam Won, Deepak Gopinath, and Jessica Hodgins. 2020. A scalable approach to control diverse behaviors for physically
    simulated characters. *ACM Transactions on Graphics (TOG)* 39, 4 (2020), 33–1.
Jungdam Won and Jehee Lee. 2019. Learning body shape variation in physics-based characters. *ACM Transactions on
    Graphics (TOG)* 38, 6 (2019), 1–12.
Jia-chi Wu and Zoran Popović. 2010. Terrain-adaptive bipedal locomotion control. *ACM Transactions on Graphics (TOG)* 29,
    4 (2010), 1–10.
Zhaoming Xie, Hung Yu Ling, Nam Hee Kim, and Michiel van de Panne. 2020. ALLSTEPS: Curriculum-driven Learning of
    Stepping Stone Skills. In *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation*.
KangKang Yin, Stelian Coros, Philippe Beaudoin, and Michiel Van de Panne. 2008. Continuation methods for adapting
    simulated skills. In *ACM SIGGRAPH 2008 papers*. 1–7.
KangKang Yin, Kevin Loken, and Michiel Van de Panne. 2007. Simbicon: Simple biped locomotion control. *ACM Transactions
    on Graphics (TOG)* 26, 3 (2007), 105–es.
Wenhao Yu, Visak CV Kumar, Greg Turk, and C Karen Liu. 2019. Sim-to-real transfer for biped locomotion. (2019).
Wenhao Yu, Greg Turk, and C Karen Liu. 2018. Learning symmetric and low-energy locomotion. *ACM Transactions on
    Graphics (TOG)* 37, 4 (2018), 1–12.