Coordinate Ascent MORE With Adaptive Entropy Control for Population-Based Regret Minimization

Maximilian Hüttenrauch
Karlsruhe Institute of Technology
Karlsruhe, Germany
m.huettenrauch@kit.edu

Gerhard Neumann
Karlsruhe Institute of Technology
Karlsruhe, Germany
gerhard.neumann@kit.edu

ABSTRACT

Model-based Relative Entropy Policy Search (MORE) is a population-based stochastic search algorithm with desirable properties such as a well defined policy search objective, i.e., it optimizes the expected return, and exact closed form information theoretic update rules. This is in contrast with existing population-based methods, that are often referred to as evolutionary strategies, such as CMA-ES. While these methods work very well in practice, the updates of the search distribution are often based on heuristics and they do not optimize the expected return of the population but instead implicitly optimize the return of elite samples, which may yield a poor expected return and unreliable or risky solutions. We show that the MORE algorithm can be improved with distinct updates based on coordinate ascent on the mean and covariance of the search distribution, which considerably improves the convergence speed while maintaining the exact closed form updates. In this way, we can match the performance of elite samples of CMA-ES while also showing a considerably improved performance of the sample average. We evaluate our new algorithm on simulated robotic tasks and compare to the state of the art CMA-ES.

CCS CONCEPTS

• **Computer systems organization → Robotics**; • **Computing methodologies → Continuous space search**; **Machine learning algorithms**; **Continuous space search**.

KEYWORDS

stochastic search, reinforcement learning, policy search, robotics

ACM Reference Format:
Maximilian Hüttenrauch and Gerhard Neumann. 2021. Coordinate Ascent MORE With Adaptive Entropy Control for Population-Based Regret Minimization. In *2021 Genetic and Evolutionary Computation Conference Companion (GECCO '21 Companion), July 10–14, 2021, Lille, France.* ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/3449726.3463183

1 INTRODUCTION

Many problems in robotics can be formulated as episodic, open-loop tasks, such as peg-in-a-hole, ball-in-a-cup or hitting a ball in robotic table tennis [2, 6]. However, they are not easy to be tackled with standard reinforcement learning approaches as the control frequency can be high and rewards non-Markovian, i.e., they have to be computed over whole movement trajectories.

One way to solve these problems is to use movement primitives such as Dynamical Movement Primitives [5] as a low dimensional way to parameterize trajectories. A trajectory is then described by a set of basis functions and a weight matrix that shapes the trajectory. The goal of the learning algorithm is to find a probability distribution over the policy's parameters (e.g. the weight matrix) that 1) maximizes the reward while 2) being robust towards perturbations of the parameters.

A class of algorithms to approach these problems are stochastic search algorithms [4, 7, 9]. The only information available to the algorithm are the function evaluations as usually no gradient information is available. Individual solution candidates are sampled from a search distribution which is typically chosen to be a multivariate normal distribution where the mean and covariance constitute the set of parameters to be optimized by the search algorithm.

In this paper, we re-visit and improve Model-based Relative Entropy Stochastic Search (MORE) [1]. The key idea behind MORE is to approximate the objective function with a surrogate model which allows for exact closed form updates. The search distribution is updated based on the surrogate model's parameters under information-theoretic constraints. Originally, a bound on the loss of entropy between iterations is supposed to ensure that the algorithm won't converge prematurely. Instead, we apply a coordinate ascent strategy which results in independent updates for the mean and covariance and only slowly update the covariance. Additionally, we take inspiration from the Covariance Matrix Adaptation - Evolutionary Strategies (CMA-ES) [4], one of the state-of-the-art stochastic search algorithms, which makes use of an entropy control mechanism in form of the step-size update. We include the step-size update into the MORE optimization and augment the algorithm with an adaptive entropy adaptation based on an evolution path (a smoothed sum over previous mean updates).

One benefit of MORE compared to CMA-ES in the policy search context is the objective it optimizes. While CMA-ES has no defined objective and mainly cares about finding one particular parameter vector that "works well", MORE optimizes an expectation of the reward under the current policy distribution over parameters. Optimizing the expectation can take into account uncertainties in execution which is especially useful in real world robot learning since badly behaving policies can result in damaging a robot.

We empirically evaluate our algorithm on two simulated robotics tasks.

GECCO '21 Companion, July 10–14, 2021, Lille, France

ACM ISBN 978-1-4503-8351-6/21/07…$15.00
https://doi.org/10.1145/3449726.3463183

2 COORDINATE ASCENT MORE

The goal of MORE is to find a search distribution (or policy) π that maximizes the expectation of an objective function $f(\boldsymbol{x})$, $\boldsymbol{x} \in \mathbb{R}^k$. This is achieved by an iterative process of sampling and updating the policy's parameters under information-theoretic constraints. Using a quadratic surrogate $f(\boldsymbol{x}) \approx \hat{f}(\boldsymbol{x}) = -1/2\, \boldsymbol{x}^\mathrm{T} \boldsymbol{A}\boldsymbol{x} + \boldsymbol{x}^\mathrm{T}\boldsymbol{a} + a_0$ allows for exact closed form updates if the search distribution is Gaussian.

In our new algorithm Coordinate Ascent MORE with Step Size Adaption (CAS-MORE), we parameterize the covariance as $\Sigma = \sigma^2 \boldsymbol{C}$ with an additional step-size parameter σ which scales the covariance matrix $\boldsymbol{C}$. This parameter controls the overall entropy of the distribution and has an additional benefit on the numerical stability of the matrix operations. We employ a coordinate ascent strategy on the following optimization problem for the mean, the covariance matrix, and the step size which allows for setting different bounds on each of the components.

$$\begin{aligned} \underset{\pi}{\text{maximize}} \quad & \int_{\boldsymbol{x}} \pi(\boldsymbol{x}) f(\boldsymbol{x}) \mathrm{d}\boldsymbol{x} \\ \text{subject to} \quad & \mathrm{KL}(\pi(\boldsymbol{x}) \parallel \pi_t(\boldsymbol{x})) \leq \epsilon \end{aligned}$$

We additionally incorporate an adaptive entropy control mechanism based on previous updates of the search distribution.

Let the current policy be $\pi_t(\boldsymbol{x}) = \mathcal{N}(\boldsymbol{x} \mid \boldsymbol{\mu}_{\pi_t}, \sigma_{\pi_t} \boldsymbol{C}_{\pi_t})$ and the new policy $\pi(\boldsymbol{x}) = \mathcal{N}(\boldsymbol{x} \mid \boldsymbol{\mu}_{\pi}, \sigma_{\pi} \boldsymbol{C}_{\pi})$. Where unambiguous, we leave out the subscript π for easier notation. After inserting the quadratic model, the integral in the objective can be solved in closed form and written as

$$\int_{\boldsymbol{x}} \pi(\boldsymbol{x}) \hat{f}(\boldsymbol{x}) \mathrm{d}\boldsymbol{x} = -\frac{1}{2}\boldsymbol{\mu}^\mathrm{T}\boldsymbol{A}\boldsymbol{\mu} - \frac{1}{2}\mathrm{tr}(\boldsymbol{A}\sigma^2\boldsymbol{C}) + \boldsymbol{\mu}^\mathrm{T}\boldsymbol{a} + a_0$$

where we can leave out a_0 since it has no influence on the optimal parameters $\boldsymbol{\mu}$, $\boldsymbol{C}$ and σ. The KL divergence between the two distributions can also be written in closed form and is given by

$$\begin{aligned} \mathrm{KL}(\pi(\boldsymbol{x}) \parallel \pi_t(\boldsymbol{x})) = & \frac{1}{2}\Big\{ (\boldsymbol{\mu}_t - \boldsymbol{\mu})^\mathrm{T} (\sigma_t^2 \boldsymbol{C}_t)^{-1} (\boldsymbol{\mu}_t - \boldsymbol{\mu}) \\ & + \frac{\sigma^2}{\sigma_t^2}\mathrm{tr}(\boldsymbol{C}_t^{-1}\boldsymbol{C}) - k + \log|\sigma_t^2 \boldsymbol{C}_t| - \log|\sigma^2 \boldsymbol{C}| \Big\}. \end{aligned}$$

The optimization problems can be solved using a non-linear optimization algorithm such as L-BFGS using the method of Lagrangian multipliers.

2.1 Updating the Mean

We start updating the mean by setting $\sigma = \sigma_t$ and $\boldsymbol{C} = \boldsymbol{C}_t$ and introducing a bound ϵ_μ to limit the change of the mean displacement. The optimization problem is given by

$$\begin{aligned} \underset{\pi}{\text{maximize}} \quad & \int_{\boldsymbol{x}} \pi(\boldsymbol{x}) \hat{f}(\boldsymbol{x}) \mathrm{d}\boldsymbol{x} \Big|_{\sigma=\sigma_t, \boldsymbol{C}=\boldsymbol{C}_t} \\ \text{subject to} \quad & \mathrm{KL}(\pi(\boldsymbol{x}) \parallel \pi_t(\boldsymbol{x}))|_{\sigma=\sigma_t, \boldsymbol{C}=\boldsymbol{C}_t} \leq \epsilon_\mu \end{aligned}$$

The optimal solution $\boldsymbol{\mu}^*$ in terms of the Lagrangian multiplier λ is then given by

$$\boldsymbol{\mu}^* = (\lambda(\sigma_t^2 \boldsymbol{C}_t)^{-1} + \boldsymbol{A})^{-1}(\boldsymbol{a} + \lambda(\sigma_t^2 \boldsymbol{C}_t)^{-1}\boldsymbol{\mu}_t)$$

2.2 Updating the Covariance Matrix

Next, we set $\boldsymbol{\mu} = \boldsymbol{\mu}_t$ and $\sigma = \sigma_t$ and solve the following optimization problem

$$\begin{aligned} \underset{\pi}{\text{maximize}} \quad & \int_{\boldsymbol{x}} \pi(\boldsymbol{x}) \hat{f}(\boldsymbol{x}) \mathrm{d}\boldsymbol{x} \Big|_{\boldsymbol{\mu}=\boldsymbol{\mu}_t, \sigma=\sigma_t} \\ \text{subject to} \quad & \mathrm{KL}(\pi(\boldsymbol{x}) \parallel \pi_t(\boldsymbol{x}))|_{\boldsymbol{\mu}=\boldsymbol{\mu}_t, \sigma=\sigma_t} \leq \epsilon_C \end{aligned}$$

The optimal solution $\boldsymbol{C}^*$ can be found analogously and is given by

$$\boldsymbol{C}^* = \nu(\nu \boldsymbol{C}_t^{-1} + \sigma_t^2 \boldsymbol{A})^{-1}.$$

where ν is again a Lagrangian multiplier.

2.3 Updating the Step Size

Last, we apply the same technique as before to update the step size. We set $\boldsymbol{\mu} = \boldsymbol{\mu}_t$ and $\boldsymbol{C} = \boldsymbol{C}_t$ and solve

$$\begin{aligned} \underset{\pi}{\text{maximize}} \quad & \int_{\boldsymbol{x}} \pi(\boldsymbol{x}) \hat{f}(\boldsymbol{x}) \mathrm{d}\boldsymbol{x} \Big|_{\boldsymbol{\mu}=\boldsymbol{\mu}_t, \boldsymbol{C}=\boldsymbol{C}_t} \\ \text{subject to} \quad & \mathrm{KL}(\pi(\boldsymbol{x}) \parallel \pi_t(\boldsymbol{x}))|_{\boldsymbol{\mu}=\boldsymbol{\mu}_t, \boldsymbol{C}=\boldsymbol{C}_t} \leq \epsilon_\sigma \end{aligned}$$

The optimal solution σ^* is given by

$$\sigma^* = \sqrt{\frac{\alpha k}{\frac{\alpha k}{\sigma_t^2} + \mathrm{tr}(\boldsymbol{A}\boldsymbol{C}_t)}}$$

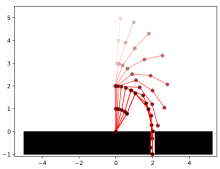
with Lagrangian multiplier α.

2.4 Incorporating an Evolution Path

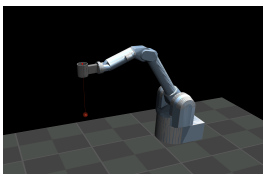
It can be shown that optimizing the expectation in MORE's objective always leads to the entropy of the search distribution becoming smaller. However, sometimes a slower decrease or even an increase in entropy is beneficial to the overall optimization process. To this end, we track consecutive mean updates $\boldsymbol{w} = \boldsymbol{C}_{\pi_t}^{-\frac{1}{2}} / 2\epsilon_\mu \sigma_t (\boldsymbol{\mu}_{t+1} - \boldsymbol{\mu}_t)$ and summarize them in an evolution path $\boldsymbol{e}_{t+1} = c_\sigma \boldsymbol{e}_t + \boldsymbol{w}$. The factor $\boldsymbol{C}_{\pi_t}^{-\frac{1}{2}} / 2\epsilon_\mu \sigma_t$ ensures $\|\boldsymbol{w}\| \leq 1$. The length of $\boldsymbol{e}$ is an indicator of the correlation between consecutive mean updates. A *long* evolution path is the result of several update steps in the same direction meaning that one larger step could have been sufficient, while a *short* evolution path is the result of having no clear update direction (see also [4]. In the first case, we want to increase the entropy to allow for larger steps, while in the second case we want to decrease entropy to find a meaningful update direction. The change in entropy is given by $\beta = p(1 - \|\boldsymbol{e}_{t+1}\| / e_{\mathrm{des}})$ where e_{des} is a desired length of a fully uncorrelated evolution path vector and p a gain factor. The adapted step size is given by $\sigma_e = \sigma^* \exp(-\beta/k)$.

2.5 Learning a Quadratic Model

The original MORE algorithm uses a Bayesian dimensionality reduction method to estimate the model parameters of the surrogate which proved to be brittle and time consuming. Instead, we propose to learn the model with standard ridge regression, augmented by several data pre-processing techniques. We apply whitening to the samples, normalization of the design matrix and a special mean and standard deviation normalization to the function values where we normalize with the mean and standard deviation of the top 50% of the data and cap outliers below -3. This has shown to stabilize the



(a) Peg-in-a-hole task.



(b) Ball-in-a-cup task.

Figure 1: Illustrations of the two tasks.

Task	ϵ_μ	ϵ_C	ϵ_σ	c_σ	p
Peg-In-A-Hole	0.5	0.005	0.001	0.9	1
Ball-In-A-Cup	0.5	0.01	0.001	0.9	1

Table 1: Hyperparameters for the tasks.

model parameter estimates especially in the case of reward functions with large jumps due to penalties. Additionally, we collect old samples in a FIFO queue buffer and start the optimization with a linear model once sufficient data is present (usually 1.5 times the number of parameters of the model). Finally, we reject a model if its solution $\boldsymbol{A}^{-1}\boldsymbol{a}$ is vastly outside a feasible range of solutions.

3 EXPERIMENTS

We evaluate CAS-MORE on two robotic benchmark tasks. A peg-in-a-hole task and the game of ball-in-a-cup. As a comparison, we use the original formulation of MORE, Coordinate Ascent MORE without step size adaptation (CA-MORE) and CMA-ES [1], a widely used stochastic search algorithm. All variants of MORE use the model learning approach as introduced in Section 2.5.

3.1 Peg-In-A-Hole

In this task, a 5-link robot arm has to reach into a narrow hole with its endeffector. Each link has a length of 1m. The base is at position (0, 0) and the hole is 25cm wide, 1m deep and at a distance of 2m. An illustration of a successful trial can be seen in Figure 1a. The trajectory is defined by a DMP where we learn 25 parameters for a weight matrix and 5 parameters for the final joint position. The length of a trajectory is 200 time steps which corresponds to 2s. The reward is given as the negative final distance of the endeffector to the bottom center of the whole minus an action cost for each time step. If the robot collides with itself, the ground, or a wall, an additional penalty is added to the reward and the episode is terminated.

Results. For our evaluation, we let each algorithm sample 14 trajectories per iteration. We collect samples in a buffer of size 744 (1.5 times the number of parameters in the quadratic model) and discard older samples once it is full. Remaining hyper-parameters are summarized in Table 1. Each algorithm is run 20 times with different random seeds. The initial position of the robot is upright as well as the initial goal position. We make several interesting conclusions from our experiments. We can see from Figure 2 that

[1] We use the official implementation from [3]

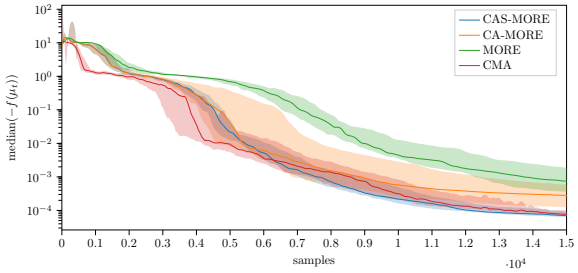


Figure 2: *Peg-In-A-Hole Task* The figure shows the median negative reward of the mean of the search distribution on a log scale (lower is better).

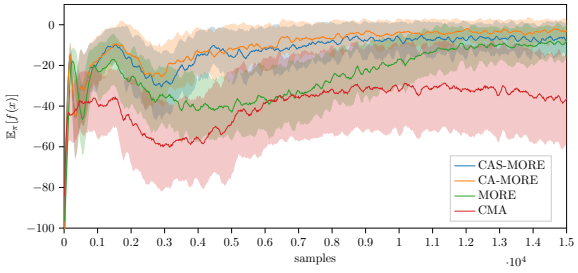


Figure 3: *Peg-In-A-Hole Task* The figure shows the mean reward of all samples drawn in an iteration.

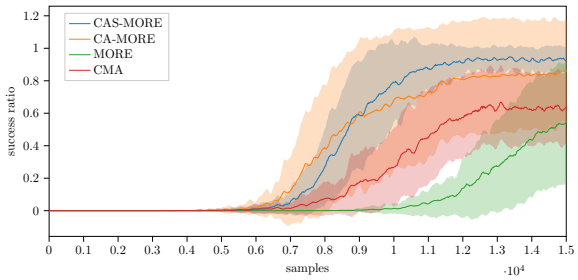


Figure 4: *Peg-In-A-Hole Task* The figure shows the mean ratio of successful samples.

our coordinate ascent MORE algorithm converges much faster than the original formulation of MORE. This can be attributed to the higher bound on the mean displacement we can choose for our new algorithm compared to the standard KL bound. While CMA-ES is quicker to converge to a point estimate, Figure 3 shows that it only cares about the mean of the search distribution. The samples drawn from the search distribution are of much worse quality. Figures 4 and 5 show the ratio of samples that achieved a final distance of 0.05 or smaller and the ratio of samples that lead to penalties over the optimization process, further illustrating this behavior.

3.2 Ball-In-A-Cup

The second task is the game of ball-in-a-cup played by a simulated 7 degrees of freedom Barrett WAM robotic arm. The trajectory is again parameterized by a DMP with 5 basis functions and we

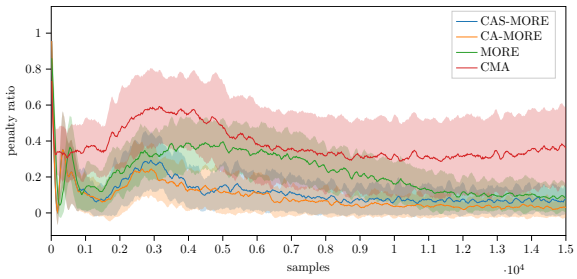


Figure 5: *Peg-In-A-Hole Task* The figure shows the mean ratio of samples where penalties occurred (lower is better).

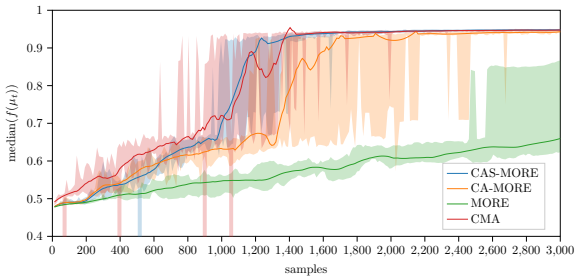


Figure 6: *Ball-In-A-Cup Task* The figure shows the median reward of the mean of the search distribution.

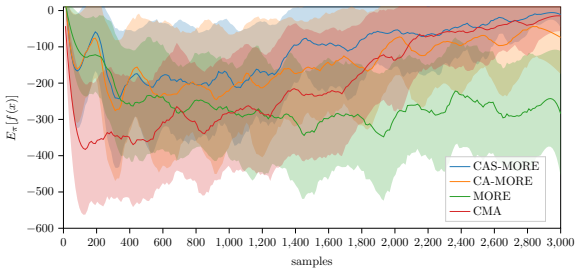


Figure 7: *Ball-In-A-Cup Task* The figure shows the mean reward of all samples drawn in an iteration.

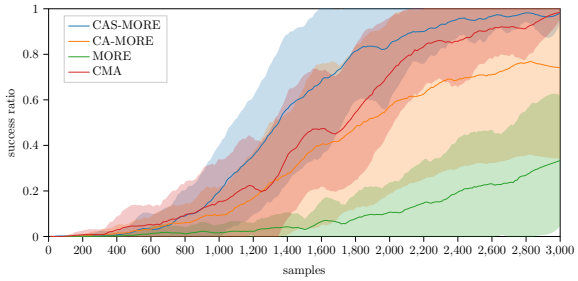


Figure 8: *Ball-In-A-Cup Task* The figure shows the mean ratio of successful samples.

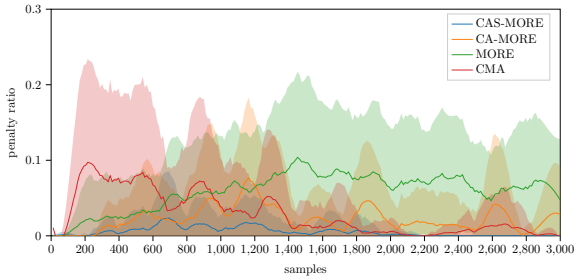


Figure 9: *Ball-In-A-Cup Task* The figure shows the mean ratio of samples where collisions occurred (lower is better).

actuate only the three joints that move the robot in the x-z plane. We choose the goal position to be the same as the start position resulting in a 15 dimensional parameter vector to be learned. The cost in this task is given as the distance of the ball to the bottom of the cup at the end of an episode plus the smallest distance of the ball to the central opening of the cup and the deviation of the cup to an upright position to encourage the cup to be upright and the ball to actually enter the cup and not simply touch it with its bottom. The reward is chosen to be the exponential of the negative cost plus an action penalty for each time step and a penalty whenever the robot goes into joint limits or collides with itself. A trajectory is deemed to be successful if no penalty occurs and the ball is in the cup at the end of an episode. We use mujoco [8] for the physical simulation of the robot. An illustration of the task can be seen in Figure 1b.

Results. For this experiment, we run 10 individual trials with CMA-ES and our new MORE algorithm. The weight matrix of the DMP is initialized with zeros meaning the task needs to be learned from scratch. In each iteration we sample 12 new weight matrices and evaluate the resulting trajectories and we keep the last 204 samples in the buffer (other hyper-parameters are shown in Table 1).

Figure 6 shows that on this task, CAS-MORE is on par with CMA-ES in terms of convergence speed and again chooses better samples during optimization (Figure 7). Upon visual inspection, the solutions found by CAS-MORE also show a more natural movement (probably as it further optimizes the energy cost) compared to CMA-ES. We can see the same behavior as in the previous experiment in the success rate of the samples drawn from the search distribution as well as the percentage of samples that lead to violations of joint limits or the simulation breaking (Figures 8 and 9).

4 CONCLUSION

In this paper, we proposed a new approach to the MORE algorithm based on a coordinate ascent strategy and an adaptive entropy regularization using an evolution path. Compared to the original formulation, we have higher convergence speed with stable closed form updates. Furthermore, we have an advantage over CMA-ES as we use the actual reward values to optimize the policy and not a ranking leading to a meaningful search distribution over parameter values.

REFERENCES

[1] Abbas Abdolmaleki, Rudolf Lioutikov, Jan R Peters, Nuno Lau, Luis Pualo Reis, and Gerhard Neumann. 2015. Model-based relative entropy stochastic search. *Advances in Neural Information Processing Systems* 28 (2015), 3537–3545.

[2] Marc Peter Deisenroth, Gerhard Neumann, Jan Peters, et al. 2013. A Survey on Policy Search for Robotics. *Foundations and Trends in Robotics* 2, 1-2 (2013), 1–142.

[3] Nikolaus Hansen, Youhei Akimoto, and Petr Baudis. 2019. CMA-ES/pycma on Github. Zenodo, DOI:10.5281/zenodo.2559634. https://doi.org/10.5281/zenodo.2559634

[4] N. Hansen and A. Ostermeier. 2001. Completely derandomized self-adaptation in evolution strategies. *Evolutionary computation* 9, 2 (2001), 159–195.

[5] Auke Jan Ijspeert, Jun Nakanishi, Heiko Hoffmann, Peter Pastor, and Stefan Schaal. 2013. Dynamical movement primitives: learning attractor models for motor behaviors. *Neural computation* 25, 2 (2013), 328–373.

[6] Jens Kober and Jan Peters. 2011. Policy search for motor primitives in robotics. *Machine Learning* 84, 1 (2011), 171–203.

[7] Shie Mannor, Reuven Y Rubinstein, and Yohai Gat. 2003. The cross entropy method for fast policy search. In *Proceedings of the 20th International Conference on Machine Learning (ICML-03)*. 512–519.

[8] Emanuel Todorov, Tom Erez, and Yuval Tassa. 2012. Mujoco: A physics engine for model-based control. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 5026–5033.

[9] Daan Wierstra, Tom Schaul, Tobias Glasmachers, Yi Sun, Jan Peters, and Jürgen Schmidhuber. 2014. Natural evolution strategies. *The Journal of Machine Learning Research* 15, 1 (2014), 949–980.