Sample and time efficient policy learning with CMA-ES and Bayesian Optimisation

Léni K. Le Goff[1], Edgar Buchanan[2] , Emma Hart[1], Agoston E. Eiben[3], Wei Li[2], Matteo de Carlo [3]
Matthew F. Hale[4], Mike Angus[2], Robert Woolley [2], Jon Timmis[5], Alan Winfield[4], Andrew M. Tyrrell[2]
[1]Edinburgh Napier University, Scotland, UK, [2]University of York, UK, [3]Vrije Universiteit Amsterdam, Netherlands
[4]University of West of England, UK, [5]University of Sunderland, UK
l.legoff2@napier.ac.uk

Abstract

In evolutionary robot systems where morphologies and controllers of real robots are simultaneously evolved, it is clear that there is likely to be requirements to refine the inherited controller of a 'newborn' robot in order to better align it to its newly generated morphology. This can be accomplished via a *learning* mechanism applied to each individual robot: for practical reasons, such a mechanism should be both sample and time-efficient. In this paper, We investigate two ways to improve the sample and time efficiency of the well-known learner CMA-ES on navigation tasks. The first approach combines CMA-ES with Novelty Search, and includes an adaptive restart mechanism with increasing population size. The second bootstraps CMA-ES using Bayesian Optimisation, known for its sample efficiency. Results using two robots built with the ARE project's modules and four environments show that novelty reduces the number of samples needed to converge, as does the custom restart mechanism; the latter also has better sample and time efficiency than the hybridised Bayesian/Evolutionary method.

Introduction

Evolutionary Computation is increasingly being used to evolve jointly the morphologies and controllers of robots. While such experiments are often conducted solely in simulation, more recently we are seeing the emergence of methods in which the resulting robots are built in the real-world, for example using soft materials (Lipson, 2014) or even biological cells (Kriegman et al., 2020). Furthermore, with the development of 3D-printing, rapid prototyping, and automated assembly the evolution of robots the real-world – as opposed to evolving in simulation– is becoming feasible, at least in an academic setting (Brodbeck et al., 2015; Vujovic et al., 2017; Jelisavcic et al., 2017). However, although these examples result in actuated robots, they tend not to include sensing capabilities. As morphologies become more complex, for example adding multiple types of sensor as well as multiple forms of actuation, an issue starts to arise that the controllers of a 'child' robot produced via reproduction operators may be sub-optimal for its new morphology. This can manifest in different ways: in the worst-case, this may result in a complete mis-match (e.g. the robot has more sensors than there are inputs in the controllers), while in the best-case it may simply require some tuning of the parameters of the controller. To address this, a general architecture for evolving robots in real-time and real-space was suggested by Eiben et al. (2013), a tangible instantiation of which is being created by the ARE project[1] (Hale et al., 2019). A key element of this architecture is the use of two loops: an evolutionary loop that produces a blue-print for morphologies and controllers that can then be fabricated, with a secondary *learning* loop that specialises the controller of a newly produced robot to its morphology (Eiben and Hart, 2020).

The learning loop can be realised in many ways, but a crucial factor is that it should take place in as few trials as possible, particularly if this is conducted directly on a physical robot. In simulation, while the number of trials is typically not an important factor, the computation time required to conduct such trials is. On a real robot, the time to conduct a trial is usually significant, and cannot be easily reduced. Trials also increase wear and tear on the robot and therefore the risk of damage, thus effort is usually geared towards reducing the number of trials needed.

The issue is further complicated by the fact that different learning methods exhibit different behaviours in terms of their *sample efficiency* (the number of samples need to reach a satisfactory solution) and their *time efficiency* (the learning progress over the time needed to obtain this progress.) For example, Bayesian Optimisation (BO) (Snoek et al., 2012) is well known to be sample efficient but has considerable computational complexity ($\mathcal{O}(n^3)$) due the time required to compute the inversion of the co-variance matrix and the maximisation of the acquisition function. On the other hand, evolutionary based search methods require constant time for generating candidate solutions, but this comes at the expense of sample efficiency.

With this in mind, this paper aims at investigating two ways to increase sample efficiency of the well-known algorithm CMA-ES (Hansen, 2006), while keeping computational time low. Firstly, CMA-ES is combined with *Nov-*

[1]www.york.ac.uk/robot-lab/are/

elty Search (Lehman and Stanley, 2011), given the deceptive nature of the tasks evaluated and previous reported success of this type of quality-diversity approaches on similar tasks (Pugh et al., 2016). A modified version of a CMA-ES variant is proposed (IPOP-CMA-ES (Auger and Hansen, 2005)), which exploits multiple restarts, each with increasing population size, increasing the global search over time (NIP-ES). Secondly, an alternative method is proposed in which the novelty-assisted CMA-ES is bootstrapped by a Bayesian Optimiser. This combination draws inspiration from the Bayesian-Evolutionary algorithm BEA proposed by Lan et al. (2020) which has shown impressive results in term of sample and time efficiency on modular robotics platform. To evaluate these algorithms, Experiments are conducted using robots with two morphologies on a maze navigation task in four different environments. Specifically, the paper investigates: (1) To what extent does adding novelty and a restart mechanism improve the sample efficiency of the CMA-ES learner? (2) Does bootstrapping CMA-ES with a Bayesian Optimiser influence its time and sample efficiency? (3) How do the best algorithms found generalise over multiple morphologies and multiple environments?

The novel contributions of the paper are twofold. Firstly, we show that adding novelty and a custom restart mechanism to CMA-ES (a time-efficient learning method) significantly improves its sample efficiency when solving deceptive navigation tasks, by appropriately balancing exploration and exploitation. We show that these results generalise across two morphologies that have multiple sensors and wheeled actuators. Secondly, on the task, the environments, and the robots used in this study, NIP-ES is shown to have better sample and time efficiency than NBO-ES, in contrast to previously reported research (Lan et al., 2020) where the hybridisation with BO was shown to be beneficial in a modular robot settting.

Related Work

Previous approaches to individual *learning*, also referred to as *policy-search*, include Bayesian Optimisation (BO), Reinforcement Learning (RL) and Evolutionary Algorithms (EA).

A recent survey compares methods which aim at learning policies in a *handful of trials* (Chatzilygeroudis et al., 2019), concluding that model-based BO methods such as PILCO (Deisenroth and Rasmussen, 2011) or Black-DROPS (Chatzilygeroudis et al., 2017) are the most sample efficient type of methods, but this is to the detriment of time-efficiency. Moreover, these methods are computationally very expensive when the parameter space is large. This can be overcome by using priors on the dynamics of the robot in tasks such as gait learning (Chatzilygeroudis and Mouret, 2018) but this is not possible when morphologies are unknown in advance, for example when morphologies are evolved.

Sample-efficiency is also a central issue in RL, as the number of evaluations needed to converge increases as a function of the action/state space's size. Efficient RL methods have been proposed both in classical RL (Yu, 2018) and in deep-RL (Feinberg et al., 2018) although the latter often need large amounts of data to converge. However, RL methods have been shown to have limitations in the case of tasks with sparse or deceptive rewards. To address this, algorithms capable of better exploration are required.

EAs are an interesting alternative to RL as they have shown to be better in term of exploration (Stulp and Sigaud, 2013). In particular, Evolutionary Strategies (ES) — specifically Co-variance Matrix Adaptation ES (CMA-ES) (Hansen, 2006) — have been used in several studies as an alternative or complementary methods to RL because of their efficiency and scalability (Pourchot and Sigaud, 2018; Salimans et al., 2017). In contrast to RL methods, ES methods evaluate an entire set of solutions per iteration, increasing their exploratory power. Moreover, their similarity to population-based EA permits the combination of ES with divergent search algorithms such as Novelty Search (NS) (Conti et al., 2018): the highly exploratory power of these methods enables them to solve tasks with deceptive or sparse rewards but at the same time, reduces sample-efficiency.

A disadvantage of EA methods compared to BO and RL is the amount of data exploited per trial: an EA relies on a single assignment of fitness from episode to generate future samples, while the former methods exploit data from multiple states. A recent approach tried to realise the benefits of both approaches in a hybridised method called the Bayesian-Evolutionary Algorithm (BEA Lan et al. (2020)) , which bootstraps an EA with the data generated by a model free Bayesian optimiser (MFBO). The time-efficiency of BO is better than the EA during the early stages of learning, but becomes worse over time — at the point at which it becomes worse than the EA, the algorithm automatically switches to the EA. However, the method was only tested in low dimensional parameter space (18 parameters). As the time complexity of BO is exponential as a function of the dimension of the parameter space (Li et al., 2018), it is unclear whether the method will scale to networks such as those used in this study with more than one hundred parameters.

Background

We first fix some vocabulary and notation given that several concepts are common in BO, RL, and EA but named differently.

- **Controller** : A specific policy which takes sensor values as input and outputs motor commands.
- **Sample** : a particular set of values of the controller's parameters, i.e. a specific point in the *parameter space*, denoted s. In the EA community, this is usually called a *genome*.

- **Reward** : the output of the objective function for a given sample, denoted r (usually referred to in an EA as *fitness*).
- **Behavioural descriptor** : a low dimensional feature vector, denoted o which uniquely describes a behaviour according to the task. The space where these descriptors are defined is called the behavioural space B. (In RL, this is called a *state* and in BO an *observation*).

All the algorithms described process data in the form of triple (s, o, r): a sample, a behavioural descriptor, and a reward.

CMA-ES and IPOP-CMA-ES

Co-variance Matrix Adaptation Evolution Strategies (CMA-ES) are a specific type of EA: following initialisation from a random sample the process : (1) draws λ samples from a normal distribution; (2) evaluates the λ samples according to the objective function; (3) selects the μ top samples according to their rewards (4) updates the co-variance matrix and the mean using an incremental rule which takes into account the new μ selected samples, controlled by a parameter called the sigma step which decreases over the generations. For more details, the reader can refer to the tutorial written by Hansen (2016).

An extension of this algorithm called increasing population CMA-ES (IPOP-CMA-ES) was proposed by Auger and Hansen (2005). The principle is simple: the algorithm starts with a small population, and then based on some criteria, is halted and restarted multiple times, each time with increasing population size. When the algorithm restarts all the parameters are reinitialised and a new random initial sample is drawn. The stopping criteria detect either the degeneration of the co-variance matrix or a need for more global exploration. In the original study multiple criteria are proposed (Auger and Hansen, 2005): here we use custom criteria more appropriate for our task (see the Methods' section). We use the implementation of IPOP-CMA-ES available in the *libcmaes* library (Benazera, 2014).

Novelty Search

Novelty Search (NS), introduced by Lehman and Stanley (2011), is an EA in which the traditional fitness is replaced by a novelty score, resulting in a divergent search. The computation of the novelty score is based on the defined behavioural space. Novelty is computed by comparing a controller to its k nearest neighbours in the population *and* an archive of past controllers, as shown in equation 1 (where o and o_i are the behavioural descriptors of the controllers, B the behavioural space, and d_B the distance defined on B).

$$S(o) = \frac{1}{K} \sum_{i=0}^{K} d_B(o, o_i) \tag{1}$$

The archive is updated at each generation by adding new controllers based on pre-defined criteria. Several criteria are proposed in the literature: in this paper the combination of the stochastic criterion and the novelty criterion defined in the empirical study of Gomes et al. (2015) are used. The former adds a new controller with a fixed probability while the latter adds a controller with a novelty above a fixed threshold. A practical feature of NS is its easy combination with any EA.

Model-Free Bayesian Optimisation

Model-Free Bayesian Optimisation (MFBO) is a black-box optimisation algorithm. In this paper, BO is used as a state-based optimiser in which a probabilistic distribution models a function (based on Gaussian Processes) that maps a sample s to its associated behavioural descriptor o.

MFBO follows a typical active learning scheme : (1) the probabilistic model is updated based on the data collected and evaluated so far; (2) an acquisition function is maximised to find the best next sample to evaluate; (3) the new sample is evaluated to obtain its behavioural descriptor and reward. The reader is referred to Brochu et al. (2010) for a detailed description. Several functions need to be defined by the user. A *kernel function* is required to shape the uncertainty model used by BO - we select the Matèrn 5/2 as used in BEA (Lan et al., 2020). An acquisition function (which controls the balance between exploration and exploitation) must also be chosen; again we refer to BEA, selecting the function GP-UCB (Lan et al., 2020). The implementation of BO available in the LIMBO framework (Cully et al., 2016) is used in this work.

Methods

A fully connected recurrent neural network (RNN) is used as a policy representation whose weights and biases need to be optimised by a learning method. The candidate algorithms are described below. All parameters required to configure the algorithms are given in table 1.

NIP-ES

To add novelty to CMA-ES (N-CMA-ES), inspiration is drawn from NSRA-ES, proposed by Conti et al. (2018). The objective is a weighted sum of the reward and the sparseness (see equation 2). A novelty ratio η determines how much of novelty or reward are taken into account in the objective function. η is decreased by a fixed amount at each iteration.

$$f(x) = \eta * S(x) + (1 - \eta) * r(x) \tag{2}$$

To extend N-CMA-ES to NIP-ES, IPOP-CMA-ES is customised by modifying the stopping criteria used to manage restarts. The criteria defined in the classical IPOP-CMA-ES (Auger and Hansen, 2005) are replaced by two stopping criteria :

- the standard deviation of the population's rewards is ≤ 0.05

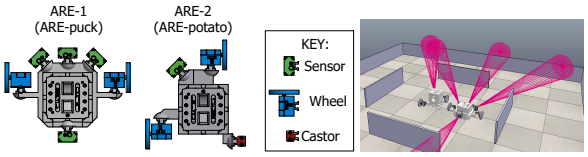


Figure 1: Two pictures of the ARE-1 (*"ARE-puck"*)and ARE-2 (*"ARE-potato"*) side by side. The sensors are proximity sensors of 1 meter range and conic dispersion (in purple on the right picture). These pictures are screenshots taken from the V-REP simulator.

- the standard deviation of the best rewards over a window of 20 iterations is ≤ 0.05

These two criteria enable the algorithm to avoid premature convergence. The main difference w.r.t classical IPOP-CMA-ES is the frequency of restart occurrence — here, the stopping criteria are designed to trigger more frequent restarts as sample efficiency is sought. Following a restart, the sigma step and the novelty ratio are reinitialised to their starting values, the population size (λ) is doubled and the algorithm restarts from a new random sample.

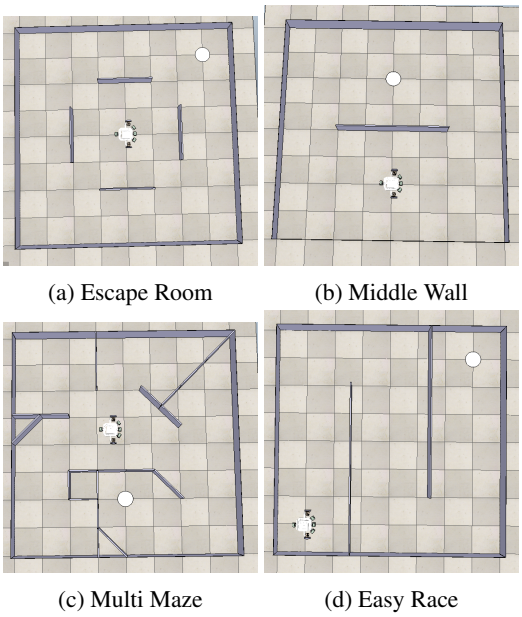
NBO-ES

Novelty driven and bootstrapped with Bayesian optimisation CMA-ES (NBO-ES) draws inspiration from BEA. The set of solutions generated by running MFBO is used to create the initial population of the EA. Like BEA, NBO-ES starts with an MFBO which is itself bootstrapped by a dataset randomly generated using Latin Hypercube sampling (Tang, 1993) and runs for a fixed number of iterations. Following this, the λ best solutions generated by MFBO are used to initialise the starting population of N-CMA-ES.

Experimental Set-up

The experiments conducted during this study are navigation tasks. The goal is for a robot to reach a specified target position within 5% of precision. The radius of the white circles shown in figure 2 indicate this target region. All learning algorithms have a maximum budget of 10000 samples to complete this task.

Four arenas are used (see figure 2). These arenas are designed to have different features and to be increasing in term of difficulty. The *escape room* is the simplest, the only difficulty for the robot is to be able to drive between the walls. The *middle wall* and *multi maze* are designed to feature a deceptive reward and thus the need to explore to reach the goal. Additionally, the *multi maze* is designed to punish excessive exploration as the robot starts in the centre of the maze and there are multiple futile routes. Finally, the *easy race* is designed to be difficult in term of control. This arena features the longest path from the initial position to the target, such that the robot has to move fast and accurately to achieve the task. Moreover, it features a deceptive reward because of the first turn which is very close to the target.



(a) Escape Room (b) Middle Wall

(c) Multi Maze (d) Easy Race

Figure 2: Environments used in the experiments. The target positions are represented by the white circles and the initial positions by the robot. The radius of the circle correspond to 5% precision required to achieve the task. The arenas are 2 by 2 meters squares. These pictures are screenshots taken from the V-REP simulator.

Two robots are used during the experiments (the ARE-1 aka *"ARE-puck"* and the ARE-2 aka *"ARE-potato"* (see figure 1) which are built from components used to autonomously evolve and fabricates robots in the ARE project. The *ARE-puck* has two wheels symmetrically positioned on each side of its body and 4 sensors: 3 on one side and 1 on another. The *ARE-potato* has two wheels and a caster wheel, placed asymmetrically in relation to its body, and one sensor. While the *ARE-puck* is a classical design for a navigation task, the *ARE-potato* was purposely designed to pose difficulties and thus challenge a learner.

The experiments are conducted with the simulator V-REP (Rohmer et al., 2013). As previously noted, the robots are controlled by an RNN with one input per sensor, one output per wheel, and 8 hidden neurons. This constitutes a parameter space of 168 dimensions for the ARE-1 (ARE-puck) (154 weights and 14 biases) and 117 dimensions for the ARE-2 (ARE-potato) (106 weights and 11 biases).

The reward is the normalised distance to the target position as defined in equation 3.

$$r(s) = 1 - \frac{d_B(p_f(s), p_t)}{D_{max}} \tag{3}$$

where, s is a sample, d_B is the spatial Euclidean distance,

BO: size of initial dataset	50
BO: Number of iterations	50
BO: GP-UCB hyperparameter (δ)	0.1
CMA-ES: Population size (λ)	50
CMA-ES: Sigma step	1
NIP-ES: Initial population size (λ)	10
Novelty: initial ratio μ (η)	1
Novelty: μ decrement	0.05
Novelty: k (nearest neighbours)	15
Novelty: threshold to add to archive	0.9
Novelty: probability to add to archive	0.4
Evaluation time	120 seconds
Maximum number of samples	10000

Table 1: The hyperparameters values used in the experiments

$p_f(s)$ the final position of s, p_t the target position, and D_{max} the maximum distance possible between two points in the current arena. Thus, completing the task corresponds to a reward above 0.95. The behavioural descriptor required to compute novelty is defined by the final position of a robot ($o = p_f(s)$).

To compare the different methods, 3 different measures are used: (1) the number of samples needed to reach the target within 5% accuracy; (2) the computational time in seconds needed to reach the target within 5% accuracy; (3) the maximum reward obtained. For each experiment, 10 replications are conducted. All experiments are launched on computers equipped with two Xeon E5-2640v3 2.60GHz 8C/16T; no GPU are used. All matrix operations are computed in parallel. The source code used for this study is available here : https://bitbucket.org/autonomousroboticsevolution/alife2020.

Results

The first set of experiments aims to assess the potential benefits of augmenting CMA-ES with a) novelty search (N-CMA-ES) and b) a restart mechanism (NIP-ES). These experiments are conducted using the ARE-1 ("*ARE-puck*") robot, and only sample efficiency is measured as CMA-ES take a constant time to process each sample hence the number of samples and the time of computation are equivalent. Figure 3 shows the number of samples needed for each algorithm to complete the task.

Overall, the best method is NIP-ES, which completes the task in less than 1000 samples on average and in around 100 samples for the multi maze. Interestingly, N-CMA-ES needs fewer samples on average than CMA-ES on the easy race and the middle wall, but it needs more samples than CMA-ES on the multi maze. This could be explained by the fact that the multi maze was designed to punish exploration by featuring multiple false paths around the initial position of the robot. A novelty based algorithm will sacrifice some samples in exploring these parts, while a purely goal based algorithm will not.

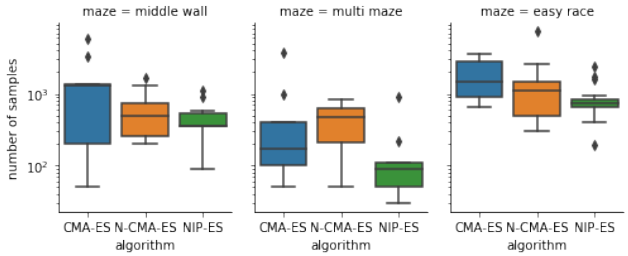


Figure 3: Box plots representing the number of evaluations needed to reach the target with 5 % of accuracy. CMA-ES, N-CMA-ES and NIP-ES are compared on the middle wall, the multi maze, and the easy race and on the ARE-1 (*"ARE-puck*). The y-axis is in logarithmic scale.

Next, the hybridised algorithm NBO-ES is considered and compare it to algorithms that comprise its constituent parts, i.e. a) a pure BO approach (MFBO) and b) N-CMA-ES. Experiments are conducted with a fixed budget of 300 samples due to exponential increase in computational time associated with MFBO. MFBO and NBO-ES both have an initial dataset of 50 samples. Figure 4 shows the best reward reached within the budget and the computational time needed to process 300 samples. It would be difficult to decide between the three methods based only on the best reward. Indeed, only on the easy race are the results clear; N-CMA-ES reaches the highest reward on average and MFBO have a large variance compared to NBO-ES.

However, in terms of computational time, differences between the methods are clear (see figure 5). MFBO needs on average around 60000 seconds, i.e. $\approx$ 17 hours, to process 300 samples, while NBO-ES needs on average around 3000 seconds, i.e. a little less than 1 hour, and N-CMA-ES on average around 300 seconds, i.e. 5 minutes. According to these results the obvious choice seems to be N-CMA-ES. However, for both NBO-ES and N-CMA-ES, many runs did not manage to complete the task (the exact number is indicated on the figure 4), i.e. obtain a reward above 0.95. Therefore, it is possible that with more samples, NBO-ES could outperform a variant of CMA-ES.

To further compare NIP-ES and NBO-ES, experiments on the four mazes and with the two robots were conducted. In these experiments, the budget is 10000 samples as described in the previous sections. The results are presented in figures 6 and 7. Overall, only a few runs did not complete the tasks on the 8 different experiments : three on the multi maze with the ARE-potato, one on the easy race with the ARE-puck, and five on the easy race with the ARE-potato (see figure 6). The most difficult experiment seems to be the easy race with the ARE-potato. The variance is large for both methods. Half of the replications with NBO-ES did not complete the task. This was expected as the ARE-potato was not designed to be efficient in term of navigation and to complete the easy

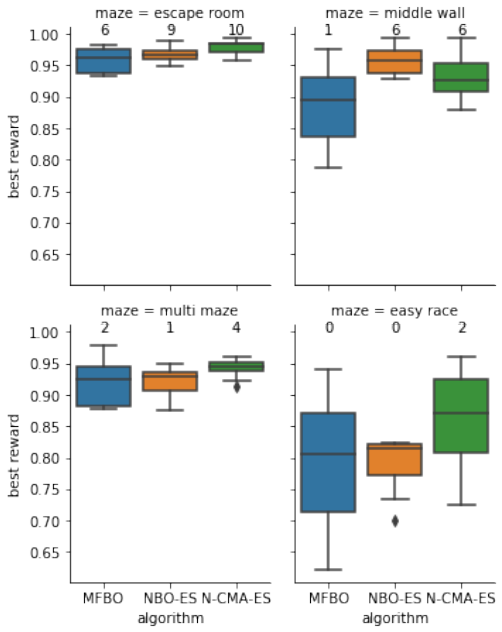


Figure 4: Box plots representing the best reward reached by MFBO, NBO-ES, and N-CMA-ES on the four mazes and with the ARE-1 ("*ARE-puck*") with a budget of 300 samples. The numbers above the boxes indicate the number of replications which have completed the task.

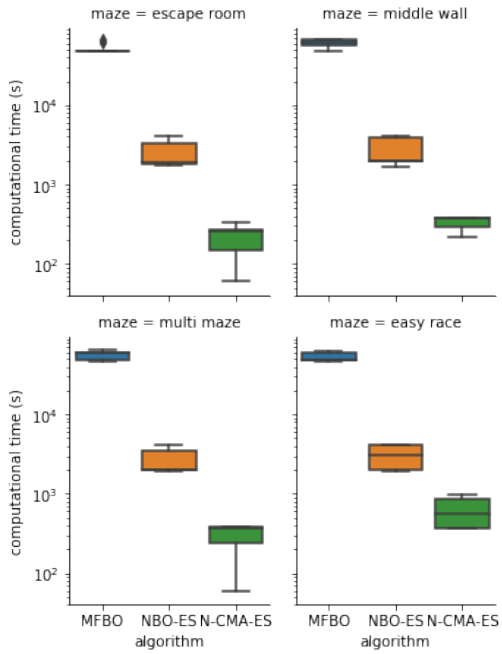


Figure 5: Box plots representing the computational time needed for processing 300 samples for MFBO, NBO-ES, and N-CMA-ES on the four mazes and with the ARE-1 ("*ARE-puck*"). The y-axis is in logarithmic scale.

race a fast and accurate controller is needed.

Apart from the experiments conducted on the middle wall, NIP-ES has a better sample efficiency than NBO-ES and on all the mazes and with both robots NIP-ES has a better time efficiency than NBO-ES (see figures 6 and 7). On average, NIP-ES is able to generate a successful controller in hundreds of samples and in fewer than 200 samples and less than 40 minutes of computational time for some runs on the simpler tasks (see table 2). In contrast, NBO-ES needs 2 hours for the longest run. Also, NIP-ES seems more reliable as its variance over the replication is globally lower than the one of NBO-ES (see figures 6, 7).

Discussion

In this study, two different approaches to improving the sample-efficiency of CMA-ES were investigated in a robot learning application. The motivational scenario is that of a morphologically evolving robot system, where newborn robots must optimise their inherited controller for their inherited body plan (Eiben and Hart, 2020). Arguably, the optimisation / learning algorithm would benefit from using the inherited controller as a starting point as opposed to a random start. However, in the current experiments we could not benefit from this effect, because the robots we used were not evolved but hand-designed. Therefore, they had no inherited controller and we started the optimisation process from scratch. This means that our analyses belong to a worst-case scenario and we expect that the results would be better if the algorithms were integrated in a full blown evolutionary robot system where newly produced robots come with a non-random controller.

The first version of the CMA-ES used a restart strategy with increasing population, starting from a small population and increasing it if more exploration is needed. The second approach bootstrapped CMA-ES with MFBO, as it is well known for its sample efficiency. By switching from MFBO to CMA-ES in the early iterations, the goal is that the overall time efficiency of the algorithm can be maintained.

However, MFBO does not appear to be an appropriate choice in the experimental conditions used in this study. Firstly, the parameter space size ($\approx$ 150) results in a significant time overhead. Moreover, the dynamics of wheeled robots such as the ARE-puck and the ARE-potato are sufficiently simple that there is little benefit to be gained by the heavy computation involved in BO. Indeed, a lot of previous research using BO to learn policies has been conducted on robot morphologies with more complex dynamics (Chatzilygeroudis et al., 2019). The result presented in this paper also directly contrasts to BEA hybrid method proposed by Lan et al. (2020): again it should be noted that the difference in the size of parameter space is considerable (18 vs 168) and in the dynamics.

	NIP-ES				NBO-ES			
	ARE-1 (*"ARE-puck"*)		ARE-2 (*"ARE-potato"*)		ARE-1 (*"ARE-puck"*)		ARE-2 (*"ARE-potato"*)	
	N	T (s)	N	T(s)	N	T(s)	N	T(s)
Escape Room	123 (142)	220 (211)	173 (160)	257 (212)	245 (99)	2482 (985)	195 (88)	2029 (426)
Middle Wall	471 (299)	715 (371)	2461 (2437)	2410 (2143)	395 (304)	2870 (888)	1040 (1335)	2457 (1385)
Multi Maze	170 (256)	290 (351)	66 (30)	116 (51)	3500 (4266)	6016 (4703)	175 (101)	1952 (403)
Easy Race	905 (544)	1454 (803)	1194 (1026)	1472 (1153)	3585 (2756)	6466 (3223)	5675 (4634)	7150 (4571)

Table 2: Average values and in parenthesis the standard deviation of number of samples (N) and of computational time (T) in seconds over the 10 replications of each experiments.

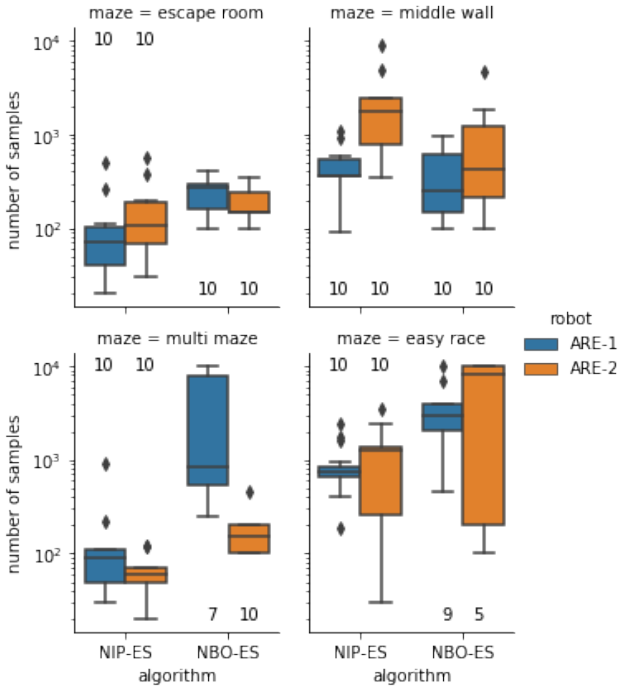


Figure 6: Box plots representing the number of samples needed for NIP-ES and NBO-ES to complete the task on the four mazes and on the both robots. The numbers under and above the boxes indicate the number of replications which have completed the task. Note the y-axis uses a logarithmic scale

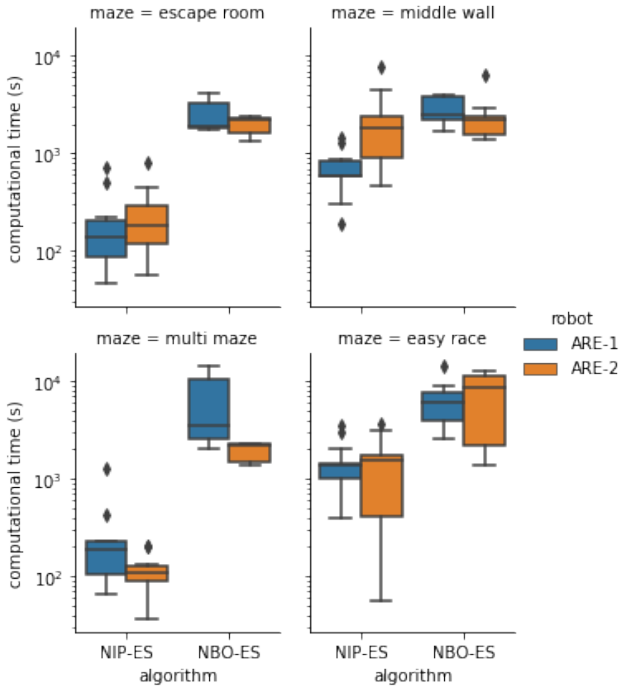


Figure 7: Box plots representing the computational time needed for NIP-ES and NBO-ES to complete the task on the four mazes and on the both robots. The y-axis is in logarithmic scale

NIP-ES is a promising method which balances exploration and exploitation, resulting in a method that is sample efficient and time efficient. The approach integrates well-understood methods: novelty search to realise exploration, IPOP-CMA-ES to realise the optimisation capability and recurrent neural networks for control. However, for the most complex set-up, i.e. easy race with both robots and middle wall arenas with the ARE-potato, NIP-ES still needs more than 1000 samples to find a successful controller. This is too large to be directly used on a physical robotic platform. Thus, although promising, there remains some room for improvement.

Conclusion

When co-evolving morphologies and controllers of real robots, it is likely that a learning phase that specialises a controller to a morphology in an individual robot is required, to mitigate against potential mis-matches between controllers and morphologies, and speed up the evolutionary process. When evolving in real-time and real-space using rapid-prototyping methods, this is particularly important to reduce the number of 'wasted' time-consuming trials. Although the wider literature provides many examples of learning approaches (Reinforcement Learning, Bayesian Optimisation, Evolutionary Algorithms), it is critical in this context that the learner is both sample and time efficient. The paper illustrates that suitably modified versions of CMA-ES are

effective and efficient in solving maze-navigation tasks on two wheeled robots. In contrast to methods such as Black-DROPS (Chatzilygeroudis et al., 2017) which achieve very high sample-efficiency by exploiting priors on the dynamics of the robot, the method proposed in this paper is applicable to morphologies produced by evolutionary methods where these priors are not known in advance. Future work will focus on improving the methods further in order to reduce the number of samples required to learn to under 100 trials, a key factor if we are to develop autonomously evolving physical robotic eco-systems in the future.

Acknowledgements

This work is funded by EPSRC ARE project, EP/R03561X, EP/R035733, EP/R035679, and the Vrije Universiteit Amsterdam.

References

Auger, A. and Hansen, N. (2005). A restart CMA evolution strategy with increasing population size. In *2005 IEEE congress on evolutionary computation*, volume 2, pages 1769–1776. IEEE.

Benazera, E. (2014). libcmaes: Multithreaded c++ 11 implementation of CMA-ES family for optimization of nonlinear non-convex blackbox functions.

Brochu, E., Cora, V. M., and De Freitas, N. (2010). A tutorial on bayesian optimization of expensive cost functions, with application to active user modeling and hierarchical reinforcement learning. *arXiv preprint arXiv:1012.2599*.

Brodbeck, L., Hauser, S., and Iida, F. (2015). Morphological evolution of physical robots through model-free phenotype development. *PloS one*, 10(6):e0128444.

Chatzilygeroudis, K. and Mouret, J.-B. (2018). Using parameterized black-box priors to scale up model-based policy search for robotics. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1–9. IEEE.

Chatzilygeroudis, K., Rama, R., Kaushik, R., Goepp, D., Vassiliades, V., and Mouret, J.-B. (2017). Black-box data-efficient policy search for robotics. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 51–58. IEEE.

Chatzilygeroudis, K., Vassiliades, V., Stulp, F., Calinon, S., and Mouret, J.-B. (2019). A survey on policy search algorithms for learning robot controllers in a handful of trials. *IEEE Transactions on Robotics*.

Conti, E., Madhavan, V., Such, F. P., Lehman, J., Stanley, K., and Clune, J. (2018). Improving exploration in evolution strategies for deep reinforcement learning via a population of novelty-seeking agents. In *Advances in neural information processing systems*, pages 5027–5038.

Cully, A., Chatzilygeroudis, K., Allocati, F., and Mouret, J.-B. (2016). Limbo: A fast and flexible library for bayesian optimization. *arXiv preprint arXiv:1611.07343*.

Deisenroth, M. and Rasmussen, C. E. (2011). Pilco: A model-based and data-efficient approach to policy search. In *Proceedings of the 28th International Conference on machine learning (ICML-11)*, pages 465–472.

Eiben, A., Bredeche, N., Hoogendoorn, M., Stradner, J., Timmis, J., Tyrrell, A. M., and Winfield, A. (2013). The triangle of life: Evolving robots in real-time and real-space. In *Artificial Life Conference Proceedings 13*, pages 1056–1063. MIT Press.

Eiben, A. and Hart, E. (2020). If it evolves, it needs to learn. In *Companion proceedings of GECCO 2020*.

Feinberg, V., Wan, A., Stoica, I., Jordan, M. I., Gonzalez, J. E., and Levine, S. (2018). Model-based value estimation for efficient model-free reinforcement learning. *arXiv preprint arXiv:1803.00101*.

Gomes, J., Mariano, P., and Christensen, A. L. (2015). Devising effective novelty search algorithms: A comprehensive empirical study. In *Proceedings of the 2015 Annual Conference on Genetic and Evolutionary Computation*, pages 943–950.

Hale, M. F., Buchanan, E., Winfield, A. F., Timmis, J., Hart, E., Eiben, A. E., Angus, M., Veenstra, F., Li, W., Woolley, R., et al. (2019). The are robot fabricator: How to (re) produce robots that can evolve in the real world. In *The 2018 Conference on Artificial Life: A Hybrid of the European Conference on Artificial Life (ECAL) and the International Conference on the Synthesis and Simulation of Living Systems (ALIFE)*, pages 95–102. MIT Press.

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

Hansen, N. (2016). The CMA evolution strategy: A tutorial. *arXiv preprint arXiv:1604.00772*.

Jelisavcic, M., De Carlo, M., Hupkes, E., Eustratiadis, P., Orlowski, J., Haasdijk, E., Auerbach, J. E., and Eiben, A. E. (2017). Real-world evolution of robot morphologies: A proof of concept. *Artificial life*, 23(2):206–235.

Kriegman, S., Blackiston, D., Levin, M., and Bongard, J. (2020). A scalable pipeline for designing reconfigurable organisms. *Proceedings of the National Academy of Sciences*.

Lan, G., Tomczak, J. M., Roijers, D. M., and Eiben, A. (2020). Time efficiency in optimization with a bayesian-evolutionary algorithm. *arXiv preprint arXiv:2005.04166*.

Lehman, J. and Stanley, K. O. (2011). Abandoning objectives: Evolution through the search for novelty alone. *Evolutionary computation*, 19(2):189–223.

Li, C., Gupta, S., Rana, S., Nguyen, V., Venkatesh, S., and Shilton, A. (2018). High dimensional bayesian optimization using dropout. *arXiv preprint arXiv:1802.05400*.

Lipson, H. (2014). Challenges and opportunities for design, simulation, and fabrication of soft robots. *Soft Robotics*, 1(1):21–27.

Pourchot, A. and Sigaud, O. (2018). Cem-rl: Combining evolutionary and gradient-based methods for policy search. *arXiv preprint arXiv:1810.01222*.

Pugh, J. K., Soros, L. B., and Stanley, K. O. (2016). Quality diversity: A new frontier for evolutionary computation. *Frontiers in Robotics and AI*, 3:40.

Rohmer, E., Singh, S. P., and Freese, M. (2013). V-rep: A versatile and scalable robot simulation framework. In *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1321–1326. IEEE.

Salimans, T., Ho, J., Chen, X., Sidor, S., and Sutskever, I. (2017). Evolution strategies as a scalable alternative to reinforcement learning. *arXiv preprint arXiv:1703.03864*.

Snoek, J., Larochelle, H., and Adams, R. P. (2012). Practical bayesian optimization of machine learning algorithms. In *Advances in neural information processing systems*, pages 2951–2959.

Stulp, F. and Sigaud, O. (2013). Robot skill learning: From reinforcement learning to evolution strategies. *Paladyn, Journal of Behavioral Robotics*, 4(1):49–61.

Tang, B. (1993). Orthogonal array-based latin hypercubes. *Journal of the American statistical association*, 88(424):1392–1397.

Vujovic, V., Rosendo, A., Brodbeck, L., and Iida, F. (2017). Evolutionary developmental robotics: Improving morphology and control of physical robots. *Artificial Life*, 23(2):169–185.

Yu, Y. (2018). Towards sample efficient reinforcement learning. In *IJCAI*, pages 5739–5743.