# Evolving-Controllers Versus Learning-Controllers for Morphologically Evolvable Robots

Miras, Karine; De Carlo, Matteo; Akhatou, Sayfeddine; Eiben, A. E.

***published in***
Applications of Evolutionary Computation
2020

***DOI (link to publisher)***
10.1007/978-3-030-43722-0_6

***document version***
Publisher's PDF, also known as Version of record

***document license***
Article 25fa Dutch Copyright Act

***citation for published version (APA)***
Miras, K., De Carlo, M., Akhatou, S., & Eiben, A. E. (2020). Evolving-Controllers Versus Learning-Controllers for Morphologically Evolvable Robots. In P. A. Castillo, J. L. Jiménez Laredo, & F. Fernández de Vega (Eds.), *Applications of Evolutionary Computation: 23rd European Conference, EvoApplications 2020, Held as Part of EvoStar 2020, Seville, Spain, April 15–17, 2020, Proceedings* (pp. 86-99). (Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics); Vol. 12104 LNCS). Springer. https://doi.org/10.1007/978-3-030-43722-0_6

# Evolving-Controllers Versus Learning-Controllers for Morphologically Evolvable Robots

Karine Miras(✉), Matteo De Carlo, Sayfeddine Akhatou, and A. E. Eiben

Vrije Universiteit Amsterdam, Amsterdam, Netherlands
k.s.m.a.dasilvamirasdearaujo@vu.nl

**Abstract.** We investigate an evolutionary robot system where (simulated) modular robots can reproduce and create robot children that inherit the parents' morphologies by crossover and mutation. Within this system we compare two approaches to creating good controllers, i.e., evolution only and evolution plus learning. In the first one the controller of a robot child is inherited, so that it is produced by applying crossover and mutation to the controllers of its parents. In the second one the controller of the child is also inherited, but additionally, it is enhanced by a learning method. The experiments show that the learning approach does not only lead to different fitness levels, but also to different (bigger) robots. This constitutes a quantitative demonstration that changes in brains, i.e., controllers, can induce changes in the bodies, i.e., morphologies.

**Keywords:** Morphological evolution · Life-time learning · Evolutionary Robotics · Modular robots

## 1 Introduction

In the field of Evolutionary Robotics, evolving the controllers of robots has been much more explored than evolving their morphologies [4,5]. This is not surprising, considering that the challenge of evolving both morphology and controller is much greater than evolving the controller alone. In case of the joint evolution of morphologies and controllers there are two search spaces and the search space for the controllers changes with every new robot morphology produced. This challenge was firstly explored in Sims' seminal work [15], and more recently in multiple studies [6,10,11,16].

In the present study we consider morphologically evolving robot populations. Our main goal is to investigate the effects of life-time learning in these populations. To this end, we set up a system where (simulated) modular robots can reproduce and create offspring that inherits the parents' morphologies by crossover and mutation. Regarding the controllers, we implement and compare two methods. Method 1 works by evolving the robot controllers. In this method, controllers are inheritable, where the controller of the offspring is produced by applying crossover and mutation to the controllers of the parents. In Method 2, controllers are not only inheritable (hence, evolvable), but also learnable. In this method, the controller of the offspring is produced by a learning method that starts with the inherited brain.

The specific research questions we are to answer here are as follows:

– How does life-time learning affect evolvability?
– How does life-time learning affect the morphological properties of the population?
– How does life-time learning drive the course of evolution?

## 2 Methodology

The datasets and code of this study are stored at ssh.data.vu.nl in the karinemiras-evostar2020 directory, and can be accessed given administrative request.

### 2.1 Morphology

We are using simulated robots based on RoboGen [2] whose morphologies ("morphologies") are composed of modules shown in Fig. 1. Any module can be attached to any other module through its attachable slots, except for the sensors, which can not be attached to joints. Our morphologies consist of a single layer, i.e., the modules do not allow attachment on the top or bottom slots, only on the lateral ones, but the joints can bend, so the robots can 'stand' in a 3D-shape. Each module type is represented by a distinct symbol in the genotype.

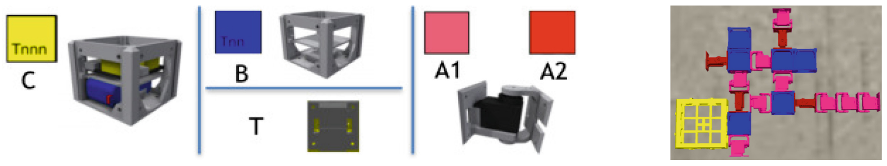

**Fig. 1.** At the left, the robot modules: Core-component with controller board (C); Structural brick (B); Active hinges with servo motor joints in the vertical (A1) and horizontal (A2) axes; Touch sensor (T). C and B have attachment slots on their four lateral faces, and A1 and A2 have slots on their two opposite lateral faces; T has a single slot which can be attached to any slot of C or B. The sequence of letters (T or n) in C and B indicate if there is a sensor on the laterals left, front, right and back (for C only), in this order. At the right, an example of robot in simulation.

### 2.2 Controller

The controller ("controller") is a hybrid artificial neural network, which we call Recurrent CPG Perceptron (Fig. 2, right).

For every joint in the morphology, there exists a corresponding oscillator neuron in the network, whose activation function is calculated through a Sine wave with three parameters: Phase offset, Amplitude, and Period. The oscillators are not interconnected, and every oscillator may or may not possess a direct recurrent connection. Additionally, every sensor is reflected as an input for the network, which might connect to one or more oscillators, having the weights of its connections ranging from $-1$ to 1. The CPG [9] generates a constant pattern of movement, even if the robot is not sensing anything, so that the sensors are used either to suppress or to reinforce movements.

**Representation and Operators.** We use an evo-devo style generative encoding to represent the robots. Specifically, our genomes –that encode both morphology and controller– are based on a Lindenmayer-System (L-system) inspired by [8]. The grammar of an L-System is defined as a tuple $G = (V, w, P)$, where

- $V$, the alphabet, is a set of symbols containing replaceable and non-replaceable symbols.
- $w$, the axiom, is a symbol from which the system starts.
- $P$ is a set of production-rules for the replaceable symbols.

The following didactic example illustrates the process of iterative-rewriting of an L-System. For a given number of iterations, each replaceable symbol is simultaneously replaced by the symbols of its production-rule. Given $w = X$, $V = \{X, Y, Z\}$ and $P = \{X : \{X, Y\}, Y : \{Z\}, Z : \{X, Z\}\}$, the rewriting goes as follows.

$$\text{Iteration 0: } X$$
$$\text{Iteration 1: } XY$$
$$\text{Iteration 2: } XYZ$$
$$\text{Iteration 3: } XYZXZ$$

In our system each genotype is a distinct grammar in the syntax specified by the types of modules we have. The alphabet is formed by symbols denoting the morphological modules and commands to attach them together, as well as commands for defining the structure of the controller. The construction of a phenotype (robot) from a genotype (grammar) is done in two stages. In the first stage (early development), the axiom of the grammar is rewritten into a more complex string of symbols (intermediate phenotype), according to the production-rules of the grammar. (Here we set the number of iterations to 3). In the second stage (late development), this string is decoded into a phenotype. The second stage of this process is illustrated in Fig. 2. The first stage was omitted because it is somewhat extensive, but it follows work flow shown in the example above. During the second stage of constructing a phenotype two positional references are always maintained in it, one for the morphology (pointing to the current module) and one for the controller (pointing to the current sensor and the current oscillator). The application of the commands happens in the current module in the case of the morphology, while for the controller it happens in (or between) the current sensor and the current oscillator. More details about the representation can be found in [12,13].

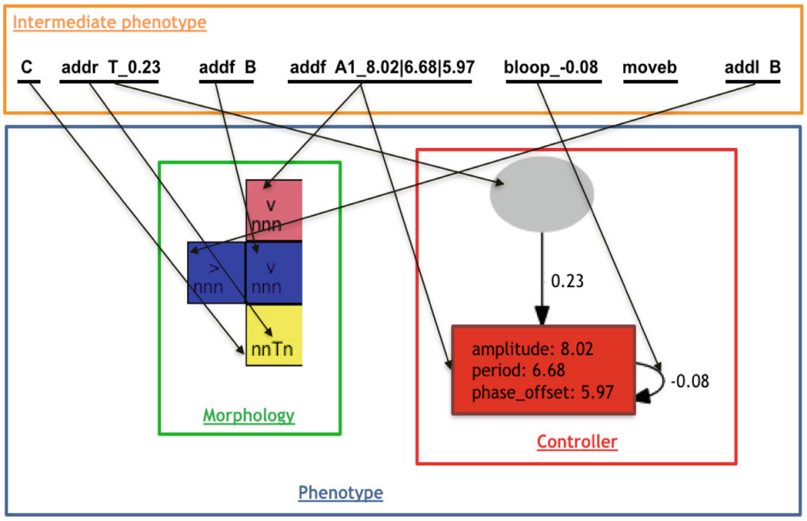

**Fig. 2.** Process of late development: decoding an intermediate phenotype into a (final) phenotype with morphology and controller.

The initialization of a genotype adds, to each production rule, one random (uniformly) symbol of each the following categories, in this order: Controller-moving commands, and Controller-Changing commands, Morphology-mounting commands, Modules, Morphology-moving commands. This can be repeated for $r$ times, being $r$ sampled from a uniform random distribution ranging from 1 to $e$. This means that each rule can end up with 1 or maximally $e$ sequential groups of five symbols (here $e$ is set to 3). The symbol C is reserved to be used exclusively at the beginning of the production rule C.

The crossovers are performed by taking complete production-rules randomly (uniform) from the parents. Finally, individuals undergo mutation by adding, deleting, or swapping one random (uniform) symbol from a random production-rule/position. All symbols have the same chance of being removed or swapped. As for the addition of symbols, all categories have equal chance of being chosen to provide a symbol, and every symbol of the category also has equal chance of being chosen. An exception is always made to $C$ to ensure that a robot has one and only one core-component. This way, the symbol $C$ is added as the first symbol of the $C$ production rule, and can not be added to any other production rules, neither removed or moved from the production rule of $C$.

Once it is possible that only the rules of one single parent end up being expressed in the final phenotype, and also as it is not rare that one mutation happens for non-expressed genes, both crossover and mutation probabilities were set high, to 80%, aiming to minimize this effect.[1]

For practical reasons (simulator speed and physical constructability) we limit the number of modules allowed in a robot to a maximum of 100.

## 2.3 Morphological Descriptors

For quantitatively assessing morphological properties of the robots, we utilized the following set of descriptors:

1. **Size:** Total number of modules in the morphology.
2. **Relative Number of Limbs:** The number of extremities of a morphology relative to a practical limit. It is defined with Eq. (1)

$$L = \begin{cases} \frac{l}{l_{max}}, & \text{if } l_{max} > 0 \\ 0 & \text{otherwise} \end{cases}$$
$$l_{max} = \begin{cases} 2 * \lfloor \frac{(m-6)}{3} \rfloor + (m-6) \pmod 3 + 4, & \text{if } m \geq 6 \\ m-1 & \text{otherwise} \end{cases} \tag{1}$$

where $m$ is the total number of modules in the morphology, $l$ the number of modules which have only one face attached to another module (except for the core-component) and $l_{max}$ is the maximum amount of modules with one face attached that a morphology with $m$ modules could have, if containing the same amount of modules arranged in a different way (Fig. 3).

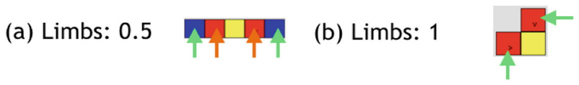

**Fig. 3.** Morphology (a) has four modules that could be extremities (considering the limit determined by the size of the morphology), but only the two indicated by green arrows are; (b) has the maximum number of extremities it could have. (Color figure online)

3. **Relative Length of Limbs:** The length of limbs relative to a practical limit. It is defined with Eq. (2):

$$E = \begin{cases} \frac{e}{e_{max}}, & \text{if } m \geq 3 \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

[1] This means that around 80% of the offspring will be result of crossovers, and also that around 80% of the offspring will suffer the above explained mutation.

where $m$ is the total number of modules of the morphology, $e$ is the number of modules which have two of its faces attached to other modules (except for the core-component), and $e_{max} = m - 2$ – the maximum amount of modules that a morphology with $m$ modules could have with two of its faces attached to other modules, if containing the same amount of modules arranged in a different way[2] (Fig. 4).

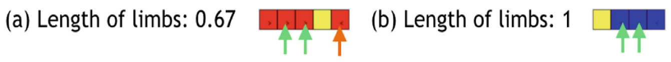

**Fig. 4.** While in morphology (b) the maximum possible quantity of modules was used as the extension of a limb, in (a), the module indicated by an orange arrow was used as an extra limb. (Color figure online)

4. **Proportion:** The length-width ratio of the rectangular envelope around the morphology. It is defined with Eq. (3):

$$P = \frac{p_s}{p_l} \tag{3}$$

where $p_s$ is the shortest side of the morphology, and $p_l$ is the longest side, after measuring both dimensions of length and width of the morphology (Fig. 5).

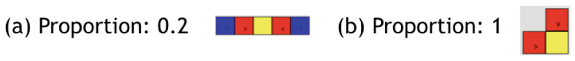

**Fig. 5.** Morphology (a) is disproportional and (b) is proportional.

A complete search space analysis of the utilized robot framework and its descriptors is available in [12,13], demonstrating the capacity of these descriptors to capture relevant robot properties, and proving that this search space allows high levels of diversity.

## 2.4 Evolution

We are using overlapping generations with population size $\mu = 100$. In each generation $\lambda = 50$ offspring are produced by selecting 50 pairs of parents through binary tournaments (with replacement) and creating one child per pair by crossover and mutation. From the resulting set of $\mu$ parents plus $\lambda$ offspring, 100 individuals are selected for the next generation, also using binary tournaments. The evolutionary process is stopped after 30 generations, thus all together

[2] The types of modules would not have to be necessarily the same, as long as the morphology had the same amount of modules.

we perform 1.550 fitness evaluations per run. For each environmental scenario the experiment was repeated 10 times independently.

The task used was undirected locomotion, and the fitness utilized was the speed (cm/s) of the robot's displacement in any direction, as defined by Eq. 4.

$$s_x = \frac{e_x - b_x}{t} \tag{4}$$

where $b_x$ is $x$ coordinate of the robot's center of mass in the beginning of the simulation, $e_x$ is $x$ coordinate of the robot's center of mass at the end of the simulation, and $t$ is the duration of the simulation.

## 2.5 Learning

The life-time learning of the robots was carried out by optimizing the parameters of the oscillators of their controllers (Fig. 2, right) using the algorithm CMA-ES [7]. The $\mu$ (population size) and $\lambda$ (offspring size) values are defined according to the dimension $N$ of the controller (number of oscillators in the controller and its parameters), and is defined as with Eqs. 5 and 6, respectively.

$$\lambda = 4 + \lfloor 3 \times log(N) \rfloor \tag{5}$$

$$\mu = \lfloor \lambda/2 \rfloor \tag{6}$$

These parameters were chosen based on [7], and the maximum number of evaluations was set to be at least 100. Because not always $\lambda$ is a divisor of 100, some runs can have a few more evaluations than that.

Given that we decided to experiment with evolution-only versus learning-only, the initial mean and standard deviation of the multivariate normal distributions were defined randomly, instead of derived from the parameters that the L-System defines.

## 2.6 Experimental Setup

All experiments were carried out on a plane flat floor with no obstacles. While the morphologies were evolving in all of the experiments, we tested two different methods for optimizing controllers: Method 1 works by evolving the robot controllers. In this system, controllers are inheritable, whereas the controller of the offspring is produced by applying crossover and mutation to the controllers of the parents. We refer to this method as Evolvable throughout the paper. In Method 2, controllers are not only inheritable (hence, evolvable), but also learnable. In this method, the controller of the offspring is produced by a learning method that starts with the inherited brain. We refer to this method as Learnable throughout the paper.

## 3 Results and Discussion

### 3.1 Evolvability and the Production Costs

As expected, adding a life-time learning capacity to the system increased the speed of the population, as depicted by Figs. 6 and 7. This was expected for two reasons: (a) the number of evaluations performed by Learnable is around 100 times higher than by Evolvable; (b) in Learnable, robots have time to fine-tune their controllers to the morphologies they were born with.

On the other hand, while the average speed of Learnable seems to be going to flatten soon, the average speed of Evolvable keeps growing. Nevertheless, this is not the aspect we are interested in discussing. Instead, we are interested in observing which method presents a faster growth of the average speed. In Fig. 6, the black line shows that around generation 9, Learnable had already obtained an average speed that took the whole evolutionary period, i.e., 30 generations, for Evolvable to achieve. Of course, we should not neglect that, as previously mentioned, Learnable at generation 9 had already spent around 55.000 evaluations, while at the final generation Evolvable spent only 1.550. However, we should consider that Learnable at generation 9 created only 550 robots while Evolvable created 1.550, i.e., around 3 times more. If we consider real physical robots, and assuming that the production cost of each robot is substantially higher than the evaluation cost, we can clearly see the advantage of introducing learning. For instance, the evaluation cost could be around 30 s and creation cost around 4 h. In this case, Evolvable would take 327.775 min versus 159.500 min for Learnable, representing a difference of 116 days.

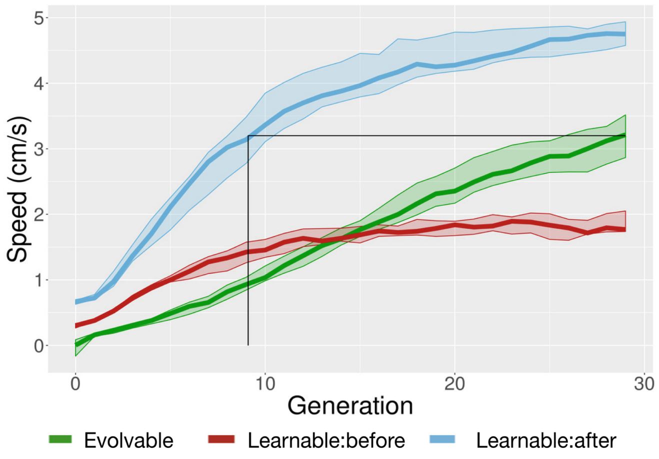

**Fig. 6.** Speed: progression of the mean of the population (quartiles over all runs). Black lines mark generation (9), when the Learnable method (after learning) achieved the levels of speed that the Evolvable method managed to achieve only in the end of the evolutionary period

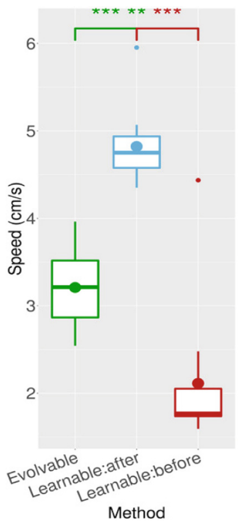

**Fig. 7.** Comparison of speed in the final generations. The method Learnable:before is equivalent to a random controller. Significance levels for the Wilcoxon tests in the boxplots are $* < 0.05$, $** < 0.01$, $*** < 0.001$, while $NS$ means non-significant.

## 3.2 Morphological Properties

In [12], a study utilizing this same robot framework observed a strong selection pressure for robots with few limbs, most often one single long limb, i.e., a snake-like morphology. Furthermore, they demonstrated that by explicitly adding a penalty to having this morphological property, the population did indeed develop multiple limbs, nevertheless, these robots were much slower than the single-limb ones.

Interestingly, we have observed in nature that "for many animals, natural selection may tend to favor structures and patterns of movement that increase maximum speed", and, "in almost every case, legged animals can move faster over land than animals of similar size that lack legs" [1]. Given such notions, in [12] it was hypothesized the following, concerning few long limbs having shown to be a predominant morphological property: "it might be due, not to some advantage of having fewer limbs, but to the challenge of having multiple limbs. For example, having one limb that permits locomotion is a challenge in itself, while having multiple limbs not only multiplies this challenge but also carries an additional challenge of synchronization, to avoid limbs pulling in different directions and impairing displacement. Perhaps adding a life-time learning ability to the robots would allow them to learn how to use their limbs better and obtain higher speed."

Though such a hypothesis seems plausible, our experiments have proven it wrong. Figures 8 and 9 show the comparison between the methods for the emergent morphological properties in the population. We see that the average Num-

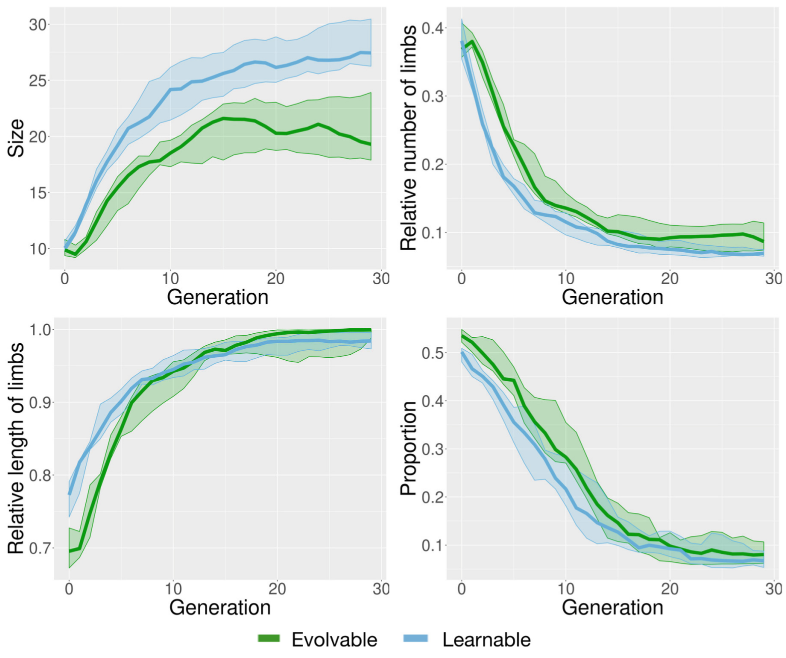

**Fig. 8.** Morphological properties: progression of the mean of the population (quartiles over all runs).

bers of Limbs, Length of Limbs, and Proportion in Learnable converge to non significantly different values than in Evolvable. In summary, with both methods, the population converged to big, disproportional robots that have few, long limbs (Fig. 10).

Despite all these similarities, there was one morphological property that showed to be different, i.e., Size, so that robots in Learnable are significantly bigger than in Evolvable. A video showing examples of robots from both types of experiments can be found in https://www.youtube.com/watch?v=szwZvJnEfYw.

## 3.3 Morphological Exploitation Through Learning

In Fig. 11 we see that the average learning $\Delta$ of the method Learnable, i.e., average speed after the parameters were learned minus average speed before the parameters were learned, grows across the generations. This growth is rather quick up to generation 15, from when becomes more moderate. Not coincidentally, it is also from generation 15 that the curves of the average morphological properties started to flatten out. These observations suggest that the life-time learning led the evolutionary search to more quickly exploit the high performing morphological properties. In other words, it was faster for the population to turn into morphologies that are big, disproportional, with few, long limbs.

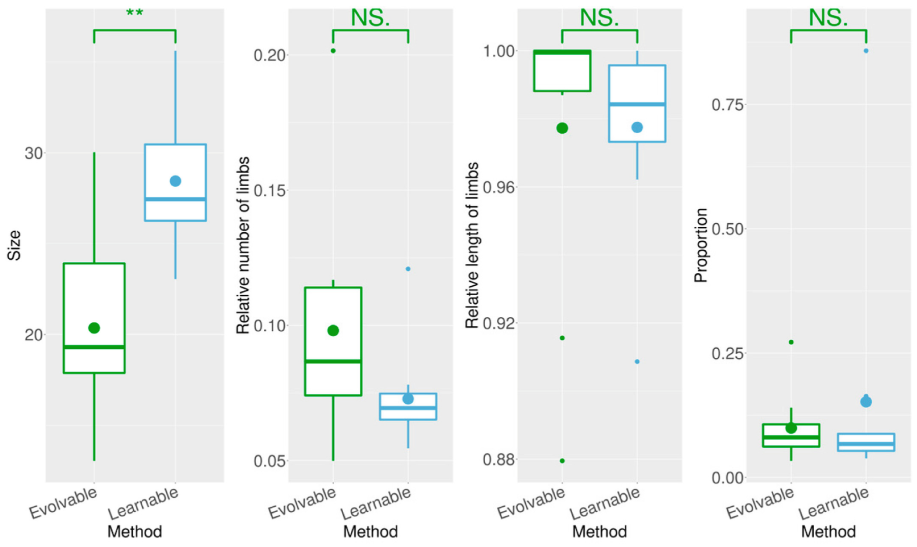

**Fig. 9.** Comparison between the morphological properties in the final generations. Significance levels for the Wilcoxon tests in the boxplots are $* < 0.05$, $** < 0.01$, $* * * < 0.001$, while $NS$ means non-significant.

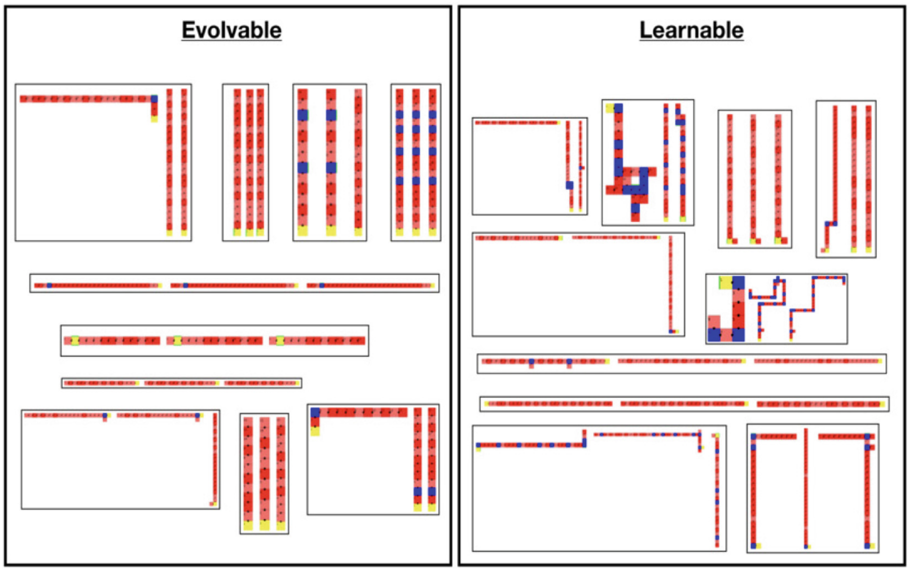

**Fig. 10.** The three best robots of each run for both control methods. Figures were scaled to fit the frame.

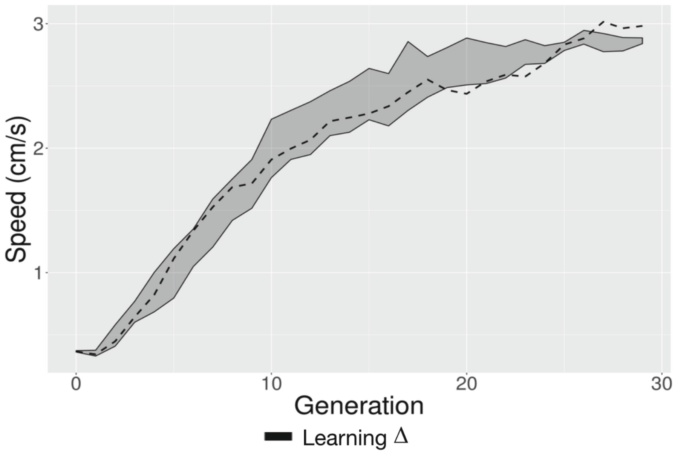

**Fig. 11.** Learning $\Delta$, i.e., average speed after the parameters were learned minus average speed before the parameters were learned. Progression of the mean of the population (quartiles over all runs).

## 4 Conclusions and Future Work

The main goal of this paper was to investigate the effects of life-time learning in populations of morphologically evolving robots. To this end, we have set up a system where (simulated) modular robots can reproduce and create offspring that inherits the parents' morphologies by crossover and mutation. In this system we implemented two options to produce controllers (for the task of locomotion). In the evolutionary method controllers are inheritable, where the controller of the offspring is produced by applying crossover and mutation to the controllers of the parents. In the learning method controllers of the offspring are also inheritable, but additionally, they are fine-tuned by a learning algorithm (specifically, we employed the CMA-ES as a learner). Conducting experiments with both methods we obtained answers to our research questions.

Firstly, if we measure time by the number of generations, learning boosts evolvability in terms of efficiency as well as efficacy, i.e., solution quality at termination, once its growth curve was steeper and ended higher than that of the evolutionary method. Of course, this is not a surprise, since the learning version performs much more search steps. However, a learning trial (testing another controller) is much cheaper than an evolutionary trial (making another robot), so we can firmly conclude the advantage of adding lifetime-learning to an evolutionary robot system.

Secondly, we have witnessed a change in the evolved morphologies when lifetime learning was applied. In particular, the sizes at the end of evolution were clearly different (while the shapes were not). We find this the most interesting outcome because it is the opposite of the well-known effect of how the body shapes the brain [14]. Our results show how the brains can shape the bodies through affecting task performance that in turn changes the fitness values that define selection probabilities during evolution. As far as we know, previously this has been demonstrated and documented only once in an artificial evolutionary context [3].

For future work, we aim to look more deeply into the "how the brain shapes the body" effect. To this end, we are extending the morphological search space by allowing more complex body shapes. Hereby we hope to introduce more regions of attraction in the morphological search space, such that the snakes are not the most dominant life forms and evolution can converge to various shapes. Last but not least, we are working on a Lamarckian combination of evolution and life-time learning, where (some of) the learned traits are inheritable.

## References

1. Alexander, R.M.: Principles of Animal Locomotion. Princeton University Press, Princeton (2003)
2. Auerbach, J., et al.: RoboGen: robot generation through artificial evolution. In: Artificial Life 14: Proceedings of the Fourteenth International Conference on the Synthesis and Simulation of Living Systems, pp. 136–137. The MIT Press (2014)
3. Buresch, T., Eiben, A.E., Nitschke, G., Schut, M.: Effects of evolutionary and lifetime learning on minds and bodies in an artificial society. In: Proceedings of the IEEE Conference on Evolutionary Computation (CEC 2005), pp. 1448–1454. IEEE Press (2005). http://www.cs.vu.nl/~schut/pubs/Buresch/2005.pdf
4. Doncieux, S., Bredeche, N., Mouret, J.B., Eiben, A.: Evolutionary robotics: what, why, and where to. Frontiers Robot. AI **2**, 4 (2015)
5. Floreano, D., Husbands, P., Nolfi, S.: Evolutionary robotics. In: Siciliano, B., Khatib, O. (eds.) Springer Handbook of Robotics, Part G. 61, pp. 1423–1451. Springer, Heidelberg (2008). https://doi.org/10.1007/978-3-540-30301-5_62
6. Hamann, H., Stradner, J., Schmickl, T., Crailsheim, K.: A hormone-based controller for evolutionary multi-modular robotics: from single modules to gait learning. In: 2010 IEEE World Congress on Computational Intelligence, WCCI 2010 - 2010 IEEE Congress on Evolutionary Computation, CEC 2010 (2010)
7. Hansen, N., Akimoto, Y., Brockhoff, D., Chan, M.: CMA-ES/pycma: r2.7.0, April 2019. https://doi.org/10.5281/zenodo.2651072
8. Hornby, G.S., Pollack, J.B.: Body-brain co-evolution using L-systems as a generative encoding. In: Proceedings of the 3rd Annual Conference on Genetic and Evolutionary Computation, pp. 868–875. Morgan Kaufmann Publishers (2001)
9. Ijspeert, A.J.: Central pattern generators for locomotion control in animals and robots: a review. Neural Netw. **21**(4), 642–653 (2008)
10. Jelisavcic, M., Kiesel, R., Glette, K., Haasdijk, E., Eiben, A.: Analysis of Lamarckian evolution in morphologically evolving robots. In: Artificial Life Conference Proceedings, vol. 14. pp. 214–221. MIT Press (2017)
11. Marbach, D., Ijspeert, A.J.: Co-evolution of configuration and control for homogenous modular robots. In: Proceedings of the Eighth Conference on Intelligent Autonomous Systems (IAS8), pp. 712–719 (2004)
12. Miras, K., Haasdijk, E., Glette, K., Eiben, A.E.: Effects of selection preferences on evolved robot morphologies and behaviors. In: Proceedings of the Artificial Life Conference, Tokyo (ALIFE 2018). MIT Press (2018)

13. Miras, K., Haasdijk, E., Glette, K., Eiben, A.E.: Search space analysis of evolvable robot morphologies. In: Sim, K., Kaufmann, P. (eds.) EvoApplications 2018. LNCS, vol. 10784, pp. 703–718. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-77538-8_47
14. Pfeifer, R., Bongard, J.: How the Body Shapes the Way We Think: A New View of Intelligence. MIT Press, Cambridge (2006)
15. Sims, K.: Evolving 3D morphology and behavior by competition. Artif. Life **1**(4), 353–372 (1994)
16. Sproewitz, A., Moeckel, R., Maye, J., Ijspeert, A.J.: Learning to move in modular robots using central pattern generators and online optimization. Int. J. Robot. Res. **27**(3–4), 423–443 (2008)