Evolving Hebbian Learning Rules in Voxel-based Soft Robots

Andrea Ferigo, Giovanni Iacca, Eric Medvet, Federico Pigozzi

***Abstract*—According to Hebbian theory, synaptic plasticity is the ability of neurons to strengthen or weaken the synapses among them in response to stimuli. It plays a fundamental role in the processes of learning and memory of biological neural networks. With plasticity, biological agents can adapt on multiple timescales and outclass artificial agents, the majority of which still rely on static Artificial Neural Network (ANN) controllers. In this work, we focus on Voxel-based Soft Robots (VSRs), a class of simulated artificial agents, composed as aggregations of elastic cubic blocks. We propose a Hebbian ANN controller where every synapse is associated with a Hebbian rule that controls the way the weight is adapted during the VSR lifetime. For a given task and morphology, we optimize the controller for the task of locomotion by evolving, rather than the weights, the parameters of the Hebbian rules. Our results show that the Hebbian controller is comparable, often better than a non-Hebbian baseline and that it is more adaptable to unforeseen damages. We also provide novel insights into the inner workings of plasticity and demonstrate that "true" learning does take place, as the evolved controllers improve over the lifetime and generalize well.**

***Index Terms*—Hebbian learning, synaptic plasticity, voxel-based soft robots, evolutionary robotics.**

I. Introduction

An ambitious goal of robotics is to develop autonomous robotic societies [1, 2], capable of supporting their own existence in hazardous and exotic environments while achieving a human-aligned target. In such a scenario, adaptability is pivotal, since robotic agents are called upon dealing with changing environmental conditions and, possibly, changing mission requirements. Moreover, it is not unlikely that future robotic agents would undergo the same stages of life (birth, maturity, death or disposal) of biological agents [3]. As such, learning and adaptation should inherently guide "infant" robots towards maturity [4], just like in animals. Indeed, learning has been acknowledged as one of the main challenges in the road towards fully autonomous robotic ecosystems [5].

For robotic societies to become reality, adaptation must take place across all the three timescales: *phylogenetic* (evolution), *ontogenetic* (development), and *epigenetic* (learning) [6]. As of now, the vast majority of robotic systems are either evolved [7] or learned [8]. While some works at the intersection of evolution and learning do exist [9], they mostly rely on a human-assigned reward signal that must be decided by a "supervisor", and thus might not be adequate for an autonomous robotic society. On the other side, biological agents share both an "innate" evolved component and a learned component. A robot that only evolves, albeit powerful evolution might be, can in principle beget offspring that are not viable. As a matter of example, when jointly evolving brain and body, mismatch can happen between the two [10]. A robot that only learns, without an innate evolved "instinct", since learning usually relies on trial-and-error, incurs the risk of damaging itself and the entities surrounding it [11]. For instance, an autonomous vehicle can provoke serious damage if it optimizes a policy by sampling random actions. Then, there is a need for robotic systems that adapt on more than one timescale, quelling the longstanding "nature vs. nurture" debate [12].

Among the shapes learning can take, we are concerned with plasticity [13], since it is "unsupervised" in the sense that it does not require a human-assigned reward signal. Additionally, plasticity is known to play a key role in the learning processes of biological agents [14] and it is a simple yet effective model. Hebbian theory [15], which states that "neurons that fire together, wire together", enshrines it. Synapses between neurons are plastic, i.e., they adapt their strength in response to stimuli, following synapse-specific rules known as "Hebbian rules". These rules are in turn expressed through well-defined parameters, according to the model employed [16].

In this work, we consider Voxel-based Soft Robots (VSRs), aggregations of elastic cubic blocks made of soft material, that actuate by contracting or expanding their volume [17]. VSRs have proven capable of amazing feats, like modeling synthetic organisms [18, 19], squeezing through tight spaces [20], and swimming in aquatic environments [21]. Thanks to their intrinsic modularity, they have emerged as a relevant formalism for designing state-of-the-art soft robotics systems [22]. Modularity is a desirable property for robotic systems [23], as it makes the morphological search space compact without harming expressivity; indeed, interesting processes (e.g., development [24]) are still plausible.

We apply Hebbian learning to an Artificial Neural Network (ANN) controlling the VSR. We embed Hebbian learning inside an Evolutionary Algorithm (EA) that optimizes the parameters of the Hebbian rules; in this sense, there are two scales of adaptation: evolution and learning, the former being the longer and slower one. Indeed, Evolutionary Computation (EC) [25] married with robotics to beget the field of evolutionary robotics this work belongs to, in which many successes

Andrea Ferigo and Giovanni Iacca are with the Department of Information Engineering and Computer Science, University of Trento, Trento, Italy. (email: andrea.ferigo@unitn.it and giovanni.iacca@unitn.it)
Eric Medvet and Federico Pigozzi are with the Department of Engineering and Architecture, University of Trieste, Trieste, Italy. (email: emedvet@units.it and federico.pigozzi@phd.units.it)
All authors contributed equally and are listed in alphabetical order.

have been achieved [26, 27].

We evolve Hebbian learning rules for two fixed VSR morphologies in a locomotion task on hilly terrains, and re-assess the VSRs performance on unseen terrains, in order to quantify their ability to generalize. With experiments, we:

(a) confirm that the Hebbian controller is never worse (and often better) than a non-Hebbian baseline, and truly shines when adapting to unforeseen damages in its body;
(b) verify that Hebbian controllers indeed learn, i.e., improve over the lifetime and can generalize;
(c) unveil some of the weight dynamics underlying learning.

This work falls in the more general study of the link between learning and evolution, which is rooted in both biology and artificial intelligence, with fundamental consequences on robotics. In this regard, a recent position paper [4] argued that evolutionary robots systems should always "contain a learning component where a newborn robot refines its inherited controller to align with its body, which will inevitably be different from its parents". The authors summarized this concept with the motto: "if it evolves it needs to learn". As already pointed out in [28], this paradigm can be shifted though, as we do here, where we show that the learning-evolution link is actually bidirectional. In our experiments, not only the evolved agents need to have a learning phase, but also the learning strategy (i.e., the parameters of the Hebbian rules) needs to evolve over the course of the generations. In other words, learning emerges through evolution. We can summarize this concept by paraphrasing the aforestated motto as follows: *if it evolves it needs to learn, but if it learns, it needs to evolve*.

The rest of the paper is structured as follows. The next section summarizes the related works. Section III details the models of the VSRs, Hebbian learning, and the evolutionary algorithm used in our experimentation. Section IV presents our experimental results. Finally, in Section V we draw the main conclusions of this study.

II. Related works

Neural plasticity [13], a form of learning, plays a key role in the development of biological neural networks. Intuitively, it is crucial in adapting to changes in environmental conditions, as well as in shaping memories [14]. Indeed, starting from [29], there has been growing evidence about the Baldwin effect, i.e., an acceleration of evolution when learning happens during lifetime. In a seminal computational study, Hinton et al. [30] showed that learning can provide a gradient for evolution to follow even on an extremely deceptive fitness landscape.

Following [31], we distinguish between structural and functional plasticity. The former refers to the ability of the nervous system to rewire its neural connections, creating new pathways among neurons and undoing existing ones. The latter refers to the ability of the nervous system to alter over time the functional properties of neurons. In this work, we are concerned with functional plasticity and, in particular, with changes in synaptic strength in response to previous activity (activity-dependent plasticity) [32]. Previous activities can induce persistent strengthening of synapses (long-term potentiation) [33] or persistent weakening (long-term depression) [34].

As a result of the well-known benefits of plasticity, there has been a considerable amount of literature devoted to engineering "plastic" ANNs. Schmidhuber [35] first proposed "fast weights", where a slow-learning ANN learns the weights of a fast-learning ANN, as a thought experiment. In the last decade, new studies have taken inspiration from fast weights, including adaptive HyperNEAT [36], fast weights for recurrent neural networks [37], and hypernetworks [38]. Moreover, there is a growing amount of research on the attention mechanism [39, 40], which can be seen as a form of "adaptive weights", and synaptic pruning of ANNs [41, 42], which can be seen as a very "sharp" form of structural plasticity.

However creative and ground-breaking these works might have been, they are still very complex to engineer. Hebbian learning [16] provides a more succinct, yet biologically-plausible representation. According to Hebbian theory [15], if a pre-synaptic neuron often stimulates the activation of a post-synaptic one, then their synapse increases in strength[1]. Past studies [44, 45] applied Hebbian learning to evolve ANN controllers for mobile robots, and found improved performance over non-plastic ANNs. More recently, several studies have successfully evolved Hebbian learning rules [46, 47, 48, 49] and achieved competitive results in reinforcement learning scenarios. For these reasons, we adopt Hebbian learning to model synaptic plasticity.

The present work is built upon our previous research on VSRs. These form a class of robotic agents that, being intrinsically modular and re-configurable, allow great freedom to the designer. Previous works have focused on the optimization of the controller [50], the morphology [51], the morphology jointly with the controller [52], or even the sensory apparatus [53]. In this work, we study the controller optimization and focus on a Hebbian controller. To the best of our knowledge, this is the first work on applying Hebbian learning to VSRs.

III. Methods

A. Background: voxel-based soft robots

Voxel-based Soft Robots (VSRs) are a kind of modular robots composed as aggregations of elastic cubic blocks (*voxels*), made of soft material. Each voxel can contract or expand its volume; it is the overall concert of volume changes that allows for the emergence of the high-level behavior of the robot. VSRs were first formalized and fabricated in [17]. In this work, we consider a 2-D variant of simulated (in discrete time and continuous space) VSRs [54], which approximate cubes with squares. While disregarding one dimension certainly makes these simulated VSRs less realistic, it also eases the optimization of VSR design, thanks to the smaller search space. We remark, however, that the representation and the algorithms adopted in this paper are easily portable to the 3-D setting, and so are the considerations on the Hebbian controller.

[1]This statement should be read with a pinch of salt: if two neurons activate at the same time, then there cannot be causation between the two activations. For causation to subsist, it must happen that one neuron took part in activating the other [43]. In this work, we consider connectionist ANNs, where there is no need to take into account the temporal dimension.

In the following, we outline the characteristics of VSRs relevant to this study, and refer the reader to [54] for more details. A VSR is completely defined by its *morphology* (i.e., the body) and its *controller* (i.e., the brain), that we explain in more detail in the following two subsections.

1) VSR morphology: The morphology of a VSR describes how the voxels, i.e., deformable squares, are arranged in a grid topology of size $w \times h$. We model each voxel as the assembly of spring-damper systems, masses, and distance constraints [54]. Each voxel is rigidly connected to its four adjacent voxels (if present).

Over time, voxels change their area according to (a) external forces acting on the voxel (e.g., other bodies, including other voxels) and (b) an actuation signal computed by the controller. The latter produces a contraction/expansion force that is modeled in the simulation as an instantaneous change in the resting length of the spring-damper systems of the voxel. The length change is linearly dependent on an *actuation value* residing in $[-1, +1]$, -1 being the greatest possible expansion and $+1$ being the greatest possible contraction.

A VSR can be equipped with sensors and each voxel can have one or more sensors. A sensor outputs, for every time step, a *sensor reading* $\boldsymbol{s} \in \mathbb{R}^m$, where m is the dimensionality of the sensor type. In this study, we endow VSRs with four different types of sensors. Area sensors sense the ratio between the current area of the voxel and its resting area ($m = 1$). Touch sensors sense whether the voxel is currently touching another body different from the robot itself (e.g., the ground) or not, and output a value of 1 or 0, respectively ($m = 1$). Velocity sensors sense the speed of the center of mass of the voxel along the x- and y-directions ($m = 2$). Lidar sensors sense the distance from the center of mass of the voxel to the closest object along a predefined set of directions; if no object is within the sensor range d, the sensor reading is set to d. We use $d = 10\,\mathrm{m}$ (the side of the voxel being $3\,\mathrm{m}$ long) and the following directions with respect to the positive x-axis: $-\frac{1}{4}\pi$, 0, $\frac{1}{4}\pi$ (so $m = 3$). We normalize all sensor readings into $[0, 1]^m$. After normalization, to simulate real-world sensor noise, we perturb every sensor reading with additive Gaussian noise with mean 0 and variance σ^2_{noise}. We set $\sigma_{\text{noise}} = 0.01$.

2) VSR controller: Let n be the number of voxels of the VSR and let $\boldsymbol{s}^{(k)} = [\boldsymbol{s}_1\ \boldsymbol{s}_2\ \dots]$ be the concatenation of sensor readings for all the VSR sensors at time step k, i.e., at time step $t = k\Delta t$, where Δt is the interval between two simulation time steps. The controller is, in general, a dynamical system that takes, at each time step k, $\boldsymbol{s}^{(k)}$ as input and outputs $\boldsymbol{a}^{(k)} \in [-1, +1]^n$, i.e., the actuation values for the voxels.

In this work, we employ ANNs as controllers. ANNs are indeed known to be universal function approximators [55] and have proven effective at exploiting sensors for robotic tasks [50]. We denote by $f_{\boldsymbol{\theta}}$ an instance of an ANN-based controller, such that $\boldsymbol{a}^{(k)} = f_{\boldsymbol{\theta}}\left(\boldsymbol{s}^{(k)}\right)$. An ANN-based controller is completely specified by the parameters $\boldsymbol{\theta} \in \mathbb{R}^p$ of the ANN. Thus, we optimize a VSR for a given task by optimizing the parameters $\boldsymbol{\theta}$. We remark however that $\boldsymbol{\theta}$ may not necessarily be the weights and biases of the ANN, as shall be discussed later. p depends on the ANN *topology*, i.e., the number of layers and their number of neurons. The number of neurons of the input and output layers are equal to the overall number of sensor readings and number of voxels, respectively. The number and size of inner layers can be decided by the designer: after preliminary experiments, we chose to use one intermediate layer with the same number of neurons as the input layer and $\tanh$ as activation function for all the neurons.

In this work, we consider two variants of ANNs: with plasticity and without (static). In the former, the weights of the ANN change over the course of life of the robot (a simulation) following a set of well-defined rules. In this case, $\boldsymbol{\theta}$ consists in the parameters governing such rules; as a result, genomes evolve to synthesize recipes to learn during lifetime. By adding a learning process on top of the existing evolutionary one, plasticity holds the potential to make individuals more adaptable. We also use, as a baseline, a static variant, where weights of the ANN do not change during life, and we call it Multi Layer Perceptron (MLP). In this case, adaptation takes place only at the evolutionary level and the controller parameters $\boldsymbol{\theta}$ consist in the weights associated with the synapses.

In this work, we adopt the Hebbian model for the plastic variant. We introduce it thoroughly in the next subsection.

B. Hebbian learning

As discussed above, Hebbian learning provides a way to optimize an ANN while performing a given task. In an ANN, the weights play the role of the synapses, as each of them modulates the connection between any pair of pre-synaptic the post-synaptic neurons. From a computational point of view [56], the general formulation of Hebbian learning updates the weights according to:

$$w_{ij}^{(k+1)} = w_{ij}^{(k)} + \Delta w_{ij}^{(k)} \tag{1}$$

$$\Delta w_{ij}^{(k)} = \eta x_i^{(k)} y_j^{(k)} \tag{2}$$

where $x_i^{(k)}$ and $y_j^{(k)}$ are the activation of the pre-synaptic and post-synaptic neurons, respectively, at time step k, and η is the learning rate. In synthesis, Equation (2) dictates to strengthen the synapse value if $x_i^{(k)}$ and $y_j^{(k)}$ are positively correlated, weaken it if they are negatively correlated, and keep it constant if at least one of the two is zero. This model takes inspiration from biological systems, in which there is evidence that plasticity in the frontal-striatal synapses arises from changes in the concentration of dopamine, which in turn is a function of the difference between observed and expected outcomes [57, 58].

While several works [59, 60, 48] have successfully employed this "generalized" Hebbian learning to train ANNs, many variations of it do exist [16]. In this work, we use the so-called Hebbian $ABCD$ model [61, 62, 63, 47], which updates the weights according to:

$$\Delta w_{ij}^{(k)} = \eta \left(A x_i^{(k)} y_j^{(k)} + B x_i^{(k)} + C y_j^{(k)} + D \right) \tag{3}$$

where $A, B, C, D \in \mathbb{R}$ are the eponymous coefficients and $\eta \in \mathbb{R}$ is the learning rate. The four coefficients determine the local weight dynamics, with A modulating the relation between the two signals, B and C modulating the pre-synaptic

and post-synaptic values, respectively, and D acting as a bias specific to the synapse.

We call the four $ABCD$ coefficients together a *rule*, and, in our model, there exists one separate rule per synapse. In line with uniform plasticity [64], all the rules share the same learning rate η (we digress on the non-uniform case in Section IV-D); after preliminary experiments, we set $\eta = 0.01$. Moreover, we initialize $w_{ij}^{(0)} = 0$ for every i, j, i.e., at the beginning of the life of the VSR, every weight is set to 0. The parameters $\boldsymbol{\theta}$ that we optimize consist then in the concatenation of the $ABCD$ coefficients for all the rules. The Hebbian controller has thus four times the number of free parameters of an MLP with the same architecture.

C. Evolutionary algorithm

We resort to EC for optimization, and, in particular, adopt Evolution Strategies (ES) [65] as our EA, described in Algorithm 1. Indeed, EAs have proven competitive for reinforcement learning problems [66]; in particular, ES have achieved state-of-the-art results for continuous control tasks and game-playing [67].

function `evolve`():
  $P \leftarrow$ `initialize`(n_{pop})
  foreach $i \in \{1, \dots, n_{\text{gen}}\}$ **do**
    $P_{\text{parents}} \leftarrow$ `bestIndividuals` $\left(P, \left\lfloor \frac{|P|}{4} \right\rfloor\right)$
    $\boldsymbol{\mu} \leftarrow$ `mean`(P_{parents})
    $P' \leftarrow \{$`bestIndividuals`$(P, 1)\}$
    while $|P'| < n_{pop}$ **do**
      $P' \leftarrow P' \cup \{\boldsymbol{\mu} + \mathcal{N}(0, \sigma^2)^p\}$
    end
    $P \leftarrow P'$
  end
  return `bestIndividuals`$(P, 1)$
end

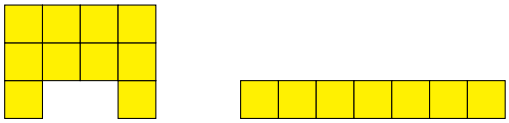
Algorithm 1: The EA used in our experiments.

The EA evolves a fixed-size population of n_{pop} individuals, i.e., numerical vectors $\boldsymbol{\theta}$ of dimension p. At first, n_{pop} individuals (i.e., the concatenation of $ABCD$ coefficients for the Hebbian controller or the weights for the MLP controller) are initialized randomly by sampling $\mathcal{U}(-1, +1)$. Then, for every generation, the fittest quarter of the individuals is chosen as parents. $n_{\text{pop}} - 1$ children are born from the parents, each one obtained by adding a Gaussian noise $\sim \mathcal{N}(0, \sigma^2)$ to each of the p components of the element-wise mean $\boldsymbol{\mu}$ of all parents. Finally, the generated offspring merges with the fittest parent to form the population for the next generation. This process is continued for n_{gen} generations, and the fittest individual is returned at the end of the evolutionary run.

After preliminary experiments, we set $n_{\text{pop}} = 40$, $n_{\text{gen}} = 500$ (corresponding to 20 000 fitness evaluations), and $\sigma = 0.35$.

IV. Experimental analysis

We performed several experiments aimed at answering the following research questions:



(a) Biped (b) Worm

Fig. 1: The two shapes used in our experiments.

- RQ1 Is evolution with Hebbian learning effective? That is, are VSRs equipped with the Hebbian controller effective in the task of locomotion? Are they able to generalize to unseen conditions as new environments or malfunctions in the morphology?
- RQ2 Is there any "true" learning?
- RQ3 Why Hebbian learning works?

We anticipate that by "true" learning, here we mean that the agent retains the ability of accomplishing the task even once Hebbian plasticity is disabled, i.e., the weights are frozen (this will become clearer below).

For answering these questions, we experimented with two different VSR shapes, depicted in Figure 1. In particular, we considered a 4×3 (size of the grid enclosing the voxels) rectangle with a 2×1 rectangle of missing voxels at the bottom-center, that we call *biped*, and a 7×1 rectangle, that we call *worm*. For each shape, we considered three sensory apparatuses differing in the number and kind of sensors they are composed of, hence in their complexity. We equipped the *high*-complexity apparatus as follows: area sensors for all the voxels, touch sensors for the voxels in the bottom row of the shape, velocity sensors for the voxels in the top row of the shape, and lidar sensors for the voxels in the rightmost column of the shape. The *medium*-complexity apparatus is equivalent to the *high* apparatus, with the exception of the lidar sensors, that are absent. Finally, we equipped the *low*-complexity apparatus with just the area sensors for the two top rows in the case of the biped shape, and the three central voxels in the case of the worm shape. For the three apparatuses respectively, the input dimension is 48, 36, and 8 for the biped shape, 31, 28, and 3 for the worm shape. We experimented with different shapes and sensory apparatuses to get a sense of the effectiveness of Hebbian learning across a wide array of morphological conditions.

For all the experiments, we considered the task of locomotion. The goal is to travel on a terrain as fast as possible along the positive x direction, in a fixed amount of simulated time t_{final}. The fitness of a VSR is given by its average velocity $\overline{v}_x$, computed considering the x-position of the VSR center of mass at the beginning and at the end of the simulation. We set $t_{\text{final}} = 60\,\text{s}$. Locomotion is a classic task in evolutionary robotics [68] and usually consists in running along a flat surface. Here, we instead use a hilly terrain. It consists in a sequence of bumps, having an average height of 3 m and an average distance of 30 m. Additionally, the random seed for procedurally generating the bumps is different at every fitness evaluation, so as to prevent individuals from "overfitting" to one single terrain profile, making adaptation more challenging.

We implemented the experimental setup in Java, build-

ing on top of two frameworks: JGEA[2] for the evolutionary optimization and 2D-VSR-Sim[3] [54] for the simulation of VSRs. For the simulator, we set the time step to $\Delta t = \frac{1}{60}$ s and the other parameters to the default values (as a result, all the voxels share the same mechanical properties). The code to reproduce the experiments is publicly available at https://github.com/ndr09/VSRevo.

For each experiment, we performed 10 evolutionary runs with different random seeds for the EA. We remark that, for a given VSR and terrain, the simulations are deterministic. After verifying the adequate hypotheses, we carried out statistical tests with the two-sided Mann Whitney U rank test [69] for independent samples using, unless otherwise specified, 0.05 as confidence level.

A. RQ1: *effectiveness of evolution with Hebbian learning*

As starting point, we are interested in investigating the effectiveness of the Hebbian controller. We measure effectiveness in two different cases: in the same conditions as evolution, and on a re-assessment procedure with slightly different conditions. In both cases, the performance index is $\overline{v}_x$, which, in the first case, is the fitness function itself. The aim of the second case is to test the generalization abilities of an evolved individual. To this end, we compute $\overline{v}_x$ across 10 unseen hilly terrains, obtained with 10 different predefined random seeds.

We compare the Hebbian model against a baseline model. In this work, we used as baseline a "vanilla" MLP controller, with the same architecture as the Hebbian controller (see Section III-A2). The baseline controller is different from the Hebbian controller in two ways. First, the weights of the MLP of the former stay the same for the entire simulation, while in the latter they change at every time step according with Equation (3). Second, in the baseline controller it is precisely the weights that we optimize, differently than in the Hebbian controller, where we optimize the $ABCD$ parameters. For the optimization, we use the same EA of Algorithm 1. As for the Hebbian controller, we couple the baseline controller with the two shapes and the three sensory apparatuses previously outlined.

We summarize the results in Figure 2, which shows $\overline{v}_x$ of the best individuals at the end of evolution, and Figure 3, which shows $\overline{v}_x$ of the best individuals on the re-assessment terrains. For the same shape and sensory apparatus, we also report the p-value for the statistical test against the null hypothesis of equality between the medians. Figure 4 plots $\overline{v}_x$ in terms of median $\pm$ standard deviation over the course of evolution.

From the figures, we find that Hebbian learning is never worse that the baseline: in fact, it is either comparable or better. In particular, if we look at the results shown in Figure 2, it turns out that the Hebbian controller consistently outperforms the MLP for the *low* sensory apparatus in both shapes, and the *high* sensory apparatus in the biped shape, which are the configurations whose p-values are significant. Also, the performance gap seems to be particularly abysmal in the *low* sensory apparatus; we speculate the reason to be

[2] https://github.com/ericmedvet/jgea
[3] https://github.com/ericmedvet/2dhmsr

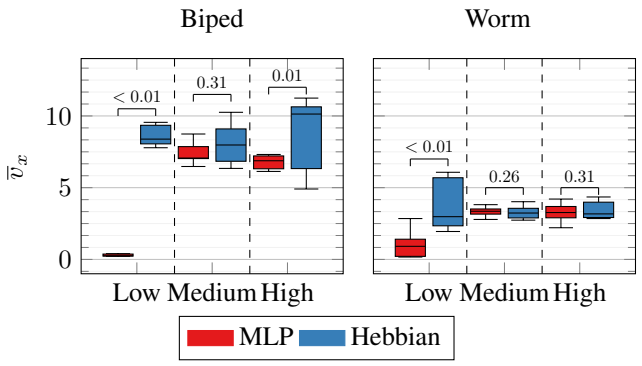


Fig. 2: Distribution of the velocity $\overline{v}_x$ of the best individuals found in each of the 10 evolutionary runs, obtained with the two controller types on each combination of shape and sensory apparatus. Hebbian learning is never worse than the baseline.

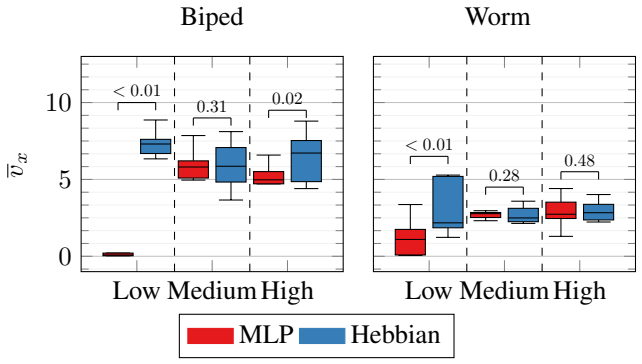


Fig. 3: Distribution of the velocity $\overline{v}_x$ across 10 unseen hilly terrains for the best individuals found in each of the 10 evolutionary runs, obtained with the two controller types on each combination of shape and sensory apparatus. Hebbian learning is never worse than the baseline.

that Hebbian learning is particularly effective with "scarce" perceptual conditions. The results on the re-assessment, shown in Figure 3, corroborate and mirror these effects, indicating that the evolved Hebbian controllers also have generalization abilities. Moreover, Figure 4 confirms that, despite having four times the number of parameters of the MLP, the Hebbian controller succeeds in converging to a plateau in the fitness landscape.

We visually inspected the robot with the evolved Hebbian controllers and found their behaviors to be highly adapted for a locomotion task on uneven terrain; bipeds hop on their legs and worms inch forward as real caterpillars do. Visual inspection is fundamental since behaviors are a strong indicator of a possible reality gap [70, 71]. We made sample videos available at https://youtu.be/bZ2Ek9ohzXI and https://youtu.be/5tCaQTRXRp8.

To further investigate the generalization abilities of the Hebbian controllers, we tested their ability to recover from unforeseen damages affecting the body of the VSR. In particular, we used the following protocol. After half of the simulation (30 s) has elapsed, the VSR experiences a trauma, with every voxel having a 0.5 probability of breaking—thus, every VSR

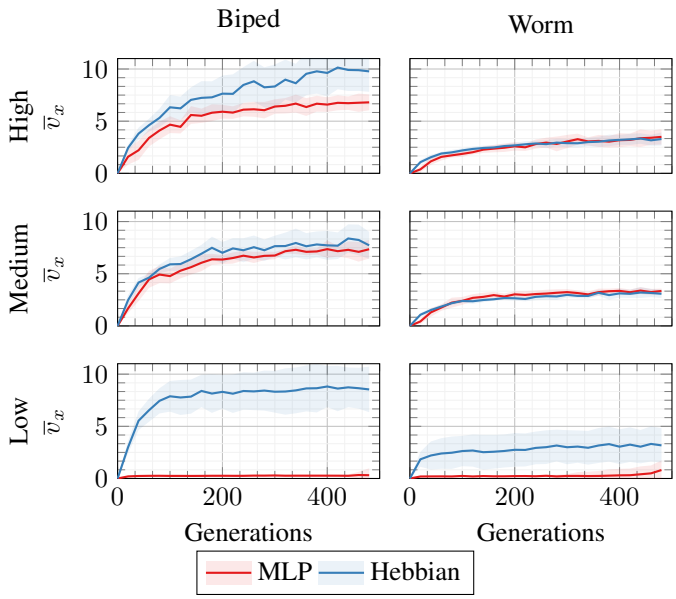


Fig. 4: Median ± standard deviation (solid line and shaded area) of the velocity $\overline{v}_x$ of the best individuals found during each of the 10 evolutionary runs, obtained with the two controller types on each combination of shape and sensory apparatus. Hebbian learning converges as fast as the baseline even if it has four times the number of parameters.

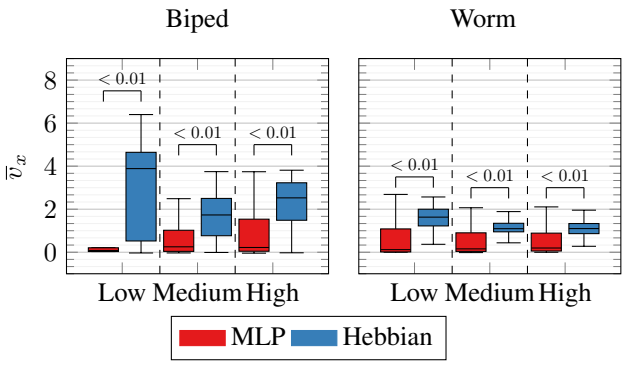


Fig. 5: Distribution of the velocity $\overline{v}_x$ across 10 unseen hilly terrains for the best individuals found in each of the 10 evolutionary runs (with damages), obtained with the two controller types on each combination of shape and sensory apparatus. Hebbian learning is decisively better than the baseline.

has, on average, 50 % of the voxels broken for the second half of the simulation. Upon breakage, a voxel does not apply anymore the actuation signal it receives from the controller, and hence its area is determined only by external forces. We ran an experimental campaign of 10 evolutionary runs for the Hebbian and MLP controller types, and re-assessed the best individuals on 10 unseen hilly terrains, as already explained. Note that, during each simulation (either in evolution or re-assessment), the VSRs experience damages affecting different voxels—this makes the task of evolving a controller harder than without the damages. We found the results in these conditions to be not significantly different from those obtained without damages, and, for the sake of conciseness, we report just the re-assessment results in Figure 5.

From the figure, we conclude that, beyond any doubt, the evolved Hebbian controller is more effective at recovering from unforeseen damages in the body. Taken from this perspective, controllers with Hebbian plasticity would fit neatly into the design of an autonomous robotic ecosystem.

These findings confirm that Hebbian learning is an effective learning paradigm and that, when coupled with evolution, generalization to unseen conditions is where it can truly shine. Two questions then arise: is the Hebbian controller really learning? Furthermore, why does it work, i.e., what is the dynamics of the weights during the robot lifetime? We set out to answer these questions in the following two subsections.

B. RQ2: is there any "true" learning?

We aim at verifying whether, in our experiments, a VSR with Hebbian learning does indeed learn. Intuitively, we say that an individual has "learned" how to perform a task if (a) it has built some internal representation of the experience collected on the task while learning and (b) it is able to re-use this experience. That is, we assume that the outcome of learning is available even after the process of learning has ended. From this assumption it follows that if we stop the learning once enough experience has been collected, the individual should still be able to achieve the task; if, instead, we stop it too early, the individual should not be able to achieve the task. If, at some point of its life, an individual that is learning is already able to achieve the task and we stop the learning, there could be two outcomes: either (a) the individual retains its ability to solve the task or (b) it loses its ability. In the latter case, we might conclude that what was happening inside the individual was not "learning" in the sense we described above, but was instead some form of fast adaptation of the brain that was itself functional to the ability of the individual. Namely, it is so functional that, if you stop it, the individual looses its ability to solve the task. Stopping the learning and looking at what happens can hence be used to verify whether true learning is happening.

Based on these considerations, we performed the following experiment. Given a VSR with an evolved Hebbian controller, we take a snapshot of it, along with its weights, at every second during the simulation. For every such snapshot, we measure its average velocity $\overline{v}_x$ in a new simulation lasting 60 s over an unseen hilly terrain with Hebbian learning turned off (i.e., we fix the weights to the frozen values of the snapshot and do not change them anymore during the simulation). We performed this procedure for the best individual (i.e., set of $ABCD$ parameters) of every evolutionary run. We report the results of such validation in Figure 6, in terms of median ± standard deviation across the 10 runs. On the x-axis, we report the time (in s) at which we took the corresponding snapshot. For the sake of conciseness, we report the results only for the *high* sensory apparatus since it is the configuration that delivered the best results for the Hebbian controller. The other cases are qualitatively similar.

From Figure 6, we observe that for both shapes the median

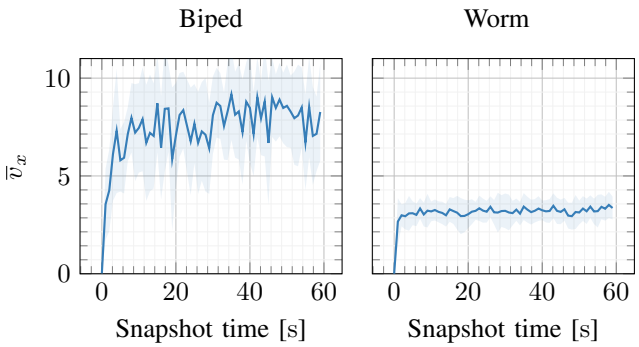


Fig. 6: Median ± standard deviation (solid line and shaded area) of the velocity $\overline{v}_x$ of the best individuals found in each of the 10 evolutionary runs, with weights frozen at different time steps of the simulation (on x-axis, in s), obtained with the Hebbian controller on the two shapes and the *high* sensory apparatus. Hebbian learning requires just $\approx 10\,\mathrm{s}$ for the biped and $\approx 5\,\mathrm{s}$ for the worm to converge to a high-performing weight configuration.

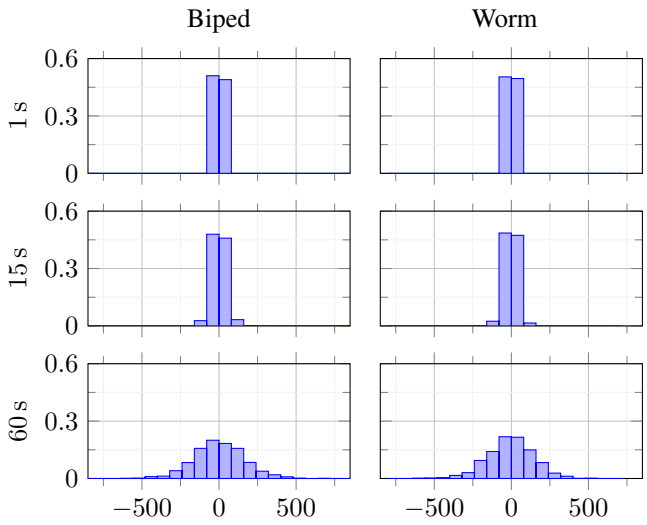


Fig. 7: Weight distributions for two sample best individuals (one per shape, both with *high* sensory apparatus), at three different time steps of the simulation: 1 s, 15 s, and 60 s. In Hebbian learning, weights do diverge.

$\overline{v}_x$ rises very steeply and then settles around a stable point. It takes about 10 s to 20 s before stabilizing for the biped shape, while for the worm it converges more rapidly in the first 5 s. This observation demonstrates that learning does indeed take place: after an initial settling period (which can be seen as the actual "learning" phase), the robots are capable of achieving high $\overline{v}_x$ even with the weights frozen—that is, they have acquired some experience through learning and they are able to exploit it to run effectively even after learning has stopped. We speculate that, by the time $\overline{v}_x$ settles into a stable point, the weights have converged to a high-performing attractor in the weight space. These findings are in line with previous research on Hebbian learning for robotic agents, in particular [47].

C. RQ3: why Hebbian learning works

We want to understand why Hebbian learning works. In order to do so, we investigate the weight dynamics underlying locomotion and see whether they disclose some insights. Also in this case, for the sake of conciseness, we report the results only for the *high* sensory apparatus, since it performed the best in terms of effectiveness (see Section IV-A). The results are qualitatively the same for the other configurations.

Figure 7 plots, separately for two sample best individuals (one per shape), the histograms of the relative frequency of weights at different time steps of the simulation.

From the figure, one finding strikes us. By the end of the simulation, weights diverge to assume a bell-shaped distribution centered on their initial value of 0.0. Moreover, there are small clusters of values that accumulate on the boundaries and assume very large values. If we consider that learning takes about 10 s to happen (see Section IV-B), weights divergence is interesting since it persists over the entire simulation. We also visualized the neuron activations and found the corresponding histograms to be bi-modal, with two peaks at the boundaries (i.e., -1 and $+1$). Considering that we employ $\tanh$ as activation function (whose output domain is indeed $[-1, +1]$), it is clear that activations become saturated. The divergence of the weights might hint that evolution is essentially performing a form of synaptic pruning [42, 41], and implicitly optimizing not really the weights, but rather the topology of the neural networks. Figure 7 thus led us to believe that Hebbian plasticity creates some kind of underlying "Boolean" dynamics in the neural networks. This fact bears similarity to what happens with weight agnostic neural networks [72] and "bang-bang" controllers [73], which are continuous-action controllers that, when optimized, degenerate to binary controllers. Inspecting the weights and activations of the other best individuals confirmed these results. Here, we plotted just two of them for the sake of conciseness.

To better understand the role that weight divergence plays, we introduce, in the form of an ablation study, a *normalized* Hebbian model. In this representation, we normalize weights to $[-1, +1]$ in the following way:

$$w_{ij}'^{(k+1)} = \frac{w_{ij}^{(k+1)}}{\left\| \boldsymbol{w}_i^{(k+1)} \right\|_2} \tag{4}$$

where $\boldsymbol{w}_i^{(k+1)}$ is the vector of pre-synaptic weights for the post-synaptic i-th neuron. Equation (4) tells that, at every time step, for every neuron, we divide its pre-synaptic weights by their Euclidean norm. We remark that, in the previous experiments, weights were unbounded. In doing so, we tap into the body of evidence on local synaptic competition between neurons in vivo [74], according to which adjacent synapses modulate their strength so as to specialize and not be subdued by the others. Moreover, the relative relevance of weights of a given neuron does not change, and so there is no bias introduced in the EA. The rationale behind the normalized Hebbian model is clearly that, since we are preventing the weights from diverging, we want to see whether it unveils

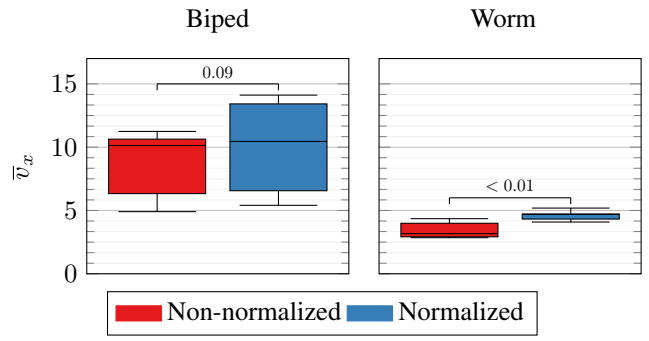


Fig. 8: Distribution of the velocity $\overline{v}_x$ of the best individuals found in each of the 10 evolutionary runs, obtained with the Hebbian controller (with and without weight normalization to $[-1, +1]$) on the two shapes and the *high* sensory apparatus.

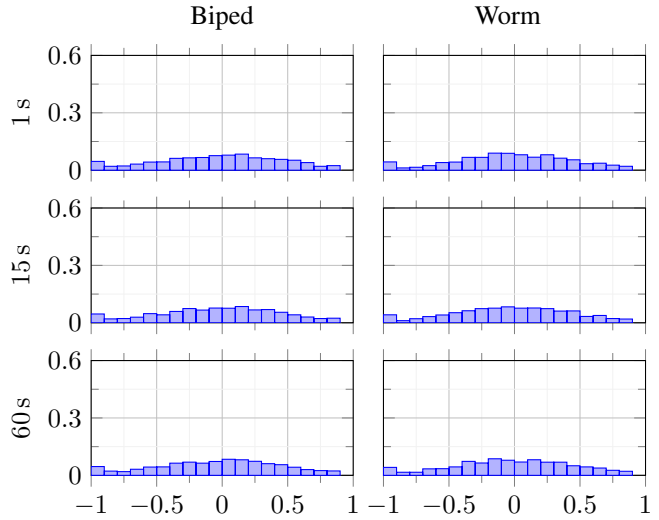


Fig. 9: Weight distributions for two sample best individuals (one per shape, both with *high* sensory apparatus), at three different time steps of the simulation: 1 s, 15 s, and 60 s. In Hebbian learning with normalization, weights do not diverge.

more insights about the original, non-normalized model whose weights diverged.

We ran an experimental campaign of 10 independent runs for the normalized Hebbian model, using the same parameters of the non-normalized Hebbian model. We report the results in Figure 8 in terms of the fitness $\overline{v}_x$ of the best individuals at the end of evolution, and compare it with the non-normalized Hebbian model results from Section IV-A. Figure 9 plots the histograms of the relative frequency of weights at different time steps of the simulation, for two sample best individuals (one biped and one worm).

From Figure 8, we first notice that normalizing the weights does not impact performance and that the normalized Hebbian model reaches fitness $\overline{v}_x$ comparable (or even better) to the non-normalized Hebbian model. Also, Figure 9 proves very insightful. As expected, weights do not diverge outside of the $[-1, +1]$ range and distribute more uniformly in that range. We also visualized the neuron activations over time and, in contrast with the non-normalized Hebbian model, they

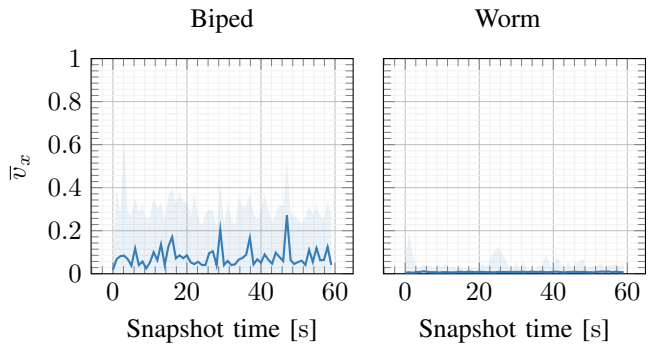


Fig. 10: Median ± standard deviation (solid line and shaded area) of the velocity $\overline{v}_x$ of the best individuals found in each of the 10 evolutionary runs, with weights frozen at different time steps of the simulation (on x-axis, in s), obtained with the Hebbian controller on the two shapes and the *high* sensory apparatus, with weight normalization to $[-1, +1]$. Hebbian learning with weight normalization does not stabilize around an attractor.

display a clear recurrent cyclical pattern (see Figures 1 and 2 shown in the Supplementary Material). We conjectured the reason for this to be that the normalized Hebbian model evolves to exploit the dynamics of the underlying soft body; intuitively, soft materials are so powerful that, in order to produce a gait, everything the controller is left to do is to instill a cyclical recurring pattern. This hypothesis appears even more grounded if we consider that the frequency of activation "cycles" corresponds to the frequency of the real gait for the two shapes, being higher and more regular for the biped (bipeds gait has a period of $\approx 1.5\,\mathrm{s}$), whereas lower and less regular for the worm (worms gait has a period of $\approx 2\,\mathrm{s}$). Under this light, the dynamical system of the soft body coupled with the dynamical system of the normalized Hebbian controller can be seen as an instance of morphological computation [75, 76], according to which the brain offloads part of the computation to the body.

To test this hypothesis, we repeated the same validation used in Section IV-B, and report the results in Figure 10 using the same semantics of Figure 6. We remark that, during the simulation of a snapshot, we freeze the corresponding weights.

Surprisingly, the results are quite different from what observed for the non-normalized Hebbian model in Section IV-B. In fact, the median average velocity $\overline{v}_x$ remains stuck at around $0\,\mathrm{m/s}$, meaning that individuals are not producing any useful locomotion at all. In the non-normalized case, see Figure 6, $\overline{v}_x$ climbed up very quickly. This analysis holds for both shapes.

Our interpretation is that, while it is possible to freeze Hebbian learning (after a settling period) in the non-normalized case, the same is not true for the normalized case. It is likely that the dynamical system of the Hebbian controller and the dynamical system of the soft body act in unison as a single dynamical system. Hebbian plasticity—in the normalized case—concurs to instill the correct gait dynamics for locomotion, but this is not "true" learning (in the sense specified in Section IV-B), since freezing Hebbian learning

results in extinguishing an entangled portion of the dynamical system.

We conclude that unbounded weights are a necessary component in the original formulation. We speculate the reason why weights divergence does not affect performance may be explained in two different ways. As mentioned earlier, one hypothesis is that a sort of "Boolean" dynamics emerges in the Hebbian model. This hypothesis sounds alluring if we consider that, in theory, the action space for each voxel (which is continuous) might be shrunk to a binary space with just contraction or expansion. To test this hypothesis it would be necessary, as a matter of example, to switch from the foundational task of locomotion to more complex ones that cannot be solved by a mere recurring pattern. We leave this investigation for future work. Another hypothesis is that those synapses that diverge are in fact synapses that do not contribute much to the output and we could, in theory, prune them. This hypothesis seems even more intriguing if we parallel it with the recently introduced "lottery ticket hypothesis" [77], according to which optimizing a dense neural network boils down to optimizing the most effectively-initialized sub-networks. However, our setting is slightly different, as weights are initialized at the same value and develop over time through Hebbian plasticity. Future work will consider how to extend the lottery ticket hypothesis to our setting. Finally, for the sake of this study, we were not concerned with other physiological processes other than Hebbian plasticity. There is evidence that other processes, e.g., homeostatic plasticity [78], complement it to maintain the overall activity of a neuron within the network, and might balance unconstrained synaptic growth. We leave this investigation as future work.

D. Additional experiments

In the following, we list experiments that we carried out, but delivered less interesting results.

First, we tried to evolve η alongside the $ABCD$ coefficients. The motivation for this is that the joint space of η and the $ABCD$ coefficients is, potentially, a more expressive representation. In particular, it might allow evolution to find learning rates that are tailored to specific Hebbian rules, in parallel with what happens in the human brain, where plasticity is known to operate on a different timescale for the striatum and the amygdala [79]. The results turned out to be not significantly different than searching in the sole $ABCD$ space. So, for the sake of this paper, we resorted to the more compact representation between the two, i.e., the one evolving only the $ABCD$ coefficients. The reason for this might be that the EA searches more effectively in the more compact $ABCD$ space, or that differing η values do not benefit the Hebbian learning model.

Second, we experimented with different Hebbian models than the one presented in Section III having a different Hebbian rule for every synapse, that we will hereon label *full*. In particular, we tested:

(a) a *single* model, where all the synapses share a single Hebbian rule;
(b) a *sensors* model, where pre-synaptic connections of input neurons corresponding to sensors of the same type share the same Hebbian rule and there is an unique rule for each one of the other two layers;
(c) a *post-synaptic* model, where the post-synaptic connections of a given neuron share the same Hebbian rule;
(d) a *pre-synaptic* model, where the pre-synaptic connections of a given neuron share the same Hebbian rule.

We ran an experimental campaign for each of the models above, using the same setup of the *full* model. The *single*, *sensors*, and *input* models turned out to be ineffective for all shapes and sensory apparatuses. The *pre-synaptic* model, on the other side, managed to evolve and settle on a plateau. Nevertheless, it still under-performed the *full* model. Since what distinguishes these models is the number of parameters to optimize, they present different compactness-expressivity trade-offs. We then believe these results are a consequence of expressivity, being the latter, for this setting, beneficial.

Finally, we experimented with the initialization of the weights at the beginning of the life of the robot. In addition to the zero initialization adopted in this paper, we considered a case with initialization of the weights from $\mathcal{U}(-1, +1)$, but we did not find any statistically significant difference. We thus resorted to zero initialization since starting from an idle posture is more sensible for a locomotion task and there is one less causal factor.

V. Concluding Remarks

Our experiments revealed a number of noteworthy findings. First, we observed that the evolved Hebbian controllers are never worse, and often better, than their counterparts based on MLP with evolved weights. This is true for all the tested combinations of two VSR shapes and three sensory apparatuses. Second, we found that the adopted Hebbian learning model does indeed "learn", i.e., the robots perform well even when the weight update is disabled, provided that a sufficient amount of time was previously allotted to the learning process. Third, we found that unbounded weights are a necessary element of learning in our model, in that they lead to an implicit form of pruning. On the contrary, when we normalize weights, robot performance with learning is still comparable; however, if we disable the weight update, performance drops dramatically.

Acknowledgments

FP was partially supported by a Google Faculty Research Award granted to EM.

References

[1] M. Hale, E. Buchanan Berumen, A. Winfield, J. Timmis, E. Hart, G. Eiben, W. Li, and A. Tyrrell, "The are robot fabricator: How to (re) produce robots that can evolve in the real world," in *International Society for Artificial Life: ALIFE2019*. York, 2019, pp. 95–102.
[2] G. Nitschke and D. Howard, "AutoFac: The Perpetual Robot Machine," *IEEE Transactions on Artificial Intelligence*, 2021.
[3] A. E. Eiben, N. Bredeche, M. Hoogendoorn, J. Stradner, J. Timmis, A. Tyrrell, and A. Winfield, "The triangle of life: Evolving robots in real-time and real-space," in *Proceedings of the European Conference on Artificial Life (ECAL-2013)*, 2013, pp. 1–8.
[4] A. E. Eiben and E. Hart, "If It Evolves It Needs to Learn," in *Proceedings of the 2020 Genetic and Evolutionary Computation Conference Companion*, ser. GECCO '20. New York, NY, USA: Association for Computing Machinery, 2020, p. 1383–1384. [Online]. Available: https://doi.org/10.1145/3377929.3398151

[5] A. Eiben, "Real-world robot evolution: why would it (not) work?" *Frontiers in Robotics and AI*, p. 243, 2021.

[6] M. Sipper, E. Sanchez, D. Mange, M. Tomassini, A. Pérez-Uribe, and A. Stauffer, "A phylogenetic, ontogenetic, and epigenetic view of bio-inspired hardware systems," *IEEE Transactions on Evolutionary Computation*, vol. 1, no. 1, pp. 83–97, 1997.

[7] T. F. Nygaard, C. P. Martin, E. Samuelsen, J. Torresen, and K. Glette, "Real-world evolution adapts robot morphology and control to hardware limitations," in *Proceedings of the Genetic and Evolutionary Computation Conference*, 2018, pp. 125–132.

[8] S. Ha, P. Xu, Z. Tan, S. Levine, and J. Tan, "Learning to walk in the real world with minimal human effort," *arXiv preprint arXiv:2002.08550*, 2020.

[9] A. Hallawa, T. Born, A. Schmeink, G. Dartmann, A. Peine, L. Martin, G. Iacca, A. Eiben, and G. Ascheid, "Evo-RL: evolutionary-driven reinforcement learning," in *Proceedings of the Genetic and Evolutionary Computation Conference Companion*, 2021, pp. 153–154.

[10] P. Pagliuca and S. Nolfi, "The dynamic of body and brain co-evolution," *Adaptive Behavior*, p. 1059712321994685, 2020.

[11] D. Grbic and S. Risi, "Safer Reinforcement Learning through Transferable Instinct Networks," *arXiv preprint arXiv:2107.06686*, 2021.

[12] D. S. Moore, *The Dependent Gene: The Fallacy of "Nature Vs. Nurture"*. Macmillan, 2003.

[13] K. Zilles, "Neuronal plasticity as an adaptive property of the central nervous system," *Annals of Anatomy-Anatomischer Anzeiger*, vol. 174, no. 5, pp. 383–391, 1992.

[14] A. R. Patten, S. Y. Yau, C. J. Fontaine, A. Meconi, R. C. Wortman, and B. R. Christie, "The benefits of exercise on structural and functional plasticity in the rodent hippocampus of different disease models," *Brain Plasticity*, vol. 1, no. 1, pp. 97–127, 2015.

[15] D. O. Hebb, *The organization of behavior: A neuropsychological theory*. Psychology Press, 2005.

[16] A. Soltoggio, K. O. Stanley, and S. Risi, "Born to learn: the inspiration, progress, and future of evolved plastic artificial neural networks," *Neural Networks*, vol. 108, pp. 48–67, 2018.

[17] J. Hiller and H. Lipson, "Automatic design and manufacture of soft robots," *IEEE Transactions on Robotics*, vol. 28, no. 2, pp. 457–466, 2011.

[18] S. Kriegman, D. Blackiston, M. Levin, and J. Bongard, "A scalable pipeline for designing reconfigurable organisms," *Proceedings of the National Academy of Sciences*, vol. 117, no. 4, pp. 1853–1859, 2020.

[19] N. Cheney, J. Clune, and H. Lipson, "Evolved electrophysiological soft robots," in *ALIFE 14: The Fourteenth International Conference on the Synthesis and Simulation of Living Systems*. MIT Press, 2014, pp. 222–229.

[20] N. Cheney, J. Bongard, and H. Lipson, "Evolving soft robots in tight spaces," in *Proceedings of the 2015 annual conference on Genetic and Evolutionary Computation*, 2015, pp. 935–942.

[21] F. Corucci, N. Cheney, F. Giorgio-Serchi, J. Bongard, and C. Laschi, "Evolving soft locomotion in aquatic and terrestrial environments: effects of material properties and environmental transitions," *Soft robotics*, vol. 5, no. 4, pp. 475–495, 2018.

[22] J. Bhatia, H. Jackson, Y. Tian, J. Xu, and W. Matusik, "Evolution gym: A large-scale benchmark for evolving soft robots," *Advances in Neural Information Processing Systems*, vol. 34, 2021.

[23] A. Faiña, "Evolving Modular Robots: Challenges and Opportunities," in *ALIFE 2021: The 2021 Conference on Artificial Life*. MIT Press, 2021.

[24] M. Naya-Varela, A. Faiña, and R. Duro, "Morphological Development in robotic learning: A survey," *IEEE Transactions on Cognitive and Developmental Systems*, 2021.

[25] K. De Jong, "Evolutionary computation: a unified approach," in *Proceedings of the 2016 on Genetic and Evolutionary Computation Conference Companion*, 2016, pp. 185–199.

[26] K. Sims, "Evolving virtual creatures," in *Proceedings of the 21st annual conference on Computer graphics and interactive techniques*, 1994, pp. 15–22.

[27] J. C. Bongard, "Evolutionary robotics," *Communications of the ACM*, vol. 56, no. 8, pp. 74–83, 2013.

[28] S. Pontes-Filho and S. Nichele, "Towards a framework for the evolution of artificial general intelligence," *arXiv preprint arXiv:1903.10410*, 2019.

[29] J. M. Baldwin, "A new factor in evolution," *American naturalist*, pp. 536–553, 2018.

[30] G. E. Hinton, S. J. Nowlan *et al.*, "How learning can guide evolution," *Adaptive individuals in evolving populations: models and algorithms*, vol. 26, pp. 447–454, 1996.

[31] C. A. Shaw, J. C. McEachern, and J. McEachern, *Toward a theory of neuroplasticity*. Psychology Press, 2001.

[32] K. Ganguly and M.-m. Poo, "Activity-dependent neural plasticity from bench to bedside," *Neuron*, vol. 80, no. 3, pp. 729–741, 2013.

[33] S. F. Cooke and T. V. Bliss, "Plasticity in the human central nervous system," *Brain*, vol. 129, no. 7, pp. 1659–1673, 2006.

[34] P. V. Massey and Z. I. Bashir, "Long-term depression: multiple forms and implications for brain function," *Trends in neurosciences*, vol. 30, no. 4, pp. 176–184, 2007.

[35] J. Schmidhuber, "A 'self-referential'weight matrix," in *International Conference on Artificial Neural Networks*. Springer, 1993, pp. 446–450.

[36] S. Risi and K. O. Stanley, "Indirectly encoding neural plasticity as a pattern of local rules," in *International Conference on Simulation of Adaptive Behavior*. Springer, 2010, pp. 533–543.

[37] J. Ba, G. E. Hinton, V. Mnih, J. Z. Leibo, and C. Ionescu, "Using fast weights to attend to the recent past," *Advances in Neural Information Processing Systems*, vol. 29, pp. 4331–4339, 2016.

[38] D. Ha, A. Dai, and Q. V. Le, "Hypernetworks," *arXiv preprint arXiv:1609.09106*, 2016.

[39] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," *arXiv preprint arXiv:1409.0473*, 2014.

[40] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in neural information processing systems*, 2017, pp. 5998–6008.

[41] T. Hoefler, D. Alistarh, T. Ben-Nun, N. Dryden, and A. Peste, "Sparsity in Deep Learning: Pruning and growth for efficient inference and training in neural networks," *arXiv preprint arXiv:2102.00554*, 2021.

[42] G. Nadizar, E. Medvet, F. A. Pellegrino, M. Zullich, and S. Nichele, "On the effects of pruning on evolved neural controllers for soft robots," in *Proceedings of the Genetic and Evolutionary Computation Conference Companion*, 2021, pp. 1744–1752.

[43] N. Caporale and Y. Dan, "Spike timing–dependent plasticity: a Hebbian learning rule," *Annu. Rev. Neurosci.*, vol. 31, pp. 25–46, 2008.

[44] S. Nolfi and D. Parisi, "Learning to adapt to changing environments in evolving neural networks," *Adaptive behavior*, vol. 5, no. 1, pp. 75–98, 1996.

[45] D. Floreano and J. Urzelai, "Evolution of plastic control networks," *Autonomous robots*, vol. 11, no. 3, pp. 311–317, 2001.

[46] T. Miconi, K. Stanley, and J. Clune, "Differentiable plasticity: training plastic neural networks with backpropagation," in *International Conference on Machine Learning*. PMLR, 2018, pp. 3559–3568.

[47] E. Najarro and S. Risi, "Meta-learning through hebbian plasticity in random networks," *arXiv preprint arXiv:2007.02686*, 2020.

[48] A. Yaman, G. Iacca, D. C. Mocanu, M. Coler, G. Fletcher, and M. Pechenizkiy, "Evolving plasticity for autonomous learning under changing environmental conditions," *Evolutionary computation*, vol. 29, no. 3, pp. 391–414, 2021.

[49] J. Jordan, M. Schmidt, W. Senn, and M. A. Petrovici, "Evolving interpretable plasticity for spiking networks," *Elife*, vol. 10, 2021.

[50] J. Talamini, E. Medvet, A. Bartoli, and A. De Lorenzo, "Evolutionary synthesis of sensing controllers for voxel-based soft robots," in *Artificial Life Conference Proceedings*. MIT Press, 2019, pp. 574–581.

[51] N. Cheney, R. MacCurdy, J. Clune, and H. Lipson, "Unshackling evolution: evolving soft robots with multiple materials and a powerful generative encoding," *ACM SIGEVOlution*, vol. 7, no. 1, pp. 11–23, 2014.

[52] E. Medvet, A. Bartoli, F. Pigozzi, and M. Rochelli, "Biodiversity in evolved voxel-based soft robots," in *Proceedings of the Genetic and Evolutionary Computation Conference*, 2021, pp. 129–137.

[53] A. Ferigo, G. Iacca, and E. Medvet, "Beyond Body Shape and Brain: Evolving the Sensory Apparatus of Voxel-Based Soft Robots," in *International Conference on the Applications of Evolutionary Computation (Part of EvoStar)*. Springer, 2021, pp. 210–226.

[54] E. Medvet, A. Bartoli, A. De Lorenzo, and S. Seriani, "2D-VSR-Sim: A simulation tool for the optimization of 2-D voxel-based soft robots," *SoftwareX*, vol. 12, p. 100573, 2020.

[55] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning*. MIT press, 2016.

[56] T. H. Brown, E. W. Kairiss, and C. L. Keenan, "Hebbian synapses: biophysical mechanisms and algorithms," *Annual review of neuroscience*, vol. 13, no. 1, pp. 475–511, 1990.

[57] B. B. Averbeck and V. D. Costa, "Motivational neural circuits underlying reinforcement learning," *Nature Neuroscience*, vol. 20, no. 4, pp. 505–512, 2017.

[58] E. O. Neftci and B. B. Averbeck, "Reinforcement learning in artificial and biological systems," *Nature Machine Intelligence*, vol. 1, no. 3, pp. 133–143, 2019.

[59] O. J. Coleman and A. D. Blair, "Evolving plastic neural networks for online learning: review and future directions," in *Australasian Joint Conference on Artificial Intelligence*. Springer, 2012, pp. 326–337.
[60] D. Floreano and J. Urzelai, "Evolutionary robots with on-line self-organization and behavioral fitness," *Neural Networks*, vol. 13, no. 4-5, pp. 431–443, 2000.
[61] C. Mattiussi and D. Floreano, "Analog genetic encoding for the evolution of circuits and networks," *IEEE Transactions on evolutionary computation*, vol. 11, no. 5, pp. 596–607, 2007.
[62] Y. Niv, D. Joel, I. Meilijson, and E. Ruppin, "Evolution of reinforcement learning in uncertain environments: Emergence of risk-aversion and matching," in *European Conference on Artificial Life*. Springer, 2001, pp. 252–261.
[63] A. Soltoggio, P. Durr, C. Mattiussi, and D. Floreano, "Evolving neuromodulatory topologies for reinforcement learning-like problems," in *2007 IEEE Congress on Evolutionary Computation*. IEEE, 2007, pp. 2471–2478.
[64] J. Schmidhuber, "Reducing the ratio between learning complexity and number of time varying variables in fully recurrent nets," in *International Conference on Artificial Neural Networks*. Springer, 1993, pp. 460–463.
[65] H.-G. Beyer and H.-P. Schwefel, "Evolution strategies–a comprehensive introduction," *Natural computing*, vol. 1, no. 1, pp. 3–52, 2002.
[66] F. P. Such, V. Madhavan, E. Conti, J. Lehman, K. O. Stanley, and J. Clune, "Deep neuroevolution: Genetic algorithms are a competitive alternative for training deep neural networks for reinforcement learning," *arXiv preprint arXiv:1712.06567*, 2017.
[67] T. Salimans, J. Ho, X. Chen, S. Sidor, and I. Sutskever, "Evolution strategies as a scalable alternative to reinforcement learning," *arXiv preprint arXiv:1703.03864*, 2017.
[68] S. Nolfi and D. Floreano, *Evolutionary robotics: The biology, intelligence, and technology of self-organizing machines*. MIT press, 2000.
[69] H. B. Mann and D. R. Whitney, "On a test of whether one of two random variables is stochastically larger than the other," *The annals of mathematical statistics*, pp. 50–60, 1947.
[70] F. van Diggelen, E. Ferrante, N. Harrak, J. Luo, D. Zeeuwe, and A. Eiben, "The Influence of Robot Traits and Evolutionary Dynamics on the Reality Gap," *IEEE Transactions on Cognitive and Developmental Systems*, 2021.
[71] E. Salvato, G. Fenu, E. Medvet, and F. A. Pellegrino, "Crossing the Reality Gap: a Survey on Sim-to-Real Transferability of Robot Controllers in Reinforcement Learning," *IEEE Access*, pp. 1–19, 2021.
[72] A. Gaier and D. Ha, "Weight agnostic neural networks," *arXiv preprint arXiv:1906.04358*, 2019.
[73] R. Bellman, I. Glicksberg, and O. Gross, "On the "bang-bang" control problem," *Quarterly of Applied Mathematics*, vol. 14, no. 1, pp. 11–18, 1956.
[74] S. El-Boustani, J. P. Ip, V. Breton-Provencher, G. W. Knott, H. Okuno, H. Bito, and M. Sur, "Locally coordinated synaptic plasticity of visual cortex neurons in vivo," *Science*, vol. 360, no. 6395, pp. 1349–1354, 2018.
[75] C. Paul, "Morphological computation: A basis for the analysis of morphology and control requirements," *Robotics and Autonomous Systems*, vol. 54, no. 8, pp. 619–630, 2006.
[76] V. C. Müller and M. Hoffmann, "What is morphological computation? On how the body contributes to cognition and control," *Artificial life*, vol. 23, no. 1, pp. 1–24, 2017.
[77] J. Frankle and M. Carbin, "The lottery ticket hypothesis: Finding sparse, trainable neural networks," *arXiv preprint arXiv:1803.03635*, 2018.
[78] G. G. Turrigiano and S. B. Nelson, "Homeostatic plasticity in the developing nervous system," *Nature reviews neuroscience*, vol. 5, no. 2, pp. 97–107, 2004.
[79] V. D. Costa, O. Dal Monte, D. R. Lucas, E. A. Murray, and B. B. Averbeck, "Amygdala and ventral striatum make distinct contributions to reinforcement learning," *Neuron*, vol. 92, no. 2, pp. 505–517, 2016.

Andrea Ferigo is a PhD student at the Department of Information Engineering and Computer Science of University of Trento, Italy, where he also completed his Bachelors and Masters degree in Computer Science in October 2018 and March 2021, respectively.

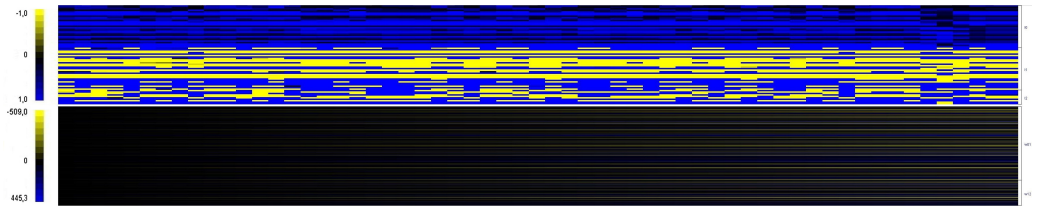
Giovanni Iacca is Associate Professor at the Department of Information Engineering and Computer Science of the University of Trento, Italy, where he founded the Distributed Intelligence and Optimization Lab. Previously, he worked as researcher in Germany (RWTH Aachen, 2017-2018), Switzerland (EPFL, 2013–2016) and The Netherlands (INCAS3, 2012–2016), as well as in industry in the areas of software engineering and industrial automation. His research focuses on computational intelligence, stochastic optimization, and distributed systems.

Eric Medvet received the degree in Electronic Engineering cum laude in 2004 and the PhD degree in Computer Engineering in 2008, both from the University of Trieste, Italy, where he is currently an associate professor in Computer Engineering and the director of the Evolutionary Robotics and Artificial Life Lab. His research interests include evolutionary robotics and artificial life, evolutionary computation, and applications of machine learning.

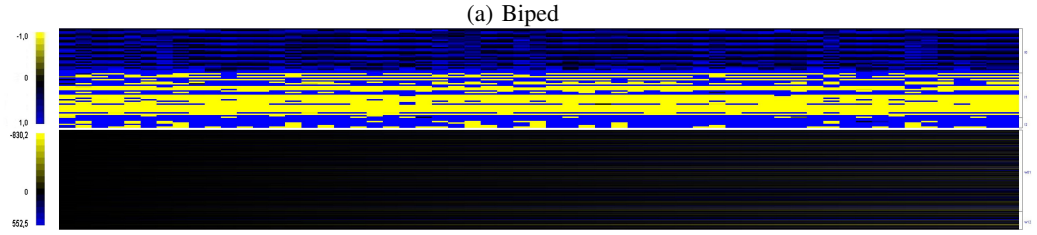
Federico Pigozzi received a Bachelors degree in Economics cum laude from the University of Trieste, Italy, in 2017. He received a Masters degree in Data Science and Scientific Computing cum laude from the University of Trieste, Italy, in 2020. He is currently working towards the PhD degree in Computer Engineering at the University of Trieste, Italy. His research interests include artificial life (especially evolutionary robotics), evolutionary computation, and artificial intelligence.

Evolving Hebbian Learning Rules in Voxel-based Soft Robots (Supplementary Material)

Andrea Ferigo, Giovanni Iacca, Eric Medvet, Federico Pigozzi

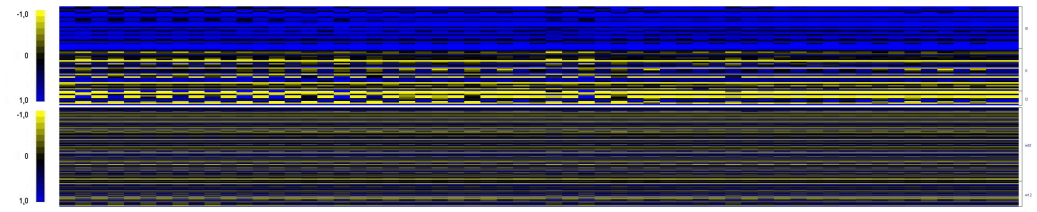


(a) Biped

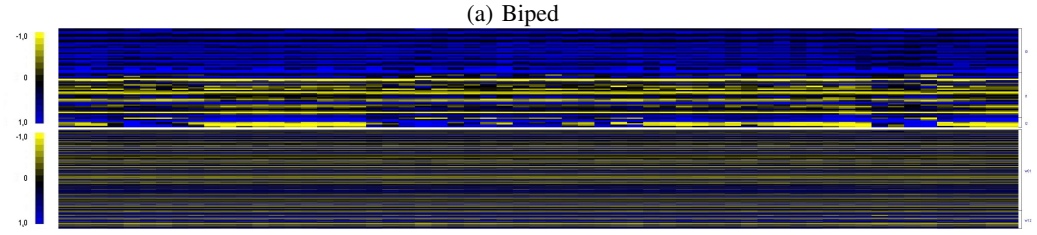


(b) Worm

Fig. 1: Time series for the weights (top row) and variance of the output signal of neurons (bottom row) for a biped and a worm of the best individuals. Time steps is on the x axis, value is on the y axis. Weights and variances range from yellow (large positive number) to blue (large negative number). Weights saturate to be very large in absolute value, either positive or negative, while the majority of neurons output a signal that is close to 0.



(a) Biped



(b) Worm

Fig. 2: Time series for the weights (top row) and variance of the output signal of neurons (bottom row) for a biped and a worm of the normalized best individuals. Time steps is on the x axis, value is on the y axis. Weights and variances range from yellow (large positive number) to blue (large negative number). Weights do not saturate to be very large in absolute value, and show a clear recurrent pattern.

Andrea Ferigo and Giovanni Iacca are with the Department of Information Engineering and Computer Science, University of Trento, Trento, Italy. (email: andrea.ferigo@unitn.it and giovanni.iacca@unitn.it) Eric Medvet and Federico Pigozzi are with the Department of Engineering and Architecture, University of Trieste, Trieste, Italy. (email: emedvet@units.it and federico.pigozzi@phd.units.it) All authors contributed equally and are listed in alphabetical order.