frontiers | Frontiers in Neurorobotics

ORIGINAL RESEARCH
published: 01 June 2022
doi: 10.3389/fnbot.2022.848084

Reach Space Analysis of Baseline Differential Extrinsic Plasticity Control

Simon Birrell, Arsen Abdulali and Fumiya Iida*

Bio-Inspired Robotics Laboratory, Department of Engineering, University of Cambridge, Cambridge, United Kingdom

The neuroplasticity rule Differential Extrinsic Plasticity (DEP) has been studied in the context of goal-free simulated agents, producing realistic-looking, environmentally-aware behaviors, but no successful control mechanism has yet been implemented for intentional behavior. The goal of this paper is to determine if "short-circuited DEP," a simpler, open-loop variant can generate desired trajectories in a robot arm. DEP dynamics, both transient and limit cycles are poorly understood. Experiments were performed to elucidate these dynamics and test the ability of a robot to leverage these dynamics for target reaching and circular motions.

Keywords: Differential Extrinsic Plasticity, self-organization, robotic control, play, intrinsic motivation, neuroplasticity, reinforcement learning, complexity

1. INTRODUCTION

Robot control is still very much a work in progress. While much has been learned of how humans and animals control their bodies (Winter, 2009), either outright or after a learning process, we still do not know enough to be able to design a robot that even approaches human dexterity. Classical control theory and more recently Reinforcement Learning (RL) have been extensively studied but are still subject to lack of robustness, the curse of dimensionality and unreasonably high learning times (Sutton and Barto, 2018).

One issue with these frameworks is the assumption that the brain directly controls the output of each available degree of freedom; typically a learning agent will adjust its body's motor torques at each time step to produce a desired result in a rigid body system within a given environment (see for example OpenAI Gym; Brockman et al., 2016). This is clearly not how biology tackles the problem. In a human, descending signals from the cortex pass through and are modified by interneurons with their own neuroplasticity mechanisms, which activate bundles of muscle fibers and drive an underactuated soft body with extremely complex dynamics (Pierrot-Deseilligny and Burke, 2005; Winter, 2009). On the face of it, we have no hope. If we can't reliably control a mathematically much simpler rigid robot, how could we possibly control an agent with a similar complexity to a human body?

On the other hand, could the complexity of the human body actually be a help and not a hindrance to the perception/control problem? The bio-inspired research agenda known as Embodied Intelligence suggests so (Pfeifer and Bongard, 2006; Cangelosi et al., 2015) and has spawned many different initiatives in this area. One approach is morphological computation, which investigates the ways that parts of the information processing burden can be offloaded to the body itself (Hauser et al., 2012; Müller and Hoffmann, 2017), through sensor morphology (as in the case of flies' eyes) (Iida and Nurzaman, 2016) or through the simplification of control (Eder et al., 2018). Physical reservoir computing uses the complexity of the body for general purpose computation

OPEN ACCESS

Edited by:
Inês Hipólito,
Humboldt University of Berlin, Germany

Reviewed by:
Tianqi Wei,
University of Edinburgh, United Kingdom
Udaya B. Rongala,
Lund University, Sweden

****Correspondence:***
Simon Birrell
sab233@cam.ac.uk

Received: *03 January 2022*
Accepted: *25 April 2022*
Published: *01 June 2022*

Citation:
Birrell S, Abdulali A and Iida F (2022) Reach Space Analysis of Baseline Differential Extrinsic Plasticity Control. Front. Neurorobot. 16:848084. doi: 10.3389/fnbot.2022.848084

(Nakajima, 2020). The richness of behavior of the peripheral nervous system, providing fast-acting reflexes and hierarchical and coordinated control (Côté et al., 2018) has been less applied to robotic research, which almost exclusively models the control problem as the agent's brain directly driving motor torques. Finally, investigations into different neuroplasticity schemes show that a surprising variety of complex, environmentally-aware behaviors can be spontaneously generated from simple, biologically plausible neuroplasticity rules within sensorimotor loops (Zappacosta et al., 2018).

These latter neuroplasticity-generated spontaneous behaviors, detailed in the book "The Playful Machine" (Der and Martius, 2012) and in related papers (Der and Martius, 2015, 2017), can drive simulated agents to explore and react to their environments in a manner that is highly suggestive of natural behaviors without building in any goals or higher-level planning of any sort. The most recent iteration of this research uses a particular neuroplasticity scheme called Differential Extrinsic Plasticity (DEP) (Der and Martius, 2015; Pinneri and Martius, 2018) to generate intriguing behaviors that are tightly coupled with the environment: a four-legged creature will appear to search for and find ways to climb over a fence; a humanoid will eventually clamber out of a hole it is trapped in. From our external observer perspective, these embodied behaviors appear to be goal-driven, but yet they are not. DEP has emerged as an interesting and promising candidate plasticity rule, but to date no practical applications for it have yet been found. It is an autonomous goal-free controller rather than a useful control method or component in a larger system.

A complicating factor in the practical usage of DEP is the current lack of an analytical solution, despite the research time invested. In all likelihood, even simple DEP systems are too complex to be fully described analytically and so research has tended to be empirical, treating DEP as a pre-existing natural phenomenon. This is not an insurmountable issue; it places DEP within the context of related research into algorithmic information and complexity theory, both areas cited in theories of the development of the human brain (Hiesinger, 2021). The behavior of DEP may not be solvable analytically even if it is deterministic. It may be undecidable: the only way to determine the output being to run it in simulation.

Given this, how can we study DEP and map out its potential? First, we must simplify: by temporarily removing environmental feedback we can map out baseline behaviors for DEP, following the methods employed in Pinneri and Martius (2018). Second, we must test the control-ability and limits of what DEP can do: to what extent can higher order systems "request" particular behaviors, and how much coverage will these behaviors provide in the context of a given task?

This paper takes the first steps in this direction. By employing "short-circuit DEP" (see below) and with a simple test case, where the output of short-circuit DEP drives a simulated 2 degree of

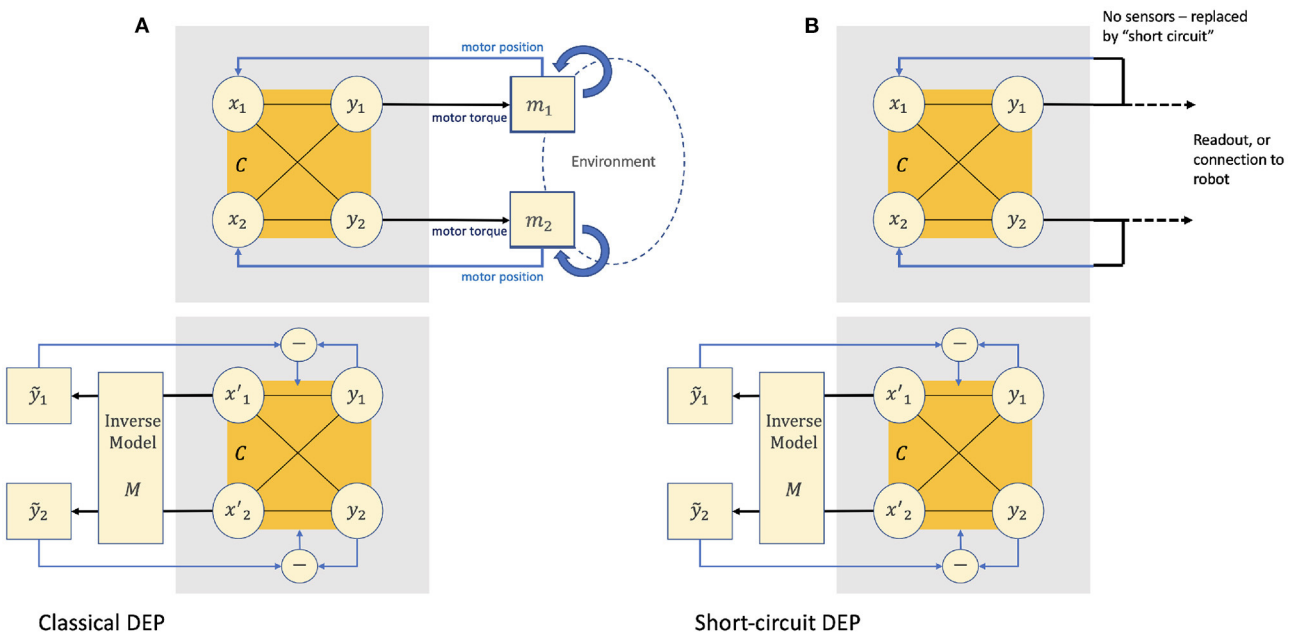

FIGURE 1 | In the "classical" version of Differential Extrinsic Plasticity (DEP) comprises two overlapping dynamical systems. [**(A)**, top] The input layer x of a feed-forward neural network C is driven by the motor positions. The output layer y drives the motor torques. These motors operate on the agents body in a given environment (potentially with collisions) and the resulting motor positions x^{t+1} will be fed back to the input to start the cycle again. [**(A)**, bottom] The positional information returned as x^{t+1} is also fed to an inverse model that infers the rate of change of the motor torques $\tilde{y}$ that would have generated them in the absence of environmental feedback. The difference between $\tilde{y}$ and the actual rate of change $\dot{y}$ captures the environmental effects of the agent's actions. This difference is then used to modify the weights of C. In the "short-circuited version" of DEP [**(B)**, top and bottom] there are no motors or sensors; the output y of the network is fed directly back to the input x.

freedom (DOF) robot arm, we show that DEP can be made to accomplish specific goals and that these goals cover a useful region of task space.

2. MATERIALS AND METHODS

2.1. How "Classical" Differential Extrinsic Plasticity Works

DEP describes a way of wiring motors and related sensors together with a neuroplasticity rule, such that a DEP-enabled agent produces a large set of "natural looking" behaviors that respond to interactions with the environment. Summarizing (Der and Martius, 2015; Pinneri and Martius, 2018), this section describes the equations that define the thermoplasticity rule for the "classic" version of DEP (see **Figure 1A**).

For a two-layer artificial neural network with input layer x_i, output layer y_i, weights C_{ij}, biases h_i, and a *tanh* activation function, the output activation is given by:

$$y_i = tanh(\sum_{j=1}^{n} C_{ij}x_j + h_i) \quad (1)$$

A simple feedback controller for an agent with rotary motors may then be constructed where y_i drives motor torques and x_i is driven by the resulting motor positions (see **Figure 1A** for a 2 degree of freedom example). In itself, this is not a very interesting controller, although given that the motor positions are ultimately determined not only by the applied torques but also by the body in which they're embedded and its interaction with the environment, nor is it trivial. This neural controller, the body and the environment together form a single dynamical system.

The behavior of this dynamical system can be overlaid by a second dynamical system driven by neural plasticity, that is, the evolution over time of the controller's weights. Many plasticity schemes have been studied (see **Table 1**). Hebbian learning modifies the weights based on the product of pre and post-synaptic activations[1]. Differential Hebbian Learning is similar (Zappacosta et al., 2018), but uses the product of the rates of change of the two activations.

Differential Extrinsic Plasticity extends Differential Hebbian learning by introducing an inverse model F that maps the rate of

TABLE 1 | Different plasticity schemes.

Plasticity scheme	Update rule
Hebbian learning	$\tau\dot{C}_{ij} = x_i y_j$
Differential Hebbian learning	$\tau\dot{C}_{ij} = \dot{x}_i \dot{y}_j$
Differential extrinsic plasticity	$\tau\dot{C}_{ij} = \dot{x}_i(\dot{y}_j + \delta\dot{y}_j) - C_{ij}$

[1]Using Hebbian learning here would give a system that resembles a continuous variable Hopfield Network, but with normalization and an inverse model.

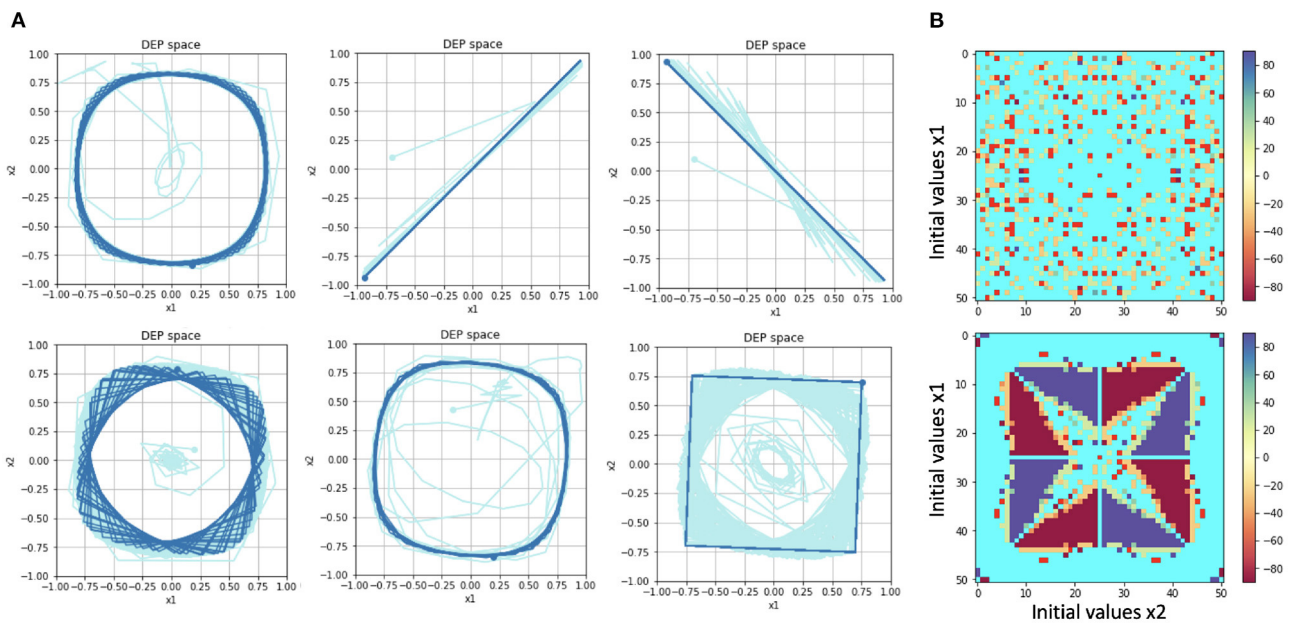

FIGURE 2 | (A) Examples of limit cycles reached by "Short-circuit" DEP. The phase diagrams show the trajectory of the system along the dimensions x_1 and x_2. The second and third trajectories oscillate between two endpoints. The other trajectories are all rotational. **(B)** Maps of the attractors reached based on initial values of x_1, x_2. The color bar refers to the rotational angle of the attractor, in the case of rotational attractors. Cyan refers to the non-rotational attractors shown in the second and third examples in **(A)**. The resulting map is shown in the top row. The map in the lower row reproduces the one shown in Pinneri and Martius (2018), but to generate it requires slightly altering the DEP algorithm (see text). Our version lacks their basins of attraction.

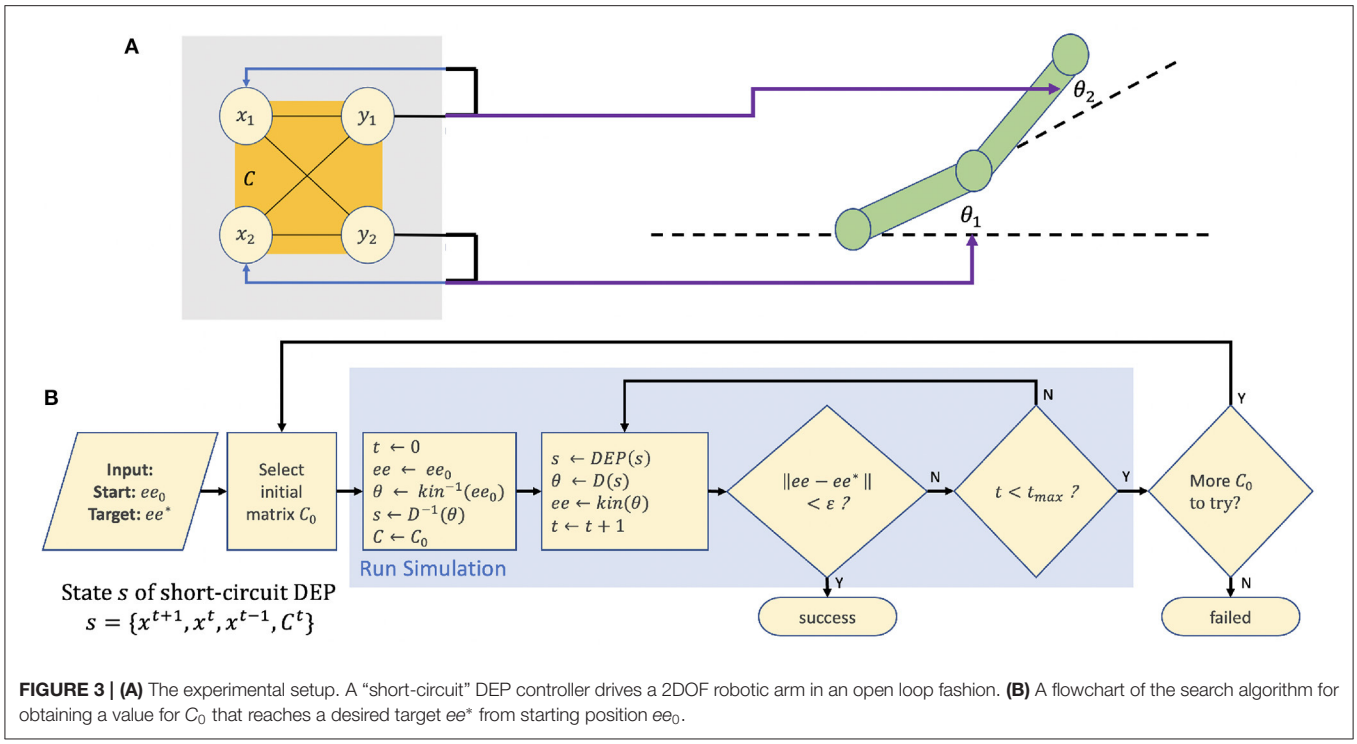

FIGURE 3 | (A) The experimental setup. A "short-circuit" DEP controller drives a 2DOF robotic arm in an open loop fashion. **(B)** A flowchart of the search algorithm for obtaining a value for C_0 that reaches a desired target ee^* from starting position ee_0.

change of received sensor values $\dot{x}^{t+1}$ back to the inferred rate of change of motor torques $\tilde{\dot{y}}^t$ that caused them:

$$\tilde{\dot{y}}^t = F(\dot{x}^{t+1}) \tag{2}$$

In most DEP implementations the inverse model F is implemented as a simple matrix M such that

$$\tilde{\dot{y}}^t = M\dot{x}^{t+1} \tag{3}$$

and, as in this paper, it is often assumed to be the identity matrix. F isn't required to be strictly accurate to reproduce DEP's behavior (Der and Martius, 2015).

The revised update rule uses $\tilde{\dot{y}}$ in place of $\dot{y}$ and adds a damping term. Dropping the time superscript t:

$$\tau \dot{C}_{ij} = \dot{x}_i \tilde{\dot{y}}_j - C_{ij} \tag{4}$$

One way to think about $\tilde{\dot{y}}$ is as the sum of the real historical value for $\dot{y}$ at t plus an error term $\delta\dot{y}$ with respect to the model F.

$$\tilde{\dot{y}} = \dot{y} + \delta\dot{y} \tag{5}$$

This substitution is shown in the final row of **Table 1**. Comparing it to the scheme for Differential Hebbian Learning shows how this "unexpected" environmental feedback is incorporated into the weight updates.

The weight matrix C is normalized to $\hat{C}$ at each time step with a factor κ and a parameter ρ that prevents a division by zero.

$$\hat{C} \leftarrow \kappa C/(\|C\| + \rho) \tag{6}$$

Finally, the activation rule is modified from Equation (1) to use the normalized $\hat{C}$ rather than C:

$$y_i = tanh(\sum_{j=1}^{n} \hat{C}_{ij} x_j + h_i) \tag{7}$$

The combination of these two overlaid dynamical systems produces an agent that cycles through a series of complex behaviors that are responsive to environmental feedback.

2.2. How "Short-Circuit" DEP Works

A simplified version of DEP was used in Pinneri and Martius (2018) for an empirical analysis of its behaviors. In this configuration there are no motors or sensors; the system output y is connected directly back to the system input x (**Figure 1B**). As $\tilde{\dot{y}}^t = M\dot{x}^{t+1}$ and $\dot{x}^{t+1} = \dot{y}^t$ and M is the identity matrix the update rule simplifies to

$$\tau \dot{C}_{ij} = \dot{x}_i \dot{y}_j - C_{ij} \tag{8}$$

This is effectively Differential Hebbian Learning with damping and normalization.

By eliminating the environment, the behaviors generated can be simplified to a set of predictable limit cycles, examples of which are shown in **Figure 2A**. The limit cycles reached depend on the initial conditions of the system, in particular the initial values for $x_1, x_2, y_1, y_2, \dot{x_1}, \dot{x_2}, \dot{y_1}, \dot{y_2}$, and C. In Pinneri and Martius (2018), all initial values were held constant except for x_1, x_2.

Following that paper, a map of the attractors reached based on differing initial conditions for x_1, x_2 is shown in **Figure 2B**. For each fixed point, the final 2x2 $\hat{C}$ matrix, is considered to be a

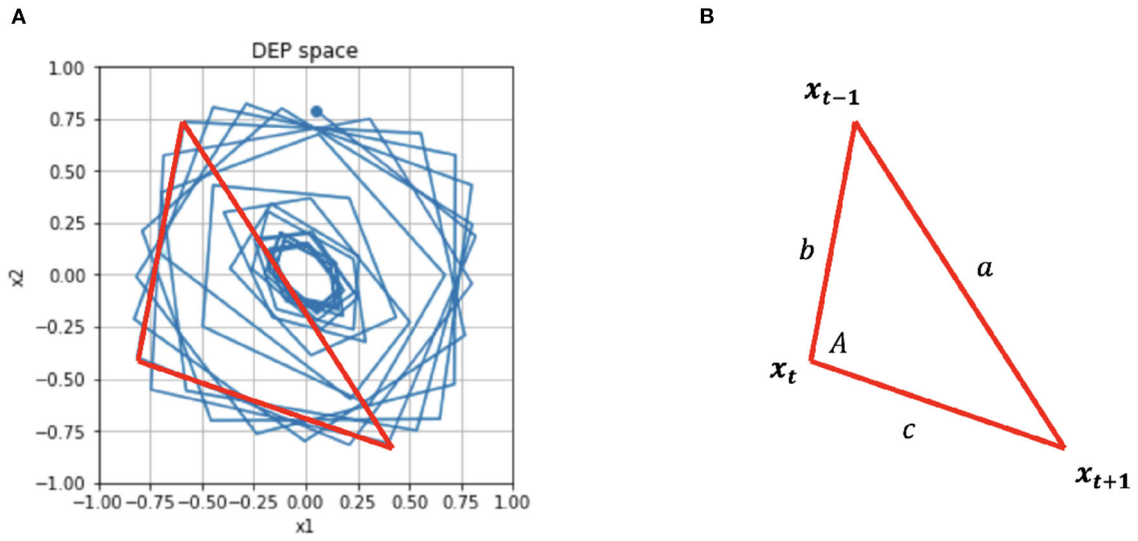

FIGURE 4 | (A) For the circular controller, θ_2 is derived from the angle between subsequent vertices of x. **(B)** This angle A is calculated by simple geometry.

rotational matrix and the corresponding angle is assigned a color. There are two cases of non-rotational matrices: a zero matrix (which is assigned bright red) and period-2 oscillations, such as the second and third examples in **Figure 2A**, which are assigned cyan. The resulting map is shown in **Figure 2B** (top).

It should be noted that (Pinneri and Martius, 2018) obtained a different pattern, as that paper used code that inadvertently reset the C matrix to zero at $t = 2$, generating different dynamics (private communication). Their results were reproduced (with the necessary code modification) in **Figure 2B** (bottom). For our experiments we followed the strict interpretation of the DEP equations. The attractors identified are the same, but the attractor map with respect to initial conditions is different; the "basins of attraction" cited in that paper being absent. In our opinion, these basins are an artifact of the previous code base and not intrinsic to DEP as such.

In the present paper's experiments, as well as x_1, x_2, the initial value C_0 of the matrix C is also varied. It was discovered that choosing different values for C_0 elicits different trajectories and ultimate limit cycles for each combination of the initial values x_1, x_2. One way of looking at this is to say that different C_0 can select different behaviors for a given initial x_1, x_2.

2.3. The Experimental Setup

In the two experiments described, the "short circuit" DEP system is used to drive a simple 2 degree of freedom robotic arm (see **Figure 3A**).

The state s of the short-circuit DEP system can be fully captured as

$$s = \{x^{t+1}, x^t, x^{t-1}, C^t\} \tag{9}$$

so that at each timestep $s^{t+1} \leftarrow DEP(s^t)$.

We can then use a robot arm with segment lengths l_1, l_2, here 0.5 m, to "read out" the state s of DEP. The joint angles θ, comprising θ_1, θ_2, are driven by a "driver" function D that is specific to a given task type, such that

$$\theta = D(s) \tag{10}$$

Note that this is an open-loop controller. None of the reported benefits of environmentally-aware "Classic" DEP are used here, in line with the goal of learning to control a very simple DEP system. The position of the robot's end effector can be considered as a simple transformation or readout of DEP's internal state s.

Two types of task are considered. In the first, the goal is for the robot arm's end effector that starts at position ee_0 to **reach** an arbitrary target position $ee^\star$. For this type of task, function $D(s) = D_{reach}(s)$ is simply

$$\begin{aligned} \theta &= D_{reach}(\{x^{t+1}, x^t, x^{t-1}, C^t\}) \\ &= \pi x^{t+1} \\ &= \pi y \end{aligned} \tag{11}$$

In other words, the output of y of short-circuit DEP directly drives the motor angles θ.

In the second type of task, the goal is for the end effector to trace a **circular** trajectory of arbitrary radius r. Here, $D(s) = D_{circle}(s)$ and we leverage the angle A between the vectors $x^{t+1} - x^t$ and $x^t - x^{t-1}$. See **Figure 4** for the simple geometry that defines a, b, c. Then, the two joint angles θ_1, θ_2 can be defined in the new driver function:

$$\begin{aligned} \theta &= D_{circular}(\{x^{t+1}, x^t, x^{t-1}, C^t\}) \\ \theta_i &= \begin{cases} \omega t & \text{if } i = 1 \\ cos^{-1}\left(\frac{b^2+c^2-a^2}{2bc}\right) & \text{if } i = 2 \end{cases} \end{aligned} \tag{12}$$

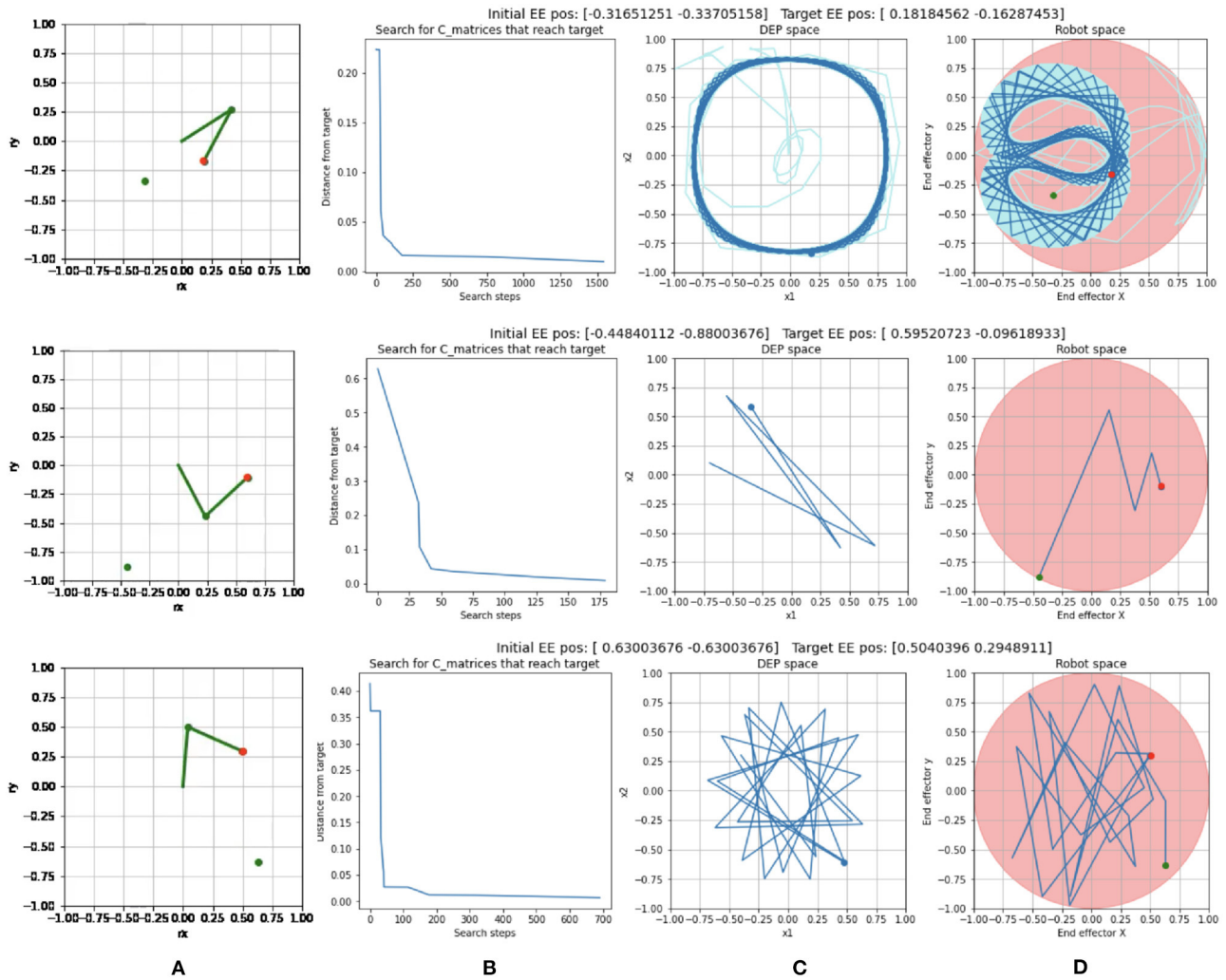

FIGURE 5 | Examples of Reaching trials with: **(A)** different end effector start positions (green dot) and target positions (red dot). **(B)** The progress of the search for a solution C matrix. **(C)** The solution trajectory in DEP space for the successful trial. **(D)** The solution trajectory in robot space for the successful trial. The most recent positions are shown in dark blue while the early part of the trajectory is in light blue.

At a fixed point of C, $\hat{C}^{t+1} \rightarrow C^t$ and if $|x| << 1$,

$$x^{t+1} = tanh(\hat{C}x^t),$$
$$x^{t+1} \approx \hat{C}x^t \tag{13}$$

Under these conditions, x is rotating around the origin in DEP space and the angle between every other point is approximately constant. As this angle drives θ_2 then θ_2 will also be a constant. θ_1 is a constant, so the robot will describe a circle.

2.4. The Search Algorithm for C_0

Given an input of an initial end effector position ee_0 and target position or trajectory $ee^\star$, our goal is to obtain an initial matrix C_0 that will drive the system to reach $ee^\star$. C_0 is obtained by a search algorithm, detailed in **Figure 3B**.

The 2×2 matrix C_0 has four parameters that here each vary between -1 and $+1$. The algorithm linearly divides the range of each parameter into eight values, giving $8 \times 8 \times 8 \times 8 = 4{,}096$ possible values for C_0. A simple grid search is performed, with each value being trialed in a rollout of 20,000 time steps.

In the case of the reaching task, at each time step of the rollout, if the distance between ee_t and $ee^\star$ is within a given tolerance ϵ, then success is declared. 10 random starting positions ee_0 and 10 random targets $ee^\star$ were combined to give 100 trials, each of which is an execution of the algorithm in **Figure 3B**.

In the case of the circular task, success is declared after a full rotation of the end effector, where the mean squared radius error with respect to r is less than ϵ. Five random starting positions ee_0 and five random radii $r^\star$ were combined to give 25 trials, each of which is an execution of the algorithm in **Figure 3B**.

The experiments were implemented in Python on Jupyter notebooks. The full source code may be downloaded from GitHub[2], inspected and run.

[2]https://github.com/SimonBirrell/dep-control

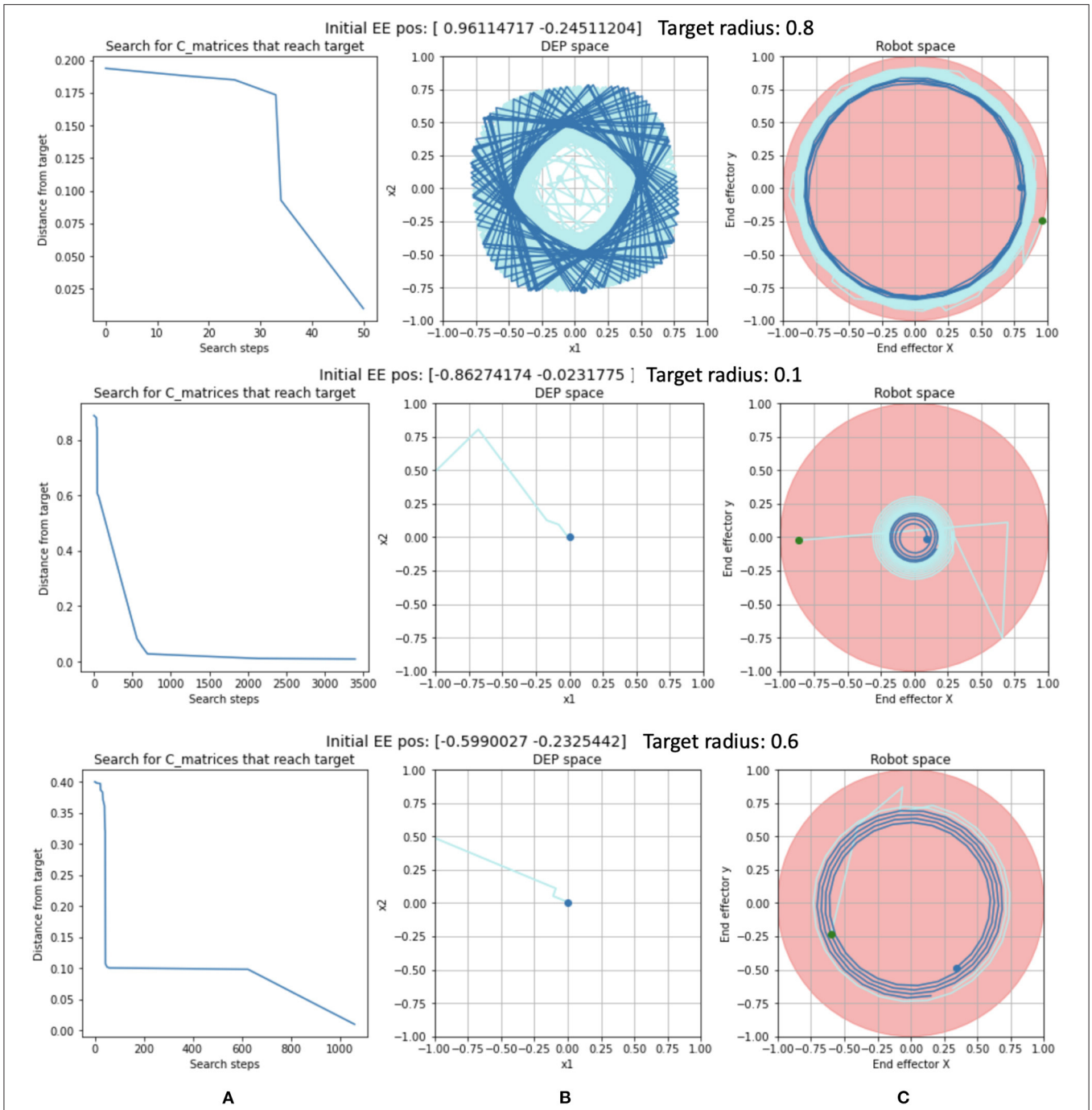

FIGURE 6 | Examples of Circular Trajectory trials with: **(A)** The progress of the search for a solution C matrix. **(B)** The solution trajectory in DEP space. **(C)** The solution trajectory in robot space. The most recent positions are shown in dark blue while the early part of the trajectory is in light blue.

3. RESULTS

The trajectories in DEP space produced in the experiments generally consisted of a transient phase where the system "wanders" in x_1, x_2 followed by a limit cycle phase. The Reaching task leveraged both transient and limit cycle phase, while the Circular task leveraged the limit cycles.

3.1. The Reaching Task

One hundred trials of the Reaching task were performed. In every case, the system reported success: it found a path to all end effector targets from all end effector starting positions. The tolerance $\epsilon|$ had a value of 0.01 m.

Trajectory examples are show in **Figure 5**. In the example in the top row, the search algorithm tested 1,541 C_0 matrices

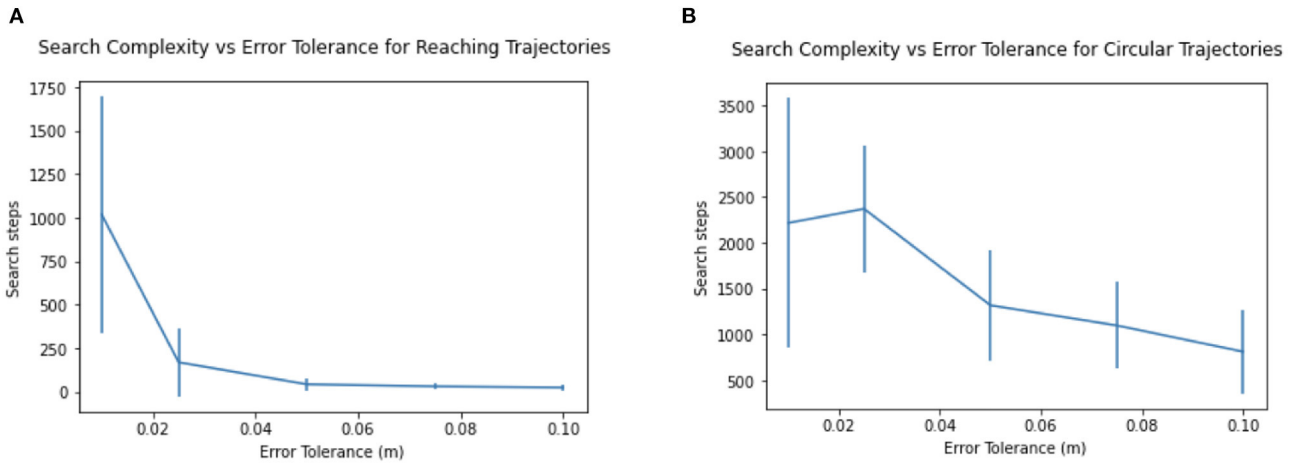

FIGURE 7 | Search complexity and variance increases with lower error tolerance, for **(A)** Reaching trajectories and **(B)** Circular trajectories.

(**Figure 5B**) before finding a value that caused the end effector (**Figure 5A**) to reach the desired target. The solution itself in DEP space (**Figure 5C**) and robot space (**Figure 5D**) show that the system had entered a rotational limit cycle before reaching the target.

A second example, in the second row of **Figure 5** shows a contrasting example where a solution was found after testing only 179 search steps. In the solution, the system was still in a transient phase when it hit the target, at only six time steps into the rollout.

3.2. The Circular Trajectory Task

Twenty-five trials of the Circular task were performed. In every case, the system reported success: it managed to describe a circular trajectory of at least one rotation where the mean squared radius error with respect to the desired radius was less than ϵ, in this case 0.01 m.

Trajectory examples are shown in **Figure 6**. In the example in the top row, the search algorithm required a mere 50 steps (**Figure 6A**) to find a limit cycle in DEP space (**Figure 6B**) that completed a circle of the desired radius (**Figure 6C**) after 551 time steps of the rollout.

In the second example, in the second row of **Figure 6**, the search algorithm required 3,412 search steps (out of a maximum of 4,096) (**Figure 6A**) to find a solution in DEP space (**Figure 6B**) that completed a circle after 332 time steps of the rollout.

The relationship of search time to tolerance ϵ can be seen in **Figure 7**. For lower, more stringent, error tolerances ϵ, the number of search steps required increases, as does its variance. Increasing the tolerance required for reaching even slightly (say from 0.01 to 0.025 m) reduces the search steps required by 75%.

4. DISCUSSION AND FUTURE WORK

The controller described in this paper is unlikely to signal the end of inverse kinematics. To borrow Dr. Johnson's phrase, it "is like a dog's walking on his hinder legs. It is not done well; but you are surprised to find it done at all" (Boswell, 1791). Why do these results, and DEP in general, matter? We can answer in three ways.

4.1. DEP as a Control Mechanism

First, what is the prognosis for DEP as a control system? The present controller has reduced a high dimensional control problem to one of simple selection of one of 4,096 different discrete values of the C_0 matrix. The original motivation for this paper was to find a way to leverage DEP within the context of Reinforcement Learning. C_0 provides a low dimensional interface for higher level systems to exploit. Yet most of the solutions are indirect, taking time for the end effector to reach its goal.

The search algorithm could be extended to optimize for lower time steps to reach the desired target position or trajectory. Different trajectory types could be produced with different driving functions, although fewer functions would be preferable to more. Driving functions could be abstract, as they are here, or derived from physical models of body elements, such as springs, tissue, or muscles.

There is scope for improving the search algorithm itself from a simple grid search, depending on what patterns, if any, can be found in the mapping of target to C_0. Are there basins of attraction for C_0? Is this controller learnable in a way that generalizes?

Once understanding of the core behavior of "short-circuit" DEP has improved, environmental awareness, one of the core supposed advantages of the neuroplasticity rule, could be reintroduced. This opens the way to recovery from perturbations and short term, "reflex" reactions to changes in the environment.

4.2. The Study of DEP

A continuing expressed frustration in the DEP literature is the lack of a full analytical treatment of DEP behavior. That may be due a lack of human resources applied to the problem, or it may be that a full treatment is simply intractable. Some algorithms are mathematically "undecidable," which is to say that their

behavior cannot be predicted without executing the algorithm itself. Perhaps DEP falls into this category.

In either case, this paper follows recent work in taking an empirical, engineering approach to analysing DEP, rather than a theoretical treatment. There remain many questions to be answered.

DEP has produced some fascinating simulations, with realistic looking and intriguing behaviors, such as gait switching, overcoming obstacles, and interaction with devices such as handles. How much of the observed behaviors are due to DEP as a neuroplasticity rule and how much are due to the particular body morphology of the simulated agents? Passive walkers also produce realistic behaviors and respond to the environment in a limited way, yet they have no neuroplasticity at all. Clearly, the agents behavior is generated by the complete system of neuroplasticity plus body plus environment. How we can disentangle the contributions of each?

Finally, does DEP scale? What are the limit cycles of higher dimensional DEP systems? Our understanding of DEP behavior is only just beginning.

4.3. Leveraging Pre-existing Complexity

DEP is an example of self-organization in action: of complexity generated from simple rules. Self-organization is easy to spot, but hard to design, yet may be necessary to enable long-term learning processes such as evolution to work effectively (Kauffman, 1995). Classical DEP is a system that, in that evocative phrase, exists "on the edge of chaos," producing a rich set of behaviors even in the "short circuit" version. Is this complexity useful to agents, or is it a simple artifact?

The leverage of pre-existing complex behaviors is seen in Physical Reservoir Computing (PRC), a field that applies a thin layer of learning over highly complex, pre-existing dynamics in a real or simulated body. A PRC system leverages a set of dynamical behaviors as if they were basis functions and combines them using a shallow artificial neural network. The network can then be trained to perform some desired function. The dimensionality of a problem that might require training a very deep neural network has been reduced to that of training a shallow one.

In the case of PRC, the pre-existing complexity is physical. In other cases it may be algorithmical. A curious example is the history of procedural content generation in computer games (Smith, 2015). The practice originated over 40 years ago with the need to generate details of thousands of planets in highly resource-constrained computers. Rather than store such details, they were generated from the Fibonacci sequence, passed through an interpretive function analogous to our "driver function." By using a predictable mathematical sequence that has inherent complexity, a vast amount of content could be generated *ex nihilio*. Other examples of Algorithmic Information have been studied, such as the "undecidability" and Turing completeness of Rule 110 (Cook et al., 2004).

What is unclear is whether and to what extent nature has leveraged these potential sources of complexity. In developmental biology, there is a gap between the information specified in the genome and the complexity of the end product (Hiesinger, 2021). In learning there is a gap between the mechanisms we have available and the complexity of the problems to solve. Does pre-existing complexity play a part in closing this gap? Is DEP an example of this?

Differential Extrinsic Plasticity remains a fascinating phenomenon. Neuroplasticity remains an under-explored component of Embodied Intelligence and a rich opportunity for future work.

DATA AVAILABILITY STATEMENT

The original contributions presented in the study are publicly available. This data can be found here: GitHub, https://github.com/SimonBirrell/dep-control.

AUTHOR CONTRIBUTIONS

SB researched DEP, performed the experiments, and wrote the paper under the supervision of FI, with suggestions and comments from AA. All authors contributed to the article and approved the submitted version.

FUNDING

This project was possible thanks to EPSRC Grant EP/L01-5889/1, the Royal Society ERA Foundation Translation Award (TA160113), EPSRC Doctoral Training Program ICASE Award RG84492 (cofunded by G's Growers), EPSRC Small Partnership Award RG86264 (in collaboration with G's Growers), and the BBSRC Small Partnership Grant RG81275.

ACKNOWLEDGMENTS

The authors would like to thank Josie Hughes for her help and encouragement. Also to Georg Martius and Cristina Pinneri for giving access to the code behind their paper.

SUPPLEMENTARY MATERIAL

The Supplementary Material for this article can be found online at: https://www.frontiersin.org/articles/10.3389/fnbot.2022.848084/full#supplementary-material

Supplementary Video 1 | Evolution of the dynamics in the first row of **Figure 5**.

Supplementary Video 2 | Evolution of the dynamics in the second row of **Figure 5**.

Supplementary Video 3 | Evolution of the dynamics in the third row of **Figure 5**.

Supplementary Video 4 | Evolution of the dynamics in the first row of **Figure 6**.

Supplementary Video 5 | Evolution of the dynamics in the second row of **Figure 6**.

Supplementary Video 6 | Evolution of the dynamics in the third row of **Figure 6**.

REFERENCES

Boswell, J. (1791). *The life of Samuel Johnson, LL.D*. London: Charles Dilly; Henry Baldwin.

Brockman, G., Cheung, V., Pettersson, L., Schneider, J., Schulman, J., Tang, J., et al. (2016). Openai gym. *arXiv [Preprint] arXiv:1606.01540*. doi: 10.48550/arXiv.1606.01540

Cangelosi, A., Bongard, J., Fischer, M. H., and Nolfi, S. (2015). "Embodied intelligence," in *Springer Handbook of Computational Intelligence*, eds J. Kacprzyk and W. Pedrycz (Springer), 697–714. doi: 10.1007/978-3-662-43505-2_37

Cook, M., et al. (2004). Universality in elementary cellular automata. *Complex Syst*. 15, 1–40.

Côté, M.-P., Murray, L. M., and Knikou, M. (2018). Spinal control of locomotion: individual neurons, their circuits and functions. *Front. Physiol*. 9:784. doi: 10.3389/fphys.2018.00784

Der, R., and Martius, G. (2012). *The Playful Machine - Theoretical Foundation and Practical Realization of Self-Organizing Robots*. Berlin Heidelberg: Springer.

Der, R., and Martius, G. (2015). Novel plasticity rule can explain the development of sensorimotor intelligence. *Proc. Natl. Acad. Sci. U.S.A*. 112, E6224–E6232. doi: 10.1073/pnas.1508400112

Der, R., and Martius, G. (2017). Self-organized behavior generation for musculoskeletal robots. *Front. Neurorobot*. 11:8. doi: 10.3389/fnbot.2017.00008

Eder, M., Hisch, F., and Hauser, H. (2018). Morphological computation-based control of a modular, pneumatically driven, soft robotic arm. *Adv. Robot*. 32, 375–385. doi: 10.1080/01691864.2017.1402703

Hauser, H., Ijspeert, A., Füchslin, R., Pfeifer, R., and Maass, W. (2012). The role of feedback in morphological computation with compliant bodies. *Biol. Cybern*. 106, 595–613. doi: 10.1007/s00422-012-0516-4

Hiesinger, P. R. (2021). *The Self-Assembling Brain: How Neural Networks Grow Smarter/Peter Robin Hiesinger*. Princeton, NJ: Princeton University Press. doi: 10.1515/9780691215518

Iida, F., and Nurzaman, S. G. (2016). Adaptation of sensor morphology: an integrative view of perception from biologically inspired robotics perspective. *Interface Focus* 6:20160016. doi: 10.1098/rsfs.2016.0016

Kauffman, S. A. (1995). *At Home in the Universe : The Search for Laws of Self-Organization and Complexity/Stuart Kauffman*. New York, NY; Oxford: Oxford University Press.

Müller, V. C., and Hoffmann, M. (2017). What is morphological computation? On how the body contributes to cognition and control. *Artif. Life* 23, 1–24. doi: 10.1162/ARTL_a_00219

Nakajima, K. (2020). Physical reservoir computing—an introductory perspective. *Japanese J. Appl. Phys*. 59:060501. doi: 10.35848/1347-4065/ab8d4f

Pfeifer, R., and Bongard, J. (2006). *How the Body Shapes the Way We Think: A New View of Intelligence*. Cambridge, MA: MIT Press. doi: 10.7551/mitpress/3585.001.0001

Pierrot-Deseilligny, E., and Burke, D. (2005). *The Circuitry of the Human Spinal Cord: Its Role in Motor Control and Movement Disorders*. Cambridge: Cambridge University Press. doi: 10.1017/CBO9780511545047

Pinneri, C., and Martius, G. (2018). "Systematic self-exploration of behaviors for robots in a dynamical systems framework," in *Proceedings of Artificial Life XI* (Cambridge, MA: MIT Press), 319–326. doi: 10.1162/isal_a_00062

Smith, G. (2015). "An analog history of procedural content generation," in *FDG* (Boston, MA). doi: 10.1201/9781315313412-9

Sutton, R. S., and Barto, A. G. (2018). *Reinforcement Learning: An Introduction, 2nd Edn*. Cambridge, MA: The MIT Press.

Winter, D. A. (2009). *Biomechanics and Motor Control of Human Movement*. Hoboken, NJ: John Wiley & Sons. doi: 10.1002/9780470549148

Zappacosta, S., Mannella, F., Mirolli, M., and Baldassarre, G. (2018). General differential Hebbian learning: capturing temporal relations between events in neural networks and the brain. *PLoS Comput. Biol*. 14:e1006227. doi: 10.1371/journal.pcbi.1006227

Conflict of Interest: The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.