Physical Scaffolding Accelerates the Evolution of Robot Behavior

David Buckingham*,**
Tufts University

Josh Bongard†
University of Vermont

Abstract In some evolutionary robotics experiments, evolved robots are transferred from simulation to reality, while sensor/motor data flows back from reality to improve the next transferral. We envision a generalization of this approach: a simulation-to-reality pipeline. In this pipeline, increasingly embodied agents flow up through a sequence of increasingly physically realistic simulators, while data flows back down to improve the next transferral between neighboring simulators; physical reality is the last link in this chain. As a first proof of concept, we introduce a two-link chain: A fast yet low-fidelity (*lo-fi*) simulator hosts minimally embodied agents, which gradually evolve controllers and morphologies to colonize a slow yet high-fidelity (*hi-fi*) simulator. The agents are thus *physically scaffolded*. We show here that, given the same computational budget, these physically scaffolded robots reach higher performance in the hi-fi simulator than do robots that only evolve in the hi-fi simulator, but only for a sufficiently difficult task. These results suggest that a simulation-to-reality pipeline may strike a good balance between accelerating evolution in simulation while anchoring the results in reality, free the investigator from having to prespecify the robot's morphology, and pave the way to scalable, automated, robot-generating systems.

Keywords
Reality gap, minimal cognition, scaffolding, evolution, simulation

1 Introduction

Evolutionary robotics [29] evaluates and modifies a large number of robots in some task environment to find those that exhibit desired behavior. Computer simulations of robots and their task environments are frequently used in lieu of physical robots to save time and avoid damage to hardware. However, significant challenges arise when solutions evolved in simulation are transferred to physical robots. Evolution often exploits inaccuracies in a simulation so that when agents that were successful in simulation are instantiated as physical robots, crossing the so-called *reality gap* [21], their performance deteriorates.

Various approaches have been proposed to combat the reality gap problem, including proactive methods that increase robustness of behavior before transferral [21, 27], retroactive methods that return data obtained from reality to increase the likelihood of success during the next transferral [5, 26, 11], and extreme methods that abandon simulation altogether [35, 7]. Here, we propose a

* Contact author.
** Department of Computer Science, Tufts University, Medford, MA 02155. E-mail: david.buckingham@tufts.edu
† Department of Computer Science, University of Vermont, 205 Farrell Hall, Burlington, VT 05405. E-mail: josh.bongard@uvm.edu

new method: Establish a chain of task environments, beginning with minimally embodied agents hosted in low-fidelity simulations, progressing to more embodied agents hosted in higher-fidelity simulations, and concluding with the automated manufacture of physical robots (Figure 1). Minimally embodied agents are less realistic, more abstract representations of real robots. They can be simulated with fewer computational resources than their higher-fidelity counterparts, but are similar enough that at least some aspects of agent control apply across simulations or in reality.

Second, this chain should be bidirectional, with agents flowing gradually forward toward manufacture, but also backward to receive more simulation, during the evolutionary process. Thus, sensor/motor data generated in one simulator (or in reality) may flow backward to a previous simulator and be used to improve the likelihood of the successful future transferral of agents between these neighboring stages.

Third, in all stages of the pipeline, agents are evolved to exhibit the same desired behavior. While evolution should reward agents for their ability to perform the desired task, it should also reward them for being evaluated in simulations with higher fidelity, that is, for being farther along the chain of task environments. If this is successfully implemented, one should observe agents of increasing ability and transferability in lower-fidelity simulations invading and displacing less capable and less transferable agents in higher-fidelity simulations (and reality). Here, ability refers generally to a controller's performance with respect to a behavioral task in one or more environments, while transferability implies similar performance between environments.

Finally, the morphologies of embodied agents in the intermediate simulators should evolve along with their controllers. If a controller is transferred forward from a non- or minimally embodied agent into the body of an embodied agent in a neighboring simulator, that body may either hinder or facilitate that transferral. In other words, one morphology may support exploitation achieved by evolution in the lower-fidelity simulator, while another morphology may invalidate that exploitation. Consider an example: Energy-efficient walking may evolve in a simple biped in which the legs are approximated as point masses. The controller that generates this walking behavior may fail when transferred into a more realistic bipedal robot with nonuniform mass distributions along its legs. However, it may transfer well into a different, more realistic bipedal robot that has uniform mass distributions along its legs. Thus, morphologies may evolve that facilitate the flow of controllers forward along the pipeline by enabling the retention of the controllers' function in increasingly high-fidelity simulations. The best of these morphologies can then be manufactured and the best controllers emplaced within them.

We hypothesize that such a pipeline may be superior to existing methods for the following reason. Evolution in simulation, coupled with a few physical trials, can in some cases reduce the time

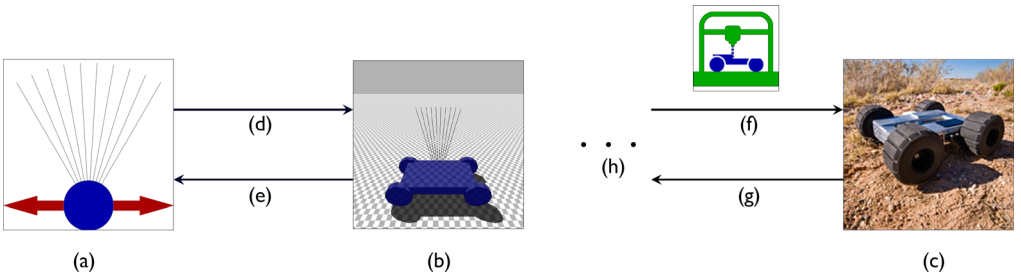

Figure 1. The proposed simulation-to-reality pipeline. (a) A fast yet crude lo-fi simulation hosts non-embodied agents. (b) A slower yet more accurate hi-fi simulation hosts embodied agents. (c) A physical instantiation of one of the embodied agents. (d) Controllers and/or bodies may flow up into the more accurate simulators. (e) Data may flow back down to the cruder simulations to ease subsequent transferrals. (f) 3D printers may transfer embodied agents from simulation to reality. (g) Data flows back from reality to ease the next simulation-to-reality transferral. (h) The pipeline is scalable and may include many simulations to smooth the physicality gradient. This report describes our implementation of parts (a), (b), (d), and (e) of this diagram. (Robot photo, by Randy Montoya, of the Precision Urban Hopper robot developed by Sandia National Laboratories and Boston Dynamics.)

required to reach some level of functionality with respect to performing evolution all on a physical robot [5, 26, 11]. By extension, then, evolution in a faster, lo-fi simulator with gradual transferral to a slower, hi-fi simulator may produce embodied agents of equal functionality in less time than evolving only in the hi-fi simulator. This performance advantage will only be obtained, however, if the lo-fi simulation is a good *scaffold* for the hi-fi simulation, that is, if agents that evolve to perform a behavior in the lo-fi simulator before being transferred to the hi-fi simulator are able to re-evolve this behavior in their new environment more rapidly than if they had never been exposed to the lo-fi simulator.

We refer to this process as *physical scaffolding*: Agents gradually evolve to become increasingly embodied in higher-fidelity simulators (and eventually in reality) if appropriate morphologies evolve to house and pass on evolving controllers arriving from lower-fidelity simulators. We introduce this term to distinguish this phenomenon from robot shaping or environmental scaffolding [13, 36], in which the robot's task environment gradually becomes more challenging, and from morphological scaffolding [3, 6], in which the robot's morphology becomes more complex in the same task environment.

As a first step toward demonstrating the performance benefits of such a pipeline, we investigated a two-simulator chain composed of a lo-fi and a hi-fi simulator. We tested the hypothesis that, given a fixed computational budget, evolution scaffolded by the lo-fi simulation yields robots in the hi-fi simulation that outperform those produced by evolution without scaffolding. We found that this is indeed the case, but only at a sufficient level of task difficulty, and only if the bodies of the embodied agents are evolved along with their controllers.

The rest of this article is organized as follows: The next section provides a literature review to place this work in context. Section 3 describes our methods. Section 4 provides our results. We discuss these results in Section 5, and provide concluding remarks in Section 6.

2 Related Work

We have developed a novel type of evolutionary scaffolding to address the reality gap, and applied it to a minimal cognition problem. Previous work has evolved solutions to minimal cognition problems in minimal simulations. Our technique uses minimal simulations as a scaffold to help produce minimally cognitive behaviors in a more realistic simulation. In this section, we discuss related work relevant to the reality gap problem, scaffolding, and minimal cognition.

2.1 The Reality Gap

The reality gap problem [21] refers to the challenge of transferring a robot (or just its controller) evolved in simulation to reality while retaining its behavior. If evolution in simulation produces a robot that performs a particular behavior in simulation, it will usually not perform the same behavior once instantiated as a physical robot in the real world. First, even the best simulations cannot capture all parts of the dynamical interaction between a robot's controller, its body, and the surrounding environment. Furthermore, the real world introduces unpredictable features into this interaction, such as sensor noise, air currents, and vibrations. Finally, evolutionary algorithms can discover and exploit unrealistic nuances of physics engines, leading to high performance on fitness functions that do not correspond to real-world possibilities. Efforts to address the reality gap problem include avoiding simulation altogether, constraining or dynamically modifying the simulation so that it accurately models aspects of robot behavior most important for successful transfer to reality, and evolving the robot in multiple simulations or in both simulation and reality.

One approach is to forgo simulation (or minimize its use) and instead evaluate controllers on physical robots. For example, there has been extensive work evolving behaviors directly on Khepera robots [29]. Hornby et al. [18] evolved gaits on the quadrupedal AIBO robot that were robust enough to withstand transfer to other AIBO robots and to walking on new types of surfaces. These results demonstrate that evolutionary robotics can avoid the reality gap problem by not using

simulation: Robot behavior can be effectively evolved directly on hardware. However, the large number of trials needed for evolution discourages the use of physical robots during the training period [28]: Evolution in simulation can be much faster, easier, and cheaper. For example, in Hornby et al. [18] an evolutionary run consisting of 500 evaluations required approximately 25 h, whereas evaluations in simulation might be several orders of magnitude faster.

Others have tried to shrink the size of the reality gap by explicitly constraining the solution space. The Golem project [34] used simulation to evolve robot controllers and morphologies simultaneously. They used a quasi-static simulation, which required the robot to be statically stable at each time frame. Although this constrained possible behaviors, such quasi-static motion is more easily transferred to reality. Francesca et al. [16] framed the reality gap problem in the context of a bias-variance tradeoff. They proposed to inject bias into the robot design process in order to decrease the variance. This injection of bias involved reducing the representational power of the controller by using probabilistic finite state machines instead of neural networks, thereby mitigating overfitting to the simulated environment. They used this technique to produce swarm robot controllers having comparable performance in simulation and in reality for aggregation and foraging tasks. The application of this technique to a specific problem requires an "expert" to adapt it to the given robotic platform and to pair it with an appropriate optimization process. Boeing et al. [4] explicitly constrained their evolutionary search space to produce approximate control solutions. This allowed evolution in simulation of a spline-based control system for bipedal locomotion that could effectively cross the reality gap.

Other work has focused on evolving agents that are robust to differences between simulation and reality. Jakobi et al. [21] used a physics-based simulation to evolve controllers for a Khepera robot to perform obstacle-avoidance and light-seeking tasks, and added noise to the simulated sensors. The simulation, which was based on empirical measurements of the physical system, captured important features of the Khepera's interactions with the environment but excluded other details that would unnecessarily burden computation. Varying amounts of noise were added, affecting the quality of the transfer. Because parts of the physical system not relevant to the target behavior either were not modeled or were obscured by noise, evolution could not exploit them, and controllers were successfully transferred. In later work [22–24], varying levels of noise were used in conjunction with minimal simulations to cross the reality gap. The use of models that captured only some parts of the environment forced evolution only to use certain key aspects of the agent-environment dynamics and not to rely on parts of the simulation that might not correspond to reality.

Sometimes evolution begun in simulation can be continued in reality to correct losses suffered during the transfer. Pollack et al. [33] used coevolution to simultaneously evolve robot bodies and controllers in simulation. Successful agents were then produced as physical robots whose controllers were subjected to further evolution. Thus, evolution in physical robots fine-tuned the results of evolution in simulation, facilitating transfer across the reality gap. Miglino et al. [28] addressed the reality gap in an object-avoidance task, using a Khepera robot and a simple simulated environment. The simulation was tuned to the dynamics of a particular robot-environment interaction by sampling the physical robot's sensors and actuators. The gap between simulation and reality was further reduced by adding noise to the simulated sensors. If a decrease in performance still occurred when the controller was transferred to the real environment, perfectly performing results were obtained by continuing the evolutionary process with the physical robot. Floreano and Urzelai [15] evolved robot neural controllers that used synaptic plasticity to develop new behaviors "on the fly." These plastic controllers were then transferred to physical robots, where they could adapt online to new conditions.

More recently, Koos et al. [25, 26] used a multi-objective evolutionary algorithm to search simultaneously for controllers that produced the desired behavior and for controllers with high transferability. Transferability was defined by a simulation-to-reality disparity measure that was evaluated with a surrogate model of the true simulation-to-reality disparity. Evaluation of physical robot behavior was needed only to develop these surrogate models. Their results suggest an antagonism between fitness in simulation and transferability: Because the fitness function in simulation is only

an approximation of reality, it is necessarily "misleading," and the best solutions in simulation may not perform well in reality.

Some experiments have subjected robot controllers to multiple simulation environments. Celis and Bongard [10] employed two different simulators that made different assumptions about how to approximate physical laws. Quadrupedal robots were then evolved to succeed in both simulators. It is possible that by using multiple simulators, evolution could be prevented from exploiting inaccuracies that are unique to one of them. By extension, the resulting evolved controllers might then transfer well to reality, although this last step was not conducted.

In another approach to the reality gap problem, Bongard et al. [5] evolved the simulator itself to better approximate reality. This not only enabled a physical robot to evolve a simulation of itself, which it then used to evolve fast gaits, but it also re-evolved the simulation to reflect unexpected damage and recover locomotion despite the injury. In a related project, Cully et al. [11] developed an intelligent trial-and-error algorithm that allowed a robot to adapt quickly to physical damage or novel environmental conditions. Before the robot was deployed, a simulation of the robot was used to create a map of the space of high-performing behaviors. Because they had not yet been tested in reality, the predicted performance of points on the map was initially assigned a low confidence. When the robot was damaged, an information acquisition function balanced exploration of low-confidence areas of the map and exploitation of points with expected high performance, allowing for the rapid discovery of compensating behaviors. Zagal et al. [42] combined evaluation in physical robots and in simulation into a single evolutionary framework. This framework automated the determination of simulation characteristics to capture relevant features of the physical system. It involved three simultaneous optimization processes: learning of the robot controller to minimize error in a simulated environment, learning of the robot controller to minimize error in a physical robot, and learning of simulator parameters to minimize differences between the fitness obtained in reality and in simulation. This technique obtained locomotive gaits more than double the speed of a manually developed gait.

Izquierdo and Bührmann [20] evolved a single controller to perform two qualitatively different behaviors when placed in different simulated bodies. In a one-legged body the controller performed legged locomotion, and in a wheeled body the controller performed chemotaxis. Further analysis showed that the behaviors were a product of coupled dynamics between controller, body, and environment. While this work was not intended to address the reality gap problem, it demonstrates that evolution can discover controllers that are robust to different body plans because behavior emerges from controller-body-environment interactions. This suggests that controllers may be able to cross the reality gap even if this involves significant changes in the task environment, as long as the dynamical interactions between the controller, body, and environment also change in a way that produces effective behavior.

2.2 Scaffolding

Doncieux and Mouret [12] reviewed evolutionary robotics strategies that focus on selective pressures during evolution, including efforts to address the reality gap problem. They distinguished between "goal refiners," which change the final solution of evolution, and "process helpers," which try to improve search efficiency. These categories were further divided into task-specific approaches, which exploit qualities of a specific goal task, and task-agnostic approaches, which can be applied to any evolutionary robotics problem. This work presented scaffolding as a type of incremental evolution, where a target task is split into subtasks to be evolved in order, thereby applying selective pressure to avoid premature convergence.

Adopting the terminology of Doncieux and Mouret [12], scaffolding is a "process helper" that provides changing structure to an agent to help it learn how to perform a task. In general, scaffolding facilitates learning by introducing gradation into a learning process [36] or, in the context of evolutionary robotics, an evolutionary process. In our physical scaffolding method, the lo-fi simulator acts as a scaffold to help controllers gradually acquire behavior to be performed in the hi-fi simulator.

Other kinds of scaffolds are also employed in evolutionary robotics. Approaches include making the desired task easier, adding terms to the fitness function, and simplifying the robot's morphology.

Singh [38] trained a robot on a succession of tasks, each of which required some subset of the previously learned tasks plus a new task. This series of elemental tasks in turn made up a composite task. Singh [38] used the term "shaping" to describe this technique, in reference to similar procedures used by psychologists to train animals. The elemental tasks were Markovian decision tasks (MDTs), where an agent specifies a control policy for a finite-state, discrete-time, stochastic dynamical system to maximize a cumulative payoff. Composite tasks were ordered sequences of MDTs. A novel learning technique, the CQ-learning architecture, used elemental tasks and subsequences of tasks to solve composite tasks, and could also learn elemental tasks on which it had not been trained.

Saksida et al. [36] used "behavior editing" to train a robot to perform multiple complex tasks, all derived from a single "innate" task. Human trainers used instrumental conditioning to supervise the learning process, using rewards to shape the robot's behavior. Robot behaviors consisted of chains of abstract states connected to each other by transition links, and possibly associated with concrete actions (e.g., raise arm) by activation links. This model produced different aspects of shaping by using several learning mechanisms: modifying state transitions; modifying preconditions that determine how stimuli cause behaviors; fading, where a target is slowly moved farther away; stimulus discrimination and generalization; and shaping action topography, where rewards are used to adjust individual actions. Behaviors learned included "follow the trainer," discriminating toys based on color, and playing fetch.

Dorigo and Colombetti [14] used "shaping" gradually to teach a robot to perform complex behavior patterns composed of simpler behaviors. Learning was implemented by an evolutionary algorithm which generated networks of classifier systems to control the robot's behavior. In holistic shaping, the whole network of classifier systems was trained together. In modular shaping, different components were trained separately. Unlike classical reinforcement learning, this shaping technique involved an external trainer, implemented as a reinforcement program, which made suggestions to be translated into effective classifier systems through learning. Similarly, investigators in the field of developmental robotics often introduce learning gradients such that their robots gradually acquire complex skills [9].

In the work of Berthouze and Lungarella [3], a humanoid robot learned a swinging task by exploring oscillator parameters to control joints. The discovered behavior was robust to the addition of nonlinear perturbations only if the number of degrees of freedom subject to exploration was varied: Alternate freezing and freeing of the degrees of freedom was enforced during the learning process. A progressive release of degrees of freedom, without alternating freezing and freedom, was not sufficient to cope with the perturbations.

Bongard [6] found that morphological change can facilitate the evolution of legged phototaxis behavior. Early in evolution, robots grew from anguilliform into legged robots during their lifetime. Over evolutionary time, the anguilliform stage was gradually lost, so that during the later stages of evolution the robots did not grow during their lifetime. Successful gaits for legged robots were found more rapidly, and the gaits were more robust to perturbing noise, than in evolving populations of legged robots that did not go through the anguilliform stage.

2.3 Minimal Cognition

Minimal cognition tasks are as simple as possible while still involving cognitively interesting issues. For evolutionary robotics, minimally cognitive behaviors may compose necessary qualities of agents able to interact intelligently with a cognitively challenging world.

Beer [2] evolved dynamical neural networks to perform simplified versions of visually guided orientation, object discrimination, and accurate pointing toward objects. Slocum et al. [39] extended this work to a wider range of more complicated minimally cognitive tasks. These included the perception of affordances, self/non-self-discrimination, short-term memory, and selective attention.

Tuci et al. [40] evolved a controller for a minimal, two-dimensional model of a Khepera robot to exhibit learning behavior. The task environment required the robot to move to a goal location that could only be perceived once the robot reached it. A landmark light source could always be perceived. In landmark-near trials the light source was on the same side of the environment as the goal; in landmark-far trials it was on the opposite side. All trials within a set of trials were either landmark-near or landmark-far: The robot had to learn which condition pertained to a set of trials in order to perform well. Previous efforts to evolve solutions to this task had been unsuccessful [41]. Tuci et al. [40] accomplished it by adding a scaling factor to the fitness function to increase the fitness score in the landmark-near environment. This adjustment applied evolutionary pressure for solutions to *pay attention* to the light source, a prerequisite to learning to use the light source to distinguish landmark-near from landmark-far environments.

Buhrmann et al. [8] used a minimal model of active categorical perception to illustrate four proposed "sensorimotor contingencies," relations pertaining to agent-environment interaction and behavior. The first describes an open-loop relationship between sensory input changes and motor activity. Another closes the loop by including changes internal to the agent. The third relation applies the coordination of sensorimotor patterns to tasks. The final relation organizes behaviors by normatively comparing such task-oriented patterns. In the active categorical perception task, the agent moved along a one-dimensional environment and used a single sensor to detect two bell-shaped gradients. The agent had to employ an active sensorimotor strategy to discriminate between wide and narrow shapes.

Iizuka et al. [19] used a minimal simulation to extend a previous [30] homeostatic adaptation model. A Khepera-like agent facing a phototaxis task was equipped with photosensors and an artificial neural network controller with adjustable weights. When low sensor activity caused controller neural activity to fall out of homeostatic range, the weights were varied until homeostasis was restored. The resulting phototaxis behavior was robust even to swapping the positions of the photosensors on the agent during evaluation.

3 Methods

In this section, we describe our physical scaffolding method, evolutionary algorithm, task environments, and robot controller. Physical scaffolding is a novel means to address the reality gap. It helps agents learn a desired behavior by gradually changing the structure of the task environment in which the behavior is performed. Our experimental setup tests the effectiveness of physical scaffolding using a two-step simulation pipeline. In each simulation, the agent must perform the same minimal cognition task: selective attention. Solving selective attention requires an agent to determine which of two moving targets will reach it first, track and intercept that target, and then intercept the other target. The first (lo-fi) simulation presents selective attention abstractly, while the second (hi-fi) simulation presents the same task more concretely, in a more physically realistic setting. We tested the pipeline by comparing experimental and control conditions. In the experimental condition, the evolutionary algorithm implemented simulation scaffolding by automatically transitioning the evaluation of robot controllers between less realistic and more realistic task environments over the course of evolution. In the control condition, controllers were only evaluated in the hi-fi simulation: The lo-fi simulation was not used.

3.1 Physical Scaffolding

Physical scaffolding uses a simpler simulation to improve the evolutionary search for desired behavior in a more realistic simulation, or in reality, by automating the transition over evolutionary time from less realistic to more realistic embodiment. In evolutionary robotics, a genome is a data structure containing genes that parameterize the controller and/or morphology of a specific robot. Evolution can explore the space of possible robots by modifying genomes [29]. In our experiments, robot controllers in both the hi-fi and the lo-fi simulators were specified by evolved genomes.

We discuss our evolutionary algorithm in more detail in Section 3.4. Our physical scaffolding scheme augments the genomes that specify agent controllers with two sets of additional information. First, genes were added to the genome specifying parameters for the hi-fi simulation. These permitted evolution to tune the hi-fi simulation to make it easier for the lo-fi simulation to scaffold it. Second, a simulation selection vector $\vec{v}$ was evolved alongside the genome. When the evolutionary algorithm evaluated the genome, $\vec{v}$ determined how much of the evaluation took place in each simulator.

Evaluation involved up to $\mathcal{T}$ trials, where $\mathcal{T}$ depended on the experiment and was one of {10, 20, 25, 30}. Which of the $\mathcal{T}$ trials were actually used to evaluate an individual agent *and* in which simulation it was evaluated were determined by $\vec{v}$, which was composed of ternary values (0, 1, or 2) and had length $\mathcal{T}$. An individual was evaluated on each trial according to the values of the corresponding elements of $\vec{v}$ as follows:

$0 \rightarrow$ skip trial
$1 \rightarrow$ evaluate trial in the lo-fi simulator
$2 \rightarrow$ evaluate trial in the hi-fi simulator

For example, an agent with $\vec{v} = (0012000011)$ would be evaluated in the lo-fi simulator on the third, ninth, and tenth trials and in the hi-fi simulator on the fourth trial. The availability of the "skip trial" option allowed more gradual increasing of task difficulty than would be possible with only two states: Early on, an agent could achieve high fitness without accomplishing all of the trials that it would eventually face.

An agent's overall performance was determined by averaging the error from trials that were not skipped (i.e., corresponding to a $\vec{v}$ value of 1 or 2). A multi-objective evolutionary algorithm applied pressure simultaneously toward increased evaluation in the hi-fi simulation and toward desirable agent behavior. Thus, the investigator did not have to determine how to transition controllers to the hi-fi simulation.

3.2 Task Environments

Selective attention, described by Slocum et al. [39] as the most difficult minimal cognition problem they considered, is the target behavior for our experiments. A horizontally moving agent must intercept two objects moving at different horizontal and vertical velocities (Figure 2). The agent must avoid being distracted by one object while orienting to the other. The task is complicated by two additional problems. The *passing objects* problem occurs when an object that is farther away reaches the agent first because it is moving faster. The *object permanence* problem occurs when one object moves outside the agent's field of view while the agent is intercepting the other object. In our

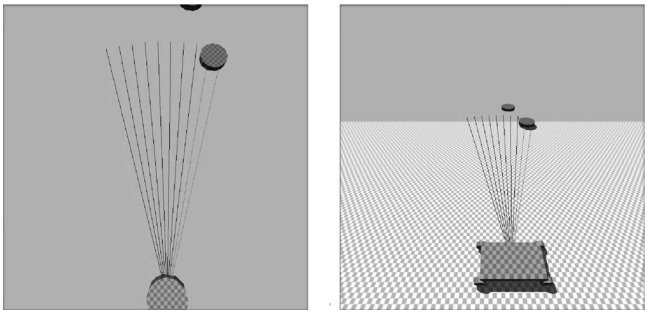

Figure 2. (a) The lo-fi simulation. (b) The hi-fi simulation.

simulation pipeline, agents evolved to perform this selective attention task in the lo-fi simulator gradually evolved the ability to retain this behavior when embodied in the hi-fi simulator.

Both the lo-fi and hi-fi simulations were implemented using the open-source Bullet Physics Library. While the robot's movement range was unbounded, a box containing the extreme possible positions of the targets, derived from Equation 3 and Table 1, was 289.5μ wide and 371.875μ tall, where μ refers to an arbitrary unit of distance. At the outset of each evaluation, the robot was placed at the origin of the environment. The robot in the lo-fi simulation (Figure 2a) was modeled kinematically and had a diameter of 15μ. The robot moved linearly along the x axis with velocity proportional to output from the controller with a constant of proportionality of 5. The hi-fi simulation used the full features of the physics engine, including friction, momentum, and collision. The robot in this simulation (Figure 2b) consisted of a rectangular solid with dimensions length = 40μ, width = 20μ, and height = 5μ, and four cylindrical wheels with width = 3μ and variable radius. The wheels were connected to the bottom edges of the base by rotating joints. Agent movement was constrained to two dimensions so that it could translate along the x and z axes and rotate around the x and y axes, but could not translate along the y axis or rotate around the z axis. This was necessary because, even though target wheel velocities were always equal, nuances in the physics engine could otherwise cause the agent to turn and leave the $y = 0$ plane.

Ten proximity sensors were projected from the agent's center in both the lo-fi and hi-fi simulators. The angles of the sensors were evenly distributed across an arc of $\pi/6$ rad. Each sensor had a range α of 205μ. If it did not intersect with a target within that distance, then its length λ was equal to α. If the sensor did intersect with a target, then λ was set to the distance from the robot to the target. The sensor returned a value according to

$$I = 10(1 - (\lambda/\alpha)) \tag{1}$$

These sensor values were input as I_i to the 10 input neurons in the CTRNN, as shown in Equation 5. If both targets were in the path of a sensor, the closer of the two would obscure the more distant one.

Two targets, *a* and *b*, which were modeled kinematically, were positioned shifted along the x and y axes relative to the robot. Each time step, both targets were displaced constant distances along both the x and y axes. Each target was defined by

$$\langle x_0, \delta_x, \delta_y, \omega \rangle \tag{2}$$

where x_0 denotes the object's unscaled initial x position, δ_x denotes the change along the x axis per time step (horizontal speed) and could be either positive or negative, δ_y denotes change along the y axis per time step (vertical speed) and was always negative, and ω denotes a scaling factor. The actual starting position of a target was

$$x = x_0 \cdot (\omega + 1)\mu, \quad y = \beta \cdot (\omega + 1)\mu, \quad z = 0\mu \tag{3}$$

where β was set to 212.5. This value was selected to place targets having $\omega = 0$ at the extreme limit of all of the robot's sensors. Since each sensor had a range of 205μ, the leftmost and rightmost sensors were angled at $\pi/12$ rad to the y axis, and the targets had a radius of 13μ, an agent with $\omega = 0$ starting at the x position of the leftmost or rightmost sensor would barely overlap that sensor.

The targets began at positions with positive values on the y axis. As the targets' y positions changed with constant speed, they moved toward the line along which the agent could move, that is, $y = 0$. The targets' x positions also changed with constant speed. A controller's performance depended on minimizing the distance between the agent and each target as the targets intersected the line on which the agent moved (Equation 4).

Table 1. Target starting position and movement parameters.

	Target *a*				Target *b*			
Trial	x_0	δ_x	δ_y	ω	x_0	δ_x	δ_y	ω
1	0	0.5	1	0	40	−0.5	2.5	0
2	−55	1.0	1	0	45	−0.7	2.5	0
3	55	0	1	0	−45	0	2.5	0
4	0	0	1	0	−45	0.5	2.5	0
5	−25	−0.5	1	0	45	−0.25	2.5	0
6	55	0	1	0	45	−0.7	2.5	0.75
7	−75	0.9	1	0	−45	0.25	2.5	0.75
8	0	−0.35	1	0	−10	0.35	2.5	0.75
9	25	0.5	1	0	45	−0.25	2.5	0.75
10	−55	0	1	0	−45	0.7	2.5	0.75
11	0	−0.5	1	0	0	0.5	2	0
12	−55	0.7	1	0	65	−1.0	2	0
13	−55	0	1	0	0	0	2	0
14	0	0.5	1	0	−55	0	2	0
15	−55	−0.25	1	0	55	−0.5	2	0
16	55	−0.7	1	0	−25	0	2	0.75
17	−65	0	1	0	−65	0.9	2	0.75
18	0	0.35	1	0	0	−0.35	2	0.75
19	−55	−0.25	1	0	−55	0.5	2	0.75
20	−75	0.65	1	0	−55	0	2	0.75
21	0	0.5	1	0	0	−0.5	2	0
22	−55	1.0	1	0	65	−0.7	2	0
23	55	0	1	0	−55	0	2	0
24	0	0	1	0	−55	0.5	2	0
25	−25	−0.5	1	0	25	−0.25	2	0

Table 1. (*continued*)

	Target *a*				Target *b*			
Trial	x_0	δ_x	δ_y	ω	x_0	δ_x	δ_y	ω
26	55	0	1	0	25	−0.7	2	0.75
27	−75	0.9	1	0	−35	0.25	2	0.75
28	0	−0.35	1	0	0	0.35	2	0.75
29	25	0.5	1	0	25	−0.25	2	0.75
30	−55	0	1	0	−35	0.7	2	0.75

Notes. Each trial contained two targets. Each target was defined by x_0 (the initial x position), δ_x (change along the x axis each time step), δ_y (change along the y axis each time step), and ω (y-axis offset).

The error e on a specific trial t was determined according to

$$e_t = \left|x_a\left(f_a\right) - x_r\left(f_a\right)\right| + \left|x_b\left(f_b\right) - x_r\left(f_b\right)\right| \tag{4}$$

where a and b denote the two targets, r is the agent, f_i is the first time step when the y coordinate of i was less than or equal to 0, and $x_i(t)$ is the x coordinate of i at time t.

Three aspects of this hi-fi simulation varied in the experiments reported below: the amount of friction between the wheels and the ground (ϕ), the wheel radius (ρ), and the motor gain (γ). These parameters applied only to the hi-fi simulation. The motor gain was multiplied by the controller output (Equation 6) to determine the target angular velocity in radians per second, which was applied equally to all four joints. In the experiments reported below, each of the morphological parameters was either evolved or fixed. When parameters were evolved, their values were mapped linearly from genes in the unit interval [0, 1], a range selected following previous work [20], to the following ranges: $\phi \in [0, 100]$, $\rho \in [0, 10]$, and $\gamma \in [0, 10]$. The absolute value of the genes was used so that, even though these parameters could evolve beyond the specified ranges, they could not become negative. In experiments where these morphological parameters were not evolved, they were fixed at the following values: $\phi = 162.972$, $\rho = 2.268$, $\gamma = 21.180$. These values were determined by a separate evolutionary run where a controller evolved using the lo-fi simulation was transferred to the hi-fi simulation. Genes specifying the controller were then held constant while the three morphological parameters were evolved in the hi-fi simulation until the error approached that in the lo-fi simulation. This required only a few minutes of evolution in the hi-fi simulator.

Figure 3 plots the behavior of a single evolved controller in each of the two simulation environments. Behavior in the two environments is markedly different: The agent in the lo-fi simulation changes direction frequently and rapidly, while the hi-fi agent changes direction more gradually. The hi-fi agent is more constrained by the laws of physics than the lo-fi agent: It has mass and momentum, and its wheels can slip or lift off the ground entirely. Nevertheless, the controller is able to intercept the targets in both environments.

3.3 Controller

Agents were controlled by a continuous-time recurrent neural network (CTRNN) [1] defined by the following state equation:

$$\tau_i \dot{y}_i = -y_i + \sum_{j=1}^{N} w_{ji}\sigma\left(g_j\left(y_j + \theta_j\right)\right) + I_i, \quad i = 1, \ldots, N \tag{5}$$

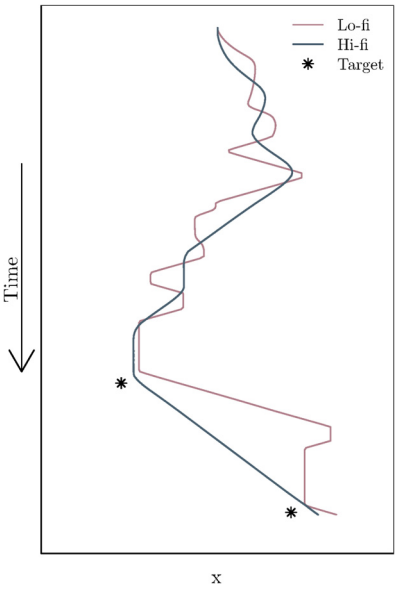

Figure 3. Comparison of behaviors for a single controller, for a single trial, placed in the lo-fi and hi-fi simulators. Stars mark the time and x position of targets when they reach the agent at $y = 0$.

where y_i is the value of the ith neuron; $\sigma(x) = (1 + e^{-x})^{-1}$ is a sigmoidal function which, when supplied with the value of y_i, produces the neuron's output; τ_i is its time constant; w_{ji} is the strength of the connection from the jth to the ith neuron; g_i is that neuron's gain; θ_i is its bias; and I_i is an external input from a sensor. For the hidden and output neurons, I was fixed at 0. For each of the input neurons, I was associated with a sensor.

There were 10 input neurons, 10 hidden neurons, and two output neurons. Input neurons projected to both the hidden neurons and the output neurons, the hidden neurons were fully interconnected and projected to the output neurons, and the output neurons were fully interconnected. Neuron states were initialized to 0. The CTRNN was integrated using the forward Euler method with an integration step size of 0.333.

The CTRNN was parameterized by $\vec{c}$, a vector of real numbers generated by the evolutionary algorithm. Elements of $\vec{c}$ (genes) in the range [0, 1] were mapped linearly into CTRNN parameters with the following ranges: connection weights $\in [-5, 5]$, input neuron biases $\in [-10, 0]$, hidden and output neuron biases $\in [-5, 5]$, gains $\in [1, 5]$, and time constants $\in [1, 2]$. All input neurons shared a single gain and a single bias. The gains of the output neurons were fixed at 1. Because genes could assume values outside [0, 1] during the course of evolution, the CTRNN parameters could also move outside these initial ranges. Nevertheless, gains were always clipped to be greater than zero, and time constants were clipped to be greater than one. The CTRNN was bilaterally symmetric: On each side of the axis of symmetry were five input neurons, five hidden neurons, and one output neuron. Each neuron and each synapse had a mirrored counterpart with equal neuron parameters and connection weights. This symmetry halved the number of genes encoding the CTRNN.[1] The output of the CTRNN was calculated as

$$\text{output} = \sigma(g_a(y_a + \theta_a)) - \sigma(g_b(y_b + \theta_b)) \tag{6}$$

where a and b index the two output neurons.

1 This CTRNN architecture was taken from Slocum et al. [39] with the following minor changes: We used 10 input neurons and sensors instead of 9. We used an integration step size of 0.333 instead of 0.1. Our initial gene range was [0, 1] instead of [−1, 1]. We mapped biases to [−5, 5] instead of [−10, 0].

3.4 Evolutionary Algorithm

An evolutionary algorithm is an iterative, metaheuristic algorithm. It uses mechanisms inspired by biological evolution to explore a parameter space in search of effective solutions to some problem. A variety of possible solutions are evaluated, and those that perform well are selected and modified to produce new solutions, which are in turn evaluated and reproduced in an iterative process. Evolutionary algorithms are not guaranteed to converge on an optimal solution. However, by combining variation with performance-based selection, they can find effective solutions to many problems. Evolutionary algorithms are particularly useful for exploring very large search spaces like robot controller parameterization.

We used *age-fitness Pareto optimization* (AFPO) [37], a multi-objective evolutionary algorithm designed to prevent early convergence by regularly introducing new genetic material into the population. Each generation, AFPO creates a new individual with age 1 and adds it to the population while incrementing the age of all other solutions. In order to maintain new genetic material in the population, the selection and reproduction process balances the objectives of low error and low age. We augmented AFPO by adding a third objective: maximize physicality, where physicality is defined as sum($\vec{v}$), the sum of the simulation selection vector. The evolutionary algorithm used age (i.e., the number of generations a genome has been subject to evolution since being randomly generated), along with error and physicality, to select controllers for reproduction. Selection was based on identifying non-dominated solutions: A solution dominated another if it had lower age, lower error, and higher sum($\vec{v}$), and a solution was non-dominated only if it was not dominated by any other solution in the population. That is, a solution was selected for reproduction only if no other solution surpassed it in all three objectives.

An evolutionary run was initialized by creating an empty population of genomes (individuals). Then, each generation of the evolutionary cycle proceeded as follows:

1. Increment the age of all individuals in the population.
2. Generate a new individual with age one, evaluate it, and add it to the population.
3. Until the number of individuals equals the population size,
   (a) select a parent individual that was not created by replication this generation,
   (b) replicate the parent and mutate it to create a child,
   (c) evaluate the child and add it to the population.
4. Remove all dominated individuals from the population, leaving a non-dominated front.
5. Unless the time limit has been reached, return to the first step.

The initial population size was 400. In the event that the size of the non-dominated front reached this population limit, the limit was automatically increased by 25. This was necessary because if the non-dominated front constituted the entire population, neither the introduction of new genetic material (step 2) nor reproduction with mutation (step 3) would be possible, and population change would cease.

Each individual consisted of a three-part genome: $\vec{c}$, a vector of 142 real values that parameterized the CTRNN; $\vec{k}$, a vector of three real values that specified the morphological parameters of the hi-fi simulation; and $\vec{v}$, the simulation selection vector defined in Section 3.2. When a new individual was generated, elements of $\vec{c}$ and $\vec{k}$ were drawn from a uniform distribution in [0, 1]. One element of $\vec{v}$ was selected at random and set to 1, while the remaining elements were set to 0.

When an individual was reproduced, mutation was applied to each part of the genome. Each element of $\vec{c}$ was mutated with probability 0.05 by adding to it a value drawn from a Gaussian distribution with a mean of $\mu = 0$ and a standard deviation of $\sigma = 1$. Each element of $\vec{k}$ was mutated with probability 0.2 by adding to it a value drawn from a Gaussian distribution with $\mu = 0$ and $\sigma = 0.5$. Mutation was free to move elements of $\vec{c}$ and $\vec{k}$ outside their original range of

[0, 1]. Each element of $\vec{v}$ was mutated with probability 0.1 by adding to it, with equal probability, either 1 or 2. This addition occurred modulo 2, so that elements of $\vec{v}$ were always integers in the range [0, 2].

4 Results

For each of the experiments described below, thirty independent evolutionary runs were performed each for a control condition and for an experimental condition. In the experimental condition, simulation scaffolding was employed as discussed in Section 3.2. In the control condition, simulation scaffolding was suppressed. This was accomplished by discarding the lo-fi simulator and redefining the elements of $\vec{v}$ as follows:

$0 \rightarrow$ skip trial
$1 \rightarrow$ evaluate trial in the hi-fi simulator
$2 \rightarrow$ evaluate trial in the hi-fi simulator

Figure 4 shows the behavior of a typical, successfully evolved controller. The controller solves the selective attention task in both the lo-fi simulator and the hi-fi simulator.

We measured performance with respect to a fixed computational budget measured in CPU hours of evaluation time. Evaluation time is defined as the accumulated CPU time spent running simulations to evaluate agents. This included time used by the CPU to execute simulation code and system calls made by the simulation, but excluded time while the CPU was idle or executing other code such as the evolutionary algorithm. We used evaluation time to measure performance because it is a common constraint in evolutionary robotics experiments, whether evaluation takes place in simulation or on physical robots. The actual time each experiment ran was bounded by a wall-clock time limit, and the total evaluation time varied. We truncated the results of all experiments to the minimum total evaluation time so that the amount of evaluation time considered for each experiment was 13 h.

First, we conducted *vary-trials* experiments where all three morphological parameters were evolved and the number of trials, $\mathcal{T}$, was varied in {10, 20, 25, 30}, that is, there were four vary-trials experiments. The purpose of these experiments was to determine the number of trials needed for physical scaffolding to improve evolution. Increasing the number of trials made the learning task more difficult, because solutions were evaluated on a greater number of tasks. Thus, increasing the number of trials to 30 increased the error generally, but also increased the advantage of physical scaffolding over the control condition. Given that some trials could be more or less difficult than others, and in order to ensure that each set of trials was strictly more difficult than any smaller set, each set included all smaller sets. For experiments where $\mathcal{T} = 10$, only the first ten trials were used; for

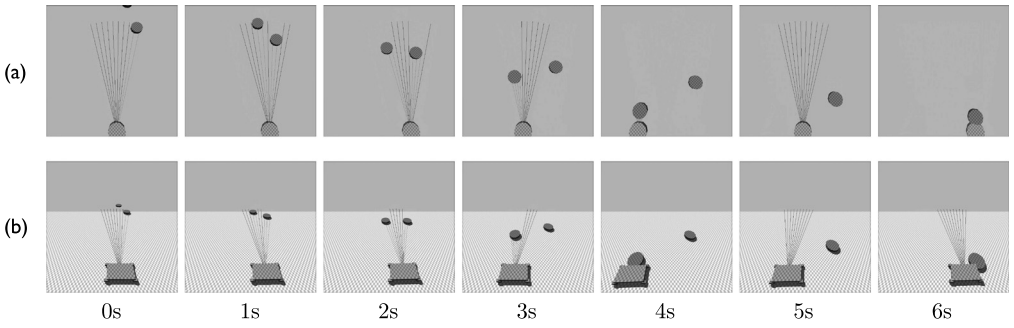

Figure 4. Time series showing a single evaluation of a typical successful, evolved controller performing the selective attention task. (a) The minimally embodied agent in the lo-fi simulator. (b) The more embodied agent in the hi-fi simulator. This example demonstrates both the passing object problem and the object permanence problem.

experiments where $\mathcal{T} = 20$, the first 20 trials were used; and for experiments where $\mathcal{T} = 25$, the first 25 trials were used (Table 1).

Second, we conducted *vary-parameters* experiments where $\mathcal{T}$ was held at 30 and the subset of the morphological parameters subject to evolution varied. We will use the notation M = {F,R,G} to indicate all eight possible combinations of evolved and constant morphological parameters, where F, R, and G are binary values indicating whether the friction, wheel radius, and gain parameters, respectively, were evolved. For example, M = {1,0,1} indicates the case where the friction and gain were evolved, but the wheel radius was held constant. We ran experiments for each possible combination of evolved and fixed parameters, without redoing the case where all parameters were evolved, that is, M = {1,1,1}. Thus, we conducted $2^3 - 1 = 7$ vary-parameter experiments.

During an evolutionary run, the best error was recorded at the end of each hour of evaluation time. The best error was determined to be the lowest error among the individuals whose $\vec{v}$ contained all 2's. If no such individual existed, the one with the lowest error among those with the largest sum ($\vec{v}$) was reevaluated with all elements in its $\vec{v}$ set to 2 to determine the best error.

The vary-trials performance showed that physical scaffolding improved performance when 30 evaluation trials were used, while the vary-parameters performance showed that physical scaffolding improved performance for M = {1,0,0} and M = {1,1,1}. These observations are demonstrated in Figures 5 and 6. Figure 5 shows the best error averaged across 30 runs for each condition of each vary-trials experiment. Figure 6 shows the best error averaged across 30 runs for each condition of each vary-parameters experiment. Error bars show 95% confidence intervals. Asterisks indicate *p*-values for a *t*-test with the alternative hypothesis that the best error was lower in the experimental condition than in the control condition. A single asterisk indicates a *p*-value below 0.05, double asterisks indicate a *p*-value below 0.01, and triple asterisks indicate a *p*-value below 0.001.

The assumption of independence implied by these methods might be unjustified, since the performance of a run at some point in the evolutionary process could be correlated with earlier or later performance. However, given that the samples were taken an hour apart (typically, approximately 70 generations) while the best agents were usually about half an hour old (typically, 30–40 generations), we present the results of hourly comparisons. Nevertheless, we will draw our main conclusions from analysis of performance at the final, 13-h time point. At the final time point, physical scaffolding produced lower error when the control method $\mathcal{T}$ was 30 and M was either {1,1,1} or {1,0,0}.

Evolution tended to select for greater friction, smaller wheels, and less motor gain when scaffolding was used, suggesting that such values increase transferability. This is demonstrated by the final evolved values of the genes corresponding to ϕ, ρ, and γ presented in Table 2. The *p*-values are from *t*-tests with the alternative hypothesis that the difference between the control mean and the experimental mean is nonzero. Low *p*-values indicate a statistically significant difference between the values of the experimental and control conditions.

Individuals gradually transitioned to an increasing number of hi-fi simulations as they evolved. However, the order in which hi-fi trials were added varied between individuals. Furthermore, for any given trial the transition from being skipped, to lo-fi simulation, to hi-fi simulation was not monotonic. This is demonstrated by Figures 7 and 8, which show the evolution of the simulation selection vector $\vec{v}$ over the lifetime of the most successful individual from nine randomly selected control and experimental runs, respectively, with $\mathcal{T} = 30$ and M = {1,1,1}. Columns show the state of $\vec{v}$ from when the individual is first produced (leftmost column) to the end of the evolutionary run (rightmost column): Black indicates a value of 0 (skip trial), dark gray indicates 1 (lo-fi for experimental, hi-fi for control), and light gray indicates 2 (hi-fi).

5 Discussion

Our experiments tested the simulation-to-reality pipeline by performing one set of experiments varying the number of trials used to evaluate controllers and a second set of experiments varying the set

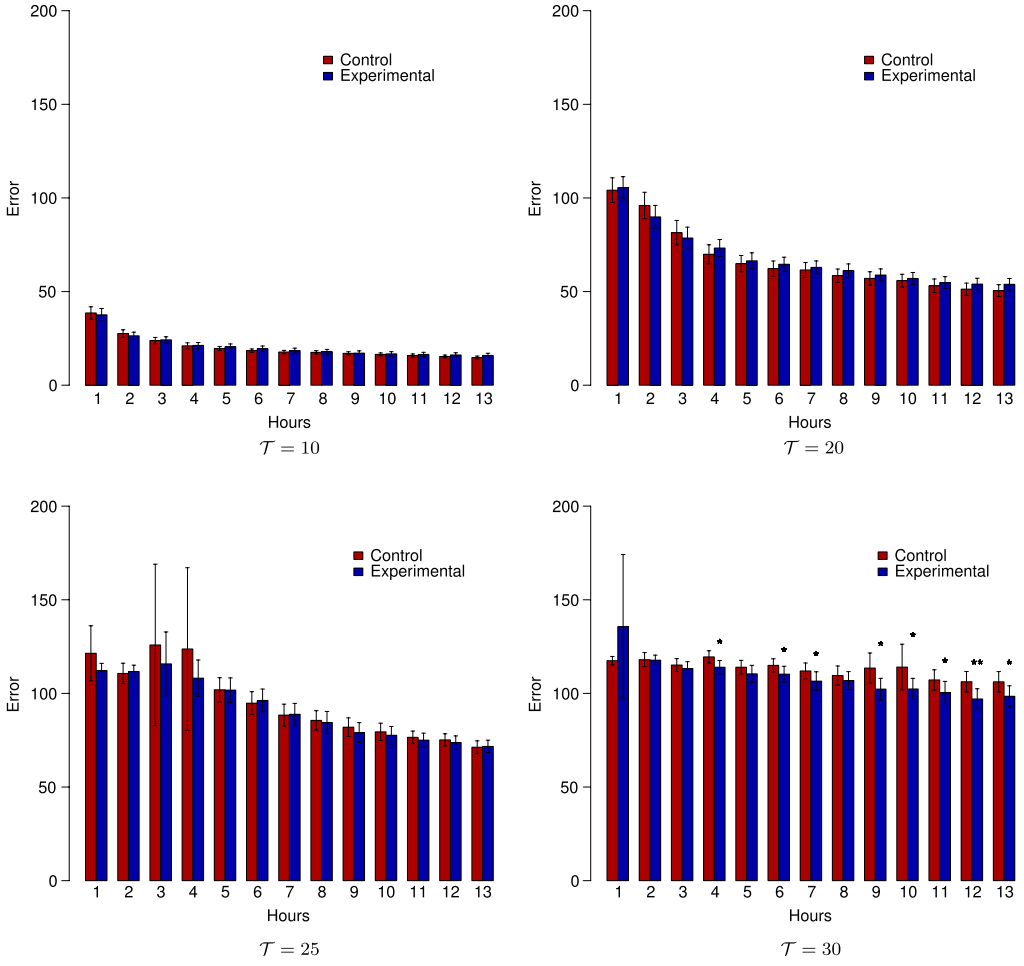

Figure 5. Vary-trials results. Error for control (no physical scaffolding) and experimental (with physical scaffolding) conditions with 10, 20, 25, and 30 evaluation trials. All three morphological parameters were subject to evolution. Asterisks indicate p-values below 0.05, and double asterisks indicate p-values below 0.01, for the alternative hypothesis that the experimental condition had lower error than the control condition. Results were averaged over 30 experimental repetitions. Error bars indicate 95% confidence intervals.

of morphological parameters subject to evolution. We compared the errors of the best evolved solutions from an experimental condition that employed our physical scaffolding technique against a control condition that did not use physical scaffolding. Results show that physical scaffolding decreased error for some of the experimental treatments. Among the experiments varying the number of evaluation trials, physical scaffolding outperformed the control condition when 30 trials were used. When the evolved set of morphological parameters varied, physical scaffolding helped when ground friction alone was evolved or when ground friction, wheel size, and motor gain were all evolved.

Overall, in the vary-trials experiments' performance decreased as the number of trials increased. This indicates that evaluation on more trials is in some sense more difficult than on fewer trials. In experiments with $\mathcal{T} = 10$, $\mathcal{T} = 20$, and $\mathcal{T} = 25$, error decreases as the amount of computation time increases, but there is no statistically significant ($p < 0.05$) difference between the control and experimental conditions. Evolution works but it is not improved by the physical scaffolding regime.

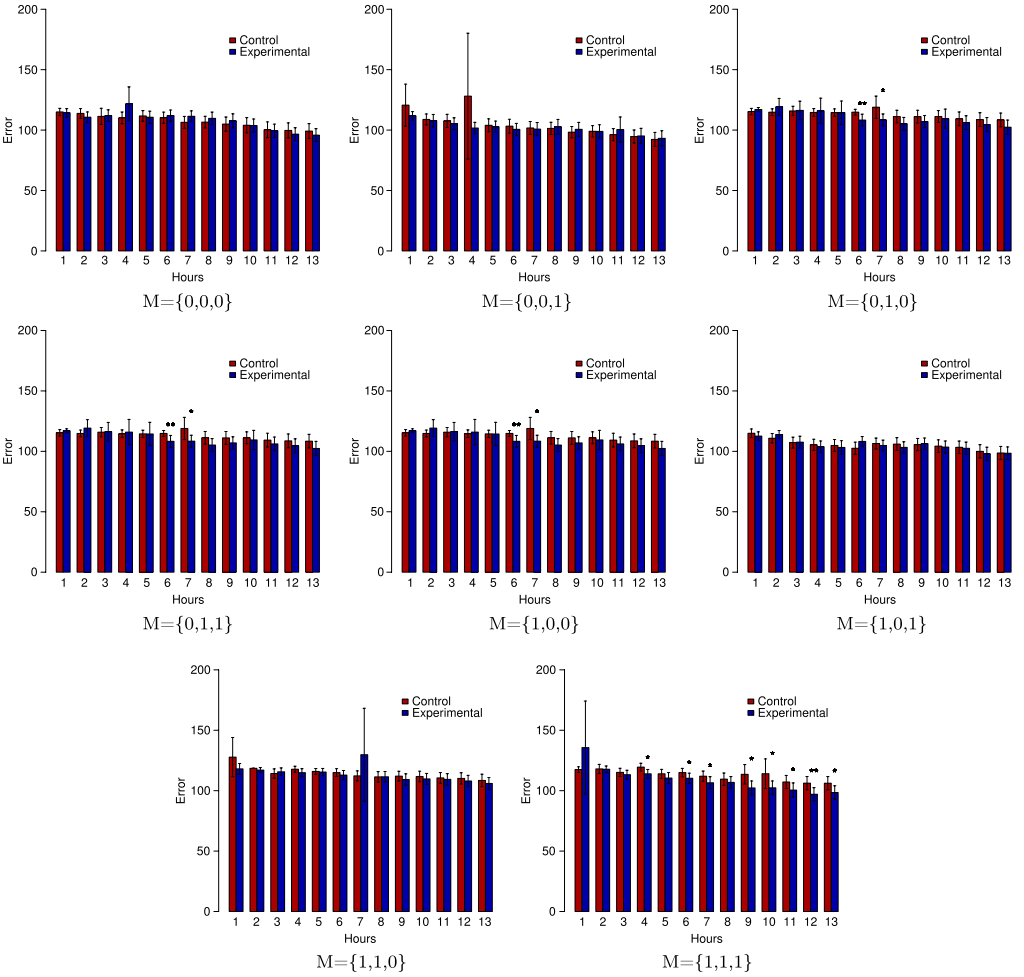

Figure 6. Vary-parameters results. Error with and without physical scaffolding. Results are shown with each subset of the three morphological parameters subject to evolution. Single asterisks indicate *p*-values below 0.05, double asterisks indicate *p*-values below 0.01, and triple asterisks indicate *p*-values below 0.001 for the alternative hypothesis that the experimental condition had lower error than the control condition. Results were averaged over 30 experimental repetitions. Error bars indicate 95% confidence intervals.

A significant difference in performance does emerge, however, when $\mathcal{T}$ is increased to 30. At the final measured time point (13 h), the error of agents evolved with physical scaffolding is lower than that of agents without ($p < 0.05$), given an equivalent computational budget. This difference in performance begins to emerge after about 8 h of evolution. Thus, physical scaffolding improves evolved

Table 2. Mean and standard deviation of the hi-fi morphological parameters evolved with $\mathcal{T} = 30$ and M = {1,1,1}.

	Control mean	Experimental mean	Control st. dev.	Experimental st. dev.	*p*-value
Friction	0.7305	1.0972	0.9999	1.6269	0.03063
Radius	0.4120	0.0130	1.0644	0.7739	0.04465
Gain	0.3148	0.0690	0.7611	1.2800	0.37861

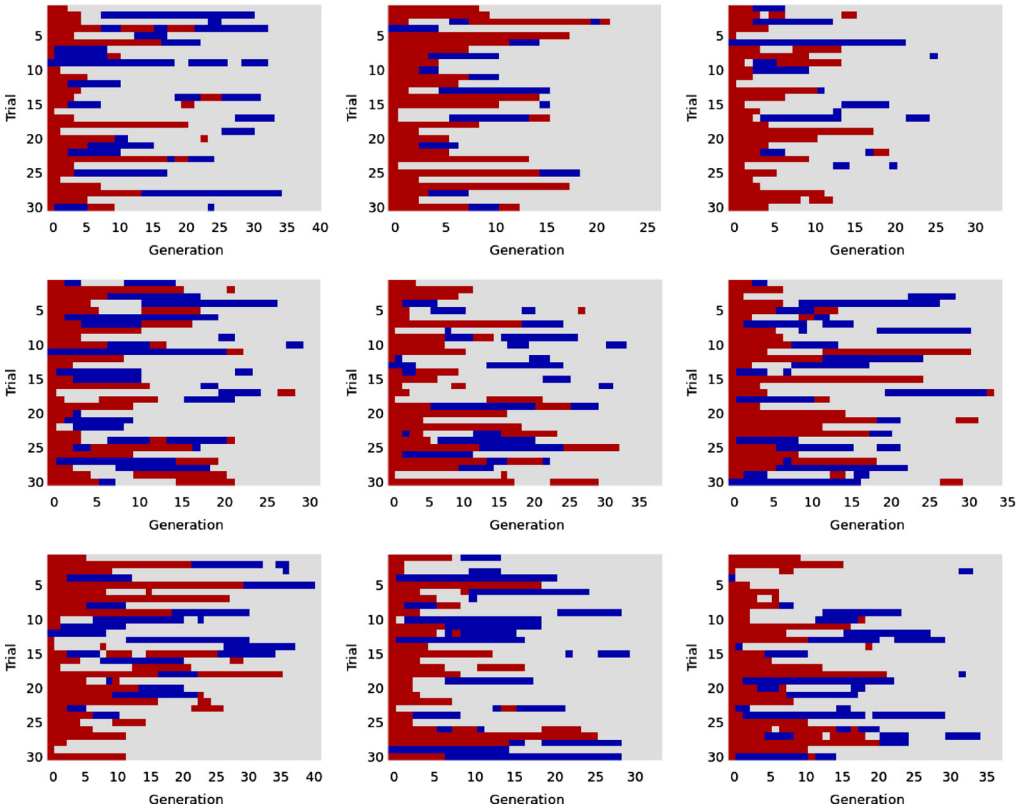

Figure 7. Evolution of $\vec{v}$ for the best evolved individual from nine control condition runs. $\mathcal{T} = 30$, M = {1,1,1}. Columns show the state of $\vec{v}$ from when the individual is first produced (leftmost column) to the end of the evolutionary run (rightmost column): Red indicates a value of 0 (skip trial), blue indicates 1 (lo-fi for experimental, hi-fi for control), and gray indicates 2 (hi-fi).

performance, but only given a sufficiently difficult task and sufficient time. This suggests a relationship between the complexity of the target behavior and the usefulness of our scaffolding technique. If the task is too simple, evolution can find solutions with low error equally well with or without scaffolding. As the task becomes more complex, the value of the scaffold increases. Moreover, it should be noted that the evolutionary improvement is specifically due to simulation scaffolding. Environmental scaffolding, in which robots may be evaluated in only a subset of the available trials, is possible in both the control and experimental conditions.

Scaffolding only provided an advantage when ground friction was the only evolved morphological parameter (M = {1,0,0}), or when all three of the morphological parameters considered were evolved (M = {1,1,1}). These results show that the body of the robot and its physical environment significantly influence controller transferability. This phenomenon can be seen in Figure 3, where the agent in the lo-fi simulation makes abrupt changes in direction while the agent in the hi-fi simulation, because of its mass properties, exhibits less abrupt changes. Thus, evolution must find morphological parameters that enable the hi-fi agent to move in a similar manner to the lo-fi agent in order to increase the probability of transferability. When friction was evolved with either wheel radius (M = {1,1,0}) or motor gain (M = {1,0,1}), but not both, scaffolding did not provide an advantage. It is possible that, given the fixed value used for the wheel radius, it was difficult for evolution simultaneously to find effective values for friction and motor gain. Similarly, given the fixed value used for the motor gain, it may have been difficult to find effective values for friction and wheel radius.

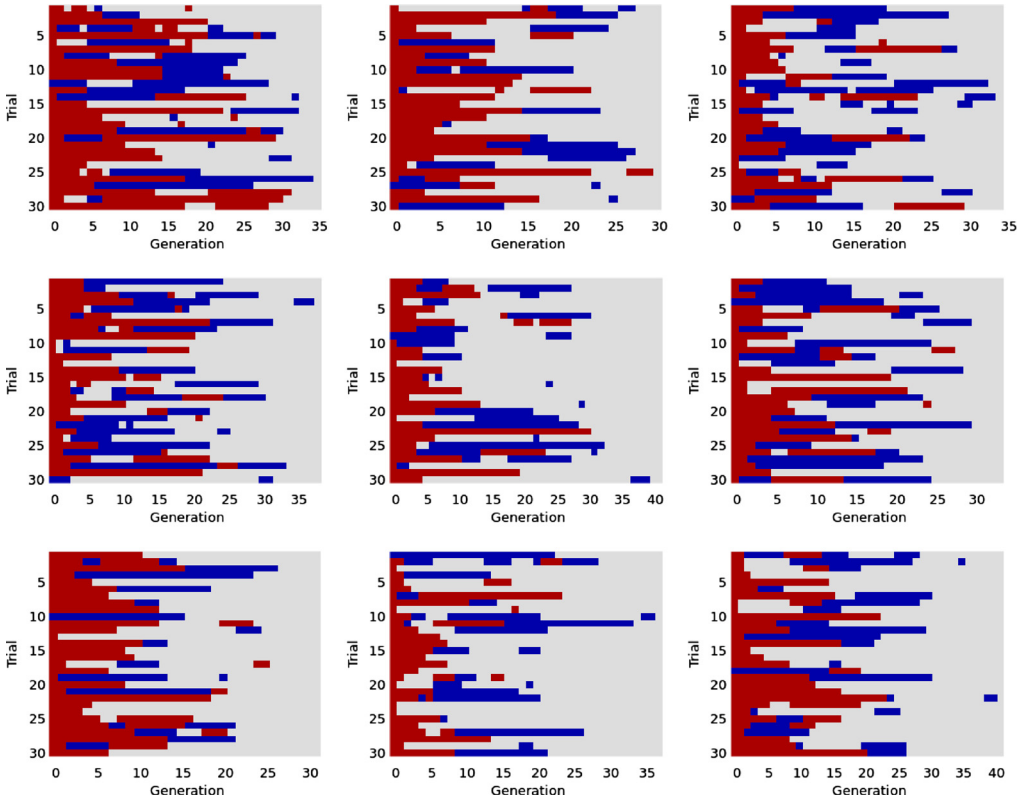

Figure 8. Evolution of $\vec{v}$ for the best evolved individual from nine experimental condition runs. $\mathcal{T} = 30$, M = {1,1,1}.

Statistics describing the evolved hi-fi simulation parameters (Table 2) suggest that evolution may have preferred greater friction, smaller wheels, and less motor gain when scaffolding was used. However, there may be changes in morphology that occurred within a lineage during evolution that are not reflected in these final morphological parameter values. For example, a hi-fi agent may evolve high motor gain to achieve rapid changes in acceleration like the lo-fi agent, but its offspring may evolve lower motor gains to enable success in a few of the remaining unsolved trials. This may result in final motor gain values that are very similar to those observed in the control runs without simulation scaffolding. Note that, because these parameters apply only to the hi-fi simulation, evaluation in the lo-fi simulation introduces noise into their evolutionary trajectories; only the hi-fi simulation exerts meaningful evolutionary pressure on these variables.

Rather than removing physical and environmental scaffolding according to a fixed schedule, we have demonstrated how the amount of scaffolding can be placed under evolutionary control and thus automated. This places evolutionary pressure on agents to solve the task in the hi-fi simulation in a way that facilitates transferability from the lo-fi simulation. Furthermore, the ability of mutation to decrease the values of elements of $\vec{v}$ allows for agents to move "backwards" in the pipeline as depicted by arrows e and g in Figure 1, bringing genetic material evolved in the hi-fi simulation back into the lo-fi simulation.

Figures 7 and 8 provide some insight into how agents moved along the simulation-to-reality pipeline. In general, $\vec{v}$ evolved from all 0's with a single 1, to all 2's, over about 30 to 40 generations. That is, the best agents gradually transitioned from experiencing a single trial to experiencing all trials in the hi-fi simulator. In the experimental condition, a period in the lo-fi simulation often intervened before the hi-fi simulation, although in some cases evolution bypassed the lo-fi simulation. This

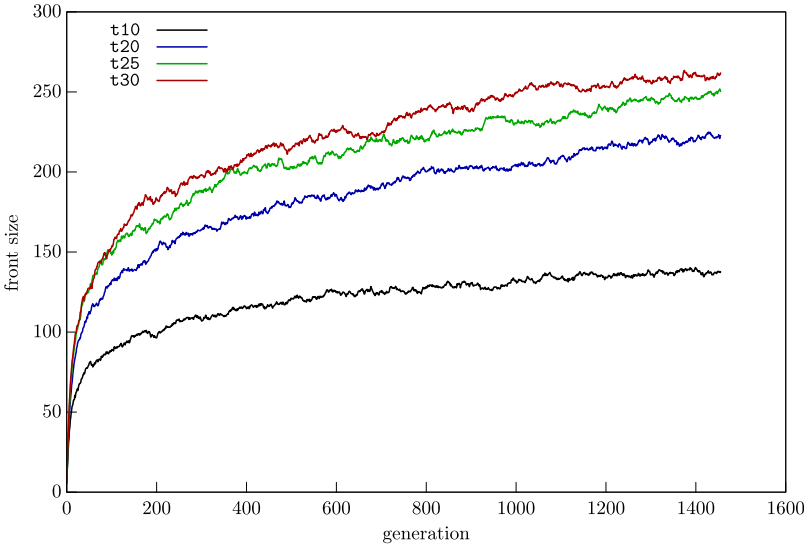

Figure 9. Size of the non-dominated front over evolutionary time. Averages across all vary-trials experiments.

progression was not monotonic. Values of $\vec{v}$ sometimes decreased from 2 to 1, or even 0, before returning to 2, corresponding to temporary backwards movement through the physical scaffolding pipeline. No clear trends emerge distinguishing the 30 trials. Patterns could be possible: for example, trial 13 seems usually to be selected for evaluation earlier than most trials, suggesting that this trial might be relatively easy. However, further analysis is needed to understand the forces governing the route taken by $\vec{v}$ during evolution.

Scaling up to tasks that require many more than 30 trials to evolve robustness as well as success raises a challenge. As the number of trials grows, so too does the number of possible values for sum $(\vec{v})$ (Figure 9). This greatly magnifies the number of non-dominated solutions that may exist in the population, thus necessitating a further increase in population size. Managing the size of the non-dominated front will thus be a priority area for future investigation.

In the proposed framework, the utility of a given morphology is its ability to facilitate the flow of controllers up the pipeline. In other work (e.g., [32, 31, 17]), morphologies have been designed or evolved to facilitate "morphological computation": the ability of the body to perform computations that would otherwise have to be explicitly performed by the controller. In future work it would be worthwhile to investigate whether morphologies evolve, within this paradigm, that support both functions.

Future work will investigate how our method scales to even more challenging tasks, to yield robustness as well as success. We will also subject more aspects of the simulations to evolutionary control by increasing the number of evolved parameters. Which simulation parameters facilitate transferability—and how those parameters differ for different tasks and robots—will also be studied. The pipeline will be expanded to admit three or more stages, or perhaps a physicality continuum that could be increased in a continuous, rather than a discrete, manner. These advances will allow a more gradual transition from the most abstract and computationally fast simulations up to extremely realistic models. The effects of the "skip trial" option in the simulation selection vector should be further explored to determine whether evolution should have the option of ignoring some trials during evaluation. Finally, the pipeline should be integrated with rapid prototyping technologies to allow for the deployment of physical robots and the collection of sensor/motor data to automatically improve the pipeline itself. These approaches promise to move us closer to an automated evolutionary process capable of crossing the reality gap.

6 Conclusion

This work demonstrates the feasibility of constructing a simulation-to-reality pipeline for evolutionary robotics, in which increasingly embodied agents evolve upward through increasingly physically realistic simulations, and the most promising agents are manufactured and deployed. We have demonstrated the feasibility of this approach by developing a technique that uses a low-physicality simulation to scaffold a high-physicality simulation, implementing a single step of the proposed pipeline. Scaffolding accelerates evolution by offloading some of the evaluation of agent behavior from the computationally demanding high-physicality simulation to the faster and more efficient low-physicality simulation.

We found that scaffolding did not help when the task behavior was relatively simple or when certain parameters of the high-physicality simulation were not under evolutionary control. However, given a fixed computational budget, a relatively complex target behavior, and the ability to tune the right simulation parameters to improve the scaffold, evolution produced better-performing simulated robots using physical scaffolding than without it.

Acknowledgments

We would like to thank the Vermont Advanced Computing Core, which is supported by NASA (NNX06AC88G) at the University of Vermont, for providing high-performance computing resources that have contributed to this work. We would like to acknowledge support from the NSF through grant PECASE-0953837.

References

1. Beer, R. D., & Gallagher, J. C. (1992). Evolving dynamical neural networks for adaptive behavior. *Adaptive Behavior*, *1*(1), 91–122.
2. Beer, R. (1996). Toward the evolution of dynamical neural networks for minimally cognitive behavior. In P. Maes, M. J. Mataric, J.-A. Meyere, J. Pollack, & S. W. Wilson (Eds.), *From animals to animats 4: Proceedings of the Fourth International Conference on Simulation of Adaptive Behavior* (pp. 421–429). Cambridge, MA: MIT Press.
3. Berthouze, L., & Lungarella, M. (2004). Motor skill acquisition under environmental perturbations: On the necessity of alternate freezing and freeing of degrees of freedom. *Adaptive Behavior*, *12*(1), 47–64.
4. Boeing, A., Hanham, S., & Bräunl, T. (2004). Evolving autonomous biped control from simulation to reality. In S. Mukhopadhyay & G. S. Gupta (Eds.), *Proceedings of the 2nd International Conference on Autonomous Robots and Agents* (pp. 440–445). Palmerston North, New Zealand: Massey University.
5. Bongard, J., Zykov, V., & Lipson, H. (2006). Resilient machines through continuous self-modeling. *Science*, *314*(5802), 1118–1121.
6. Bongard, J. (2011). Morphological change in machines accelerates the evolution of robust behavior. *Proceedings of the National Academy of Sciences of the U.S.A.*, *108*(4), 1234–1239.
7. Brodbeck, L., Hauser, S., & Iida, F. (2015). Morphological evolution of physical robots through model-free phenotype development. *PloS One*, *10*(6), e0128444.
8. Buhrmann, T., Di Paolo, E. A., & Barandiaran, X. (2013). A dynamical systems account of sensorimotor contingencies. *Frontiers in Psychology*, *4*(1), 285.
9. Cangelosi, A., Schlesinger, M., & Smith, L. B. (2015). *Developmental robotics: From babies to robots*. Cambridge, MA: MIT Press.
10. Celis, S. E., & Bongard, J. (2012). Not all physics simulators can be wrong in the same way. In T. Soule (Ed.), *Proceedings of the 14th Annual Conference Companion on Genetic and Evolutionary Computation* (pp. 659–660). New York: ACM.
11. Cully, A., Clune, J., Tarapore, D., & Mouret, J.-B. (2015). Robots that can adapt like animals. *Nature*, *521*(7553), 503–507.
12. Doncieux, S., & Mouret, J.-B. (2014). Beyond black-box optimization: A review of selective pressures for evolutionary robotics. *Evolutionary Intelligence*, *7*(2), 71–93.

13. Dorigo, M., & Colombetti, M. (1994). Robot shaping: Developing autonomous agents through learning. *Artificial Intelligence*, *71*(2), 321–370.

14. Dorigo, M., & Colombetti, M. (1998). *Robot shaping: An experiment in behavior engineering*. Cambridge, MA: MIT Press.

15. Floreano, D., & Urzelai, J. (2001). Evolution of plastic control networks. *Autonomous Robots*, *11*(3), 311–317.

16. Francesca, G., Brambilla, M., Brutschy, A., Trianni, V., & Birattari, M. (2014). Automode: A novel approach to the automatic design of control software for robot swarms. *Swarm Intelligence*, *8*(2), 89–112.

17. Hauser, H., Ijspeert, A. J., Füchslin, R. M., Pfeifer, R., & Maass, W. (2011). Towards a theoretical foundation for morphological computation with compliant bodies. *Biological Cybernetics*, *105*(5–6), 355–370.

18. Hornby, G. S., Takamura, S., Yokono, J., Hanagata, O., Yamamoto, T., & Fujita, M. (2000). Evolving robust gaits with AIBO. In B. Carlisle & O. Khatib (Eds.), *IEEE International Conference on Robotics and Automation: Proceedings* (pp. 3040–3045). Piscataway, NJ: IEEE.

19. Iizuka, H., Ando, H., & Maeda, T. (2013). Extended homeostatic adaptation model with metabolic causation in plasticity mechanism—toward constructing a dynamic neural network model for mental imagery. *Adaptive Behavior*, *21*(4), 263–273.

20. Izquierdo, E., & Bührmann, T. (2008). Analysis of a dynamical recurrent neural network evolved for two qualitatively different tasks: Walking and chemotaxis. In S. Bullock, J. Noble, R. A. Watson, & M. A. Bedau (Eds.), *Artificial life XI: Proceedings of the 11th International Conference on the Simulation and Synthesis of Living Systems* (pp. 257–264). Cambridge, MA: MIT Press.

21. Jakobi, N., Husbands, P., & Harvey, I. (1995). Noise and the reality gap: The use of simulation in evolutionary robotics. In F. Morán, A. Moreno, J. J. Merelo, & P. Chacón (Eds.), *Advances in artificial life: Third European Conference on Artificial Life, Proceedings* (pp. 704–720). Berlin, Heidelberg: Springer.

22. Jakobi, N. (1997). Evolutionary robotics and the radical envelope-of-noise hypothesis. *Adaptive Behavior*, *6*(2), 325–368.

23. Jakobi, N. (1998). *Minimal simulations for evolutionary robotics*. Unpublished doctoral dissertation, University of Sussex, Falmer, East Sussex.

24. Jakobi, N. (1998). Running across the reality gap: Octopod locomotion evolved in a minimal simulation. In P. Husbands & J.-A. Meyer (Eds.), *Evolutionary robotics: First European Workshop, Proceedings* (pp. 39–58). Berlin, Heidelberg: Springer.

25. Koos, S., Mouret, J.-B., & Doncieux, S. (2010). Crossing the reality gap in evolutionary robotics by promoting transferable controllers. In M. Pelikan & J. Branke (Eds.), *Proceedings of the 12th Annual Conference on Genetic and Evolutionary Computation* (pp. 119–126). New York: ACM.

26. Koos, S., Mouret, J.-B., & Doncieux, S. (2013). The transferability approach: Crossing the reality gap in evolutionary robotics. *IEEE Transactions on Evolutionary Computation*, *17*(1), 122–145.

27. Lehman, J., Risi, S., D'Ambrosio, D., & Stanley, K. O. (2013). Encouraging reactivity to create robust machines. *Adaptive Behavior*, *21*(1), 484–500.

28. Miglino, O., Lund, H. H., & Nolfi, S. (1995). Evolving mobile robots in simulated and real environments. *Artificial Life*, *2*(4), 417–434.

29. Nolfi, S., & Floreano, D. (2000). *Evolutionary robotics: The biology, intelligence, and technology of self-organizing machines*. Cambridge, MA: MIT Press.

30. Paolo, E. A. D. (2000). Homeostatic adaptation to inversion of the visual field and other sensorimotor disruptions. In J.-A. Meyer, A. Berthoz, D. Floreano, H. L. Roitblat, & S. W. Wilson (Eds.), *From animals to animats 6: Proceedings of the Sixth International Conference on the Simulation of Adaptive Behavior* (pp. 440–449). Cambridge, MA: MIT Press.

31. Paul, C. (2006). Morphological computation: A basis for the analysis of morphology and control requirements. *Robotics and Autonomous Systems*, *54*(8), 619–630.

32. Pfeifer, R., Iida, F., & Gómez, G. (2006). Morphological computation for adaptive behavior and cognition. *International Congress Series*, *1291*(1), 22–29.

33. Pollack, J. B., Lipson, H., Ficici, S., Funes, P., Hornby, G., & Watson, R. A. (2000). Evolutionary techniques in physical robotics. In G. S. Hornby, L. Sekanina, & P. C. Haddow (Eds.), *Evolvable systems: From biology to hardware* (pp. 175–186). Berlin, Heidelberg: Springer.

34. Pollack, J. B., & Lipson, H. (2000). The Golem project: Evolving hardware bodies and brains. In J. Lohn, A. Stoica, D. Keymeulen, & S. Colombano (Eds.), *Proceedings of the Second NASA/DoD Workshop on Evolvable Hardware* (pp. 37–42). Piscataway, NJ: IEEE.

35. Rieffel, J., & Sayles, D. (2010). Evofab: A fully embodied evolutionary fabricator. In G. Tempesti, A. M. Tyrrell, & J. F. Miller (Eds.), *Evolvable systems: From biology to hardware: 9th International Conference, ICES 2010, Proceedings* (pp. 372–380). Berlin, Heidelberg: Springer.

36. Saksida, L. M., Raymond, S. M., & Touretzky, D. S. (1997). Shaping robot behavior using principles from instrumental conditioning. *Robotics and Autonomous Systems*, *22*(3), 231–249.

37. Schmidt, M., & Lipson, H. (2011). Age-fitness Pareto optimization. In R. Riolo, T. McConaghy, & E. Vladislavleva (Eds.), *Genetic programming theory and practice VIII* (pp. 129–146). New York: Springer.

38. Singh, S. P. (1992). Transfer of learning by composing solutions of elemental sequential tasks. *Machine Learning, 8*(3–4), 323–339.

39. Slocum, A. C., Downey, D. C., & Beer, R. D. (2000). Further experiments in the evolution of minimally cognitive behavior: From perceiving affordances to selective attention. In J.-A. Meyer, A. Berthoz, D. Floreano, H. L. Roitblat, & S. W. Wilson (Eds.), *From animals to animats 6: Proceedings of the Sixth International Conference on the Simulation of Adaptive Behavior* (pp. 430–439). Cambridge, MA: MIT Press.

40. Tuci, E., Quinn, M., & Harvey, I. (2002). An evolutionary ecological approach to the study of learning behavior using a robot-based model. *Adaptive Behavior*, *10*(3–4), 201–221.

41. Yamauchi, B., & Beer, R. (1994). Integrating reactive, sequential, and learning behavior using dynamical neural networks. In D. Cliff, P. Husbands, J.-A. Meyer, & S. W. Wilson (Eds.), *From animals to animats 3: Proceedings of the Third International Conference on Simulation of Adpative Behavior* (pp. 382–391). Cambridge, MA: MIT Press.

42. Zagal, J. C., Ruiz-del-Solar, J., & Vallejos, P. (2004). Back to reality: Crossing the reality gap in evolutionary robotics. In J. Santos-Victor & M. I. Ribeiro (Eds.), *Intelligent autonomous vehicles 2004 (IAV 2004): A proceedings volume from the 5th IFAC/EURON Symposium* (on CD-ROM). Amsterdam: Elsevier.