ARTICLE

Check for updates

Embodied intelligence via learning and evolution

Agrim Gupta [1]✉, Silvio Savarese[1], Surya Ganguli[2,3,4] & Li Fei-Fei [1,4]✉

The intertwined processes of learning and evolution in complex environmental niches have resulted in a remarkable diversity of morphological forms. Moreover, many aspects of animal intelligence are deeply embodied in these evolved morphologies. However, the principles governing relations between environmental complexity, evolved morphology, and the learnability of intelligent control, remain elusive, because performing large-scale in silico experiments on evolution and learning is challenging. Here, we introduce Deep Evolutionary Reinforcement Learning (DERL): a computational framework which can evolve diverse agent morphologies to learn challenging locomotion and manipulation tasks in complex environments. Leveraging DERL we demonstrate several relations between environmental complexity, morphological intelligence and the learnability of control. First, environmental complexity fosters the evolution of morphological intelligence as quantified by the ability of a morphology to facilitate the learning of novel tasks. Second, we demonstrate a morphological Baldwin effect i.e., in our simulations evolution rapidly selects morphologies that learn faster, thereby enabling behaviors learned late in the lifetime of early ancestors to be expressed early in the descendants lifetime. Third, we suggest a mechanistic basis for the above relationships through the evolution of morphologies that are more physically stable and energy efficient, and can therefore facilitate learning and control.

[1] Department of Computer Science, Stanford University, Stanford, CA, USA. [2] Department of Applied Physics, Stanford University, Stanford, CA, USA. [3] Wu-Tsai Neurosciences Institute, Stanford University, Stanford, CA, USA. [4] Stanford Institute for Human-Centered Artificial Intelligence, Stanford University, Stanford, CA, USA. ✉email: agrim@stanford.edu; feifeili@stanford.edu

Evolution over the last 600 million years has generated a variety of "endless forms most beautiful"[1] starting from an ancient bilatarian worm[2], and culminating in a set of diverse animal morphologies. Moreover, such animals display remarkable degrees of embodied intelligence by leveraging their evolved morphologies to learn complex tasks. Indeed the field of embodied cognition posits that intelligent behaviors can be rapidly learned by agents whose morphologies are well adapted to their environment[3–5]. In contrast, the field of artificial intelligence (AI) has focused primarily on disembodied cognition, for example in domains of language[6], vision[7], or games[8].

The creation of artificial embodied agents[9,10] with well-adapted morphologies that can learn control tasks in diverse, complex environments is challenging because of the twin difficulties of (1) searching through a combinatorially large number of possible morphologies, and (2) the computational time required to evaluate fitness through lifetime learning. Hence, common strategies adopted by prior work include evolving agents in limited morphological search spaces[11–17] or focusing on finding optimal parameters given a fixed hand-designed morphology[17–20]. Furthermore, the difficulty of evaluating fitness forced prior work to (1) avoid learning adaptive controllers directly from raw sensory observations[11–13,16,21–24]; (2) learn hand-designed controllers with few (≤100) parameters[11–13,16,24]; (3) learn to predict the fitness of a morphology[15,21]; (4) mimic Lamarckian rather than Darwinian evolution by directly transmitting learned information across generations[12,15]. Moreover, prior works were also primarily limited to the simple task of locomotion over a flat terrain with agents having few degrees of freedom (DoF)[13,15] or with body plans composed of cuboids to further simplify the problem of learning a controller[11–13].

In this work, our goal is to elucidate some principles governing relations between environmental complexity, evolved morphology, and the learnability of intelligent control. However, a prerequisite for realizing this goal is the ability to simultaneously scale the creation of embodied agents across 3 axes of complexity: environmental, morphological, and control without using the above-mentioned heuristics to speed up fitness evaluation. To address this challenging requirement we propose Deep Evolutionary Reinforcement Learning (DERL) (Fig. 1a), a conceptually simple computational framework that operates by mimicking the intertwined processes of Darwinian evolution over generations to search over morphologies, and reinforcement learning within a lifetime for learning intelligent behavior from low level egocentric sensory information. A key component of the DERL framework is to use distributed asynchronous evolutionary search for parallelizing computations underlying learning, thereby allowing us to leverage the scaling of computation and models that has been so successful in other fields of AI[6,25–27] and bring it bear on the field of evolutionary robotics.

DERL opens the door to performing large-scale in silico experiments to yield scientific insights into how learning and evolution cooperatively create sophisticated relationships between environmental complexity, morphological intelligence, and the learnability of control tasks. Our key contributions are the following scientific insights. First, we create a paradigm to evaluate morphological intelligence by assessing how well each morphology facilitates the speed and performance of reinforcement learning in a large suite of novel tasks. We furthermore leverage this paradigm to demonstrate that environmental complexity engenders the evolution of morphological intelligence. Second, we demonstrate a morphological Baldwin effect in agents which both learn and evolve. In essence, we find that evolution rapidly selects morphologies that learn faster, thereby enabling behaviors learned late in the lifetime of early ancestors to be expressed early in the lifetime of their descendants. Third, we discover a mechanistic basis for the first two results through the evolution of morphologies that are more physically stable and energy efficient.

Results

DERL: a computational framework for creating embodied agents.
Previous evolutionary robotic simulations[11–13,15] commonly employed generational evolution[28,29], where in each generation, the entire population is simultaneously replaced by applying mutations to the fittest individuals. However, this paradigm scales poorly in creating embodied agents due to the significant computational burden imposed by training every member of a large population before any further evolution can occur. Inspired by recent progress in neural architecture search[30–32], we decouple the events of learning and evolution in a distributed asynchronous manner using tournament-based steady-state evolution[28,31,33]. Specifically, each evolutionary run starts with a population of $P = 576$ agents with unique topologies to encourage diverse solutions. The initial population undergoes lifetime learning via reinforcement learning[34] (RL) in parallel and the average final reward determines fitness. After initialization, each worker (CPUs) operates independently by conducting tournaments in groups of 4 wherein the fittest individual is selected as a parent, and a mutated copy (child) is added to the population after evaluating its fitness through lifetime learning. To keep the size of the population fixed we consider only the most recent P agents as alive[29,35]. By moving from generational to asynchronous parallel evolution, we do not require learning to finish across the entire population before any further evolution occurs. Instead, as soon as any agent finishes learning, the worker can immediately perform another step of selection, mutation, and learning in a new tournament.

For learning, each agent senses the world by receiving only low-level egocentric proprioceptive and exteroceptive observations and chooses its actions via a stochastic policy determined by the parameters of a deep neural network (Fig. 1b) that are learned via proximal policy optimization (PPO)[36]. See the "Methods" section for details about the sensory inputs, neural controller architectures, and learning algorithms employed by our agents.

Overall, DERL enables us to perform large-scale experiments across 1152 CPUs involving on average 10 generations of evolution that search over and train 4000 morphologies, with 5 million agent-environment interactions (i.e., learning iterations) for each morphology. At any given instant of time, since we can train 288 morphologies in parallel asynchronous tournaments, this entire process of learning and evolution completes in <16 h. To our knowledge, this constitutes the largest scale simulations of simultaneous morphological evolution and RL to date.

UNIMAL: A UNIversal aniMAL morphological design space.
Prior work on studying artificial embodied evolution in expressive design spaces has primarily been studied in the context of soft robotics, generally using predetermined actuation patterns for control[22–24,37–40]. Instead, here we use the MuJoCo[41] simulator as our focus is on sensorimotor learning via RL. MuJoCo is currently the dominant platform for the development of reinforcement and robot learning algorithms[42]. Due to the limited expressiveness of prior[15,18–20] hand-designed design spaces of rigid body agents, we develop the UNIMAL design space (Fig. 1d). Concurrently, Zhao et al.[21] also introduced a rich design space for rigid body agent design for locomotion tasks, although this work did not address the problem of learning controllers for these morphologies without requiring a complete and accurate physics model of the agent and the simulator. In contrast, the UNIMAL design space contains morphologies that can learn locomotion and mobile manipulation in challenging stochastic
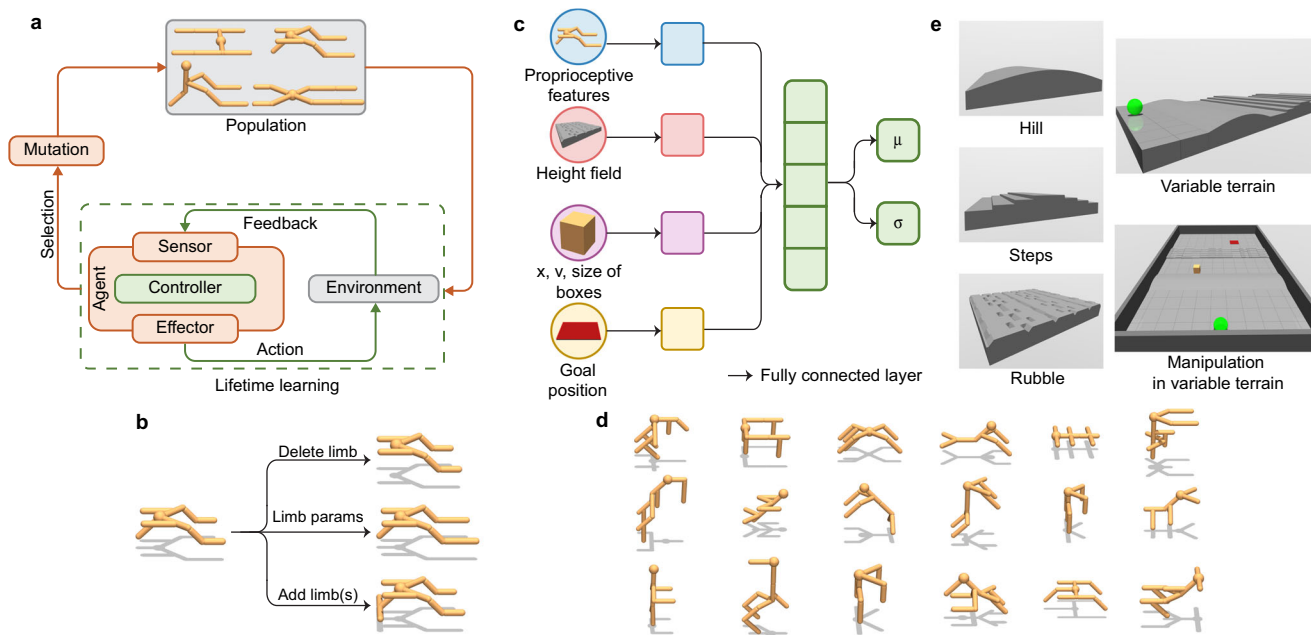
Fig. 1 DERL overview. a DERL is a general framework to make embodied agents via two interacting adaptive processes. An outer loop of evolution optimizes agent morphology via mutation operations, some of which are shown in (**b**) and an inner reinforcement learning loop optimizes the parameters of a neural controller (**c**). **d** Example agent morphologies in the UNIMAL design space. **e** Variable terrain consists of three stochastically generated obstacles: hills, steps, and rubble. In manipulation in variable terrain, an agent must start from an initial location (green sphere) and move a box to a goal location (red square).

environments without requiring any model of either the agent or the environment.

Our genotype, which directly encodes the agent morphology, is a kinematic tree corresponding to a hierarchy of articulated 3D rigid parts connected via motor actuated hinge joints. Nodes of the kinematic tree consist of two-component types: a sphere representing the head which forms the root of the tree, and cylinders representing the limbs of the agent. Evolution proceeds through asexual reproduction via three classes of mutation operations (see "Methods") that: (1) either shrink or grow the kinematic tree by growing or deleting limbs (Fig. 1d); (2) modify the physical properties of existing limbs, like their lengths and densities (Fig. 1d); (3) modify the properties of joints between limbs, including degrees of freedom (DoF), angular limits of rotation, and gear ratios. Importantly we only allow paired mutations that preserve bilateral symmetry, an evolutionarily ancient conserved feature of all animal body plans originating about 600 million years ago[2]. A key physical consequence is that the center of mass of every agent lies on the sagittal plane, thereby reducing the degree of control required to learn left-right balancing. Despite this constraint, our morphological design space is highly expressive, containing $\sim 10^{18}$ unique agent morphologies with <10 limbs (see Supplementary Table 1).

**Evolution of diverse morphologies in complex environments.** DERL enables us for the first time to move beyond locomotion in flat terrain to simultaneously evolve morphologies and learn controllers for agents in three environments (Fig. 1e) of increasing complexity: (1) Flat terrain (FT); (2) Variable terrain (VT); and (3) Non-prehensile manipulation in variable terrain (MVT). VT is an extremely challenging environment as during each episode a new terrain is generated by randomly sampling a sequence of obstacles. Indeed, prior work[43] on learning locomotion in a variable terrain for a simple 9 DoF planar 2D walker required 10^7 agent-environment interactions, despite using curriculum learning and a morphology-specific reward function.

MVT posses additional challenges since the agent must rely on complex contact dynamics to manipulate the box from a random location to a target location while also traversing VT. See "Methods" for a detailed description of these complex stochastic environments.

DERL is able to find successful morphological solutions for all three environments (Fig. 2a; see Supplementary Video for illustrations of learned behaviors). Over the course of evolution the average fitness of the entire population improves by a factor of about 3 in FT/VT and a factor of 2 in MVT. Indeed morphological evolution plays a substantial role in increasing the fitness of even the best morphologies across all three environments (Supplementary Fig. 1b).

DERL also finds a diversity of successful solutions (Fig. 2b–h). Maintaining solution diversity is generically challenging for most evolutionary dynamics, as often only 1 solution and its nearby variations dominate. In contrast, by moving away from generational evolution in which the entire population competes simultaneously to survive in the next generation, to asynchronous parallel small tournament based competitions, DERL enables ancestors with lower initial fitness to still contribute a relatively large abundance of highly fit descendants to the final population (Fig. 2b). Given the initial population exhibits morphological diversity, this evolutionary dynamics, as visualized by both phylogenetic trees (Fig. 2c–e) and Muller plots[44] (Fig. 2f–h), thereby ensures final population diversity without sacrificing fitness. Indeed, the set of evolved morphologies include different variations of bipeds, tripeds, and quadrupeds with and without arms (Fig. 2i–k, Supplementary Figs. 3, 4).

We analyze the progression of different morphological descriptors across the three environments (Supplementary Fig. 2), finding a strong impact of environment on evolved morphologies. While agents evolved in all environments have similar masses and control complexity (as measured by DoF ≈ 16), VT/MVT agents tend to be longer along the direction of forward motion and shorter in height compared FT agents. FT agents are less space-
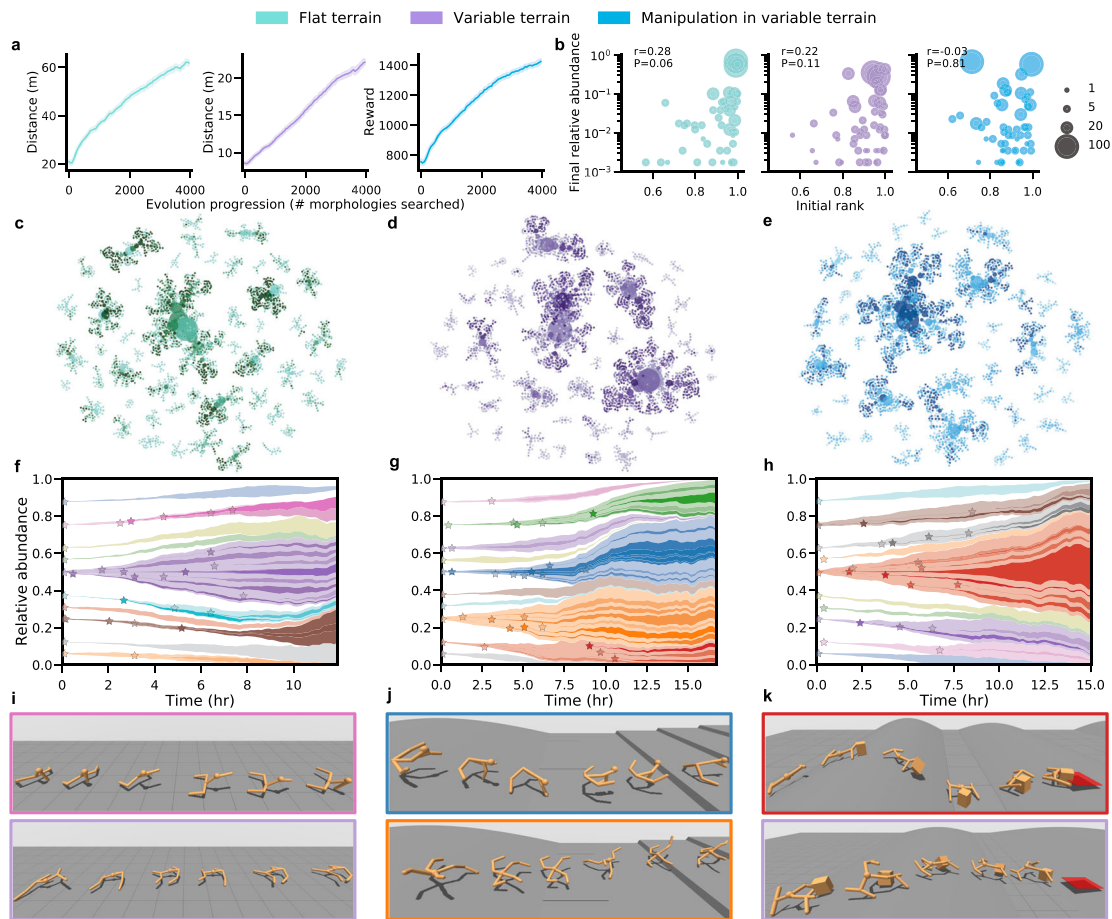
Fig. 2 Evolutionary dynamics in multiple environments. a Mean and 95% bootstrapped confidence intervals of the fitness of the entire population across 3 evolutionary runs. **b** Each dot represents a lineage that survived to the end of one of 3 evolutionary runs. Dot size reflects the total number of beneficial mutations (see "Methods") accrued by the lineage. The founder of a lineage need not have extremely high initial fitness rank in order for it's lineage to comprise a reasonably high fraction of the final population. It can instead achieve population abundance by accruing many beneficial mutations starting from a lower rank (i.e., large dots that are high and to the left). **c–e** Phylogenetic trees of a single evolutionary run where each dot represents a single UNIMAL, dot size reflects number of descendants, and dot opacity reflects fitness, with darker dots indicating higher fitness. These trees demonstrate that multiple lineages with descendants of high fitness can originate from founders with lower fitness (i.e., larger lighter dots). **f–h** Muller diagrams[44] showing relative population abundance over time (in the same evolutionary run as in (**c–e**) of the top 10 lineages with the highest final population abundance. Each color denotes a different lineage and the opacity denotes its fitness. Stars denote successful mutations which changed agent topology (i.e., adding/deleting limbs) and resulted in a sub-lineage with more than 20 descendants. The abundance of the rest of the lineages is reflected by white space. **i–k** Time-lapse images of agent policies in each of the three environments with boundary color corresponding to the lineages above. **b** Shown are the correlation coefficients (r) and P values obtained from two-tailed Pearson's correlation.

filling compared to VT/MVT agents, as measured by the coverage[45,46] of the morphology (see "Methods" for the definition of coverage). The less space-filling nature of FT agents reflects a common strategy to have limbs spaced far apart on the body giving them full range of motion (Fig. 2i, Supplementary Figs. 3a, 4a). Agents in FT exhibit both a falling forward locomotion gait and a lizard-like gait (Fig. 2i). Agents evolved in VT are often similar to FT but with additional mechanisms to make the gait more stable. For example, instead of having a single limb attached to the head which breaks falls and propels the agent forward, VT agents have two symmetrical limbs providing greater stability and maneuverability (Fig. 2j, k, Supplementary Fig. 3a, b). Finally, agents in MVT develop forward reaching arms mimicking pincer or claw-like mechanisms that enable guiding a box to a goal position (Fig. 2k, Supplementary Figs. 3c, 4c).

Environmental complexity engenders morphological intelligence. The few prior analyses of the impact of environment on evolved morphologies have focused on measuring various

morphological descriptors[46] or on morphological complexity[45]. However, a key challenge to designing any intelligent agent lies in ensuring that it can rapidly adapt to any new task. We thus focus instead on understanding how this capacity might arise through combined learning and evolution by characterizing the intelligence embodied in a morphology as a consequence of it's evolutionary environment. Concretely, we compute how much a morphology facilitates the process of learning a large set of test tasks. This approach is similar to evaluating the quality of latent neural representations by computing their performance on downstream tasks via transfer learning[47–49]. Thus in our framework, intelligent morphologies by definition facilitate faster and better learning in downstream tasks. We create a suite of 8 tasks (Fig. 3a; see Supplementary Video for illustrations of learned behaviors) categorized into 3 domains testing agility (patrol, point navigation, obstacle, and exploration), stability (escape and incline), and manipulation (push box incline and manipulate ball) abilities of the agent morphologies. Controllers for each task are learned from scratch, thus ensuring that differences in performance are solely due to differences in morphologies.
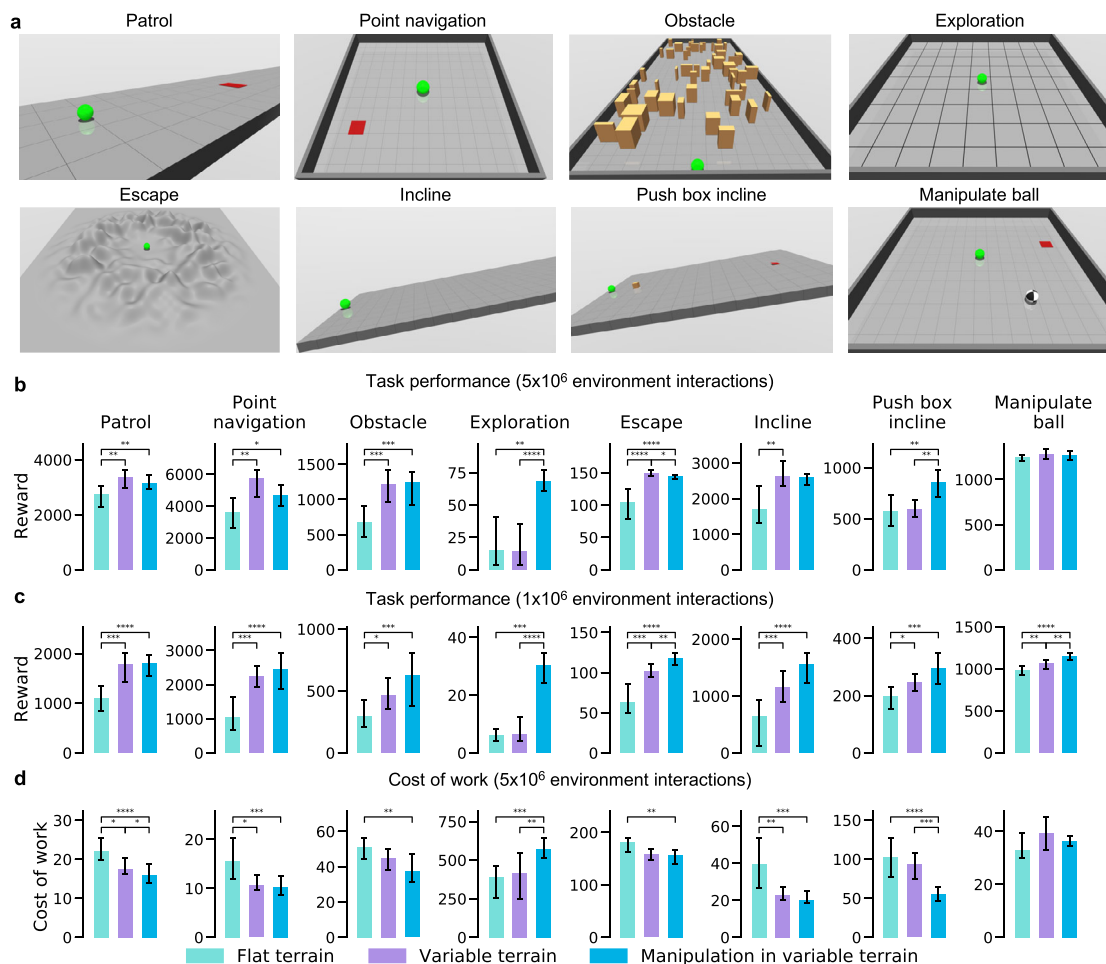
Fig. 3 Environmental complexity fosters morphological intelligence. a Eight test tasks for evaluating morphological intelligence across 3 domains spanning stability, agility, and manipulation ability. Initial agent location is specified by a green sphere, and goal location by a red square (see "Methods" for detailed task descriptions). **b–d** We pick the 10 best-performing morphologies across 3 evolutionary runs per environment. Each morphology is then trained from scratch for all 8 test tasks with 5 different random seeds. Bars indicate median reward ($n = 50$) (**b, c**) and cost of work (**d**) with error bars denoting 95% bootstrapped confidence intervals and color denoting evolutionary environment. **b** Across 7 test tasks, agents evolved in MVT perform better than agents evolved in FT. **c** With reduced learning iterations (5 million in (**b**) vs 1 million in (**c**)) MVT/VT agents perform significantly better across all tasks. **d** Agents evolved in MVT are more energy efficient as measured by lower cost of work despite no explicit evolutionary selection pressure favoring energy efficiency. Statistical significance was assessed using the two-tailed Mann–Whitney U Test; $*P < 0.05$; $**P < 0.01$; $***P < 0.001$; $****P < 0.0001$.

We first test the hypothesis that evolution in more complex environments generates more intelligent morphologies that perform better in our suite of test tasks (Fig. 3b). We find that across 7 test tasks, agents evolved in MVT perform better than agents evolved in FT. VT agents perform better than FT agents in 5 out of 6 tasks in the domains of agility and stability, but have similar performance in the manipulation tasks. To test the speed of learning, we repeat the same experiment with 1/5th of the learning iterations (Fig. 3c). The differences between MVT/VT agents and FT agents are now more pronounced across all tasks. These results suggest that morphologies evolved in more complex environments are more intelligent in the sense that they facilitate learning many new tasks both better and faster.

Demonstrating strong morphological Baldwin effect. The coupled dynamics of evolution over generations and learning within a lifetime have long been conjectured to interact with each other in highly nontrivial ways. For example, Lamarckian inheritance, an early but now disfavored[50] theory of evolution, posited that behaviors learned by an individual within its lifetime could be directly transmitted to its progeny so that they would be available as instincts soon after birth. We now know however that known heritable characters are primarily transmitted to the next generation through the genotype. However, over a century ago, Baldwin[51] conjectured an alternate mechanism whereby behaviors that are initially learned over a lifetime in early generations of evolution will gradually become instinctual and potentially even genetically transmitted in later generations. This Baldwin effect seems on the surface like Lamarckian inheritance, but is strictly Darwinian in origin. A key idea underlying this conjecture[52,53] is that learning itself comes with likely costs in terms of the energy and time required to acquire skills. For example, an animal that cannot learn to walk early in life may be more likely to die, thereby yielding a direct selection pressure on genotypic modifications that can speed up learning of locomotion. More generally, in any environment containing a set of challenges that are fixed over evolutionary timescales, but that also come with a fitness cost for the duration of learning within a lifetime, evolution may find genotypic modifications that lead to faster phenotypic learning.

In its most general sense, the Baldwin effect deals with the intertwined dynamics of any type of phenotypic plasticity within a lifetime (e.g., learning, development), and evolution over

generations[52]. Prior simulations have shown the Baldwin effect in non-embodied agents which learn and evolve[54–56]. In the context of embodied agents, Kriegman et al.[24] showcase the Baldwin effect in soft-robots evolved to locomote in a flat terrain, where the within lifetime phenotypic plasticity mechanism is development. The developmental process in this work consists of a single ballistic straight-line trajectory in the parameters of a morphology and its controller. As noted earlier, although the design space used in the context of soft robotics is more expressive, it generally comes at the cost of using predetermined, rather than learned, actuation patterns for control[22–24,37–40]. Hence, the ballistic developmental process in Kriegman et al.[24] is non-adaptive; i.e., it cannot learn from or be influenced in any way by environmental experience. Moreover, controller parameters are directly transmitted to the next generation and are not learned from scratch.

However, another common form of phenotypic plasticity, which we focus on in this paper, lies in sensorimotor learning of a controller through direct interaction with the environment, starting from a tabula rasa state with no knowledge of the parameters of controllers from previous generations. In this general setting, we also find evidence for the existence of a morphological Baldwin effect, as reflected by a rapid reduction over generations in the learning time required to achieve a criterion level of fitness for the top 100 agents in the final population in all three environments (Fig. 4a). Remarkably, within only 10 generations, average learning time is cut in half. As an illustrative example of how learning is accelerated, we show the learning curves for different generations of an agent evolved in FT (Fig. 4d). The 8th generation agent not only outperforms the 1st generation agent by a factor of 2 at the end of learning, but can also achieve the final fitness of the first generation agent in 1/5th the time. Moreover, we note that we do not have any explicit selection pressure in our simulations for fast learning, as the fitness of a morphology is determined solely by its performance at the end of learning. Nevertheless, evolution still selects for faster learners without any direct selection pressure for doing so. Thus

we showcase a stronger form of the Baldwin effect by demonstrating that an explicit selection pressure for the speed of skill acquisition is not necessary for the Baldwin effect to hold. Intriguingly, the existence of this morphological Baldwin effect could be exploited in future studies to create embodied agents with lower sample complexity and higher generalization capacity.

Mechanistic underpinning. We next search for a potential mechanistic basis for how evolution may both engender morphological intelligence (Fig. 3b, c) as well select for faster learners without any direct selection pressure for learning speed (i.e., the stronger form of the Baldwin effect in Fig. 4a). We hypothesize, along the lines of conjectures in embodied cognition[3–5], that evolution discovers morphologies that can more efficiently exploit the passive dynamics of physical interactions between the agent body and the environment, thereby simplifying the problem of learning to control, which can both enable better learning in novel environments (morphological intelligence), and faster learning over generations (Baldwin effect). Any such intelligent morphology is likely to exhibit the physical properties of both energy efficiency and passive stability, and so we examine both properties.

We define Cost of Work (COW) as the amount of energy spent per unit mass to accomplish a goal; with lower COW indicating higher energy efficiency (see "Methods"). Surprisingly, without any direct selection pressure for energy efficiency, evolution nevertheless selected for more energy-efficient morphological solutions (Fig. 4c). We verify such energy efficiency is not achieved simply by reducing limb densities (Supplementary Fig. 2e). On the contrary, across all three environments, the total body mass actually increases suggesting that energy efficiency is achieved by selecting for morphologies which more effectively leverage the passive physical dynamics of body-environment interactions. Moreover, morphologies which are more energy-efficient perform better (Fig. 5a) and learn faster (Fig. 5b) at any fixed generation. Similarly, evolution selects more passively stable (see "Methods") morphologies over time in all three
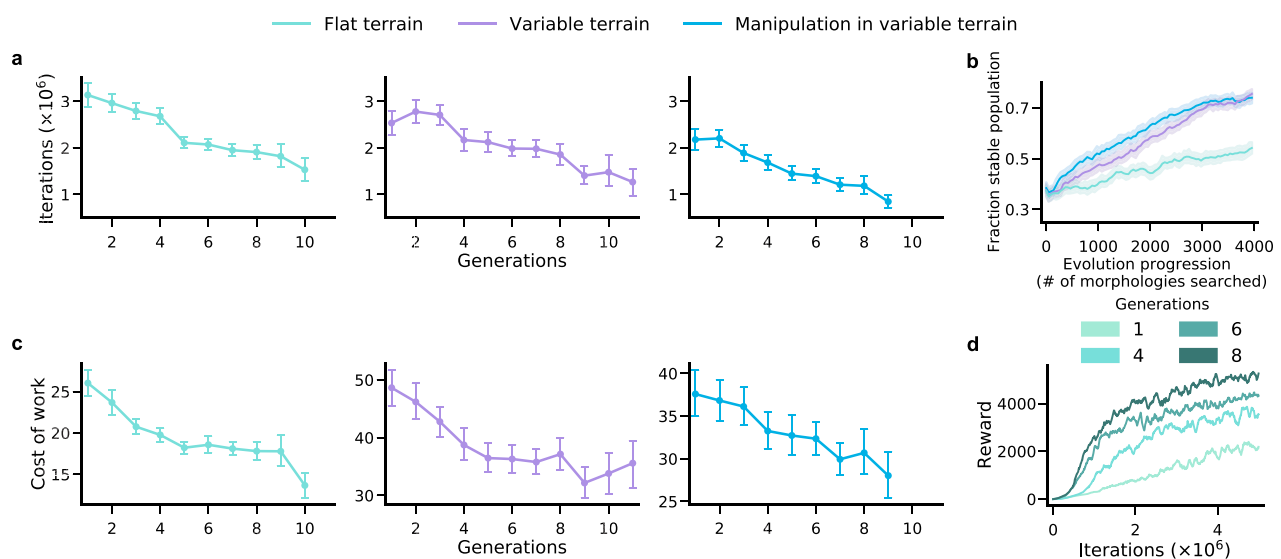
Fig. 4 A morphological Baldwin effect and its relationship to energy efficiency and stability. a Progression of mean ($n = 100$) iterations to achieve the 75th percentile fitness of the initial population for the lineages of the best 100 agents in the final population across 3 evolutionary runs. **b** Fraction of stable morphologies (see "Methods") averaged over 3 evolutionary runs per environment. This fraction is higher in VT and MVT than FT, indicating that these more complex environments yield an added selection pressure for stability. **c** Mean cost of work (see "Methods") for same lineages as in (**a**). **d** Learning curves for different generations of an illustrative agent evolved in FT indicate that later generations not only perform better but also learn faster. Thus overall evolution simultaneously discovers morphologies that are more energy efficient (**c**), stable (**b**), and simplify control, leading to faster learning (**a**). Error bars (**a**, **c**) and shaded region (**b**) denote 95% bootstrapped confidence interval.

environments, though the fraction of stable morphologies is higher in VT/MVT relative to FT, indicating higher relative selection pressure for stability in these more complex environments (Fig. 4b). Thus, over evolutionary time, both energy efficiency (Fig. 4c) and stability (Fig. 4b) improve in a manner that is tightly correlated with learning speed (Fig. 4a).

These correlations suggest that energy efficiency and stability may be key physical principles that partially underpin both the evolution of morphological intelligence and the Baldwin effect. With regards to the Baldwin effect, variations in energy efficiency lead to positive correlations across morphologies between two distinct aspects of learning curves: the performance at the end of a lifetime, and the speed of learning at the beginning. Thus evolutionary processes that only select for the former will implicitly also select for the latter, thereby explaining the stronger form of the evolutionary Baldwin effect that we observe. With regards to morphological intelligence, we note that MVT and VT agents possess more intelligent morphologies compared to FT agents as evidenced by better performance in test tasks (Fig. 3b), especially with reduced learning iterations (Fig. 3c). Moreover,

VT/MVT agents are also more energy efficient compared to FT agents (Fig. 3d). An intuitive explanation of this differential effect of environmental complexity is that the set of subtasks that must be solved accumulates across environments from FT to VT to MVT. Thus, MVT agents must learn to solve more subtasks than FT agents in the same amount of learning iterations. This may result in a higher implicit selection pressure for desirable morphological traits like stability and energy efficiency in MVT/VT agents as compared to FT agents. And in turn, these traits may enable better, faster, and more energy-efficient performance in novel tasks for MVT/VT agents relative to FT agents (Fig. 3b–d).

Discussion

The field of AI, which seeks to reproduce and augment biological intelligence, has focused primarily on disembodied learning methods. In contrast, embodied cognition[3–5] posits that morphologies adapted to a particular environment could greatly simplify the problem of learning intelligent behaviors[3,4]. In this
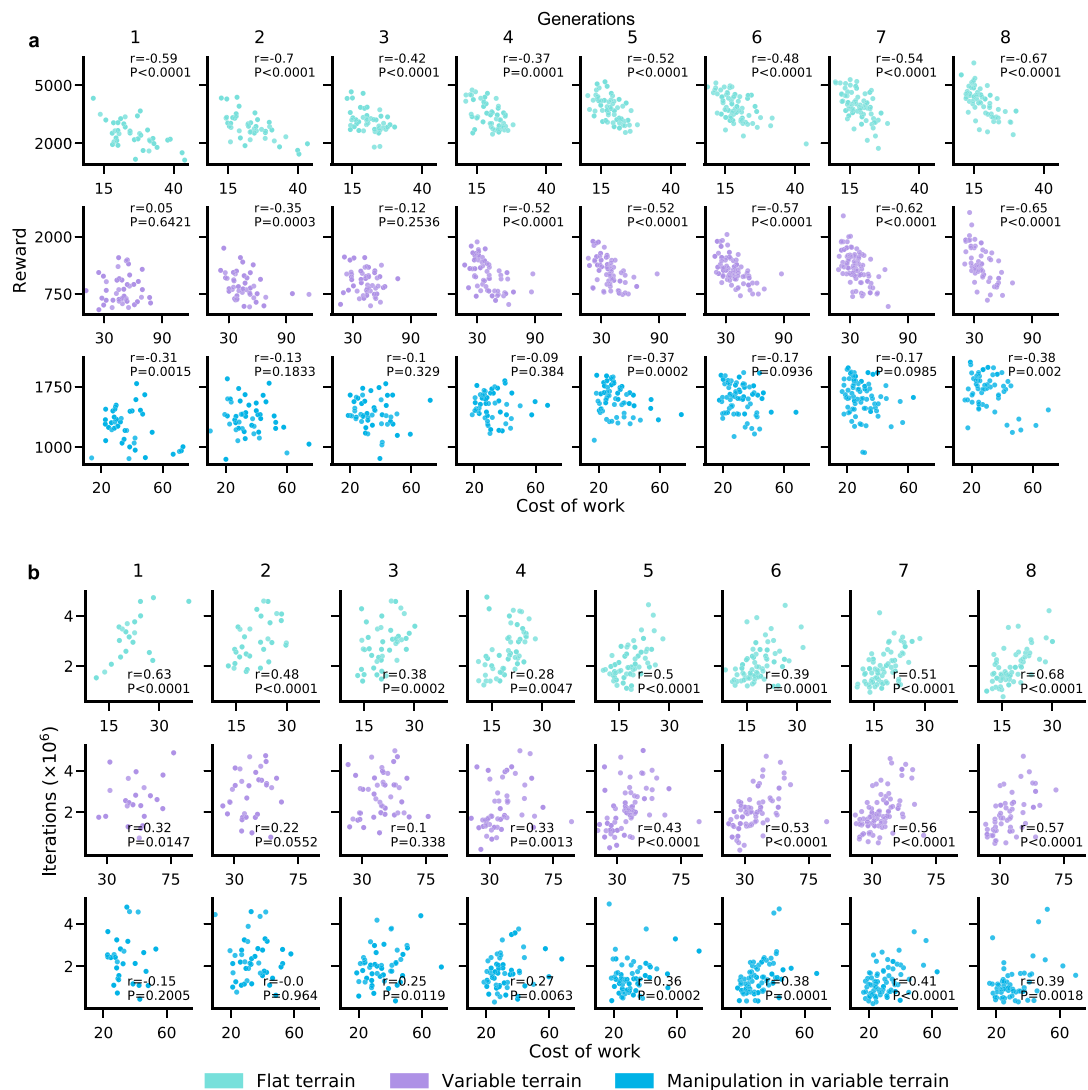
Fig. 5 Relationship between energy efficiency, fitness, and learning speed. a Correlation between fitness (reward at the end of lifetime learning) and cost of work for the top 100 agents across 3 evolutionary runs. **b** Correlation between learning speed (iterations required to achieve the 75th percentile fitness of the initial population same as Fig. 4a) and cost of work for same top 100 agents as in (**a**). Across all generations, morphologies which are more energy-efficient perform better (negative correlation) and learn faster (positive correlation). **a**, **b** Shown are the correlation coefficients (*r*) and *P* values obtained from two-tailed Pearson's correlation.

work, we take a step towards creating intelligent embodied agents via deep evolutionary reinforcement learning (DERL), a conceptually simple framework designed to leverage advances in computational and simulation capabilities. A key strength of our framework is the ability to find diverse morphological solutions, because asynchronous nature of the evolutionary dynamics, in conjunction with the use of aging criteria for maintaining population size, results in a weaker selection pressure compared to techniques based on generational evolution. Although DERL enables us to take a significant step forward in scaling the complexity of evolutionary environments, an important line of future work will involve designing more open-ended, physically realistic, and multi-agent evolutionary environments.

We leverage the large-scale simulations made possible by DERL to yield scientific insights into how learning, evolution, and environmental complexity can interact to generate intelligent morphologies that can simplify control by leveraging the passive physics of body-environment interactions. We introduce a paradigm to evaluate morphological intelligence by measuring how well a morphology facilitates the speed and performance of reinforcement learning in a suite of new tasks. Intriguingly, we find that morphologies evolved in more complex environments are better and faster at learning many novel tasks. Our results can be viewed as a proof of concept that scaling the axis of environmental complexity could lead to the creation of embodied agents which could quickly learn to perform multiple tasks in physically realistic environments. An important future avenue of exploration is the design of a comprehensive suite of evaluation tasks to quantify how agent morphologies could enhance learning of more complex and human-relevant behavior.

We also find that the fitness of an agent can be rapidly transferred within a few generations of evolution from its phenotypic ability to learn to its genotypically encoded morphology through a morphological Baldwin effect. Indeed this Baldwinian transfer of intelligence from phenotype to genotype has been conjectured to free up phenotypic learning resources to learn more complex behaviors in animals[57], including the emergence of language[58] and imitation[59] in humans. The existence of a morphological Baldwin effect could potentially be exploited in future studies to create embodied agents in open-ended environments which show greater sample efficiency and generalization capabilities in increasingly complex tasks. Finally, we also showcase a mechanistic basis for morphological intelligence and the Baldwin effect, likely realized through implicit selection pressure for favorable traits like increased passive stability and better energy efficiency.

Overall, we hope our work encourages further large-scale explorations of learning and evolution in other contexts to yield new scientific insights into the emergence of rapidly learnable intelligent behaviors, as well as new engineering advances in our ability to instantiate them in machines.

Methods

Distributed asynchronous evolution. Simultaneously evolving and learning embodied agents with many degrees of freedom that can perform complex tasks, using only low-level egocentric sensory inputs, required developing a highly parallel and efficient computational framework which we call Deep Evolutionary Reinforcement Learning (DERL). Each evolutionary run starts with an initial population of P agents (here $P = 576$) with unique randomly generated morphologies (described in more detail below) chosen to encourage diverse solutions, as evidenced in (Fig. 2b), and prevent inefficient allocation of computational resources on similar morphologies. Controllers for all P initial morphologies are learned in parallel for 5 million agent-environment interactions (learning iterations) each, and the average reward attained over approximately the last 100,000 iterations at the end of lifetime learning yields a fitness function over morphologies. Starting from this initial population, nested cycles of evolution and learning proceed in an asynchronous parallel fashion.

Each evolutionary step consists of randomly selecting $T = 4$ agents from the current population to engage in a tournament[31,35]. In each tournament, the agent morphology with the highest fitness among the 4 is selected to be a parent. Its morphology is then mutated to create a child which then undergoes lifetime learning to evaluate its fitness. Importantly, the child starts lifetime learning with a randomly initialized controller, so that only morphological information is inherited from the parent. Such tabula rasa RL can be extremely sample inefficient especially in complex environments. Hence, we further parallelize experience collection for training each agent over 4 CPU cores. Two hundred and eighty-eight such tournaments and child training are run asynchronously and in parallel over 288 workers each consisting of 4 CPUs, for a total of 1152 CPUs. The entire computation takes place over 16 Intel Xeon Scalable Processors (Cascade Lake) each of which has 72 CPUs yielding the total of 1152 CPUs. We note that we simply chose the population size to be an integer multiple of 288, though our results do not depend on this very specific choice.

Typically in generational evolution, in each cycle, the whole population is replaced in a synchronized manner. Thus, maintaining population size is trivial albeit at the cost of synchronization overhead. An alternative and perhaps more natural approach is steady-state evolution wherein usually a small fraction of individuals are created at every step of the algorithm and then they are inserted back into the population, consequently coexisting with their parents[33]. The population size can be maintained by either removing the least fit individual in a tournament or utilizing the notion of age. In DERL, we keep the population size P the same by removing the oldest member of the population after a child is added[31]. We choose aging criteria for maintaining population size for the following reasons:

(1) Encourage diversity: The notion of age has also been utilized in the context of generational evolution to foster diversity of solutions by avoiding premature convergence[60]. This is typically achieved by using age in conjunction with fitness as a selection criteria while performing tournaments. In DERL, each agent gets multiple opportunities to accumulate a beneficial mutation that increases its fitness. If multiple mutations do not confer a fitness advantage to an agent, then, and only then, will the agent eventually get selected out since we keep the most recent P agents alive. In contrast, if we removed the least fit individual in a tournament then low fitness agents in the initial population will quickly get eliminated and result in lower final diversity. This weaker selection pressure in conjunction with the asynchronous nature of the algorithm results in a diversity of successful solutions (Fig. 2b–h).

(2) Robustness: Aging criteria results in renewal of the whole population i.e., irrespective of fitness level the agent will be eventually removed (non-elitist evolution). Thus, the only way for a genotype to survive for a long time is by inheritance through generations and by accumulating beneficial mutations. Moreover, since each agent is trained from scratch this also ensures that an agent did not get lucky with one instance of training. Hence, aging ensures robustness to genotypes in the initial population which might have high fitness by chance but are actually not optimal for the environment. Indeed, we find that initial fitness is not correlated with final abundance across all three environments (Fig. 2b).

(3) Fault tolerance: In a distributed setting fault tolerance to any node failure is extremely important. In case of aging evolution if any compute node fails, a new node can be brought online without affecting the current population. In contrast, if population size is maintained by removing least fit individual in a tournament, compute node failures will need additional book keeping to maintain total population size. Fault tolerance significantly reduces the cost to run DERL on cloud services by leveraging spare unused capacity (spot instances) which is often up to 90% cheaper compared to on-demand instances.

UNIversal aniMAL (UNIMAL) design space. The design of any space of morphologies is subject to a stringent tradeoff between the richness and diversity of realizable morphologies and the computational tractability of finding successful morphologies by evaluating it's fitness. Here, we introduce the UNIMAL design space (Fig. 1d), which is an efficient search space over morphologies that introduces minimal constraints while containing physically realistic morphologies and gaits that can learn locomotion and mobile manipulation.

A genotype for morphologies can either be represented via direct encoding or indirect encoding. Prior work on evolving soft robots has often used powerful indirect encoding schemes like CPNN-NEAT[61] encodings which can create complex regularities such as symmetry and repetition[22,23]. However, for evolving rigid body agents direct encoding is often used. Hence, we use direct encoding and consequently our genotype is a kinematic tree, or a directed acyclic graph, corresponding to a hierarchy of articulated 3D rigid parts connected via motor actuated hinge joints. Nodes of the kinematic tree consist of two component types: a sphere representing the head which forms the root of the tree, and cylinders representing the limbs of the agent. Evolution proceeds through asexual reproduction via an efficient set of three classes of mutation operations (Fig. 1b) that: (1) either shrink or grow the kinematic tree by starting from a sphere and growing or deleting limbs (grow limb(s), delete limb(s)); (2) modify the physical properties of existing limbs, like their lengths and densities (mutate density, limb params); (3) modify the properties of joints between limbs, including degrees of freedom (DoF), angular limits of rotation, and gear ratios (mutate DoF, joint angle, and gear). During the population initialization phase, a new morphology is created by first sampling a total number of limbs to grow and then applying mutation

operations until the agent has the desired number of limbs. We now provide a description of the mutation operations in detail.

Grow limb(s). This mutation operation grows the kinematic tree by adding at most 2 limbs at a time. We maintain a list of locations where a new limb can be attached. The list is initialized with center of the root node. To add a new limb we randomly sample an attachment location from a uniform distribution over possible locations, and randomly sample the number of limbs to add as well as the limb parameters. Limb parameters (Supplementary Table 1) include radius, height, limb density, and orientation w.r.t. to the parent limb. We enforce that all limbs have the same density, so only the first grow limb mutation samples limb density, and all subsequent limbs have the same density. However, due to the mutate density operation, all limb densities can change simultaneously under this mutation from a parent to a child. We also only allow limb orientations which ensure that the new limb is completely below the parent limb; i.e., the kinematic tree can only grow in the downward direction. The addition of a limb is considered successful, if after attachment of the new limb the center of mass of the agent lies on the sagittal plane and there are no self intersections. Self intersections can be determined by running a short simulation and detecting collisions between limbs. If the new limb collides with other limbs at locations other than the attachment site, the mutation is discarded. Finally, if the limb(s) were successfully attached we update the list of valid attachment locations by adding the mid and end points of the new limb(s). Symmetry is further enforced by ensuring that if a pair of limbs were added then all future mutation operations will operate on both the limbs.

Delete limb(s). This mutation operation only affects leaf nodes of the kinematic tree. Leaf limb(s) or end effectors are randomly selected and removed from the kinematic tree in a manner that ensures symmetry.

Mutate limb parameters. A limb is modeled as cylinder which is parameterized by it's length and radius (Supplementary Table 1). Each mutation operation first selects a limb or pair of symmetric limbs to mutate and then randomly samples new limb parameters for mutation.

Mutate density. In our design space, all limbs have the same density. To mutate the density we randomly select a new density value (Supplementary Table 1). Similarly, we can also mutate the density of the head.

Mutate DoF, gear, and joint angle. We describe the three mutations affecting the joints between limbs together, due to their similarity. Two limbs are connected by motor actuated hinge joints. A joint is parameterized by it's axis of rotation, joint angle limits, and motor gear ratio[41]. There can be at most two hinge joints between two limbs. In MuJoCo[41], each limb can be described in its own frame of reference in which the limb is extended along the z-axis. In this same frame of reference of the child limb, the possible axes of rotations between the child and parent limbs, correspond to the x-axis, the y-axis, or both (see Joint axis in Supplementary Table 1). The main thing this precludes is rotations of a limb about its own axis. Each mutation operation first selects a limb or pair of symmetric limbs to mutate and then randomly samples new joint parameters (Supplementary Table 1). For mutating DoF, in addition to selecting new axes of rotations, we also select new gears and joint angles.

The UNIMAL design space is discrete as each mutation operation chooses a random element from a corresponding list of possible parameter values (see Supplementary Table 1). Although we could easily modify the possible parameter values to be a continuous range, this would significantly increase the size of our search space. The design choice of using discrete parameters helps us avoid wasting computational resources on evaluating morphologies which are very similar, e.g., if the only difference between two morphologies is limb height of 0.10 m vs 0.11 m. Another design choice due to computational constraints is limiting the maximum number of limbs to at most 10. Each additional limb increases control complexity by increasing the number of DoFs. Hence, the primary consideration for a cap on the number limbs was to ensure that the time it takes agents to learn to maximize fitness through RL within a lifetime would remain <5 million agent-environment interactions across all three environments. We found that the learning curves of all our agents largely saturate to their maximal value within this time. Given more compute budget, the limit on the number of limbs can be increased with a commensurate increase in the number of agent-environment interactions. Finally, although we constrain the UNIMAL design space to be symmetric, it might be interesting to explore the relationship between bilaterally symmetric morphologies and morphological intelligence similar in spirit to work done by Bongard et al.[62].

Environments. DERL enables us to simultaneously evolve and learn agents in three environments (Fig. 1e) of increasing complexity: (1) Flat terrain (FT); (2) Variable terrain (VT); and (3) Non-prehensile manipulation in variable terrain (MVT). We use the MuJoCo[41] physics simulator for all our experiments. To ensure physical realism of simulation we adopt the simulation parameters (e.g., friction coefficient, gravity etc.) from standard robot and reinforcement learning benchmarks[63,64]. Moreover, these parameters have been used in numerous successful simulation to

real-world transfer experiments[65–67]. We now provide a detailed description of each environment:

Flat terrain. The goal of the agent is to maximize forward displacement over the course of an episode. At the start of an episode, an agent is initialized on one end of a square arena of size (150×150 square meters (m^2)).

Variable terrain. Similar to FT, the goal of the agent is to maximize forward displacement over the course of an episode. At the start of an episode, an agent is initialized on one end of a square arena of size (100×100 m^2). In each episode, a completely new terrain is created by randomly sampling a sequence of obstacles (Fig. 1e) and interleaving them with flat terrain. The flat segments in VT are of length $l \in [1, 3]$ m along the desired direction of motion, and obstacle segments are of length $l \in [4, 8]$ m. Each obstacle is created by sampling from a uniform distribution over a predefined range of parameter values. We consider 3 types of obstacles: 1. Hills: Parameterized by the amplitude a of sin wave where $a \in [0.6, 1.2]$ m. 2. Steps: A sequence of 8 steps of height 0.2 m. The length of each step is identical and is equal to one-eighth of the total obstacle length. Each step sequence is always 4-steps up followed by 4-steps down. 3. Rubble: A sequence of random bumps created by clipping a repeating triangular sawtooth wave at the top such that the height h of each individual bump clip is randomly chosen from the range $h \in [0.2, 0.3]$ m. Training an agent for locomotion in variable terrain is extremely challenging as prior work[43] on learning locomotion in a similar terrain for a hand-designed 9 DoF planar 2D walker required 10^7 agent-environment interactions, despite using curriculum learning and a morphology specific reward function.

Manipulation in variable terrain. This environment is like VT with an arena of size (60×40 m^2). However, here the goal of the agent is to move a box (a cube with side length 0.2 m) from it's initial position to a goal location. All parameters for terrain generation are the same as VT. In each episode, in addition to creating a new terrain, both the initial box location and final goal location are also randomly chosen with the constraint that the goal location is always further along the direction of forward motion than the box location.

These environments are designed to ensure that the number of sub-tasks required to achieve high fitness is higher for more complex environments. Specifically, a FT agent has to only learn locomotion on a flat terrain. In addition, a VT agent needs to also learn to walk on hills, steps, and rubble. Finally, along with all the sub-tasks which need to be mastered in VT, a MVT agent should also learn directional locomotion and non-prehensile manipulation of objects. The difference in arena sizes is simply chosen to maximize simulation speed while being big enough that agents cannot typically complete the task sooner than an episode length of 1000 agent-environment interactions (iterations). Hence, the arena size for MVT is smaller than VT. Indeed it would be interesting to extend these results to even more physically realistic and complex environments.

Reinforcement learning. The RL paradigm provides a way to learn efficient representations of the environment from high-dimensional sensory inputs, and use these representations to interact with the environment in a meaningful way. At each time step, the agent receives an observation o_t that does not fully specify the state (s_t) of the environment, takes an action a_t, and is given a reward r_t. A policy $\pi_\theta(a_t|o_t)$ models the conditional distribution over action $a_t \in A$ given an observation $o_t \in O(s_t)$. The goal is to find a policy which maximizes the expected cumulative reward $R = \sum_{t=0}^{H} \gamma^t r_t$ under a discount factor $\gamma \in [0, 1)$, where H is the horizon length.

Observations. At each time step, the agent senses the world by receiving low-level egocentric proprioceptive and exteroceptive observations (Fig. 1c). Proprioceptive observations depend on the agent morphology and include joint angles, angular velocities, readings of a velocimeter, accelerometer, and a gyroscope positioned at the head, and touch sensors attached to the limbs and head as provided in the MuJoCo[41] simulator. Exteroceptive observations include task-specific information like local terrain profile, goal location, and the position of objects and obstacles.

Information about the terrain is provided as 2D heightmap sampled on a non-uniform grid to reduce the dimensionality of data. The grid is created by decreasing the sampling density as the distance from the root of the body increases[43]. All heights are expressed relative to the height of the ground immediately under the root of the agent. The sampling points range from 1 m behind the agent to 4 m ahead of it along the direction of motion, as well as 4 m to the left and right (orthogonal to the direction of motion). Note the height map is not provided as input in tasks like patrol, point navigation etc. where the terrain is flat and does not have obstacles. Information about goal location for tasks like point navigation, patrol etc., and the position and velocity of objects like ball/box for manipulation tasks are provided in an egocentric fashion, using the reference frame of the head.

Our sensors make simplifying assumptions like avoiding high dimensional image input and providing the location of the box for the task of manipulation as part of exteroceptive observation. Evolving agents which could perform the same tasks in the real world would require additional computational processing to infer

the height field information from raw camera images, and development of sophisticated exploration strategies for finding objects in a new environment.

Rewards. The performance of RL algorithms is dependent on good reward function design. A common practice is to have certain components of the reward function be morphology dependent[43]. However, designing morphology-dependent reward functions is not feasible when searching over a large morphological design space. One way to circumvent this issue is to limit the design space to morphologies with similar topologies[15]. But this strategy is ill-suited as our goal is to have an extremely expressive morphological design space with minimal priors and restrictions. Hence, we keep the reward design simple, offloading the burden of learning the task from engineering reward design to agent morphological evolution.

For FT and VT at each time step t the agent receives a reward r_t given by,

$$r_t = w_x v_x - w_c \|a\|^2, \qquad (1)$$

where v_x is the component of velocity in the $+x$ direction (the desired direction of motion), a is the input to the actuators, and w_x and w_c weight the relative importance of the two reward components. Specifically, $w_x = 1$, and $w_c = 0.001$. This reward encourages the agent to make forward progress, with an extremely weak penalty for very large joint torques. Note that for selecting tournament winners in the evolutionary process, we only compare the forward progress component of the reward i.e., $w_x v_x$. Hence, there is no explicit selection pressure to minimize energy. We adopt a similar strategy for tournament winner selection in MVT.

For MVT at each time step t the agent receives a reward r_t given by,

$$r_t = w_{ao} d_{ao} + w_{og} d_{og} - w_c \|a\|^2, \qquad (2)$$

Here d_{ao} is geodesic distance between the agent and the object (box) in the previous time step minus this same quantity in the current time step. This reward component encourages the agent to come close to the box and remain close to it. d_{og} is geodesic distance between the object and the goal in previous time step minus this same quantity in the current time step. This encourages the agent to manipulate the object towards the goal. The final component involving $\|a\|^2$ provides a weak penalty on large joint torques as before. The weights w_{ao}, w_{og}, w_c determine the relative importance of the three components. Specifically, $w_{ao} = w_{og} = 100$, and $w_c = 0.001$. In addition, the agent is provided a sparse reward of 10 when it is within 0.75 m of the initial object location, and again when the object is within 0.75 m of goal location. This sparse reward further encourages the agent to minimize the distance between the object and the goal location while being close to the object.

In addition, we use early termination across all environments when we detect a fall. We consider an agent to have fallen if the height of the head of the agent drops below 50% of its original height. We found employing this early termination criterion was essential in ensuring diverse gaits. Without early termination, almost all agents would immediately fall and move in a snake-like gait.

Policy architecture. The agent chooses it's action via a stochastic policy π_θ where θ are the parameters of a pair of deep neural networks: a policy network which produces an action distribution (Fig. 1c), and a critic network which predicts discounted future returns. Each type of observation is encoded via a two-layer MLP with hidden dimensions [64, 64]. The encoded observations across all types are then concatenated and further encoded into a 64-dimensional vector, which is finally passed into a linear layer to generate the parameters of a Gaussian action policy for the policy network and discounted future returns for the critic network. The size of the output layer for the policy network depends on the number of actuated joints. We use tanh non-linearities everywhere, except for the output layers. The parameters (~250,000) of the networks are optimized using Proximal Policy Optimization[36] (PPO). Although these networks are shallow compared to modern neural networks, we believe that with an increase in simulation and computation capabilities our framework could be scaled to use more sophisticated and deeper neural networks. Our policy architecture is monolithic as we do not have to share parameters across generations. Modular policy architectures[68–70] leveraging graph neural networks could be used in Lamarckian evolutionary algorithms.

Optimization. Policy gradient methods are a popular class of algorithms for finding the policy parameters θ which maximize R via gradient ascent. Vanilla policy gradient[71] (VPG) is given by $L = \mathbb{E}[\hat{A}_t \nabla_\theta \log \pi_\theta]$, where $\hat{A}_t$ is an estimate of the advantage function. VPG estimates can have high variance and be sensitive to hyperparameter changes. To overcome this PPO[36] optimizes a modified objective $L = \mathbb{E}\big[\min(l_t(\theta)\hat{A}_t, \text{clip}(l_t(\theta), 1 - \epsilon, 1 + \epsilon)\hat{A}_t)\big]$, where $l_t(\theta) = \frac{\pi_\theta(a_t|o_t)}{\pi_{old}(a_t|o_t)}$ denotes the likelihood ratio between new and old policies used for experience collection. We use Generalized Advantage Estimation[72] to estimate the advantage function. The modified objective keeps $l_t(\theta)$ within ϵ and functions as an approximate trust-region optimization method; allowing for the multiple gradient updates for a mini-batch of experience, thereby preventing training instabilities and improving sample efficiency. We adapt an open source[73] implementation of PPO (see Supplementary Table 2) for hyperparameter values. We keep the number of learning iterations the

same across all evolutionary environments. In all environments, agents have 5 million learning iterations to perform lifetime learning.

Evaluation task suite. A key contribution of our work is to take a step towards quantifying morphological intelligence. Concretely, we compute how much a morphology facilitates the process of learning a large set of test tasks. We create a suite of 8 tasks (Fig. 3a) categorized into 3 domains testing agility (patrol, point navigation, obstacle, and exploration), stability (escape and incline), and manipulation (push box incline and manipulate ball) abilities of the agent morphologies. Controllers for each task are learned from scratch, thus ensuring that differences in performance are solely due to differences in morphologies. Note that from RL perspective both task and environment are the same, i.e., both are essentially Markov decision processes. Here, we use the term environment to distinguish between evolutionary task and test tasks. We now provide a description of the evaluation tasks and rewards used to train the agent.

Patrol. The agent is tasked with running back and forth between two goal locations 10 m apart along the x axis. Success in this task requires the ability to move fast for a short duration and then quickly change direction repeatedly. At each time step the agent receives a reward r_t given by,

$$r_t = w_{ag} d_{ag} - w_c \|a\|^2, \qquad (3)$$

where d_{ag} is geodesic distance between the agent and the goal in the previous time step minus this same quantity in the current time step, $w_{ag} = 100$, and $w_c = 0.001$. In addition, when the agent is within 0.5 m of the goal location, we flip the goal location and provide the agent a sparse reward of 10.

Point navigation. An agent is spawned at the center of a flat arena (100×100 m^2). In each episode, the agent has to reach a random goal location in the arena. Success in this task requires the ability to move in any direction. The reward function is similar to the patrol task.

Obstacle. The agent has to traverse a dense area of static obstacles and reach the end of the arena. The base and height of each obstacle vary between 0.5 and 3 m. The environment is a rectangular flat arena (150×60 m^2) with 50 random obstacles initialized at the start of each episode. Success in this task requires the ability to quickly maneuver around obstacles. The obstacle information is provided in the form of terrain height map. The reward function is similar to that of locomotion in FT.

Exploration. The agent is spawned at the center of a flat arena (100×100 m^2). The arena is discretized into grids of size (1×1 m^2) and the agent has to maximize the number of distinct squares visited. At each time step agent receives,

$$r_t = w_e(e_t - e_{t-1}) - w_c \|a\|^2, \qquad (4)$$

where e_t denotes total number of locations explored till time step t, $w_e = 1$, and $w_c = 0.001$. In addition to testing agility this task is challenging, as unlike in the case of dense locomotion rewards for previous tasks, here the agent gets a sparser reward.

Escape. The agent is spawned at the center of a bowl-shaped terrain surrounded by small hills[63] (bumps). The agent has to maximize the geodesic distance from the start location (escape the hilly region). This task tests the agent's ability to balance itself while going up/down on a random hilly terrain. At each time step the agent receives reward,

$$r_t = w_d d_{as} - w_c \|a\|^2, \qquad (5)$$

where d_{as} is geodesic distance between the agent and the initial location in the current time step minus this same quantity in the previous time step, $w_d = 1$, and $w_c = 0.001$.

Incline. The agent is tasked to move on rectangular arena (150×40 m^2) inclined at 10°. Reward is similar to FT.

Push box incline. A mobile manipulation task, where the objective is to push a box (of side length 0.2 m) along an inclined plane. The agent is spawned at the start of a rectangular arena (80×40 m) inclined at 10°. Reward is similar to MVT.

Manipulate ball. A mobile manipulation task, where the objective is to move a ball from a source location to a target location. In each episode, a ball (radius 0.2 m) is placed at a random location on a flat square arena (30×30 m) and the agent is spawned at the center. This task poses a challenging combination of locomotion and object manipulation, since the agent must rely on complex contact dynamics to manipulate the movement of the ball while also maintaining balance. Reward is similar to MVT.

Reporting methodology. The performance of RL algorithms is known to be strongly dependent on the choice of the seed for random number generators[74]. To control for this variation, within an evolutionary run we use the same seed for all lifetime learning across all morphologies. However, we take several steps to ensure robustness to this choice. First, we repeat each evolutionary run three times for each environment with different random seeds. Then to find the best morphologies for each environment in a manner that is robust to choice of seed, we select the top 3 agents from all surviving lineages across all 3 evolutionary runs. Typically, in a single evolutionary run we find 15–20 surviving lineages, yielding a total of 135–180 good morphologies per environment (at 3 per lineage over 3 evolutionary runs). Then we further train these morphologies 5 times with 5 entirely new random seeds. This final step ensures robustness to choice of seed without having to run evolution many times. Finally, we select the 100 best agents in these new training runs for each environment. These 100 agents are used to generate the data shown in Fig. 2a, Fig. 4a, c, and Fig. 5. We also compare the performance of the top 10 out of these 100 agents across the suite of 8 test tasks (Fig. 3b–d). For all test tasks we use the same network architecture, hyperparameter values, and learning procedure as used during evolution, and train the controller from scratch with 5 random seeds.

Cost of work. Cost of transportation[75] (COT) is a dimensionless measure that quantifies how much energy is applied to a system of a specified mass M in order to move the system a specified distance D. That is,

$$\text{COT} = \frac{E}{MgD}, \qquad (6)$$

where E is the total energy consumption for traveling distance D, M is the total mass of the system, and g is the acceleration due to gravity. COT and it's variants have been used in a wide range of domains to compare energy-efficient motion of different robotic systems[76], vehicles[75], and animals[77]. We note that COT essentially measures energy spent per unit mass per unit distance, as the normalization factor g required to make this measure dimensionless is the same for all systems. We adapt this metric to more general RL tasks to measure energy per unit mass per unit reward instead of energy per unit mass per unit distance. That is we define a cost of work (COW) by,

$$\text{COW} = \frac{E}{Mgr}, \qquad (7)$$

where E is the energy spent, M is the mass, and r is the reward achieved. For most locomotion tasks like locomotion in FT/VT, patrol, obstacle, escape and incline where reward is proportional to the distance traveled; COW and COT are essentially the same albeit with different units. We measure energy as the absolute sum of all joint torques[78]. This definition was used to compute energy efficiency (with lower COW indicating higher energy efficiency) in both evolutionary environments (Fig. 4c) and test tasks (Fig. 3b–d).

Stability. Informally, passive stability is the ability to stand without falling and is achieved via mechanical design of the agent/robot. Dynamic stability is the ability to move without falling over and is achieved via control. Formally an agent is passively stable, when the center of mass is inside the support polygon and the polygon's area is greater than zero[79]. The support polygon is the convex hull of all of the agent's contact points with the ground. We measure passive stability by checking if the agent falls over without any control. The agent is initialized at the center of arena and we measure the position of the head at the beginning and after 400 time steps (a full episode is 1000 time steps). An agent is passively stable if the head position after 400 time steps is above 50% of original height. Note that we use the violation of this same condition for early termination of the episode (see Rewards). We use this notion of passive stability in Fig. 4b.

Beneficial mutations. We define a mutation to be beneficial if the difference between the child and parent fitness is above a certain threshold. Although any non-zero increase in fitness is a beneficial mutation, small changes in fitness acquired via RL might not be statistically meaningful, especially since during evolution the fitness is calculated using a single seed. Hence, we heuristically set the threshold as a minimum increase in the final average reward by 300 for FT and 100 for VT and MVT. Roughly these numbers correspond to the 75th percentile in the distribution of fitness increases across all mutations in a given environment. We use this definition of beneficial mutations in Fig. 2b.

Data availability
The configuration files necessary to reproduce the data used in this work have been made available on Github (https://github.com/agrimgupta/derl).

Code availability
We have released a Python implementation of DERL on GitHub (https://github.com/agrimgupta/derl). In addition, the repository also includes code for creating UNIMALS, generating evolutionary environments, and evaluation task suite. The code, setup instructions, and the configuration files needed for reproducing the results of the paper are available on GitHub (https://github.com/agrimgupta/derl).

References
1. Darwin, C. *On the Origin of Species by Means of Natural Selection*, Vol. 167 (John Murray, London, 1859).
2. Evans, S. D., Hughes, I. V., Gehling, J. G. & Droser, M. L. Discovery of the oldest bilaterian from the Ediacaran of south Australia. *Proc. Natl Acad. Sci. USA* **117**, 7845–7850 (2020).
3. Pfeifer, R. & Scheier, C. *Understanding Intelligence* (MIT Press, 2001).
4. Brooks, R. A. New approaches to robotics. *Science* **253**, 1227–1232 (1991).
5. Bongard, J. Why morphology matters. *Horiz. Evolut. Robot.* **6**, 125–152 (2014).
6. Brown, T. B. et al. Language models are few-shot learners. *Advances in Neural Information Processing Systems*. (eds Larochelle, H. Ranzato, M. Hadsell, R. Balcan, M. F. & Lin, H.) **33**, 1877–1901 (Curran Associates, Inc., 2020).
7. He, K., Zhang, X., Ren, S. & Sun, J. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 770–778 (2016).
8. Silver, D. et al. Mastering the game of go with deep neural networks and tree search. *Nature* **529**, 484–489 (2016).
9. Lipson, H. & Pollack, J. B. Automatic design and manufacture of robotic lifeforms. *Nature* **406**, 974–978 (2000).
10. Eiben, A. E., Kernbach, S. & Haasdijk, E. Embodied artificial evolution. *Evolut. Intell.* **5**, 261–272 (2012).
11. Sims, K. Evolving 3d morphology and behavior by competition. *Artif. Life* **1**, 353–372 (1994).
12. Jelisavcic, M., Glette, K., Haasdijk, E. & Eiben, A. Lamarckian evolution of simulated modular robots. *Front. Robot. AI* **6**, 9 (2019).
13. Auerbach, J. E. & Bongard, J. C. Environmental influence on the evolution of morphological complexity in machines. *PLoS Comput. Biol.* **10**, e1003399 (2014).
14. Auerbach, J. et al. Robogen: Robot generation through artificial evolution. in *Artificial Life Conference Proceedings* Vol. 14, 136–137 (MIT Press, 2014).
15. Wang, T., Zhou, Y., Fidler, S. & Ba, J. In *International Conference on Learning Representations* (2019).
16. Miras, K., De Carlo, M., Akhatou, S. & Eiben, A. E. In *Applications of Evolutionary Computation*, 86–99 (Springer International Publishing, 2020).
17. Liao, T. et al. In *2019 International Conference on Robotics and Automation (ICRA)*, 2488–2494 (2019).
18. Luck, K. S., Amor, H. B. & Calandra, R. Data-efficient co-adaptation of morphology and behaviour with deep reinforcement learning. in *Conference on Robot Learning*, 854–869 (PMLR, 2020).
19. Schaff, C., Yunis, D., Chakrabarti, A. & Walter, M. R. Jointly learning to construct and control agents using deep reinforcement learning. In *2019 International Conference on Robotics and Automation (ICRA)*, 9798–9805 (IEEE, 2019).
20. Ha, D. Reinforcement learning for improving agent design. *Artif. Life* **25**, 352–365 (2019).
21. Zhao, A. et al. Robogrammar: graph grammar for terrain-optimized robot design. *ACM Trans. Graph.* **39**, 1–16 (2020).
22. Cheney, N., MacCurdy, R., Clune, J. & Lipson, H. Unshackling evolution: evolving soft robots with multiple materials and a powerful generative encoding. *SIGEVOlution* **7**, 11–23 (2014).
23. Cheney, N., Bongard, J., SunSpiral, V. & Lipson, H. Scalable co-optimization of morphology and control in embodied machines. *J. R. Soc. Interface* **15**, 20170937 (2018).
24. Kriegman, S., Cheney, N. & Bongard, J. How morphological development can guide evolution. *Sci. Rep.* **8**, 1–10 (2018).
25. Kaplan, J. et al. Scaling laws for neural language models. Preprint at https://arxiv.org/abs/2001.08361 (2020).
26. Henighan, T. et al. Scaling laws for autoregressive generative modeling. Preprint at https://arxiv.org/abs/2010.14701 (2020).
27. Chen, T., Kornblith, S., Swersky, K., Norouzi, M. & Hinton, G. E. Big self-supervised models are strong semi-supervised learners. in *Advances in Neural Information Processing Systems* 33 (2020).
28. Alba, E. *Parallel Metaheuristics: A New Class of Algorithms*. (Wiley-Interscience, 2005).
29. Syswerda, G. In *Foundations of Genetic Algorithms* Vol. 1, 94–101 (Elsevier, 1991).
30. Real, E. et al. Large-scale evolution of image classifiers. In *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, ICML'17, 2902–2911 (JMLR.org, 2017).
31. Real, E., Aggarwal, A., Huang, Y. & Le, Q. V. Regularized evolution for image classifier architecture search. In *Proceedings of the AAAI Conference on Artificial Intelligence* Vol. 33, 4780–4789 (2019).
32. Zoph, B. & Le, Q. V. In *International Conference on Learning Representations* (2017).

33. Alba, E. & Tomassini, M. Parallelism and evolutionary algorithms. *IEEE Trans. Evolut. Comput.* **6**, 443–462 (2002).

34. Sutton, R. S. & Barto, A. G. *Reinforcement Learning: An Introduction* (MIT Press, 2018).

35. Goldberg, D. E. & Deb, K. In *Foundations of Genetic Algorithms* Vol. 1, 69–93 (Elsevier, 1991).

36. Schulman, J., Wolski, F., Dhariwal, P., Radford, A. & Klimov, O. Proximal policy optimization algorithms. Preprint at https://arxiv.org/abs/1707.06347 (2017).

37. Kriegman, S. et al. Automated shapeshifting for function recovery in damaged robots. in *Proceedings of Robotics: Science and Systems* http://www.roboticsproceedings.org/rss15/p28.pdf (2019).

38. Hiller, J. & Lipson, H. Dynamic simulation of soft multimaterial 3d-printed objects. *Soft Robot.* **1**, 88–101 (2014).

39. Medvet, E., Bartoli, A., De Lorenzo, A. & Seriani, S. 2D-VSR-Sim: a simulation tool for the optimization of 2-D voxel-based soft robots. *SoftwareX* **12**, 100573 (2020).

40. Rus, D. & Tolley, M. T. Design, fabrication and control of soft robots. *Nature* **521**, 467–475 (2015).

41. Todorov, E., Erez, T. & Tassa, Y. Mujoco: a physics engine for model-based control. in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 5026–5033 (IEEE, 2012).

42. Collins, J., Chand, S., Vanderkop, A. & Howard, D. *A Review of Physics Simulators for Robotic Applications* (IEEE Access, 2021).

43. Heess, N. et al. Emergence of locomotion behaviours in rich environments. Preprint at https://arxiv.org/abs/1707.02286 (2017).

44. Muller, H. J. Some genetic aspects of sex. *Am. Naturalist* **66**, 118–138 (1932).

45. Auerbach, J. E. & Bongard, J. C. On the relationship between environmental and morphological complexity in evolved robots. In *Proceedings of the 14th Annual Conference on Genetic and Evolutionary Computation*, 521–528 (2012).

46. Miras, K., Ferrante, E. & Eiben, A. Environmental influences on evolvable robots. *PLoS ONE* **15**, e0233848 (2020).

47. Pratt, L. Y., Mostow, J., Kamm, C. A. & Kamm, A. A. Direct transfer of learned information among neural networks. in *AAAI* Vol. 91, 584–589 (1991).

48. Chen, T., Kornblith, S., Norouzi, M. & Hinton, G. A simple framework for contrastive learning of visual representations. (eds In III, H. D. & Singh, A.) *Proceedings of the 37th International Conference on Machine Learning*, Vol. 119 of *Proceedings of Machine Learning Research*, 1597–1607 (PMLR, Virtual, 2020).

49. He, K., Fan, H., Wu, Y., Xie, S. & Girshick, R. Momentum contrast for unsupervised visual representation learning. in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 9729–9738 (2020).

50. Weismann, A. *The Germ-plasm: A Theory of Heredity* (Scribner's, 1893).

51. Mark, B. J. A new factor in evolution. *Am. Naturalist* **30**, 441–451 (1896).

52. Turney, P. D. In *ICML Workshop on Evolutionary Computation and Machine Learning*, 135–142 (1996).

53. Mayley, G. Landscapes, learning costs, and genetic assimilation. *Evolut. Comput.* **4**, 213–234 (1996).

54. Hinton, G. E. & Nowlan, S. J. How learning can guide evolution. *Complex Syst.* **1**, 495–502 (1987).

55. Ackley, D. & Littman, M. Interactions between learning and evolution. *Artif. life II* **10**, 487–509 (1991).

56. Anderson, R. W. Learning and evolution: a quantitative genetics approach. *J. Theor. Biol.* **175**, 89–101 (1995).

57. Waddington, C. H. Canalization of development and the inheritance of acquired characters. *Nature* **150**, 563–565 (1942).

58. Deacon, T. W. *The Symbolic Species: the Co-evolution of Language and the Brain* 202 (WW Norton & Company, 1998).

59. Giudice, M. D., Manera, V. & Keysers, C. Programmed to learn? the ontogeny of mirror neurons. *Developmental Sci.* **12**, 350–363 (2009).

60. Hornby, G. S. Alps: The age-layered population structure for reducing the problem of premature convergence. in *Proceedings of the 8th Annual Conference on Genetic and Evolutionary Computation*, GECCO '06, 815–822 (Association for Computing Machinery, 2006).

61. Stanley, K. O. Compositional pattern producing networks: a novel abstraction of development. *Genet. Program. Evol. Mach.* **8**, 131–162 (2007).

62. Bongard, J. C. & Paul, C. in *From Animals to Animats: The Sixth International Conference on the Simulation of Adaptive Behaviour* (Citeseer, 2000).

63. Tassa, Y. et al. dm_control: software and tasks for continuous control. Preprint at https://arxiv.org/abs/2006.12983 (2020).

64. Brockman, G. et al. OpenAI Gym. Preprint at https://arxiv.org/abs/1606.01540 (2016).

65. OpenAI et al. Solving Rubik's cube with a robot hand. Preprint at https://arxiv.org/abs/1910.07113 (2019).

66. Li, Z. et al. Reinforcement learning for robust parameterized locomotion control of bipedal robots. Preprint at https://arxiv.org/abs/2103.14295 (2021).

67. Peng, X. B., Andrychowicz, M., Zaremba, W. & Abbeel, P. Sim-to-real transfer of robotic control with dynamics randomization. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, 3803–3810 (2018).

68. Wang, T., Liao, R., Ba, J. & Fidler, S. In *International Conference on Learning Representations* (2018).

69. Pathak, D., Lu, C., Darrell, T., Isola, P. & Efros, A. A. In *Advances in Neural Information Processing Systems* (eds Wallach, H. et al.) Vol. 32 (Curran Associates, Inc., 2019).

70. Huang, W., Mordatch, I. & Pathak, D. One policy to control them all: shared modular policies for agent-agnostic control. In *International Conference on Machine Learning*, 4455–4464 (PMLR, 2020).

71. Williams, R. J. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Mach. Learn.* **8**, 229–256 (1992).

72. Schulman, J., Moritz, P., Levine, S., Jordan, M. & Abbeel, P. In *Proceedings of the International Conference on Learning Representations (ICLR)* (2016).

73. Kostrikov, I. Pytorch implementations of reinforcement learning algorithms. https://github.com/ikostrikov/pytorch-a2c-ppo-acktr-gail (2018).

74. Henderson, P. et al. In *Thirty-Second AAAI Conference on Artificial Intelligence* (2018).

75. Von Karman, T. & Gabrielli, G. What price speed? specific power required for propulsion of vehicles. *Mech. Eng.* **72**, 775–781 (1950).

76. Siciliano, B. & Khatib, O. In *Springer Handbooks* (2016).

77. Alexander, R. M. Models and the scaling of energy costs for locomotion. *J. Exp. Biol.* **208**, 1645–1652 (2005).

78. Yu, W., Turk, G. & Liu, C. K. Learning symmetric and low-energy locomotion. *ACM Trans. Graph.* **37**, https://doi.org/10.1145/3197517.3201397 (2018).

79. McGhee, R. B. & Frank, A. A. On the stability properties of quadruped creeping gaits. *Math. Biosci.* **3**, 331–351 (1968).

Acknowledgements

Author contributions

A.G., S.G., and L.F.-F. conceptualized the problem and designed the framework. A.G. designed and built the computational framework, design space and evaluation framework. A.G. ran experiments and analyzed data. A.G., S.G., and L.F.-F. wrote the paper with contributions from S.S.

Competing interests

The authors declare no competing interests.

Additional information

Supplementary information The online version contains supplementary material available at https://doi.org/10.1038/s41467-021-25874-z.

Correspondence and requests for materials should be addressed to Agrim Gupta or Li Fei-Fei.

Peer review information *Nature Communications* thanks Eliseo Ferrante, and the other, anonymous, reviewer(s) for their contribution to the peer review of this work.

Reprints and permission information is available at http://www.nature.com/reprints

Publisher's note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.