Evolving Gaits for Physical Robots with the HyperNEAT Generative Encoding: The Benefits of Simulation

Suchan Lee[1], Jason Yosinski[1], Kyrre Glette[2], Hod Lipson[1], and Jeff Clune[1,3]

[1] Cornell University, USA
[2] University of Oslo, Norway
[3] University of Wyoming, USA
{sl746,jy495,hod.lipson,jeffclune}@cornell.edu, kyrrehg@ifi.uio.no

Abstract. Creating gaits for physical robots is a longstanding and open challenge. Recently, the HyperNEAT generative encoding was shown to automatically discover a variety of gait regularities, producing fast, coordinated gaits, but only for simulated robots. A follow-up study found that HyperNEAT did not produce impressive gaits when they were evolved directly on a physical robot. A simpler encoding hand-tuned to produce regular gaits was tried on the same robot, and outperformed HyperNEAT, but these gaits were first evolved in simulation before being transferred to the robot. In this paper, we tested the hypothesis that the beneficial properties of HyperNEAT would outperform the simpler encoding if HyperNEAT gaits are first evolved in simulation before being transferred to reality. That hypothesis was confirmed, resulting in the fastest gaits yet observed for this robot, including those produced by nine different algorithms from three previous papers describing gait-generating techniques for this robot. This result is important because it confirms that the early promise shown by generative encodings, specifically HyperNEAT, are not limited to simulation, but work on challenging real-world engineering challenges such as evolving gaits for real robots.

1 Introduction

Legged robots can operate in a much wider range of environments than their wheeled counterparts. However, designing gaits for legged robots is a difficult and time-consuming process for human engineers [16, 24], and must be repeated every time a robot is created or modified [11]. Scientists thus investigate how to automatically produce gaits via machine learning and evolutionary algorithms, and the result is often a better gait than those created by human engineers [10–12, 23, 25]. While it has been shown that gaits perform better if they are *regular*—i.e., that they have coordinated movements, such as left-right symmetry or front-back symmetry [4, 6, 7, 23]—experimenters usually have to explicitly decide and specify these regularities [1, 15, 23, 22]. Such manual intervention is time consuming, requires expert knowledge, and adds constraints that may hurt performance.

A.I. Esparcia-Alcázar et al. (Eds.): EvoApplications 2013, LNCS 7835, pp. 540–549, 2013.

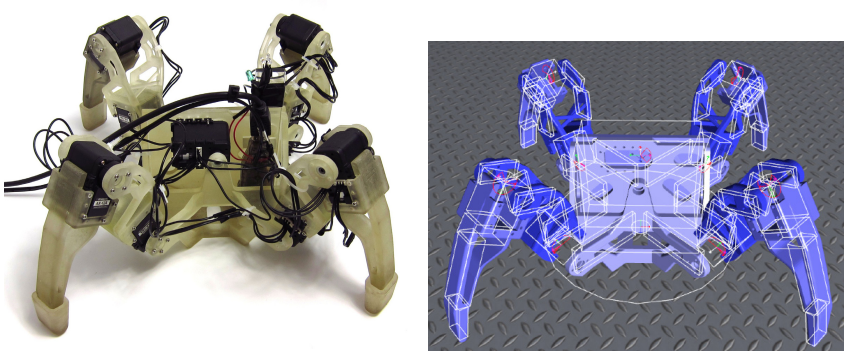

Fig. 1. The QuadraTot robot platform on which gaits were evolved. **Left:** The physical robot, which is composed of 3-D printable and off-the-shelf components. **Right:** The representation of the robot in the simulator.

Previous work has shown that the Hypercube-based NeuroEvolution of Augmenting Topologies (HyperNEAT) generative encoding [20] can automatically generate a variety of regular gaits that outperform gaits evolved with direct encodings [4, 7]. However, that work only verified these claims in simulation. Yosinski et al. evolved gaits with HyperNEAT directly in hardware on the QuadraTot robot platform (Figure 1). They found that HyperNEAT's gaits outperformed manually designed, parameterized learning algorithms, but still did not produce impressive, natural gaits [25].[1] A follow-up study built a simulator for QuadraTot to test whether the inclusion of a simulator would improve results and found that it did: when gaits were evolved in simulation with a simple direct encoding and then transferred to the QuadraTot robot, the resulting gaits were faster than the gaits produced by evolving gaits with HyperNEAT directly on the robot [9]. The simulator helped because it afforded much larger population sizes and more generations than were possible when evolving directly in hardware, resulting in 333 times more evaluations per run (60000 vs. 180) [9, 25].

The work with the simulator [9] evolved gaits with a simple encoding manually constrained to produce specific regularities. The success of that work raises the question of whether the performance gains were due to the added benefit of a simulator or the use of a simple, hand-designed encoding. We hypothesized that HyperNEAT, which has been previously shown to automatically discover complex regularities to produce high-performing gaits [7, 25], would outperform the simpler encoding from Glette et al. if combined with a simulator. Here we test that hypothesis by evolving gaits with HyperNEAT in the same simulator from Glette et al. and then transferring those gaits to the QuadraTot robot. Our experiments confirmed the hypothesis: with the simulator, HyperNEAT evolved the highest-performing gaits observed to date for the QuadraTot platform.

[1] Videos available at `http://creativemachines.cornell.edu/evolved-quadruped-gaits`

2 Methods

Robot Hardware. We performed experiments on the QuadraTot quadrupedal robot platform (Figure 1-Left) [25]. It has 9 degrees of freedom: two joints per leg and one joint that rotates along the robot's midline. The QuadraTot hardware designs and the software for this project are open source[2], and all hardware components are either off-the-shelf or 3D-printed. There are results on the platform for nine different learning algorithms from three previous publications [9, 18, 25].

The joints are powered by Robotis Dynamixel servos; five AX-18A servos for the inner joints of each leg and the single midline joint, and four AX-12A servos for the the outer joints of each leg, which require less power and can thus have less expensive motors. Each servo has a built-in safety mechanism that shuts itself off to prevent damage if the servo's current, range, temperature, or torque is too high. This safety mechanism activated frequently and inconsistently, adding significant noise to the evaluation process. As pointed out in a previous study [25], evolved gaits on QuadraTot are highly variable and produce many shutdowns because they force the servos to exert too much torque. To prevent collisions between different pieces of the robot's body, we limited the allowable range of movement for the inner, outer, and hip joints to [-85°, +60°], [-113°, +39°], and [-28°, +28°], respectively. We also implemented the Smart Cropping System from Shen et al. [18], which prevents combinations of joint positions for the inner and outer joint of each leg that generate extreme amounts of torque. A final method of reducing torque was to reduce the weight of the robot. Yosinski et al. had the small Linux computer that performed all computation on the robot, but we removed it and sent commands from it to the robot via a cable. We tracked the robot's position using an infrared LED observed by a Wiimote.

Simulator. Gaits evolved in the simulator from Glette et al.[2], which represents QuadraTot in the Nvidia PhysX physics engine, including the mass and size of the QuadraTot components and its degrees of freedom (Figure 1-Right). In the simulator, each individual joint range was limit as described above, but Smart Cropping was not included because we found that it hindered performance by limiting the types of gaits evolution could explore in early generations.

HyperNEAT. HyperNEAT is an algorithm for evolving artificial neural networks (ANNs) [20]. It has been repeatedly described in detail [7, 8, 20], so here we provide only a summary. Instead of directly encoding each ANN weight individually on the genome, in HyperNEAT the genome is a compositional pattern producing network (CPPN) [19]. The CPPN specifies the weights in a similar way to how natural organisms develop. In nature, phenotypic attributes are specified as a function of their geometric location, and such positional information is conveyed through chemical morphogen gradients [3]. For example, the concentration of one chemical could indicate the position along the head-to-tail axis and

[2] A parts list, hardware CAD files, software (including the simulator), and gait videos are available at `http://creativemachines.cornell.edu/evolved-quadruped-gaits`

another chemical in bands could indicate if a cell is in an odd- or even-numbered segment. Based on the relative concentrations of these chemicals, a cell can know where it is geometrically and, thus, what type of cell to become [3].

With CPPNs this process is abstracted as a network of math functions that operate in a Cartesian geometric space. The coordinates of phenotypic elements are provided as inputs to the CPPN and the outputs specify phenotypic traits. For example, when CPPNs encode 2D pictures, the coordinates of each pixel are iteratively input into the genome and the output is the grayscale value at that coordinate [17]. Because a CPPN network is composed of math functions, these functions can create geometric regularities in the phenotype. For example, a Gaussian function of an axis can provide symmetry (e.g. left-right), and a repeating function (e.g. sine) of an axis could provide repetition (e.g segmentation). Both 2D pictures and 3D objects evolved with CPPNs look like natural and engineered objects, and contain complex regularities, such as symmetries and repeated motifs, with and without variation [5, 17].

In HyperNEAT, CPPNs encode the weights of the connections between neurons as a function of the geometric locations of those neurons (Figure 2). The Cartesian coordinates of the two neurons at the end of each connection are input into the CPPN, and the output is the weight of that connection. If the output is smaller than a threshold, the weight is set to zero, functionally removing the connection. The process is repeated for each possible connection. Just as in 2D pictures and 3D objects, the CPPN can create complex, regular, geometric patterns (e.g. left-right symmetry or repeated modules), but in this case the patterns are in the weights of a neural network [7]. The neural regularities produced by HyperNEAT enable significantly improved performance on problems that are regular [7, 20], including evolving quadruped gaits in simulation [4, 6, 7].

In HyperNEAT, the CPPN genomes evolve via the NeuroEvolution of Augmenting Topologies (NEAT) algorithm [21], which has three major components. First, NEAT starts with small genomes that encode simple networks and complexifies them via mutations that add nodes and links to the networks. Second, NEAT has a fitness-sharing system that preserves diversity and allows for new innovations to be tuned by evolution before competing them against more adapted rivals. Third, historical information is recorded that facilitates crossover in a way that is effective, yet avoids the need for expensive topological analysis. A full explanation of NEAT can be found in Stanley and Miikkulainen 2002 [21].

In this study, the ANN inputs, outputs, activation functions, and the size of the hidden layer are the same as in Yosinski et al. [25]. The ANN had a fixed topology of three 3×4 Cartesian grids of nodes for the input, hidden, and output layers. The inputs to the substrate were the angles requested in the previous time step for each of the 9 joints of the robot and a sine and cosine wave to facilitate periodic motion. The outputs of the substrate at each time step were nine numbers (for each joint) in the range [-1, 1] which were scaled to the allowable ranges for the servos. As in Yosinski et al. [25], we generated pseudo-positions at 160Hz and then downsampled over consecutive blocks of four time steps to obtain the actual commanded positions at 40Hz; this reduced the

number of gaits which commanded switches from extreme negative to extreme positive numbers at 40Hz, which overly taxed the servos.

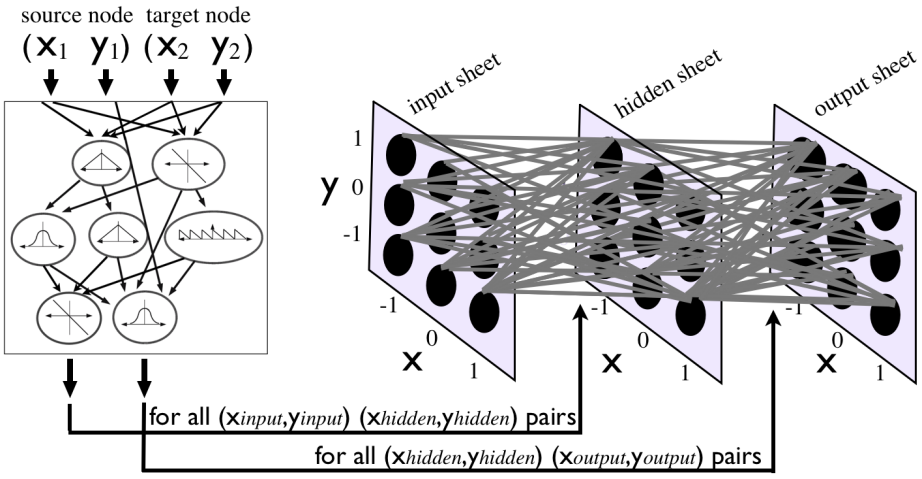

Fig. 2. A CPPN specifying a neural network. In HyperNEAT, weights are a function of the Cartesian coordinates of the source and target node for each connection. All pairwise combinations of source and target node coordinates are iteratively passed into a CPPN to determine the weight of each ANN link. Figure from Clune et al. [7].

Evolutionary Process and Parameters: Each run had a population size of 200 and lasted 200 generations. We performed 20 HyperNEAT runs that differed only in the seed provided to the random number generator, which affected stochastic events such as mutation. To make a statistical comparison to the encoding from Glette et al., we conducted 19 runs using the original code to supplement the one run performed for that paper. Each gait was evaluated for fourteen seconds in reality, with interpolation from and to a stationary position in the first and last two seconds, respectively, as in Yosinski et al., effectively resulting in 12 seconds of full-speed motion. In simulation gaits were evaluated for 12 seconds. All HyperNEAT parameters were identical to those in Yosinski et al. except for the frequency of the sine wave input to the ANN, which was lowered from 4.2Hz to 0.64Hz to reduce servo shutdowns. To further reduce servo shutdowns, we punished high-frequency gaits during evolution. We calculated the frequency of a gait as the average number of servo direction changes per leg per second. If this frequency was higher than the experimentally-selected threshold of 1.67 Hz, the measure of distance traveled by the center of mass during the gait was reduced exponentially by multiplying it by a discount factor of $e^{(1.67-\text{freq})}$. Following Clune et al. [4], the fitness equation was 2^{distance^2}. After evolving the gaits in simulation, the champion gait of the last generation of each of the 20 runs was transferred onto the real robot and the distance traveled was measured.

3 Results and Discussion

The simulator enabled HyperNEAT to evolve fast, natural gaits. In simulation, HyperNEAT gaits were faster than those from Glette et al. (Figure 3-Top, $p < 6.8 \times 10^{-8}$ when comparing the best gaits in the final generation of each run via Matlab's Wilcoxon rank sum test). Specifically, HyperNEAT gaits were 52.1% faster in simulation (25.4 cm/s $\pm$ 3.4 SD versus 16.7 cm/s $\pm$ 1.9 SD). To facilitate comparisons to earlier works [9, 25] we report mean $\pm$ SD, but our qualitative conclusions are the same when using medians. Plots of servo positions over time reveal that the evolved HyperNEAT gaits are regular and coordinated (Figure 3-Left), confirming previous results with HyperNEAT in simulation [4, 7].

On the physical robot, HyperNEAT gaits from this study outperformed gaits from all previous QuadraTot studies (Table 1) [9, 18, 25], including those of Glette et al. However, comparing performance in hardware between studies performed in different laboratories is difficult, not only because reality is inherently noisy, but because even two copies of the same robot are not identical and may produce different speeds for the same input gait. Gaits for this study and two previous studies [18, 25] were evaluated on a copy of the QuadraTot robot in the Cornell Creative Machines Lab (CCML), while the gaits in Glette et al. were evaluated on a different copy of the same robot in the Robotics and Intelligent Systems (ROBIN) lab at the University of Oslo. The two robots were evaluated on two different surfaces, and had different material enveloping their feet to increase friction (compare Figure 1-Left to Figure 2 of Glette et al).

The previous fastest gait on any copy of a QuadraTot was from Glette et al., and traveled 17.8 cm/s on the ROBIN QuadraTot. The fastest gait produced by HyperNEAT with a simulator in the experiments for this paper traveled 14.5 cm/s on the CCML QuadraTot. It was unclear whether this difference in performance was due to the differences in the gaits themselves or dissimilarities between hardware. To control for this possibility, we ran the fastest gait from Glette et al. 10 times on CCML and measured a mean speed of only 12.95 cm/s $\pm$0.93 SD (vs. 17.8 cm/s measured on ROBIN) and a maximum of 13.8 cm/s (Table 1). It thus appears that the CCML version of the robot is slower. Moreover, the HyperNEAT gait (14.5 cm/s) is faster than the best gait of Glette et al. on the CCML QuadraTot. Unfortunately, it was not possible to test the best HyperNEAT gait on the ROBIN QuadraTot. Because HyperNEAT outperformed the simple encoding from Glette et al. on the same robotic hardware, we tentatively conclude that HyperNEAT produces faster gaits for the QuadraTot robot. This conclusion is supported by the fact that HyperNEAT also outperformed the encoding from Glette et al. in simulation.

We now discuss the differences between the gaits evolved by HyperNEAT directly on the physical robot [25], and those first evolved in a simulator and then transferred to the robot (this study). On the physical robot, the gaits produced by HyperNEAT with a simulator were faster, more natural, and more repeatable than those evolved directly on the QuadraTot robot [25]. The gaits were also more regular, as they were in simulation (Figure 3-Left). This result is important because it confirms that HyperNEAT can produce the important property of

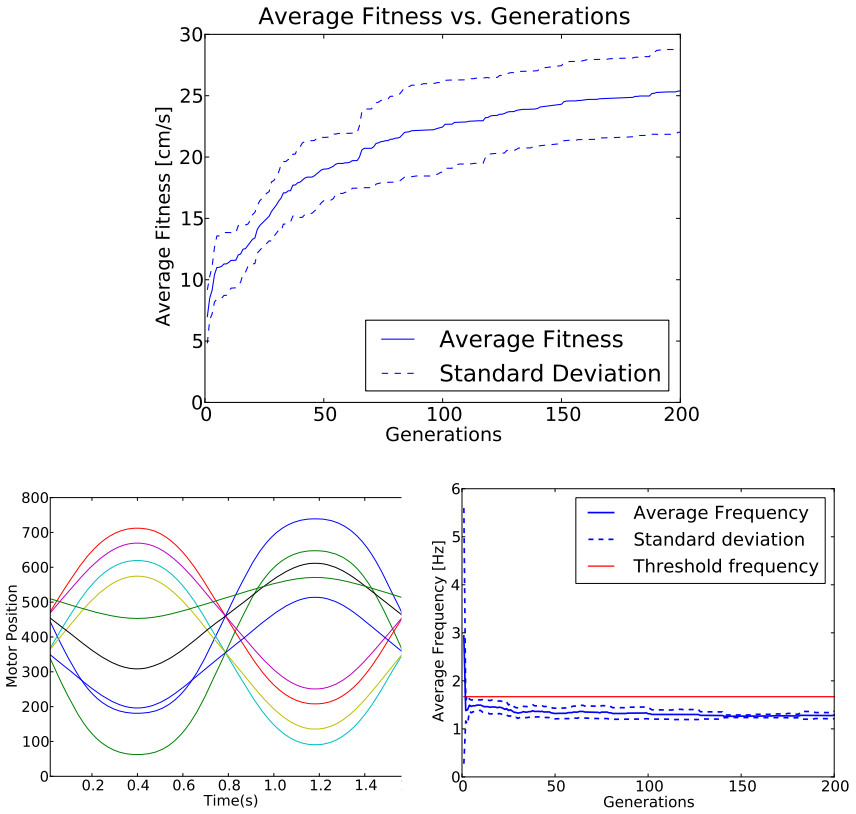

Fig. 3. Top: HyperNEAT outperforms a genetic algorithm with a simple encoding [9] when both algorithms are combined with a simulator. Plotted are means over 20 runs in simulation (solid lines) ± SD (dashed lines). HyperNEAT gaits are 52.1% faster in simulation and 5.1% faster in reality than those from a previous study [9] (details in Table 1). **Left:** Servo positions over time (for nine servos) for a representative simulated HyperNEAT gait. HyperNEAT produced smooth and symmetrical gaits that contained complex regularities. **Right:** Mean gait frequency averaged over 20 runs. Gaits with frequencies above a threshold (horizontal line) receive a fitness penalty. HyperNEAT quickly learned to produce gaits with frequencies low enough to avoid this penalty.

regularity in a challenging, real-world domain, which was not previously observed when evolving directly on the hardware [25]. Producing regular solutions is a key to exploiting regularity in difficult engineering problems [7].

The simulator likely improved performance because of the number of evaluations it enabled, both in terms of the population size (200 vs. 9) and the number of generations (200 vs. 20), leading to a total difference of 40000 vs. 180 per evolutionary run compared to Yosinski et al. [25]. Another potential cause of improved performance is the lower noise in the simulator, which could have helped HyperNEAT find coordinated, regular, gaits, which perform better. On the physical

Table 1. Velocities of evolved gaits in simulation and on two different copies of the QuadraTot robot. Subject to availability, data are reported from the experiments for this paper and three previous studies. Reported are the total number of evaluations per run, the mean of the fastest gaits produced in each run in simulation, and the single fastest gait produced on the CCML and ROBIN copies of the QuadraTot robot (see text for their differences). *Instead of using the single fitness value reported in [9], we ran 19 additional runs and used the mean fitness of those 20 runs. Velocities are in cm/s, and bold indicates the best performance. **The median fitness that corresponds with this mean is 26.9 cm/s, 95% confidence interval [23.8 cm/s, 26.75 cm/s].

	Evaluations	Simulated Velocity	Real Vel. (CCML)	Real Vel. (ROBIN)
Parameterized gaits + optimization [25]	153	–	5.8	–
HyperNEAT in hardware [25]	180	–	9.7	–
RL PoWER Spline [18]	300	–	11.1	–
GA + simulator [9]	60000	*16.7	13.8	**17.8**
HyperNEAT + simulator [this paper]	40000	****25.4**	**14.5**	–

robot, the noise in the evaluation was substantial, preventing effective learning [25]. To investigate this hypothesis, we performed 20 runs in simulation with only 180 fitness evaluations, which was the number used in Yosinski et al. [25]. The simulated gaits performed slightly, but not significantly, better than those evolved in hardware ($p = 0.1571$, mean fitness 7.9 ± 2.14 cm/s). Reduced noise thus may have had a small affect on performance, but the substantial performance gains that resulted from using a simulator likely came from the additional evaluations the simulator afforded. The encouragement of low-frequency gaits in this study also may have aided performance, especially since in simulation the gaits were high-frequency in a few early generations before rapidly settling to a range below the penalized threshold (Figure 3-Right).

While this study was able to produce the fastest QuadraTot gait to date, most of the gaits in simulation did not transfer well to reality. Many gaits that were fast in simulation performed poorly on the real robot, largely due to servos that were too weak and shut down, or because of differences between simulation and reality. Repeated attempts to minimize these problems were unsuccessful. In future studies we will use a robot that has more mechanical advantage and requires less torque from each servo, such as the Aracna platform [14]. That we did not model the servos in simulation, especially with their frequent failures, suggests that even better results could be obtained via a simulator that contained or learned servo models. In future work we will also incorporate techniques to minimize the gap between the simulator and reality [2, 13, 26].

4 Conclusion

With a simulator, HyperNEAT evolved the fasted gait yet recorded for the QuadraTot robot, outperforming nine machine learning algorithms from three

previous publications [9, 18, 25], including an improvement of 52.1% in simulation and 5.1% in reality over the previous best QuadraTot gait by Glette et al. These results provide an important demonstration that the HyperNEAT generative encoding can evolve state-of-the-art results for challenging engineering problems, in this case evolving gaits for a legged robot. The results further reaffirm the benefits of using simulators when solving real-world challenges with evolutionary algorithms. Our results additionally confirm—with a different robot and simulator—previous work that has shown that HyperNEAT is an effective encoding for automatically evolving coordinated, regular gaits in simulation [4, 7, 25]. That HyperNEAT outperformed the encoding hand-designed by Glette et al. shows that HyperNEAT can outperform even evolutionary algorithms manually designed to incorporate human domain knowledge regarding which regularities are thought to be beneficial for a problem. The results thus demonstrate that automatically discovering regularities can be a superior approach to specifying them, even on problems that are relatively well-understood.

References

1. Beer, R., Gallagher, J.: Evolving dynamical neural networks for adaptive behavior. Adaptive Behavior 1(1), 91–122 (1992)
2. Bongard, J.C.: Synthesizing Physically-Realistic Environmental Models from Robot Exploration. In: Almeida e Costa, F., Rocha, L.M., Costa, E., Harvey, I., Coutinho, A. (eds.) ECAL 2007. LNCS (LNAI), vol. 4648, pp. 806–815. Springer, Heidelberg (2007)
3. Carroll, S.: Endless Forms Most Beautiful: The New Science of Evo Devo and the Making of the Animal Kingdom. Norton, New York (2005)
4. Clune, J., Beckmann, B., Ofria, C., Pennock, R.: Evolving coordinated quadruped gaits with the HyperNEAT generative encoding. In: Proceedings of the IEEE Congress on Evolutionary Computation, pp. 2764–2771 (2009)
5. Clune, J., Lipson, H.: Evolving three-dimensional objects with a generative encoding inspired by developmental biology. In: Proceedings of the European Conference on Artificial Life, pp. 144–148 (2011)
6. Clune, J., Ofria, C., Pennock, R.: The sensitivity of HyperNEAT to different geoemtric representations of a problem. In: Proceedings of the Genetic and Evolutionary Computation Conference (GECCO), pp. 2764–2771 (2009)
7. Clune, J., Stanley, K.O., Pennock, R., Ofria, C.: On the performance of indirect encoding across the contiuum of regularity. IEEE Transactions on Evolutioanry Computation 15, 346–367 (2011)
8. Gauci, J., Stanley, K.: Generating large-scale neural networks through discovering geometric regularities. In: Proceedings of the Genetic and Evolutionary Computation Conference, pp. 997–1004. ACM (2007)
9. Glette, K., Klaus, G., Zagal, J., Torresen, J.: Evolution of locomotion in a simulated quadruped robot and transferral to reality. In: Proceedings of the Seventeenth International Symposium on Artificial Life and Robotics (2012)
10. Hornby, G., Lipson, H., Pollack, J.B.: Generative representations for the automated design of modular physical robots. IEEE Transactions on Robotics and Automation 19, 703–719 (2003)

11. Hornby, G., Takamura, S., Tamamoto, T., Fujita, M.: Autonomous evolution of dynamic gaits with two quadruped robots. IEEE Transactions on Robotics 21(3), 402–410 (2005)
12. Kohl, N., Stone, P.: Machine learning for fast quadrupedal motion. In: The Nineteenth National Conference on Articifial Intelligence (AAAI), pp. 611–616 (2004)
13. Koos, S., Mouret, J., Doncieux, S.: The transferability approach: Crossing the reality gap in evolutionary robotics. IEEE Trans. Evolutionary Computation 1, 1–25 (2012)
14. Lohmann, S., Yosinksi, J., Gold, E., Clune, J., Blum, J., Lipson, H.: Aracna: An open-source quadruped platform for evolutionary robotics. In: Proceedings of the 13th International Conference on the Synthesis and Simulation of Living Systems, pp. 387–392 (2012)
15. Raibert, M., Chepponis, M., Brown Jr., H.: Running on four legs as though they were one. IEEE Journal of Robotics and Automation 2(2), 70–82 (1986)
16. Ridderstrom, C.: Legged locomotion control–a literature survey. In: Tech Report: Royal Institute of Technology. pp. 1400–1179. No. TRITA-MMK, Stockholm, Sweden (1999)
17. Secretan, J., Beato, N., D'Ambrosio, D., Rodriguez, A., Campbell, A., Folsom-Kovarik, J., Stanley, K.: Picbreeder: A Case Study in Collaborative Evolutionary Exploration of Design Space. Evolutionary Computation 19(3), 373–403 (2011)
18. Shen, H., Yosinski, J., Kormushev, P., Caldwell, D.G., Lipson, H.: Learning fast quadruped robot gaits with the RL power spline parameterization. In: AIMSA Workshop on Advances in Robot Learning and Human-Robot Interaction (2012)
19. Stanley, K.O.: Compositional pattern producing networks: A novel abstraction of development. Genetic Programming and Evolvable Matter 8(2), 131–152 (2007)
20. Stanley, K.O., D'Ambrosio, D.B., Gauci, J.: A hypercube-based encoding for evolving large-scale neural networks. Artificial Life 15(2), 185–212 (2009)
21. Stanley, K.O., Miikkulainen, R.: Evolving neural networks through augmenting topologies. Evolutionary Computation 10(2), 99–127 (2002)
22. Téllez, R.A., Angulo, C., Pardo, D.E.: Evolving the Walking Behaviour of a 12 DOF Quadruped Using a Distributed Neural Architecture. In: Ijspeert, A.J., Masuzawa, T., Kusumoto, S. (eds.) BioADIT 2006. LNCS, vol. 3853, pp. 5–19. Springer, Heidelberg (2006)
23. Valsalam, V., Miikkulainen, R.: Modular neuroevolution for multilegged locomotion. In: Proceedings of the Genetic and Evolutionary Computation Conference, pp. 265–272 (2008)
24. Wettergreen, D., Thorpe, C.: Gait generation for legged robots. In: IEEE International Conference on Intelligent Robots and Systems, pp. 1413–1420 (1992)
25. Yosinski, J., Clune, J., Hidalgo, D., Nguyen, S., Zagal, J., Lipson, H.: Evolving robot gaits in hardware: the HyperNEAT generative encoding vs. parameter optimization. In: Proceedings of the 20th European Conference on Artificial Life, pp. 11–18 (2011)
26. Zagal, J., Ruiz-del-Solar, J., Vallejos, P.: Back to reality: Crossing the reality gap in evolutionary robotics. In: Proceedings of IAV 2004, the 5th IFAC Symposium on Intelligent Autonomous Vehicles (2004)