VERSITA

BULGARIAN ACADEMY OF SCIENCES

CYBERNETICS AND INFORMATION TECHNOLOGIES • Volume **12**, No 3

Sofia • 2012

Print ISSN: 1311-9702; Online ISSN: 1314-4081
DOI: 10.2478/cait-2012-0022

Learning Fast Quadruped Robot Gaits with the RL PoWER Spline Parameterization

Haocheng Shen, Jason Yosinski, Petar Kormushev, Darwin G. Caldwell and Hod Lipson

Cornell University, 239 Upson Hall, Ithaca, NY 14853, USA
Department of Advanced Robotics, Istituto Italiano di Tecnologia, via Morego 30, 16163 Genova, Italy
Emails: hs454@cornell.edu yosinski@cs.cornell.edu Petar.Kormushev@iit.it

Abstract: *Legged robots are uniquely privileged over their wheeled counterparts in their potential to access rugged terrain. However, designing walking gaits by hand for legged robots is a difficult and time-consuming process, so we seek algorithms for learning such gaits to automatically using real world experimentation. Numerous previous studies have examined a variety of algorithms for learning gaits, using an assortment of different robots. It is often difficult to compare the algorithmic results from one study to the next, because the conditions and robots used vary. With this in mind, we have used an open-source, 3D printed quadruped robot called QuadraTot, so the results may be verified, and hopefully improved upon, by any group so desiring. Because many robots do not have accurate simulators, we test gait-learning algorithms entirely on the physical robot. Previous studies using the QuadraTot have compared parameterized splines, the HyperNEAT generative encoding and genetic algorithm. Among these, the research on the genetic algorithm was conducted by (G l e t t e et al., 2012) in a simulator and tested on a real robot. Here we compare these results to an algorithm called Policy learning by Weighting Exploration with the Returns, or RL PoWER. We report that this algorithm has learned the fastest gait through only physical experiments yet reported in the literature, 16.3% faster than reported for HyperNEAT. In addition, the learned gaits are less taxing on the robot and more repeatable than previous record-breaking gaits.*

Keywords: *Evolvable splines, parameterized gaits, HyperNEAT, machine learning, quadruped.*

1. Introduction

Various learning algorithms have been proved to be effective for legged robots. Algorithms, such as HyperNEAT (Y o s i n s k i et al. [12]), Genetic Algorithms (C h e r n o v a and V e l o s o [10]) and others (H o r n b y et al. [7]; Z y k o v et al. [13]; T e´ l l e z et al. [2]; V a l s a l a m and M i i k k u l a i n e n [11]) have been tested to be effective for automatic learning gaits for robots. Despite competitive performance, a major task is usually hidden in the published results: tuning the parameters for these evolutionary algorithms (K o r m u s h e v et al. [9]). Here we present results using a different way of learning gaits: a Reinforcement Learning algorithm called Policy learning by Weighting Exploration with the Returns, or RL PoWER, proposed by K o b e r and P e t e r s [8]. In our experiment, the main focus of the research is on the applicability of RL PoWER to quadruped robot gait learning. Another motivation is to compare the state-of-the-art neural network algorithm, HyperNEAT, with our proposed method in quadruped robot gait learning.

Fig. 1. The open source, 3D printed QuadraTot robot used in this research. The white printed booties are new additions to prevent sliding on surfaces and to minimize measurement error

2. Problem definition

As in Yo s i n s k i et al. [12], we define the gait learning problem to be the search for a gait that maximizes some specific metric. Mathematically, we define a gait as a function that specifies a vector of commanded motor positions for a robot over time. Gaits without a feedback – also called open-loop gaits, can be defined as

$$\boldsymbol{x} = g(t)$$

According to this definition, open-loop gaits are deterministic. One particular gait should behave exactly the same when it is run from a trial to a trial. However, the actual robot motion and fitness measured will vary due to the errors and uncertainty of the real world physics. In our trials, the gaits generated were sent to the robot and executed in an open loop manner. We will measure and analyze the performance between HyperNEAT and RL PoWER, the latter of which will be the focus of discussion in this paper. The metric used for the fitness in this paper will be described later.

3. Related work

Many attempts have been made on robot gaits learning using machine learning algorithms, producing competitive results (Chernova and Veloso [2]; Hornby et al. [7]; Zykov et al. [13]; Clune et al. [3, 4]; Tellez et al. [10]; Valsalam and Miikkulainen [11]). In fact, Jason et al. tested and compared six different algorithms for quadruped robot gait learning in their studies. HyperNEAT, a generative encoding neural network algorithm, achieved the best performance among all six methods. A following research done by Kyrre et al., using a tuned simulator, generated even more competitive results. In Kormushev et al. [9] bipedal robot energy reduction research, a reinforcement learning algorithm was used to optimize the performance. The results from Kormushev's research showed this algorithm's potential for walking problems. This reinforcement learning method still needs more tests for explorations.

4. Methods

4.1. Policy representation by splines

The simplest model with back-compatibility is geometric splines. For example, for a given model $f(x)$ with K knots, we can preserve the exact shape of the generated curve while adding extra knots to the original spline. If we put one additional knot between every two consecutive knots of the original spline, we end up with a $2K - 1$ knots and a spline that has the same shape as the original one. In order to do this, we need to define an algorithm for evolving the parameterization from K to L knots ($L > K$), which is formulated as in 1. Without loss of generality, the policy parameters are normalized into [0, 1], and appropriately scaled and shifted as necessary upon use.

Algorithm 1. EvolvePolicy-Spline ($P_{current}$: current policy, L: desired new number of parameters)

1: $K \leftarrow P_{current}$.numberOfParameters
2: $X_{current} \leftarrow [0, \frac{1}{K-1}, \frac{2}{K-1}, ..., 1]$
3: $Y_{current} \leftarrow P_{current}$.parameter values
4: $S_{current} \leftarrow$ ConstructSpline($X_{current}$, $Y_{current}$)
5: $X_{new} \leftarrow [0, \frac{1}{L-1}, \frac{2}{L-1}, ..., 1]$
6: $Y_{new} \leftarrow$ EvaluateSplineAtKnots($S_{current}$, X_{new})
7: $S_{new} \leftarrow$ ConstructSpline(X_{new}, Y_{new})
8: P_{new}.numberOfParameters $\leftarrow$ L
9: P_{new}.parameterValues $\leftarrow S_{new}.Y_{new}$
10: return P_{new}

4.2. Parameterized gaits by RL PoWER

Here we used an RL approach to change the complexity of the policy representation dynamically while the trial is running. In earlier studies on reducing energy consumption for bipedal robots (K o r m u s h e v et al. [9]), a mechanism that can evolve the policy parameterization was used. The method starts from a very simple parameterization and gradually increases its representational capability. The method was tested to be capable of generating an adaptive policy parameterization that can accommodate increasingly more complex policies. Presented in the studies of K o r m u s h e v et al. [9], the policy generated by this approach can reach the global optimum at a fast rate when applied to the energy reduction problem. Another property found about this method is its chance of converging to a suboptimal solution is reduced, because in the lower-dimensional representation this effect is less exhibited.

K o b e r and P e t e r s [8] proposed a RL algorithm named Policy learning by Weighting Exploration with the Returns (RL PoWER).

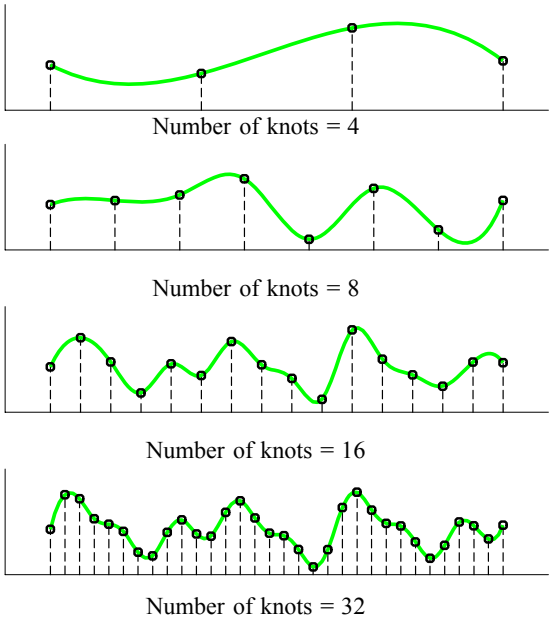

Fig. 2. An example for an evolving policy parameterization based on spline representation of the policy. The set of spline knots is the policy parameterization. The spline knots are the actual policy parameter values. This original parameterization starts from 4 knots and grows up to 32 knots

Maximization algorithm. The reason for using this is its relatively fewer parameters that need tuning. We evolved the policy parameterization only on those past trials ranked the highest by the importance sampling technique used by the PoWER algorithm. The intuition behind is that highly ranked parameterizations have more potential to evolve even better in the future. Besides, evolving all the parameterizations increases the exploring space. Since our experiment is done on a

physical robot, exploring all the variations of every parameterization is not practical. Future work may incorporate simulations into the studies, as illustrated in Bongard et al. [1].

For the experiment, we set the splines to have 3 knots for each servo, and there are 8 servos in total. The servo in the hip is not used in our experiment. Previous work has verified that quadruped gaits perform better when they are coordinated (Clune et al. [4]; Valsalam and Miikkulainen [11]). For each spline, we calculate its corresponding parameterized gait for one unit time cycle of 1.8 seconds and then apply the same pattern to every cycle throughout the 12 seconds of one trial. Specifically, each spline (a set of 3 knots) is interpreted to its corresponding servo positions as in the following equation and in Table 1.

$$g(t) = \begin{cases} R \cdot f(s1, s2, s3) + C \\ 0 + Cc \end{cases}. \quad (1)$$

Table 1. The RL PoWER motion model parameters

Parameters in θ	Description	Range
$f(s1, s2, s3)$	Spline function	[0, 1]
R	Position multiplier	[256, 768]

5. Experimental setup

The quadruped robot used, QuadraTot, was assembled from parts purchased online and parts printed by the Objet Connex 500 3D Printing System. The robot actuation system consists of 5 AX-18+ Dynamixel servos and 4 AX-12+ Dynamixel servos: one inner joint with one AX-18+ servo and one outer joint with AX-12+servo in each of the four legs, and one AX-18+ servo at the center. To avoid the formerly reported problem with AX-18+ servos are used in this robot because of their stronger actuation power than that of AX-12+. Each servo could be set to a position in the range [0, 1023] by using pydynamixel library, corresponding roughly to a physical range [–120°, +120°]. Also, to prevent collisions with the robot body, the control module filter out the commands to a safe range. This range was [–85°, +60°] for the inner leg servos, [–113°, +39°] for the outer leg servos, and [–28°, +28°] for the central hip servo. In the studies of this paper, tethered cables powered both the computer and the servos. It measures approximately 39.5 centimeters from leg to opposite leg in the crouch position shown in Fig. 1. Our performance metric was the displacement over the evaluation period of 12 seconds for each. Same as Yosinski et al. [12], the displacement was measured using a Wii remote that was placed on the ceiling. Different from the original model described in Yosinski et al. [12], the quadruped robot was equipped with a three-infrared-LED cluster on top rather than just one. The reason for this setup is that when fierce gaits were executed, the Wii remote loses tracking of the robot position due to the limited visible angle of a single LED. These three LEDs were placed tightly together to act as one signal emitter. Each LED was tilted outwards in order to maximize the visible range. A separate tracking server ran on the robot PC interacted with the Wii

remote via bluetooth by using the CWiid library. A corresponding client communicates with this server via socket connection. Our fitness evaluation code talks with the Wii remote by using this client and updates the position from beginning to end during each run. The metric for evaluating gaits was the Euclidian distance the robot travelled during a 12-second run on flat surface. Previous work done in Yosinski et al. [12] and Clune et al. [3] suggested that extremely fierce gaits are not viable in general. These gaits tend to either overburden the servos or flip the robot. Thus, RL PoWER was tested after carefully cropping out the extremely fierce gaits each time. As described earlier, each leg has two joints, inner joint $j1$ and outer $j2$. The position of one leg is determined by the sum of the values of $j1$ and $j2$, in the range of [0, 1023]. Extremely fierce gaits usually have relatively small values of $\boldsymbol{g}$. As shown in Fig. 3, this cropping method works by mapping the wild gaits right onto the boundary of a quasi-triangular area in the two-dimensional space of $j1$ and $j2$. In our experimentation, we set the threshold to 730 to balance between gait performance and safety.

Table 2. The average and standard deviation of the best gaits found for RL PoWER and HyperNEAT during each of three runs, in cm/sec. Also included is the GA gaits optimized by simulation. The data are not directly comparable because of different experiment methodology. But the best single is worth noting

Algorithm	Average	Best Single	Std. Dev
RL PoWER	**7.62**	11.05	2.1
HyperNEAT	6.28	9.7	1.26
GA	N/A	12.95	N/A

However, an obvious trade-off to this method is the reduced exploratory region of the parameters. An extension would be experiments using smaller g, cropping less gaits, to enhance the performance in general.

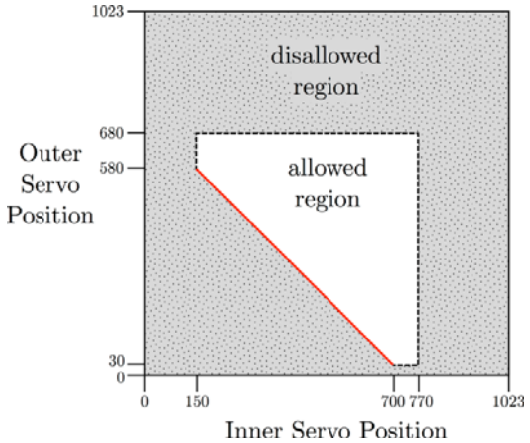

Fig. 3. The red line is the smart cropping border, $g = j1 + j2$. The dashed grey lines are the physical limits of the joints. As shown in Fig. 4, the fierce movements are prevented by mapping these out-of-range positions to the borderline as shown in the graph. The value of g cannot drop below 730, otherwise $j1$ and $j2$ are projected right to the red line

6. Results

The results for the gaits evolved by RL PoWER are shown in Fig. 5 and Table 2.The HyperNEAT performance is shown in Fig. 6. A total of 900 evaluations were performed for RL PoWER (300 for each of three runs). The reason these two algorithms have a different number of trials is that runs continued until the performance plateaued, which we defined as when there was no improvement during the last third of a run. Overall, the RL PoWER gaits were faster by far, beating the HyperNEAT when comparing either average or best gaits.

Fig. 4. The left two positions are allowed; middle two positions are on the red border line from Fig. 3; right two positions are deemed as "extremely fierce", thus are disallowed. Future work may involve studying a larger range of *g* for increased performance

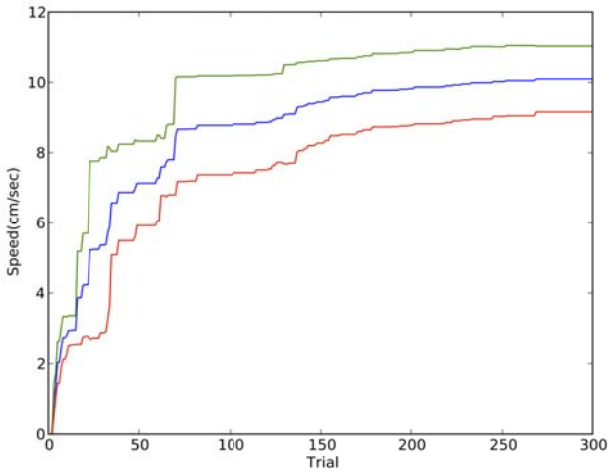

Fig. 5. Average fitness (± SE) in cm/sec of the highest performing run of RL PoWER's three runs

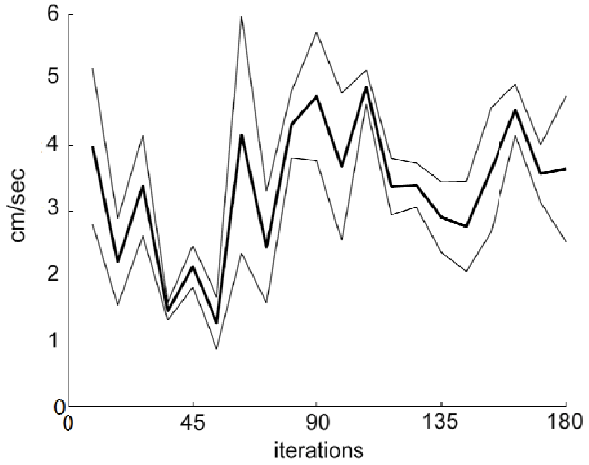

Fig. 6. Average fitness (± SE) in cm/s of the highest performing individual in one HyperNEAT run, reproduced from Y o s i n s k i et al. [12]

We believe that this is because RL PoWER uses evolvable splines to represent gaits. As explained earlier, The single best gait found during this study had a speed of 11.09 cm/s, 16.3% better than the best HyperNEAT gait. Fig. 7 shows a typical RL PoWER gait that had high fitness. Compared with Fig. 8, the pattern of RL PoWER's motion is less complex but more regular than the HyperNEAT, but producing higher speed. As shown in Fig. 7, multiple motors are more coordinated than the HyperNEAT gait.

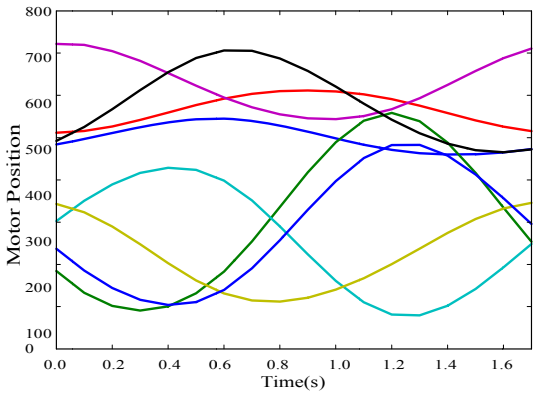

Fig. 7. The motor positions of a typical well performing gait over one cycle. According to the spline transformation, the curves of joint positions are merely linear transformations of the spline curves. Compared with the pattern of a well-performing HyperNEAT gait, the spline representation of the parameterized gaits are simple yet powerful

A corresponding observation from the graph on this property is that the noisiness of RL PoWER is higher than HyperNEAT. One reason for this is that the RL PoWER intentionally adds noise when exploring the space near the best-ranked policies. Another phenomenon seen from Fig. 5, is that one run of RL PoWER's general performance largely depends on initial gaits, which is a trait of hill climbing algorithms. The larger standard deviation in RL PoWER for space

exploring can be easily fixed by tuning the noise parameter. RL PoWER converges to optimality at a higher rate. This is due to the more heuristically guided reinforcement learning of RL PoWER. HyperNEAT, on the other hand, has a larger dimension to explore due to its generative encoding method. While the evolvable spline representation used by RL PoWER has lower dimensions. The convergence of HyperNEAT is thus slower.

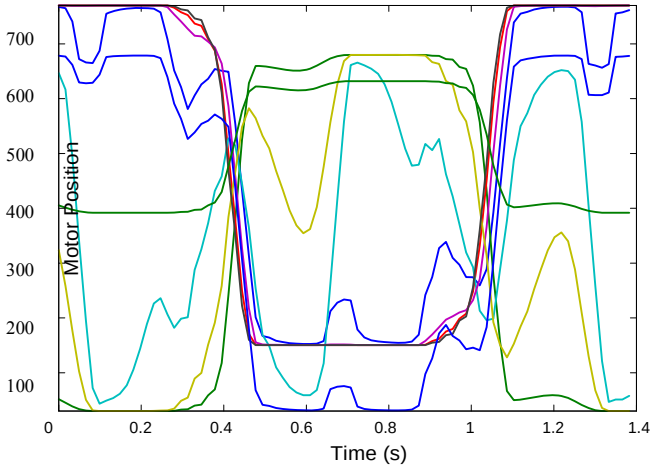

Fig. 8. A typical well performing HyperNEAT gait's pattern of motor positions, reproduced from Yosinski et al. [12]

7. Conclusions and future work

We have presented results from a recently introduced reinforcement-learning-based algorithm for optimizing a quadrupedal gait for linear speed, tested on a physical robot. We implemented the algorithm, Policy learning by Weighting Exploration with the Returns (a.k.a RL PoWER), for parameterized gaits and compared its results with those produced by HyperNEAT generative encoding and GA in a refined simulator. Each of the three methods resulted in an improvement over the robots' previous naïve gaits.

Over 900 trials have been made to investigate the applicability of RL PoWER to quadruped robots. It is difficult to gather the enough trials that would be necessary to properly rank the methods. One direction for future work could be to obtain many more trials. But due to the physical limitations, obtaining one solution to this is simulation. The results from Glette et al.'s GA algorithm, as seen in Table 2, show how simulation can help accelerate the experimentation process. Because of the low cost of simulation, it would produce the necessary volume of trials to allow the learning methods to be effective, and the hardware trials would serve to continuously ground and refine the simulator.

One hypothesis supported by this study is that for feedback-oriented tasks, reinforcement learning methods are more fit by the nature of gait learning tasks. Despite the complexities of HyperNEAT, a structurally simpler algorithm, such as RL PoWER delivered better performance in general. Also, evolvable spline interpolation

is shown to be simple and representationally powerful at the same time. Evolvable splines can serve as a general representation for various other learning problems.

Acknowledgements. This work was supported by the National Science Foundation's Office of Emerging Frontiers in Research and Innovation (Grant Number 0735953) and an NSF Postdoctoral Research Fellowship in Biology to Jeff Clune (DBI-1003220).

References

1. Bongard, J., V. Zykov, H. Lipson. Resilient Machines Through Continuous Self-Modeling. – Science, Vol. **314**, 2006, No 5802, 1118-1121.
2. Chernova, S., M. Veloso. An Evolutionary Approach to Gait Learning for Four-Legged Robots. – In: Intelligent Robots and Systems, 2004. (IROS'2004). Proceedings. 2004 IEEE/RSJ International Conference on, Vol. **3**, 2004, 2562-2567.
3. Clune, J., B. Beckmann, C. Ofria, R. Pennock. Evolving Coordinated Quadruped Gaits with the Hyperneat Generative Encoding. – In: Evolutionary Computation, 2009. CEC'09. IEEE Congress on, 2009, 2764-2771.
4. Clune, J., K. Stanley, R. Pennock, C. Ofria. On the Performance of Indirect Encoding across the Continuum of Regularity. Evolutionary Computation. – IEEE Transactions, Vol. **15**, 2011, No 3, 346-367.
6. Glette, K., G. Klaus, J. C. Zagal, J. Torresen. Evolution of Locomotion in a Simulated Quadruped Robot and Transferral to Reality. – In: Proceedings of the 17th International Symposium on Artificial Life and Robotics, 2012.
7. Hornby, G., S. Takamura, T. Yamamoto, M. Fujita. Autonomous Evolution of Dynamic Gaits with Two Quadruped Robots. – Robotics, IEEE Transactions on, Vol. **21**, 2005, No 3, 402-410.
8. Kober, J., J. Peters. Learning Motor Primitives for Robotics. – In: Robotics and Automation, 2009. ICRA'09. IEEE International Conference on, 2009, 2112-2118.
9. Kormushev, P., B. Ugurlu, S. Calinon, N. Tsagarakis, D. Caldwell. Bipedal Walking Energy Minimization by Reinforcement Learning with Evolving Policy Parameterization. – In: Intelligent Robots and Systems (IROS), 2011 IEEE/RSJ International Conference on, 2011, 318-324.
10. Tel´lez, R., C. Angulo, D. Pardo. Evolving the Walking Behaviour of a 12 Dof Quadruped using a Distributed Neural Architecture. – Biologically Inspired Approaches to Advanced Information Technology, 2006, 5-19.
11. Valsalam, V., R. Miikkulainen. Modular Neuroevolution for Multilegged Locomotion. – In: Proceedings of the 10th Annual Conference on Genetic and Evolutionary Computation, ACM, 2008, 265-272.
12. Yosinski, J., J. Clune, D. Hidalgo, S. Nguyen, J. C. Zagal, H. Lipson. Evolving Robot Gaits in Hardware: the Hyperneat Generative Encoding Vs. Parameter Optimization. – In: Proceedings of the 20th European Conference on Artificial Life, 2011.
13. Zykov, V., J. Bongard, H. Lipson. Evolving Dynamic Gaits on a Physical Robot. – In: Proceedings of Genetic and Evolutionary Computation Conference, Late Breaking Paper, GECCO, Vol. **4**, 2004.