DTU Library

Adaptive Strategy for Online Gait Learning Evaluated on the Polymorphic Robotic LocoKit

Christensen, David Johan; Larsen, Jørgen Christian; Stoy, Kasper

Published in:
Proceedings of the IEEE Conference on Evolving and Adaptive Intelligent Systems

Link to article, DOI:
10.1109/EAIS.2012.6232806

Publication date:
2012

Adaptive Strategy for Online Gait Learning Evaluated on the Polymorphic Robotic LocoKit

David Johan Christensen
Department of Electrical Engineering
Technical University of Denmark
Kgs. Lyngby, Denmark
Email: djchr@elektro.dtu.dk

Jørgen Christian Larsen
The Maersk Mc-Kinney Moller Institute
University of Southern Denmark
Odense, Denmark
Email: jcla@mmmi.sdu.dk

Kasper Stoy
The Maersk Mc-Kinney Moller Institute
University of Southern Denmark
Odense, Denmark
Email: kaspers@mmmi.sdu.dk

***Abstract*—This paper presents experiments with a morphology-independent, life-long strategy for online learning of locomotion gaits, performed on a quadruped robot constructed from the LocoKit modular robot. The learning strategy applies a stochastic optimization algorithm to optimize eight open parameters of a central pattern generator based gait implementation. We observe that the strategy converges in roughly ten minutes to gaits of similar or higher velocity than a manually designed gait and that the strategy readapts in the event of failed actuators. In future work we plan to study co-learning of morphological and control parameters directly on the physical robot.**

I. Introduction

Reconfigurable modular robots are polymorphic in the sense that by assembling the modules in different configurations, robots with different morphologies and thereby capabilities can be constructed. Since a robot's mobility is highly dependent on the details of its morphology, the flexibility of a modular morphology makes modular robots an attractive platform for studying robot locomotion.

Control of locomotion must be designed with careful attention to the interdependence with the robot's morphology and environment. However, since modular robots are polymorphic, we desire a control strategy which is not designed for a specific morphology. The strategy should rather be adaptive to enable optimization of a variable number of control parameters, for a class of morphologies. The morphology of a modular robot can change over time, either due to module failures, adding or removing of modules, or due to voluntary morphosis. Therefore, the strategy should ideally be life-long to enable adaptation to changes in morphology or environment during the lifetime of the robot.

In this paper we address the challenge of interdependence between environment, morphology, and locomotion control by proposing a morphology-independent, life-long, online learning strategy. The strategy control each actuator based on periodic actuation patterns generated by an oscillator. The robot is controlled by a network of coupled oscillators that form an entrained central pattern generator (CPG). To enable life-long learning based on noisy fitness measurements we apply the model-less Simultaneous Perturbation Stochastic Approximation (SPSA) method [20]. The phase-shift of each oscillator is optimized based on the robot's measured velocity.

In previous work we studied the proposed strategy on simulated Roombot robots [5]. This paper contributes physical experimental validation of the proposed strategy on an 8-DOF quadruped LocoKit robot. We successfully apply the same online learning strategy on a LocoKit quadruped as we did on the Roombots, although the two systems are very different in terms of degrees of freedom and modularity. In this paper all experiments are controlled from a single processor, but both the control and the learning can be distributed without any centralized control necessary (as we did in [5]).

In this paper we first in Sec. II provide an overview of related work with a focus on adaptive locomotion of modular robots. Then, in Sec. III, we describe the LocoKit and the design of a quadruped utilized for experiments. The proposed control and learning strategy is described in Sec. IV. The experimental setup with the robot on a boom and establishment of learning parameters is described in Sec. V. Experiments on online learning and adaptation to failures are described in Sec. VI. Future work and conclusions are given in Sec. VII.

II. Related Work

Homogeneous reconfigurable modular robots are systems where all the modules have the same combination of mechanics and electronics. Alternatively, heterogeneous modular robots contains several types of modules with different functionality, the degree to which the modules are self-contained can vary from autonomous mobile modules [8] to mechanical components as in LEGO MINDSTORMS. Heterogeneous systems [6], [23], [27] includes the LocoKit [13] which we utilize in this paper. More details on the history and mechatronics of modular robots can be found in recent surveys [22], [26].

Evolutionary algorithms are a popular way to optimize locomotion gaits for modular robots. In the early 90's, Karl Sims pioneered the field by co-evolving the morphology and control of simulated modular robots [19]. Later work succeeded in transferring similar co-evolved robots from simulation to hardware [14], [17]. An example of adaptation by evolution in modular robots was conducted by Kamimura et al., who evolved the coupling parameters of central pattern generators for straight line locomotion of modular M-TRAN robots [11]. By incorporating sensory entrainment in the optimization the authors were able to bridge the reality gap. Although appealing, one challenge with evolutionary approaches is that once transferred, the robot is typically no longer able to adapt to major changes in the morphology or environment.

To overcome this limitation of evolutionary algorithms, locomotion gaits can be optimized online. This was studied by Marbach and Ijspeert on the YaMoR modular robotic system [18]. Their strategy was based on Powell's method, which performed a localized search in the space of selected parameters of coupled oscillators. Parameters were manually extracted from the modular robot by exploiting symmetries. Follow-up work by Spröwitz et al. demonstrated online optimization of 6 parameters on a physical robot in roughly 25-40 minutes [21]. We also try to realize simple, robust, fast, model-free, life-long learning on a modular robot. The main difference is that we seek to automate the controller design further in the sense that no parameters have to be extracted from symmetric properties of the robot.

In most related work, control and optimization is performed in a centralized fashion. However, our approach utilizes a control and optimization strategy which is also appropriate for a distributed implementation. A similar approach was taken by Maes and Brooks who performed distributed learning of locomotion on a 6-legged robot [15]. The learning was distributed to the legs themselves. The potential advantages of a distributed strategy include inherent morphology independence and fault tolerance.

Our strategy is not dependent on the robot's specific morphology. Similarly, Bongard et al. demonstrated learning of locomotion and adaptation to changes in the configuration of a modular robot [1]. They took a self-modeling approach, where the robot developed a model of its own configuration by performing basic motor actions. In a physical simulator a model of the robot configuration was evolved to match the sampled sensor data (from accelerometers). By co-evolving the model with a locomotion gait, the robot could then learn to move with different morphologies. Our work presented here is similar in purpose but different in approach: The strategy is simple, model-less and computationally cheap to allow implementation on small embedded devices, such as typical modular robots.

This paper includes experiment on adaptation after failures of the robots actuators. It is an attractive possibility to realize fault tolerance and self-repair by taking advantage of modular robot's redundancy and ability to adapt and self-reconfigure. This has previously been demonstrated on modular robots engaged in locomotion and self-reconfiguration [1], [2], [7], [24], [28]. For example, in a paper by Mahadavi and Bentley [16] the control of a snake like robot was optimized online using a genetic algorithm. The algorithm was shown to recover from failures in the SMAs actuating the robot.

This paper utilizes a model of central pattern generators (CPGs) to generate actuation patterns for locomotion. For snakes and legged robots, CPGs are often applied to control locomotion [9]. The advantages of CPGs include: stable limit cycle behavior, appropriate for distributed implementation, few control parameters, suited to integrate sensory feedback signals, and offers a good substrate for learning and optimization algorithms [9].

The choice of learning strategy critically affects the performance of the system. An experimental comparison of different algorithms for online optimization of locomotion gaits for the AIBO robot was presented in a paper by Kohl and Stone [12]. They compared four machine learning algorithms and found that simpler algorithms (hill climbing and policy gradient) performed better on the problem than the more complex algorithms (amoeba and genetic algorithm). In this paper we utilize a stochastic optimization algorithm (SPSA) to optimize the parameters of the central pattern generators. In previous work we used distributed reinforcement learning (DRL) for morphology independent learning of discrete actions and gait-tables to control locomotion of modular robots [2], [4]. The main advantage of SPSA over DRL is that it allows optimization in a continuous space which is appropriate for central pattern generators.

III. LocoKit - A Robotic Building Kit

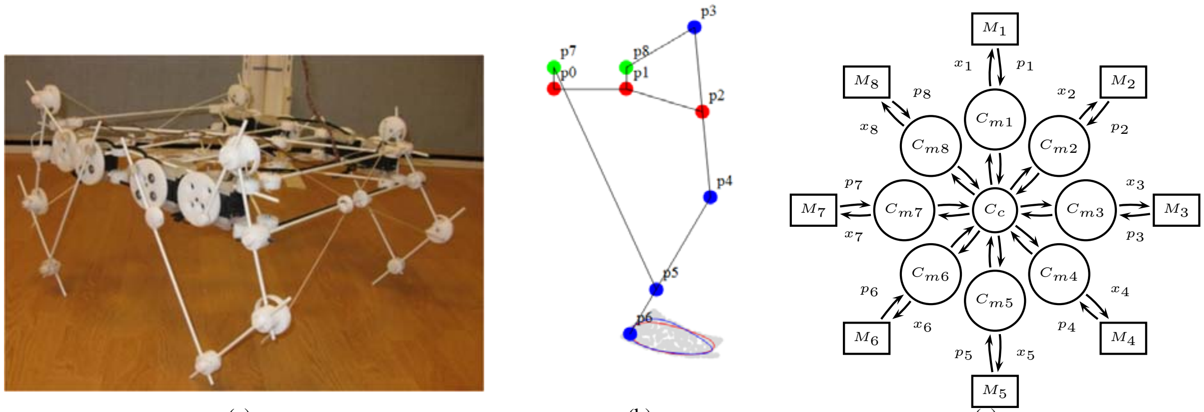
The robot, used for experiments in this paper, is build from the polymorphic modular robotic building kit called "LocoKit" [13]. The objective of LocoKit is to realize a flexible user reconfigurable modular robotic kit which is light weight, affordable and can be used to quickly realize energy efficient and robust legged robots. The LocoKit aims at realizing these design goals through a layered heterogeneous structure, with the following layers:

- Skeleton layer - low weight mechanical components
- Actuation layer - currently electrical servos
- Electronics layer - sensing, power and computation

The LocoKit separates the mechanics, actuation and electronics in simple reconfigurable modular components to increase flexibility and reduce the weight and complexity of individual modules.

The LocoKit components include glass fiber-reinforced plastic rods that connect everything in the system, and connection components made of 3D printed acryl. For actuation, the system currently uses the Dynamixel RX-10 servo. Note that the prototype of the LocoKit used in this paper is a relative early prototype and the details of the kit has changed significantly in newer versions.

Based on the LocoKit components we constructed a quadruped robot as our experimental platform, see Fig. 1(a). Each leg is based on a 4-bar linkage which is actuated using two actuators that are able to rotate infinite, the reachable space of the foot is illustrated in Fig. 1(b). This design is inspired by Theo Jansen's StrandBeest [10]. It is a long term design goal of the LocoKit to enable the design of energy efficient robots. This affects several of our design choices: 1) The actuators are placed in the body of the robot to keep the weight (and thereby momentum) of the legs to a minimum. 2) A typical gait will be generated by continuous phase-shifted rotations of the robots eight actuators - by not using oscillatory actuation we avoid that the actuators work against the momentum of itself and its gearbox. In this paper, experiments are performed with the



(a) (b) (c)

Fig. 1. (a) A quadruped robot with eight degrees of freedom constructed from the polymorphic robotic LocoKit. (b) The reachable space and two example foot trajectories are shows for the 2 DOF leg. (c) The CPG network architecture of coupled oscillators to control the LocoKit robot. The network consists of eight motors controlled by oscillators coupled to a single central oscillator.

servos controlled from a central PC and with external power, for simplicity and reliability. However, the system does already in the current version include the modules for onboard power (batteries + power electronics boards) and onboard control (micro-controller boards and communication boards).

IV. A Strategy for Life-long Online Learning

This section describes an adaptive locomotion strategy based on CPG's for generating periodic actuation patterns for gait implementation and SPSA for online optimization of gait parameters.

A. Central Pattern Generator and Network Architecture

Biological CPGs are special neural circuits found in vertebrates, able to produce a rhythmic signal without any external sensory input, where they for example control muscles during locomotion. We apply a CPG model for gait control because of their ability to generate periodic actuation patterns, ability to self-synchronize in a distributed system, open parameters which are appropriate for optimization, and finally since CPGs are biologically plausible. A review of CPGs and their use in robot control can be found in [9]. The specific CPG model we utilize is a Hopf oscillator in Cartesian space with diffusive coupling [25]. The advantages of this model include its simplicity, stable limit-cycle behavior, and explicit parameters for setting phase, amplitude and frequency. For an oscillator i the coupled differential equations are:

$$\dot{x}_i = \gamma(\mu - r_i^2)x_i - \bar{\omega}y_i \tag{1}$$

$$\dot{y}_i = \gamma(\mu - r_i^2)y_i + \bar{\omega}x_i \tag{2}$$

Where $r_i = \sqrt{x_i^2 + y_i^2}$ and the state variables are x and y. γ is a parameter that affects the speed of convergence towards the oscillators amplitude μ^2. $\bar{\omega}$ is the oscillator's frequency which is a function of a frequency parameter, ω, and is also affected by the sum of couplings to other oscillators. A coupling from oscillator i to oscillator j has a weight parameter, w_{ij}, and a desired phase difference ϕ_{ij}. Then the oscillator may be coupled to other oscillators using:

$$\bar{\omega} = \omega + \sum_{j=1}^{N} \frac{w_{ij}}{r_i} \left[(x_i y_j - x_j y_i)\cos\phi_{ij} - (x_i x_j + y_i y_j)\sin\phi_{ij}\right] \tag{3}$$

We can make the actuator oscillate with a given frequency, phase-shift and amplitude by setting the setpoint of the actuator to $\theta = x_i$. For the LocoKit robot, to make the actuator rotate continuously we set it to $\theta = \arctan(x_i/y_i)$ and select the appropriate quadrant.

The LocoKit robot is programmed with nine coupled oscillators: eight which are used as set-points for its actuators (C_{m1}, C_{m2} ... C_{m8}) and one which acts as a central clock (C_c). The architecture is illustrated in Fig. 1(c). The centralized architecture can easily be made distributed without significantly affecting the system performance, as in previous work [5].

B. Learning Algorithm

For online optimization of CPG parameters we apply the Simultaneous Perturbation Stochastic Approximation (SPSA) method by Spall [20]. This algorithm requires no explicit gradient and therefore no model of the robot. It is designed to build an approximation of the gradient from direct, generally noisy, measurements of the objective function. Further, SPSA only requires two measurements of the objective function per iteration (i.e. two robot trials with different controllers) independent on the number of adjustable parameters. Also, these measurements are made based on small perturbations of the same parameter set. Hence the robot's behavior only alters slightly while it is learning, unlike optimization based on population-based methods such as evolutionary algorithms. Finally, SPSA is simple to implement in a distributed fashion since each module may independently optimize its own parameters without knowledge of the other modules parameters or the need for any other coordination than simple synchronization of when the parameters are updated.

The SPSA method optimizes the parameter set $\hat{\theta}$ defined by the experimenter. In an iteration, k, it estimates the gradient, $g(\hat{\theta})$, based on two noisy measurements of the objective function $y(\hat{\theta})$:

$$\hat{g}_k(\hat{\theta}_k) = \frac{y(\hat{\theta}_k + c_k\Delta_k) - y(\hat{\theta}_k - c_k\Delta_k)}{2c_k} \begin{bmatrix} \Delta_{k1}^{-1} \\ \Delta_{k2}^{-1} \\ \vdots \\ \Delta_{kp}^{-1} \end{bmatrix} \quad (4)$$

Where c_k is a learning parameter and Δ_k is an vector of randomized ± 1. SPSA then updates $\hat{\theta}$ based on $\hat{g}_k(\hat{\theta}_k)$:

$$\Delta\hat{\theta}_k = -a_k \cdot \hat{g}_k(\hat{\theta}_k) \quad (5)$$

$$\hat{\theta}_{k+1} = \hat{\theta}_k + sign(\Delta\hat{\theta}_k) \cdot min(|\Delta\hat{\theta}_k|, \epsilon) \quad (6)$$

a_k is a learning parameter, we also added a max step-size, ϵ, to reduce the risk of instability.

V. Experimental Setup

A. Physical Setup

In the process of learning how to walk, the robot will need the freedom to try out numerous different gaits, while observing its locomotion speed. For our experiments we mount the robot on a boom, which gives us the advantage of being able to run experiments for a longer period of time without human interaction. The boom provides threaded power to the robot and removes 665 grams of weight of the robot (i.e. 41 percent) by using counterweights, and thereby minimizing the risk of the robot breaking itself during the experiments. Clearly this lift together with the momentum of the boom affects the dynamics of the robot. We do however accept this source of error, since the purpose of the experiments is to validate the learning strategy on a physical robot, not to find efficient gaits for the particular quadruped operating without a boom. The boom has a radius of 1.5 m and an encoder measures the position of the robot with resolution of 0.5 cm/degree in the end of the arm where the robot is mounted. To fasten the robot onto the boom, a universal joint is placed between the robot and the boom, in order to make it possible for the robot to move in the roll and pitch angle while still being fastened in the yaw angle. The robot can move in the up/down and forward/backward direction but not sideway.

B. Control Parameters

Several parameters need to be established before experimental trials can be performed with the robot. The reward signal optimized by the learning system is a measurement of the velocity. We estimate the velocity as the distance moved by the robot in five steps, $v = |p_{t+T} - p_t|/T$. This corresponds to a time of $T = 5 \cdot 1/\omega = 7.14$ sec, where the actuators are set to rotate with a frequency of $\omega = 0.7$ Hz. The oscillators are tightly coupled with weight parameters, w_{ij}, as described in previous work [5]. Both learning parameters, c_k and a_k, are set at fixed values to enable life-long learning. Generally, c_k can be set based on about how much the control parameters should be changed to cause an measurable effect on the objective function. For our purpose we set $c_k = 0.025$ which corresponds to a variation in the phase-shifts of ± 9.0 degrees while learning. Similarly, a_k can be set based on how much a control parameter should be changed given a typical measured velocity difference. We set $a_k = 0.0015$ which corresponds to a phase shift change of 10.8 degrees at a typical 1 cm/sec velocity difference. These learning parameters are set quite high to achieve fast convergence, potentially at the cost of convergence to a local optima or divergence.

The learning strategy optimizes the eight phase-shifts for the eight actuators from an initial gait which has all eight phase-shifts set to 0. For comparison we utilize an ideally symmetrical trot gait which has four phase-shifts set to π and four to $-\pi$. This trot gait was previously manually designed for high velocity for the purpose of a public demonstration. The strategy is implemented as part of the "Assemble and Animate" (ASE) control framework for modular robots [3].

Exp nr.	Learning	Playback
Trot	-	12.9 cm/sec
1	12.8 cm/sec	13.2 cm/sec
2	13.6 cm/sec	17.5 cm/sec
3	14.7 cm/sec	16.9 cm/sec
4	13.3 cm/sec	14.3 cm/sec
5	14.0 cm/sec	15.5 cm/sec

TABLE I
Average velocity in five learning trials.

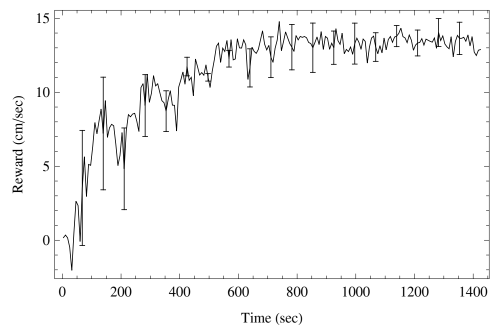


Fig. 3. Average of 5 trials of online learning with LocoKit quadruped. Error bars indicate one standard deviation.

VI. Experiments

A. Online Gait Learning

In this experiment we validate the learning strategy on the quadruped robot. As explained above the robot is mounted on a boom, feedback from the boom-encoder is sent wireless to a PC using ZigBee. The PC controls the robot's actuators based on the proposed SPSA and CPG based strategy. In each trial we let the robot learn for a minimum of 20 minutes until the gait velocity has stabilized.

The reward graph of a typical learning example is shown in Fig. 2(a). For comparison the graph also shows the measured velocity of a manually designed trot gait which has an average velocity of 12.9 cm/sec (with a standard deviation of σ=0.49). We observe that the robot while learning improves its initial velocity from around 0 cm/sec until it stabilizes around 13.0 cm/sec after approximately eight minutes. Further, when running the final learned gait without online learning we find

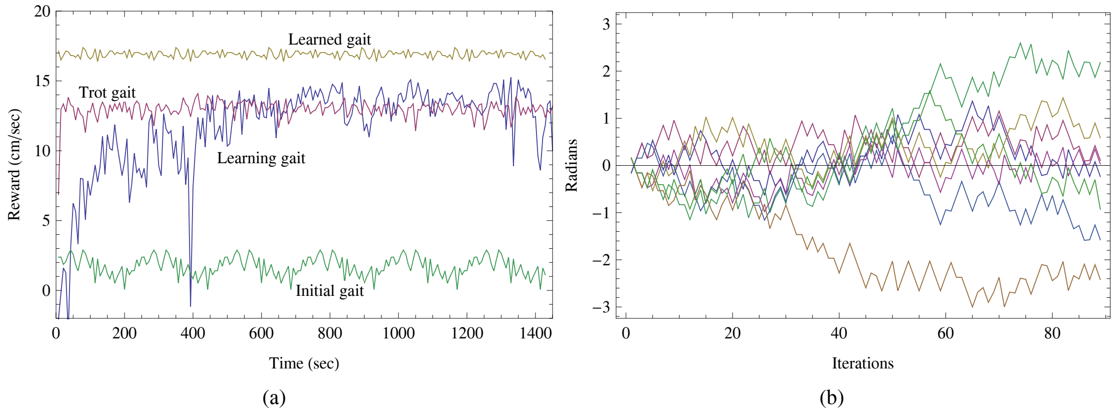


Fig. 2. Online learning with quadruped robot. (a) The reward (average robot velocity in five steps) is shown as a function of time in four cases: for a learning gait, for the initial start-gait of learning, for a manually designed trot gait, and for the learned gait (CPG control and no learning). (b) Online learning of eight control parameters (phase-shifts).

that the robot is moving with a velocity of 17.5 cm/sec which is significant faster than the manually designed gait. This increase in velocity when not learning is due to a decrease in average velocity caused by the learning strategy's exploration of CPG parameters.

The adaptations of the corresponding eight open parameters are shown in Fig. 2(b). The parameters do not show any clear convergence towards specific values, but fluctuate over time. However, from observing the behavior of the robot, it is clear that the gait quickly converge to a trot-like gait.

A total of five learning trials have been run with similar results. An overview of the results are shown in Table I. The table shows the average velocity at the end of five learning trials and the average velocity of "playing back" the learned controller, i.e. CPG control without learning enabled. For comparison also the average velocity of a manually designed trot gait is shown. Fig. 3 illustrate the average fitness graph for the five trials.

We observe that although the different trials converge to different gaits they all have similar velocity (from 12.8 to 14.7 cm/sec). On average this is slightly faster (13.7 cm/sec) than the manually designed trot gait with a velocity of 12.9 cm/sec. The average velocity of playing back the learned gaits is 15.5 cm/sec which means that the gait variations while learning decrease the velocity with on average 1.8 cm/sec. In the five trials it takes 4-11 minutes (on average 8 minutes) before the learning robot is moving with an average velocity faster than 12 cm/sec. This fast convergence confirms our results with simulated Roombots which would optimize 45-54 open parameters in 5-30 min. to gaits with 35-50% faster velocity than gaits found by blind random search [5].

In summary the learning strategy is fast, reliable, and effective in converging to gaits comparable or slightly better than the manually designed gait.

B. Online Adaptation to Failures

In this experiment we use the quadruped LocoKit robot to study the strategy's ability to adapt to failures of the actuators. In both experiments we let the robot continue to learn after several of the actuators has been stopped in a predefined position to simulate a failure of the actuators.

In the first experiment the two actuators controlling a back leg is stopped. The result is shown in Fig. 4(a). If learning is not enabled the robot's average velocity drops to 6.9 cm/sec, however, due to the online learning the robot regains an average velocity of 9.61 cm/sec while learning. Although we

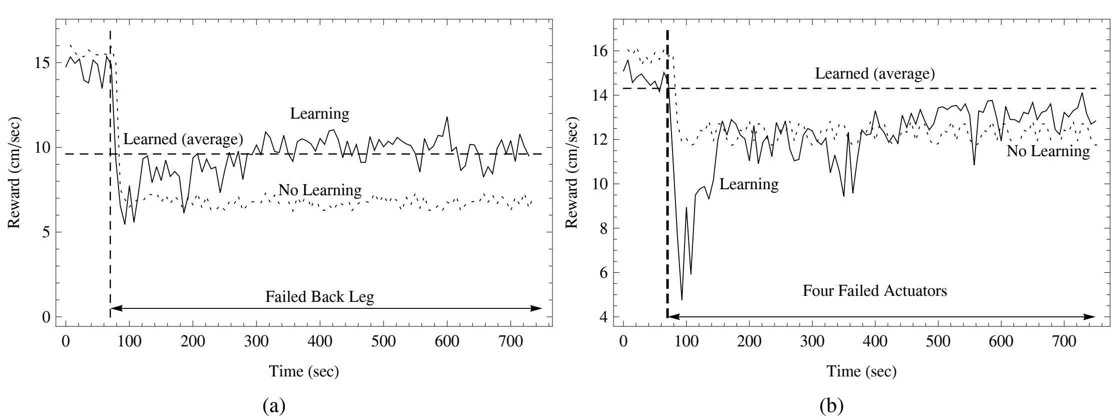


Fig. 4. Online adaptation to actuator failures in the LocoKit quadruped. (a) The effect of single back leg failure (two actuators stops). (b) The effect of failure on all knees (four actuators stop).

cannot make strong conclusions based on a single trial, we expect this to be a beneficial effect of adaptation. Simulation experiments on adaptation to failures shows a significant effect for the same strategy in previous work [5].

In the second experiment four actuators controlling the "knee" of each leg is stopped. The result is shown in Fig. 4(b). This causes the velocity to drop to 12.2 cm/sec (no learning). If learning is enabled the robot achieves an average velocity of 13.1 cm/sec while learning and of 14.4 cm/sec if executing the learned gait. In this trial the effect of adaptation is less clear. In fact is seems that the quick drop in velocity due to the failure, makes the gait diverge temporary to an less efficient gait, which the strategy then readapts to its previous performance within a couple of minutes.

In summary the experiments indicate that the online learning strategy is able to adapt to morphological changes such as failed actuators, but the concrete effect depends on complex interactions between the robot, its environment, the control system, and the type of failure.

VII. Conclusion and Future Work

This paper described a control strategy for online life-long learning of locomotion gaits. The strategy was experimentally evaluated on a robot constructed from the polymorphic robotic LocoKit. We found that the strategy was able to find efficient locomotion gaits by online optimization of eight open parameters on average within 10 minutes. We also performed experiments on continued adaptation after failures of several actuators and found that the system was able to readapt after such failures. In future work we will refine the mechanical components of the LocoKit and extend the system with components that enables the system to change its morphology online. We can then study online co-learning of morphological and control parameters. Further, numerous improvements could be studied for the proposed strategy, most significantly, adaptive learning parameters in order to make the strategy even more generic.

VIII. Acknowledgements

This work was performed as part of the "Locomorph" project funded by the EU's Seventh Framework Programme (Future Emerging Technologies, Embodied Intelligence) and as part of the "Assemble and Animate" project funded by the Danish Council for Independent Research.

References

[1] J. Bongard, V. Zykov, and H. Lipson. Resilient machines through continuous self-modeling. *Science*, 314(5802):1118–1121, 2006.

[2] D. J. Christensen, M. Bordignon, U. P. Schultz, D. Shaikh, and K. Stoy. Morphology independent learning in modular robots. In *Proceedings of International Symposium on Distributed Autonomous Robotic Systems 8 (DARS 2008)*, pages 379–391, 2008.

[3] D. J. Christensen, U. P. Schultz, and M. Moghadam. The assemble and animate control framework for modular reconfigurable robots. In *Proceedings of the IROS Workshop on Reconfigurable Modular Robotics: Challenges of Mechatronic and Bio-Chemo-Hybrid Systems*, 2011.

[4] D. J. Christensen, U. P. Schultz, and K. Stoy. A distributed strategy for gait adaptation in modular robots. In *Proceedings of the IEEE Int. Conference on Robotics and Automation (ICRA)*, pages 2765–2770, 2010.

[5] D. J. Christensen, A. Sproewitz, and A. J. Ijspeert. Distributed online learning of central pattern generators in modular robots. In *Proceedings of the 11th International Conference on Simulation of Adaptive Behavior (SAB2010)*, pages 402–412, Paris, France, 2010.

[6] D. Duff, M. Yim, and K. Roufas. Evolution of polybot: A modular reconfigurable robot. In *Proceedings of Harmonic Drive International Symposium*, Nagano, Japan, 2001.

[7] R. Fitch, D. Rus, and M. Vona. A basis for self-repair using crystalline modules. In *Proceedings, Intelligent Autonomous Systems (IAS-6)*, Venice, Italy, 2000.

[8] R. Groß, M. Bonani, F. Mondada, and M. Dorigo. Autonomous self-assembly in a swarm-bot. In *Proc. of the 3rd Int. Symp. on Autonomous Minirobots for Research and Edutainment (AMiRE 2005)*, pages 314–322. Springer, Berlin, Germany, 2006.

[9] A. J. Ijspeert. Central pattern generators for locomotion control in animals and robots: a review. *Neural Networks*, 21(4):642–653, 2008.

[10] T. Jansen. *Theo Jansen: The Great Pretender*. OIO Publishers, 2007.

[11] A. Kamimura, H. Kurokawa, E. Yoshida, S. Murata, K. Tomita, and S. Kokaji. Automatic locomotion design and experiments for a modular robotic system. *IEEE/ASME Transactions on Mechatronics*, 10(3):314–325, June 2005.

[12] N. Kohl and P. Stone. Machine learning for fast quadrupedal locomotion. In *The Nineteenth National Conference on Artificial Intelligence*, pages 611–616, July 2004.

[13] J.C. Larsen, R.F.M. Garcia, and K. Stoy. Increased versatility of modular robots through layered heterogeneity. In *Proceedings of the ICRA Workshop on Modular Robots, State of the Art*, pages 24–29, Anchorage, Alaska, May 2010.

[14] H. Lipson and J. B. Pollack. Automatic design and manufacture of robotic lifeforms. *Nature*, 406:974–978, 2000.

[15] P. Maes and R. A. Brooks. Learning to coordinate behaviors. In *National Conference on Artificial Intelligence*, pages 796–802, 1990.

[16] S. H. Mahdavi and P. J. Bentley. An evolutionary approach to damage recovery of robot motion with muscles. In *Seventh European Conference on Artificial Life (ECAL03)*, pages 248–255. Springer, 2003.

[17] D. Marbach and A. J. Ijspeert. Co-evolution of configuration and control for homogenous modular robots. In *Proc., 8th Int. Conf. on Intelligent Autonomous Systems*, pages 712–719, Amsterdam, Holland, 2004.

[18] D. Marbach and A. J. Ijspeert. Online Optimization of Modular Robot Locomotion. In *Proceedings of the IEEE Int. Conference on Mechatronics and Automation (ICMA 2005)*, pages 248–253, 2005.

[19] K. Sims. Evolving 3d morphology and behavior by competition. In R. Brooks and P. Maes, editors, *Proc., Artificial Life IV*, pages 28–39. MIT Press, 1994.

[20] J. C. Spall. Multivariate stochastic approximation using a simultaneous perturbation gradient approximation. *IEEE Transactions on Automatic Control*, 37(3):332–341, Mar 1992.

[21] A. Sproewitz, R. Moeckel, J. Maye, and A. J. Ijspeert. Learning to move in modular robots using central pattern generators and online optimization. *Int. J. Rob. Res.*, 27(3-4):423–443, 2008.

[22] K. Stoy, D. Brandt, and D. J. Christensen. *Self-Reconfigurable Robots: An Introduction*. Intelligent Robotics and Autonomous Agents series. The MIT Press, 2010.

[23] K. Stoy, A. Lyder, R.F.M. Garcia, and D. J. Christensen. Hierarchical robots. In *Proc. of the IROS Workshop on Self-Reconfigurable Modular Robot*, San Diego, CA, Nov. 2007.

[24] K. Stoy and R. Nagpal. Self-repair through scale independent self-reconfiguration. In *Proceedings of IEEE/RSJ International Conference on Robots and Systems (IROS)*, pages 2062–2067, Sendai, Japan, 2004.

[25] J. van den Kieboom. Biped locomotion and stability a practical approach. Master's thesis, University of Groningen, The Netherlands, 2009.

[26] M. Yim, W-M Shen, B Salemi, Daniela Rus, M. Moll, H Lipson, and E Klavins. Modular self-reconfigurable robot systems: Challenges and opportunities for the future. *IEEE Robotics & Automation Magazine*, 14(1):43–52, March 2007.

[27] M. Yim, B. Shirmohammadi, J. Sastra, M. Park, M. Dugan, and C. J. Taylor. Towards robotic self-reassembly after explosion. In *Video Proceedings of the IEEE/RSJ Intl.Conf.on Intelligent Robots and Systems (IROS)*, San Diego CA, 2007.

[28] E. Yoshida, S. Murata, K. Tomita, H. Kurokawa, and S. Kokaji. An experimental study on a self-repairing modular machine. *Robotics and Autonomous Systems*, 29:79–89, 1999.