A distributed and morphology-independent strategy for adaptive locomotion in self-reconfigurable modular robots

Christensen, David Johan; Schultz, Ulrik Pagh; Stoy, Kasper

Published in:
Robotics and Autonomous Systems

Link to article, DOI:
10.1016/j.robot.2013.05.009

Publication date:
2013

Robotics and Autonomous Systems ▮ (▮▮▮▮) ▮▮▮–▮▮▮

Contents lists available at SciVerse ScienceDirect

Robotics and Autonomous Systems

journal homepage: www.elsevier.com/locate/robot

A distributed and morphology-independent strategy for adaptive locomotion in self-reconfigurable modular robots

David Johan Christensen [a,*], Ulrik Pagh Schultz [b], Kasper Stoy [b]

[a] Department of Electrical Engineering, Technical University of Denmark, Kgs. Lyngby, Denmark
[b] The Maersk Mc-Kinney Moller Institute, University of Southern Denmark, Odense, Denmark

HIGHLIGHTS

- We propose an online learning strategy for locomotion in modular robots.
- The strategy is designed to be minimalistic and distributed.
- We experimentally study the strategy in simulation and on physical modular robots.
- Our findings include the fact that the strategy is morphology independent and can adapt to module faults and self-reconfiguration.

ARTICLE INFO

Article history:
Received 29 June 2012
Received in revised form
24 May 2013
Accepted 31 May 2013
Available online xxxx

Keywords:
Self-reconfigurable modular robots
Locomotion
Online learning
Distributed control
Fault tolerance

ABSTRACT

In this paper, we present a distributed reinforcement learning strategy for morphology-independent life-long gait learning for modular robots. All modules run identical controllers that locally and independently optimize their action selection based on the robot's velocity as a global, shared reward signal. We evaluate the strategy experimentally mainly on simulated, but also on physical, modular robots. We find that the strategy: (i) for six of seven configurations (3–12 modules) converge in 96% of the trials to the best known action-based gaits within 15 min, on average, (ii) can be transferred to physical robots with a comparable performance, (iii) can be applied to learn simple gait control tables for both M-TRAN and ATRON robots, (iv) enables an 8-module robot to adapt to faults and changes in its morphology, and (v) can learn gaits for up to 60 module robots but a divergence effect becomes substantial from 20–30 modules. These experiments demonstrate the advantages of a distributed learning strategy for modular robots, such as simplicity in implementation, low resource requirements, morphology independence, reconfigurability, and fault tolerance.

1. Introduction

Self-reconfigurable modular robots have a flexible morphology. The modules may autonomously or manually connect to form various robot configurations. Each module typically has its own micro-controller and can communicate locally with connected, neighbor modules. These characteristics entail that conventional centralized control strategies are often not suited for modular robots. Instead, we apply distributed control strategies which in general may have several advantages over centralized strategies in terms of ease of implementation, robustness, reconfigurability, scalability, and biological plausibility.

In this paper we focus on online learning suitable for modular robots; see Fig. 1. Such a learning strategy must require a minimal amount of resources and ideally be simple to implement on a distributed system. Furthermore, the number of different robot configurations is in general exponential with the number of modules. Even with few modules, it is intractable to manually explore the whole design space of robot morphologies and controllers by hand. Further, modular robots may decide to self-reconfigure or be manually reconfigured, and modules may realistically break down or be reset. Hence, the learning strategy must be robust and able to adapt to such events. Therefore, we are exploring morphology-independent learning, where we do not tie the learning strategy and underlying gait generator to a specific robot morphology. This strategy is different from most related work on monolithic robots, where the learning space can be reduced, e.g., from kinematic models or by exploiting symmetries.

We anticipate that such attractive features can be obtained by utilizing a simple, distributed learning strategy. We let each module locally optimize its behavior independently and in parallel with the others based on a single, broadcast reward signal which indirectly is used to optimize the global gait of the robot. The

* Corresponding author. Tel.: +45 26850903.
E-mail addresses: djchr@elektro.dtu.dk, Davidjohan@gmail.com (D.J. Christensen), ups@mmmi.sdu.dk (U.P. Schultz), kaspers@mmmi.sdu.dk (K. Stoy).

0921-8890/$ – see front matter © 2013 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.robot.2013.05.009

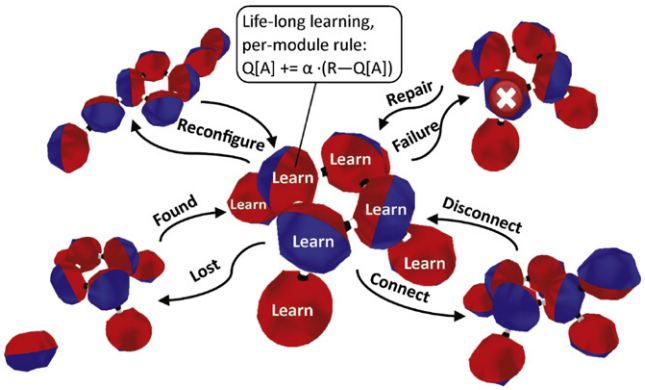


Fig. 1. Illustration of concept. The minimal strategy demonstrated in this paper is: (i) distributed, since the modules learn independently and in parallel, and (ii) morphology-independent, since the robot may self-reconfigure, modules may break down, be removed or added to the robot.

learning is life-long in the sense that there is no special learning phase followed by an exploitation phase. We hypothesize that a distributed strategy may be more robust and flexible since it can be indifferent to the robot's morphology and can adapt online to module failures or morphology changes. Ultimately, we anticipate that by studying distributed learning, we may gain insights into how adaptive sensory-motor coordination, such as locomotion patterns, can emerge and self-organize in animals.

1.1. Summary of contributions

This paper contributes an experimental study of a distributed online learning strategy for modular self-reconfigurable robots. The proposed strategy is minimalistic and distributed, and we present experiments both in simulation and on physical robots, that demonstrate that the strategy is morphology independent, fault-tolerant, generalizes to several module types, and is able to learn effective locomotion gaits in a matter of minutes. We also illustrate the limitations of the strategy, how it is not appropriate for all robot morphologies, and how it has limited scalability. This paper provides a comprehensive report of our previous work [1,2] and in addition presents a more exhaustive survey of related work, including more analysis of the strategy, and additional experiments with gait-table learning, tolerance of module loss, and scalability analysis which are previously unpublished.

1.2. Outline

The paper is organized as follows. First, we review related work in Section 2. Then we describe the learning strategy and how it can be applied to learn both action-based controllers as well as gait control tables (Section 3). The experimental platforms and the experimental setups are described in Sections 4 and 5 respectively. Further, a number of experiments are presented in Section 6. Finally, we discuss the results obtained in Section 7 and give conclusions in Section 8.

2. Related work

2.1. Self-reconfigurable modular robots

The concept of modular self-reconfigurable robots was introduced in the late 80's by Fukuda and Nakagawa [3]. Compared to conventional robots the motivations for such robots was: optimal shape in a given situation, fault tolerance and the ability to self-repair [4]. Since then more than two dozen systems have been designed and implemented, and the field is still growing as indicated in recent surveys [5–8].

Noteworthy systems developed during the first decade of research includes: the mobile CEBOT [4], chain-style modular robots [9–11], planar self-reconfigurable systems [12–14] and 3D self-reconfigurable systems [15–17].

Recent systems which are able to self-reconfigure in 3D and form mobile robots include M-TRAN III [18], SuperBot [19], CKBot [20], Roombots [21], ATRON [22], Replicator-Symbrion [23], Sambot [24], UBot [25], Cross-Ball [26] and SMORES [27].

2.2. Locomotion strategies for self-reconfigurable modular robots

In the context of modular robots, locomotion was first studied by Yim [28] using the Polypod robot which could be configured as a snake-like chain, hexapod, and a rolling track. Since then numerous types of gaits have been implemented on various robot configurations. Especially chain- and hybrid-style systems have proven appropriate for constructing various different robot morphologies that are able to perform locomotion. Moreover, by utilizing self-reconfiguration the robots can transform between different morphologies as first demonstrated by Kurokawa et al. [29]. Below we review some of the more generic locomotion strategies proposed.

Gait control tables is a generic strategy which was first used to control Polypod robots [30]. A gait control table is a two-dimensional table of actuator actions (e.g., a goal position or to act as a spring), where the columns are identified with a actuator id and the rows are identified with a condition for transition to the following row (e.g., a time-stamp or a sensor condition). To produce periodic locomotion gaits, the table is evaluated in a cyclic fashion.

Hormone-based control is a strategy, proposed by Shen et al. [31], which enable information to be distributed between the modules. Digital hormones represent information that propagates through the configuration of modules. Using digital hormones modules can detect topological changes, coordinate and synchronize actions for locomotion or self-reconfiguration [32].

Role-based control was originally developed to control locomotion of CONRO robots [33]. Each module, dependent on its position in the configuration tree, automatically selects a role. This allows a CONRO robot to shift from one locomotion style to another when it is manually reconfigured [34].

Phase automata was a generalization of role-based control developed to control the locomotion of the later Polybot system [35,36]. Phase automata patterns are state machines that define the actions of a module in a given state [37]. The shift from one state to another is event (time or sensor) driven.

Central Pattern Generators (CPGs) are a biologically inspired strategy which is a popular approach to locomotion control also for monolithic robots [38]. Genetically optimized CPGs were used to control M-TRAN walkers and snakes by Kamimura et al. [39,40]. Similarly, CPGs controlling the YaMoR modular robot were genetically evolved and online optimized in order to achieve locomotion by Marbach and Ijspeert [41]. CPGs were also use to control a simulated quadruped with 390 Catom modules [42]. Moreno and Gomez [43] combined CPGs with hormone-based control to integrate sensor feedback in the control system.

Cluster-flow locomotion is a radically different mode of locomotion based on continuous self-reconfiguration of the modules to move the collective ensemble. Such cluster-flow locomotion can be realized as a highly scalable emergent behavior arising from simple, distributed, rule-based control, as demonstrated on the sliding cube model [44–46] and on the ATRON modules [47]. For the M-TRAN system, cluster-flow locomotion has been realized based on a two-layer centralized planner by Yoshida et al. [48,49] and Kurokawa et al. [18], who demonstrated distributed cluster-flow on physical MTRAN modules. Related work on the Slimebot demonstrates a similar form of locomotion where self-reconfiguration is an emergent process that arises from local module oscillation, resulting in phototaxis [50].

Please cite this article in press as: D.J. Christensen, et al., A distributed and morphology-independent strategy for adaptive locomotion in self-reconfigurable modular robots, Robotics and Autonomous Systems (2013), http://dx.doi.org/10.1016/j.robot.2013.05.009

2.3. Adaptive self-reconfigurable modular robots

The polymorphic and metamorphic nature of self-reconfigurable modular robots gives unique constraints and opportunities for utilizing adaptive strategies, such as evolutionary and learning, for locomotion control.

Sims [51] pioneered the field of evolutionary robotics by co-evolving the morphology and control of simulated modular robots. Later works succeeded in transferring co-evolved robots from simulation to hardware [52,53]. For M-TRAN robots evolutionary algorithms were utilized to produce locomotion and self-reconfiguration [54], and Kamimura et al. [40] evolved the coupling parameters of central pattern generators for straight line locomotion. Similarly, Handier and Hornby [55] evolved a sinusoidal gait for a quadruped modular robot. Pouya et al. [56] used particle swarm optimization to optimize CPG-based gaits for Roombot robots that both contained rotating and oscillating actuators. Evolutionary algorithms have also been used to optimize reactive locomotion controllers based on HyperNEAT [57], chemical hormones models [58] and fractal genetic regulatory network [59]. For co-evolution of morphology and control, Brunete et al. [60] represented the morphology of a heterogeneous snake-like microrobot directly in the chromosome, Faíña et al. [61] represented a legged robot morphology as a tree, and Bongard and Pfeifer [62] evolved a genetic regulatory that would direct the growth of the robot instead of a direct representation.

Although appealing, one challenge with evolutionary approaches is to bridge the reality gap [63] and once transferred the robot is typically no longer able to adapt.

To address this limitation, evolutionary optimization of gaits can be performed directly on the physical robot [64–66]. However, sequential evaluation of a population of gaits is not fitting for life-long learning since it requires a steady convergence to a single gait.

Instead, life-long learning strategies can be utilized. Marbach and Ijspeert [41] used a life-long strategy, based on Powell's method, which performed a localized search in the space of selected parameters of central pattern generators. Parameters were manually extracted from the YaMoR modular robot by exploiting symmetries. Follow-up work by Sproewitz et al. [67] demonstrated online optimization of 6 parameters on a physical robot in roughly 25–40 min. As is the case in our paper, they try to realize simple, robust, fast, model-less, life-long learning on a modular robot. The main difference is that we seek to automate the controller design completely in the sense that no parameters have to be extracted from the symmetric properties of the robot.

Further, our approach utilizes a form of distributed reinforcement learning. A similar approach was taken by Maes and Brooks [68] who performed distributed learning of locomotion on a 6-legged robot. The learning was distributed to the legs themselves. Similarly, in the context of multi-robot systems, distributed reinforcement learning has been applied for learning various collective behaviors [69].

The strategy proposed in this paper is independent from the robot's morphology. Similarly, Bongard et al. [70] demonstrated learning of locomotion and adaptation to changes in the configuration of a modular robot. 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: Our strategy is simple, model-less and computational cheap to allow implementation on resource constrained modular robots.

In recent work we have studied distributed and morphology-independent learning based on a CPG architecture optimized online by a stochastic approximation method [71,72]. Our paper utilizes a strategy which has a simpler implementation and is appropriate for optimization of discrete control parameters. The CPG-based strategy is more complex and allows optimization of continuous control parameters. Both strategies illustrate the advantages of utilizing a distributed adaptation and control strategy in modular robots.

Algorithm 1 Normal Learning Strategy.

```
Initialize Q[A] = R, for all A evaluated in random order
loop
  Select Action A with max Q[A] with prob. 1 − ϵ, otherwise random action
  Execute Action A for T seconds
  Receive Reward R
  Update Q[A]+ = α · (R − Q[A])
end loop
```

3. A strategy for learning locomotion gaits

In this section we describe the basic reinforcement learning strategy and propose a heuristics for accelerated learning. Further, we explain how the strategy can be applied to optimize gait control tables.

3.1. Normal learning strategy

We utilize a simple stateless reinforcement learning strategy that is executed independently and in parallel by each module of the robot (see Algorithm 1). Each module must select which action, A, to perform amongst a small fixed set, $S = \{A_x, A_y, A_z, \ldots\}$, of possible actions. In the initialization phase each module executes each of these actions in random order and initializes the expected reward, $Q[A]$, with the global reward, R, received after performing action A. Note that the time it takes to perform this initialization is independent of the number of modules since the robot will not try every possible combination of module actions. After this phase, in a learning iteration, each module will perform an action, A, and then receive a global reward, R, for that learning iteration. The discounted expected reward, $Q[A]$, is estimated by an exponential moving average with a smoothing factor, α, which suppresses noise and ensures that if the reward of an action changes with time so will its estimation. Each module independently selects which action to perform based on an ϵ-greedy selection policy, where a module selects the action with highest estimated $Q[A]$ with a probability of $1 - \epsilon$ and a random action otherwise. The algorithm can be categorized as temporal difference learning ($TD(0)$) with discount factor $\gamma = 0$ and with no representation of the sensor state [73].

3.2. Accelerated learning strategy

We observe that for a locomotion task, the performance of a module is tightly coupled with the behavior of the other modules in the robot. Therefore, the best action of a module changes over time in response to the other modules changing their actions. The learning speed is limited by the fact that it must rely on randomness to select a fitter but underestimated action a sufficient number of times before the reward estimation becomes accurate. Let us now assume that the robot performs an action, A, and receives a reward, R, which is fixed and noise free. Then, we can compute the number of iterations, k_{ema}, required for the exponential moving average of $Q[A]$ to estimate R with a give precision e:

$$k_{ema} = \frac{\log(e)}{\log(1-\alpha)} \tag{1}$$

Please cite this article in press as: D.J. Christensen, et al., A distributed and morphology-independent strategy for adaptive locomotion in self-reconfigurable modular robots, Robotics and Autonomous Systems (2013), http://dx.doi.org/10.1016/j.robot.2013.05.009

Algorithm 2 Accelerated Learning Strategy.

```
Initialize Q[A] = R, for all A evaluated in random order
loop
  if max(Q) < R then
    Repeat Action A
  else
    Select Action A with max Q[A] with prob. 1 − ϵ, otherwise
    random action
  end if
  Execute Action A for T seconds
  Receive Reward R
  Update Q[A]+ = α · (R − Q[A])
end loop
```

where e is the final estimation error in percent relative to initial error. By Taylor expansion we obtain:

$$k_{ema} \approx \frac{-\log(e)}{\alpha}. \tag{2}$$

Taking into account the ϵ-greedy selection policy, assuming that $Q[A]$ is underestimating R, the number of required learning iterations, k_{learn} becomes:

$$k_{learn} \approx \frac{-\log(e)}{\epsilon \cdot \alpha}. \tag{3}$$

Since $\epsilon, \alpha < 1$ we can expect a slow convergence to R. Therefore, to accelerate the convergence we tested a simple heuristics (see Algorithm 2): If the received reward, R, after a learning period is higher than the highest estimation of any action, $\max(Q)$, the evaluated action may be underestimated and fitter than the current highest estimated action. Therefore, we repeat the potentially underestimated action, A, to accelerate the estimation convergence. In the accelerated case the number of required learning iterations then becomes:

$$k_{learn} = k_{ema} \approx \frac{-\log(e)}{\alpha}. \tag{4}$$

For example, with $\alpha = 0.1$ and $\epsilon = 0.2$, the number of learning iterations to reach an estimation within 5% of the correct reward is $k_{learn} \approx 148$ and $k_{learn} \approx 28$ for the normal and accelerated learning strategy respectively. In general, the acceleration heuristic will speed up the estimation of a fitter but underestimated action by a factor of $1/\epsilon$. However, note that the acceleration heuristic is trading off exploration for exploitation, which may cause the strategy to stay longer in local optima. In Section 6 we will experimentally compare the two strategies.

3.3. Learning strategy: applied to gait control tables

In Algorithms 1 and 2 each module learns to always perform a single action such as rotating clockwise. To enable a more versatile learning strategy that may work on any module type, we combine this strategy with gait control tables [10]. As explained in Section 2.2, gait control tables are a generic strategy that can be used to realize locomotion on most modular robots. In our work we optimize gait control tables that contain actuator set-points and which transition to the next row based on fixed time intervals.

To learn the set-points in a gait-table, we let each module learn the set-points of its own columns in the gait-table. Each module runs one parallel learning process per entity in its column. Each learning process selects a set-point, A, amongst a set of predefined set-points, S. The learning processes learn independently and in parallel based on a shared reward signal. This extended strategy is illustrated in Fig. 2. To utilize this approach for a given system we must define the set of set-points that can fill the table and the number of rows in the table, the number of columns is indirectly automatically adjusted when adding or removing modules.

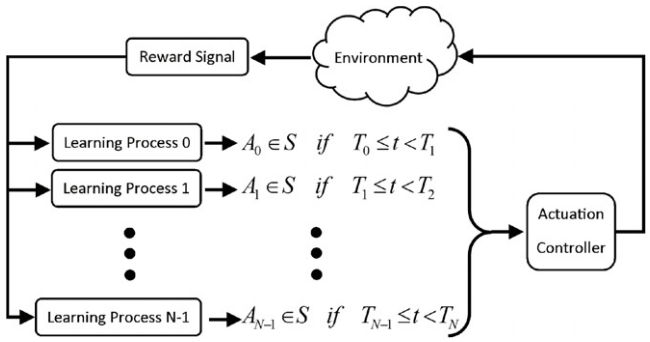


Fig. 2. Illustration of gait-table based learning strategy for a single module. Independent and parallel learning processes learn each entry, $A \in S$, in the gait-table. Therefore, each module learns its own column of the gait-table.

4. Hardware and simulation platforms

4.1. The ATRON self-reconfigurable modular robot

The ATRON self-reconfigurable robotic system is a homogeneous modular system, which means that all modules are identical in both hardware and typically also in software [22]. We have manufactured 100 ATRON modules; a single module is shown in Fig. 3(a). Modules can be assembled into a variety of robots: Robots for locomotion (e.g. snake-like chain, wheeled robot, and legged robot as shown in Fig. 3(b)), robots for manipulation (e.g. small robot arms) or robots that achieve some functionality from their physical shape, such as structural support (see Fig. 3(c)) [74,75]. By self-reconfiguring, a group of modules can change their shape, for example from a wheeled robot to a snake-like robot to a legged robot.

An ATRON module has a spherical exterior composed of two hemispheres, which can actively rotate relative to each other (see Fig. 3(a)). On each hemisphere, a module has two actuated male connectors and two passive female connectors. Rotation around the center axis is, for self-reconfiguration, always done in 90° steps. This moves a module, connected to the rotating module, from one lattice position to another. A 360° rotation takes approximately 6 s. Encoders sense the rotation of the center axis. Male connectors are actuated and shaped as three hooks, which grasp on to passive female connector bars. A connection or a disconnection takes about two seconds. Located next to each connector are an infrared transmitter and receiver, which allow modules to communicate with neighbor modules and sense distance to nearby objects. Connector positions and orientations are such that the ATRON modules sit in a global surface-centered cubic lattice structure. Furthermore, each module is equipped with tilt sensors that allow the module to know its orientation relative to the direction of gravity.

4.2. Unified simulator for self-reconfigurable robots

Simulation experiments are performed in an open-source simulator named Unified Simulator for Self-Reconfigurable Robots (USSR) [76]. We have developed USSR as an extendable physics simulator for modular robots. The simulator is based on Open Dynamics Engine [77] which provides simulation of collisions and rigid body dynamics. Through socket connections USSR is able to run module controllers which can also be cross-compiled for some of the physical platforms.

USSR includes implementations of several existing modular robots, e.g., ATRON and M-TRAN as utilized in this paper. The ATRON module is modeled in the simulator and the model parameters, e.g., strength, speed, weight, etc., have been calibrated

Please cite this article in press as: D.J. Christensen, et al., A distributed and morphology-independent strategy for adaptive locomotion in self-reconfigurable modular robots, Robotics and Autonomous Systems (2013), http://dx.doi.org/10.1016/j.robot.2013.05.009

Fig. 3. (a) A single ATRON module: on the top hemisphere the two male connectors are extended, on the bottom hemisphere they are contracted. (b) Different configurations of ATRON modules. (c) Meta-modules for self-reconfiguration.

Table 1
Learning parameters.

Exp. type	Robot	α	$1-\epsilon$	T (s)
Action-based	ATRON	0.1	0.8	7
Gait-table	ATRON	0.1	0.96	7
Gait-table	M-TRAN	0.0333	0.96	1.5

to match the existing hardware platform to ease the transfer of controllers developed in simulation to the physical modules. The M-TRAN module fills two cells in a cubic lattice and has six connector surfaces. Each module has two actuators which can rotate in the interval $\pm 90^\circ$. We have implemented a model of the M-TRAN III module in the USSR simulator based on available specifications [18]. However, we do not have access to the physical M-TRAN modules. Therefore, although the kinematics is correct, specific characteristics might be somewhat different from the real system. We accept this, since our purpose is to evaluate the learning strategy on a different system than the ATRON, not to find efficient transferable locomotion gaits for M-TRAN robots.

5. Experimental setups

5.1. Learning parameters and reward signal

In the physical and the simulated experiments, described in Section 6, each module runs identical learning controllers with parameters set as indicated in Table 1. In some experiments we compare with randomly moving robots, i.e. we set $1-\epsilon = 0$ in Algorithm 1.

Each module, part of a robot, shares and optimizes its behavior based on the same reward signal. The reward signal is a measurement of the velocity, which we estimate as the distance moved by the robot's center of mass in T s:

$$R = |p_{t+T} - p_t|/T. \quad (5)$$

Generally, to reduce noise in the reward signal, T must be a multiple of a full gait period, i.e., the time it takes a module to rotate 360° or oscillate back and forth. Here, T is selected to be a single full gait period, which we found gave sufficiently high signal-to-noise ratio to achieve convergence in the presented experiments. α balances the amount of noise in the reward signal against convergence time, and ϵ controls the exploration/exploitation trade-off. The learning parameters in Table 1 were tuned in simulation for reliable and fast convergence. Note that the relationship between experimental time, iteration time, T, and number of iterations, k, is simply: time $= k * T$.

5.2. Action space

To utilize the learning strategy we must define a set of actions or gait-table set-points. We select actions/set-points that are module-type specific but somewhat generic with respect to robot morphology.

In the action-based experiments we utilize ATRON modules that always perform one of the following three actions:

- *HomeStop* (rotates to 0° and stop).
- *RightRotate* (rotate clockwise 360°).
- *LeftRotate* (rotate counter-clockwise 360°).

Therefore the basic action set for ATRON is:

$S = \{HomeStop, RightRotate, LeftRotate\}$.

After a learning iteration, a module should ideally be back at its home position to ensure repeatability. Therefore, the modules will synchronize their progress to follow the rhythm of the learning iteration based on the shared reward signal.

In the gait-table experiments, to study how the strategy works on a system with a different kinematics, we will use joint-limited ATRON modules that are limited to rotate within a $\pm 90^\circ$ interval. In effect, the gait-table-based learning strategy must find gaits based on oscillations to move the robots, instead of gaits based on continuous rotations. For each robot morphology we define an initial pose that the actuation is performed relative to. The gait-table has five rows, so each module must learn five angle values from the set-point set:

$S = \{-60, -30, 0, 30, 60\}$ degrees.

Fig. 18 shows examples of ATRON gait-tables.

The M-TRAN module has two actuators. Therefore we let each actuator be controlled by independent gait-tables. Each gait-table has five rows, where an entry can contain one of three set-points:

$S = \{-30, 0, 30\}$ degrees.

In general, selecting an initial pose and the set-point sets is a tradeoff between high potential to move and being stable so that the robot does not fall over while learning. Section 7 discuss such assumptions and limitations in more detail.

5.3. Physical setup

The ATRON modules are not equipped with a sensor that allows them to measure their own velocity or distance traveled. Instead, we have constructed a setup, which consists of an arena with an overhead camera connected to a server (see Fig. 4). The server tracks the robot, computes the robot's velocity, and sends the reward signal wirelessly to the robot.

The accelerated learning algorithm, as specified in Algorithm 2, is running on the physical modules. At a frequency of 10 Hz each module sends a message containing its current state (paused or learning), time-step and reward to all of its neighbors through its infra-red communication channels. The time-step is incremented and the reward is updated from the server side every $T = 7$ s. When a new update is received, a module performs a learning update and start a new learning iteration. The state can from the

Please cite this article in press as: D.J. Christensen, et al., A distributed and morphology-independent strategy for adaptive locomotion in self-reconfigurable modular robots, Robotics and Autonomous Systems (2013), http://dx.doi.org/10.1016/j.robot.2013.05.009

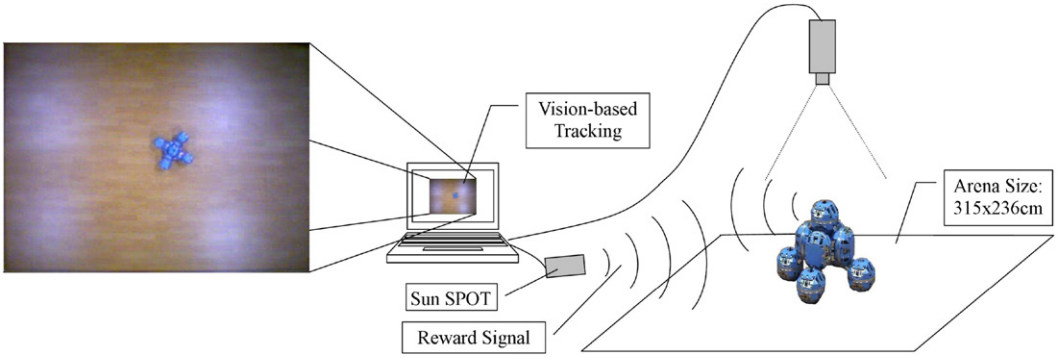
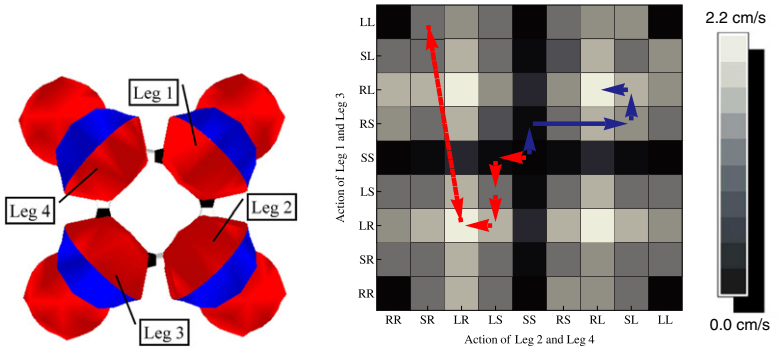
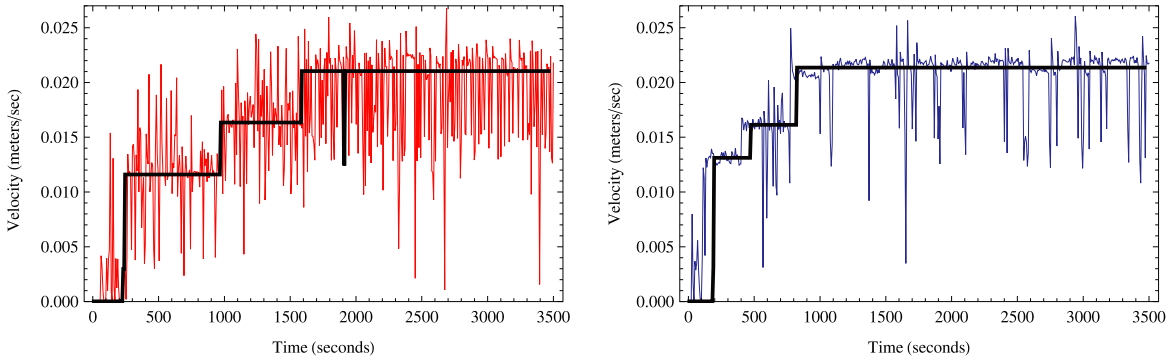


Fig. 4. Experimental setup of online learning.



(a) Quadrupedal. (b) Path through control space.



(c) Normal learning. (d) Accelerated learning.

Fig. 5. Typical simulated learning examples with and without the acceleration heuristic. (a) Eight module quadrupedal robot (four active modules). (b) Contour plot with each point indicating the velocity of a robot performing the corresponding controller (average of 10 trials per point). The arrows show the transitions of the preferred controller of the robot. (c) And (d) shows the corresponding rewards received by the robots during 1 h. The horizontal lines indicate the expected velocity based on the same data as the contour plot.

server side be set to *paused* or *learning*. The robot is paused by the server when it moves beyond the borders of the arena and is then manually moved back onto the arena before the learning is continued. In the presented results, the paused time intervals have been removed.

6. Experiments

6.1. A typical ATRON robot

In this experiment we study how the normal and accelerated action-based learning strategy behaves on a typical modular robot. We consider a simulated quadrupedal consisting of 8 ATRON modules. To facilitate the analysis we simplify and control the experimental variables and the initial conditions: (i) we disable four of the modules (i.e., they are stopped in the home position) and only allow the four modules acting as legs to be active, as indicated in Fig. 5(a). (ii) Also, we force the robot to start learning from a completely stopped state by initializing $Q[A]$ to 0.1 for the HomeStop action and to 0.0 for the other actions. Note, that these two simplifications will prolong the convergence time (robot will seldom move initially) and make the robot less able to locomote (only four of eight modules are active). The motivation is that the result of each trial will be less dependent on starting conditions and the gait learning analysis will be simpler. Therefore, we can compare the two strategies more directly and analyze their behavior. The other experiments described in this paper do not make these simplifications.

First, consider the two representative learning examples given in Fig. 5. The contour plot in Fig. 5(b) illustrates the fitness landscape (found using systematic search) and how an example robot controller transitions to gradually improve itself. The controller

Please cite this article in press as: D.J. Christensen, et al., A distributed and morphology-independent strategy for adaptive locomotion in self-reconfigurable modular robots, Robotics and Autonomous Systems (2013), http://dx.doi.org/10.1016/j.robot.2013.05.009

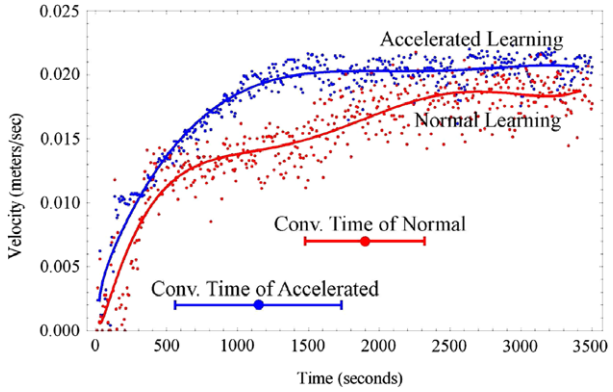


Fig. 6. The velocity of a quadrupedal robot with four active modules as a function of time. Each point is the average of 10 trials. The horizontal bars indicate average convergence time and standard deviation. Note that accelerated learning converges significantly faster ($P = 0.0023$) for this robot.

eventually converges to one of the four optimums, which corresponds to the symmetry axes of the robot (although in one case the robot has a single step fallback to another controller). The graphs in Fig. 5(c) and (d) show how the velocity of the robots jumps in discrete steps, that correspond to changes in the preferred actions of modules.

Fig. 6 compares the convergence speed and performance of the learning with and without the acceleration heuristic. The time to converge is measured from the start of a trial until the controller transitioned to one of the four optimal solutions (found with systematic search). In all 20 trials the robot converged, in 4 trials the robot had short fallbacks to non-optimal controllers (as in Fig. 5(c)). On average accelerated learning converged faster (19 min or 1146 iterations) than normal learning (32 min or 1898 iterations). The difference is statistically significant ($P = 0.0023$[1]). Note that accelerated learning on average reaches a higher velocity, but not due to the type of gaits found. Rather, the faster velocity is due to the acceleration heuristics, which tends to repeat good-performing actions at the cost of random exploration. This can also be seen by comparing the amount of noise in Fig. 5(c) with (d).

As summarized in Table 2, the learning strategy behaves in roughly the same way independently of the acceleration heuristic. A typical learning trial consists of 4–5 controller transitions,

Table 2
Average number of controller transitions to reach optimal solution, with standard deviations in parentheses. To measure the number of controller transitions, very brief transitions of one or two learning steps (7–14 s) are censored away. The results are based on 10 trials of quadrupedal with 4 active modules learning to move. Note, that there is no significant difference in the type of controller transitions. Also, 1-step transitions are by far the most common, which indicates that the search is localized.

	Normal	Accelerated
Transitions per trial	4.4 (1.17)	4.0 (0.94)
1-step transitions (%)	87	90
2-step transitions (%)	13	6
3-step transitions (%)	0	4
4-step transitions (%)	0	0

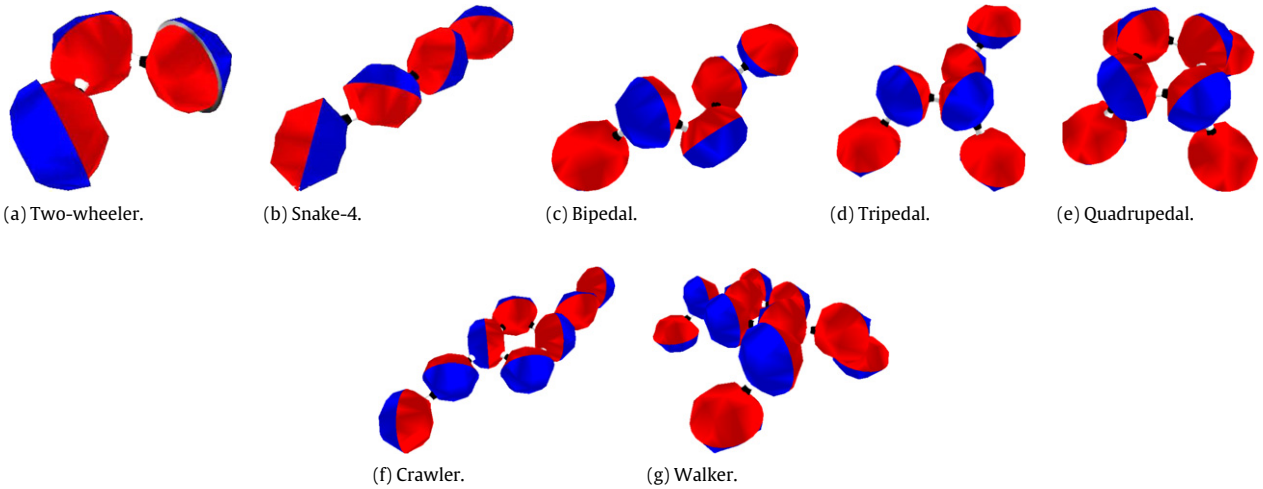
where a module changes its preferred action before the controller converges. In about 90% of these transitions it will only change the action of one module. This indicates that at a global level the robot is performing a *localized random search* in the controller space. Although the individual modules are not collectively searching in any explicit manner, this global strategy emerges from the local strategy of the individual modules.

6.2. Morphology independence

In this simulated experiment, we perform online learning with seven different simulated ATRON robots to study the degree to which the learning strategy is morphology independent (see Fig. 7). In each trial, the robot had 60 min to optimize its velocity. For each robot type 10 independent trials were performed with the normal, accelerated and random strategy. Results are shown in Fig. 8.

Compared to randomly behaving robots, both normal and accelerated learning improves the average velocity significantly. We observe that each robot always tends to learn the same, i.e., symmetrically equivalent gaits. There is no difference in which types of gaits the normal and accelerated learning strategy finds. Overall, the learning of locomotion is effective and the controllers are in most cases identical to those we would design by hand using the same action primitives. A notable exception is the snake robot which has no known good controller given the current set of action primitives. The other robots converged within 60 min to best-known gaits in 96% of the trials (115 of 120 trials). Convergence time was on average less than 15 min for those robots, although single trials would be caught in suboptimal solutions for extended time periods. We found no general trend in the how the morphology affects the learned gaits. For example, there is no trend that smaller robots or larger robots are faster, except that wheeled locomotion is faster than legged locomotion.

[1] Using Student's t-test.



(a) Two-wheeler. (b) Snake-4. (c) Bipedal. (d) Tripedal. (e) Quadrupedal.

(f) Crawler. (g) Walker.

Fig. 7. Seven learning ATRON robots consisting of 3–12 modules.

Please cite this article in press as: D.J. Christensen, et al., A distributed and morphology-independent strategy for adaptive locomotion in self-reconfigurable modular robots, Robotics and Autonomous Systems (2013), http://dx.doi.org/10.1016/j.robot.2013.05.009

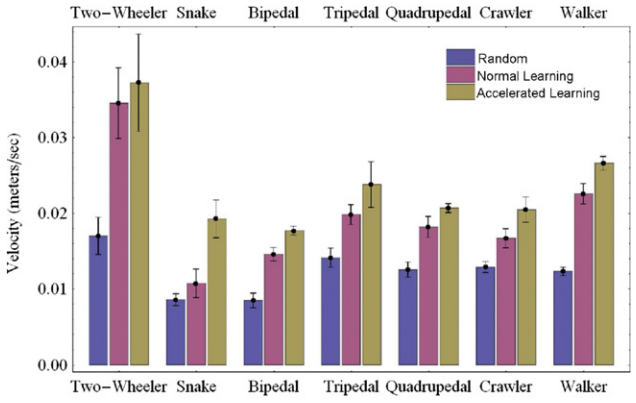


Fig. 8. Velocity at the end of learning in simulation. Each bar is the average velocity (reward) from the 50th to the 60th minute of 10 independent trials. Error bars indicate one standard deviation of average robot velocity. Note that both normal and accelerated learning has an average higher velocity than random movement.

6.3. Physical ATRON robots

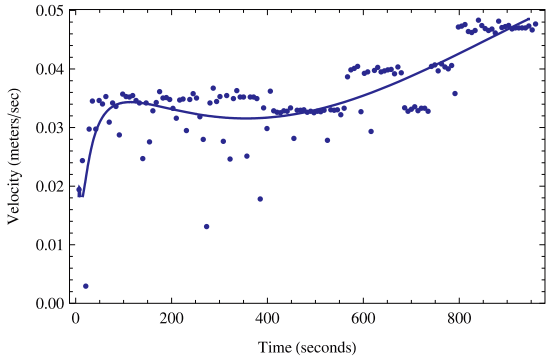
In this section we present experiments with physical ATRON robots to investigate if the accelerated learning strategy can successfully be transferred from simulation to physical robots. The learning strategy is executed by the modules and only the reward signal is computed externally. We utilize two different robots, a three-module two-wheeler and an eight-module quadrupedal, which has a passive ninth module used only for wireless communication. For each robot, we report on five experimental trials, two extra experiments (one for each robot) were excluded due to mechanical failures during the experiments. An experimental trial ran until the robot had converged to a near optimal gait (which were known from the equivalent simulation experiments) and stayed unchanged for several minutes. Since not all physical experiments are of equal duration, we extrapolate some experiments with the average velocity of its last 10 learning iterations to generate the graphs of Fig. 9(a) and (c). In total, we report on more than 4 h of physical experimental time.

6.3.1. Two-wheeler

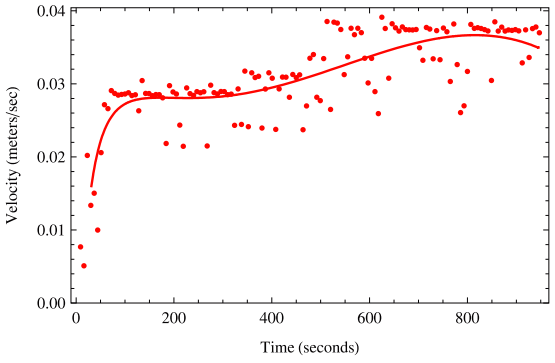
Table 3 shows some details for five experimental trials with a two-wheeler robot. The time to converge to driving either forward or backward is given (i.e., the fastest known gaits for the two-wheeler). For comparison the equivalent convergence time measured in simulation experiments is also given. In three of the five experiments, the robot converges to the best-known solution within the first minute. As was also observed in simulation trials, in the other two trials the robot was stuck for an extended period in a suboptimal behavior before it finally converged. We observe that the physical robot on average converges a minute slower than the simulated robot, but there is no significant difference ($P = 0.36$) between simulation and physical experiments in terms of mean convergence time. Fig. 9 shows the average velocity (reward given to the robot) as a function of time for the two-wheeler in both simulation and on the physical robot. The found gaits are symmetrically identical, however the physical robot moves faster than in simulation due to simulator inaccuracies.

6.3.2. Quadrupedal

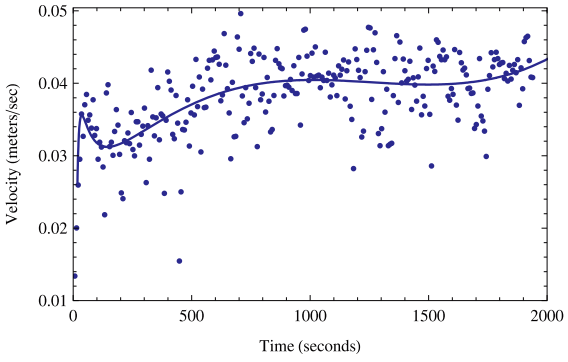
Pictures from an experimental trial are shown in Fig. 10, where a 9-module quadrupedal (8 active modules and 1 for wireless communication) learns to move. Table 3 summarized the result of five experimental trials. In all five trials, the robot converges to a best-known gait (symmetrically identical gaits to those found in simulation). The average convergence time is less than 15 min, which is slower than the average of 12 min it takes to converge



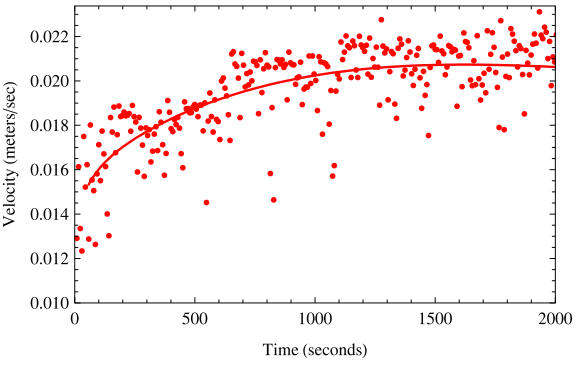
(a) Physical two-wheeler.



(b) Simulated two-wheeler.



(c) Physical quadrupedal.

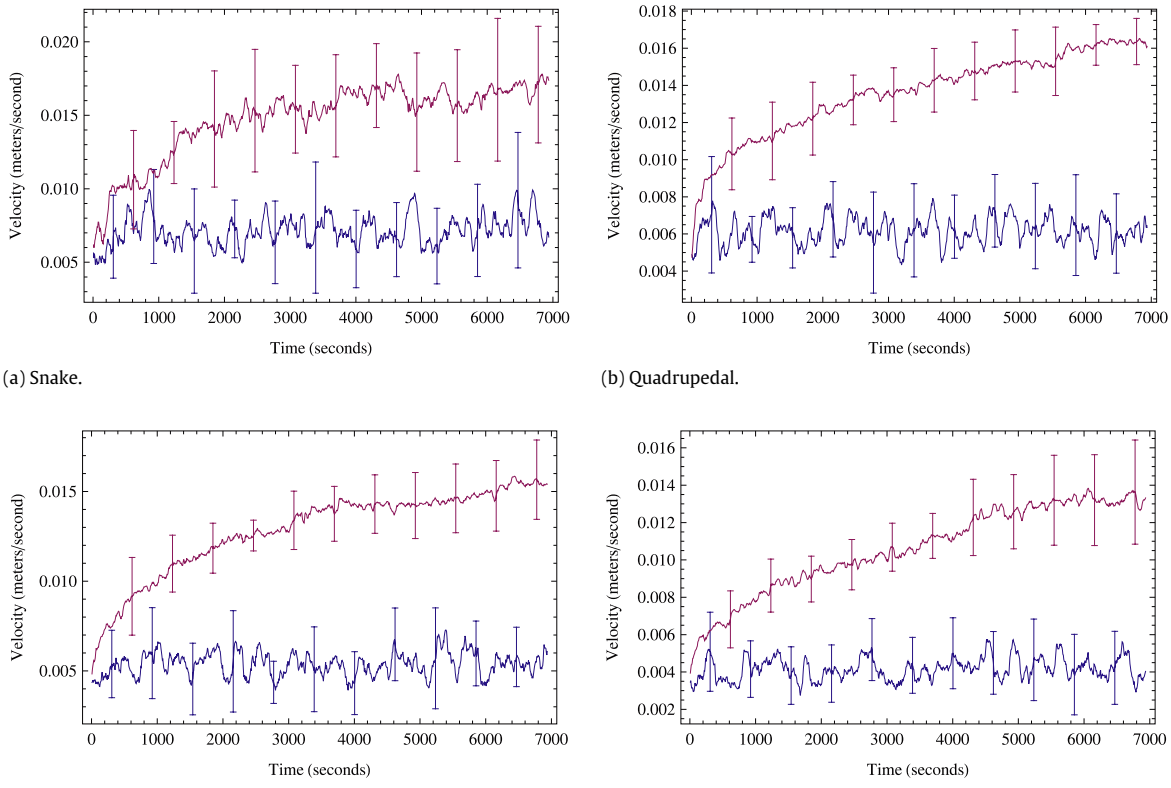


(d) Simulated quadrupedal.

Fig. 9. Average velocity of five trials as a function of time for both physical and simulated experiments for a two-wheeler and a quadrupedal. Points are the average reward in a given timestep and the lines indicate the trend.

Please cite this article in press as: D.J. Christensen, et al., A distributed and morphology-independent strategy for adaptive locomotion in self-reconfigurable modular robots, Robotics and Autonomous Systems (2013), http://dx.doi.org/10.1016/j.robot.2013.05.009

Fig. 10. Pictures from learning experiment with quadrupedal walker. A 7 s period is shown. The robot starts in its home position, performs a locomotion period, and then returns to its home position. In each of the five experiments, the quadrupedal converged to symmetrically equivalent gaits. All five gaits were equivalent to the gaits found in simulation.



(a) Snake.

(b) Quadrupedal.

(c) Millipede-2.

(d) Millipede-3.

Fig. 11. Convergence graphs for the four different robots assembled from joint-limited ATRON modules. As a baseline the average velocity of random moving robots is also shown. Each graph is the average of 10 trials. Error bars indicate standard deviation.

Table 3
Results of online learning on two-wheeler and quadrupedal robots.

Exp.	Two-wheeler		Quadrupedal	
	Conv. time (s)	Exp. time (s)	Conv. time (s)	Exp. time (s)
1	28	629	1680	2 401
2	35	707	1232	2 157
3	567	967	448	1 891
4	28	695	700	2 236
5	798	1020	336	2 468
Total	1456	4018	4396	11 153
Phy. mean	**291**	804	**879**	2 231
Sim. mean	**225**	–	**701**	–

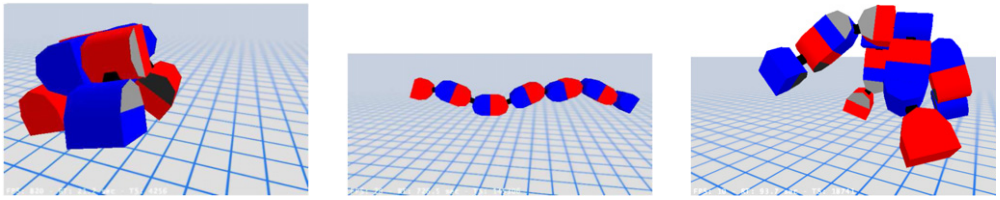
in simulation. The difference is, however, not statistical significant ($P = 0.29$). Fig. 9 shows the average velocity versus time for both simulated and physical experiments with the quadrupedal. We observe that the measured velocity in the physical trials contains more noise than the simulated trials. Furthermore, the physical robot also achieves a higher velocity than in simulation (due to simulator inaccuracies). Another observation we made was that the velocity difference between the fastest and the second-fastest gait is smaller in the physical experiments than in simulation, which together with the extra noise may explain why the physical trial on average converges almost 3 min slower than in simulation.

6.4. Gait-table learning with ATRON modules

In this experiment we apply the accelerated learning strategy on gait-control tables for an ATRON snake (chain with seven modules), millipede-2 (two leg pairs, 10 modules), millipede-3 (three leg pairs, 15 modules) and a quadrupedal (8 modules). We do this to study if the learning strategy can be utilized to optimize gait-tables while still being morphology independent.

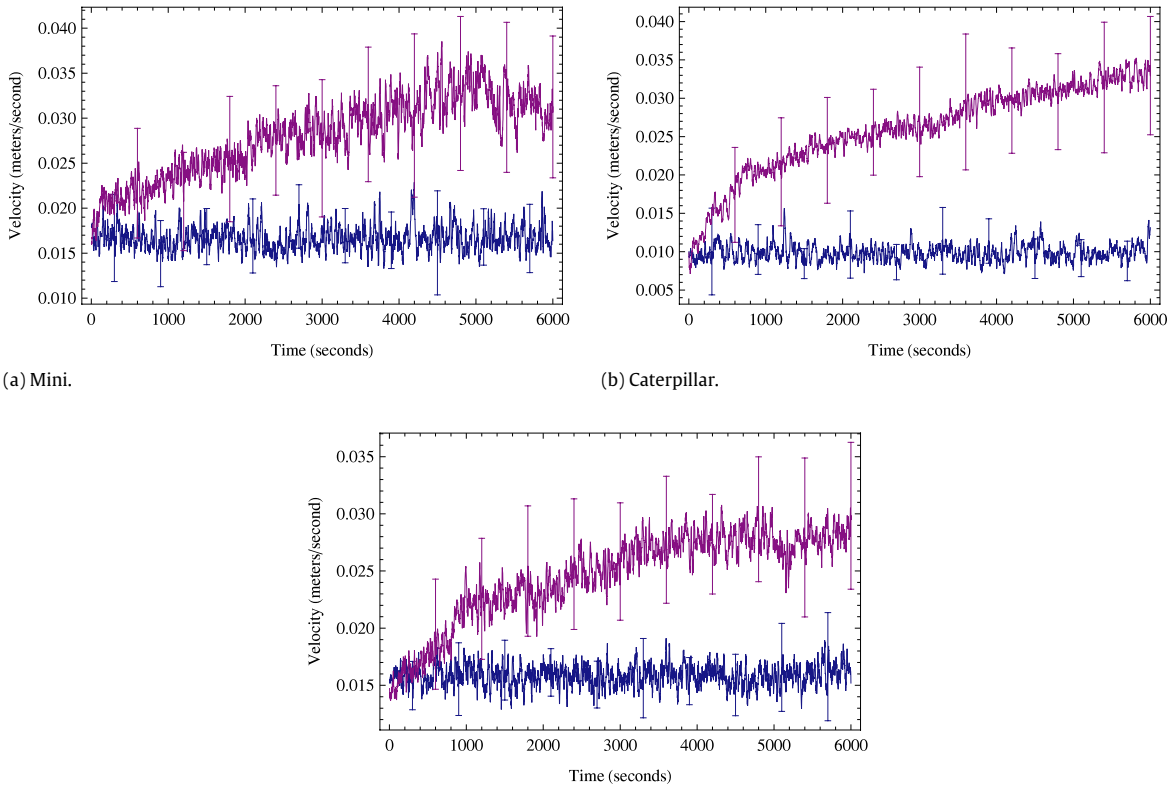
Fig. 11 shows the convergence as the average velocity achieved over time compared with randomly moving robots. Note that a robot tends to quickly learn a better than random gait, and that this gait gradually improves over time. Compared to blind random search the convergence speed is similar but the learning strategy finds significantly better gaits, e.g., on average 9.9 cm/s and 13.3 cm/s respectively for the millipede-3 robot. Although the robots moves slower than if they could perform unlimited rotation, the gaits found are quite efficient. Also note that in the case of the snake robot, the action-based learning strategy fails to converge since the robot entangles itself, while this gait-table based strategy

Please cite this article in press as: D.J. Christensen, et al., A distributed and morphology-independent strategy for adaptive locomotion in self-reconfigurable modular robots, Robotics and Autonomous Systems (2013), http://dx.doi.org/10.1016/j.robot.2013.05.009



(a) Mini. (b) Caterpillar. (c) Walker.

Fig. 12. Three M-TRAN robots consisting of 4, 6, and 8 modules.



(a) Mini. (b) Caterpillar.

(c) Walker.

Fig. 13. Average velocity as a function of time for three M-TRAN robots. Each graph is the average velocity of 10 independent trials. Average velocity of randomly moving robots is shown for comparison. Error bars indicate one standard deviation.

converges to undulation-style gaits. All the 40 experimental trials converged to good performing gaits. Divergence happens in a few cases, e.g., when a snake robot rolls upside down during learning and then had to learn to move from this state.

Some typical gaits found: The snake is moving with a side-winding gait, with a traveling wave down the chain. The snake lifts parts of its body off the ground as it moves. A typical quadrupedal gait could use a back foot partly as a wheel, partly as a foot. Its side legs moves back and forward for movement, while the front leg is used just for support. The millipede-2 has a trot style gait, where the diagonal opposite legs move together. The millipede-3 uses a similar gait with each leg oscillating back and forward with some unrecognizable scheme of synchronization between the legs.

6.5. Gait-table learning with M-TRAN

In this experiment we apply the gait-table-based learning strategy on three simulated M-TRAN robots (see Fig. 12): A 6-module caterpillar (12 DOF), a 4-module mini walker (8 DOF) and an 8-module walker (16 DOF). We perform the study to see if the learning strategy can be utilized on a different modular robotic platform than ATRON.

Fig. 13 shows convergence graphs for the three robots. Notice that the performance of the gaits quickly becomes better than random and that the gaits gradually improve over time. The learning succeeds in finding efficient gaits for all three robots. Because of the short learning iteration ($T = 1.5$ s), even a pose shift can be measured as quite high velocity. Therefore, randomly moving robots incorrectly seems to move quite fast, but this does not affect the learning strategy since it quickly converges to gaits with movement in a specific direction. We observe that the large learning space leaves room for incremental learning.

A major challenge with learning M-TRAN gaits is that the robot often falls over while learning. This happened in 23%, 8% and 47% of the two hour trials with the mini walker, caterpillar and walker respectively. These trials were censored away in the presented results, which is based on 10 completed trials per robot.

Some typical learned gaits: Typical gaits for the mini walker consist of hopping movement, with two modules producing movement and two modules creating stability. For the caterpillar, the learning typically finds gaits either with a horizontal traveling wave down the chain of modules or gaits that use the head and tail modules to push on the ground. Successful gaits for the walker take relatively short steps, since the robot would otherwise fall over. In

Please cite this article in press as: D.J. Christensen, et al., A distributed and morphology-independent strategy for adaptive locomotion in self-reconfigurable modular robots, Robotics and Autonomous Systems (2013), http://dx.doi.org/10.1016/j.robot.2013.05.009

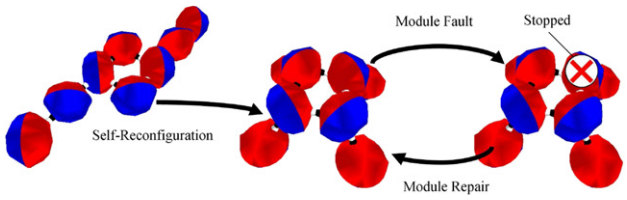


Fig. 14. Procedure for testing adaptation to self-reconfiguration and module fault. Initially the robot is of a crawler type, it then self-reconfigures to a quadrupedal, then a module fails and finally the module again becomes functional.

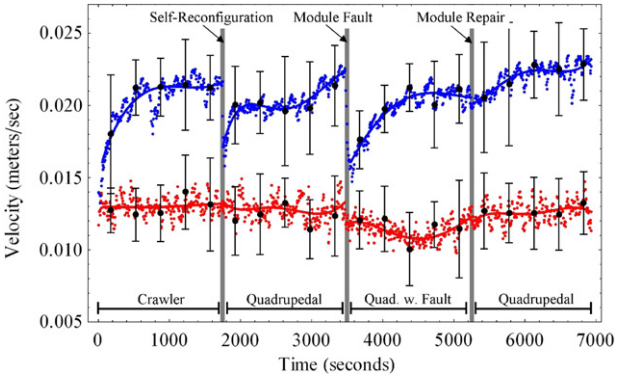


Fig. 15. Learning during robot self-reconfiguration and module fault. In each case, the learning enables the robot to adapt to the new situation by changing the locomotion gait. The graph is the average of 10 trials, with standard deviation as error bars. The bottom line is 10 trials of the equivalent robot moving randomly.

one example trial the walker used three legs to produce movement, while the fourth leg is kept lifted off the ground in front of the robot.

6.6. Self-reconfiguration and module fault

In this experiment, we study if the accelerated action-based learning strategy is able to adapt to changes in the robot morphology due to self-reconfiguration and module faults. Initially we let a crawler type robot (8 modules) learn to move (see Fig. 14). At learning iteration 250 (after 29 min), the robot is programmed to self-reconfigure into a quadrupedal type robot. Afterwards, the learning is continued without resetting the learning system. After an additional 250 iterations, we simulate a module failure by stopping a leg module in a non-home position. 250 iterations later, we reactivate the module and let the learning continue for another 250 iterations.

Fig. 15 shows the average results of 10 trials. After both the self-reconfiguration and the module fault, we observe a drop in fitness as expected. In both cases, the learning system is able to adapt to its changed morphology and regain a higher velocity. In the case where a leg module is reactivated there is no initial drop in fitness, but afterwards the robot learns again to use its leg and the average velocity increases.

6.7. Gait-table learning with module loss

In this experiment we study if a gait-table controlled ATRON quadrupedal robot is able to adapt to the loss of a module used as a lower limb. The gait-table is optimized using the accelerated learning strategy. First the quadrupedal learns to move for 1000 iterations. Then the robot disconnects a specific module thereby disabling its movement, as illustrated in Fig. 16. The robot is then given additional 1000 iterations to learn to move with this new morphology.

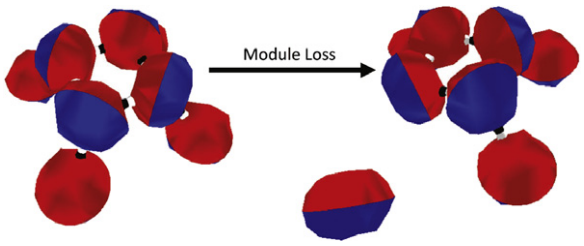


Fig. 16. Procedure for testing adaptation of gait-table to module loss. Initially the robot is of a quadrupedal type, it then disconnects a module as shown.

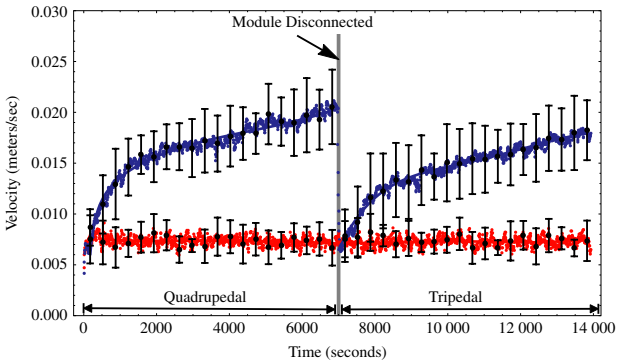


Fig. 17. Learning after loss of module using an ATRON quadruped controlled with a gait-table. The graph is the average of 10 trials, with standard deviation as error bars. The bottom line is 10 trials of the equivalent robot moving randomly.

Fig. 17 shows the average velocity of 10 independent trials. For the first 1000 iterations the robot gradually increases its velocity. Then the module is disconnected and we observe a drastic drop in velocity to the level of a randomly moving robot. The robot slowly adapts to its new morphology and regains the majority of its lost velocity. Fig. 18 shows two gait-tables from a typical trial. The first gait-table shown is just before the module is disconnected and the other gait-table is at the end of the trial. We observe that the two gait-tables contains mainly different joint values (77%) which may indicate that the two robots required quite different gait patterns for effective locomotion.

6.8. Scalability

We performed simulated experiments with a scalable millipede ATRON robot to study the scalability of the learning strategy (see Fig. 19). In the following experiments, we vary the number of legs from 4 to 36 in steps of 4 with 10 learning trials per robot.

For this specific experiment we define the time of convergence as the time at which 85% of the leg modules have learned to contribute to the robot movement. That is, the leg module rotates either left or right dependent on its position in the robot and the direction of locomotion. The time to converge is shown in Fig. 20(a). As expected, an increase in the number of modules also increases the convergence time, the relation is approximately linear for this robot in the interval shown. The increase in convergence time is rather slow, for each module added the convergence time is prolonged with 52 s (based on a least square fit):

$$\text{convergenceTime} = 52 \cdot \#\text{modules} + 182\ \text{s}. \quad (6)$$

Beyond this interval of up to 60 modules, divergence becomes the dominating factor, i.e., the robot forgets the already learned behavior.

We measure learning divergence as a major drop in number of leg modules contributing to moving the millipede. The frequency of diverges of each robot is shown in Fig. 20(b). We observe that the divergence frequency increases with the number of modules.

Please cite this article in press as: D.J. Christensen, et al., A distributed and morphology-independent strategy for adaptive locomotion in self-reconfigurable modular robots, Robotics and Autonomous Systems (2013), http://dx.doi.org/10.1016/j.robot.2013.05.009

	m0	m1	m2	m3	m4	m5	m6	m7
t0	30	-30	60	60	-60	30	30	30
t1	0	-60	0	60	-30	0	-60	30
t2	-60	-30	-60	60	-60	-60	30	0
t3	-30	30	30	-60	-30	-30	60	-60
t4	0	-60	60	-60	0	30	0	-30

(a) Before module loss.

	m0	m1	m2	m3	m4	m5	m6	m7
t0	30	-60	-60	30	*	-60	-30	30
t1	30	-30	0	-30	*	30	-30	0
t2	-60	-60	-60	30	*	-30	-30	-30
t3	-60	-30	30	-60	*	60	-60	30
t4	0	30	-30	0	*	60	-60	-60

(b) After module loss.

Fig. 18. Typical trial of gait-table learning with module loss experiment. (a) Gait-table for the eight modules (m0…m7) just before m4 is disconnected (iteration 1000). (b) Gait-table at end of trial (iteration 2000), 27 of 35 joint values in the table have changed as a result of the optimization (shaded cells).

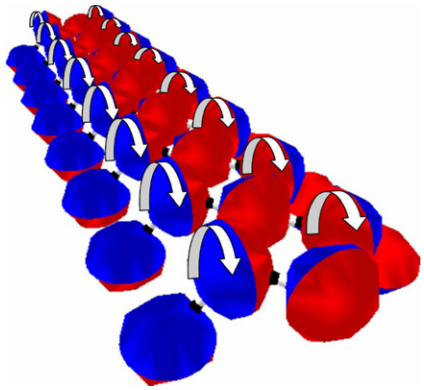
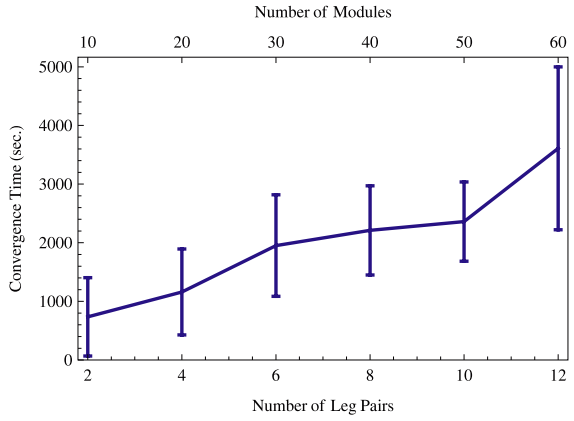


Fig. 19. Example of millipede robot with 8 leg pairs and 40 modules used to study scalability. Given the basic actions of ATRON, the best know controller rotates the legs as indicated.

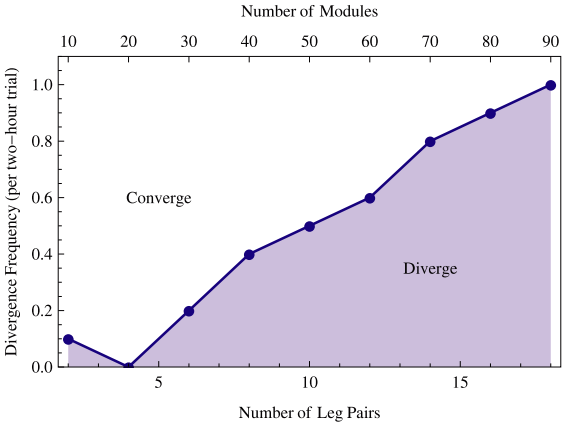
The reason behind this is that as the number of modules increases, the effect that any individual module has on the robot decreases. Therefore, for a given module the estimates for each of its actions will almost be identical and a small disturbance can cause the divergence effect. This decrease of the signal-to-noise ratio is illustrated in Fig. 21 for a 6-legged and 20-legged millipede. The graphs show the result of using a simple, regressing controller on the robots: Initially the robot is moving with the best know controller. Then, every 200 s, one of the leg modules reverses its action from rotating left to right and vice versa. This causes the robot to slow down its movement until it is almost fully stopped. For the small robot, the difference in velocity from one module change to the other is large and a module can easily detect the difference. For the large robot, this difference is small compared to noise, which explains why the divergence effect increases with the number of modules.

7. Discussion

Adaptive robots often utilize a centralized learning strategy. In this paper the learning is distributed to the individual modules

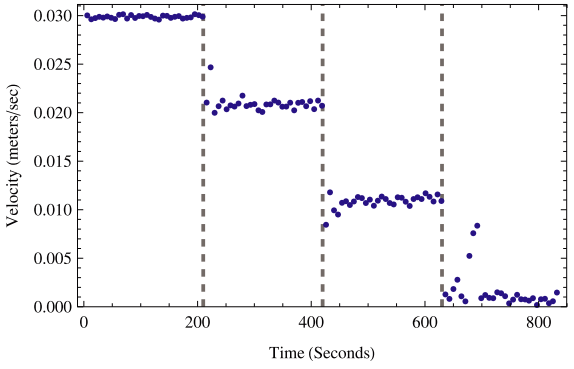


(a) Convergence.

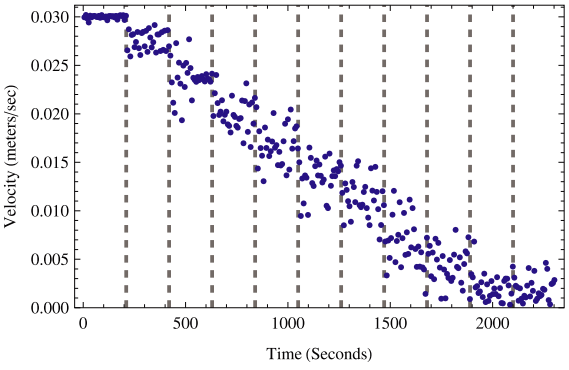


(b) Divergence.

Fig. 20. (a) Convergence time versus number of modules. (b) Divergence versus number of modules. The robots are millipedes with 4–24 legs. Error bars indicate one standard deviation.



(a) 6-leg millipede.



(b) 20-leg millipede.

Fig. 21. Divergence in large-scale robots. If the number of modules are low a single module may have a large effect on the overall velocity compared to noise, the opposite is the case for many module robots.

Please cite this article in press as: D.J. Christensen, et al., A distributed and morphology-independent strategy for adaptive locomotion in self-reconfigurable modular robots, Robotics and Autonomous Systems (2013), http://dx.doi.org/10.1016/j.robot.2013.05.009

which optimize the behavior independently and in parallel based only on a simple centralized reward signal. In Section 6.1 we observed that a centralized strategy similar to localized random search emerged from these distributed adaptive modules. The extent to which this emergent phenomenon can be utilized in other learning scenarios than locomotion is an open research question.

A robot controller is typically designed specifically for a robot's morphology. The proposed strategy does not make any assumptions about how the modules are connected to form a specific robot configuration. Combined with the distributed implementation, this makes the strategy morphology independent and it will automatically adapt to a large class of different morphologies, as was experimentally demonstrated in Section 6.2. For the same reasons the strategy is also able to adapt to dynamic changes in its morphology, e.g., due to self-reconfiguration, module failure or loss, as the experiment in Sections 6.6 and 6.7 demonstrated.

The strategy will, however, generally not work on morphologies that fall over or self-collide while learning. Currently, the strategy makes no attempt to deal with these issues and it is up to the robot designer to select a morphology and initial pose that minimize the risk of such issues. Higher level strategies could be developed to increase the system's robustness to such events. For example, Morimoto and Doya [78] demonstrated a robot able to learn to stand up after falling over. Similarly, for fault-tolerant self-reconfiguration, distributed strategies have been developed to deal with internal collisions, unreliable communication, and hardware failures [79,80].

No single optimization strategy will be universally good for all problems [81], therefore the strategy must be selected to match the problem of interest. The strategy proposed in this paper is minimalistic to facilitate a simple distributed implementation with low resource requirements. Clearly, less minimal strategies that for example utilize gradient information in the optimization process might in some cases converge faster and find gaits with a higher velocity. We have demonstrated such a distributed strategy in recent work which is based on a stochastic optimization of central pattern generator parameters [71,72].

Ideally, a control strategy should generalize and be applicable to more than a single robotic platform. We demonstrated in a number of experiments that the proposed strategy could not only be applied to freely rotating ATRON robots, but also to joint-limited ATRON and M-TRAN robots by optimizing gait control tables (Sections 6.4 and 6.5). Gait control tables are highly portable to both modular and monolithic robots. However, the strategy can only fill the table with values from a small fixed set of possible joint positions. This can in practice limit the usability of the strategy, and methods that will optimize floating point values may be more suitable for gait-control-table optimization.

The problem of transferring a robot controller from simulation to the physical robot needs special attention due to the difficulty of accurately simulating the complex interactions between the robot and its environment [82,63]. The proposed strategy is model-less, online adaptive, and does not currently utilize direct sensor feedback, which makes the transference problem less problematic as we demonstrated with ATRON robots in Section 6.3. However, if the strategy should be applied to robots learning in natural terrain with onboard sensors for measuring velocity, special attention must be given to the signal-to-noise ratio of the reward signal.

In this paper the reward signal is measured externally (in the simulator or using overhead camera) and globally distributed to the modules. To increase autonomy onboard sensors could be used to estimate the robot's velocity, e.g., an inertial measurement unit (IMU) or an optical flow sensor. However, whether such sensors are sufficiently reliable for the presented learning strategy is an open question. To realize a fully distributed learning system, each module must optimize their behavior based on a local reward signal based on local sensors, such a system was demonstrated by Varshavskaya et al. [83] for learning cluster walking behavior for self-reconfigurable robots.

In practical robotic applications, a learning strategy must converge fast and reliably. For the ATRON robots we studied in Section 6.2, we found that the non-snake robots would converge to best-known gaits in 96% of the cases within on average 15 min. For the gait-table experiments the convergence time were more often around 60 min. We anticipate that this might be sufficiently reliable and fast enough for some practical applications but probably not all. Research still remains in exploring how to best improve these characteristics.

Scalability in terms of the number of modules is of special concern in distributed robotics. We found, for an example ATRON robot, that learning convergence time would only grow slowly, approximately linearly, with the number of modules. However, divergence would become increasingly frequent with the number of modules, and beyond 50 modules it would be more likely to happen than not during a two-hour trial (Section 6.8). The scale effects were due to a decrease in the signal-to-noise ratio which will have a negative effect on all learning algorithms. We anticipate that additional inspiration from the neuroscience of biological organisms might enable us to form hypothesis about how to cope with this scalability issue.

The module loss experiment, presented in Section 6.7, is comparable in form and complexity to the experiments by Bongard et al. [70] on self-repair based on self-modeling. Both experiments utilize an 8 DOF walking quadruped robot controlled by a gait-table. Further, in both experiments, damage is induced by removing a lower limb to study the process of self-repair. However, the underlying mechanisms are very different resulting in different trade-off between the two methods. In [70], to recover from damage, a self-model (with 16 open parameters) and a gait-table is optimized using only 16 physical actions and ≈300 k simulated actions. In our experiment the strategy requires ≈1 k physical actions and no simulated actions. Therefore, compared to the self-modeling strategy, the advantages of our distributed strategy is that it is simple, computationally minimal, does not require a simulation system and makes fewer assumptions about the robot morphology. The advantages of the self-modeling strategy is that it requires much less physical actions and that the self-model can also be used for predictive purposes (e.g. to predict if a given action will make the robot fall over). None of the strategies can be considered a stand-alone control system, and it is highly dependent on the application which would be more appropriate. How to best combine the two strategies is an open question.

8. Conclusion

In this paper, we studied a distributed and morphology-independent strategy for adaptive locomotion in modular self-reconfigurable robots. We described a simple reinforcement learning strategy and proposed a heuristic for accelerating the learning.

We applied the strategy to optimize both simple actions and gait-control tables. We presented simulated and physical experiments for a range of different robots constructed from ATRON and M-TRAN robots. In simulation we studied a learning quadrupedal robot and found that from its independently learning modules, a higher-level learning strategy emerged, which was similar to localized random search. We performed experiments in simulation of ATRON modules, which indicate that the strategy is sufficient to learn quite efficient locomotion gaits for a large range of different morphologies up to 12-module robots. A typical learning trial converged in less than 15 min depending on the size

Please cite this article in press as: D.J. Christensen, et al., A distributed and morphology-independent strategy for adaptive locomotion in self-reconfigurable modular robots, Robotics and Autonomous Systems (2013), http://dx.doi.org/10.1016/j.robot.2013.05.009

and type of the robot. Further, we performed experiments with physical ATRON robots online learning to move. These experiments validated our simulation results. Moreover, we experimentally found that the strategy was able to optimize gait control tables for both joint-limited ATRON and M-TRAN modules. However, as anticipated, the increased size of the learning space came at the cost of prolonged time to learn a gait. Yet, even the most complex gaits are typically learned within one hour. In addition, we presented simulated experiments with faults, self-reconfiguration and module loss that illustrated the advantages of utilizing a distributed and configuration-independent learning strategy. We observed that the modules after self-reconfiguration were able to learn to move with a new morphology and adapt to module faults and loss. In simulation, we studied the scalability characteristics of the learning strategy and found that it could learn to move a robot with up to 60 modules. However, the effects of divergence in the learning would eventually become dominant and prevent the robot from being scaled further up. We also found that the convergence time increased slowly, approximately linearly, with the number of modules within the functional range.

In conclusion, we found that learning can effectively be distributed by introducing independent processes that learn in parallel for online, locomotion-gait learning in modular robots. Furthermore, this approach has several advantages compared to centralized methods, such as simplicity in implementation, low resource requirements, morphology independence, adaptability to self-reconfiguration, and inherent fault tolerance.

Acknowledgments

This work was performed as part of the "Locomorph" project funded by the EUs 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 (Technology and Production Sciences).

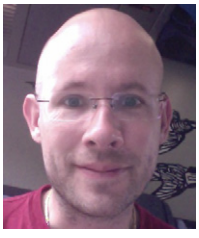
References

[1] D.J. Christensen, M. Bordignon, U.P. Schultz, D. Shaikh, K. Stoy, Morphology independent learning in modular robots, in: Proceedings of International Symposium on Distributed Autonomous Robotic Systems 8, DARS 2008, April 2008, pp. 379–391.
[2] D.J. Christensen, U.P. Schultz, K. Stoy, A distributed strategy for gait adaptation in modular robots, in: Proceedings of the IEEE International Conference on Robotics and Automation, ICRA2010, Anchorage, Alaska, May 2010, pp. 2765–2770.
[3] T. Fukuda, S. Nakagawa, Dynamically reconfigurable robotic system, in: Proceedings of the IEEE International Conference on Robotics & Automation, ICRA'88, 1988, pp. 1581–1586.
[4] T. Fukuda, S. Nakagawa, Y. Kawauchi, M. Buss, Self organizing robots based on cell structures—CEBOT, in: Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS'88, 1988, pp. 145–150.
[5] M. Yim, B. Shirmohammadi, D. Benelli, Amphibious modular robot astrobiologist, in: Proceedings of SPIE Unmanned Systems Technology IX, Vol. 656, April 2007.
[6] K. Stoy, D. Brandt, D.J. Christensen, Self-Reconfigurable Robots: An Introduction, in: Intelligent Robotics and Autonomous Agents Series, The MIT Press, 2010.
[7] K. Gilpin, D. Rus, Modular robot systems, IEEE Robotics & Automation Magazine 17 (3) (2010) 38–55.
[8] S. Murata, H. Kurokawa, Self-Organizing Robots, Springer, 2012.
[9] A. Castano, W.-M. Shen, P. Will, CONRO: towards deployable robots with inter-robot metamorphic capabilities, Autonomous Robots 8 (3) (2000) 309–324.
[10] M. Yim, New locomotion gaits, in: Proceedings, International Conference on Robotics & Automation, ICRA'94, San Diego, California, USA, 1994, pp. 2508–2514.
[11] M. Yim, D. Duff, K. Roufas, Modular reconfigurable robots, an approach to urban search and rescue, in: Proceedings of 1st International Workshop on Human-friendly Welfare Robotics Systems, Taejon, Korea, 2000.
[12] G. Chirikjian, Kinematics of a metamorphic robotic system, in: Proc. of IEEE Intl. Conf. on Robotics and Automation, 1994, pp. 449–455.
[13] S. Murata, H. Kurokawa, S. Kokaji, Self-assembling machine, in: Proceedings of IEEE International Conference on Robotics and Automation, ICRA'94, 1994, pp. 441–448.
[14] D. Rus, M. Vona, A physical implementation of the self-reconfiguring crystalline robot, in: Proceedings of the IEEE International Conference on Robotics and Automation, ICRA'00, Vol. 2, 2000, pp. 1726–1733.
[15] K. Kotay, D. Rus, M. Vona, C. McGray, The self-reconfiguring robotic molecule, in: Proc., IEEE Int. Conf. on Robotics & Automation, ICRA'98, Leuven, Belgium, 1998, pp. 424–431.
[16] H. Kurokawa, S. Murata, E. Yoshida, K. Tomita, S. Kokaji, A 3-d self-reconfigurable structure and experiments, in: Proceedings of the IEEE/RJS International Conference on Intelligent Robots and Systems, IROS'98, Vol. 2. Victoria, BC, Canada, 1998, pp. 860–665.
[17] S. Murata, E. Yoshida, K. Tomita, H. Kurokawa, A. Kamimura, S. Kokaji, Hardware design of modular robotic system, in: Proceedings, IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, IROS'00, Takamatsu, Japan, 2000, pp. 2210–2217.
[18] H. Kurokawa, K. Tomita, A. Kamimura, S. Kokaji, T. Hasuo, S. Murata, Distributed self-reconfiguration of M-TRAN III modular robotic system, International Journal of Robotics Research 27 (3–4) (2008) 373–386.
[19] W.-M. Shen, M. Krivokon, H. Chiu, J. Everist, M. Rubenstein, J. Venkatesh, Multimode locomotion via superbot reconfigurable robots, Autonomous Robots 20 (2) (2006) 165–177.
[20] M. Yim, W.-M. Shen, B. Salemi, D. Rus, M. Moll, H. Lipson, E. Klavins, Modular self-reconfigurable robot systems: challenges and opportunities for the future, IEEE Robotics & Automation Magazine 14 (1) (2007) 43–52.
[21] A. Sproewitz, S. Pouya, S. Bonardi, J. van den Kieboom, R. Mockel, A. Billard, P. Dillenbourg, A. Ijspeert, Roombots: reconfigurable robots for adaptive furniture, IEEE Computational Intelligence Magazine 5 (3) (2010) 20–32. Special issue on Evolutionary and Developmental Approaches to Robotics.
[22] E.H. Østergaard, K. Kassow, R. Beck, H.H. Lund, Design of the ATRON lattice-based self-reconfigurable robot, Autonomous Robots 21 (2006) 165–183.
[23] S. Kernbach, E. Meister, O. Scholz, R. Humza, J. Liedke, L. Ricotti, J. Jemai, J. Havlik, W. Liu, Evolutionary robotics: the next-generation-platform for on-line and on-board artificial evolution, in: IEEE Congress on Evolutionary Computation, 2009, CEC'09, IEEE, 2009, pp. 1079–1086.
[24] H. Wei, Y. Chen, J. Tan, T. Wang, Sambot: a self-assembly modular robot system, IEEE/ASME Transactions on Mechatronics 16 (4) (2011) 745–757.
[25] J. Zhao, X. Cui, Y. Zhu, S. Tang, A new self-reconfigurable modular robotic system UBot: multi-mode locomotion and self-reconfiguration, in: 2011 IEEE International Conference on Robotics and Automation, ICRA, IEEE, 2011, pp. 1020–1025.
[26] Y. Meng, Y. Zhang, A. Sampath, Y. Jin, B. Sendhoff, Cross-ball: a new morphogenetic self-reconfigurable modular robot, in: 2011 IEEE International Conference on Robotics and Automation, ICRA, IEEE, 2011, pp. 267–272.
[27] J. Davey, N. Kwok, M. Yim, Emulating self-reconfigurable robots—design of the SMORES system, in: 2012 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS, October 2012, pp. 4464–4469.
[28] M. Yim, A reconfigurable modular robot with many modes of locomotion, in: Proceedings of the JSME International Conference on Advanced Mechatronics, Tokyo, Japan, 1993, pp. 283–288.
[29] H. Kurokawa, A. Kamimura, E. Yoshida, K. Tomita, S. Kokaji, S. Murata, M-TRAN II: metamorphosis from a four-legged walker to a caterpillar, in: Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems, 2003, pp. 2454–2459.
[30] M. Yim, Locomotion with a unit-modular reconfigurable robot. Ph.D. Thesis, Department of Mechanical Engineering, Stanford University, 1994.
[31] W.-M. Shen, B. Salemi, P. Will, Hormones for self-reconfigurable robots, in: Proc., Int. Conf. on Intelligent Autonomous Systems, IAS-6, Venice, Italy, 2000, pp. 918–925.
[32] W.-M. Shen, B. Salemi, P. Will, Hormone-inspired adaptive communication and distributed control for CONRO self-reconfigurable robots, IEEE Transactions on Robotics and Automation 18 (2002) 700–712.
[33] K. Stoy, W.-M. Shen, P. Will, Using role based control to produce locomotion in chain-type self-reconfigurable robots, IEEE Transactions on Mechatronics 7 (4) (2002) 410–417.
[34] K. Støy, W.-M. Shen, P. Will, Implementing configuration dependent gaits in a self-reconfigurable robot, in: Proc., IEEE Int. Conf. on Robotics and Automation, ICRA'03, Taipei, Taiwan, 2003.
[35] M. Yim, D. Duff, Y. Zhang, Closed chain motion with large mechanical advantage, in: Proceedings, IEEE/RSJ International Conference on Intelligent Robots and Systems, Maui, Hawaii, USA, 2001, pp. 318–323.
[36] M. Yim, S. Homans, K. Roufas, Climbing with snake-like robots, in: Proceedings of IFAC Workshop on Mobile Robot Technology, Jejudo, Korea, 2001.
[37] Y. Zhang, M. Yim, C. Eldershaw, D. Duff, K. Roufas, Phase automata: a programming model of locomotion gaits for scalable chain-type modular robots, in: Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS'03, Las Vegas, Nevada, USA, 2003, pp. 2442–2447.
[38] A.J. Ijspeert, Central pattern generators for locomotion control in animals and robots: a review, Neural Networks 21 (4) (2008) 642–653.
[39] A. Kamimura, H. Kurokawa, E. Yoshida, K. Tomita, S. Murata, S. Kokaji, Automatic locomotion pattern generation for modular robots, in: IEEE International Conference on Robotics and Automation, ICRA, 2003, pp. 714–720.
[40] A. Kamimura, H. Kurokawa, E. Yoshida, S. Murata, K. Tomita, S. Kokaji, Automatic locomotion design and experiments for a modular robotic system, IEEE/ASME Transactions on Mechatronics 10 (3) (2005) 314–325.
[41] D. Marbach, A.J. Ijspeert, Online optimization of modular robot locomotion, in: Proceedings of the IEEE Int. Conference on Mechatronics and Automation, ICMA 2005, 2005, pp. 248–253.

Please cite this article in press as: D.J. Christensen, et al., A distributed and morphology-independent strategy for adaptive locomotion in self-reconfigurable modular robots, Robotics and Autonomous Systems (2013), http://dx.doi.org/10.1016/j.robot.2013.05.009

[42] D.J. Christensen, J. Campbell, K. Stoy, Anatomy-based organization of morphology and control in self-reconfigurable modular robots, Neural Computing and Applications (NCA) 19 (6) (2010) 787–805.
[43] R. Moreno, J. Gomez, Central pattern generators and hormone inspired messages: a hybrid control strategy to implement motor primitives on chain type modular reconfigurable robots, in: ICRA, IEEE, 2011, pp. 1014–1019.
[44] Z. Butler, D. Rus, Distributed locomotion algorithms for self-reconfigurable robots operating on rough terrain, in: Proceedings of IEEE International Symposium on Computational Intelligence in Robotics and Automation, CIRA'03, 2003 pp. 880–885.
[45] R. Fitch, Z. Butler, Million module march: scalable locomotion for large self-reconfiguring robots, International Journal of Robotics Research 27 (2008) 331–343.
[46] P. Varshavskaya, L. Kaelbling, D. Rus, Automated design of adaptive controllers for modular robots using reinforcement learning, International Journal of Robotics Research 27 (3–4) (2008) 505–526. Special issue on Self-Reconfigurable Modular Robots.
[47] E.H. Østergaard, H.H. Lund, Distributed cluster walk for the ATRON self-reconfigurable robot, in: Proceedings of the 8th Conference on Intelligent Autonomous Systems, IAS-8, Holland, Amsterdam, 2004, pp. 291–298.
[48] E. Yoshida, S. Murata, A. Kamimura, K. Tomita, H. Kurokawa, S. Kokaji, A self-reconfigurable modular robot: reconfiguration planning and experiments, The International Journal of Robotics Research 21 (10) (2002) 903–916.
[49] E. Yoshida, S. Murata, A. Kamimura, K. Tomita, H. Kurokawa, S. Kokaji, A motion planning method for a self-reconfigurable modular robot, in: Proceedings, IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS'01, Maui, Hawaii, USA, 2001, pp. 590–597.
[50] A. Ishiguro, M. Shimizu, T. Kawakatsu, Don't try to control everything!: an emergent morphology control of a modular robot, in: Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS, 2004, pp. 981–985.
[51] K. Sims, Evolving 3D morphology and behavior by competition, in: R. Brooks, P. Maes (Eds.), Proc., Artificial Life IV, MIT Press, 1994, pp. 28–39.
[52] H. Lipson, J.B. Pollack, Automatic design and manufacture of robotic lifeforms, Nature 406 (2000) 974–978.
[53] D. Marbach, A.J. Ijspeert, Co-evolution of configuration and control for homogenous modular robots, in: Proc., 8th Int. Conf. on Intelligent Autonomous Systems, Holland, Amsterdam, 2004, pp. 712–719.
[54] E. Yoshida, S. Murata, A. Kamimura, K. Tomita, H. Kurokawa, S. Kokaji, Evolutionary synthesis of dynamic motion and reconfiguration process for a modular robot M-TRAN, in: Proceedings 2003 IEEE International Symposium on Computational Intelligence in Robotics and Automation, CIRA, Kobe, Japan, 2003, pp. 1004–1010.
[55] M.D. Handier, G.S. Hornby, Evolving quadruped gaits with a heterogeneous modular robotic system, in: Proceedings of the IEEE Congress on Evolutionary Computation, IEEE, 2007, pp. 3631–3638.
[56] S. Pouya, J. van den Kieboom, A. Spröwitz, A.J. Ijspeert, Automatic gait generation in modular robots: "to oscillate or to rotate; that is the question", in: Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems. Taipei, Taiwan, 2010, pp. 514–520.
[57] E. Haasdijk, A. Rusu, A. Eiben, Hyperneat for locomotion control in modular robots, in: Evolvable Systems: From Biology to Hardware, 2010, pp. 169–180.
[58] H. Hamann, J. Stradner, T. Schmickl, K. Crailsheim, A hormone-based controller for evolutionary multi-modular robotics: from single modules to gait learning, in: 2010 IEEE Congress on Evolutionary Computation, CEC, IEEE, 2010, pp. 1–8.
[59] P. Zahadat, T. Schmickl, K. Crailsheim, Evolving reactive controller for a modular robot: benefits of the property of state-switching in fractal gene regulatory networks, in: From Animals to Animats 12, 2012, pp. 209–218.
[60] A. Brunete, M. Hernando, E. Gambao, Offline ga-based optimization for heterogeneous modular multiconfigurable chained microrobots, IEEE/ASME Transactions on Mechatronics 18 (2) (2013) 578–585.
[61] A. Faíña, F. Bellas, D. Souto, R. Duro, Towards an evolutionary design of modular robots for industry, in: Foundations on Natural and Artificial Computation, 2011, pp. 50–59.
[62] J.C. Bongard, R. Pfeifer, Evolving complete agents using artificial ontogeny, in: Morpho-functional Machines: The New Species, Springer, 2003, pp. 237–258.
[63] M. Mataric, D. Cliff, Challenges in evolving controllers for physical robots, Robotics and Autonomous Systems 19 (1) (1996) 67–83.
[64] G. Hornby, S. Takamura, J. Yokono, O. Hanagata, T. Yamamoto, M. Fujita, Evolving robust gaits with aibo, in: Proceedings of the IEEE International Conference on Robotics and Automation, 2000, pp. 3040–3045.
[65] S.H. Mahdavi, P.J. Bentley, An evolutionary approach to damage recovery of robot motion with muscles, in: Seventh European Conference on Artificial Life, ECAL03, Springer, 2003, pp. 248–255.
[66] V. Zykov, J. Bongard, H. Lipson, Evolving dynamic gaits on a physical robot, in: Proceedings of Genetic and Evolutionary Computation Conference, Late Breaking Paper, GECCO'04, 2004.
[67] A. Sproewitz, R. Moeckel, J. Maye, A.J. Ijspeert, Learning to move in modular robots using central pattern generators and online optimization, International Journal of Robotics Research 27 (3–4) (2008) 423–443.
[68] P. Maes, R.A. Brooks, Learning to coordinate behaviors, in: National Conference on Artificial Intelligence, 1990, pp. 796–802.
[69] M.J. Mataric, Reinforcement learning in the multi-robot domain, Autonomous Robots 4 (1) (1997) 73–83.
[70] J. Bongard, V. Zykov, H. Lipson, Resilient machines through continuous self-modeling, Science 314 (5802) (2006) 1118–1121.
[71] D.J. Christensen, A. Sproewitz, 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, Paris, France, August 2010, pp. 402–412.
[72] D.J. Christensen, J.C. Larsen, K. Stoy, Adaptive strategy for online gait learning evaluated on the polymorphic robotic locokit, in: Proceedings of the IEEE Conference on Evolving and Adaptive Intelligent Systems, EAIS, 2012.
[73] R. Sutton, A. Barto, Reinforcement Learning—An Introduction, The MIT Press, 1998.
[74] D. Brandt, D.J. Christensen, A new meta-module for controlling large sheets of ATRON modules, in: Proc. of IEEE/RSJ International Conference on Intelligent Robots and Systems, San Diego, California, November 2007, pp. 2375–2380.
[75] D.J. Christensen, Evolution of shape-changing and self-repairing control for the ATRON self-reconfigurable robot, in: Proceedings of the IEEE International Conference on Robotics and Automation, ICRA'06, Orlando, Florida, 2006, pp. 2539–2545.
[76] D.J. Christensen, U.P. Schultz, D. Brandt, K. Stoy, A unified simulator for self-reconfigurable robots, in: Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems, 2008, pp. 870–876.
[77] R. Smith, Open dynamics engine, 2005. www.ode.org.
[78] J. Morimoto, K. Doya, Acquisition of stand-up behavior by a real robot using hierarchical reinforcement learning, Robotics and Autonomous Systems 36 (1) (2001) 37–51.
[79] D.J. Christensen, Experiments on fault-tolerant self-reconfiguration and emergent self-repair, in: Proceedings of Symposium on Artificial Life part of the IEEE Symposium Series on Computational Intelligence. Honolulu, Hawaii, April 2007, pp. 355–361.
[80] U.P. Schultz, M. Bordignon, K. Stoy, Robust and reversible execution of self-reconfiguration sequences, Robotica 29 (1) (2011) 35–57.
[81] D.H. Wolpert, W.G. Macready, No free lunch theorems for optimization, IEEE Transactions on Evolutionary Computation 1 (1) (1997) 67–82.
[82] R.A. Brooks, Artificial life and real robots, in: Toward a Practice of Autonomous Systems: Proceedings of the First European Conference on Artificial Life, Cambridge, MA, USA, 1992, pp. 3–10.
[83] P. Varshavskaya, L.P. Kaelbling, D. Rus, Learning distributed control for modular robots, in: Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems, 2004, IROS, Vol. 3, 2004, pp. 2648–2653.

David Johan Christensen received his M.Sc. degree in computer systems engineering in 2006 and a Ph.D. degree in robotics in 2008 from the Maersk McKinney Moller Institute, University of Southern Denmark. He was a postdoctoral researcher at University of Southern Denmark until 2010, since then he has been an Assistant Professor at Center for Playware, Technical University of Denmark. His research interests include adaptive and self-organizing control of modular robots for self-reconfiguration, locomotion and playful interaction.



Ulrik Pagh Schultz received an M.Sc. degree in computer science from the University of Aarhus, Aarhus, Denmark in 1999 and the Ph.D. degree in computer science from the University of Rennes I, Rennes, France in 2000. He was an Assistant Professor at the University of Aarhus until 2005, since then he holds the position of Associate Professor with the Maersk McKinney Moller Institute, University of Southern Denmark, where he is also the Co-Director of the Modular Robotics Laboratory. His research interests include software engineering for modular robots, program generation and transformation, and domain-specific languages.

Kasper Stoy received the dual M.Sc. degree in computer science and physics from the University of Aarhus, Aarhus, Denmark, in 1999 and the Ph.D. degree in computer system engineering from the University of Southern Denmark, Odense, Denmark, in 2003. He holds the position of Associate Professor with the Maersk McKinney Moller Institute, University of Southern Denmark, where he is also the Co-Director of the Modular Robotics Laboratory. He was a Visiting Scholar with the University of Southern California, Los Angeles, in 2000 and 2002 and with Harvard University, Cambridge, MA, in 2010. His research interests include modular robots, self-reconfigurable robots, and biology-inspired multirobot coordination.

Please cite this article in press as: D.J. Christensen, et al., A distributed and morphology-independent strategy for adaptive locomotion in self-reconfigurable modular robots, Robotics and Autonomous Systems (2013), http://dx.doi.org/10.1016/j.robot.2013.05.009