**International Journal of Automation, Control and Intelligent Systems**
Vol. 1, No. 4, 2015, pp. 92-96
http://www.aiscience.org/journal/ijacis
ISSN: 2381-7526 (Print); ISSN: 2381-7534 (Online)

# Adaptive Locomotion Control System for Modular Robots

**Demin Alexander***

Institute of Informatics Systems, Siberian Branch, Russian Academy of Sciences, Novosibirsk, Russia

## Abstract

This paper proposes a learning control system for modular robots with many degrees of freedom. The system is based on cooperative module training, from discovering common monitor rules for all the modules to their subsequent specification in accordance with semantic probabilistic inference approach. Using an interactive 3D-simulator, a series of successful experiments was conducted in teaching the models of snake-like and multiped robots. The results of experiments have shown that the proposed control system model is quite effective and can be used to control complex modular systems with many degrees of freedom.

## Keywords

Control System, Pattern Recognition, Knowledge Discovery, Data Mining

Received: September 25, 2015 / Accepted: October 25, 2015 / Published online: November 18, 2015



## 1. Introduction

Currently we can observe a new direction of robotics - “modular robotics” [1, 2] actively developing. The basic idea of this approach is construction of robots using many simple modules of similar functionality, which themselves have low mobility, but, connected to each other, are capable of forming complex mechanical systems with many degrees of freedom.

These robots have a number of peculiar features that allow them to exceed conventional robots’ abilities. Firstly, it is possible to create different designs of the same module, allowing to solve various tasks using the same set of modules. It is much cheaper and more convenient than constructing a number of specialized robots for each specific task. Moreover, it is possible to create a transformer robot, independently changing its design depending on the tasks given and adapting to environmental conditions. Second, the modular structure and availability of a large number of degrees of freedom (hyper-redundancy) allows one to create fault-tolerant robot models. Such robot’s individual modules’ failure is not critical to the operation of the entire system, and causes minimal performance degradation. Thirdly, production and use of such robots is economically advantageous and cost-effective because modules of the same type are simpler and cheaper to manufacture and repair.

However, while modular robots provide several advantages associated with hyper-redundancy, they also pose a problem of being much more complex in control and maintaining. In particular, the current task is creating a locomotion control system for a predefined robot configuration.

While for traditional robots the conventional approach to creating control systems is manual programming, for modular robots, this approach proves inefficient. Because of the large number of degrees of freedom it is extremely difficult for a developer to foresee and to program all the possible forms of movement and the situations where they need to be applied, and particularly - taking the ability to adapt in the event of individual modules’ failure or a sudden environment change into account. Therefore, development of control system automatic generation methods based on

* Corresponding author
E-mail address: alexandredemin@yandex.ru

different learning models is becoming a relevant task.

However, the use of popular techniques, such as reinforcement learning, directly for hyper-redundant robots control systems generation is difficult because of the large number of degrees of freedom in such robots. Thus currently many developers generally prefer using the evolutionary method, in particular, genetic algorithms and genetic programming, as well as their combinations with conventional training methods [3-6].

But evolutionary methods application also has its drawbacks, the main ones are as follows: firstly, usually a significant time is required to carry out the calculations, as at each evolutionary step, every decision requires evaluating the locomotion method effectiveness. Secondly, using this method makes adapting to the real work conditions virtually impossible, because the method requires the availability of a population of robots.

In this paper, we propose a learning control system using the logic-probabilistic approach to extracting knowledge for monitor rules generation from the system's environment interaction experience. The specialty of the proposed approach is that the system first attempts to locate the monitor rules that are common for all the modules, and only then - the rules that are specific to each individual module. The effectiveness of the approach will be evaluated by the example of teaching the locomotion methods for the two typical representatives of simple hyper-redundant modular robots, snake-like and multiped robot.

## 2. Simulator

For conducting the experiments with the proposed control model an interactive 3D-simulator with graphical user interface (GUI) was developed. The main purpose of the program is to conduct experiments on controlling in a close to real world environment. The program has virtual environment visualization capabilities as well as the capability of recording experiments in a video file. Open Dynamic Library (ODE) [7] is used as the simulator's physics engine, which allows one to simulate the dynamics of solid bodies with different joint types. The advantages of this library are speed, high integration stability, as well as built-in collision detection. Two models of robots were built in the said simulator: snake-like and multiped.

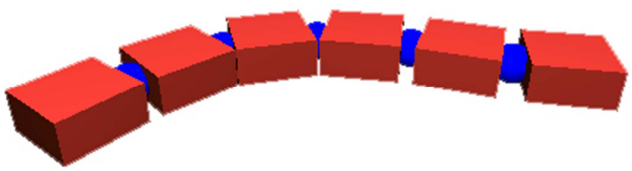

**Fig. 1.** Snake-like robot model.

Snake-like robot model is presented as a set of six identical rectangular blocks ("vertebrae") connected together by universal joints (Fig. 1). All joints are identical and have two angular motors ("muscles"), which provide joint rotation in the vertical and horizontal planes. The proposed design, despite its simplicity, provides sufficient model flexibility and allows simulating body positions specific to biological snakes.

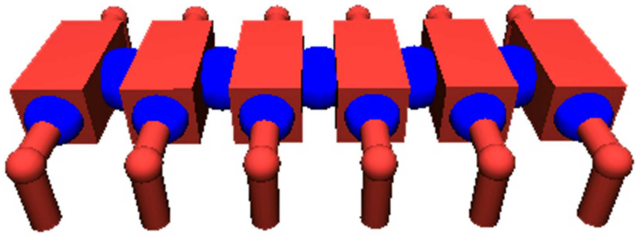

**Fig. 2.** Multiped robot model.

The second model is a multiped robot presented in the form of six identical modules, connected to each other with stiff joints (Fig. 2). Each module has a pair of L-shaped legs on robot's right and left sides, respectively. Thus, the robot has a total of twelve "leg" limbs. Each leg is connected to the module by a universal joint with two angular motors allowing a joint to rotate the foot in the horizontal and vertical planes. In general, the design of the robot reminds of biological centipedes and allows for transportation methods typical for this species.

## 3. Control System

This paper proposes using neural networks consisting of trainable logical neurons, each of which controls a separate module of the robot, for a modular robot control system.

Logical neurons operate on a discrete time scale $t = 0, 1, 2, \ldots$. Each neuron contains a set of inputs $input_1, \ldots, input_k$, taking real values, and one output $output$ taking values from a predetermined set $\{y_1, \ldots, y_m\}$. At any time, $t$, the neuron inputs are supplied with information by assigning real values of the inputs $input_1 = x_1, \ldots, input_k = x_k$, $x_1, \ldots, x_k \in \mathbb{R}$. Neuron's work results in an output signal $output = y$, $y \in \{y_1, \ldots, y_m\}$ taking one of possible values $\{y_1, \ldots, y_m\}$.

After all the neurons in the network provide an output signal, the reward comes from the external environment. Reward function is set based on the final objectives and provides an assessment of control quality. Control system task is detecting patterns of neuron functioning that would ensure getting the maximal reward.

We propose to search for the multitude of the neuron work

governing patterns as logical laws with estimates, as follows:

$$\forall i(P(i), X_1(i),..., X_m(i), Y(i) \to r), \quad (0)$$

where $i = 1,..,n$ – neurons' object indexes iterator variable.

$X_j(i) \in \mathbb{X}$ – predicates from a set of input predicates $\mathbb{X}$, describing inputs $j$ for neurons $N_i$ $(i = 1,..,n)$. E.g., in the simplest case, the data predicates can be defined as $X_j(i) = (input_k(i) = x_r)$, where $x_r$ – constants from the range of input signals that may be defined, for example, by quantizing the possible value range of the respective neuron inputs.

$Y_j(i) \in \mathbb{Y}$ – predicates from a set of output predicates $\mathbb{Y}$, describing neuron outputs $N_i$ $i = 1,..,n$ looking as follows: $Y_j(i) = (output(i) = y_r)$, where $y_r$ – constants from the output signals value set.

$P(i) \in \mathbb{P}$ – predicates from a set of predicates $\mathbb{P}$, looking as follows: $(i = j)$, where $j = 1,..,n$, with the purpose of narrowing the scope of the rules of the (0) type down to individual neurons.

$r$ – reward, its maximization being the constant task of a neuron.

These patterns predict that if a neuron gets input signals $N_i$, $i = 1,..,n$ meeting the input predicates $X_1(i),..., X_m(i)$ from the rule premise, and an output signal sent by the neuron is specified in the output predicate $Y(i)$, then the expectation of the reward will be equal to a value of $r$.

We will also note that if a neuron $N_j$ gets an input specific for only that neuron, we presume that the predicate $X(i)$, describing this input, will take a value of zero ("0") for all $i \neq j$, i.e. for all the other neurons. Similarly, if a neuron's output $N_j$ can take some value $y$, specific for only that neuron, then corresponding output predicate $(output(i) = y)$ will also take a value of zero ("0") for all $i \neq j$.

We will now explain the need for introducing a set of predicates $\mathbb{P}$. In the case of rules (0) not containing predicates from $\mathbb{P}$, they will look like $\forall i(X_1(i),..., X_m(i), Y(i) \to r)$ and will describe the patterns that are common to all neurons $N_i$, $i = 1,..,n$. Adding a predicate rule from $\mathbb{P}$ to the premise automatically narrows the rule's scope down to a particular neuron. Thus, rules containing predicates from $\mathbb{P}$ describe patterns specific to particular neurons. It is also worth mentioning that narrowing down the rules' (0) scope may can take place not only because of $\mathbb{P}$ predicates, but also input and output predicates from $\mathbb{X}$ and $\mathbb{Y}$, describing specific inputs or particular neurons' outputs.

We propose using semantic probabilistic inference-based algorithm described in [8, 9] for finding patterns of the (0) type. This algorithm helps analyze the variety of neuron network statistical data (neurons' inputs and outputs, as well as received reward) and extract all the statistically significant patterns of the (0) type.

We will not be giving a description of the semantic probabilistic inference algorithm in this paper. A detailed description can be found in [8-11]. We will only note that the essence of the algorithm is successive rules clarification and adjustment, starting with one-unit rules, by adding new predicate rules to the premise and then checking the rules on belonging to probabilistic patterns. Essentially, a "directional" rule brute-forcing is realized, allowing to significantly reduce the search space. Search space reduction is achieved by using heuristics, which is - starting from the moment when the length of the rule's premise reaches some predetermined value, or, the depth of the basic brute-forcing, only the rules that are probabilistic patterns are consequently clarified.

The advantage of using the semantic probabilistic inference and rules of the (0) form is organizing the rule search in a way that rules common to all neurons will be detected in the first place, more complex rules, including specific rules for individual neurons, will only be detected after common ones. As a result, when encountering modular robot control tasks, if at least some of the modules have similar functions, which can be described by common rules, the proposed approach can significantly reduce the solution searching time.

Neural network operates as follows: at each network cycle neuron inputs accept input signals. Then, successively for each neuron, a decision making procedure is started, during which rules that apply to the current neuron for current input signals are selected from the variety of rules describing the operation of the neurons. Then, a rule predicting maximal reward $r$ mathematical expectation is chosen from the selected ones. Then an output signal $output = y$ specified in the chosen rule is fed to the neuron output. On the network's initial stage of operation, while the set of rules describing the neurons' work is still empty or when there are no rules applicable to the current set of input signals, the output is determined randomly. After all neurons generate output signals, the reward is given from the external environment and training is conducted in the course of which new rules are discovered and current rules are adjusted in accordance with the offered pattern search algorithm.

## 4. Snake-Like Robot Locomotion Control System

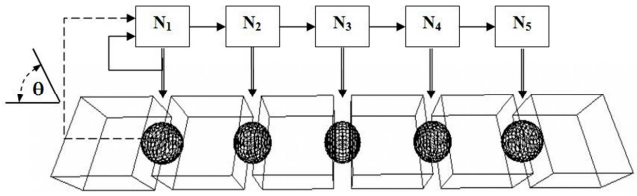


**Fig. 3.** Modular robot locomotion control system neural circuit scheme.

In previous studies [12], we proposed a model of neural circuit for nematode C. Elegans locomotion control, which showed high efficiency in experimental undulating movement learning process. The circuit assumed that the nematode's head acts as an oscillation source, based solely on feedback from stretch receptor. Then the signal is distributed over the nematode's body with a certain time delay, providing the distinctive undulating movement. Since the snake-like robot proves similar to the nematode's model, it was decided to use a similar neural circuit scheme to control the robot's locomotion.

Finally a neural circuit consisting of five neurons (Fig. 3) was selected. Each neuron $N_i$, $i = 1,...,5$ controls a single joint by applying activating signals to the angular motors placed in the said joint. Head neuron $N_1$ receives input information on the bending angle between the head and the subsequent segment. Also neuron receives a signal from its own output with a time delay $\Delta t$ via feedbacks. Other neurons $N_i$, $i = 2,...,5$ only receive a signal from the previous neuron's $N_{i-1}$ output with a time delay of $\Delta t$.

A multitude of input and output predicates for the neurons is specified by quantizing the range of possible values of the neuron's respective inputs and outputs. The reward for the whole locomotion control neural circuit is determined by the speed that the robot will develop over a time interval of $\Delta t$: higher speed means higher reward.

A number of successful experiments were conducted using the 3D-simulator on learning of the proposed model for locomotion methods. The results of the experiments show that the control system manages to consistently learn an effective way of moving forward, based on undulating body movement in horizontal plane. This method of movement is most common among snakes, and is typical for some other animals, e.g. nematodes. Fig. 4 shows optimum movement sequences for forward motion found by the system during training.

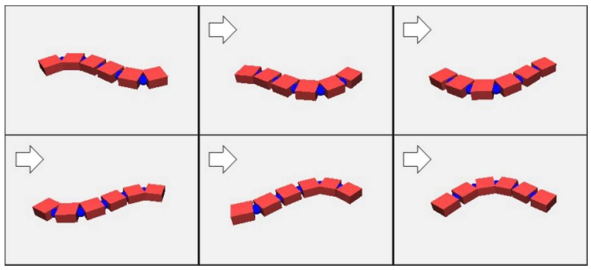

**Fig. 4.** Snake-like robot movement sequence for forward motion.

## 5. Multiped Robot Locomotion System

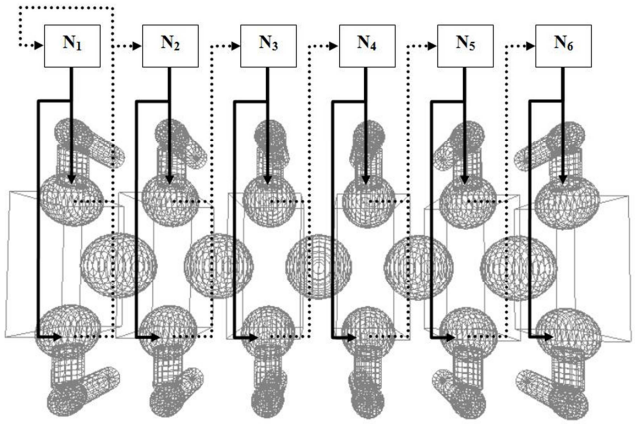


**Fig. 5.** Multiped robot locomotion control system neural circuit scheme.

For controlling the multiped robot a neural circuit consisting of six neurons was selected - having one neuron for each module of the robot (Fig. 5). Each neuron $N_i$, $i = 1,...,6$ controls the movement of it's module's left and right legs by applying activating signals to corresponding angular motors that rotate the limb in the joint. To simplify the task, the movements of the right and left legs of the robot were synchronized so that the motion of one foot is always the other leg's counterphase. Eg., the forward movement of the left foot is always accompanied by the backward movement of the right foot. Thus, it is in fact enough for a neuron to only control one leg's movement, because the second leg will be repeating the same movement, only in counterphase.

The head neuron $N_1$ receives input information on the first module's leg position. Other neurons $N_i$, $i = 2,...,6$ receive input information on previous module's leg position. Leg position status is defined as horizontal and vertical planes' limb joint bending angle. As in the previous task, a multitude of input and output predicates for the neurons is specified by quantizing the range of possible values of the neuron's respective inputs and outputs. Analogically, the reward for the whole locomotion control neural circuit is determined by the speed that the robot will develop over a time interval of $\Delta t$: higher speed means higher reward.

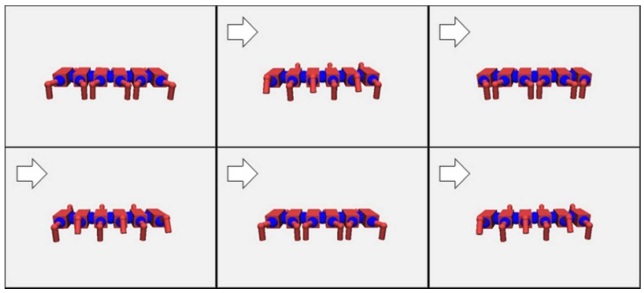

**Fig. 6.** Multiped robot movement sequence for forward motion.

A number of successful experiments were conducted using the 3D-simulator on learning of the proposed multiped robot model for locomotion methods. The results of the experiments show that the control system successfully manages to discover and learn coordinated limb movement sequences ensuring effective forward motion. Fig. 6 shows an example of optimal movement sequences found by the system during training.

## 6. Conclusions and Results

In this paper we proposed an approach to learning control system for modular robots with many degrees of freedom. The system is based on cooperative module training, from discovering common monitor rules for all the modules to their subsequent specification in accordance with semantic probabilistic inference approach. The main advantages of the proposed approach are: (1) the ability of learning during real work based only on environment interaction experience, and (2) high learning speed achieved through the effective use of the modules' functional similarity properties and directional rule search algorithm. In addition to that, the proposed approach scales well with the number of modules increasing. In particular, the addition of new segments to the structure of snake-like and multiped robots in the experiments has little effect on learning efficiency, because it does not change the number of modules' common rules. However, it's worth noting that the effectiveness of this approach depends on the number of similar modules in the robot's design. With the similar modules' share decreasing in the robot's design the benefits from the common rules' usage are lost. From a practical point of view, results of experiments in teaching the models of snake-like and multiped robots have shown that the proposed approach is quite effective and can be used to control complex modular systems with many degrees of freedom.

## Acknowledgments

This work is supported by RFBR Grant № 14-07-00386.

## References

[1] Yim M.H., Duff D.G., Roufas K.D. Modular reconfigurable robots, an approach to urban search and rescue // 1st International Workshop on Human Welfare Robotics Systems (HWRS2000). – 2000. – pp. 19-20.

[2] Stoy K., Brandt D., Christensen D.J. Self-Reconfigurable robots: an introduction // Intelligent robotics and autonomous agents series. – MIT Press, 2010. – 216 p.

[3] Kamimura A., Kurokawa H., Yoshida E., Tomita K., Murata S., Kokaji S. Automatic locomotion pattern generation for modular robots // Proceedings of 2003 IEEE International Conference on Robotics and Automation. – 2003. – pp. 714-720.

[4] Ito K., Matsuno F. Control of hyper-redundant robot using QDSEGA // Proceedings of the 41st SICE Annual Conference (2002). – 2002. – V. 3. – pp. 1499-1504.

[5] Marbach D., Ijspeert A.J. Co-evolution of configuration and control for homogenous modular robots // Proceedings of the eighth conference on Intelligent Autonomous Systems (IAS8). – IOS Press, 2004. – pp. 712-719.

[6] Tanev I., Ray T., Buller A. Automated Evolutionary Design, Robustness and Adaptation of Sidewinding Locomotion of Simulated Snake-like Robot // IEEE Transactions on Robotics. – V.21. – N. 4. – August 2005. – pp. 632-645.

[7] Smith R. Open Dynamics Engine. – URL: http://ode.org/.

[8] Vityaev E.E. The logic of prediction. In: Mathematical Logic in Asia. Proceedings of the 9th Asian Logic Conference (August 16-19, 2005, Novosibirsk, Russia), edited by S.S. Goncharov, R. Downey, H. Ono, World Scientific, Singapore, 2006, pp.263-276.

[9] Vityaev E.E. A formal model of neuron that provides consistent predictions // Biologically Inspired Cognitive Architectures 2012. Proceedings of the Third Annual Meeting of the BICA Society (A. Chella, R.Pirrone, R. Sorbello, K.R. Johannsdottir, Eds). In Advances in Intelligent Systems and Computing, v.196, Springer: Heidelberg, New York, Dordrecht, London. 2013, pp. 339-344.

[10] Vityaev E.E., Demin A.V. Recursive subgoals discovery based on the Functional Systems Theory // Frontiers in Artificial Intelligence and Applications. 2011. V. 233. p. 425-430.

[11] Demin A.V. Logical Model of the Adaptive Control System Based on Functional Systems Theory // Young Scientist USA. Applied science. – Auburn, Washington, 2014. – pp. 113-118.

[12] Demin A.V., Vityaev E.E. Learning in a virtual model of the C. elegans nematode for locomotion and chemotaxis // Biologically Inspired Cognitive Architectures (2014). – Elsevier, 2014. – V. 7. – pp. 9-14.