Available online at www.sciencedirect.com

SciVerse ScienceDirect

Procedia Computer Science 6 (2011) 329–334

Procedia
Computer Science

Complex Adaptive Systems, Volume 1
Cihan H. Dagli, Editor in Chief
Conference Organized by Missouri University of Science and Technology
2011- Chicago, IL

Evolving spiking neural networks for robot control

R. Batllori[a]*, C. B. Laramee, W. Land, J. D. Schaffer[b]*

Department of Bioenginnering, Binghamton University, New York, USA

Abstract

We describe a sequence of experiments in which a robot "brain" was evolved to mimic the behaviours captured under control of a heuristic rule program (imitation learning). The task was light-seeking while avoiding obstacles using binocular light sensors and a trio of IR proximity sensors. The "brain" was a spiking neural network simulator whose parameters were tuned by a genetic algorithm, where fitness was assessed by the closeness to target output spike trains. Spike trains were frequency encoded. The network topology was manually designed, and then modified in response to observed difficulties during evolution. We noted that good performance seems best approached by judicious mixing of excitation and inhibition. Besides robotic applications, the domain of "smart" prosthetics also appears promising.

Keywords robotics; spiking neural networks; genetic algorithms

1. Introduction

About a decade ago, a new neural network formalism emerged, spiking neural networks (SNNs) [1]. This formalism has had several exciting computational properties proven for it [2]: these models can do everything the older models could, and often with many fewer neurons. In addition, since they compute in spikes whose only distinguishable property is spike time, perhaps they could be used for spatio-temporal pattern tasks and do better than conventional NNs that generally are hard to apply to time series tasks. The challenge is that there are no known design rules to devise a SNN for a given task; the designer is on his own.

We set out to explore evolutionary computation (EC) for this challenge. Herein, we follow the lead of Sichtig [3] who developed a simulator (SSNNS) and used a genetic algorithm (GA) to adjust the free parameters in a model

* Corresponding author.
E-mail address: robbatllori@gmail.com (R. Batllori)
dschaffe@binghamton.edu (J. D. Schaffer)

1877–0509 © 2011 Published by Elsevier Ltd. Open access under CC BY-NC-ND license.
doi:10.1016/j.procs.2011.08.060

whose topology was manually specified. She demonstrated that for a temporal version of the XOR task, her approach found a topology that was smaller than the currently best known (EC essentially removed synapses by setting the delays to large values), and a noise-robust version that escaped the hair-trigger behavior of the then best known topology. In addition, with Rosen et al. [4] she devised a model of the nucleus of the solitary tract (a rat mid-brain structure). Here, we apply this paradigm to a task in robotics. We note that, in addition to the domain of autonomous robotics, this approach seems to hold promise of applications in smart prosthetics. The advantages would be fast and sophisticated computation, performed in the same information format (spike trains) used for command inputs and sensory feedback from/to living neurons.

2. Background

2.1 Spike Response Model and Evolutionary Computation

The spike response model (SRM) is a leaky integrate and fire model consisting of neurons and synapses that connect them [1]. Each neuron has a refractory period and a time constant for recovery after firing. Each synapse has time constants for the rise and decay of its post synaptic potential (PSP) and a weight and delay. Based on Sichtig's previous experience, we anticipated significant epistasis in the tuning of SNN parameters, so we followed her lead in choosing Eshelman's CHC [5].

The chromosome representation was direct encoding: a binary gene for each free network parameter. In the experiments reported here, we evolved only the weights and delays for the network synapses. All other parameters were manually specified. Following the lead of Rosen et al. [4], we explored a number of network topologies, starting with a reasonable guess and then making changes after observing their performances.

2.2 Evolutionary SNN Robotics

Although GAs and SNNs have been applied in robotics for a decade now [6], relatively little work has been published, and most of these efforts have been of the proof-of-concept nature, involving a single robot task with a single sensor modality. Floreano and Mattiussi [7] used the GA to set signs (excite/inhibit) and presence/absence of synapses between 16 vision sensors of a Khepera robot and an internal layer of 10 neurons, all potentially interconnected, and from four of the internal neurons to the wheels. The task was wall following where the walls were painted with a random pattern of vertical stripes (making each view unique). Later this group applied much the same approach to a blimp-like flying robot [8]. In contrast with our work, these efforts performed online learning: the fitness of each chromosome was computed on the performance of the real robot. Hagras et al. [9] focused on a real-time GA for faster online learning, and following Floreano & Mattiussi, achieved wall-following behavior using ultrasound sensors. Wang et al. [10] presented experiments with online learning using Hebbian weight adjustment. They used a perceptron-like topology that served as inspiration for our first topology and showed improved obstacle avoidance with learning, even though it was unsupervised. In contrast with these efforts, we have attempted imitation learning.

3. Methods

The robot used was a commercial open robot manufactured by Abe Howell Robotics (www.abotics) (Fig 1a). It has a Bluetooth module for connecting wirelessly to an external controller, and two types of sensors: three Sharp GPD120 IR proximity sensors and two cadmium sulfide cell light sensors. The sensor locations are shown in Fig 1b.

3.1 Robot task

Regardless of the controller (heuristic rules or SNN), the robot should:

- Approach the brightest light in the maze.
- Do this facing frontward.
- Avoid hitting obstacles.

3.2 Training protocol & data recording

The protocol of the research was to replicate with a SNN controller the robot's behavior from examples generated by the robot itself using a heuristic rule-based controller. This is called imitation learning [11]. A set of 48 heuristic rules was developed to perform the robot's task. The light sensors in the heuristic rules were only used to determine the difference between left and right (not absolute magnitude). The wheel speeds depended on the value of one of the three IR-sensors which detected the nearest obstacle. All the inputs coming from the sensors and the outputs generated by the heuristic rules were recorded. An initial data set was recorded directly from a run of the robot in the maze, but this proved to instantiate too few of the heuristic rules. Instead two learning pools were formed by taking: a set of 23 inputs/outputs instantiating 13 rules (data set 1), and a set of 40 inputs/outputs instantiating 21 rules (dataset 2), from several robot trajectories devised to exercise many rules. These learning pools were used to evolve SNNs offline, feeding the inputs to the SNN simulator and using the outputs as targets.

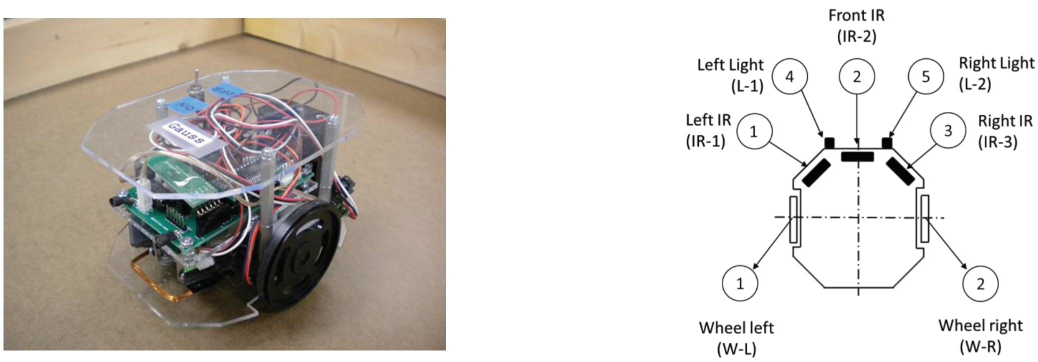

Figure 1. (a) Picture of robot, (b) top view schematic, wheel and sensor locations

3.3 Data recording

The analog values from both inputs and outputs were converted into spikes. Two coding methods were tested: 'delay coding' and 'spike frequency'. In delay coding, a stronger stimulus makes neurons fire earlier than weaker ones. This coding has been observed in olfactory neurons. In frequency coding, a stronger stimulus produces more spikes than a weaker one. The signal from the IR-sensor was encoded considering a stronger stimulus as nearer the obstacle. The signal coming from the light sensor was encoded considering a stronger stimulus when the light was more intense. The wheel speeds were encoded such that more spikes make the wheel go forward faster.

The epoch time (time for one input-output transaction) was obtained from the heuristic rules, where the cycle time was observed, and for the SNN, it was divided into two parts, the first part was to encode the inputs coming from the sensors and the second part was reserved to encode the outputs. The epoch time division is illustrated in Figure 2a.

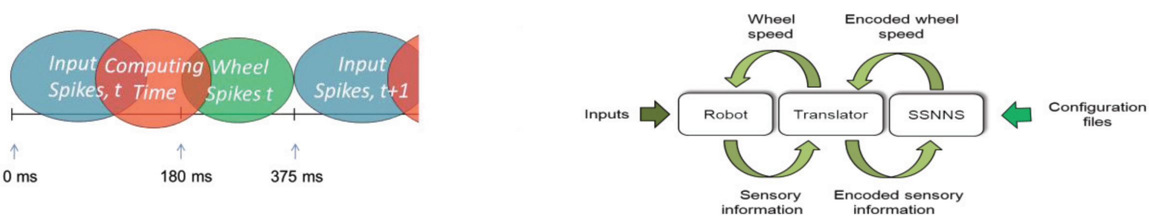

Figure 2. (a) SSNNS timing diagram, (b) data flow from robot to SSNNS through translator

3.4 Translator

This program served as the interface between the robot and the SNN simulator (SSNNS). It obtained the inputs from the robot sensors and converted them into spikes, sending them to SSNNS, which returned the output spikes that were then converted into analog values and sent to the robot. The computing cycle is shown in Figure 2b.

4. Results

4.1 Test A

We began with a feed-forward topology and using spike delay coding. The results were poor for the simplest task: go towards a light, without obstacles. The robot failed to reach half way even after considerable genetic search. One problem seemed to be the brittleness of delay coding under our fitness function that assesses closeness to the target output spike trains [12].

4.2 Test B

This observation motivated a change of topology: we adopted a new topology inspired by Braitenberg [13] and by Wang et al. [10] where each light sensor excites the wheel on the opposite side. The side-pointing IR-sensors excite the wheels on the same side and inhibit the light inter-neurons before the excitation crosses to the opposite side. The front IR sensor inhibits both inter-neurons. This topology is shown in Figure 3a. The results of evolution were taken to be the chromosome most repeated and best fit, from the last 25% of evolutionary trials.

The robot was able to approximate its task depending on the situation (some obstacle collisions, see Fig. 3b), but was unable to stop upon reaching the light source. The robot also could get stuck in corners. We recognized that this expected halting behavior is an exception to the general behavior and was implemented in the rules set by special exception rules. We expected some special modifications might be needed in the SNN.

4.3 Test C

To help with the halting and the behavior stability, two new inhibitory synapses were created: from sensor IR-2 (front) to the output neurons giving extra inhibition when facing an obstacle (Figure 4a). We also noted that, while the topology was symmetrical, the GA was not constrained to symmetrical solutions. To see if symmetry had a significant effect, we took the best evolved chromosome and replaced both left & right parameters with their average. The result was deemed better than the evolved solution and is shown in Figure 4b. The robot motion did seem more directed, and didn't get stuck in corners, but close to the light, the strong firing of the light sensors overwhelmed the IR sensors and the desired halting was still not seen.

4.4 Test D

Given the value of symmetry, we modified the code to enforce symmetry: one gene for both parameters. A highly desirable side effect of this was a substantial ($O(10^{30})$) reduction in the search space size. We also added inhibitory synapses directly from the light sensors to the wheels on the same side reasoning that the light-seeking behavior shouldn't be harmed, and perhaps the high frequency firing near the light source could be harnessed for halting (Figure 5a). The result was an improvement in the behavior. The movements were smoother, and the obstacle avoidance rather better (judgment call over several tests). It was more sensitive to light, but halting was deemed still not satisfactory. The result is shown in Figure 5b.

4.5 Test F

Another test was performed (Test E, data not shown) with the topology above, but reconnecting the light sensor inhibitory synapses to the inter-neurons on the opposite side. This provided no improvement over Test D. Hoping we were moving in the right direction, the final topology we present combined all these light-driven inhibitory synapses and is shown in Figure 6a. This last topology was able to reduce the speed when it approached brightest light, but the correct response was delayed a few seconds and, it still was unable to react completely as desired. The robot appeared to be much more sensitive to light than with the other topologies and sometimes (in over reflective corners) could get confused (Figure 6b). Perhaps we have reached a point of diminishing returns with this basic topology and more neurons and a more complex topology will be needed to achieve the desired behaviors.

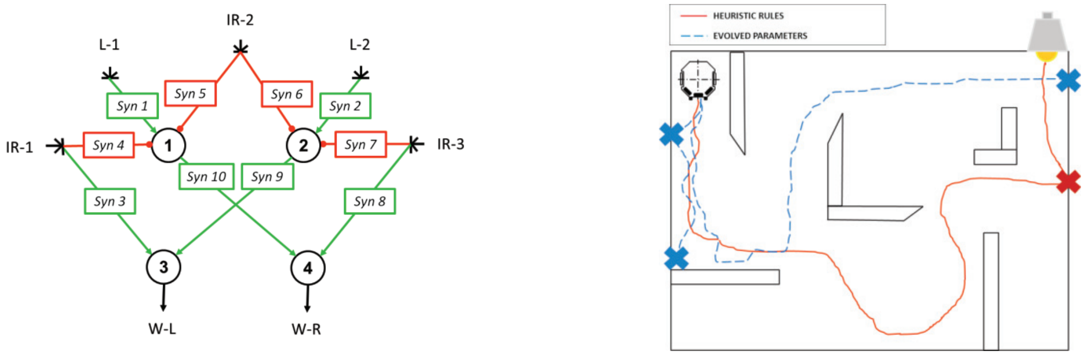

Figure 3. (a) Test B topology, (b) evolved performance

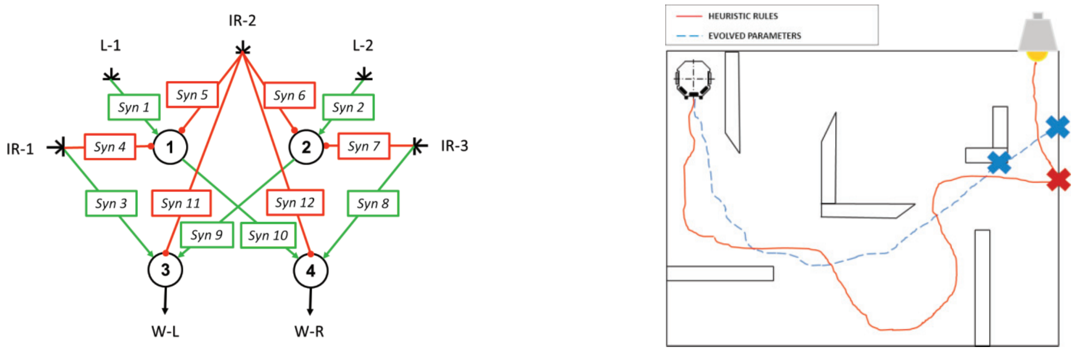

Figure 4. (a) Test C topology, (b) evolved performance after averaging

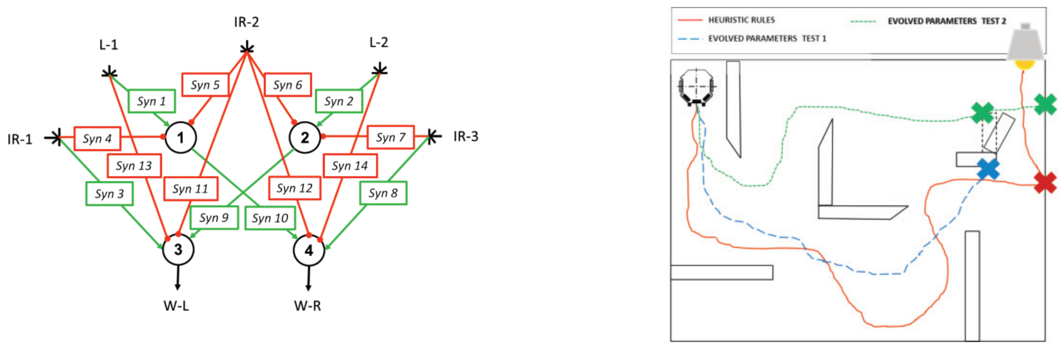

Figure 5. (a) Test D topology, (b) evolved performance

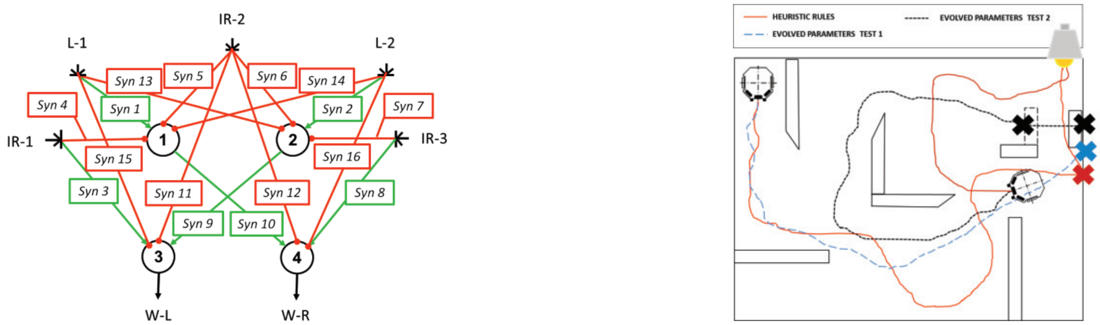

Figure 6. (a) Test F topology, (b) evolved performance

4. Discussion

We have presented a first set of experiments with a newly built SNN controller for a mobile robot that is adjusted by evolution offline in an imitation learning way. This learning protocol does seem practical, enabling parallel computation during learning without a need for multiple robots. It also seems able to cope with the inevitable hardware imperfections that are included in the hardware-generated learning instances, and to generalize successfully by sampling from a necessarily limited set of training instances. The superior behavior using frequency coding for the sensory inputs and motor outputs, we believe is a general observation. Nature also seems to have arrived at this conclusion, and a long time ago. Delay coding seemed to be brittle under our fitness function.

The task we chose can seem simple, but as far as we know, such a task involving two sensor modalities and requiring a SNN to balance the potentially competing objectives of light-seeking and obstacle avoidance has not been published before. It is dangerous to attempt to generalize from such a small set of experiments, but our experience does seem to point to the need for increasing complexity in the topology in order to cope with this task. Evolution (as opposed to simpler optimization schemes) does seem to be needed to cope with the epistasis in tuning the many free parameters. If we are able to devise a reasonable scheme for having evolution also take over the task of topology exploration, we hope to escape the limits of the manual and incremental changes we report here. Our intuition suggests that the halting behavior we were seeking will call for a topology of a complexity we have trouble foreseeing

5. References

1. W. Gerstner and W.M. Kistler, Spiking neuron models - single neurons, populations, plasticity, Cambridge University Press, August 2002.
2. W. Maass, *Noisy spiking neurons with temporal coding have more computational power than sigmoidal neurons*, Advances in Neural Information Processing Systems (M. Mozer, M. I. Jordan, and T. Petsche, eds.), vol. 9, MIT Press, Cambridge University Press, 1997, pp. 211-217.
3. H. Sichtig, The SGE Framework -- Discovering Spatio-temopral Patterns in Biological Systems with Spiking Neural Networks (S), a Genetic Algorithm (G), and Expert Knowledge (E), PhD Dissertation, Binghamton University, Binghamton, NY 2009.
4. A.M. Rosen, H. Sichtig, J.D. Schaffer and P.M. Di Lorenzo, *Taste-specific cell assemblies in a biologically informed model of the nucleus of the solitary tract.* J Neurophysiol. 2010 Jul;104(1):4-17.
5. L. J. Eshelman, *The CHC adaptive search algorithm How to have safe search when engaging in nontraditional genetic recombination*, Foundations of Genetic Algorithms (Gregory J. E. Rawlins, ed.), Morgan Kaufmann, San Mateo, CA, 1990, pp. 265-283.
6. D. Floreano and C. Mattiussi, *Evolution of spiking neural controllers for autonomous vision-based robots*, Evolutionary Robotics IV, From Intelligent Robotics to Artificial Life, Springer-Verlag, (2001), pp. 38-61.
7. D. Floreano, P. Dürr, and C. Mattiussi, *Neuroevolution from architectures to learning*, Evolutionary Intelligence 1 (2008), no. 1, pp. 47-62.
8. J-C. Zufferey, D. Floreano, M. van Leeuwen, and T. Merenda, Evolving Vision-based Flying Robots, Proceedings Second International Workshop on Biologically Motivated Computer Vision (BMCV02), Springer_Verlag, London, 2002, pp. 592-600.
9. H. Hagras, A. Pounds-Cornish, and M. Colley, *Evolving spiking neural network controllers for autonomous robots*, The Proceeding of IEEE International Conference on Robotics & Automation 2004, New Orleans, LA, USA, 2004, pp. 4620–4626.
10. X.Wang, Z. Hou, A. Zou, M. Tan and L. Cheng, *A behavior controller based on spiking neural networks for mobile robots*, Neurocomputing 71 (2008), pp. 655-666.
11. D. Floreano and C. Mattiussi , Bioinspired Artificial Intelligence, Theories, Methods, and Technologies, MIT Press, Cambridge. MA, 2008.
12. J.D. Schaffer, H. Sichtig, C. Laramee, A Series of Failed and Partially Successful Fitness Functions for Evolving Spiking Neural Networks, Genetoc and Evolutionary Computation Conference (GECCO2009 Companion), 2009, pp. 2661-2664.
13. V. Braitenberg, Vehicles: Experiments in synthetic psychology. Cambridge, MA: MIT Press, 1984.