Evolution of Neural Controllers with Adaptive Synapses and Compact Genetic Encoding

Dario Floreano and Joseba Urzelai

Laboratory of Microprocessors and Interfaces (LAMI)
Swiss Federal Institute of Technology, CH-1015 Lausanne (EPFL)
Dario.Floreano@epfl.ch, Joseba.Urzelai@epfl.ch

Abstract. This paper is concerned with artificial evolution of neurocontrollers with adaptive synapses for autonomous mobile robots. The method consists of encoding on the genotype a set of local modification rules that synapses obey while the robot freely moves in the environment [2]. The synaptic weights are *not* encoded on the genotype. In the experiments presented here, a "behavior-based fitness" function gives reproductive advantage to robots that can solve a sequential task. The results show that evolutionary adaptive controllers solve the task much faster and better than evolutionary standard (non-adaptive) controllers, that the method scales up well to large architectures whereas standard controllers do not, and that evolved adaptive controllers are not trivial and cannot be reduced to a fixed-weight network.

1 Evolution and Learning

Artificial evolution of adaptive individuals can provide computational advantages and richer adaptive dynamics [1] with respect to evolution of individuals whose defining parameters are entirely genetically-determined. Several hypotheses have been suggested to explain the observed advantages of the combination of evolution and learning [7, 8, 11]. In general, these advantages amount to discovery of better solutions for a given problem, to faster convergence, and to improved robustness in face of changing fitness landscapes. They are thus relevant for artificial evolution of robotic control systems.

Despite the growing worldwide interest in Evolutionary Robotics, remarkably little work has been done in this direction. A review of the combination of evolution and learning for sensory-motor controllers can be found in [5, 10]. Most of the work done so far and effectively applied to robots, or realistically simulated organisms, shares two components: all synaptic weights are individually specified and directly encoded on the genetic string, and learning amounts to some standard gradient-descent algorithm.

In previous work we employed a different approach where synaptic strengths are not genetically specified and adaptation during life consists of Hebbian synaptic changes [2–4]. For each synapse, the genetic string encoded four Hebbian rules, a learning rate, the sign, and the postsynaptic effect of the travelling signal (driving or modulatory). At the beginning of an individual's "life", all

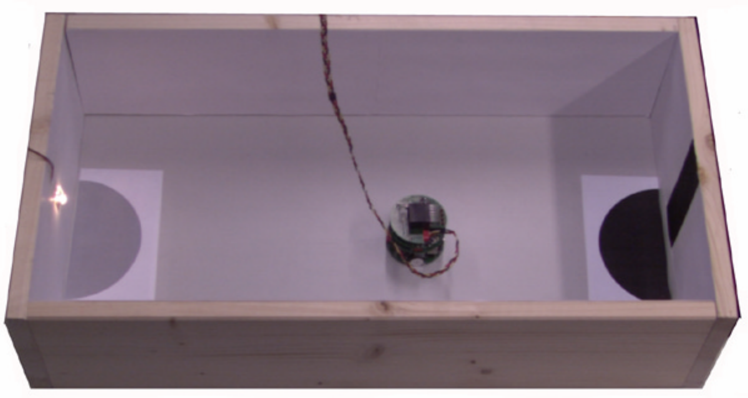

Fig. 1. A mobile robot equipped with a vision module gains fitness by staying on the gray area only when the light is on. The light is normally off, but it can be switched on if the robot passes over the black area positioned on the other side of the arena. The robot can detect ambient light and the color of the wall, but not the color of the floor.

synapses were initialized to small random values and, while the robot was freely moving around the environment, each synapse could modify its own strength every 100 ms according to the genetically specified Hebbian rule. Evolved individuals displayed more robust behaviors [2] and consistently won tournaments in a competitive co-evolutionary scenario [4].

In this paper, we extend previous work by using a much more compact genetic representation of adaptive neurocontrollers and systematically compare its performance with respect to direct encoding of synaptic weights and to encoding of noisy synapses. In a further set of experiments, we show that compact encoding of adaptive networks scales up to large neurocontrollers whereas direct encoding fails. Finally, we analyze a family of evolved controllers under different conditions and show that their competitive advantage comes indeed from evolved adaptive synapses.

2 Environment, task, architecture, and genetic encoding

A mobile robot Khepera equipped with a vision module is positioned in the rectangular environment shown in figure 1. A light bulb is attached on one side of the environment. This light is normally off, but it can be switched on when the robot passes over a black-painted area on the opposite side of the environment. A black stripe is painted on the wall over the light-switch area. Each individual of the population is tested on the same robot, one at a time, for 500 sensory motor cycles, each cycle lasting 100 ms. At the beginning of an individual's life, the robot is positioned at a random position and orientation and the light is off.

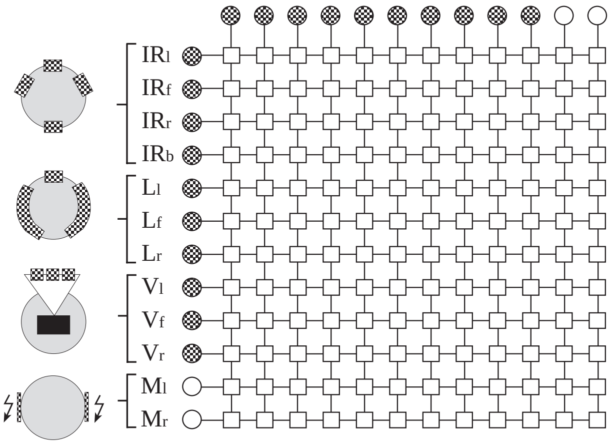

Fig. 2. The neural controller is a fully-recurrent discrete-time neural network composed of 12 neurons giving a total of 12 x 12= 144 synapses (here represented as small squares of the unfolded network). 10 sensory neurons receive additional input from one corresponding pool of sensors positioned around the body of the robot shown on the left (l=left; r=right; f=front; b=back). ***IR***=Infrared Proximity sensors; ***L***=Ambient Light sensors; ***V***=vision photoreceptors. Two motor neurons ***M*** do not receive sensory input; their activation sets the speed of the wheels ($M_i > 0.5$ forward rotation; $M_i < 0.5$ backward rotation)

The fitness function is described as the number of sensory motor cycles spent by the robot on the gray area beneath the light bulb *when the light is on* divided by the total number of cycles available (500). In order to maximize this fitness function, the robot should find the light-switch area, go there in order to switch the light on, and then move towards the light as soon as possible, and stand on the gray area[1]. Since this sequence of actions takes time (several sensory motor cycles), the fitness of a robot will never be 1.0. Also, a robot that cannot manage to complete the entire sequence will be scored with 0.0 fitness.

A light sensor placed under the robot is used to detect the color of the floor—white, gray, or black— and passed to a host computer in order to switch on the light bulb and compute fitness values. The color of the floor is *not* given as input to the neural controller. After 500 sensory motor cycles, the light is switched off and the robot is repositioned by applying random speeds to the wheels for 5 seconds.

The controller is a fully-recurrent discrete-time neural network (figure 2). It has access to three types of sensory information: infrared light (object proximity), ambient light, and vision. The active infrared sensors positioned around the robot

[1] Notice that the fitness function does not explicitly reward this sequence of actions, but only the final outcome of the overall behavior chosen by the robot. Therefore, we call it a *behavior-based* fitness function.

measure the distance from objects (up to 4 cm). Their values are pooled into four pairs and the average reading of each pair is passed to a corresponding neuron. The same sensors are used to measure ambient light too. These readings are pooled into three groups and the average values are passed to the corresponding three light neurons. The vision module consists of an array of 64 photoreceptors covering a visual field of 36°. The visual field is divided up in three sectors and the average value of the photoreceptors (256 gray levels) within each sector is passed to the corresponding vision neuron. Two motor neurons are used to set the rotation speed of the wheels. Neurons are updated every 100 ms according to the following equation

$$y_i \leftarrow \sigma\left(\sum_{j=0}^{N} w_{ij} yj\right) + I_i,$$

where y_i is the activation of the ith neuron, w_{ij} is the strength of the synapse between presynaptic neuron j and postsynaptic neuron i, N is the number of neurons in the network, $0 \leq I_i < 1$ is the corresponding external sensory input, and $\sigma(x) = (1 + e^x)^{-1}$ is the sigmoidal function. $I_i = 0$ for the motor neurons.

Each synaptic weight w_{ij} can be updated after every sensory-motor cycle (100 ms) using one of the four modification rules specified in the genotype.[2] The four rules are called Hebbian because they are a function of the pre-synaptic activation, of the post-synaptic activation, and of the current value of the weight itself. The *Plain Hebb rule* strengthens the synapse proportionally to the correlated activity of the two neurons. The *Postsynaptic rule* behaves as the plain Hebb rule, but in addition it weakens the synapse when the postsynaptic node is active but the presynaptic is not. Conversely, in the *Presynaptic rule* weakening occurs when the presynaptic unit is active but the postsynaptic is not. Finally, the *Covariance rule* strengthens the synapse whenever the difference between the activations of the two neurons is less than half their maximum activity, otherwise the synapse is weakened. Synaptic strength is maintained within a range $[0, 1]$ (notice that a synapse cannot change sign) by adding to the modification rules a self-limiting component inversely proportional to the synaptic strength itself [2, 3, for more details].

Encoding	Bits for one synapse					Bits for one node				
Genotype	1	2	3	4	5	1	2	3	4	5
A	sign	strength				sign	strength			
B	sign	Hebb rule		rate		sign	Hebb rule		rate	
C	sign	strength		noise		sign	strength		noise	

Table 1. Genetic encoding of synaptic parameters for Synapse Encoding (left) and Node Encoding (right). In the latter case the sign encoded on the first bit is applied to all outgoing synapses whereas the properties encoded on the remaining four bits are applied to all incoming synapses. A: Genetically determined controllers; B: Adaptive synapse controllers; C: Noisy synapse controllers.

[2] These four rules co-exist within the same network.

Two types of genetic (binary) encoding are considered (see table): *Synapse Encoding* and *Node Encoding.* Synapse Encoding is also known as direct encoding [12]. Every synapse is individually coded on 5 bits, the first bit representing its sign and the remaining four bits its properties (either the weight strength or its adaptive rule). Node Encoding instead codes only the properties of the nodes in the network. These properties are then applied to all its incoming synapses (consequently, all incoming synapses to a given node have the *same* properties). Each node is characterized by 5 bits, the first bit representing its sign and the remaining four bits the properties of its incoming synapses. Synapse Encoding allows a detailed definition of the controller, but for a fully connected network of N neurons the genetic length is proportional to N^2. Instead Node Encoding requires a much shorter genetic length (proportional to N), but it allows only a rough definition of the controller.

Independently of the type of genetic encoding, the following three types of properties can be encoded on the last 4 bits. A) *Genetically determined*: Weight strength. The synaptic strength is genetically determined and cannot be modified during "life". B) *Adaptive synapses*: Adaptive rule on 2 bits (four rules) and learning rate (0.0, 0.3, 0.6, 0.9) on the remaining 2 bits. The synapses are always randomly initialized when an individual starts its life and then are free to change according to the selected rule. C) *Noisy synapses*: Weight strength on 2 bits and a noise range on the remaining two bits $(0.0, \pm 0.3, \pm 0.6, \pm 0.9)$. The synaptic strength is genetically determined at birth, but a random value extracted from the noise range is freshly computed and added after each sensory motor cycle. A limiting mechanism cuts off sums that exceed the synaptic range $[0, 1]$. This latter condition is used as a control condition to check whether the effects of Hebbian adaptation amount to random synaptic variability.

In previous work we always resorted to Synapse Encoding and showed that evolution of adaptive synapses for an obstacle avoidance task develops levels of performance similar to those obtained by evolution of genetically-determined synapses [2–4]. Since in our approach adaptive synapses do not require a specification of initial strength, in this new set of experiments we have employed Node Encoding for adaptive synapses and systematically compared it to genetically-determined controllers using both Synapse Encoding and Node Encoding.

3 Experiments

The experiments have been carried out in simulations sampling sensor activation and adding 5% uniform noise to these values [9]. In addition, we have repeated the evolutionary experiments for the most significative conditions on the physical robot. Since the results on the physical robot do not differ significantly from those obtained in simulation, we report them in the appendix. For each experimental condition, 10 different[3] populations of 100 individuals each have been independently evolved for 200 generations. Each individual is tested three times

[3] Using different sequences of random number.

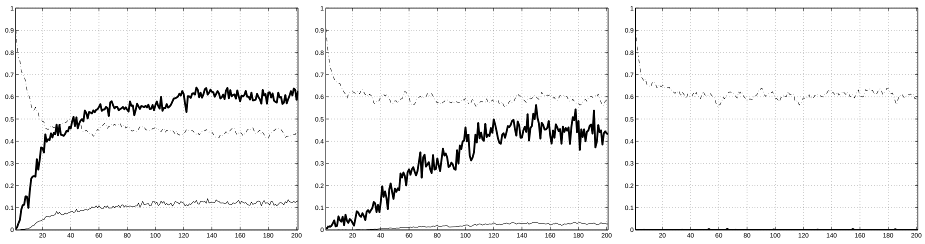

Fig. 3. Comparison of adaptive synapses with Node Encoding (*left*) versus genetically-determined synapses with Synapse Encoding (*center*) and Node Encoding (*right*). Thick line=best individual; thin line=population average; dashed line=genetic diversity. Each data point is an average over 10 replications with different random initializations.

and the fitness value is averaged. The 20 best individuals reproduce by making 5 copies of their genetic string. Strings are crossed over with probability 0.2 and mutated with probability 0.05 (per bit). In the case of adaptive synapses, synaptic weights of individuals are randomly initialized within the range $[0.0, 0.1]$ at the beginning of each test.

The fitness results reported in figure 3 show that individuals with adaptive synapses and Node Encoding (graph on the left) are much better than individuals with genetically-determined synapses and Synapse Encoding (graph in the center) in that: a) both the fitness of the best individuals and of the population report higher values (0.6 against 0.5); b) they reach the best value obtained by genetically-determined individuals in less than half generations (40 against more than 100); c) they display much less variability across generations. Individuals evolved with genetically-determined synapses and Node Encoding (graph on the right) never managed to complete the task reliably in any of the ten replications. The genetic variance[4] of the populations of adaptive individuals is reduced more markedly than in all other conditions, *probably* indicating a more reliable selection of individuals and preservation of genetic building blocks.

Two sets of control experiments –one using Synapse Encoding (figure 4, left) and the other Node Encoding (figure 4, right)– have been carried out using noisy synapses in order to check whether the improvements obtained by evolving adaptive synapses were simply due to a random sampling of the fitness surface surrounding each individual. In both cases the results were considerably worse than those obtained with adaptive synapses (figure 3, left) and than those obtained with genetically-determined synapses (figure 3, center).

[4] Measured as the average dispersion of individual vectors from the center of mass of the population and further normalized by the string length.

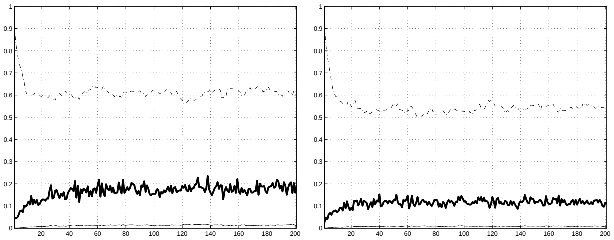

Fig. 4. Evolution of noisy synapses using Node Encoding (*left*) and Synapse Encoding (*right*). Thick line=best individual; thin line=population average; dashed line=genetic diversity. Each data point is an average over 10 replications with different random initializations.

3.1 Scaling up

The choice of a neural architecture is often difficult and may affect the outcome of an experiment. A large architecture may be computationally more powerful, but it may also entail a larger genotype and stronger epistatic effects. Unless one knows that a larger search space for the genotype/phenotype mapping considered has the same proportion of solutions as a smaller one, shorter genotypes may be preferrable because evolutionary search could be faster and more effective.

We have performed a new series of experiments using a larger neural network. The architecture shown in figure 2 was extended by adding 20 hidden neurons. These neurons were fully connected to themselves and to other neurons in the network, but did not receive sensory input and were not used to set the speeds of the wheels. The length of the genetic string grows from 60 to 160 bits for Node Encoding and from 720 to 5120 bits for Synapse Encoding. The results shown in figure 5 indicate that evolution of adaptive synapse with Node Encoding reports fitness values still comparable to the case of a smaller network;

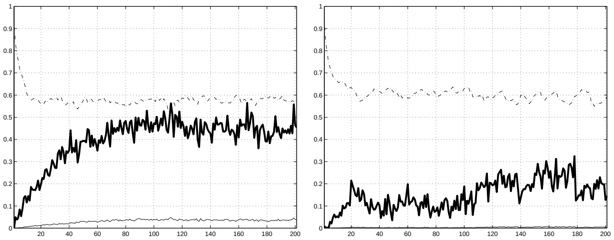

Fig. 5. Evolution of a large controller with 20 hidden nodes. *Left:* Adaptive synapses with Node Encoding. *Right:* Genetically-determined synapses with Synapse Encoding. Thick line=best individual; thin line=population average; dashed line=genetic diversity. Each data point is an average over 10 replications with different random initializations.

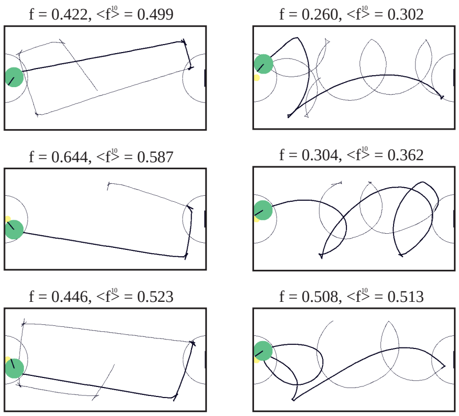
f = 0.422, <f^{10}> = 0.499
f = 0.260, <f^{10}> = 0.302
f = 0.644, <f^{10}> = 0.587
f = 0.304, <f^{10}> = 0.362
f = 0.446, <f^{10}> = 0.523
f = 0.508, <f^{10}> = 0.513

Fig. 6. Behaviors of three best individuals with adaptive synapses and Node Encoding (*left column*) and of three best individuals with genetically-determined synapses and Synapse Encoding (*right column*). Individuals belong to the last generation of three different replications (randomly chosen out of ten) for each condition. When the light is turned on, the trajectory line becomes thick. The corresponding fitness value is printed on the top of each box along with the average fitness of the same individual tested ten times from different positions and orientations.

instead, evolution of genetically-determined controllers with Synapse Encoding is badly affected in this condition. Evolution of genetically-determined synapses with Node Encoding (data not shown) remained closed to zero fitness, whereas evolution of synaptic strength and noise range with both Node Encoding and Synapse Encoding reported the same results as for the smaller network (data not shown).

The fact that genetically-determined controllers with Synapse Encoding perform badly may indicate that the search space here contains proportionally less solutions than the smaller search space of the network pictured in figure 2. The slower convergence and slightly lower fitness values of the controller with adaptive synapses (compare with left graph of figure 3) may be explained by the increased length of the genetic string, but also by the fact that the architecture is fixed and fully connnected. Since in Node Encoding the properties of a node propagate to all incoming synapses, there might be a high number of ”parasitic” connections that cannot be individually eliminated. We shall come back to this point in the final discussion.

4 Behavioral Analysis

Figure 6 shows the behaviors of three best individuals evolved with adaptive synapses and Node Encoding (left) and with genetically-determined weights and Synapse Encoding (right). In both cases individuals aim at the area with the

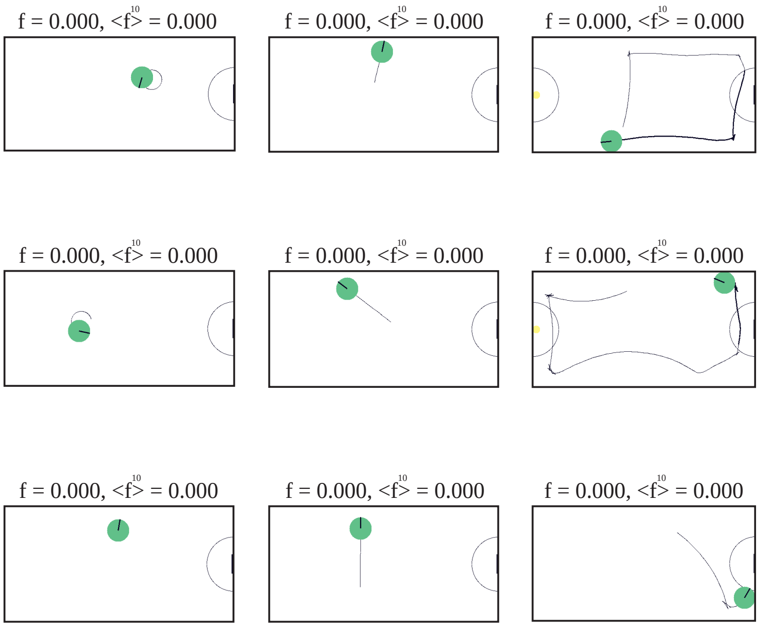

Fig. 7. Disabling adaptation for three best individuals with adaptive synapses (shown on the left of figure 6). *Left column:* Synapses are initialized to random values in the range $[0.0, 0.1]$, as during evolution. *Center column:* Synapses are all initialized to 1.0. *Right column:* Synapses are set to their average value recorded during a full test of the individual. The corresponding fitness value is printed on the top of each box along with the average fitness of the same individual tested ten times from different positions and orientations. The values are always 0.0 because none of the individuals ever manage to complete the task under these test conditions.

light switch[5] and, once the light is turned on, they move towards the light and remain there. The better fitness of the adaptive controllers (given on the top of each box, see figure caption) is given by straight and faster trajectories whereas genetically-determined individuals display loopy trajectories (and sometimes are not capable of standing still on the fitness area, as in the case of the third individual on the bottom right of the figure).

Another set of tests has been carried out to assess the role of adaptation in the behavior of the individuals with adaptive synapses. For example, one might argue that what matters is the sign of the synapse and not its strength as long as it is non-zero, or that adaptive synapses may have the same effect of fixed synapses with strengths set to their average values[6]. The same three best

[5] Their performance is badly affected if the vision input is disabled, indicating that they do not use random search to locate the switch (data not shown).

[6] This latter suggestion was made by Flotzinger [6] who replicated our previous experiments on Adaptive Synapses with Synapse Encoding

individuals with adaptive synapses shown in the left column of figure 6 were tested again disabling adaptation in three different conditions (figure 7). In the first condition the synapses were initialized to small random values in the range $[0.0, 0.1]$ (figure 7, left column), as during evolution. In the second condition, the weights were all set to their maximum strength 1.0 (figure 7, center column). In the third condition the weights were set to their average value (figure 7, right column). (The average values had been previously computed while testing the robot in adaptive mode and recording the synaptic strength of each connection after every update.) For each condition, the three individuals were tested ten times from different positions and orientations. None of the individuals ever managed to complete the task in any of the three conditions.

5 Conclusions

We have shown through a set of systematic comparisons that evolution of adaptive synapses brings a number of advantages with respect to evolution of synaptic weights. It can generate viable controllers in much less generations and the evolved controllers display more performant behaviors. Since adaptive synapses here need not be specified on the genetic string because their strength is always randomly initialized at the beginning of an individual's test, this approach can rely on a very compact genetic encoding that specifies only the adaptive properties of individual nodes. Such a compact encoding scales up very well to large networks with many synapses. The data obtained from control experiments with noisy synapses and from behavioral tests of evolved individuals with adaptation disabled all suggest that Hebbian adaptation plays a specific role in the functioning of the controllers both during evolution and during the "life" of an individual.

When describing our controllers with changing synapses, we have accurately avoided the term "learning" because we have no evidence that the controller acquires new knowledge or skills, or that it may easily acquire new abilities for a different task (implementing, for example, something functionally similar to reinforcement learning). However, we have used the term "adaptation" because synapses change according to the states of the sensors and of the other neurons in the controller of the robot. In other words they adapt their initial random configuration to a dynamically-stable configuration that depends on the behavior of the robot. The adaptation rules are genetically specified and have evolved to satisfy a specific fitness function. As they stand, our results indicate that we have developed a smart genetic specification of neural controllers suitable for evolution. One of our current projects aims at establishing to what extent can this approach scale up to more complex behavioral tasks and to other more traditional learning problems. Another project aims at testing (and possibly extending) this approach for behavioral problems where learning is traditionally considered necessary.

We think that evolution of adaptive synapses may be very suitable for evolving neural morphologies where one cannot specify the strength of individual

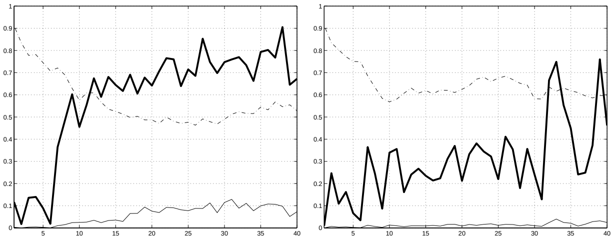

Fig. 8. Comparison of adaptive synapses with Node Encoding (*left*) versus genetically-determined synapses with Synapse Encoding (*right*) for experiments carried out on the physical robot.

synapses on the genotype and at the same time wishes to keep the genetic string as compact as possible. The methods proposed so far for evolution of morphologies all need very complex genetic encoding, require much domain-specific knowledge (e.g., symmetries, connectivity types), and have not yet been shown to be competitive with direct-coding methods. The Node Encoding scheme that we have proposed may be a first step in the direction of morphology evolution in the sense that synapse details are not specified in the genetic code, but are taken care of by adaptive online rules. In a current project we are extending our approach by adding genes for expression of connection growth and recursive rules to the node specification.

Appendix: Evolution on the physical robot

Two sets of experiments have been repeated on the physical robot: adaptive synapses with Node Encoding and genetically-determined synapses with Synapse Encoding (figure 8). The main differences from those carried out in simulations are: the population size is 80, each run lasts 40 generations, only one run has been carried out for each condition, and each individual is tested only once in the environment. The last restriction means that the effects of chance are more marked on the performance and are the cause of the higher oscillation observed, especially for the individual with genetically-determined synapses. These data should be compared to those shown in the graphs at the left and center of figure 3. The performances obtained with the physical robots are better than those obtained in simulation because the latter include severe constraints. For example, when a simulated robot pushes against a wall, it cannot move unless it backs away; instead, real robots can often get away by sliding against the walls.

Acknowledgements

Review of this paper has been organized by Inman Harvey with the help of three independent referees. We thank the referees and Inman for useful comments and criticisms, especially on the size of the search space in evolution. D.F. acknowledges support by the Swiss National Science Foundation, project nr. 21-49174.26. J.U. is supported by grant nr. BF197.136-AK from the Basque government.

References

1. R. K Belew and M. Mitchell, editors. *Adaptive Individuals in Evolving Populations: Models and Algorithms.* Addison-Wesley, Redwood City, CA, 1996.
2. D. Floreano and F. Mondada. Evolution of plastic neurocontrollers for situated agents. In P. Maes, M. Matarić, J-A. Meyer, J. Pollack, H. Roitblat, and S. Wilson, editors, *From Animals to Animats IV: Proceedings of the Fourth International Conference on Simulation of Adaptive Behavior*, pages 402–410. MIT Press-Bradford Books, Cambridge, MA, 1996.
3. D. Floreano and F. Mondada. Evolutionary Neurocontrollers for Autonomous Mobile Robots. *Neural Networks*, 11:1461–1478, 1998.
4. D. Floreano and S. Nolfi. Adaptive behavior in competing co-evolving species. In P. Husbands and I. Harvey, editors, *Proceedings of the 4th European Conference on Artificial Life*, Cambridge, MA, 1997. MIT Press.
5. D. Floreano and J. I. Urzelai. Evolution and learning in autonomous robots. In D. Mange and M. Tomassini, editors, *Bio-Inspired Computing Systems.* PPUR, Lausanne, 1998.
6. D. Flotzinger. Evolving plastic neural network controllers for autonomous robots. Msc dissertation 9580131, COGS, University of Sussex at Brighton, 1996.
7. I. Harvey. Is there another new factor in evolution? *Evolutionary Computation*, 4(3):313–329, 1997.
8. G. E. Hinton and S. J. Nowlan. How learning can guide evolution. *Complex Systems*, 1:495–502, 1987.
9. O. Miglino, H. H. Lund, and S. Nolfi. Evolving Mobile Robots in Simulated and Real Environments. *Artificial Life*, 2:417–434, 1996.
10. S. Nolfi and D. Floreano. Learning and evolution. *Autonomous Robots*, 7(1):forthcoming, 1999.
11. D. Parisi and S. Nolfi. How learning can influence evolution within a non-lamarckian framework. In R. K. Belew and M. Mitchell, editors, *Adaptive Individuals in Evolving Populations.* MIT Press, Cambridge (MA), 1996.
12. X. Yao. A review of evolutionary artificial neural networks. *International Journal of Intelligent Systems*, 4:203–222, 1993.