Evolving Neuro-Modules and their Interfaces to Control Autonomous Robots *

B. Lara[1], M. Hülse[1], and F. Pasemann[1,2]

[1]TheorieLabor, Friedrich Schiller University, D-07740 Jena

[2]Max Planck Institute for Mathematics in the Sciences, D-04103 Leipzig

Abstract

An evolutionary algorithm for the creation of recurrent network structures is presented. The aim is to develop neural networks controlling the behaviour of miniature robots. Two neuro-modules are created separately using this evolutionary approach. The first neuro-module gives the agents the ability to move within an environment without colliding with obstacles. The second neuro-module provides the agents with a phototropic behaviour. The interaction of the neuro-modules is then investigated evolving the necessary interface to provide the agents with a coherent obstacle avoidance and phototropic behaviour. The evolution process is carried out in a simulated environment and individuals with high performance are also tested on a physical environment with the use of Khepera robots.

*in: Proceedings of the *The 5th World Multi-Conference on Systemics, Cybernetics and Informatics* (SCI 2001), Orlando, Florida USA, July 22-25, 2001.

1 Introduction

Within the frame of Synthetic Modeling and Embodied Cognitive Science [8] an evolutionary algorithm is presented. The Evolution of Neural Systems by Stochastic Synthesis (ENS^3) [7], evolves the structure and size of artificial neural networks and optimizes the corresponding parameters at the same time. It is designed especially to generate networks with recurrent connectivity. In acquiring both network topology and parameter optimization simultaneously it is similar to the GNARL algorithm [1].

The starting hypothesis for the presented experiments is that internal dynamical properties of recurrent networks are the basis for cognitive capabilities and therefore should be also used for the control of robot behavior. It is well known that even small recurrent networks present a variety of dynamical characteristics such as bifurcating attractors, hysteresis effects, and chaotic attractors co-existing with periodic or quasi-periodic attractors [6]. So the idea is to use the ENS^3 algorithm to generate recurrent network structures for agents having to exist and survive in a complex environment.

Modularity is analyzed in the context of behaviour from parallel processes. In the field of Embodied Cognitive Science, interconnectivity among neuro-modules with different behavioural functionality is a major issue. In this paper, the interface between two modules is evolved using the ENS^3, minimizing the designer influence on the control strategy for the entire agent. The resulting structure presents complex dynamics and interconnectivity and has graceful degradation.

For the experiments described in this investigation Khepera robots are used [3]. Evolution is carried out using a Khepera simulator [2]. Finally, the recurrent networks with highest performance are tested on real Khepera robots. Section 2 of this paper introduces the ENS^3 evolutionary algorithm. Section 3 describes the experimental set-up and the results are outlined in Section 4 which presents also a discussion of these results. Finally Section 5 concludes the paper.

2 ENS^3 Description

The algorithm was originally designed to study the appearance of complex dynamics and the corresponding structure-function relationship in artificial sensorimotor systems. The basic assumption is that in "intelligent" systems the behavior relevant features of neural subsystems, called neuro-modules, are due to their internal dynamical properties which are provided by their recurrent connectivity structure. But the structure and interaction of such

nonlinear subsystems with recurrences in general can not be designed to produce a reasonably complex dynamical behavior. Thus, the main focus of the ENS^3-algorithm is to evolve an appropriate structure of artifical neural networks.

To start the algorithm one first has to decide which type of neurons to use for the network. For output and internal units standard additive neurons with sigmoidal transfer functions will be used, whereby input units serve only as buffers. The number of input and output units is chosen according to the definition of the problem. Nothing else is determined, neither the number of internal units nor their connectivity, i.e. self-connections and every kind of recurrences are allowed, as well as excitatory and inhibitory connections. Because input units are only buffering data, no backward connections to these are allowed.

To evolve the desired neuro-module a population $p(t)$ of $n(t)$ neuro-modules undergoing a variation-evaluation-selection loop is considered, i.e. $p(t+1) = S \cdot E \cdot V \cdot p(t)$. The *variation operator* V is defined as a stochastic operator, and allows for the insertion and deletion of neurons and connections as well as for alterations of bias and weight terms. Its action is determined by fixed per-neuron and per-connection probabilities. The *evaluation operator* E is defined problem-specific, and it is usually given in terms of a fitness function. After evaluating the performance of each individual network in the population the number of network copies passed from the old to the new population depends on the *selection operator* S. This operator performs the differential survival of the varied members of the population according to evaluation results. In consequence of this selection process the average performance of the population will tend to increase. Thus, after repeated passes through the variation-evaluation-selection loop populations with networks solving the problem can be expected to emerge.

3 Experimental Set-Up

The aim of the experiments is to find recurrent neural networks which are able to control specific behavior of the Khepera robots and to study the structure and connectivity of parallel processes. To achieve this, modules are evolved using the ENS^3 algorithm described in Section 2. As aforementioned the only necessary initial conditions, on terms of structure and size, are the input and output neurons. These constrains are defined by the structure of the Khepera robot.

The Khepera robots are miniature cylindrical robots (diameter of 55 mm), with the basic configuration the robot has 8 Infra Red (IR) sensors (6 at the

front of the robot and 2 at the rear), and two wheels, each controlled by a DC motor with an incremental encoder (10 pulses per mm of advancement of the robot). Each sensor can be used in two modalities: as a proximity sensor (by emitting and measuring the reflected Infra Red light), and as a light sensitive sensor (by measuring the Infra Red component of the environment light).

Two basic neuro-modules are studied: the first module provides the agents with an obstacle avoidance behaviour, the second a light seeking / following behaviour. The time interval for evaluating these individuals is set to 2000 simulator time steps. Furthermore, the size of resulting networks can be influenced by changing the probabilities of increase and/or decrease of neurons and/or synapsis and adding cost terms for neurons and connections to the given fitness functions, as described in Section 2.

For evaluating the performance of the robots the two dimensional Khepera simulator [2] was used. A suitable interface for the communication between the Kephera simulator and the evolutionary program was implemented. In this way every neuro-module of a generation is sent to the simulator and will be directly used for controlling the simulated Kephera. The start position of the robots is randomly set for every generation but fixed for the evaluation of all neuro-modules in one generation. The fitness function for the calculation of the performance is implemented in the simulator. Performance of every neuro-module (controlling the simulated Khepera) is calculated for the life span of each individual in each generation and then sent back to the ENS^3.

4 Experiments and Results

Two neuro modules are evolved using the ENS^3 evolutionary approach. To complement the algorithm, the necessary fitness functions have to be coded and interfaced with the performance measure of the evolution selection process. The whole approach, through the fitness functions attempts to minimize the programmer/human intervention in the creation and design of the resulting controllers. The modules are created as follows:

4.1 OANM - Obstacle Avoidance Neuro-Module

For evolving neuro-modules which have an obstacle avoidance behavior the 8 infra-red (IR) proximity sensors of the Khepera are used. The number of sensors determines the number of input neurons of the neuro-controllers. To control the motors governing the robot wheels positive and negative signals are required. A $tanh$ transfer function satisfies this requirement and is therefore used for the output neurons as well as for hidden neurons. The

bias term of the *tanh* function is set constant to zero for all of the neurons throughout the whole evolution process. The initial structure of the individual neuro-modules have only 8 linear input neurons, as buffers and two nonlinear output neurons.

The link between the simulated agents and the evolution algorithm is provided by a performance measure of every individual in every generation. This performance measure is coded as a fitness function, Φ. To obtain obstacle avoidance behaviour a variation on [4] is used:

$$\Phi_{oa} := \sum_{t=1}^{2000} (M_l + M_r) \cdot (1 - \Delta_m) \cdot (1 - H_{ir}) \,, \tag{1}$$

where $-1 < M_l, M_r < 1$ represent the displacement values of the left and right motors respectively. Optimization of this term obliges the agents to maximize displacement. Δ_m is given by:

$$\Delta_m = | \, M_r - M_l \, | \,, \tag{2}$$

for $0 < \Delta_m < 2$, and aims to reduce the turning of the agents, as an optimization of the previous term could be made by rotating on its own axis. Finally, the last term deals with obstacle avoidance, where H_{ir} is the value of the IR sensor with highest activation and $0 \leq H_{ir} \leq 1$.

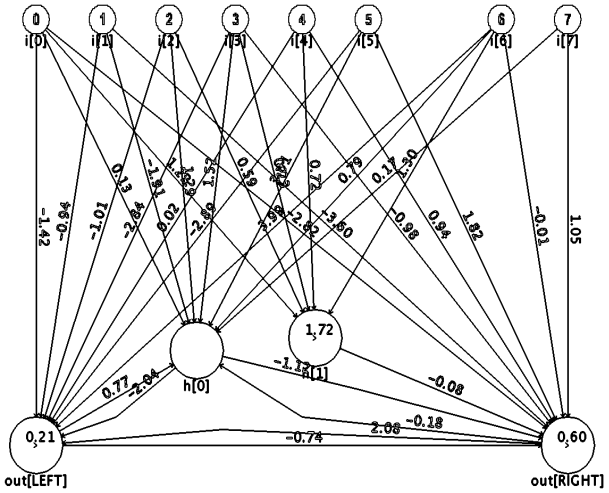

Figure 1: Obstacle avoidance neuro-module (OANM) evolved with respect to the fitness function Φ_{oa} (Eq. 1).

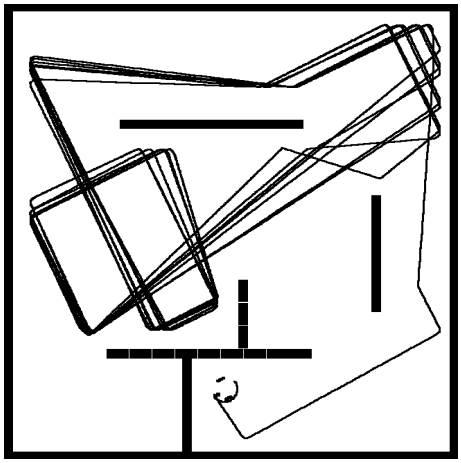

Figure 2: Behaviour generated by OANM in a simulated world.

Environments are build on the Khepera simulator, a typical simulated world is shown in Fig. 2. It is worth noting that the evolved network structures are expected to behave at least with the same performance in the physical robot as they do in the simulated environments.

A network, which generates good behavior in the simulated environment as well as on the physical robot is shown in Fig. 1. Input neurons are represented by the 8 top neurons, with the position they have in both, the simulated and real robot.

It uses two hidden neurons, one of this has a self-excitatory connection larger than 1 which establishes it as a hysteresis element: For certain different inputs to the neuron there are sudden jumps of neuron activity from one fixed point attractor to another [5]. It seems that this hysteresis effect is used for instance to get out of situations which usually generate deadlocks. The output neuron driving the left motor is self-excitatory, and has an inhibitory connection from the other output neuron, which drives the right motor.

The behavior in the real robot generated by this network is similar to the plotted paths in Fig. 2. The robot presents obstacle avoidance as well as exploration abilities. Letting the robot move in its physical environment for a period of time makes the agent visit almost all areas within reach. This of course can not be achieved by pure wall following behavior.

The good exploration abilities and performance for navigating without colliding with obstacles make this neuro-module a perfect candidate for the study of modularity.

4.2 LSNM - Light Seeking Neuro-Module

For the second experiment in addition to the 8 IR sensors the 8 light sensors of the Khepera are used; i.e., now there are 16 input neurons.

The goal in this experiment is to find a light source as fast as possible and move towards it. In the case of a moving light source the robot should follow it. Aiming to find a network structure only for this type of behavior the individuals should not be concerned with obstacles and are therefore evolved in simulated world containing only lights. The fitness function is coded as:

$$\Phi_{ls} := \sum_{t=1}^{2000} (M_l + M_r) \cdot (1 - \Delta_m) \cdot (1 - H_{ls}) \,, \tag{3}$$

where M_l, M_r and Δ_m are defined as in Eq. 1. The last term deals with the agents seeking/following light as H_{ls} is a value proportional to the activity of the light sensitive sensor with the highest activity. A typical world where these individuals are evolved is shown in Fig. 4.

After only a few generations the evolved structures present light seeking behaviour. The individuals advance towards the light when this is within the reach of the light sensors. The individual with the highest performance of the 50th generation is shown in Fig. 3. Again, black circles show input neurons, the first group of 8 neurons represents the proximity sensors, the second group (bigger circles) represents the light sensors. In this case the sensors in the rear are situated at the bottom of the image. It can be observed that there exist connections from the proximity sensors, even though there was no specific data requirement from their input in the definition of the fitness function. These connections are related to the straight trajectory of the robot in the absence of light. Trajectories, with different starting positions, followed by the best individual of generation 50 are shown in Fig.4.

When the physical robot was used, the individuals go in straight trajectories towards the light. If the light is moved they follow it. When there is no light present in the environment the individuals move in a wide semi-circular path. If the connections coming from the first 8 inputs neurons were manually deleted the robot presented a different behaviour: In the absence of light, it will turn in its own central axis, but when presented with the light source it would advance towards it and keep following it if the light source was in movement.

It is seen that disregarding the starting position the robot always move towards the light and stay around it. Again, this neuro-module would be used for the study of connectivity presented in the following subsection.

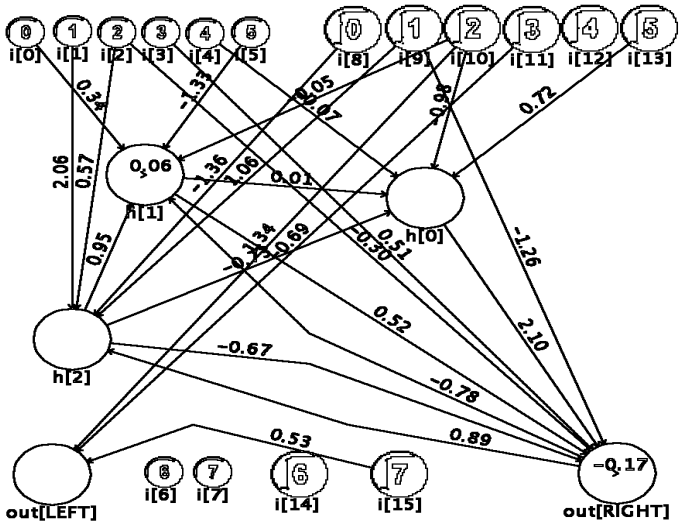

Figure 3: Light seeking neuro-module (LSNM) evolved with respect to the fitness function Φ_{ls} (Eq. 3).

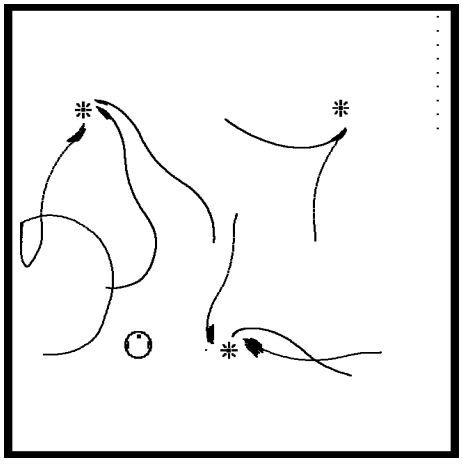

Figure 4: Behaviour generated by the LSNM in a simulated world.

4.3 Evolving an Interface

The two previously described neuro-modules are created separately and with each independently the agents present the desired respective behaviour. However, as expected, the superimposition of this two modules does not provide the agents with obstacle avoidance *and* phototropic behaviour. In order to achieve this, the two neuro-modules are superimposed and subjected to a further evolution process using the following fitness function:

$$\Phi_{in} := \sum_{t=1}^{2000} \alpha(1 - H_{ir}) \; + \; \beta(1 - H_{ls}), \tag{4}$$

where H_{ir} and H_{ls} are proportional to the values of the proximity and light sensors with highest activity respectively. During the evolution process, the two already evolved neuro-modules are not allowed to change in number of neurons nor in synaptic values, forcing the evolution to create an interface between the already existing structures. The simulated environments for the evolution of the interface contained both, obstacles and lights (Fig. 6)

At the beginning of the evolution process the created structures grew considerably in both, connections and number of neurons. Only the increase of costs for these made the networks decrease in size. A typical structure is shown in Fig. 5.

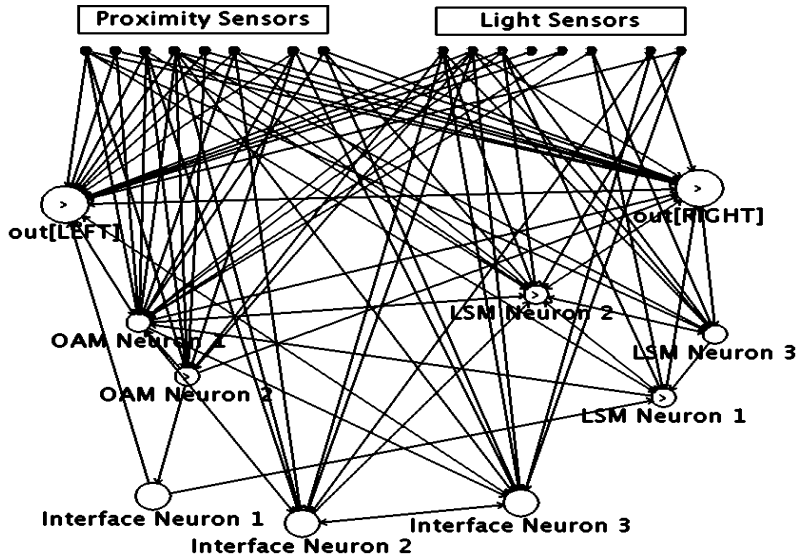

Figure 5: OANM, LSNM and respective interface evolved with fitness function Φ_{in} (Eq. 4).

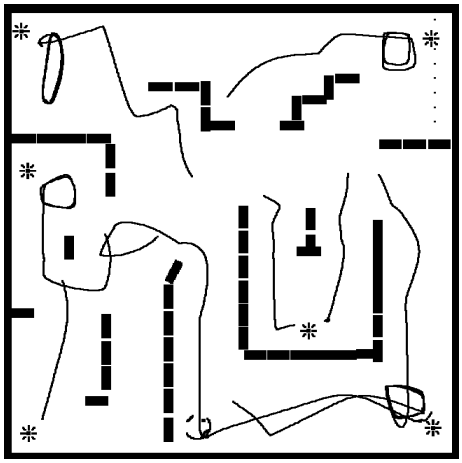

Figure 6: Behaviour generated by OANM, LSNM and respective interface in a simulated world.

The bottom three neurons represent the interface for the two original modules. It is worth noting that if the two modules are manually removed the remaining neurons and synapsis (representing the interface) do not present capabilities for obstacle avoidance and/or light seeking. This test was done to rule out the possibility that the functionality of the original modules had been overwritten by the new fitness function and respective evolution process.

Typical behavior of this network structure is shown in Fig. 6. It can be seen that the agent looks for light while avoiding obstacles.

Despite the complexity of the task, the agent presents high performance for both tasks. To further investigate the evolved structure different neurons were manually removed. The performance of the agent decreases smoothly, up to a point (i.e. when all the interface neurons are removed) where the behaviour it is not anymore that of the original structure. This graceful degradation characteristic is typical of neuro-structures in artificial as well as in real brains.

5 Conclusions and Discussion

An evolutionary algorithm that optimizes the structure and characteristics of recurrent neural networks for control of miniature robots has been presented. Two different *tasks* were investigated. For both tasks, the results of evolution present small structures that exhibit complex behaviour after relatively small number of generations.

A novel approach to the development of modules with different (and overlapping) functionality was presented. It is worth noting that the evolved interface has to deal with two main problems. First, in order to be efficient both modules need to write their output to the same neurons. Second, the two different types of Khepera sensors are embedded on the same device. On the one hand, when working as IR sensors, these present high activation when near an obstacle and when working as light sensors present low activation when near a light source. Despite these facts, the interface provides the solution to the task with minimum designer intervention.

Further analysis of the structure reveals oscillatory behaviour on the output to the right motor, this is assumed to be due to the high connectivity of the neurons. This neuron is implicated in loops with high synaptic weights presenting complex dynamics.

The activation of hidden neurons corresponds to the task they are supposed to be engaged on. This is, when the agent finds an obstacle on its way, neurons belonging to the OANM present the same activation pattern as when they were the only present module. Correspondingly, neurons in the LSNM have the same behaviour when encountered with light sources.

References

[1] Angeline, P.J., Saunders, G.B., Pollack J.B. (1994), An Evolutionary Algorithm that Evolves Recurrent Neural Networks, *EEE Transactions on Neural Networks*, **5**, 54–65.

[2] Michel, O., *Khepera Simulator* Package version 2.0: Freeware mobile robot simulator written at the University of Nice Sophia-Antipolis by Oliver Michel. Downloadable from the World Wide Web at `http://wwwi3s.unice.fr/~om/khep-sim.html`

[3] Mondala, F., Franzi, E., and Ienne, P.(1993), Mobile robots miniaturization: a tool for investigation in Control Algorithms. In *Proceedings of ISER' 93*, Kyoto, October 1993.

[4] Nolfi, S., and Floreano, D. (2000), *Evolutionary Robotics: The Biology, Intelligence, and Technology of Self-Organizing Machines* MIT Press, Cambridge.

[5] Pasemann, F. (1993), Dynamics of a single model neuron, *International Journal of Bifurcation and Chaos*, **2**, 271–278.

[6] Pasemann, F. (1998), Structure and Dynamics of Recurrent Neuromodules, *Theory in Biosciences,* **117***, 1–17*

[7] Pasemann, F. (1998), Evolving neurocontrollers for balancing an inverted pendulum, *Network: Computation in Neural Systems*, **9**, 495–511.

[8] Pfeifer, R., and Scheier, C. (2000), *Understanding Intelligence*, MIT Press, Cambridge.