Evolving integrated controllers for autonomous learning robots using dynamic neural networks

Elio Tuci **Inman Harvey** **Matt Quinn**
Centre for Computational Neurosciences and Robotics,
School of Cognitive and Computing Sciences
University of Sussex, Brighton BN1 9QH, United Kingdom
phone: (+44) (0)1273 872945 - fax (+44) (0)1273 671 320
eliot, inmanh, matthewq@sussex.ac.uk

Abstract

In 1994, Yamauchi and Beer (1994) attempted to evolve a dynamic neural network as a control system for a simulated agent capable of performing learning behaviour. They tried to evolve an *integrated* network, i.e. not modularized; this attempt failed. They ended up recommending independent evolution of separate controller modules, arbitrarily partitioned by the researcher. Moreover, they "provided" the agents with hard-wired reinforcement signals.

The model we describe in this paper demonstrates that it is possible to evolve an *integrated* dynamic neural network that successfully controls the behaviour of a khepera robot engaged in a simple learning task. We show that dynamic neural networks, based on leaky-integrator neuron, shaped by evolution, appear to be able to integrate reactive and learned behaviour with an *integrated* control system which also benefits from its own evolved reinforcement signal.

1 Introduction

In 1994, Yamauchi and Beer (1994) pointed out how research on learning robots typically drew a sharp distinction between the mechanisms responsible for an agent's behaviour and those responsible for learning (Kaelbling, 1993). They claimed that this distinction is difficult to defend biologically, because many of the same biological processes are involved in both reactive and learned behaviour. They based their claim looking at current theory in the study of learning in biology, which seems to suggest that animals' learning capabilities are exquisitely tuned by evolution, to the particular niche that they occupy (Garcia and Koelling, 1966; Wilcoxon et al., 1971; Davey, 1989).

Therefore, they set out an experiment in which they tried to evolve an autonomous learning robot by selectively integrating the necessary plasticity directly into the mechanisms responsible for the robot's behaviour. Their methodology was meant to guarantee that any structural or functional decomposition that may be found in the analysis of the behaviour or in the evolved control system should result entirely from the evolutionary contingencies (as it is in animals). Previous research on learning robots relied on explicitly hand-designed decomposition or modularization both of the behaviour and of the robot's control system (Kaelbling, 1993).

The approach undertaken by Yamauchi and Beer (1994), is part of a more general way of assuming an evolutionary perspective in designing control architecture for autonomous agents. This research area is generally referred to as Evolutionary Robotics (Harvey et al., 1997; Nolfi and Floreano, 2000). The potential benefits of an evolutionary approach to the design of simulated or real agents' control systems and possibly agents' morphologies, either for engineering purposes or as a new methodology for biology, are widely debated (see Webb, 2000; Nolfi, 1998). However, generally speaking, the appeal of an evolutionary approach to robotics is two-fold. Firstly, and most basically, it offers the possibility of automating a complex design task (Meyer et al., 1998; Nolfi and Floreano, 2000; Harvey et al., 1997). Secondly, since artificial evolution needs neither to understand, nor to decompose a problem in order to find a solution, it offers the possibly of exploring regions of the design space that conventional design approaches are often constrained to ignore (Harvey et al., 1992).

A growing amount of research in Evolutionary Robotics has been focusing on the evolution of controllers with the ability to modify the behaviour of a robot, in order to adapt to variation in its operating conditions. By variation we mean changes in the relationship between a robot's sensors and actuators, and its environment (e.g. Floreano and Urzelai, 2000, 2001b,a; Floreano and Mondada, 1996; Nolfi and Floreano, 1999; Eggenberger et al., 1999; DiPaolo, 2000). Yamauchi and Beer's approach clearly represents one of the first attempts in which continuous time recurrent neural networks have been exploited to integrate reactive, sequential and learned behaviour in a simulated robot.

Apart from Yamauchi and Beer (1994), current research in this area has been investigating the use of neural network controllers with some form of synaptic plasticity, that is, networks which incorporate mechanisms that change connection weights. Yamauchi and Beer (1994) did something distinctively different and unique, due to the characteristics of the control system and the singularity of the learning task employed. They employed continuous time recurrent neural networks (CTRNNs: described in their paper and also briefly in section 3.4 below), with genetically defined and *fixed* (as contrasted with plastic) connection weights, and leaky-integrator neuron (i.e. neurons whose activation decays with time). There is a further difference between the learning task described in their paper and the robot learning experiments of (e.g. Nolfi and Parisi, 1997; Floreano and Urzelai, 2000). The former specifically required the robot to learn the current relationship between a goal and a landmark, a relationship that changes from time to time; the latter require the robot to adapt to rather general variation concerning the relationship between a robot's sensors and actuators, and its environment — such as variation in the color of the arena walls (as in Nolfi and Parisi, 1997) or variation in lighting levels (as in Floreano and Urzelai, 2000).

However, Yamauchi and Beer (1994)'s attempts to evolve an *integrated* (i.e. not modularized) dynamic neural network, as a control system for a simulated agent capable of performing learning behaviour, failed. They ended up recommending independent evolution of separate controller modules, some of which are dedicated to reactive and sequencing behaviour and others dedicated to learning. Moreover, they hard-wired into the model a reinforcement signal that works as a feedback signal for the learning module. For reasons to be explained in section 2 we believe that is important to carry through the Yamauchi and Beer experiments as originally intended, even though they were forced to make compromises that we find unsatisfactory.

The model we are describing in this paper is intended simply as a proof of concept. We go beyond the Yamauchi and Beer (1994) approach, demonstrating that it is possible to evolve an *integrated* dynamic neural network, with fixed connection weights and leaky-integrator neurons that successfully control the behaviour of a khepera robot engaged in a learning task, similar to one proposed by (Yamauchi and Beer, 1994). We will bring evidence that the leaky-integrator neuron, if shaped by evolution, appears to be able to integrate reactive and learned behaviour within a single control system that is not explicitly modularized. The *integrated* control system we are describing also benefits from its own evolved reinforcement signals to generate the appropriate behavioural responses; this is in contrast to their explicit introduction of hard-wired reinforcement signals.

1.1 Structure of the paper

In what follows, we will firstly present a very brief description of the Yamauchi and Beer (1994) experiment (see section 2). We will then point out that, although a promising one, the Yamauchi and Beer (1994)'s approach is not completely satisfactory because the autonomy of the simulated agent is severely compromised by an external reinforcement signal and externally imposed modularization of the controller 2.1.

In section 3 we describe the methodology used, the task, the evaluation function used for evolution, the integrated network, the simulated robot and the Genetic Algorithm.

In section 4 we show experimental results. In section 5 we make comparisons with the original Yamauchi and Beer (1994) experiment. We discuss the differences that might have singularly contribute to make it possible for us to evolve an *integrated* dynamic neural network where we assume neither an *a priori* separated controller module nor an *a priori* external reinforcement signal as in Yamauchi and Beer (1994). Conclusions are drawn in section 6.

2 Yamauchi and Beer's model of a learning robot

Yamauchi and Beer set up an experiment where a simulated robot had to repeatedly find a goal using a landmark for guidance. The relationship between the position of the goal and the position of the landmark changed occasionally within the sequence of searches, so successful behaviour required the robot to "learn" when these changes occurred.

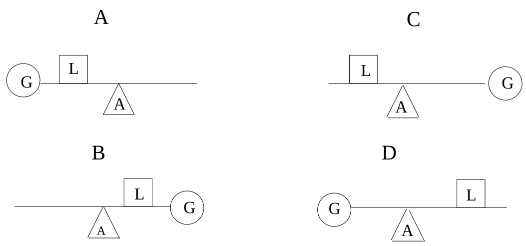

Figure 1: This picture is adopted from Yamauchi and Beer (1994). Environment for the one-dimensional navigation task. The triangle represents the agent, the circle represents the goal and the rectangle represents the landmark. The landmark-near environments (A and B) are on the left, the landmark-far environments (C and D) are on the right.

In the Yamauchi and Beer's model, an agent could move in either direction along a one-dimensional continuum environment (see figure 1). The environment contains a goal and a landmark. At the start of each trial, the agent was positioned in the center, and the goal was positioned randomly at either the left or right end. There were two possible way in which the landmark can be located within the environment. In the landmark-near environments 1A and 1B, the landmark was located on the same side of the agent as the goal. In landmark-far environments 1C and 1D, the landmark was located on the opposite side of the agent from the goal. The agent's task was to learn, over a series of successive trials, whether the current environment was landmark-far or landmark-near, and then reach the goal. Each trial lasted until either the agent reached the goal, the agent reached the non-goal end of the continuum, or the time limit was exceeded.

Attempts by Yamauchi and Beer to evolve a single fully CTRNN capable of solving the task were unsuccessful, so they used a modular, incremental approach instead, one in which the network architectures was split into three separately evolved subnetworks: one for goal-seeking behaviour in a landmark-far environment, one for goal-seeking behaviour in a landmark-near environment and one for environment classification. Each subnetwork consisted of a five-neuron fully interconnected CTRNN. All of these subnetworks had a sensor input for landmark detection, activated when the agents were directly in contact with the landmark, and two motor outputs (i.e. a left and a right motor neuron). In addition, the classifier network had a sensor input for reinforcement, which was provided every time the network reached the goal, and a classification output used to determine the environment type.

The two goal-seeking networks were specifically evolved to properly navigate in one or the other type of environment.

The classifier network were specifically evolved to control the behaviour of the agent during the first trial and to decide whether the agent was living in a landmark-far or in a landmark-near environment. The classification of the environment was landmark-far if the output of the classification neuron at the end of the first trial was less than 0.5, and landmark-near otherwise.

The evolved strategy was quite simple. The classifier network starts with the classifier output high, indicating a choice for landmark-near environment, and moving the robot towards the left side. Sensing a landmark causes the reversion of the direction of movement, while a reinforcement signal, hard-wired in to the model, makes the classifier output go low, indicating a choice for the landmark-far environment. If no reinforcement is provided, the classifier output remains high regardless of whether or not the agents had encountered a landmark.

These evolved responses make the classifier network able to correctly identify all four possible combination of environment type and goal location.

2.1 Thoughts and comments

We believe that the Yamauchi and Beer's model of evolved learning behaviour in artificial agents is valuable because it clearly represents one of the first attempts in which artificial evolution has been exploited to design the internal dynamics of a CTRNN to integrate the perceptions of the agents and to determine the agent's actions based upon these perceptions. In their model a fitness function rewards the behavioural responses that fulfil the conditions for learning.

However, they made use of an 'external' (i.e. not evolved) reinforcement signal, directly available to the robot, which signals to the robot when it performs correctly. This reward signal is clearly an externally imposed supervision, which severely compromises the autonomy of the system. Moreover, their attempts to evolve an *integrated* CTRNN, as a control system for a simulated agent capable of performing learning behaviour, failed. They ended up recommending independent evolution of separate controller modules, some of which are dedicated to reactive and sequencing behaviour and others dedicated to learning. This modularization of the controller reintroduces a structural and functional decomposition which is also externally imposed to facilitate the time-integration of different behavioural responses.

We argue that the Yamauchi and Beer (1994)'s model, relying as it does on independent evolution of separate controller modules (that are arbitrarily designed by the researcher and clearly dedicated to separate behavioural responses), and on an external reinforcement signal, failed to evolve an autonomous learning robot by selectively integrating the necessary plasticity directly into the mechanisms responsible for the robot's behaviour.

3 Evolving CTRNNs as controllers for autonomous learning robots

Drawing inspiration from the Yamauchi and Beer (1994)'s experiment, we set out to evolve an *integrated* dynamic neural network, with fixed synaptic weights and 'leaky integrator' neurons, as a control system for a robot engaged in a task where learning behaviour is required.

3.1 Description of the task

The task requires navigation within a rectangular arena in order to find a goal represented by a black stripe on the white arena's floor. The learning aspect requires the robots "to learn", over a series of successive trials, the relationship between the position of the goal with respect to a landmark that the robot must use for guidance to find the goal.

At the start of each trial, a robot is positioned in the left or right side of an empty arena (see the black points

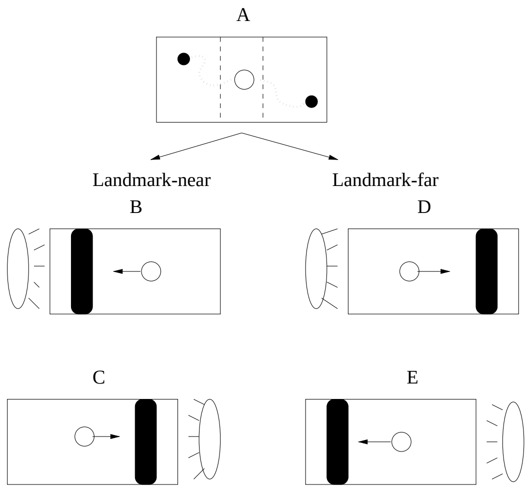

Figure 2: Depiction of the task. The small circle represents the robot. The white oval represents the landmark (i.e. a light source) and the black stripe in the arena is the goal. Picture A at the top represents the arena at the beginning of each single trial without landmark and goal. The black points represent two possible starting points. The curved dotted lines represent two possible routes to the central part of the arena which is delimited by dashed lines. Pictures B and C represent the two possible arrangement of the landmark and goal within the landmark-near environment. The pictures D and E represent the two possible arrangement of the landmark and goal within the landmark-far environment. The arrows, in pictures B,C,D,E represent the directions towards which a successful robot should moves.

in figure 2A). As soon as the robot reaches the central part of the arena (delimited in figure 2A by dashed lines), a goal (represented by a black stripe) is randomly positioned on the white arena's floor. This can be either in front of, or behind the robot, and is not perceivable by its proximity sensors (see figure 2B, 2C, 2D, 2E). The robot's behaviour will be evaluated according to a fitness function that rewards or penalises certain behaviours; this evaluation will be used to determine how many offspring it has within the evolutionary algorithm, but is not in any sense a reinforcement directly available to the robot. The robot is rewarded for leaving its current central position towards the goal until it finds it (see arrows in figure 2B, 2C, 2D, 2E); but it is penalized if it sets off away from the goal. The robot must "decide" which way to go; however, without any clue as to where the goal is, it can do no better than making a random decision.

The robot can use a landmark for guidance. The landmark is a light which lights up as soon as the goal is positioned, and it can be perceived by the robot from anywhere within the arena. The relationship between landmark and goal can vary; in some trials the landmark is close to the goal ('landmark-near environment' figure 2B and 2C), in other trials it is on the opposite side ('landmark-far environment' figure 2D and 2E). Both possibilities can be exploited by the robot to find the right way to the goal. If a robot approaches the landmark within the landmark-near environment, or moves away from the landmark within the landmark-far environment, it will consequently find the goal significantly more often than would be expected in a random walk. However, the current relationship between the landmark and the goal remains unknown to the robot until it has gathered enough experience to be able to disambiguate the environment between landmark-near or landmark-far.

Therefore, the task requires a robot to learn, over a series of successive trials, which of these two landmark-goal relationships currently holds true. The robot can disambiguate the environment between landmark-near or landmark-far by predicating its behaviour on its previous experience of the relationship between the location of the light and location of the goal. However, the simplicity of our scenario means that the light is only significant for the robot in this particular context. That is, the only adaptive reason for the robot to "pay attention" to the light is because the light is a cue to be exploited as part of the learning process. The robot has no reason to "pay attention" to the light in any other context. For a non-learning robot, the light presents no useful information because, on any given trial, there is no predictable relationship between the light and the goal. This presents a slight problem given that we wish to evolve the control mechanisms necessary for the robot to learn from the relative positioning of the light. It seems reasonable to suggest that unless the robot has *already* evolved to "pay some attention" to the light, it will be very difficult for it to evolve to learn from the relative relationship of the light position and the goal position. This problem can be addressed by given the light some significance other than as a learning cue. One way of doing this is to bias the proportion of each type of environment that the robots experience. For example, if robots are presented with landmark-near environment more often than the landmark-far environment, then the goal will be close to the light more often than it is far from it. In this way, the light can serve as a cue which can be exploited by even a purely reactive robot, since a robot which moves toward the light will be correct more often than it is wrong. Fully successful behaviour will nevertheless still require that the robot can learn to distinguish each type of environment and act appropriately. Rather than bias the proportion of times each environment is presented, we kept the proportion equal, but biased the relative weighting of the scores achieved in each environment, (as detailed in section 3.2) in order to achieve the same effect.

The robot undergoes two test sessions in the landmark-near and two more in the landmark-far environment. Each time the environment changes, the robot's control system is reset, so that there is no internal state or "memory" remaining from the previous environment. Within each of these four test sessions, a trial corresponds to the lapse of time that is given to the robot for reaching the middle of the arena and then finding the goal. The robot has 18 seconds to reach the middle of the arena. Then the landmark and the goal are positioned, and the robot has another 18 seconds to find the goal. Each trial can be terminated earlier either because the first time limit to reach the middle of the arena is exceeded; or because the agent reaches and remains on the goal for 10.0 seconds; or because the robot crashes into the arena wall. At the beginning of the first trial, and for every trial following an unsuccessful one the robot is positioned on either the left or the right part of an rectangular arena close to the short side. For every trial following a successful one, the robot normally keeps the position and orientation with which it ended the previous trial, but there is a small probability that it is replaced randomly. The simulation is deliberately noisy, with noise added to sensors and motors (see section 3.3); this is also extended to the environment dimensions. Every time the robot is replaced, the width and length of the arena, and the width of the central area that triggers the appearance of the landmark and the goal, are redefined randomly within certain limits. The robot undergoes 15 trials for each test session. During this time the relationship between landmark and the goal is kept fixed; the relationship remains unknown to the robot unless and until it can 'discover' it through experience. The position of the goal within the arena (i.e. left or right), is randomly determined every single trial and the landmark is subsequently appropriately positioned (depending on whether it is currently landmark-near or landmark-far). Over a full set of trials, all the 4 landmark/goal combinations (figure 2B, 2C, 2D, 2E) occur with equal likelihood.

3.2 Evaluation function

The robot is rewarded by the evaluation function F_{st} (per test session s, per trial t) for reaching the central area within the arena and then moving towards the goal until it find it (see arrows in figure 2B, 2C, 2D, 2E); but it is penalized either if it fails to reach the central area or if, once on the central area, it set off away from the goal.

$$F_{st} = (abcd)((3.0\frac{(d_f - d_n)}{d_f}) + \frac{p}{p_{max}})$$

with:

$s = [1, ..., 4]$, and $t = [2, ..., 15]$. The robot doesn't get any score during the first trial of each test session, because it is assumed that the robot doesn't know what kind of environment it is situated (i.e. landmark-near or landmark-far environment).

d_f represents the furthest distance that the robot reaches from the goal after the light is on. At the time when the light goes on, d_f is fixed as the distance between the centre of the robot body and the nearest point of the goal. After this, d_f is updated every time step if the new d_f is bigger than the previous one.

d_n represents the nearest distance that the robot reaches from the goal after the light is on. At the time when the light goes on, d_n is fixed as equal to d_f, and it is subsequently updated every time step both when the robot gets closer to the goal and when the robot goes away from the goal. d_n is also updated every time d_f is updated. In this case d_n is set up equal to the new d_f.

p represents the number of steps that the robot makes into the goal during its longest period of permanence. $p_{max} = 50$ is the maximum number of steps that the robot is allowed to make into the goal before the trial is ended;

a is a bias term for the type of environment; it is set to 3 in landmark-near environment, and to 1 in landmark-far environment.

b, c, d are the penalties. b is set to 0 if the robot fails to reach the central part of the arena, either because time limit is exceeded, or because it crashes into the arena wall; c is set to $\frac{1}{3}$ if the robot leaves the central area towards the opposite side of the arena respect to the goal (i.e unsuccessful behaviour); d is set to $\frac{1}{5}$ if the robot crashes into the arena wall; If the robot doesn't fall in any of these penalties, b, c, d are set to 1.

The total fitness of each robot is given by averaging the robot performance assessed in each single trail of each test session.

3.3 Simulated robot

A simple 2-dimensional model of a Khepera's robot-environment interactions within an arena was responsible for generating the base set aspects of the simulation (see Jakobi, 1997, for a definition of base set). The implementation of our simulator, both for the function that updates the position of the robot within the environment and for the function that calculates the values returned by the infra-red sensors and ambient light sensors, closely matches the way in which Jakobi designed his minimal simulation for a Khepera robot within an infinite corridor (see Jakobi, 1997, for a detailed description of the simulator). During the simulation, robot sensor values are extrapolated from a look-up table. Noise is applied to each sensor reading. Our robot sensor ability includes all its infra red sensors (Ir_0 *to* Ir_7 in figure 3), three ambient light sensors, positioned 45 degrees left and right and 180 degrees with respect to its face direction (A_1, A_4, A_6 in figure 3). The robot has a left and right motor, which can be independently driven forward or in reverse, allowing it to turn fully in any direction.

The light is modelled as a bar that illuminates the

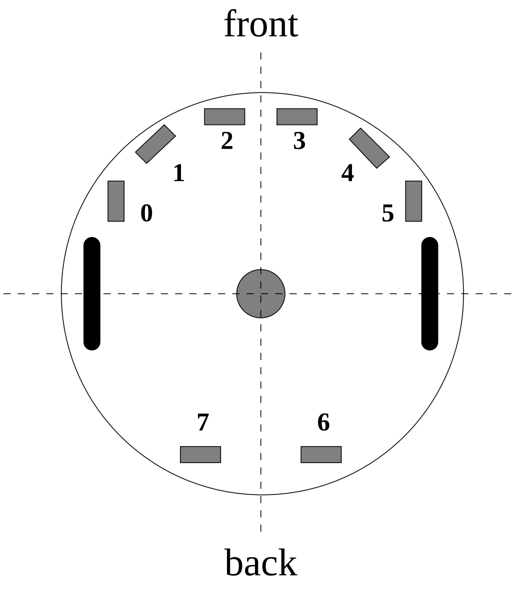

Figure 3: Plan of a Khepera mini-robot showing sensors and wheels. The robot is equipped with 6 infra red sensors (Ir_0 *to* Ir_5) and 3 ambient light sensors (A_1, A_4, A_6). It also has a floor sensor indicated by the central gray circle (F).

whole arena with the same luminosity. Each ambient light sensor faces out from a point on the exterior of the robot and detects any light within plus or minus ± 30 degrees from the normal to the boundary. The values returned by the ambient light sensors when impinged by the light, are set up to 1 if they exceed a fixed threshold otherwise they return 0. Our robot has an extra floor sensor, positioned on the belly of the robot, that can be functionally simulated as an ambient light sensor, that returns 0 when the robot is positioned over a white floor and 1 for a black floor. The simulation was updated the equivalent of 5 times a second.

3.4 The Network

Fully connected, 13 neuron CTRNNs are used. All neurons are governed by the following state equation:

$$\frac{dy_i}{dt} = \frac{1}{\tau_i}(-y_i + \sum_{j=1}^{k} \omega_{ji} z_j + g I_i)$$

$$with\ z_j = \frac{1}{1+exp[(y_j+\beta_j)]},\ i = 1,.,13$$

where, using terms derived from an analogy with real neurons, y_i represents the cell potential, τ_i the decay constant, β_j the bias term, z_j the firing rate, ω_{ji} is the strength of synaptic connections from neuron j^{th} to neuron i^{th}, k corresponding to the number of input connections to neuron i both from other neurons and from itself, I_i the intensity of the sensory perturbation on sensory neuron i. 11 network's neurons receive input (I_i) from the robot sensors. These input neurons receive a real value (in the range [0.0 : 1.0]), which is a simple linear scaling of the reading taken from its associated sensor (i.e. neuron N_1 from infra red sensor Ir_0, N_2 from Ir_1, N_3 from Ir_2, N_4 from Ir_3, N_5 from Ir_4, N_6 from Ir_5, N_7 from $\frac{Ir_6+Ir_7}{2}$, N_8 from ambient light A_1, N_9 from A_4, N_{10} from A_6, N_{11} from floor sensor F). The 12^{th} and the 13^{th} neuron don't receive any input from the robot's sensors. Their cell potential y_i, mapped into [0.0 : 1.0] by a sigmoid function, and then linearly scaled into [-10.0 : 10.0], set the robot motors output. The strength of synaptic connections ω_{ji}, the decay constant τ_i, the bias term β_j, and the gain factor g are genetically encoded parameters. States are initialized to 0 and circuits are integrated using the forward Euler method with an integration step-size of 0.2. States are set to 0 every time the network is reset.

3.5 The Genetic Algorithm

A simple generational genetic algorithm (GA) was employed (Goldberg, 1989). Populations contained 100 genotypes. Each genotype is a vector of real numbers (length 196, given by 13 neurons, 169 connections, 13 decay constants, 13 bias terms, 1 gain factor). Initially, a random population of vectors is generated by initializing each component of every genotype to random values uniformly distributed over the range [0, 1]. Subsequent generations were produced by a combination of selection with elitism, recombination and mutation. In each new generation, the two highest scoring individuals ("the elite") from the previous generation were retained unchanged. The remainder of the new population was generated by fitness-proportional selection from the 70 best individuals of the old population. 100% of new genotypes, except "the elite", were produced by recombination. Mutation entails that a random Gaussian offset is applied to each real-valued vector component encoded in the genotype, with a probability of 0.1. The mean of the Gaussian is 0, and its s.d is 0.1. During evolution, vector component values were constrained within the range [0, 1].

Search parameters were linearly mapped into CTRNN parameters with the following ranges:

- biases $\beta_j \in [-2, 2]$;
- connection weights $\omega_{ji} \in [-4, 4]$;
- gain factor $g \in [1, 7]$.

Decay constants were firstly linearly coded in the range $\tau_i \in [-0.7, 1.3]$ and then exponentially mapped into $\tau_i \in [10^{-0.7}, 10^{1.3}]$

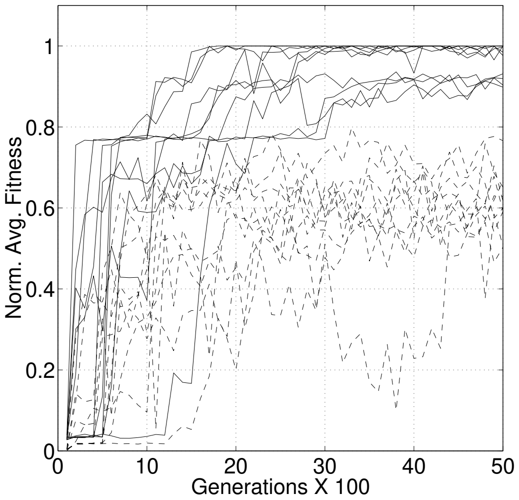

Figure 4: The graph shows the normalized fitness of the best robot (continuous line), and the normalized average fitness of the population (dotted line), for each generation for each run.

i	**j**					
	1	2	3	4	5	6
1	3.1	4	-1.7	3.6	0.2	-3.7
2	2.8	3.0	-0.6	-2.1	-2.0	0.4
3	-2.1	-3.3	-0.7	-1.8	-4	-4
4	-0.7	2.4	2.7	-4	0.7	2.8
5	-3.0	3.1	1.0	1.4	-1.2	-3.7
6	-1.5	-4	-0.0	-1.6	0.8	4
7	2.0	-1.6	2.3	3.4	-3.4	-2.3
8	4	1.6	-2.7	2.2	-2.2	-0.6
9	2.8	-0.5	-1.0	0.9	-3.9	-4
10	-3.7	-4	-2.8	4	4	-4
11	-4	2.9	-2.8	0.1	2.9	1.9
12	-2.9	-2.6	0.1	-4	-4	1.0
13	-0.4	0.8	2.5	-0.1	4	1.5

Table 1: The table above gives connection weights ω_{ij} from neuron N_i, (with $i = 1, ..., 13$) to neuron N_j (with $j = 1, ..., 6$) of the best network at generation 5000 of run n.1.

	i					
	1	2	3	4	5	6
τ_i	15.3	18.3	0.2	0.2	18.9	0.2
β_i	-0.9	-1.4	-1.4	-0.5	0.1	0.0

Table 2: Decay constants τ_i and the bias terms β_i, for neurons N_i, (with $i = 1, ..., 6$) of the best network at generation 5000 of run n.1.

4 Results

Ten evolutionary simulations, each using a different random seed, were run for 5000 generations (see figure 4). We examined the best individual of the final generation from each of these runs in order to establish whether they had evolved an appropriate learning strategy. Recall that to perform its task successfully the robot makes use of the light source in order to navigate towards its goal. Thus, over the course of a test session, the robot must learn—and come to exploit—the relationship between the light source and the goal that exists in the particular environment in which it is placed. In order to test their ability to do this, each of these final generation controllers was subjected to 500 test sessions in each of the two types of environment (i.e. landmark-near and landmark-far). As before each session comprised 15 consecutive trials, and controllers were reset before starting a new session. During these evaluations we recorded the number of times the controller successfully navigated to the target (i.e. found the target directly, without first going to the wrong end of the arena).

The results for the best controller of each of the ten runs can be seen in figure 5, which shows the percent of successful behaviour per trial during a session under each environmental condition. It is clear that in six of the runs (i.e. Run n.1, 3, 5, 7, 9 and 10) robots quickly

i	**j**						
	7	8	9	10	11	12	13
1	-0.9	0.3	2.0	-3.3	1.4	3.4	2.7
2	2.1	4	4	-4	-1.8	-1.6	-3.4
3	-0.6	-2.3	3.2	-0.1	-1.9	-4	-3.6
4	-3.1	-1.4	0.3	1.3	-2.4	2.0	-4
5	-3.8	1.6	-1.3	-0.7	-1.8	1.5	-2.8
6	1.8	-4	3.8	3.4	-0.6	-2.5	-3.5
7	3.1	-0.7	1.4	1.2	4	3.0	-0.9
8	-0.7	-3.9	-1.1	-3.2	0.2	2.4	-4
9	-4	-4	-0.5	-0.0	-1.8	1.0	-0.5
10	-0.5	4	2.9	0.0	1.4	3.1	-1.9
11	1.9	-3.1	-3.5	2.6	2.9	-2.4	-4
12	-2.9	-4	2.4	-1.5	-3.9	2.6	4
13	-3.6	3.3	-3.2	1.4	3.0	3.0	0.4

Table 3: The table above gives connection weights ω_{ij} from neuron N_i, (with $i = 1, ..., 13$) to neuron N_j (with $j = 7, ..., 13$) of the best network at generation 5000 of run n.1.

	i						
	7	8	9	10	11	12	13
τ_i	3.4	0.6	0.2	0.3	0.2	1.3	0.2
β_i	-0.9	-2	-0.0	-1.4	0.7	0.8	-1.9

Table 4: Decay constants τ_i and the bias terms β_i, for neurons N_i, (with $i = 7, ..., 13$) of the best network at generation 5000 of run n.1. The gain factor g is equal to 6.88.

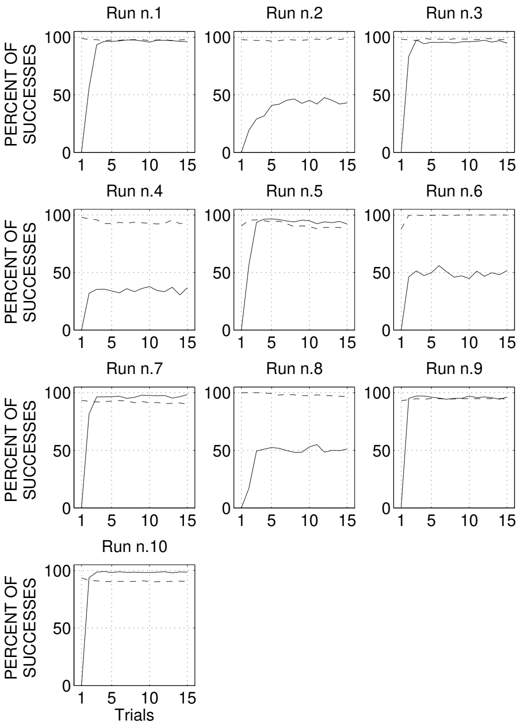

Figure 5: Each single graph refers to the performance of the best control system of the final generation of a single run, evaluated over 500 test sessions in both types of environment. Dashed lines indicate the percent of successful behaviour per trial in landmark-near environment. Continuous lines indicate the percent of successful behaviour per trial in landmark-far environment.

come to successfully use the light to navigate toward the goal, irrespective of whether they are in an environment which requires moving towards or away from the light source. All these controllers employ a default strategy of moving toward the light, and hence are always successful in the landmark-near environment (see figure 5 dashed lines). Consequently they are initially very unsuccessful in the landmark-far environment (see figure 5 continuous lines). Nevertheless, as can be seen from figure 5, when these controllers are placed in the landmark-far environment they are capable of adapting their behaviour over the course of very few trials and subsequently come to behave very differently with respect to the light. Each of these controllers, when placed in the landmark-far environment, initially navigates toward the light, fails to encounter the target and subsequently reaches the wrong end of the arena. At this point it turns around and proceeds back toward the correct end of the arena where it ultimately encounters the target stripe. The fact that in subsequent trials the robot moves away from the light rather than towards it, demonstrates that the robot has learnt from some aspect, or aspects, of its earlier experience of the current environment. More prosaically, these controllers learn from their mistakes.

It should be noted that the controllers from the other five runs (i.e. 2, 4, 6, and 8), whilst performing the task with varying degrees of success, all modify their behaviour over the course of the initial trails; in each case this results in improved performance. For example, the controller from run 6 initially has a 80% success rate in landmark-near environment and a 0% success rate in landmark-far environment. In the landmark-near environment its success rate climb from around 80% to 100%. In the landmark-far environment its success rate will climb from 0% to around 50% within the course a few trials. Similarly, with the controller from run 2, 4 and 8, success in a landmark-far environment increases rapidly from 0% up to around 50% whilst success in the landmark-near remains consistently around 100%.

As reference, we provide connection weights (see table 1 and table 3), bias terms, decay constants and gain factor (see table 2 and table 4), of one of the best evolved network, i.e. best network at generation 5000 of run n.1. It should be notice that few weights are equal to zero (i.e. 3 out of 169). This suggests that all the aspects of the robot behaviour are underpinned by mechanisms which seem to relay on the highly distributed network of causal relationships among the neurons, rather than on modularized structures of smaller number of neurons, functionally dedicated to a particular sub-tasks.

5 Discussion

Clearly there are strong similarities between Yamauchi and Beer (1994) and our experiment. We employed the same network style to control the behaviour of the robot, and we tested our robot in similar environmental conditions to the one described in (Yamauchi and Beer, 1994). However there are also several differences which might have singularly contributed to our successful results.

First of all, contrary to (Yamauchi and Beer, 1994), we biased the relative weighting of the score achieved in each environment in order to give the landmark some significance other than as a learning cue. The effects of this feature of our fitness function, can be clearly singled out both in figure 4 and in figure 5. From an evolutionary perspective, almost all the best robots in each run get to a stage in which they employ photo-taxis (see the plateau in continuous lines around fitness 0.8 in figure 4). This means that all the best robots make use of the light to navigate within the arena. Although the evolution of the mechanisms for photo-taxis contributes only three quarters of the total score, they might have significantly helped and guided the evolution towards the subsequent appearance of the learning controllers. The evolved learning robots consistently look for the light, and only when the conditions which reinforce the photo-taxis run out (i.e in the 'landmark-far environment') the robots change their behaviour to anti-photo-taxis (see figure 5).

Secondly, we didn't terminate the trial every time a robot finished up in the wrong end of the arena, far away from the goal, as it was for Yamauchi and Beer (1994)' agents. The robots, although slightly penalized for this, are allowed to recover from their mistakes. This produces a richer environment in which having more "time" and "freedom" to explore might help to evolve a proper reinforcement signal which makes the robot employ the right behaviour for each environment (i.e. landmark-near or landmark-far). However, the complexity and richness of our set up makes it harder to identify the condition of reinforcement as exploited by each learning robot. This is clearly an aspect of the evolved learning behaviour which needs to be clarified by further investigation.

Thirdly, our fitness function has not been designed in a discrete fashion either to strongly reward successful behaviour or to strongly penalize unsuccessful behaviour. In (Yamauchi and Beer, 1994) the agents get 1 for reaching the goal and 0 if they don't reach the goal. Instead of this 'black-and-white' approach, we arranged for 'shades of grey' by gradually increasing the reward starting from completely unsuccessful behaviours (e.g robots that don't manage to make the landmark and the goal appear on the arena) to very successful ones (e.g robots that reach the central part of the arena and then head successfully to the goal).

These are the three main differences between Yamauchi and Beer (1994) and our experiment which might have contributed to our successful results. However to assess the contribution that each of them brought to the achievement of our results, further tests and analysis need to be carried out.

6 Conclusion

In 1994 Yamauchi and Beer wanted to determine whether dynamic neural networks could provide an effective control mechanism for integrating reactive, sequential and learned behaviour in autonomous systems, without having to assume a sharp division between the components that are reactive and those that are in charge of sequencing or learning (Yamauchi and Beer, 1994). We gave a brief description of their experiment, which due to the characteristics of the control system and the singularity of the learning task employed, still is a distinctive an unique example in the area of autonomous learning robots (see section 2).

Their approach clearly represents one of the first attempts in which CTRNNs, with fixed connection weights and leaky-integrator neurons, have been exploited to integrate reactive, sequential and learned behaviour in a simulated robot. However, their attempts to evolve *integrated* CTRNNs, as controllers for simulated agents capable to perform learning behaviour, failed. They ended up recommending independent evolution of separate controller modules, some of which were dedicated to reactive and sequencing behaviour and others dedicated to learning. Moreover, they hard-wired into the model a reinforcement signal intended to be a feedback signal for the learning module.

The model we described in this paper demonstrated that it is possible to evolve *integrated* CTRNNs that successfully controls the behaviour of a simulated khepera mini-robot engaged in a learning task. We showed that dynamic neural network, if shaped by evolution, can integrate reactive and learned behaviour with a single control system that does not rely on independent evolution of separate controller modules, arbitrarily partitioned by the researcher, and does not rely on a hard-wired reinforcement signal. The evolved *integrated* control systems benefits from its own evolved reinforcement signals to generate the appropriated behavioural responses.

We hope that the evolutionary approach undertaken in our model might represent both a further methodological tool to test specific hypothesis about the selection pressures involved in the evolution of specific learning skills possessed by particular species, and a possible solution for engineering plastic control systems for learning robots.

Acknowledgments

The authors like to thank all the members of the Centre for Computational Neuroscience and Robotics (http://www.cogs.sussex.ac.uk/ccnr/) for constructive discussion and the Sussex High Performance Computing Initiative (http://www.hpc.sussex.ac.uk/) for computing support. ET likes to thank Elisa Niccolai, Andrea Vaccaro.

References

Davey, G. (1989). *Ecological learning theory.* Routledge, London, England UK.

DiPaolo, E. (2000). Homeostatic adaptation to inversion of the visual field and other sensorimotor disruptions. In *From Animals to Animats VI: Proceedings of the* 6^{th} *Interntional Conference on Simulation of Adaptive Behavior.*

Eggenberger, P., Ishiguro, A., Tokura, S., Kondo, T., and Uchikawa, Y. (1999). Toward seamless transfer from simulated to real worlds: A dynamically-rearranging neural network approach. In Watt, J. and Demiris, J., (Eds.), *Advances in Robot Learning, 8th European Workshop on Learning Robots, EWLR-8, Lausanne, Switzerland, September 18, 1999, Proceedings*, volume 1812 of *Lecture Notes in Computer Science.* Springer.

Floreano, D. and Mondada, F. (1996). Evolution of plastic neurocontrollers for situated agents. In Maes, P., Mataric, M., Meyer, J.-A., Pollack, J., and Wilson., S., (Eds.), *From Animals to Animats IV: Proceedings of the 4^{th} International Conference on Simulation of Adaptive Behavior*. MA: MIT Press.

Floreano, D. and Urzelai, J. (2000). Evolutionary Robots: The Next Generation. In *The 7^{th} International Symposium on Evolutionary Robotics (ER2000): From Intelligent Robots to Artificial Life*.

Floreano, D. and Urzelai, J. (2001a). Evolution of plastic control networks. *Autonomous Robots*. in press.

Floreano, D. and Urzelai, J. (2001b). Neural Morphogenesis, Synaptic Plasticity, and Evolution. *Theory in Biosciences*. in press.

Garcia, J. and Koelling, R. A. (1966). Relation of cue to consequence in avoidance learning. *Psychonimic Science*, 4:123–124.

Goldberg, D. E. (1989). *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison-Wesley.

Harvey, I., Husband, P., Thompson, A., and Jakobi, N. (1997). Evolutionary Robotics: the Sussex approach. *Robotics and Autonomous Systems*, 20:205–224.

Harvey, I., Husbands, P., and Cliff, D. (1992). Issues in evolutionary robotics. In Meyer, J.-A., Roitblat, H., and Wilson, S., (Eds.), *From Animals to Animats II: Proceedings of the 2^{nd} International Conference on Simulation of Adaptive Behavior*, pages 364–373, Cambridge MA. MIT Press/Bradford Books.

Jakobi, N. (1997). Evolutionary robotics and the radical envelope of noise hypothesis. *Adaptive Behavior*, 6:325–368.

Kaelbling, L. P. (1993). *Learning in Embedded Systems*. MIT Press.

Meyer, J.-A., Husbands, P., and Harvey, I. (1998). Evolutionary Robotics: A survey of Applications and Problems. In Husbands, P. and Meyer, J.-A., (Eds.), *Evolutionary Robotics: Proceedings of the 1^{st} European Workshop, EvoRobot98*. Springer.

Nolfi, S. (1998). Evolutionary robotics: Exploring the full power of self-organization. *Connection Science*, 10:167–184.

Nolfi, S. and Floreano, D. (1999). Learning and evolution. *Autunomous Robots*, 7(1):89–113.

Nolfi, S. and Floreano, D. (2000). *Evolutionary Robotics: The Biology, Intelligence, and Technology of Self-Organizing Machines*. MA: MIT Press/Bradford Books.

Nolfi, S. and Parisi, D. (1997). Learning to adapt to changing environments in evolving neural networks. *Adaptive Behavior*, 5(1):75–98.

Webb, B. (2000). What does robotics offer animal behaviour? *Animal Behaviour*, 60:545–558.

Wilcoxon, H. C., Dragoin, W. B., and Kral, P. A. (1971). Illness-induced aversions in rat and quail: relative salience of visual and gustatory cues. *Science*, 171:826–828.

Yamauchi, B. M. and Beer, R. D. (1994). Integrating Reactive, Sequential, and Learning Behavior Using Dynamical Neural Network. In Cliff, D., Husbands, P., Meyer, J.-A., and Wilson, S. W., (Eds.), *From Animals to Animats III: Proceedings of the 3^{rd} International Conference on Simulation of Adaptive Behavior*.