Neuronal Plasticity and Temporal Adaptivity: GasNet Robot Control Networks

Tom Smith[1,3], Phil Husbands[1,2], Andy Philippides[1,3], Michael O'Shea[1,3]
[1]*Centre for Computational Neuroscience and Robotics (CCNR), University of Sussex*
[2]*School of Cognitive and Computing Sciences (COGS), University of Sussex*
[3]*School of Biological Sciences (BIOLS), University of Sussex*

Designing controllers for autonomous robots is not an exact science, and there are few guiding principles on what properties of control systems are useful for what kinds of task. In this article we analyze the functional operation of robot controllers developed using evolutionary computation methods, to elucidate the strengths and weaknesses of the underlying control system class. By comparing and contrasting robot controllers based on two different classes of artificial neural network, the *GasNet* and *NoGas* networks, we show that the increased evolvability of the GasNet class on a visual shape discrimination task is due to the temporally adaptive nature of the GasNet, where neuronal plasticity mediated through the concentration of virtual neuromodulatory "gases" occurs over a wide range of time courses. We argue that the availability of mechanisms operating over a wide range of potential time courses is a crucial property for controllers used to generate adaptive behavior over time, and that the design process should easily be able to adapt those time courses to the natural time scales in the environment.

Keywords evolutionary robotics · artificial neural networks · GasNets · neuromodulation · neuronal plasticity

A good performance, like a human life, is a temporal affair: a process in time.
– Mortimer J. Adler

1 Introduction

If we are to see evolutionary computation and other artificial evolution methodologies applied regularly in real-world robotics problems, it is crucial that we understand the strengths and weaknesses of the underlying control classes used. In a number of recent articles we have investigated the search spaces underlying two neural network control classes used in evolutionary robotics experiments, in an attempt to relate the properties of the fitness landscape to the ease of finding successful controllers (Smith, Husbands, Layzell, & O'Shea, 2002; Smith, Husbands, & O'Shea, 2001a, b, in press). In this article we use functional analysis of evolved robot control solutions to relate properties of the two underlying network classes to the ease of finding good solutions. Such understanding can give us an insight into properties of control classes that may be useful in a wider range of problems than simply the task at hand.

Correspondence to: T. Smith, CCNR, BIOLS, University of Sussex, Brighton BN1 9QH, UK. *E-mail*: toms@cogs.susx.ac.uk; *Tel.*: +44-1273-872952

We analyze robot controllers based on two different classes of artificial neural network, the *GasNet* and *NoGas* networks. We show that the increased evolvability[1] of the GasNet class on a shape discrimination task is due to the *temporally adaptive* nature of the GasNet, where neuronal plasticity mediated through the concentration of virtual neuromodulatory "gases" occurs over a wide range of modifiable time courses. We argue that the availability of processes operating over a wide range of potential time courses is a crucial property for controllers used to generate adaptive behavior over time, and that the ease with which agent controllers can be tuned to the particular temporal characteristics of the environment is a principal determinant of the suitability of the underlying solution class to the problem at hand. Finally, we propose that if we are to develop further evolvable artificial neural network classes for adaptive control, the starting point must be from within the class of temporally adaptive networks of which the GasNet is a member.

It is clear that allowing agents access to temporal information is necessary for a range of complex cognitive behaviors, not least because they can then exploit the temporal structure inherent in the interaction between agent and environment. For example, Gallagher & Beer (1999) argue that "nontrivial behavior requires the integration of experiences across time and the ability to initiate actions independent of an agent's immediate circumstances" (p. 1277). Here we argue that agents performing simpler tasks, such as the visual shape discrimination investigated here, can also benefit from such temporally adaptive control classes. In particular, we see that agents based on such control classes display a range of rich temporal dynamics such as pattern generation and active perception, and furthermore these complex dynamics are exploitable by artificial evolutionary processes. In other words, temporally adaptive control systems are more evolvable.

Section 2 describes the GasNet and NoGas robot control classes, the evolutionary computation algorithm used, the robot control task used, and rate of evolution results. The task is a visual shape discrimination experiment; starting from an arbitrary position and orientation in a black arena, robot controllers must navigate to a white triangle while ignoring a white square. Successful GasNet controllers consistently evolve faster than NoGas controllers, and a central theme of the article is that we can use analysis of evolved controllers to understand the reasons for this faster evolution.

Section 3 addresses the question of what might lead to differences in evolutionary search time, outlining a number of possibilities. In Section 4 we introduce the methods of dynamical systems analysis, illustrating the techniques through analysis of the operation of a GasNet controller pattern generation subnetwork. We then go on to use the dynamical systems analysis to identify possible reasons for this increased evolutionary rate, with Section 5 using the analysis of a single GasNet robot controller to frame a number of hypotheses for the suitability of the GasNet class to robot control. In particular, we show how the properties of gas diffusion can be used to filter out sensor input noise, produce simple pattern generation networks, and switch networks from one stable state to another. We hypothesize that these properties lead to GasNet solution spaces in which it is easier to find good controllers than in the corresponding NoGas solution spaces.

In Section 6 we go on to compare the operation of two controllers, one GasNet solution and one NoGas solution, which utilize the same visual shape discrimination strategy. We argue that the GasNet controller is easier to tune to the particular characteristics of the environment than the functionally equivalent NoGas controller, and in Section 7 we find evidence to support such an argument through re-evolution of the functionally equivalent controllers in environments with modified characteristics. We then extend the re-evolution analysis to a larger sample of previously evolved GasNet and NoGas controllers, showing that GasNet controllers are faster to re-evolve in modified environments, backing up the hypothesis that GasNet controllers are easier to tune to the particular characteristics of the environment. The article closes with summary and discussion.

2 GasNet and NoGas Robot Control Networks

The *GasNet* class of artificial neural networks (ANNs) incorporates an abstract model of a gaseous diffusing neuromodulator into a more standard ANN (Husbands, 1998; Husbands, Smith, Jakobi, & O'Shea, 1998). In previous work the networks have been used in a variety of evolutionary robotics tasks, comparing the rates of evolution for networks with and without (the *NoGas*) the gas signaling mechanism active. In a variety of robotics tasks, GasNet controllers evolve significantly faster than NoGas controllers (see, for example, Husbands, 1998; Husbands et al., 1998). Initial work aimed at identifying the reasons for this faster search has focused on the search spaces underlying the GasNet control class, investigating the ruggedness and modality of the spaces (Smith et al., 2001b), nonadaptive phases of evolution (Smith et al., 2001a), and the local landscape evolvability surrounding solutions (Smith et al., in press). In this article we analyze successfully evolved controllers to highlight the properties of GasNets leading to faster evolutionary search.

2.1 The GasNet Architecture

The GasNet is an arbitrarily recurrent ANN augmented with a model of diffusing gaseous modulation, in which the instantaneous activation of a node is a function of both the inputs from connected nodes and the current concentration of gas(es) at the node. Thus in addition to the standard electrical activity "flowing" between nodes, an abstract process analogous to the diffusion of gaseous modulators such as nitric oxide is at work (Philippides, Husbands, & O'Shea, 2000). In this process, the virtual gases do not alter the electrical activity in the network directly but rather act by changing the gain of transfer function mapping between node input and output in a concentration-dependent manner.

The network underlying the GasNet model is a discrete time step, recurrent neural network with a variable number of sigmoid transfer function nodes. These nodes are connected by either excitatory (with a weight of +1) or inhibitory (with a weight of –1) links with the output O_i^t, of node i at time step t determined by a continuous mapping from the sum of its inputs, as described by the following equation:

$$O_i^t = \tanh\left[K_i^t\left(\sum_{j \in C_i} w_{ji} O_j^{t-1} + I_i^t\right) + b_i\right] \qquad (1)$$

where C_i is the set of nodes with connections to node i with connection weights w_{ji}, O_j^{t-1} the output of node j on the previous time step, I_i^t the external (sensory) input to node i at time t, and b_i a genetically set node bias (ranging from –1 to +1). Each node has a genetically set default transfer function parameter K_i^0 (see Section 2.3), and for the NoGas class this transfer parameter is fixed over the operation of the network: $K_i^t = K_i^0 \forall t$.

2.2 Gas Diffusion in the Networks

To incorporate the gas concentration model, the network is placed in a two-dimensional plane, with node $\{x, y\}$ positions specified genetically. The GasNet diffusion model is controlled by two genetically specified parameters, namely the radius of influence r around the emitting node (ranging from 10% to 50% of the two dimensional plane dimensions), and the rate of build up and decay s (ranging from 1 to 11 time steps). Spatially, the gas concentration varies as an inverse Gaussian of the distance from the emitting node with a spread governed by r, and the concentration set to zero for all distances greater than r (Equation 2). This is loosely analogous to the length constant of the natural diffusion of nitric oxide, related to its rate of decay through chemical interaction (Philippides et al., 2000). The maximum concentration at the emitting node is one, and the concentration builds up and decays linearly with time at a rate determined by s, shown in Equations 3 and 4. For an emitting node, the concentration of gas $C(d, t)$ at distance d from the node and time t is given by Equations 2 to 4:

$$C(d, t) = \begin{cases} C_0 \times e^{-(d/r)^2} \times T(t) & d < r \\ 0 & \text{else} \end{cases} \qquad (2)$$

$$T(t) = \begin{cases} H\left(\frac{t - t_e}{s}\right) & \text{emitting} \\ H\left(H\left(\frac{t_s - t_e}{s}\right) - H\left(\frac{t - t_s}{s}\right)\right) & \text{not emitting} \end{cases} \qquad (3)$$

$$H(x) = \begin{cases} 0 & x \le 0 \\ x & 0 < x < 1 \\ 1 & \text{else} \end{cases} \qquad (4)$$

where $C(d, t)$ is the concentration at a distance d from the emitting node at time t, t_e is the time at which emission was last turned on, t_s is the time at which emission was last turned off, and s (controlling the slope of the function T) is genetically determined for each node. To summarize, within a radius of r from the node, gas builds up (and decays) linearly to a maximum of

$C_0 e^{-(d/r)^2}$ in s time steps. The total concentration at a node is then determined by summing the concentrations from all other emitting nodes (nodes are not affected by their own concentration, to avoid runaway positive feedback).

2.3 Modulation by the Gases

There are two virtual gases in the network, gas 1 and gas 2, which increase and decrease K_i^t (see Equation 1) respectively in a concentration-dependent fashion. Both the type of gas emitted by a node and the conditions under which it emits are specified genetically. Nodes emit either (a) gas 1, (b) gas 2, or (c) no gas, and emission occurs when either (a) the node activity increases beyond the electrical threshold 0.5, or (b) the local concentration of gas 1 increases beyond the threshold 0.1, or (c) the local concentration of gas 2 increases beyond the threshold 0.1. The concentration-dependent modulation is described by Equations 5 to 8, with transfer parameters updated on every time step as the network runs. Thus we have:

$$K_i^t = \mathbf{P}[D_i^t] \tag{5}$$

$$\mathbf{P} = \{-4.0, -2.0, -1.0, -0.5, -0.25, -0.125, 0.0, 0.125, 0.25, 0.5, 1.0, 2.0, 4.0\} \tag{6}$$

$$D_i^t = f\left(D_i^0 + \frac{C_{i1}^t}{C_0 \times K}(N - D_i^0) - \frac{C_{i2}^t}{C_0 \times K} D_i^0\right) \tag{7}$$

$$f(x) = \begin{cases} 0 & x \le 0 \\ \lfloor x \rfloor & 0 < x < N \\ N - 1 & \text{else} \end{cases} \tag{8}$$

where $\mathbf{P}[j]$ refers to the jth element of set $\mathbf{P}$, D_i^t is the node i's pointer into the set $\mathbf{P}$ of possible discrete values that K_i^t can assume, N is the number of elements in $\mathbf{P}$ (13 are shown in Equation 6), D_i^0 is the genetically set default value for D_i^t (ranging from 1 to 13) , C_{i1}^t is the concentration of gas 1 at node i on time step t, C_{i2}^t is the concentration of gas 2 at node i on time step t, and C_0 and K are global constants (both set to 1 in the experiments reported in this article).

Thus, the concentration of each gas is directly proportional to any change in D_i^t, with a corresponding change in K_i^t. Although the change in K_i^t is nonlinear these values represent a smooth change in the slope of the transfer function. Since the transfer functions can change throughout the lifetime of the network, this system provides a form of neuronal plasticity not seen in most other neural network classes.

2.4 Visual Shape Discrimination

The evolutionary task at hand is a visual shape discrimination task; starting from an arbitrary position and orientation in a black-walled arena, the robot must navigate under extremely variable lighting conditions to one shape (a white triangle) while ignoring the second shape (a white square). Fitness over a single trial was taken as the fraction of the starting distance moved toward the triangle by the end of the trial period, and the evaluated fitness was returned as the weighted average over N trials of the controller from different initial conditions:

$$F = \frac{2}{N(N+1)} \sum_{i=1}^{i=N} i\left(1 - \frac{D_i^F}{D_i^S}\right) \tag{9}$$

where D_i^F is the distance to the triangle at the end of the ith trial, and D_i^S the distance to the triangle at the start of the trial, and the i trials are sorted in descending order of $1 - \frac{D^F}{D^S}$. Thus good trials, in which the controller moves some way toward the triangle, receive a smaller weighting than bad trials, encouraging robust behavior on all trials. In practice we use 16 trials, changing the relative positions of the triangle and square, and the starting orientation and position of the robot, on each trial.

Success in the task was taken when an evaluated fitness of 1.0 was obtained over 30 successive generations of the evolutionary algorithm. In the work reported here, fitness evaluations are carried out in a verified *minimal simulation* (Jakobi, 1998); see Figure 1 for a screen-shot of a fitness evaluation in simulation. Evolved controllers have been successfully transferred to the real robot (Husbands, 1998). As in many problems requiring controllers to provide sensor-to-motor mappings over time, fitnesses are extremely time consuming to evaluate (in the work presented here, evaluating a sample of 10^6 fitnesses takes around 24 hours on a Pentium II 700 MHz machine) and inherently very noisy.

Figure 2 shows the distribution of fitnesses from a single controller over 10,000 evaluations. It should be emphasized that the environmental noise for the robot controllers is not simply variation in the received fit-

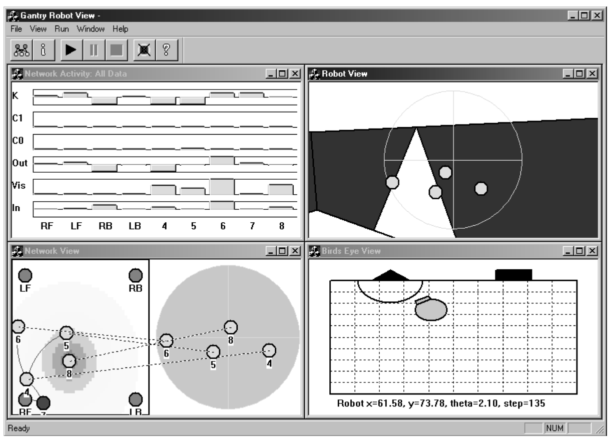


Figure 1 Screen shot of the simulated arena and robot. The bottom-right view shows the robot position in the arena with the triangle and square. Fitness is evaluated on how close the robot approaches the triangle. The top-right view shows what the robot "sees," along with the pixel positions selected by evolution for visual input. The top-left view shows the instantaneous activity of all nodes in the neural network. The bottom-left view shows the robot control neural network.

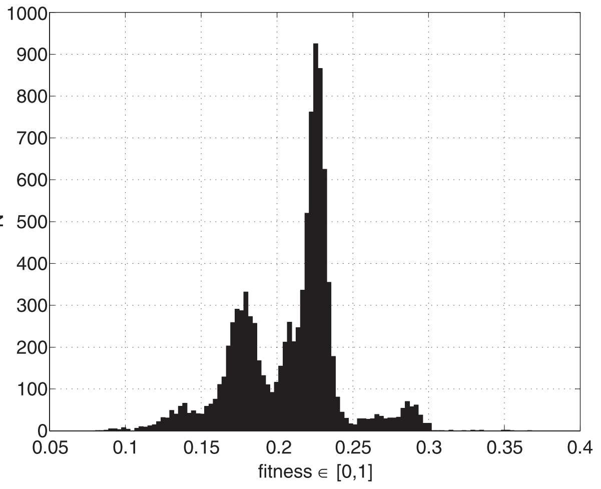


Figure 2 The fitness distribution of a single genotype evaluated 10,000 times in the minimal simulation evaluation environment, where 95% of the fitnesses lie in the range [0.1343, 0.2856], with possible controller fitness ∈ [0, 1].

ness score but is a crucial feature of the robot minimal simulation model. Controllers must evolve to be robust to such noise, so as to transfer successfully to the real world; two controllers may be of equal fitness when evaluated in a noiseless environment but may be of very different fitnesses in the full noise model. The level of noise in the model is higher than that found in the real world, to evolve robot controllers successfully in simulation able to operate successfully in the real world (Jakobi, 1998).

2.5 The Solution Representation

The neural network robot controllers were encoded as variable length strings of integers, with each integer allowed to lie in the range [0, 99]. Each node in the network was coded for by 19 parameters, controlling such properties as node connections, sensor input, node bias, and all the variables controlling gas diffusion as described in Sections 2.2 and 2.3. Both the robot control network, an arbitrarily recurrent ANN, and the robot sensor input morphology, that is, the position of the input pixels on the visual array, were also under evolutionary control. Thus mutation of parent solutions (Section 2.6) is able to produce offspring solutions with varying network architecture, network node properties, and sensor morphology. In all experiments, the evolutionary population was initially seeded with randomly generated genotypes coding for networks of 10 neurons. For further details see Husbands et al. (1998) and Smith (2002).

2.6 The Evolutionary Algorithm and Mutation Operator

A distributed asynchronously updating evolutionary algorithm was used, with a population of 100 solutions arranged on a 10 ×10 grid. Fitness was awarded on the fraction of the distance moved toward the triangle over a series of 16 trials with different initial conditions (see Equation 9). Parents were selected with probability proportional to their fitness ranked in ascending order over the mating "pool" consisting of a randomly chosen grid point plus its eight nearest neighbors. The parent solution was mutated to create the offspring solution that was placed back in the mating pool, replacing a solution chosen with probability proportional to their fitness ranked in descending order over the mating pool. One generation was specified as 100 such breeding events. Figure 3 shows the pseudocode for the evolutionary algorithm.

In each breeding event, three mutation operators were applied to solutions with mutation rate μ (for the experiments detailed here, $\mu = 0.04$). The *creep* operator was applied to each integer in the string, with mutation in a Gaussian distribution $N(0, 10)$ centered

```
– Initialise population of 100 solutions on 10×10 grid.
– Evaluate each solution fitness.
– Repeat until success criterion met, or MaxGenerations reached:
  – Repeat 100 times for 1 generation:
    – Select solution at random.
    – Create mating pool of solution plus 8 nearest grid neighbours.
    – Pick parent P through ascending rank-based selection on mating pool.
    – Create offspring 0 through mutation of P, and evaluate fitness.
    – Place 0 in solution grid, replacing mating pool solution picked
          through descending rank-based selection.
```

Figure 3 Pseudo-code for the asynchronously updating evolutionary algorithm.

on its current value (20% of these mutations completely randomized the integer). The *addition* operator was applied to the whole genotype, adding one neuron to the network, that is, increasing the genotype length by 19 parameters. Finally, the *deletion* operator deleted one randomly chosen neuron from the network, that is, decreasing the genotype length by 19. It should be noted that the mutation rate $\mu = 0.04$ used in these experiments is a much larger level of mutation than typically used in artificial evolution optimization (and certainly much larger than in biological evolution). However, lower levels of mutation lead to extremely slow evolution of successful solutions (Smith, 2002).

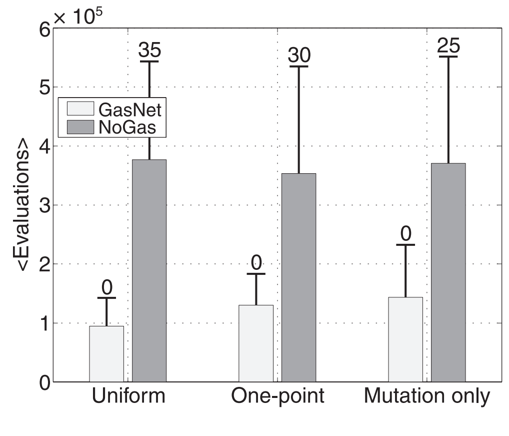
2.7 Rates of Evolution: Previous Results

The evolution of solutions based on the GasNet class consistently produces successful robot control solutions in significantly fewer evaluations required for the evolution of solutions based on the NoGas class. This result holds over a number of different evolutionary algorithms, with a number of different mutation and recombination rates used, including fixed length genotypes, uniform and one-point crossover recombination, and mutation rates affecting from 1% to 62% of the genotype loci. Two different network connectivity schemes were also investigated, with both seen to show faster GasNet evolution. For full results, see Smith (2002). Figures 4 and 5 show example results. In the next section, we outline possible reasons for this evolutionary rate difference.

3 Why are GasNets More Evolvable?

What are the reasons for the rate of evolution differences seen between the GasNet and NoGas spaces, shown in Figures 4 and 5? We can frame many of the possible reasons as properties of the underlying fitness landscapes. There may simply be many more successful GasNet than NoGas solutions, simplifying the search problem. The underlying search spaces may differ in their ruggedness, local modality, degree of neutrality, or some other property, making it easier for search processes to find solutions of increasing fitness. More subtle effects may also be important, as properties of the fitness landscapes may not be homogenous across the space; for example, the GasNet space may contain smaller regions that are easier to search in some way.

In previous work, Smith et al. (2001b) have shown no evidence for increased numbers of GasNet solutions in the search space; massive random sampling



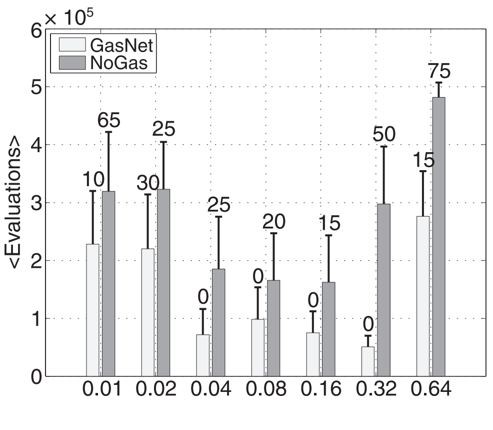


Figure 4 The mean number of evaluations required for (a) uniform, one-point, and no recombination, and (b) no recombination, varying mutation rate $\mu \in \{0.01, 0.02, 0.04, 0.08, 0.16, 0.32, 0.64\}$. Data averaged over 20 runs of the distributed evolutionary algorithm. The error bars represent 95% confidence limits for the mean; the number above the bar gives the percentage of runs failing to finish in 1,000,000 evaluations.

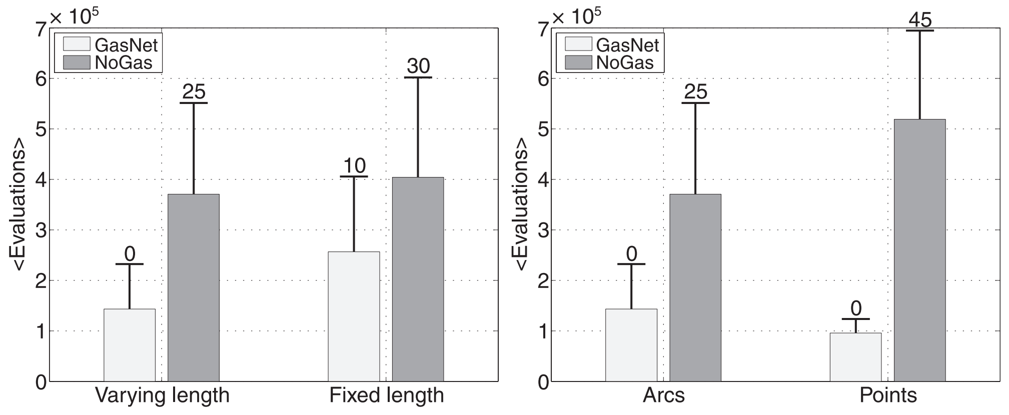


Figure 5 The mean number of evaluations required for (a) variable and fixed-length genotypes, and (b) arcs and points node connectivity schemes. Data averaged over 20 runs of the distributed evolutionary algorithm. The error bars represent 95% confidence limits for the mean; the number above the bar gives the percentage of runs failing to finish in 1,000,000 evaluations.

shows very few solutions of either class above 50% fitness, even though this fitness is relatively easy to find using evolutionary computation methods. It is entirely possible that the number of high-fitness GasNet solutions is significantly larger than the number of NoGas solutions, but this is extremely difficult to show without exhaustive sampling of the space, which is clearly impractical. Similarly, it was seen that the spaces do not measurably differ in ruggedness and modality at different fitness levels. Smith et al. (2002) develop the technique of *fitness evolvability portraits*, using the fitness distribution of the search space surrounding solutions to build up a description of the fitness landscape. However, applying such measures to the GasNet and NoGas search spaces shows no measurable differences in ruggedness, modality and neutrality between the landscapes underlying the two classes (Smith et al., in press).

The control classes are also of similar robustness to mutation; Figure 6 shows the fitness distribution of all one-point mutations from the sample of successfully evolved controllers used in Section 7. The robustness of the GasNet and NoGas controllers are extremely similar; in both cases roughly 80% of mutations are neutral, with 10% catastrophic (it should be noted that the mutation operators used during optimization typically mutate more than one loci value, so the observed degree of neutrality will be significantly smaller than this 80%).

However, the understanding of search difficulty in terms of the fitness landscape properties is not simple in such a complex search space. The differences between the spaces may be small enough to be obscured by variation; it may be the case that search processes only "recognize" these differences when iterated over large numbers of fitness evaluations. It is also possible that the distinctions between the spaces are not measurable by the techniques at hand; some other property of high-dimensional search spaces may be involved. In this article we develop a different approach, through analysis of successfully evolved controllers, allowing us to identify general properties of the GasNet and NoGas classes that may hold in a wider class of problems than just the visual shape discrimination task

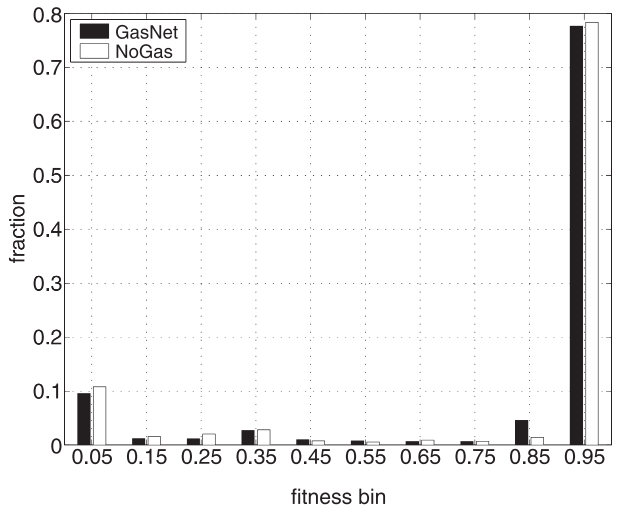


Figure 6 Robustness of GasNet and NoGas solutions; fitness of the one-point mutations from sample of successfully evolved controllers. Fitness evaluated in non-noisy simulation.

used here. In the next section we apply dynamical systems analysis to the operation of evolved robot control networks.

4 Dynamical Systems Analysis

The dynamical systems approach analyzes how a system behaves over time, in particular investigating the future behavior of the system given the current state of the system. A full description of dynamical systems theory and analysis is well beyond the scope of this article; there is an extremely large body of literature dealing with a variety of dynamical systems, including physical (see, for example, Goldstein, 1980), biological (see, for example, Rosen, 1970; Rubinow, 1975) and chemical systems (see, for example, Gavalas, 1968).

Previous work on evolving robot controllers has also used dynamical systems analysis to examine the behavior of evolved controllers, most especially in the work by Beer and coworkers (see, for example, Beer, 1990, 1995; Beer & Gallagher, 1992; Chiel, Beer, & Gallagher, 1999; Calvatti & Beer, 2001), typically to give insight into design principles from controllers evolved on different evolutionary runs. However, here we use the analysis to identify potential reasons for the faster evolutionary search seen in Section 2.7. In the next section we apply the basic methods to an example subnetwork from the GasNet controller analyzed more fully in Section 5.

4.1 A Discrete Dynamical Pattern Generation Network

In this section we consider a two-node pattern generation network, part of the evolved robot controller analyzed more fully in Section 5, and shown in Figure 7. In particular, we want to see how the properties of both the individual nodes and the gas diffusion mechanism lead to pattern generation.

In the robot controller shown in Figure 7, the two-node subnetwork in the top-right of the node plane acts as a pattern generator, in which the output of the right-back motor node "spikes" once every 8 time steps. Figure 8 gives the behavior of the two nodes over 100 time steps, showing node output (the bottom two graphs Y_2, Y_5), node transfer parameter (Graphs K_2, K_5), and the concentrations of the two gases (graphs C_1Y_2, C_1Y_5, C_2Y_2, C_2Y_5). Note the spiking behavior shown in the motor node Y_2 graph; once in every 8 time steps, the output of this motor node is positive.

As we shall see in Section 5.2, this spiking behavior is crucial to the final fitness of the solution; with both motors on, the robot would move straight forward. However, the right-back motor node turning on once in every eight time steps produces a slow clockwise

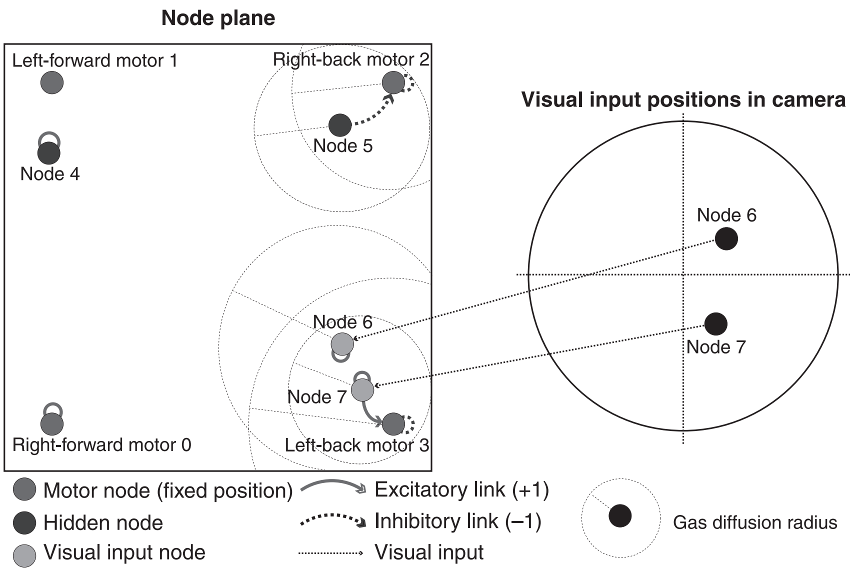


Figure 7 *Open-loop* GasNet visual discrimination network. Gas diffusion radii are shown only where diffusion occurs. The node plane is shown with {*x*, *y*} positions of each node, the connections between each node (indicating whether excitatory/inhibitory) including recurrent connections, and the position in the visual field of any external inputs.

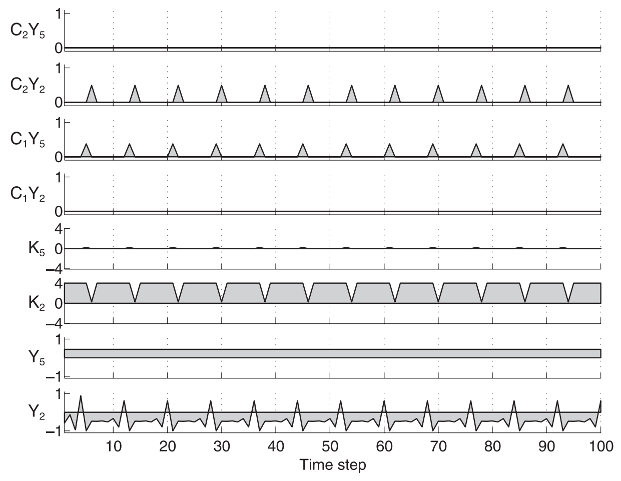


Figure 8 For nodes 2 (the right-back motor node) and 5, involved in the "spiking" subnetwork, the figure shows data over a run of 100 time steps for node output $Y \in [-1, 1]$, node transfer parameter $K \in [-4, 4]$, positive and negative gas concentrations C_1, $C_2 \in [0, 1]$ at the node site. Area between the output and time axis is shaded for clarity.

turn in the robot, which results in the robot arcing back toward the triangle. So, how is this spiking behavior generated?

The two systems are described by the dynamical equations for the nodes, governed by the input–output transfer function (Equation 1). For our two-node system, the right-back motor node (Y_2) receives an inhibitory recurrent input, and an excitatory input from the second node in the network (Y_5). The second node does not receive any input, and neither node receives external sensor input. Both nodes are potential gas diffusion emitters, with the motor node emitting gas 1 (thus increasing K on nearby nodes) when output activity is high, and node 5 emitting gas 2 (decreasing K on nearby nodes) when the concentration of gas 1 at node 5 is high. Thus we can write down our dynamical equations for the nodes:

$$Y_2^t = \tanh(K_2^t(-Y_2^{t-1} - Y_5^{t-1}) - 0.66) \tag{10}$$

$$Y_5^t = \tanh(K_5^t.(0) + 0.48) \approx 0.44 \tag{11}$$

Note that because node 5 receives no input, recurrent or otherwise, it has a constant output over time of approximately 0.44 for all values of the transfer parameter K_5.

The transfer parameter K is dependent on the current concentrations of gas at the node (Section 2.3), and as shown in Figure 8, it changes during the operation of the network. To consider the dynamical system as autonomous, we would need to incorporate the change of K over time into the equations. However, we can instead regard the dynamical equations as non-autonomous, changing over time through responding to external input, and solve for the values of K that the system is likely to encounter. By default (the genetically determined value) $K_2 = 4$, thus our motor node system is simply the one-dimensional:

$$y = \tanh(4(x - 0.44) - 0.66) \tag{12}$$

with Y_2^t rewritten as y, and Y_2^{t-1} rewritten as x. We find the fixed point(s) by setting $y = x$ and solving, obtaining a single fixed point for the motor node at $y = x = a = Y_2 \approx -0.48$. Now we want to know the stability of this point a. For the one-dimensional case we can use the first differential directly (Sandefur, 1990):

$$\left|\frac{dy}{dx}\right|_{y=x=a} \begin{cases} < 1 & \text{stable equilibrium value} \\ > 1 & \text{unstable equilibrium value} \\ = 1 & \text{inconclusive; need to look at higher derivatives} \end{cases} \tag{13}$$

Now, for the tanh transfer function used in this article:

$$y = \tanh(Kx + b) \tag{14}$$

$$\left|\frac{dy}{dx}\right|_{y=x=a} = |K(1 - a^2)| \tag{15}$$

which gives us the fixed point at $Y_2 \approx -0.48$ as unstable. Intuitively, we can understand this instability as a result of the high level of inhibitory feedback on the motor node. The output over time of the node, in the absence of high gas concentrations, will thus be an oscillating ± 1 2-cycle. In other words the node behavior is a limit cycle of period 2, in which node output alternates between $+1$ and -1 (see Figure 9a).

However, when the activity of the motor node is high, which will happen on every other time step once transients have died down, the motor node will emit gas 1. The distance between the nodes in the subnetwork is such that the concentration of gas 1 near node 5 will cause node 5 in turn to emit negative gas 2. Consideration of the concentration of this negative gas in the region of the motor node shows that K_2 will decrease from 4 to 0.25, seen in Figure 8. Application of fixed point stability analysis shows that the new

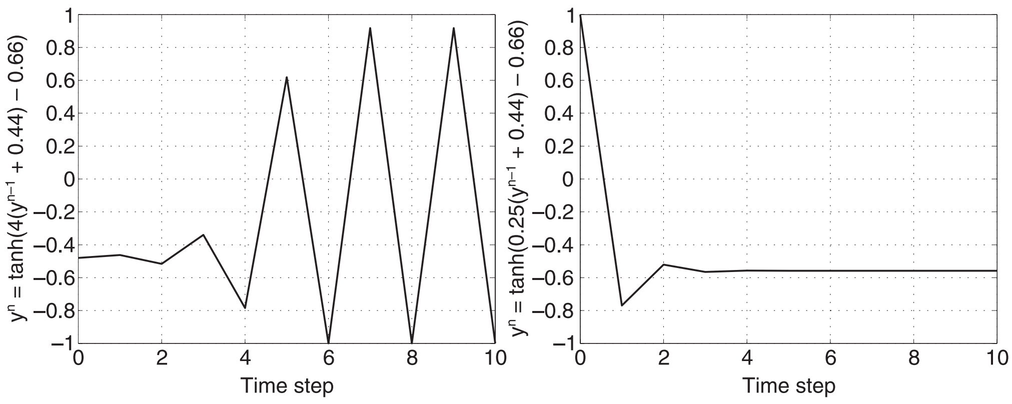


Figure 9 $Y_2^t = \tanh(-K_2^t(2Y_2^{t-1} + 0.44) - 0.66)$ behavior over time for different K_2^t values (modulated by the concentration of gas 2 at the node). The gas-concentration-mediated switch between these two dynamical states is the basis for pattern generation. (a) $K_2^t = 4$ has an unstable fixed point at $Y_2 \approx -0.48$. When gas concentration is low, the behavior is a ± 1 2-cycle. (b) $K_2^t = 0.25$ has a stable fixed point at $Y_2 \approx -0.56$ when gas concentration is high.

one-dimensional system for the motor node possesses a stable equilibrium point at $Y_2 \approx -0.56$, understood intuitively through the much smaller inhibitory recurrency on the motor node. Figure 9 shows the output over time of the motor node under the two gas concentrations.

Now, this stable fixed point will result in the motor node ceasing to emit gas due to low output activity, which in turn results in node 5 ceasing to emit gas, as emission was stimulated by the presence of gas 1. The fall in gas concentration will then increase K_2, destabilizing the fixed point. The motor node will return to the ± 1 2-cycle, and the pattern repeats.

Thus we have explained the pattern generation subnetwork. The base behavior of the motor node is a ± 1 2-cycle oscillation, providing the single spike seen in Figure 8. This spike stimulates gas emission from both nodes, resulting in the creation of a stable fixed point to which the motor node returns. This equilibrium state is destabilized by the subsequent decay of gas concentration, and the pattern repeats. Thus it is the interaction between the gas and electrical mechanisms in the network that produces the fixed limit cycle spiking behavior; high electrical activity of node 2 stimulates gas emission, which in turn inhibits node 2, in turn stopping the emission of gas which in turn finally allows node 2 to return to high electrical activity.

In a number of other successfully evolved GasNet controllers we have observed similar subnetworks. It appears that the properties of the GasNet class lend themselves readily to pattern generation. In the next section, we describe the operation of an entire robot control network used for visual discrimination in a noisy environment, using the techniques developed in this section.

5 Open-Loop GasNet Controller Analysis

In this section we analyze in detail the operation of the GasNet controller shown in Figure 7. It is seen that the mechanism underlying successful triangle discrimination is a permanent *open-loop* switch from one dynamical system to another, regulated by gas modulation of node properties. The open-loop nature of the switch ignores further external input. We show that the dynamical systems approach can be used to identify a number of possible reasons for the evolvability of the GasNet class, and we also analyze the failure modes of the controller.

Figure 7 shows the network layout for the open-loop GasNet controller, and Figure 10 shows two evaluations of the controller. In both evaluations, the robot rotates counter-clockwise until *after* it has rotated past the triangle, at which point it moves forward with a slow clockwise arcing turn that brings it back to the triangle.

The controller behavior is based on the two subnetworks in the right-hand corners of the node plane (Figure 7); both are required for accurate triangle finding behavior despite the lack of explicit interaction between the two networks. The first pattern generation subnetwork (consisting of nodes 2 and 5) was

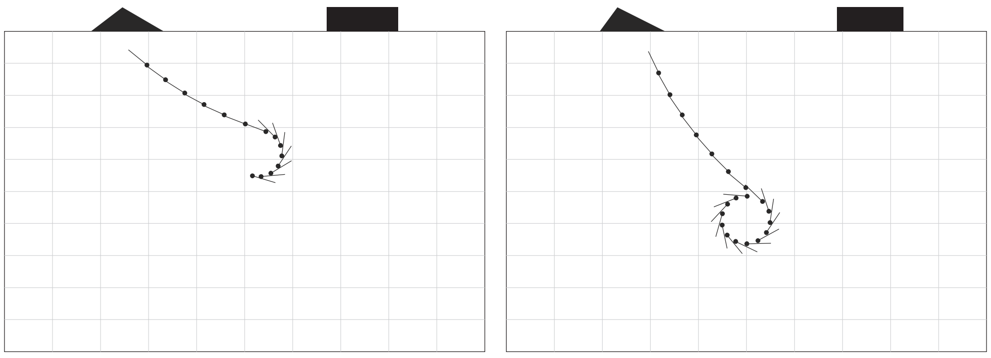


Figure 10 Two evaluations of the open-loop controller analysed in Section 5, showing the arena with the triangle and square shapes on the wall. The robot is represented by the black circle, with the line showing the forward direction. Note how the robot curves back in toward the triangle once it starts moving forwards, due to the pattern generation subnetwork analyzed in Section 4.1.

described fully in Section 4.1. This subnetwork produces a periodic output of the right-back motor node, in which the motor node is on for one time step in every eight, producing the slow clockwise turn once the triangle has been rotated past. The second subnetwork (consisting of nodes 3, 6, and 7) is described in Section 5.1 and produces a fixed behavior in which the subnetwork permanently switches from one stable state to another, again once the triangle has been rotated past. Both networks rely heavily on gas diffusion effects (disabling gas diffusion results in the failure of both networks), and both are required for the overall triangle discrimination behavior.

The network is described by the node transfer functions:

$$\text{Right-forward motor } Y_0^t = \tanh(-0.5Y_0^{t-1} + 0.48) \approx 0.31 \quad (16)$$

$$\text{Left-forward motor } Y_1^t = \tanh(0.16) \approx 0.16 \quad (17)$$

$$\text{Right-back motor } Y_2^t = \tanh(K_2^t(-Y_2^{t-1} - Y_5^{t-1}) - 0.66) \quad (18)$$

$$\text{Left-back motor } Y_3^t = \tanh(K_3^t(-Y_3^{t-1} + Y_7^{t-1}) + 0.62) \quad (19)$$

$$Y_4^t = \tanh(0.25Y_4^{t-1} - 0.28) \approx -0.35 \quad (20)$$

$$Y_5^t = \tanh(0.48) \approx 0.44 \quad (21)$$

$$Y_6^t = \tanh(K_6^t(Y_6^{t-1} + I_6^t) - 0.38) \quad (22)$$

$$Y_7^t = \tanh(K_7^t(Y_7^{t-1} + I_7^t) - 0.32) \quad (23)$$

where fixed values for the transfer function parameters K_i^t are shown only for nodes where no increase in gas concentration occurs during evaluation.

None of the three nodes *not* involved in the subnetworks receive any external input; nodes 0 and 1 (respectively the right- and left-forward motor nodes) stabilize at constant positive values given by $Y_0 = \tanh(-0.5Y_0 + 0.48)$ and $Y_1 = \tanh(0.16)$, respectively, while node 4 stabilizes at a constant negative value, $Y_4 = \tanh(0.25Y_4 - 0.28)$ but is unused by the network. Thus both forward motor nodes are continually on, and behavior is governed by the two subnetworks acting on the back motor nodes. In the next section we analyze the switching subnetwork.

5.1 Stable-State Switching

In the open-loop controller, the only nodes receiving external visual input are in the subnetwork involved in the triangle discrimination network, consisting of the left-back motor node 3, and nodes 6 and 7. The subnetwork, shown in the bottom-right corner of the node plane (Figure 7), regulates the left-back motor node through electrical synapse and gas diffusion effects. Both nodes 6 and 7 receive recurrent and visual input, while the motor node 3 receives recurrent input, plus an input from node 7. Figure 11 shows the output Y, transfer function parameter K, and gas concentrations C_1, C_2 for the three nodes. The three node subnetwork produces a dynamic system that can produce a permanent switch from one stable state to another, when a specific combination of high external sensory input is received. Note that we are treating the subnetwork as

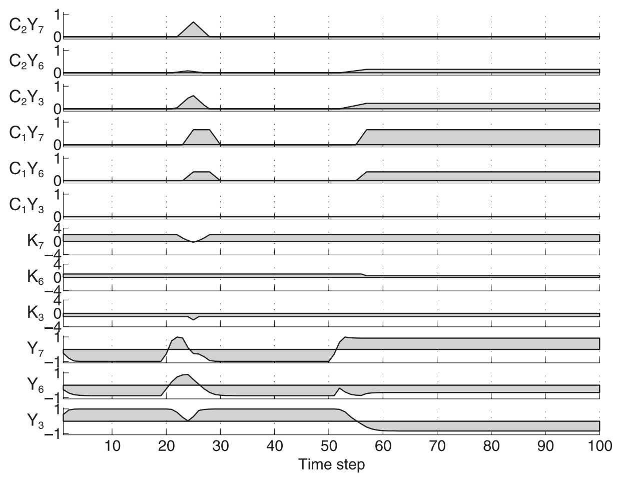


Figure 11 For nodes 3 (the left-back motor node), 6 and 7, involved in the "switch" subnetwork described fully in Section 5.1, the figure shows data over a run of 100 time steps for node output $Y \in [-1, 1]$, node transfer parameter $K \in [-4, 4]$, positive and negative gas concentrations C_1, $C_2 \in [0, 1]$ at the node site. Area between the output and time axis is shaded for clarity.

a nonautonomous dynamical system, receiving external input from the environment. External visual input is received by nodes 6 and 7, but only when the visual input level is above the genetically specified node input thresholds. We analyze the subnetwork behavior for the cases when inputs are below threshold, and when inputs are above visual input threshold levels.

5.1.1 Inputs Below Threshold Both nodes 6 and 7 have the same high visual input threshold, with only intensities above 0.84 having any effect. So we can investigate the case when input is below this, where the equations simplify to $Y_6^t = \tanh(Y_6^{t-1} - 0.38)$ and $Y_7^t = \tanh(2Y_7^{t-1} - 0.32)$ (in the absence of gas, $K_3^t = -1$, $K_6^t = 1$, and $K_7^t = 2$). The stable solution to these equations is $Y_3 \approx 1.0$; $Y_6 \approx -0.8$; $Y_7 \approx -1.0$. Note that Y_7 has 3 stable fixed points (with values $Y_7 \in \{-1.0, 0.3, 0.9\}$), but applying fixed point stability analysis (Section 4) shows that from an initial position of $Y_7 = 0.0$, the $Y_7 \approx -1.0$ solution is reached. However, the other solutions are crucial when input is above threshold, a situation that is analyzed in the sections below. Both nodes 6 and 7 emit negative gas when output activity is high, but this is not the case for the stable point. In the presence of negative gas, node 3 emits positive gas—again this is not the case for the stable point.

Thus we have the general picture when no visual input is received above the threshold level of 0.84. Both visual input nodes 6 and 7 are highly inhibited, and the left-back motor node 3 is highly excited. No nodes are emitting gas, and gas concentrations are zero in the neighborhood of each node. Thus the left motor is inhibited, and the robot circles counter-clockwise, due to the right motor being on for seven in eight time steps (remember that the spiking subnetwork on the left-back motor node only turns off the motor one in eight time steps; see Section 4.1). So what happens when inputs are above threshold?

5.1.2 Inputs Above Threshold The following analysis assumes inputs take their maximum value of 1.0 but is qualitatively the same for all values above the visual input thresholds of 0.84. In the presence of high input to both nodes (again in the absence of high gas concentrations), the equations simplify to $Y_6^t = \tanh(Y_6^{t-1} + 0.62)$ and $Y_7^t = \tanh(2Y_7^{t-1} + 1.68)$; the stable solution is $Y_3 \approx -0.8$; $Y_6 \approx 0.9$; $Y_7 \approx 1.0$. Note how all the node output activities have reversed; the previously inhibited nodes 6 and 7 are now excited, while the previously excited left-back motor node is now inhibited. The input threshold has produced an on/off "switch." The immediate effect of high visual input to node 7 is to turn off the left-back motor node through the inhibitory connection, thus turning on the left motor, so the robot follows a slow clockwise turn.

However, the picture is complicated by the emission of gas from the subnetwork nodes. Both nodes 6 and 7 emit negative gas when highly active, and node 3 emits positive gas in the presence of high negative gas concentrations. Three different scenarios are investigated: where both inputs go high at the same time, and where either input goes high first.

In the model of gas diffusion used, gas concentration builds up according to Equations 2 to 4, reaching a maximum concentration $C = C_0 e^{-(d/r)^2}$. The node 6 characteristics ensure negative gas spreads out very quickly over a large area: The concentration of negative gas at node 7 due to node 6 emission quickly affects the transfer function (on the very next time step). The small distance between nodes 6 and 7, and the high value of the radius of gas emission r for node 6, produce a gas concentration that drops K_7 from 2 to -0.25. Now $Y_7^t = \tanh(-0.25Y_7^{t-1} - 0.57)$ has a stable negative solution (–0.43) even with high positive sensory input. Thus, if node 6 receives bright input before node 7, node 7 is inhibited despite receiving bright input, so does not inhibit the left-back motor node, and the robot continues rotating.

The case where both inputs are bright at the same time is very similar to the case where node 6 receives high input before input 7, due to the faster diffusion of gas from node 6. Node 7 will inhibit the left-back motor node briefly (even more so as the combined negative gas from nodes 6 and 7 is concentrated enough to affect the left-back motor node transfer function parameter) but the negative gas build-up due to node 6 quickly inhibits node 7, and the left-back motor node will return to its previous excited state where $Y_3 \approx 1.0$.

Finally we turn to the case where node 7 receives bright input before node 6. The immediate effect is for node 7 to inhibit the left-back motor node. The secondary effect is for node 7 to emit negative gas. Despite the slower diffusion of gas from node 7 gas than from node 6, it is still enough to inhibit node 6 so long as node 7 receives bright input for *four or more time steps* before node 6; this time period is crucial to the behavior. Even with high input, node 6 cannot now produce output sufficient to emit gas so cannot inhibit node 7. Now, the three solutions to the node 7 equation with no input [$Y_7^t = \tanh(2Y_7^{t-1} - 0.32)$] mentioned previously come into play. From an initial condition of $Y_7 \approx 1.0$, even with no external input, there is a stable solution at $Y_7 \approx 0.9$. Thus the network is now in a highly stable state with node 7 output at near maximum with or without external input, node 6 inhibited due to negative gas emitted by node 7, and the left-back motor inhibited due to node 7 synapse output. The overall effect is to switch the network into a permanent open-loop behavior where further external input is irrelevant. Due to the inhibition of the left-back motor node, the left motor is on and the robot continues in a slow clockwise turn. So under what conditions does node 7 receive bright visual input four or more time steps before node 6?

5.1.3 Visual Input Positions, Success and Failure Modes

Figure 7 shows that the visual inputs to nodes 6 and 7 are vertically aligned in the visual field, with 7 directly below 6. Scanning across the square will cause both nodes to receive bright input at roughly the same time, thus node 7 will be inhibited by node 6, and the robot will continue rotating. However, scanning across the triangle will cause node 7 to receive bright visual input significantly before node 6, thus inhibiting node 6. This in turn will cause the network to switch into the permanent open-loop state, and the robot will continue in the slow clockwise turn. Table 1 summarizes the behavior of the robot as determined by the switching subnetwork.

Table 1 Summary of "switch" subnetwork behavior, showing the robot motion based on the visual input to nodes 6 and 7. The robot rotates fast counter-clockwise (ccw) for the cases where no bright visual input is received, and for the cases where node 6 receives bright input before or at the same time as node 7. The robot moves in a slow clockwise (cw) turn only when node 7 receives bright input significantly before node 6. Due to the visual input to node 6 being higher in the visual field than the input to node 7, node 7 will only reliably receive bright visual input before node 6 when the triangle is scanned across. Thus the robot will rotate past the square, but move toward the triangle

Node 6 input	Node 7 input	Node receiving first bright input	Left-back motor node	Robot motion
Dark	Dark	–	Excited	Fast ccw rotation
Bright	Dark	6	Excited	Fast ccw rotation
Bright	Bright	6	Excited	Fast ccw rotation
Bright	Bright	Same time	Excited	Fast ccw rotation
Dark	Bright	7	Inhibited	Slow cw rotation
Bright	Bright	7	Inhibited	Slow cw rotation

We can also see the failure modes from this analysis. It is the four (or more) required time steps of bright input to node 7, without bright input to node 6, that produces visual input noise filtration. However, an extremely noisy environment may "fool" the controller into the permanent dynamical system switch, through only node 7 receiving bright input. Such a situation may occur with flashes of light aimed at only certain parts of the arena, which might be erroneously identified as triangles by the controller. Similarly, bright shapes with angled edges leading to node 7 receiving input before node 6 will be identified by the controller as triangles, and approached. Finally, triangles with edges insufficiently angled to allow input 7 to be bright for four time steps before input 6 will not be approached by the controller. Among these unidentified triangles will be upside-down triangles and right-angled triangles. In the next section, we summarize the overall behavior of the controller and draw some general principles of GasNet robot controller operation.

5.2 Open-Loop GasNet Controller Summary

The overall behavior of the robot controller can be summarized as follows. In the absence of bright visual input, the robot rotates counter-clockwise, with the right motor permanently excited, and the left motor inhibited by the switching subnetwork. This behavior continues until the robot scans across a bright object, such that the lower half of the visual field receives bright input significantly before the upper half. This permanently switches off the left-back motor node, exciting the left motor and causing the robot to move forward. Now the effect of the spiking subnetwork is seen; once every eight time steps the right motor is turned off, thus the robot moves in a slow clockwise arc back toward the triangle, which it has rotated past. So, we have explained in full the behavior seen in the two example evaluations, shown in Figure 10.

The two subnetworks analyzed are crucial to the understanding of the robot controller triangle discrimination, in conjunction with the robot–environment coupling. The primary robot–environment coupling is the permanent switch mechanism; scanning across the square will produce no change in the robot motion beyond a slight slowing of the turn. By contrast, scanning across the triangle will lock the robot into a fixed behavior in which no subsequent external input affects the network, and with both motors full on the robot goes forward toward the bright object. The second subnetwork is used by the controller to compensate for both the relatively slow time course of the permanent switch mediated by the gas diffusion, and the momentum of the robot. While rotating past the triangle, the permanent switch behavior takes some time to come into play, and the robot motors take some time to overcome momentum and friction. It is the right motor turning off once in every eight time steps that adjusts for these effects, turning the robot back toward the triangle. Without this spiking behavior, the robot would overshoot and run into the wall past the triangle; it is the lack of active "closed-loop" tracking that produces the need for this compensation. In the next section we hypothesize why the GasNets network class is more evolvable than the NoGas network class.

5.3 Why are GasNets Good for Evolution?

From this detailed analysis of the open-loop controller, we can frame some preliminary conclusions on the usefulness of the mechanisms utilized in GasNet controllers for the generation of adaptive behavior over time. First, tunable pattern generation is extremely easy to produce using GasNet controllers. In general, pattern generation is based on limit cycle behavior, with the system cycling through some set of states (Beer, Chiel, & Gallagher, 1999; Chiel et al., 1999). As we have seen from the analysis in Section 4.1, the spiking subnetwork used by the open-loop controller operated in exactly such a fashion; the high fitness of the controller is due to this subnetwork slowly turning the robot back toward the triangle. This leads to our first hypothesis for why the GasNet class is more evolvable than the NoGas class; the GasNets are more amenable to being "tuned" to the specific characteristics of the environment. The pattern produced in which the right-back motor node spiked once in every eight time steps was perfectly tuned to the speed and size of the robot wheels, the size of the triangle, and the size of the arena in which the robot operated. A different pattern would not have produced such high fitness in this environment, and the same pattern would not have produced such high fitness in a different environment. We hypothesize that the same kind of environmental tuning is more difficult with the NoGas class.

The tuning of generated patterns is closely related to our second hypothesis regarding useful properties in the GasNet class for adaptive behavior: the ability to switch between stable states, in other words a discontinuous change of behavior determined by external input, and the ability to mediate such switching. This is clearly possible to achieve without gas modulation, but the features of the gas diffusion mechanism allow such a switch to take place over several time steps, through the build-up of gas concentration levels. This was seen in the functionally equivalent mechanisms of Section 5.1, where the switch from fast counter-clockwise rotation to slow clockwise rotation was inhibited by bright input, *only when such bright input was received over several time steps*. Thus the switching can be based on input patterns received over time, not just at a single time point.

Finally, the ability to filter out noisy input is straightforward to produce when using the GasNet controller class, similarly through requiring that input be consistent over several time steps. This was seen in the requirement that bright input was received several time steps earlier by visual input nodes lower in the visual field, before the controller responded (Section 5.1). Thus, bright flashes and other noisy environment effects were efficiently excluded by the robot controller.

In the next two sections, we investigate these hypotheses in two ways. First, we analyze and compare two solutions utilizing the same shape discrimination strategy, one GasNet controller and one NoGas controller, to compare the underlying mechanisms. Second, we re-evolve previously evolved controllers in an environment with different characteristics to the environment in which they were originally evolved, to compare the tunability of the mechanisms used by the evolved controllers.

6 Functionally Equivalent GasNet and NoGas Controllers

Two controllers, one evolved using the GasNet class and one evolved using the NoGas class, were analyzed using the dynamical system methods of the previous sections. It was found that both employed the same strategy for the triangle–square discrimination task, based on a method of timing the duration for which bright visual input was received in the upper half of the visual field. Due to triangles being narrower at the top than squares, this allows the controllers to discriminate successfully between the two shapes. In this section we investigate the GasNet and NoGas mechanisms for timing the duration over which bright input is received and argue that the GasNet mechanism is simpler to tune to the characteristics of the environment.

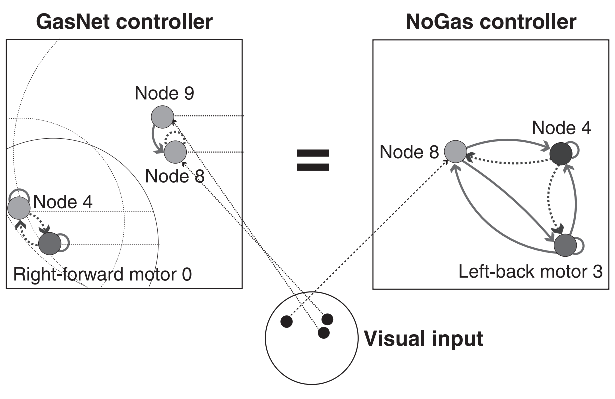


Figure 12 The two functionally equivalent subnetworks analyzed in Section 6. Both employ the same strategy for triangle–square discrimination; timing the duration of receiving bright input in the upper half of the visual field. Because triangles are narrower than squares at the top, this allows the shapes to be discriminated successfully.

Figure 12 shows the two functionally equivalent subnetworks; both controllers time the duration over which bright input is received from visual inputs in the upper half of the visual field. A second visual input mechanism (not shown) acts simply as a ''bright object finding detector.'' This bright object finding mechanism is "later" in the visual field than the timing mechanism, in the sense that the position of visual input and direction of robot rotation is such that the bright object finding mechanism will "see" things after the timing mechanism. The bright object behavior is inhibited if the duration of bright input to the timing mechanism is sufficiently long, which is the case when scanning across the square, but not when scanning across the triangle. Thus the controller approaches the triangle but rotates past the square. In the next sections we describe the methods by which the GasNet and NoGas solutions produce such a timing mechanism.

6.1 The GasNet "Timer"

With the GasNet class, it is simple to produce a timing mechanism that retains activity for some time after the initial input has been received. A single node receiving visual input, and with the property that gas emission occurs when the node output activity is high, will

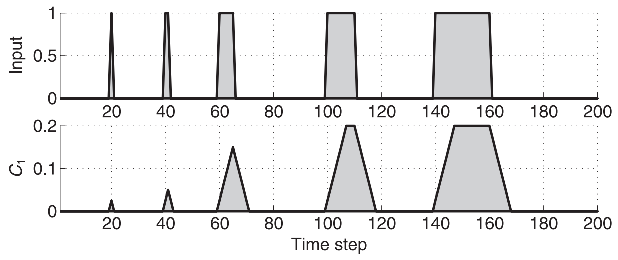


Figure 13 Gas concentration C_1 over 200 time steps. A square wave external visual input of increasing width is applied as input, to illustrate the differences between the output seen for the triangle and for the square. Area between the output and time axis is shaded for clarity.

start emitting gas when bright input is received. The gas concentration built up during emission will take some time to decay once bright input is no longer received, with this decay time being a function of the genetically specified rate of gas concentration build-up. Remember from Equations 2 to 4 (Section 2.2) that gas concentration decays in a Gaussian fashion with distance from the emitting node, but increases linearly over time during emission, and decreases linearly over time once emission stops. As the time taken for gas concentration to build up to a maximum is specified by the genotype, the mechanism is simple to tune to the characteristics of the environment. Figure 13 shows the concentration of gas over time for a given square wave input of varying duration, simulating the effect of bright visual input to the emitting node.

For this timing mechanism to affect the controller operation, we require the built-up gas concentration to affect the motor node activity. Again, this is relatively simple to effect, as the gas concentration can modulate either the motor node, or as in the controller analyzed here, another visual input node that has an output connection to the motor node. The left-hand side of Figure 12 shows the subnetwork responsible for the GasNet timing mechanism; bright input to nodes 8 and 9 produces concentration of gas 1 at node 4. If gas concentration is high enough, the increase in the node 4 transfer parameter K increases output sufficiently to inhibit the right-forward motor node. Thus, the robot rotates clockwise past the square, but moves straight toward the triangle. In the next section we describe the intuitively less-obvious operation of the NoGas timing mechanism.

6.2 The NoGas "Timer"

A fully connected three-node subnetwork based around a motor node (see the right-hand side of Figure 12) allows the NoGas solution to create exactly the same timing mechanism seen in the previous section. Dynamical systems analysis of the subnetwork shows a single stable equilibrium point for the system when visual input is below the input threshold, and a different single stable equilibrium when visual input is above threshold.

The key to the timing mechanism is how the system moves between these fixed points when the visual input changes. The fully connected feedback nature of this three-node system makes it impossible to give a full quantitative description of the behavior, but qualitative features can be outlined. With no bright input to node 8, the system settles into the first stable fixed point described above, while with bright input the system moves toward the second stable point. Once bright input is no longer received, the system slowly decays back to the first stable fixed point.

The feedback between the nodes ensures that the decay between stable states is fairly slow, producing an effect that can build up and decay over time, in a similar fashion to that of gas concentration. The longer that bright input is received for, the nearer to the high visual input stable state the system reaches, and the longer it takes to decay back to the low visual input stable state. Figure 14 shows the outputs for the three nodes in the system when a square wave visual input is applied to node 8, and clearly shows the slow change from one state to another when visual input changes from dark to bright, and vice versa. It takes roughly 10 time steps for node 3 to reach the bright input stable state, and roughly 30 steps to decay back to the dark visual input regime. The motor node activity Y_3 is the crucial value; as this goes from negative to positive, the left motor is inhibited, and the robot does not approach the bright object. Only when sufficient bright input has been received will this occur, for instance, when the square has been scanned across.

Thus we have our NoGas timing mechanism. Instead of using the GasNet build-up and decay of gas concentration to modulate motor node properties, the NoGas version uses a fully connected subnetwork that decays from one stable equilibrium state to another, depending on the level of visual input received. As proposed in Section 5.3, we hypothesize that the Gas-

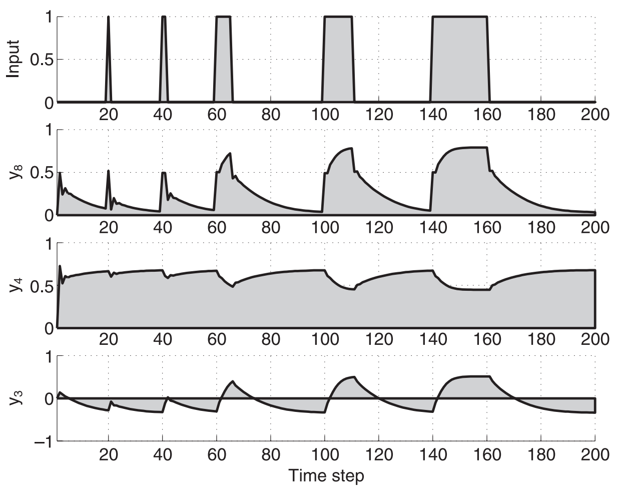


Figure 14 Node output data (ranging from ±1) for nodes 3, 4, and 8 over 200 time steps. A square wave external visual input is applied to node 8 to illustrate the two fixed points of the system, and the slow decay from one state to the other. Area between the output and time axis is shaded for clarity.

Net version is easier to tune to the specific characteristics of the environment than the NoGas version. In the next section, we test this hypothesis for the two functionally equivalent controllers through re-evolving the controllers in environments with modified properties.

7 Re-Evolution of Controllers in Modified Environments

The hypothesis that GasNet controllers are easier to tune to the specific properties of the environment than NoGas controllers can be investigated through the behavior of the controllers in environments with modified properties. In this section, we analyze controllers when evaluated in two separate environments, where the robot motor speeds are respectively set to double and quarter the usual motor speeds. This has the effect of making the robot move at a different speed in the arena, in particular spinning past the two shapes at very different rates from the speeds encountered during the original evolutionary phase. Note that the environments could be modified similarly through altering the size and properties of the shapes, and/or the size of the arena. Other modifications could also investigate the effect of re-evolving from lesioned or similarly modified control networks. However, in this work we focus on modification of the robot motor speeds.

7.1 Re-Evolution of the Functionally Equivalent Controllers

We would expect the crucial timing mechanisms described in Section 6 to be affected by evaluation in environments with modified robot speeds, with the time spent spinning past the triangle and square much shorter in the double-speed environment, and much longer in the quarter-speed environment. However, the hypothesis that the GasNet mechanism is in some sense easier to tune to the particular properties of the environment can be tested through seeding the controllers back into the evolutionary process, with fitness based on evaluation in the modified environments. We can then re-run the evolutionary process from the controller seeds, assessing how long before controllers of 100% fitness are again achieved. Although this will not tell us directly how easy the controller was to originally tune to the environment, we argue that the evolutionary tuning processes involved are similar. In other words, if it is much easier to tune the evolved GasNet controller to the specific characteristics of the modified environment, it would also have been much easier to tune the GasNet controller to the original environment.

The two controllers were used to seed the initial populations for the distributed evolutionary algorithm (Section 2.6), and evolution repeated 20 times for each controller in each modified environment until controllers of 100% fitness were observed. In this re-evolution, we allow only the parts of the genotype involved in the timing mechanism to be affected by the evolutionary process; we are assessing how easy it is to modify the actual mechanism itself, not the rest of the network.

Results for re-evolution studies of the two controllers are given in Table 2. In the double speed environment, both controllers drop in fitness to well under 20%, with no significant differences seen between the fitness of the two controllers. However, there is massive difference in the number of generations required to re-evolve controllers of 100% fitness; the GasNet controller is much easier to tune to the modified properties of the environment (10 generations on average compared with 409 generations). In the quarter-speed environment, the GasNet controller achieves significantly higher fitness than the NoGas controller, but the difference in the number of generations required to reach 100% fitness controllers is much larger than might be predicted by this fitness difference (30 generations

Table 2 Data for the two functionally equivalent networks shown in Figure 12, re-evolved in two modified environments. The robot motors are set to double-speed and quarter-speed, respectively, and the two controllers evaluated 100 times for fitness, then used to seed the initial populations for the evolutionary algorithm until 100% fitness controllers were produced (20 runs were performed for each controller on each condition). The evaluated fitnesses, and mean, median, and standard deviation of the number of generations of re-evolution required to reach 100% fitness controllers are shown, with significant differences between the GasNet and NoGas controllers highlighted (both parametric *t*-tests and nonparametric Mann–Whitney *U*-tests were performed; $*p < 0.05$, $**p < 0.01$)

	Double speed		Quarter speed	
	GasNet	NoGas	GasNet	NoGas
Number of runs	20	20	20	20
Mean evaluated fitness (σ)	0.17 (0.074)	0.15 (0.066)	0.36* (0.10)	0.21 (0.016)
Mean re-evolution generations (σ)	10 (5)**	409 (336)	30 (31)**	591 (346)
Median re-evolution generations	10**	360	19**	608

on average compared with 591 generations). So in both modified environments, the GasNet controllers are much easier to evolve successful controllers than would be predicted from their fitnesses; the GasNets are more tunable.

From the results presented in this section, we can support the hypothesis of the previous section, namely, that the GasNet controllers are easier to tune to the specific properties of the environment than the corresponding NoGas controllers. In the next section we extend this to a sample of previously evolved controllers.

7.2 Re-Evolution of a Sample of Controllers

It may be argued that the two functionally equivalent controllers investigated are based on a mechanism that is in principle easier both to produce and tune using the GasNet control class. Thus re-evolving the timing mechanism in modified environments will unfairly favor the GasNet controller. By contrast other mechanisms may favor NoGas classes; here we counter this argument through extending the re-evolution analysis to a random sample of 40 previously evolved GasNet and NoGas controllers of 100% fitness.

The 40 controllers were used to seed the initial populations for the distributed evolutionary algorithm, which was run until controllers once more showed 100% fitness, with fitness evaluated in the same double- and quarter-speed environments described in the previous section. Table 3 shows the results for the two conditions, averaged over 10 evolutionary runs of each of the 40 controllers. The results are not as striking as those from the functionally equivalent controllers, lending some weight to the hypothesis that the previous analysis unfairly favored the GasNet mechanism. However, the GasNet controllers still showed significantly faster re-evolution than the NoGas controllers. In the double-speed environment, both samples of controllers fell to average fitnesses of 0.26, but the GasNet controllers on average re-evolved in 107 generations compared with 240 generations for the NoGas controllers. In the quarter-speed environment, the differences are much smaller, with comparable mean numbers of generations for re-evolution, but there is evidence of faster evolution from the median numbers of generations. Thus from our sample of GasNet controllers, we also see evidence of significantly faster re-evolution to modified environments; the GasNets are more tunable.

Smith (2002) has also investigated the evolution and re-evolution of abstract central pattern generation networks, similarly finding that the GasNet class is more tunable to the required pattern than the corresponding NoGas class. In the next section we draw together the various experiments carried out in this article.

8 Summary

The detailed analysis of a number of GasNet and NoGas controllers allowed us to frame two hypotheses regard-

Table 3 Data for a sample of 20 GasNet and 20 NoGas controllers, re-evolved in two modified environments. The robot motors are set to double-speed and quarter-speed, respectively, and the two controllers evaluated 100 times for fitness, then used to seed the initial populations for the evolutionary algorithm until 100% fitness controllers were produced (10 runs were performed for each controller on each condition). The evaluated fitnesses, and mean, median, and standard deviation of the number of generations of re-evolution required to reach 100% fitness controllers are shown, with significant differences between the GasNet and NoGas controllers highlighted (both parametric *t*-tests and nonparametric Mann–Whitney *U*-tests were performed; $^{*}p < 0.05$, $^{**}p < 0.01$)

	Double speed		Quarter speed	
	GasNet	NoGas	GasNet	NoGas
Number of runs	200	200	200	200
Mean evaluated fitness (σ)	0.27 (0.13)	0.26 (0.18)	0.35 (0.27)	0.29 (0.019)
Mean re-evolution generations (σ)	107 (190)**	240 (363)	108 (229)	116 (252)
Median re-evolution generations	36**	49	13**	21

ing the suitability of the GasNet class to robot control. First, the ability both to produce and modify central pattern generation output was seen to be central to a number of evolved control solutions. This seems surprising. We are not investigating such behaviors as walking and swimming gaits, or rhythmic feeding, where behavior is often based on central pattern generation. Our visual shape discrimination task might not appear at first sight to be related to such pattern generation. However, a number of GasNet controllers were seen to use pattern generation subnetworks in the final evolved behavior.

Second, the ability to switch between dynamical states dependent on external input, and the ability to mediate this switch over a number of time steps, was seen to be extremely useful both in behavior generation and in filtering environmental noise. From analysis of the functionally equivalent GasNet and NoGas controllers in Section 6, we argued that the kinds of active perception timing strategies able to mediate such behavior switching and noise filtration were much easier to evolve using the GasNet class. To develop and tune the NoGas timing mechanism required the construction of a complex fully connected circuit, whereas the corresponding GasNet timing mechanism was based on the build-up and decay of gas.

The twin hypotheses that GasNet classes were more amenable to both the development and tuning of pattern generation and the development and tuning of switching mechanisms were supported by the re-evolution studies. We saw that the functionally equivalent GasNet controller was much easier to tune to a modified environment than the corresponding NoGas controller. To a lesser extent, although still significant, this same re-evolution tunability was seen over a large sample of previously evolved controllers.

So, can we draw any conclusions from this work on what makes an evolvable network class for the visual discrimination problem? The simple answer is yes. The key feature of the GasNets seen to be useful for this task is the ability to adapt smoothly to the temporal characteristics of the environment. This encompasses the initial development and subsequent tuning of the controllers to the detailed properties of the robot and environment in which it finds itself. Included in this ability to adapt smoothly to the temporal characteristics of the environment is the ability to generate a rich variety of temporal patterns through the interaction of the gas diffusion mechanism and the electrical synaptic mechanism. The different time courses over which these two mechanisms operate were seen to be crucial to this pattern generation, along with the ability of the evolutionary process to modify those time courses. Indeed, there is a tantalizing link here with the role of the gaseous neuromodulator nitric oxide in real nervous systems, where NO is often implicated in the modulation of central pattern oscillations (see, for example, Gelperin, Flores, Raccuia-Behling, & Cooke, 2000).

By showing that the evolvability of the GasNets is due to this principle of temporal adaptivity, we have provided some support for the intuition of many evolutionary robotics practitioners, namely, that robot controllers operating in the real world must incorporate temporal structure, and that the evolutionary process must be able to adapt that structure easily. For example, Harvey (1993) makes the point that "in environments where physical events have natural time scales, the dimension of time is not an optional extra, but fundamental" (p. 50). For a robot operating over time and moving in some environment, changing any aspect of that environment may affect the observed sensory input over time. For example, changing either the size of objects, or the speed at which they move, will change the time period over which they impact on to the robot sensory inputs. Thus, the ability for evolution easily to adapt solutions to the temporal characteristics of the environments in which they operate is crucial to the evolution of controllers able to generate adaptive behavior over time.

On this principle of temporal adaptivity, the GasNet neural network class falls squarely into a larger class that is likely to contain other networks with temporally modifiable properties, such as continuous time recurrent networks (Beer & Gallagher, 1992), pulsed neural networks (Maass & Bishop, 1999), and networks with time-lagged synaptic activity (Harvey, 1993). However, we argue that simple recurrent networks such as the NoGas are not members of this class; although activity is retained over time and the connection architecture may be arbitrarily modified, it is not straightforward for the evolutionary process to modify the time courses of mechanisms operating in the network. For example, as seen in the NoGas timer (Section 6.2), it is not easy to set up an effect that lasts for some given period of time. Similarly, generating and tuning patterns is more difficult with simple recurrent networks (Smith, 2002).

9 Discussion

It is often implicitly assumed that the incorporation of plasticity into agent controllers is primarily useful to adapt to change in the immediate environment of the agent. This definition also includes developmental plasticity, in which the agent adapts to the initial environment in which it finds itself (see, for example, Floreano & Urzelai, 1999). Recent interest in the Baldwin effect (Hinton & Nowlan, 1987; Mayley, 1996) has extended this picture, highlighting the role of lifetime plasticity in increasing the evolvability of the underlying solution class. However, in this article we have investigated a third possibility: plasticity as a mechanism by which controllers can utilize a range of different time courses during operation. In the majority of work on plasticity in artificial systems, the time course of the plasticity process is not explicitly specified; for example, Hebbian learning rules typically update synaptic weights instantaneously. However, here we have explicitly allowed the time course of the plastic neuromodulatory effect to vary, dependent on gas emission from nearby nodes.

In the GasNets investigated in this article, evolved controllers used the modulation of neuron transfer functions as a process operating over a different time course to that of the underlying network synaptic activity. The particular plasticity mechanism used, concentration-dependent neuronal modulation, allowed the GasNet controllers easily to generate and tune temporal patterns, and tune behavior to the particular temporal characteristics of the environment. This was seen to have a direct effect on the evolvability of the GasNet class. However, in highlighting the functional role of the gaseous diffusion in robot control, we have also implicitly addressed two other issues, discussed below.

It might be argued that the gaseous modulation of node parameters during operation plays a similar role to stochastic learning in evolutionary simulations of the Baldwin effect (Baldwin, 1896). The role of learning in evolution is often likened to a search of nearby volumes of genotype space, with the final fitness returned as the average over the search volume. For example, Hinton & Nowlan (1987) show that randomly setting undecided parameters in a bit-string genotype, in other words, allowing a form of stochastic lifetime learning, enables evolutionary search to solve a needle-in-a-haystack problem. In essence the lifetime learning provides the evolutionary process with the information that "you're getting warm." This same process may occur during the evolution of GasNet controllers, if the neuromodulatory effect of the gases is to explore nearby controller genotypes through modifying the network node properties. In other words, it may be that the current controller is no good, but altering the gains of one or more nodes generates more successful controllers; the gaseous neuromodulation would then provide selective pressure toward these controllers. Unfortunately, the functional role seen for the gaseous modulation does not support this argument; the gaseous diffusion is playing an active part in the evolved behavior, not simply sampling similar networks with modified node properties. However, it is an interesting possibility that the effect plays a part earlier in the evolutionary process.

A different argument for the usefulness of a similar neuromodulatory mechanism was made by Eggenberger et al. (1999), when evolving robot controllers for a block-pushing task. In their model, associative learning of synaptic weights in a fixed architecture network controller could be turned on and off by the concentration of a number of global gases. They argue that this process does not primarily play a functional role in the behavior but rather acts as a mechanism to allow transferral from controllers evolved in simulation to operation in the real world.[2] In other words, the plasticity in the network is useful not in the original evolution, but in allowing the controllers to operate in slightly different (real-world) environments from the (simulated) environments in which they have been evaluated during evolution. There are obvious parallels with the re-evolutionary analysis carried out in Section 7, in which the GasNets were seen to evolve to modified environments much faster than the NoGas. Although there was no evidence that the GasNets were more robust to the modified environment before re-evolution, it may well be that such robustness would be seen in less drastically changed environmental properties. This is an intriguing possibility for the future development of robust controllers able to operate in environments that change over time.

And finally, to return to the central theme of this article. The promotion of evolutionary robotics and other artificial evolution methodologies into practical techniques for real-world applications depends crucially on the design and evaluation of robust evolvable solution classes. In this article, we have shown that such evaluation can come from analysis of the operation of successfully evolved solutions. The development of evolvable solution classes remains a significant challenge. The techniques developed in this article provide one potential way to address this challenge.

Notes

1 Commonly defined as the "ability to evolve"; see Smith et al. (2002) for discussion.

2 The mechanism behind the model derived by Eggenberger et al. (1999) may turn out to be similar to the homeostatic networks evolved by Di Paulo (2000), in which associative learning is turned on by node activity falling outside some given range. Di Paolo has shown that such networks can evolve to cope with radically disruptive change, such as reversing the position of sensory inputs.

Acknowledgments

The authors would like to thank the three anonymous reviewers. We would also like to thank members of the Centre for Computational Neuroscience and Robotics (http://www.cogs.susx.ac.uk/ccnr/) for constructive comments and discussion, and the Sussex High Performance Computing Initiative (http://www.hpc.sussex.ac.uk/) for computing support. TS and AP are funded by British Telecom Future Technologies Group (http://innovate.bt.com/projects/ftg.htm) sponsored Biotechnology and Biology Science Research Council Case awards.

References

Baldwin, J. M. (1896). A new factor in evolution. *American Naturalist*, *30*:441–451.

Beer, R. D. (1990). *Intelligence as adaptive behaviour: An experiment in computational neuroethology.* Cambridge, MA: Academic Press.

Beer, R. D. (1995). On the dynamics of small continuous-time recurrent neural networks. *Adaptive Behaviour*, *3*(4):469–509.

Beer, R. D., Chiel, H. J., & Gallagher, J. C. (1999). Evolution and analysis of model CPGs for walking II. General principles and individual variability. *Journal of Computational Neuroscience*, *7*(2):119–147.

Beer, R. D., & Gallagher, J. C. (1992). Evolving dynamical neural networks for adaptive behaviour. *Adaptive Behaviour*, *1*:94–110.

Calvatti, A., & Beer, R. D. (2001). Analysis of a distributed model of leg coordination I. Individual coordination mechanisms. *Biological Cybernetics*, *82*:197–206.

Chiel, H. J., Beer, R. D., & Gallagher, J. C. (1999). Evolution and analysis of model CPGs for walking I. Dynamical Modules. *Journal of Computational Neuroscience*, *7*(2):99–118.

Di Paolo, E. A. (2000). Homeostatic adaptation to inversion of the visual field and other sensorimotor disruptions. In J.-A. Meyer, A. Berthoz, D. Floreano, H. Roitblat, & S. W. Wilson (Eds.), *From animals to animats 6: Proceedings of the Sixth International Conference on Simulation of Adaptive Behaviour (SAB'2000)* (pp. 440–449). Cambridge, MA: MIT Press.

Eggenberger, P., Ishiguro, A., Tokura, S., Kondo, T., Kawashima, T., & Aoki, T. (2000). Towards seamless transfer from simulated to real worlds: A dynamically-rearranging neural network approach. In J. Wyalt & J. Demiris (Eds.), *Advances in robot learning—8th European Workshop on Learning Robots EWLR-8 Lausanne, Switzerland, September 18, 1999 Proceedings* (pp. 4–13). Berlin: Springer.

Floreano, D., & Urzelai, J. (1999). Evolution of neural controllers with adaptive synapses and compact genetic encoding. In D. Floreano, J.-D. Nicoud, & F. Mondada (Eds.), *Advances in artificial life: 5th European Conference on Artificial Life (ECAL'99)* (pp. 183–194). Berlin: Springer.

Gallagher, J. C., & Beer, R. D. (1999). Evolution and analysis of dynamical neural networks for agents integrating vision, locomotion, and short-term memory. In W. Banzhaf & R. E. Smith (Eds.), *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO'99)* (pp. 1273–1280). San Mateo, CA: Morgan Kaufmann.

Gavalas, G. R. (1968). *Nonlinear differential equations of chemically reacting systems*. Berlin: Springer.

Gelperin, A., Flores, J., Raccuia-Behling, F., & Cooke, I. R. C. (2000). Nitric oxide and carbon monoxide modulate oscillations of olfactory interneurons in the terrestial mollusk. *Journal of Neurophysiology*, *83*:116–127.

Goldstein, H. (1980). *Classical mechanics* (2nd ed.). Redwood, CA: Addison-Wesley.

Harvey, I. (1993). *The artificial evolution of adaptive behaviour.* Unpublished doctoral dissertation, School of Cognitive and Computing Sciences, University of Sussex, UK.

Hinton, G. E., & Nowlan, S. J. (1987). How learning can guide evolution. *Complex Systems*, *1*:495–502.

Husbands, P. (1998). Evolving robot behaviours with diffusing gas networks. In P. Husbands & J.-A. Meyer (Eds.), *Evolutionary robotics: First European workshop, EvoRobot98* (pp. 71–86). Berlin: Springer.

Husbands, P., Smith, T. M. C., Jakobi, N., & O'Shea, M. (1998). Better living through chemistry: Evolving GasNets for robot control. *Connection Science*, *10*(3-4):185–210.

Jakobi, N. (1998). Evolutionary robotics and the radical envelope of noise hypothesis. *Adaptive Behavior*, *6*:325–368.

Maass, W., & Bishop, C. M. (Eds.). (1999). *Pulsed neural networks*. Cambridge, MA: MIT Press.

Mayley, G. (1996). Landscapes, learning costs and genetic assimilation. *Evolutionary Computation*, *4*(3):213–234.

Philippides, A., Husbands, P., and O'Shea, M. (2000). Four-dimensional neuronal signaling by nitric oxide: A computational analysis. *Journal of Neuroscience*, *20*(3):1199–1207.

Rosen, R. (1970). *Dynamical system theory in biology*, Volume I. Stability theory and its applications. New York: Wiley.

Rubinow, S. I. (1975). *Introduction to mathematical biology.* New York: Wiley.

Sandefur, J. T. (1990). *Discrete dynamical systems: Theory and applications*. Oxford, UK: Oxford University Press.

Smith, T. M. C. (2002). *The evolvability of artificial neural networks for robot control*. Unpublished doctoral dissertation, School of Biological Sciences, University of Sussex, UK.

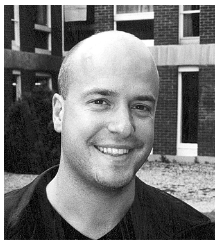
Smith, T. M. C., Husbands, P., Layzell, P., & O'Shea, M. (2002). Fitness landscapes and evolvability. *Evolutionary Computation*, *10*(1):1–34.

Smith, T. M. C., Husbands, P., & O'Shea, M. (2001a). Neutral networks and evolvability with complex genotype-phenotype mapping. In J. Kelemen & P. Sosík (Eds.), *Advances in artificial life, 6th European Conference (ECAL'2001)* (pp. 272–281). Berlin: Springer.

Smith, T. M. C., Husbands, P., & O'Shea, M. (2001b). Not measuring evolvability: Initial exploration of an evolutionary robotics search space. In J. H. Kim, B.-T. Zhang, G. Fogel, & I. Kuscu (Eds.), *Proceedings of the 2001 Congress on Evolutionary Computation (CEC'2001)* (pp. 9–16). Piscataway, New Jersey: IEEE Press.

Smith, T. M. C., Husbands, P., & O'Shea, M. (in press). Local evolvability of statistically neutral GasNet robot controllers. *Biosystems*.

About the Authors



Tom Smith received his D.Phil. in biologically inspired approaches to robotics from the School of Biological Sciences at Sussex in 2002. Prior to that he received an M.Sc. in knowledge-based systems from the School of Cognitive and Computing Sciences at Sussex, and an M.A. in natural sciences from King's College, Cambridge, specializing in theoretical and astrophysics. He is currently a research fellow at the Centre for Computational Neuroscience and Robotics at the University of Sussex, and his research interests include evolutionary computation, biologically inspired control systems, data visualization, and the generative arts.

Phil Husbands received the B.Sc. degree in physics from Manchester University in 1981, the M.Sc. degree in computer systems engineering from South Bank Polytechnic, London in 1985, and the Ph.D. degree in computer-aided engineering from Edinburgh University in 1990. He is currently professor of artificial intelligence at the University of Sussex, head of the Sussex Evolutionary and Adaptive Systems Group, and codirector of the Sussex Centre for Computational Neuroscience and Robotics. His current research interests include evolutionary robotics, biologically inspired neurocontrol systems, computational neuroscience, and evolutionary computing. *Address*: CCNR, University of Sussex, Brighton, BN1 9QH, UK. *E-mail*: philh@cogs.susx.ac.uk.

Andrew Philippides received an M.A. in mathematics from King's College, Cambridge, an M.Sc. in knowledge-based systems from the School of Cognitive and Computing Sciences at the University of Sussex, and a D.Phil. in modeling the diffusion of nitric oxide in brains from the School of Biological Sciences at Sussex. He is currently a research fellow at the Centre for Computational Neuroscience and Robotics at the University of Sussex. His research interests include computational neuroscience, evolutionary robotics, statistical pattern recognition and evolutionary computation. *Address*: CCNR, University of Sussex, Brighton, BN1 9QH, UK. *E-mail*: andrewop@cogs.susx.ac.uk.

Michael O'Shea received a first in biosciences (B.Sc.) from the University of Leicester in 1968 and a Ph.D. in neuroscience from the University of Southampton in 1971. He held a number of postdoctoral and faculty posts at the University of California, Berkeley; University of Cambridge; University of Southern California; University of Geneva, Switzerland. Since 1990 he has been professor of neuroscience at the University of Sussex, head of the Sussex Centre for Neuroscience, and codirector of the Sussex Centre for Computational Neuroscience and Robotics. His current research interests include the molecular mechanisms of learning and memory, neural signaling by diffusing gases such as NO, the evolution of noncoding genes, and biologically inspired evolutionary robotics. *Address*: CCNR, University of Sussex, Brighton, BN1 9QH, UK. *E-mail*: M.O-Shea@sussex.ac.uk.