Circle In The Round: State Space Attractors for Evolved Sighted Robots

Philip Husbands and Inman Harvey and Dave Cliff
School of Cognitive and Computing Sciences
University of Sussex, Brighton BN1 9QH, England
`email: philh or inmanh or davec@cogs.susx.ac.uk`

Abstract. This paper presents an analysis of an artificially evolved dynamical network-based control system for a simulated autonomous mobile robot engaged in simple visually guided tasks.
Keywords. Genetic algorithms, recurrent dynamical networks, visually guided behaviours, dynamical systems.

1 Introduction

After explaining our methodology for artificially evolving control systems for autonomous mobile robots, this paper presents and analyses recent results of experiments in concurrently evolving simulated robot control systems and sensor morphologies. The robots are engaged in simple navigation-based tasks in differing environments. Recurrent dynamical 'neural' networks make up the control system, and the primary sensory input is visual, via a pair of minimal bandwidth sensors. The structure and size of the networks are under evolutionary control as are properties of the visual sensors.

The work described here forms part of the early stages of a research program to thoroughly explore the potential of artificial evolution for developing autonomous agents. In order to progress most effectively, we think it is important to have a firm understanding of the evolutionary mechanisms and the systems they produce. General forms of analysis are necessary to throw light on, for example, the necessary conditions for the development of certain types of behaviours; whether or not there are underlying general behaviour-generating principles for classes of evolved agents[1]; whether or not different classes of artificial neurons result in significantly different evolutionary trajectories. They will also be needed to find out how robust any given evolved agent is, how general its behaviours are, and so on. Hence a major aim of this paper is to present some of our tools for analysis and to show how they can be used.

[1] A detailed elucidation of these may have interesting repercussions throughout AI and Cognitive Science.

Using a dynamical systems perspective, we give a thorough and general analysis of one of the successful evolved simulated robots. We show that it is robust in the face of noise and is highly fit over a large range of environments from a particular class. The controller is general: its dynamics are *not* the same in the different environments. The analysis of behaviour makes use of an appropriate state space. We are able to show that in some of these environments the entire state space is a single basin of attraction for a point attractor corresponding to a high scoring behaviour according to the task-based evaluation function used during evolution. The state spaces for the other environments have two attractors, both corresponding to highly fit behaviours. In other words, the robot has evolved to the point where it is 100% guaranteed to succeed at its given task – the visuo-motor couplings, via network dynamics and the visual structure of the robot's world, are in perfect harmony relative to the evaluation task. It is pointed out that these robots, being dynamical systems, use forms of animate vision.

Some of the problems of simulation work, particularly those involving vision, are briefly discussed. This leads into a description of ongoing work in which specialised visuo-robotic equipment is used to concurrently evolve visual morphologies and control networks *without* recourse to simulation of the agent environment coupling.

The structure of the paper is as follows. The early sections establish the scope of our work, justify the use of artificial evolution, explain its mechanisms, and discuss our reasons for basing the control systems on recurrent dynamical neural nets. The particular artificial neurons used in the work reported in later sections are then detailed. Next we move on to describe a series of experiments involving the concurrent evolution of control networks and visual morphologies. Techniques for analysing the evolved control systems are presented. These are then used to give a detailed analysis of one of the systems evolved in the experiments described earlier. There follows a discussion on how such analyses might be extended to tackle more complex cases. The paper ends with an introduction to current work in which artificial evolution is performed in the real world without the need for simulated sensing.

Lack of space means that a great many issues relating to the work are not dealt with or only very briefly covered. For further details, especially on methodological issues, see [22, 10, 16, 9, 23].

2 What sort of robots?

We are investigating mobile robots in environments with flat floors, where it is assumed that wheeled motion is relatively easy (subject to some slippage); and where there are obstacles and walls. So the focus is on navigation, using touch sensors and vision for sensory inputs, and motor outputs to wheels. One of the mobile robots built in our group is battery-powered, about 40 cms diameter and 30 cms high, running on left and right wheels with independent motors, and a third trailing castor for stability [31]. The sensors on this include the whiskers and bumpers as shown in the plan view of Figure 1. An onboard notebook 486 PC implements any desired control system. The signals to the motors can be represented as a real value in the range [-1.0,1.0]. This range is divided up into five more or less equal segments, and depending on which segment the signal falls into, the wheel will either remain stationary or rotate half/full speed forwards/backwards.

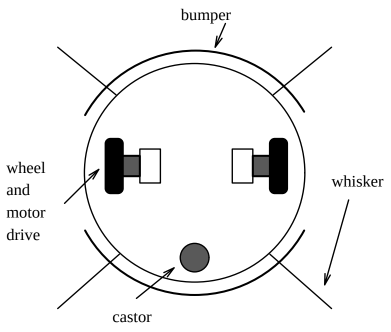

Figure 1: Plan view of the mobile robot.

Our simulation work has been based on this robot, and an earlier prototype, with the addition of vision. The simulations are at the physical level, with the characteristics of the motors, collisions with obstacles, slippage on the floor etc. modelled to include appropriate noise profiles. For vision, both in simulation and in reality, low bandwidth 'insect-like' eyes are assumed; in effect a small number of photoreceptors with genetically specified angle of acceptance and direction. The vision simulation uses ray-tracing with anti-aliasing via 16-fold super-sampling. Simulated vision rapidly becomes very expensive, and difficult to make accurate, as the required resolution increases and the environments become more complex. Towards the end of this paper we discuss our solution to this problem. However, the experiments described in this paper were of a minimalist enough nature that simulation was feasible.

The aim of the work described here was to explore some of the basic issues in artificial evolution, rather than to develop a practical controller for a particular robot. We regard realistic simulations as being suitable for these purposes. However, in other work we have used our evolutionary methodology to develop controllers for real robots [17, 24].

3 What model of cognition?

Inevitably anyone's approach to building autonomous robot control systems is determined, consciously or unconsciously, by their model of cognition. Our starting point is that we see a robot, and indeed any organism, as a physical dynamical system, with its own internal dynamics, coupled to a world (also a dynamical system) through sensory-inputs and motor-outputs. The sensory-inputs perturb the internal dynamics without constraining them [2, 32, 33]. We completely ignore the 'problem' of how internal models can accurately represent the 'real' world. We believe that such types of analysis, which may or may not be useful at the behavioural level of description, are positively harmful if taken uncritically as a blueprint for the construction of a physical dynamical system, a 'robot brain'.

A dynamical system is taken here to be any system that can be characterised by a finite number of state variables, and a dynamical law that specifies how those state variables change with time. Computers inhabit one very small corner of the space of all dynamical systems, and the all-too-common assumption that cognition is some form of computation is, we feel, a stultifying restriction on possible models for a robot control system, as well as in the rest of AI.

We are sympathetic to Brook's rejection of the dogma of functional decomposition [6, 7, 5]. However, we are sceptical of the alternative of behavioural decomposition advocated by Brooks, if it also is taken as a dogma (not that we are suggesting that he does). The methodology we advocate does not rely *a priori* on any form of decomposition, although control systems thus produced are open to analysis *a posteriori*; which may reveal in practice that some sort of decomposition is then possible or useful.

4 Why use evolution?

Brooks' approach very sensibly advocates an incremental approach to building up more sophisticated behaviours 'on top of' previous simpler ones. Each layer of behaviour is implemented from sensors through to motor-output largely independently of the other layers, the interactions between such layers being restricted to suppression and inhibition of lower layers. Through this method, the decomposition by activity of the behaviour is converted into decomposition by layer of the behaviour-producing mechanism. Each layer should be thoroughly debugged before adding the next layer, and the restrictions on interactions between the layers aids the designer's task.

Nevertheless, even with such restrictions, the addition of a new layer of mechanism has repercussions over and above any directly wired in links of suppression or inhibition — repercussions arising from the fact that the robot is coupled with its world. As the number of layers increases, the number of such repercussions, intended or unanticipated, can scale up horrendously; as well as first-order repercussions, these in turn can stimulate higher-order consequences. In design, the purpose of modularity, or of decomposition of any kind, is to help the human designer minimise the unanticipated, but we believe that the practical limits of this approach are already being reached

[22].

Evolution in the natural world does not suffer the same limitations that a human designer toils under, and each of us is an existence proof of the ability of evolution, given appropriate resources, to produce sophisticated control systems. Artificial evolution (subject to some caveats) allows us to specify *what* behaviour we desire without specifying *how* this should be achieved. Hence, as further discussed in [16, 9], we advocate artificial evolution for the development of control systems for autonomous mobile robots. Here sensors are taken to be an essential and integrated part of the control system. Hence they should be subject to evolutionary alteration along with the rest of the system. Ideally so should motor properties. However, the genetic specification of motor properties and general robot morphologies are currently beyond the scope of our work.

5 What is artificial evolution?

Artificial evolution involves the application of Genetic Algorithms (GAs) [14, 20], which are algorithms using a few ideas borrowed from natural evolution, and primarily used for function optimisation in highly complex domains where analytic solutions are not possible [27, 4, 13]. Standard GAs can be considered as search techniques operating in pre-defined search spaces, using populations of trial points in the spaces to guide where the next generation of such trial points should search.

A mapping is defined from a string of characters (the analogy with the genotype of an organism) to a trial point in the search space which can be evaluated (the 'phenotype'). One of the caveats alluded to above is that this mapping should be such that, in some sense, similar genotypes are associated with phenotypes that are also in general similar. In the present context, a mapping should be chosen such that any string of characters from some alphabet, of a predefined length, can be interpreted as specifying the control system for a robot.

In a standard GA[2], an initial population (perhaps of size 100) of randomly created genotypes would specify the initial population of control systems. One at a time, each of these is evaluated on the given task(s) to be tackled in the given environment; the result of evaluation is a real number. This number is translated into a measure of fitness. In an initial randomly generated population, of course, most will be useless; but any that score higher than average are given a higher than average chance of contributing genetic material to the next generation.

This is done by creating a reproductive pool of genotypes biased towards fitter members of the population. From this pool of 'parents' genotypes are

[2] There are a number of variations on this basic algorithm, some involving asynchronous parallel distributed processes, but in all the basic elements of selection, reproduction, recombination and mutation are still at the core [21].

taken in pairs; a randomly chosen crossover point is taken, and an offspring genotype is created by taking part, as defined by the crossover, from one of the parents, and the remainder from the other parent. A mutation genetic operator, applied at a given small rate of probability, mutates some of the characters in the genotype. The offspring thus created replace members of the previous generation and the evaluate-breed-replace cycle continues.

This process continued through successive generations, with the aim of seeing fitness rise over time. Although individual elements of the algorithm are random, the result is *not* random search, but something much more powerful. It can be shown [14, 20] that the algorithm exploits useful building blocks or schemata — roughly speaking, any fairly short lengths of a genotype which may 'code for' some useful part of the phenotype. In a population of size n, the number of such schemata that are usefully processed each generation is $\mathcal{O}(n^3)$.

6 Is our GA different?

GAs are usually used for optimisation, but evolution in the natural world is not optimisation in a pre-defined search space. There was no 'problem' posed several billion years ago for which present-day animals are some sort of 'solution'. The world that we live in was itself largely formed in co-evolution with our ancestors. What has happened in evolution, which is not normally permitted in standard GAs, is the development over the long term of more complex structures (associated in general with longer genotypes) from simpler ones. The notion of a pre-defined search space, whose dimensionality is associated with a fixed genotype length, becomes less useful when genotype lengths are allowed to increase to any arbitrary length.

A single well-defined task, or set of tasks, for a robot control system to tackle, can be taken as defining a restricted search space. Standard GAs are one possible tool for optimising a control system for the given problem. But looking ahead to the need to add new tasks for robots to do that were not originally anticipated — and on the inevitable demands for incremental adaptation of pre-existing systems — it follows that we need to be able to do evolution rather than optimisation. The SAGA framework [18] has been developed to deal with the rather different dynamics of GAs when genotype lengths must be allowed to increase to arbitrary lengths through evolution.

It turns out that when this happens, populations are inevitably largely genetically converged, as a *species* — SAGA stands for Species Adaptation Genetic Algorithms. The genetic operators must allow genotype lengths to change from one generation to the next, but it can be demonstrated that any such increases in genotype length will be restricted to gradual small steps[3]. Selection needs to be rank-based, and generally should be signif-

[3]What counts here as a small step is dependent on the ruggedness of the particular fitness landscape in genotype space. It is associated with the correlation length of such a

icantly stronger than in a standard GA (where people are usually trying to avoid premature genetic convergence; in SAGA, the population is always converged). Whereas in standard GAs, recombination is taken to be the dominant genetic operator, with mutation relegated to a background operator for ergodicity, in SAGA mutation is a much more significant force, and is applied at a higher rate, of the order of one mutation per genotype [18, 19, 15].

7 What building blocks for a control system?

We are relying on evolution for the design of a control system, but we must choose appropriate building blocks for it to work with (this is another of the caveats mentioned earlier). Some have advocated production rules [12] (Classifier Systems [14] are the GA version of these). Some propose Lisp-like programming languages [26]. Brooks [8] has proposed using Koza's ideas applied to a high-level behaviour language. Beer [3, 34] has used dynamical neural networks. It is only this last approach that we are in broad agreement with.

Our intuitions, supported by our simulation results and recent work with real robots [17], are that the primitives manipulated by the evolutionary process should be at a rather low-level. Any high level semantic groupings inevitably incorporate the human designer's prejudices, and will probably give rise to a more coarse-grained fitness landscape with more steep precipices. But evolution requires that the fitness landscape is in general not too rugged. It might be thought that the use of low-level primitives necessitates aeons of trials before any interesting high-level behaviour emerges, but our experience indicates otherwise.

Another factor concerning high-level languages is that the injection of noise into a system seems contrary to the rationale for such languages. The injection of noise into the lowest levels of a control system can be shown to have valuable effects on the dynamics; and has the additional benefit of blurring the fitness landscape and making it less rugged for evolution.

Our criteria for deciding on component primitives are:

- They are the primitives of a dynamical system.
- The system should operate in real time, and the timescales on which the components work must be in some sense appropriate for the world the robot inhabits.
- The system should be 'evolvable', not 'brittle'; in the sense that many of the possible small changes in the way components are bolted together should result in only small changes in resulting behaviour.

landscape [25].

- Incremental change in the complexity of any structure composed of such primitives should be possible.

There may be many possible components and general architectures that meet these criteria. The particular choice we have focused on is that of recurrent dynamic realtime networks, where the primitives are the nodes in a network, and links between them. The nodes act much as many artificial 'neurons' in a neural network, though with particular characteristics. The links are unidirectional, have time delays between units, and may be weighted. There are no restrictions on network topologies, arbitrarily recurrent nets being allowed. When some of these nodes are connected to sensors, and some to actuators, the network acts as a control system, generating behaviours in the robot.

We are exploring a whole class of networks with these general properties. The next section details the particular type of net used in the experiments described later. It was chosen as a simple example of the sort of net we are interested in.

7.1 Artificial neuron mechanism

The artificial neuron mechanism employed to date has been designed for its usefulness in control applications rather than for biological plausibility or ease of analysis. Figure 2 represents the operation of a neuron. There are separate channels for excitation and inhibition. Real values in the range [0,1] propagate along excitatory links subject to delays associated with the links. The inhibitory (or veto) channel mechanism works as follows. If the sum of excitatory inputs exceeds a threshold, T_v, the value 1.0 is propagated along any inhibitory output links the unit may have, otherwise a value of 0.0 is propagated. Veto links also have associated delays. Any unit that receives a non zero inhibitory input has its excitatory output reduced to zero (i.e. is vetoed). Note that this means that a unit may be vetoed but still able to produce inhibitory output, as long as the sum of its excitatory inputs are high enough. In the absence of inhibitory input, excitatory outputs are produced by summing all excitatory inputs, adding a quantity of noise, and passing the resulting sum through a simple linear threshold function, $F(x)$, given in Equation 1 below. Noise was added to provide further potentially interesting and useful dynamics and to give an indication of the properties of a physical implementation of a unit which would be likely to include naturally occurring noise. The noise was uniformly distributed in the real range [-N,+N] (see Section 8 for further discussion of the role of noise). In the work described here all connections had unit weights.

$$F(x) = \begin{cases} 0, & \text{if } x \leq T_1 \\ \frac{x-T_1}{T_2-T_1}, & \text{if } T_1 < x < T_2 \\ 1, & \text{if } x \geq T_2. \end{cases} \tag{1}$$

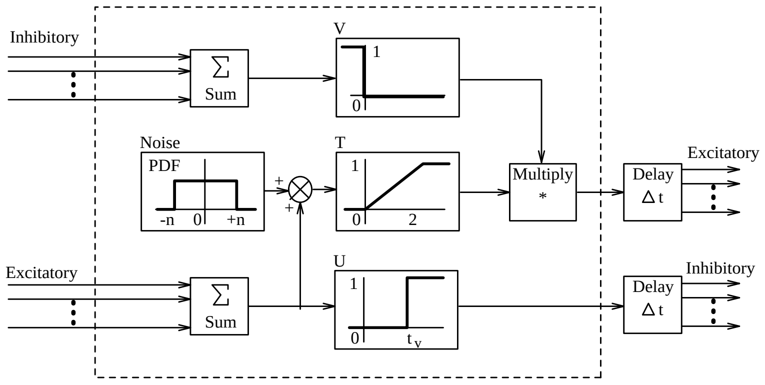

Figure 2: Block diagram of neuron operation.

In all our work to date the networks have been simulated in software. Their continuous nature is modelled by using a very fine time slice approach. At each time step of the robot kinematics simulation, sensor readings are fed into the network, the network is run through a number of cycles (with a variance to counter distorting periodic effects), and then the network outputs are converted into motor signals which allows the calculation of the new position of the robot. Time delays on the connections refer to network cycles, hence a unit delay is much smaller than the duration of a single robot step. In the work described in this paper unit delays are used throughout. The values of the other parameters mentioned above were set at N=0.1, T_v=0.75, T_1=0.0 and T_2=2.0, giving a slope of 0.5 to F.

7.2 Evolving the networks

In the experiments analysed later, a fixed number of nodes are identified as input nodes. These receive input directly from the sensors. There is one such node for each sensor. There are also a fixed number of output nodes, two for each motor. The outputs of each motor neuron pair are differenced to give the full motor signal range of [-1.0,1.0]. These signals are passed on to the relevant motors. There are an arbitrary number of internal nodes which fall into neither of the previous two classifications. The genetic encoding specifies the number of internal nodes, individual neuron properties, and the how the neurons are linked up. The size and topology of the network is unrestricted. The genotypes are strings of characters interpreted in a sequential manner from left to right. The encoding has been designed so that genetic operations (crossover and mutation) on 'parents' always produce 'children' coding for legal networks (see [16, 9] for full details).

As mentioned earlier, in much of our work to date, the weights and delays on network connections are set to a unit value, and the value of this delay implicitly determines the timescale on which the network operates. We are now moving towards having delays of widely ranging values. Learning (to use a behavioural level of description) can be considered as no more than the consequence of internal dynamics at different timescales; changing of weights in a network is just one of the ways in which this can be achieved.

8 Why do we use noise?

As will be shown later, the internal noise in the networks significantly alters their dynamics. In addition this helps to make the control system more evolvable and less brittle. Incidentally, it underlines the point that it is not very useful to try and interpret the networks as 'computing' outputs from inputs; rather, the network is a particular dynamical system perturbed by the inputs.

In addition, in our simulations at the physical level a significant amount of noise is deliberately added to sensor inputs, motor outputs, and movement in the world such as that arising from collisions. The noise profiles are based on measurements of the real system. Each robot is scored over a number of trials, and the *worst* of these scores is used as the final evaluation. In this way we wish to encourage robustness, and on transfer from simulation to the real world our goal is that the 'real' values of parameters should lie within the 'envelope of noise' we have created around the parameter values used in simulation. For details of recent work in which this transfer is successfully achieved see [24].

9 Experimental results: vision in an arena

Earlier results in simulation using navigation without vision in a cluttered environment, and with vision (two photoreceptors) in a single fixed circular arena, have been reported elsewhere [10, 9, 16]. This paper will report on new results using vision in differing circular arenas.

The arena has a black wall, and a white floor and ceiling. As well as the tactile sensors shown in Figure 1, the simulated robot is provided with two photoreceptors. These are placed at equal angles (the angle of eccentricity) either side of the the robot's forward facing midline. They each have the same angle of acceptance which specifies their cone of vision. Angle of eccentricity and angle of acceptance are illustrated in Figure 3. The simulated sensors produce a real number in the range [0.0,1.0] proportional to the brightness level in their field of view. As well as the control network size and topology, the photoreceptors' angles of eccentricity and acceptance are also under genetic control. The angle of eccentricity can vary between 0 and $\frac{\pi}{2}$ and the angle of acceptance between 0 and π. In this way the visual morphology is concurrently evolved along with the control network. The visual structure of

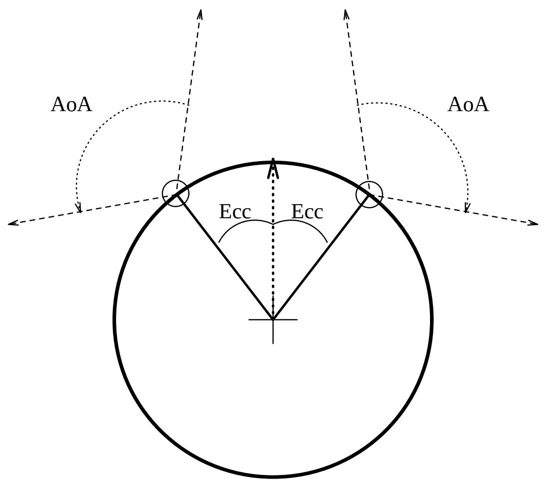

Figure 3: Angle of acceptance and eccentricity for the two-photoreceptor robot. A top-down view of the robot, and the relevant angles.

the robot's world varies markedly over the range of values of these two angles (see [9] for details).

The larger part of the genotype encodes a network, as described earlier. A genetic algorithm using SAGA principles was then used, with populations of size 60.

The evaluation of each robot involved decoding the genotype for the control network and visual morphology. The resulting robot was then given 8 trials. At the start of each trial it was placed in a random orientation and position (this position being restricted to a band near the wall). A fixed length of time was allowed (100 simulation timesteps), during which it was evaluated according to:

$$\mathcal{E} = \sum_{\forall t} \exp(-s \mid \mathbf{r}(t)|^2)$$

where $\mathbf{r}(t)$ is the 2-D vector from the centre of the arena to the robot's position at time t, and $\forall t$ denotes the entire (finite) lifetime of the robot. The sum is over the time steps of the simulation. The parameter s in this Gaussian is chosen so as to ensure that the robots collect virtually no score near the edge of the arena, and hence the evaluation function $\mathcal{E}$ implicitly sets the goal of: reach the centre of the arena as soon as possible and stay there. At the end of the 8 trials the *worst* (i.e. lowest) value of $\mathcal{E}$ is assigned as the fitness of the robot: this encourages robustness.

In the first set of experiments the cylinder radius was fixed at 20 units and the wall height at 15 units. The robot radius was 2 units. Success on this task was reached on a number of runs, within 100 generations each time. Two different successful control systems, termed C1 and C2, have been described elsewhere [11]. In both cases, the robots make a smooth approach towards the center of the arena, and then circle there, either on the spot or in a minimum radius circle. A typical behaviour for C1 is shown in Figure 4 and the network that generated it is illustrated in Figure 5. Compare these with the behaviours and networks of C2 given later; they are quite different in many respects but share some basic underlying principles revealed by the kind of deeper analysis given later. Before analysing the C1 and C2 results fully, we had speculated that the control system might be recognising that the centre of the arena had been reached by monitoring the absolute values of the photoreceptor inputs: at the centre of the world, the photoreceptor inputs would take on an absolute value determined by the wall-height:floor-radius ratio of the particular circular arena used during evolution.

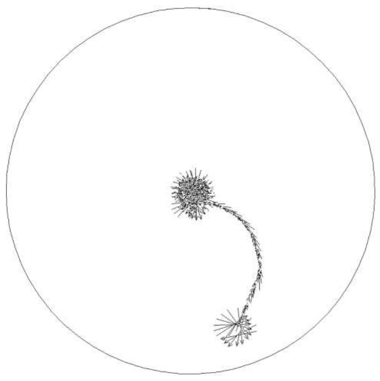

Figure 4: *Typical path of a successfully evolved robot, which heads fairly directly for the centre of the room and circles there, using input from 2 photoreceptors. The direction the robot is facing is indicated by arrows for each time step, largely superimposed.*

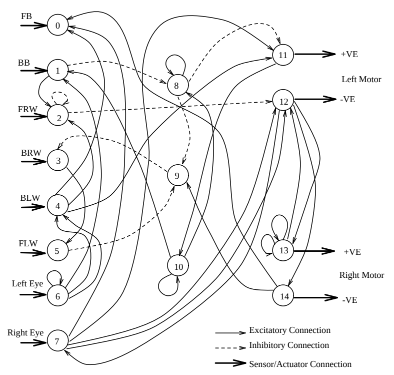

Figure 5: *The network diagram for the robot that produces the behaviour shown in the previous figure.*

With this in mind, we set up a more difficult task, where the height of the wall could vary over one order of magnitude, from 4 to 40 units; the diameter of the arena remaining at 40. The full range of possible heights was divided into 10 equal sections, and each robot was given 10 trials, with a height of wall chosen at random from each in turn of the 10 sections. In this way it could be ensured that it was tested across the full range. Thus no use could be made of absolute light values; as is usual in our experiments, the evaluation of the robot was based on the *worst* score it obtained across its trials.

Once again, on several runs success was reached within 100 generations. Furthermore, on analysing the successful robots it was found that they had evolved with either both photoreceptors pointing straight ahead, or at the minimum angle apart such that their cones of acceptance largely overlapped. In other words, for all practical purposes they had thrown one photoreceptor away and were succeeding with a single receptor of one pixel.

For the purposes of comparison, some robots that had earlier been successfully evolved for a single fixed wall-height, were tested across the full range of wall-heights used in the second experiment. Much to our surprise, although many failed, C2 succeeded, and in fact did better than any of those evolved under the varying conditions. On analysis, there were certain similarities among all the successful robots, including C2.

Whereas C2 had its photoreceptors, with a 45^o angle of acceptance, placed at 60^o to each side of the robot's centre line, a detailed analysis shows that it only ever makes use of a single receptor at any given time, although whether it is using the left or the right one varies with the immediate environment. Hence C2 is effectively monocular, although in a more subtle way than the others. Figure 6 shows the average performance of C2 over the range of wall heights. The average of 100 runs at each height are shown. The theoretical optimum score is between 75 and 80. But because the whole system is shot through with noise and the Gaussian tapers off very sharply from the centre, any score over 60 is highly fit, meaning the robot spent the vast majority of its lifetime very close to the centre.

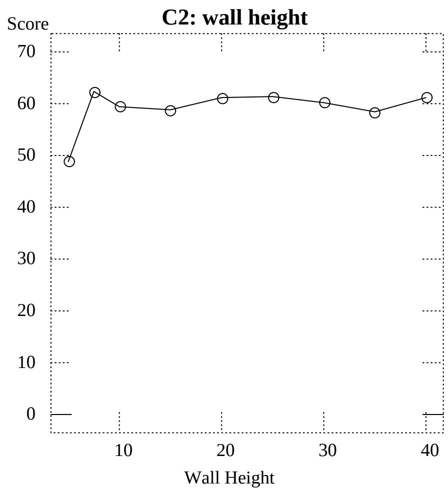

Figure 6: C2 evaluation scores over different wall heights.

Space does not allow a full analysis of more than one network, so because it has the largest number of interesting properties, many of them shared with one or more of the other successful controllers, and illustrates the analysis techniques particularly well, C2 has been chosen. Before it can be analysed in detail, the techniques used must be introduced.

10 Analysis

In the following sections we show that the evolved controllers can be thoroughly understood and we explain how they provide general solutions. The first two subsections deal with the main techniques used in the later full analysis of C2. These are an analysis of the control networks based around its feedback loops, and the use of a low dimensional state space for visualising how the robot behaviours vary with the visual signals being received. The crucial idea is that to fully understand how the evolved control system works, it is necessary to analyse in a general way the interplay between (i) the visual structure of the robot's world determined by its evolved visual morphology and (ii) the control network dynamics. This can be done if the visual structure of the robot's world can be mapped out, and it is understood how the motor outputs vary for all the possible left photoreceptor/right photoreceptor visual signal combinations in the world, while taking into account the internal network dynamics. With this information it should be feasible to construct vector flow fields, or phase portraits, showing how the robot behaviour varies across its world.

In the following analysis phase portraits are constructed in this way for the particular networks and class of worlds used in the experiments described earlier. The details would need to be a little different for other experimental setups, and the resulting state spaces may not be so easy to visualise, but the general idea should carry through. These issues are discussed more fully in Section 14.

11 Network analysis

Before going on to analyse particular controller networks, a commonly occurring sub-network configuration will be looked at in detail. This will enable a deeper understanding of the full networks described later. The details of the analysis are peculiar to the networks used in this work, but the general method could be used with many other types of noisy net. The use of more analytically contrived networks would probably entail a different, and presumably cleaner, analysis.

Nearly all the successful evolved networks we have studied involve one or more neurons with excitatory feedback loops. This arrangement often allows internal noise to rapidly build up the output from the unit to the upper threshold level without requiring any external input to the network. The unit then acts as a source of constant high output (unless vetoed) which typically contributes to the motor signals giving the basic behaviour of the robot. We will refer to such neurons as *generator units.* Their properties will fall out of the following general analysis of simple feedback loops. An understanding of such sub-networks can greatly simplify the analysis of full evolved controller networks. Their properties are not entirely obvious since internal noise is uniformly distributed with mean zero. In the following all connections have unit time delays, which is the case for all the experiments described in this paper.

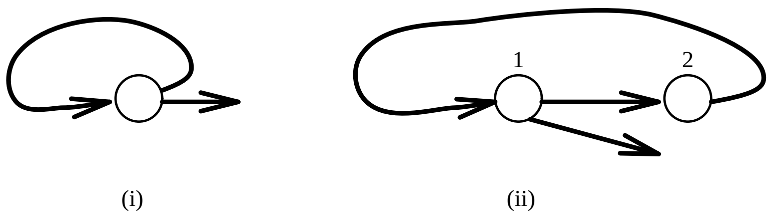

Figure 7: Excitatory feedback loops.

Consider the single neuron feeding back onto itself as shown in Figure 7(i). The input at time t, I_t, is related to the output at time $t-1$, O_{t-1}, by $I_t = F(O_{t-1} + n_t)$, where F is the transfer function of Equation 1 and n_t is the noise injected by the unit at time t. Ignoring the threshold terms for the moment, and concentrating on the linear part of the function, we can write:

$$O_t = k(O_{t-1} + n_t) \tag{2}$$

where k is the slope of the linear part of F. Hence,

$$\begin{aligned} O_0 &= kn_0 && (3)\\ O_1 &= k(kn_0 + n_1) && (4)\\ \cdots O_t &= k^{t+1}n_0 + k^t n_1 + \cdots + kn_t && (5)\\ &= \sum_{i=0}^{t} k^{t+1-i} n_i = \sum_{i=0}^{t} a_i(t) && (6) \end{aligned}$$

where $a_i(t)$ is a substituted variable. Now the noise terms, n_i, can be considered as independent random variables uniformly distributed over the real range [-N,+N], giving them probability distribution function $P(x) = \frac{1}{2N}$. Using elementary probability theory, it can be shown that they each have mean $\mu_{n_i} = 0$ and variance $\sigma^2_{n_i} = N^2/3$. The linear relationships between the $a_i(t)$s and n_is (for any fixed t and i) means that the $a_i(t)$s can be regarded as independent random variables with mean $\mu_{a_i(t)} = 0$ and variance $\sigma^2_{a_i(t)} = k^{t+1-i}N^2/3$. In other words the output of the unit can be regarded as the sum of a series of independent random variables. To return to the temporarily ignored threshold conditions of F, because O_t can never become negative (lower threshold condition), we can use a version of the central limit theorem to guarantee that the distribution of the random variable Z given below,

$$Z = \frac{O_t - \sum_{i=0}^{t} \mu_{a_i(t)}}{\sqrt{\sum_{i=0}^{t} \sigma^2_{a_i(t)}}} \tag{7}$$

is closely approximated by the *positive* half of the standardized normal distribution $\mathcal{N}(0,1)$. Now, $\sum_{i=0}^{t} \mu_{a_i(t)} = 0$ and $\sum_{i=0}^{t} \sigma^2_{a_i(t)} = \sum_{i=0}^{t} k^{t+1-i} \frac{N^2}{3} = \frac{N^2}{3} \times \frac{k-k^{t+2}}{1-k}$, hence:

$$Z = \frac{O_t}{\frac{N}{\sqrt{3}} \times \sqrt{\frac{k-k^{t+2}}{1-k}}} \quad ,(k \neq 1) \tag{8}$$

Clearly the properties of the feedback loop depend crucially on the value of k. Given that we have used a fixed slope of 0.5 for the neuron transfer function, k can be *effectively* altered by changing the number of feedback loops. This is because the transfer function involves a summation of inputs. With two loops $k = 1$, and $O_t = \sum_{i=0}^{t} n_t$. Because of the threshold conditions on the neurons, this means the output will follow a drunkard's walk between 0 and 1. With one loop $k = 0.5$ and, given that the maximum possible value for n_t is 0.1, the higher order terms become vanishingly small. Hence the output can be well approximated by:

$$O_t = k^3 n_{t-2} + k^2 n_{t-1} + k n_t \tag{9}$$

The expression for Z given in Equation 8 can be used to estimate the expected time for O_t to reach the upper threshold value of 1.0. Clearly this depends on the value of k, or at least the number of feedback loops as explained above. The larger k, the shorter the time. From Equation 2 and using the fact that the maximum negative amount of noise is -N, once the upper threshold has been reached it can only be maintained if $k(1-N) > 1$, i.e. $k > 1/(1-N)$. Setting N=0.1, the value used in all our experiments to date, then $k > 1.11$. With three loops $k = 1.5$, and the probability of O_t reaching 1.0 by $t = 40$ is given by $2(1 - \Phi(0.0025))$, from the formula for Z, and where Φ is the cumulative distribution function of $\mathcal{N}(0,1)$. The value of this expression is greater than 0.999.

Hence we can conclude that with three feedback loops, and in the absence of other inputs, a neuron will easily reach the saturation condition within 40 network cycles, the minimum possible per *single* robot step, and therefore will act as a generator unit. With two feedback loops it will act as a noisy generator unit with its output performing a random walk between 0 and 1. This is exactly what is observed. In other words, these sorts of circuits rapidly settle down into stable dynamical regimes, although those regimes may involve stochastic elements. Since during this (extremely brief) settling down period changes in visual signals are ignored and motor outputs are not altered, networks built around circuits of this kind are amenable to the particular style of analysis developed over the following sections. This point will be covered in more detail later, where it will be seen to be crucial in allowing the use of the vector flow fields introduced in Section 13.2.

Surprisingly, the two unit mutual feedback loop shown in Figure 7(ii) has properties equivalent to the single unit loop. Using the same notation as above, $O_{1_t} = k(O_{2_{t-1}} + n_{1_t})$, with a symmetric expression holding for O_{2_t}. It is straightforward to show that,

$$O_{1_t} = k^{t+1}\eta_{1_0} + k^t\eta_{2_1} + \cdots + k^2\eta_{2_{t-1}} + k\eta_{1_t} \tag{10}$$

where,

$$\begin{aligned} \eta_t &= n_{1_t} \quad ,\text{even } t & (11)\\ &= n_{2_t} \quad ,\text{odd } t & (12) \end{aligned}$$

Since n_1 and n_2 have exactly the same distribution this expression is identical to that in Equation 6 and the same analysis holds. Indeed, the interplay between the delays on the connections and the noise injected at each unit, means that a loop containing any number of units will have the same behaviour; any of its neurons can act as generator units.

Any external inputs to loops can be treated in a simple additive fashion, with the same sort of analysis as above holding. Specifically, if a unit has external input I_t and a single feedback loop,

$$O_t = \sum_{i=0}^{t} k^{t+1-i} n_i + \sum_{i=0}^{t} k^{t+1-i} I_i \tag{13}$$

When the loop has $k = 0.5$, this can be approximated by:

$$O_t = k^3(n_{t-2} + I_{t-2}) + k^2(n_{t-1} + I_{t-1}) + k(n_t + I_t) \tag{14}$$

If the loop from a unit involves m additional units the external input will be attenuated by k^m (assuming $k < 1$) before it feeds back into the unit. Because of the rapid settling down period of these circuits (over which time the visual inputs can be regarded as unchanging), visual inputs to networks built around these types of feedback loops can be regarded as perturbing the network dynamics in a straightforward manner.

Results from this section will be used later in analysing evolved control networks, where we will see that characterising a network in terms of its feedback loops can be a very useful tool in determining its stable state signal levels.

12 $r\phi$ Space

In order to understand the generation of the robot's behaviour, and in particular how the evolved network/visual sensor system is adapted to its simple environment, a global view of the visual structure of the (robot's) world would be useful. Coupled with an understanding of the control network's dynamics, particularly how motor signals change with visual inputs, in principle this global view would allow the construction of a vector flow field indicating the robot motion at each point in the world. How feasible this is for any given control system will depend largely on how noisy the network dynamics are. However, by definition, the dynamics of the most successful visually guided systems cannot be too noisy; otherwise there would be no correlations between visual signals and behaviours. Another factor in such an analysis is the dimensionality of the state space used. We can only usefully visualise low dimensional spaces. It turns out that the networks of interest here have properties which allow the use of a low dimensional state space and so analysis was feasible.

The uniformly lit cylindrical arena, with its uniformly coloured walls, floor and ceiling, affords some useful symmetries that can be exploited in such an analysis and visualisation of the robot control system dynamics. Cylindrical symmetry means that, for any given visual morphology, the visual inputs[4] are determined by the robot's distance from the centre of the arena (r) and the angle it makes with the arena radius passing through its centre (ϕ – in the following it is defined as the clockwise angle between the arena radius and the robot orientation arrow). This is illustrated in Figure 8: in terms of visual signals, situation A is identical to situation B. Hence the visual structure of the world for a particular robot can be mapped out by sampling visual signals for a set of values of ϕ between 0 and 2π at each of a set of distances (r) along a radius from the arena centre to the edge. Such a space can be used to individually map out signal levels for the left and right photoreceptors, or the combination of these two. The most global view is gained from mapping out the regions with different (left photoreceptor signal, right photoreceptor signal) combinations.

The later analyses using $r\phi$ spaces will be more easily understood after a brief examination of some of the geometric properties of this space. Three types of motion used in later analyses are covered. Note that in the ϕ dimension the 2π boundary wraps round to the 0 boundary, giving a cylindrical space.

- *Straight line.* Straight line motion in cartesian space does not map onto a straight line in $r\phi$ space. From Figure 9 it is readily seen that moving in a straight line backwards forces the following relationship:

$$rsin(\phi) = k \tag{15}$$

  When the robot starts from a position with $r\phi$ coordinates (r_0, ϕ_0),

$$k = r_0 \sin(\phi_0) \tag{16}$$

- *Rotation about right wheel.* Figure 10 illustrates a counter-clockwise rotation about the right wheel. Using the cosine rule, it can be seen

[4]Noisy sensors mean that average visual signals are being referred to.

that the following $r\phi$ equations hold:

$$\begin{aligned} k^2 &= r^2 + rad^2 - 2\cos(\pi/2 - \phi) && (17) \\ &= r^2 + rad^2 - 2\sin(\phi) && (18) \end{aligned}$$

Hence,

$$r^2 - 2\sin(\phi) = k^2 - rad^2 = K \qquad (19)$$

When the robot starts from a position with $r\phi$ coordinates (r_0, ϕ_0), and it has wheel base radius rad units,

$$K = r_0^2 - 2\sin(\phi_0) \qquad (20)$$

- *Rotation about left wheel.* Similarly, for a clockwise rotation about the left wheel it can be shown that:

$$r^2 + 2\sin(\phi) = Q \qquad (21)$$

where Q is a constant.

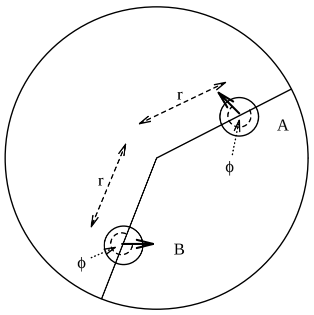

Figure 8: Illustration of r and ϕ parameters. Robots at A and B, oriented along the thick arrows, have identical values for r and ϕ.

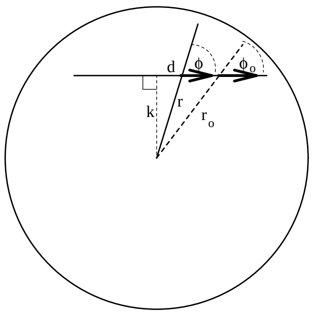

Figure 9: Relation between r and ϕ for straight line motion.

13 C2 analysis

13.1 The network

The full C2 controller network is shown in Figure 11, its evolved visual morphology is: angle of eccentricity= 60^0,angle of acceptance= 45^0. This provides it with two fairly wide-angle photoreceptors with a large angular separation. A typical behaviour generated by C2, for arena wall height 15, is

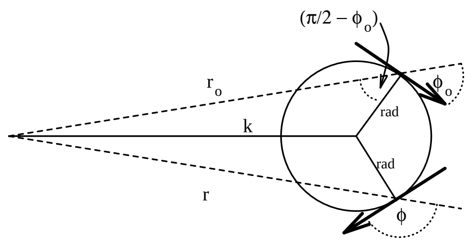

Figure 10: Relation between r and ϕ for rotation about right wheel.

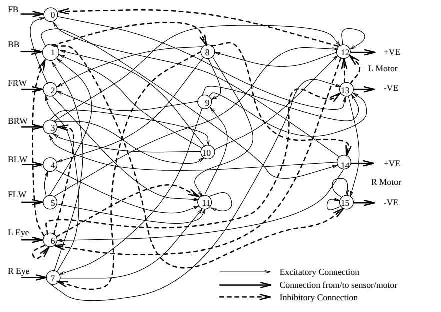

Figure 11: Full C2 control network. The left-hand column are units originally designated as input units: FB=Front Bumper; BB=Back Bumper; FRW=Front Right Whisker; BRW=Back Right Whisker; BLW=Back Left Whisker; FLW=Front Left Whisker. Right-hand column shows output units, which are paired and differenced to give two motor signals in the range [-1,1] from four 'neuron' outputs in the range [0,1]. Centre column shows 'hidden units'.

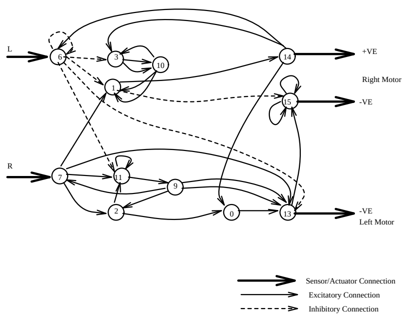

Figure 12: C2 visual guidance pathways. Note that, for the sake of clarity, the positions of the left and right motor outputs have been interchanged.

shown in Figure 13. As we shall see, at very low wall heights the behaviour is very similar except for the direction of rotation about the centre.

We will start our analysis of the network/visual sensor system by investigating some of the basic properties of the network. In this analysis we are interested only in the visually guided aspects of C2; unless it is started up against the wall, the tactile sensors are never active — it has evolved to make very good use of vision in successfully accomplishing its implicit task. Units with no outputs can be immediately eliminated. By studying time plots of neuron activities, motor outputs and sensor inputs, as shown in Figure 14, over a range of environmental conditions, it may be possible to highlight neurons that play no part in visually guided behaviours. In this case a small number of neurons can be eliminated. We can usefully redraw the slightly reduced C2 network to emphasise the visual pathways, as shown

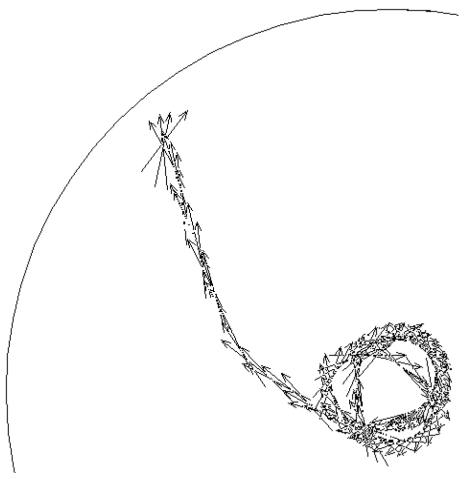
Figure 13: Typical behaviour of the C2 controller, with noise. The robot starts near the edge of the arena, moves to the centre, and then rotates around the right wheel. As can be seen, the C2 controller drives the robot in reverse (backwards).

in Figure 12. Note that the tactile sensor input neurons have been taken over as internal units in the evolved network. Only those neurons referred to in the main argument have been included in the figure. The other neurons have no significant role in behaviour generation.

The first thing to note is that unit 6 (left photoreceptor input neuron) has only inhibitory outputs. These will only be active if its inputs sum to greater than T_v (0.75). Note that one of its inhibitory links goes to the left motor's negative input neuron (unit 13). Given all the excitatory links feeding into 13 from unit 7 (right photoreceptor input neuron), clearly, if active, the inhibitory link from 6 to 13 will have a major effect on the motor outputs. So, understanding the conditions under which the inhibitory links from 6 become active is one of the keys to understanding the behaviour generating mechanisms of the network.

Another important aspect of the architecture is the fact that unit 1 (with direct input from unit 7) has an inhibitory link to the right motor's negative input neuron (unit 15). Since 15 doubly feeds back onto itself it will act as a noisy generator node unless vetoed. Hence, understanding the conditions under which unit 1 produces inhibitory output is also important in elucidating the network operation. This will entail unravelling the series of interconnected feedback loops involving unit 7.

We will start by understanding the characteristics of unit 7's output. It has visual input from the environment (here referred to as E_r) and excitatory input from 9. If the inhibitory links from 6 are active it is clear that all the right photoreceptor visual pathways are rendered inactive, so in the following it is assumed that 6 is not vetoing. The tangle of network surrounding unit 9

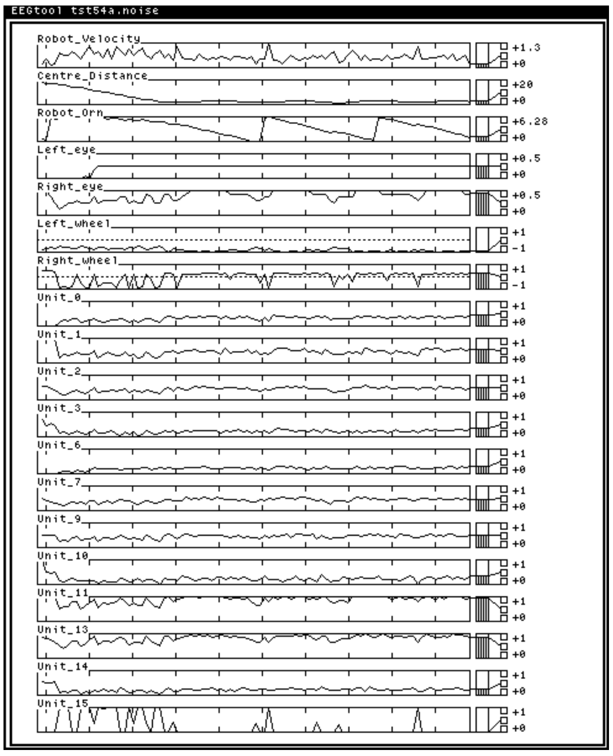

Figure 14: Record of observables and activity levels for the (with-noise) activity illustrated in Figure 13.

is probably best understood by concentrating on unit 11. This unit is involved in two feedback loops sharing no common units other than itself; 11-9-2-11 and the direct feedback loop. From the arguments of Section 11, these two loops are enough to ensure that this neuron acts as a noisy generator unit with output varying between 0 and 1. In addition it has external visual input from 7 (direct and via 2). The effect of this input can be approximated by thinking of 11 as having a single direct feedback loop and two external inputs of $\frac{E_r}{2}$ and $\frac{E_r}{4}$. Approximating the feedback loop with $k^3 + k^2 + k$ (from Equation 9), this gives an expected output of about $\frac{3E_r}{4}$. Given the additional loops 11 is involved in, and the attenuation argument of section 11.1, its visually-based output, in the stable state, should be approximately E_r. Adding this to its noisy generator behaviour, 11's output should vary approximately between E_r and 1. Unit 9's output will be about half this, hence 7's output will be approximately:

$$O_7 = \frac{3E_r}{4} + \frac{n_7}{4} \tag{22}$$

Where n_7 varies randomly between 0 and 1.

Returning to unit 1, we now understand its inputs from 7 and therefore shift the focus to unit 10. Unit 10 is involved in two feedback loops, 10-3-10 and 10-1-14-3-10. Since these share a common node other than 10, namely 3, this is *not* the situation of Equation 8 with $k = 1$. Rather, we have the case of one loop feeding into another giving an output close to $2\sum_{i=0}^{t} k^{t+1-i} n_i$ with $k = 0.5$ (from Equation 6). Using Equation 9, the value of this expression varies randomly between 0 and approximately 0.2. Unit 10 also has input from 7 via 14 and 3. Hence 10's output is approximately given by,

$$O_{10} = n_{10} + \frac{O_7}{4} = n_{10} + \frac{3E_r}{16} + \frac{n_7}{16} \tag{23}$$

Where n_{10} varies between 0 and approximately 0.2.

Hence unit 1's input is approximately:

$$I_1 = O_7 + 2O_{10} = 1.125E_r + 0.38n_7 + 2n_{10} \tag{24}$$

The maximum values of the random parts of this expression, n_7 and n_{10}, are high enough to mean that unit 1 can act as a veto unit even in the absence of visual input. However in that case its inhibition will be very noisy. It is guaranteed to have inhibitory effect only if $E_r > 0.66$. So in general unit 1 will act as a noisy veto unit, but the higher E_r the more likely are its inhibitory links to be active, and the less likely the lower E_r.

As stated earlier, as soon as the inputs to unit 6 exceed the veto threshold, its inhibitory links become active and it closes down all the right eye visual pathways. Inputs to 6 are from the environment (here referred to as E_l) and from unit 14. Hence,

$$I_6 = E_l + O_{14} \approx E_l + \frac{O_1}{2} \approx E_l + 0.28E_r + 0.09n_7 + 0.5n_{10} \tag{25}$$

Vetoing is guaranteed if $E_l > 0.75$. Given the value of n_{10}, it will occur in a noisy way for $E_l > 0.65$. At other values vetoing will be very noisy; if the above expression exceeds 0.75 the right eye pathways are closed down and their contributions to the expression go to zero, pulling the value below the threshold. It should be noted that the vetoing mechanism turns off excitatory output from a unit but *not* inhibitory outputs. Hence the veto feedback to unit 6 does not affect the above argument.

A crucial point to note at this juncture is that because the C2 network is built around the kinds of feedback loops described earlier, allowing the above analysis, then, by virtue of the 'settling down' argument of Section 11, the visual inputs can be seen to perturb the underlying network dynamics in a straightforward manner. This means that we can predict the motor behaviour given the visual input without explicitly having to take into account temporal factors (the robot's state history). It is this fact that lets us use the simple visualisable state spaces presented in the next section. More recent work with real robots has made use of the same type of network but with slightly altered timescales. Here state history has to be taken into account (the networks are no longer guaranteed to settle down between successive input samples) and slightly different higher dimensional state spaces, including a time dimension, have been used to analyse behaviours [24]. Networks operating on the same timescales as in this paper, and therefore amenable to exactly the same analysis, have also been successfully evolved to control real robots [17].

We can now summarise the motor output characteristics of the network. When E_l and E_r are both low, unit 15 acts as a noisy generator unit and unit 13 receives strong input from the noisy generator unit 11 via the double connection from 9. Hence we expect both motors to be activated in reverse, given a noisy straight line backwards behaviour. When E_r is high and E_l is low, 13 still receives strong input but 15 is now vetoed. Hence we expect a backwards rotation about the right wheel. The rotation will be noisy (jumping between rotation and straight line backwards) unless $E_r > 0.66$. When E_l is high and E_r is low, 13 is vetoed but 15 is left free to act as a noisy generator. Hence we expect a noisy backwards rotation about the left wheel. At intermediate combinations of E_l and E_r we expect a noisy alternation between backwards rotation about the left wheel and backwards rotation about the right wheel.

This network analysis can be combined with a use of $r\phi$ space to gain a global picture of the dynamic of the whole control system (network plus sensors). Only with this global picture can we really understand how the evolved controllers work.

13.2 The global picture

Figure 16 shows how $r\phi$ space is divided up into regions of different (E_l, E_r) visual signal combinations when the cylinder wall height is 15 units. Each bounded region has a constant combination over its area. This is different from the combinations in its neighbouring regions. Each region is labeled, along its right hand boundaries, with a unique integer. Figure 15 shows the actual grey level inputs (black=0, white=1) to the left and right photoreceptors represented in $r\phi$ space for this wall height. Table 1 gives the (E_l, E_r) combinations for each labelled region along with the dominant motor behaviour these will produce. These behaviours were deduced from the above network analysis. From this table the phase portrait of Figure 17 was produced. Using the geometrical properties of $r\phi$ space discussed earlier, a vector flow field was constructed across the regions. Where there is a noisy oscillation between two behaviours, a vector average was used. We can see that there appears to be a single attractor at $(rad, \frac{\pi}{2})$, where the robot wheel span radius is rad. This attractor[5] (marked with an X) corresponds to a rotation backwards about the right wheel, with the right wheel at the centre of the arena. Indeed the *whole* state space appears to be a basin of attraction

[5]The word attractor is being used a little loosely here, given the noise in the system, including the actuators.

region	*(lsig,rsig)*	*behaviour*	*region*	*(lsig,rsig)*	*behaviour*	*region*	*(lsig,rsig)*	*behaviour*
1	(0.33,0.33)	SlB	2	(0.33,0.39)	RrB	3	(0.33,0.44)	RrB
4	(0.39,0.33)	RrB	5	(0.44,0.33)	RrB	6	(0.33,0.50)	RrB
7	(0.39,0.39)	RrB	8	(0.50,0.33)	LrBsl	9	(0.33,0.56)	RrB
10	(0.56,0.33)	LrBsl	11	(0.39,0.50)	RrB	12	(0.50,0.50)	LrBsl
13	(0.50,0.39)	LrBsl	14	(0.44,0.50)	RrB	15	(0.50,0.44)	LrBsl
16	(0.56,0.39)	LrBsl	17	(0.39,0.56)	RrB	18	(0.44,0.56)	RrB
19	(0.56,0.44)	LrBsl	20	(0.28,0.33)	RrB	21	(0.22,0.39)	RrB
22	(0.11,0.44)	RrB	23	(0.11,0.50)	RrB	24	(0.22,0.56)	RrB
25	(0.28,0.56)	RrB	26	(0.56,0.28)	LrB	27	(0.56,0.22)	LrB
28	(0.56,0.11)	LrB	29	(0.50,0.11)	LrB	30	(0.44,0.11)	LrB
31	(0.39,0.22)	SlB	32	(0.33,0.28)	SlB	33	(0.22,0.33)	RrB
34	(0.11,0.39)	RrB	35	(0.11,0.56)	RrB	36	(0.39,0.11)	LrB
37	(0.33,0.22)	SlB	38	(0.28,0.28)	SlB	39	(0.11,0.33)	RrB
40	(0.33,0.11)	LrB	41	(0.17,0.33)	RrB	42	(0.17,0.56)	RrB
43	(0.56,0.17)	LrB	44	(0.33,0.17)	SlB	45	(0.22,0.22)	SlB
46	(0.17,0.28)	SlB	47	(0.28,0.17)	SlB	48	(0.11,0.28)	SlB
49	(0.28,0.11)	SlB	50	(0.17,0.17)	SlB	51	(0.11,0.22)	SlB
52	(0.44,0.44)	RrB	53	(0.22,0.11)	LrB	54	(0.11,0.11)	LrB
55	(0.11,0.17)	SlB	56	(0.17,0.11)	SlB			

Table 1: Table showing visual signals and dominant C2 behaviours in distinct visual regions of world for wall height 15. RrB= backwards rotation about right wheel; LrB= likewise about left wheel; SlB= straight line backwards; LrBsl= noisy oscillation between LrB and SlB.

for this single attractor. That is, from wherever, and at whatever orientation, the robot is started it will go to the centre and rotate in a very low radius circle. In other words, it cannot help but succeed at its implicit task. This is exactly the behaviour that was observed on countless runs of the simulated robot, as illustrated for a typical run in Figure 13. Some of these runs were translated into $r\phi$ space to see if the predicted phase portrait was confirmed. The resulting trajectories are shown in Figure 18, plotted over the right photoreceptor grey levels. Clearly the vector flow field is confirmed, with an attractor at $(rad, \frac{\pi}{2})$.

Remember that C2 generalised very well over a wide range of wall heights. Figure 19 shows the grey level inputs for left and right photoreceptors at wall height 5 represented in $r\phi$ space. Figure 20 shows how the space is divided up into (E_l, E_r) combination regions. Clearly the visual structure is quite different. In a similar way to that described earlier, Table 2 was constructed. From this the vector flow field of Figure 21 was produced. It can be seen that this is different from the one constructed for wall height 15. There is an attractor at $(rad, \frac{3\pi}{2})$ and what looks like a weak attractor at $(rad, \frac{\pi}{2})$. The phase portrait is confirmed by Figure 22 which plots trajectories in $r\phi$ space from many runs of the simulated robot at this low wall height. Note that the existence of the two attractors, one much stronger than the other, is confirmed. The $(rad, \frac{3\pi}{2})$ attractor corresponds to a backwards rotation about the left wheel with the left wheel at the centre. Again the controller acts very robustly, always moving to the centre and staying there.

C2 is an example of the power of artificial evolution. The visual morphology and network dynamics have evolved together to produce a very robust and general control system for the implicit task described by the evaluation function. In the face of noise, indeed by exploiting noise, the controller cannot help but succeed. Although the controller succeeds no matter where it is started in the arena, during evolution it was always started near the edge. Although space does not allow a full discussion of them, there are a number of other observations worth noting. The phase portraits show that at both wall heights for large r the robot will always rotate in the direction that will bring it normal to the wall in the shortest time; it then moves backwards towards the centre. The phase portrait in Figure 17 suggests that, at wall height 15, if the robot actually took a route directly to the centre (ϕ constant at 0), it may get stuck in a cyclical attractor. But the noise in the actuators means this is impossible (ϕ cannot be held at 0) and in practice the robot moves in slightly off-centre, getting drawn rapidly to the point attractor. Lastly, it is worth noting that C2's network architecture is somewhat reminiscent of a Brooksian Subsumption architecture [7], with the left and right visual pathways almost separate, and the left photoreceptor network being able to inhibit the right photoreceptor network under certain environmental conditions.

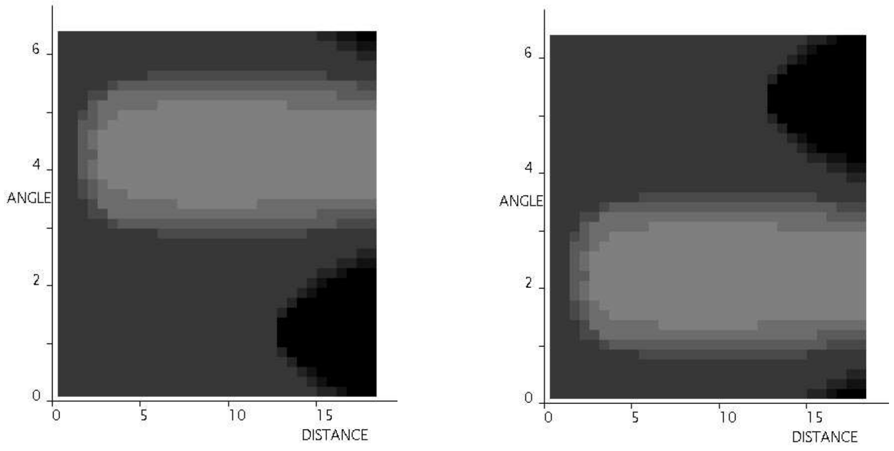

Figure 15: Grey level 0-1 (black-white) $r\phi$ representations of left and right visual signals for C2 for wall height 15.

14 Extending the analysis

The network analysed has many interesting subtleties but is relatively simple. So were its environments and its sensory signals. So, although we were able to provide a thorough analysis and, in particular, were able to show how the network dynamics and visual morphology had evolved in tandem to provide a highly robust system for achieving the arena centring task, the question still remains as to how far this kind of analysis can be taken.

region	*(lsig,rsig)*	*behaviour*	*region*	*(lsig,rsig)*	*behaviour*	*region*	*(lsig,rsig)*	*behaviour*
1	(0.56,0.56)	LrRrb	2	(0.56,0.61)	LrRrb	3	(0.56,0.67)	LrRrb
4	(0.61,0.56)	LrB	5	(0.67,0.56)	LrB	6	(0.56,0.72)	LrRrb
7	(0.61,0.61)	LrB	8	(0.72,0.56)	LrB	9	(0.56,0.78)	LrRrb
10	(0.78,0.56)	LrB	11	(0.61,0.72)	LrB	12	(0.72,0.72)	LrB
13	(0.72,0.61)	LrB	14	(0.67,0.72)	LrB	15	(0.72,0.67)	LrB
16	(0.78,0.61)	LrB	17	(0.61,0.78)	LrB	18	(0.67,0.78)	LrB
19	(0.78,0.67)	LrB	20	(0.50,0.56)	LrRrb	21	(0.50,0.61)	LrRrb
22	(0.39,0.67)	RrB	23	(0.33,0.72)	RrB	24	(0.39,0.78)	RrB
25	(0.50,0.78)	LrRrb	26	(0.78,0.50)	LrB	27	(0.78,0.44)	LrB
28	(0.72,0.33)	LrB	29	(0.67,0.39)	LrB	30	(0.61,0.44)	LrB
31	(0.56,0.50)	LrRrb	32	(0.44,0.56)	RrB	33	(0.33,0.61)	RrB
34	(0.33,0.67)	RrB	35	(0.33,0.78)	RrB	36	(0.44,0.78)	RrB
37	(0.78,0.33)	LrB	38	(0.67,0.33)	LrB	39	(0.61,0.33)	LrB
40	(0.56,0.44)	LrB	41	(0.39,0.56)	RrB	42	(0.78,0.39)	LrBsl
43	(0.56,0.33)	LrBsl	44	(0.50,0.50)	LrBsl	45	(0.33,0.56)	RrB
46	(0.83,0.44)	LrBsl	47	(0.56,0.39)	LrBsl	48	(0.44,0.50)	RrB
49	(0.33,0.83)	RrB	50	(0.44,0.83)	RrB	51	(0.50,0.89)	LrRrb
52	(0.50,0.83)	LrRrb	53	(0.56,0.83)	LrRrb	54	(0.83,0.50)	LrRrb
55	(0.89,0.50)	LrRrb	56	(0.89,0.44)	LrB	57	(0.83,0.33)	LrB
58	(0.50,0.44)	LrB	59	(0.44,0.44)	RrB	60	(0.39,0.50)	RrB
61	(0.39,0.89)	RrB	62	(0.44,0.89)	RrB	63	(0.83,0.56)	LrBsl
64	(0.89,0.39)	LrBsl	65	(0.50,0.39)	LrBsl	66	(0.33,0.50)	RrB
67	(0.28,0.56)	RrB	68	(0.22,0.61)	RrB	69	(0.11,0.67)	RrB
70	(0.11,0.72)	RrB	71	(0.22,0.78)	RrB	72	(0.28,0.78)	RrB
73	(0.28,0.89)	RrB	74	(0.33,0.94)	RrB	75	(0.50,0.94)	LrRrb
76	(0.56,0.89)	LrRrb	77	(0.94,0.50)	LrB	78	(0.94,0.33)	LrB
79	(0.89,0.33)	LrB	80	(0.89,0.28)	LrB	81	(0.78,0.22)	LrB
82	(0.78,0.11)	LrB	83	(0.72,0.11)	LrB	84	(0.67,0.11)	LrB
85	(0.61,0.22)	LrB	86	(0.56,0.28)	LrB	87	(0.50,0.28)	LrB
88	(0.50,0.33)	LrB	89	(0.39,0.39)	RrB	90	(0.28,0.44)	RrB
91	(0.28,0.50)	RrB	92	(0.22,0.56)	RrB	93	(0.11,0.61)	RrB
94	(0.11,0.78)	RrB	95	(0.22,0.89)	RrB	96	(0.28,0.94)	RrB
97	(0.39,1.00)	RrB	98	(0.44,0.94)	RrB	99	(0.89,0.56)	LrRrb
100	(0.94,0.44)	LrRrb	101	(1.00,0.39)	LrB	102	(0.94,0.28)	LrB
103	(0.89,0.22)	LrB	104	(0.61,0.11)	LrB	105	(0.56,0.22)	LrB
106	(0.44,0.28)	SlB	107	(0.33,0.33)	SlB	108	(0.22,0.50)	RrB
109	(0.11,0.56)	RrB	110	(0.11,0.89)	RrB	111	(0.22,0.94)	RrB
112	(0.33,1.00)	RrB	113	(0.94,0.22)	LrB	114	(0.89,0.11)	LrB
115	(0.56,0.11)	LrB	116	(0.50,0.22)	LrB	117	(0.28,0.28)	SlB
118	(0.22,0.39)	SlB	119	(0.17,0.44)	RrB	120	(0.11,0.50)	RrB
121	(0.11,0.83)	RrB	122	(0.17,0.94)	RrB	123	(0.28,1.00)	RrB
124	(0.39,0.94)	RrB	125	(0.83,0.61)	LrRrb	126	(1.00,0.28)	LrB
127	(0.94,0.17)	LrB	128	(0.83,0.11)	LrB	129	(0.50,0.11)	LrB
130	(0.50,0.17)	LrB	131	(0.44,0.22)	LrB	132	(0.22,0.22)	SlB
133	(0.17,0.33)	SlB	134	(0.11,0.44)	RrB	135	(0.11,0.94)	RrB
136	(0.17,1.00)	RrB	137	(0.22,1.00)	RrB	138	(1.00,0.33)	LrB
139	(1.00,0.22)	LrB	140	(1.00,0.17)	LrB	141	(0.94,0.11)	LrB
142	(0.44,0.11)	LrB	143	(0.33,0.17)	SlB	144	(0.11,0.33)	SlB
145	(0.11,1.00)	RrB	146	(1.00,0.11)	SlB	147	(0.33,0.11)	SlB
148	(0.17,0.17)	SlB	149	(0.11,0.22)	SlB	150	(0.67,0.67)	LrRrb
151	(0.94,0.39)	LrRrb	152	(0.39,0.11)	SlB	153	(0.22,0.11)	SlB
154	(0.11,0.11)	SlB	155	(0.11,0.17)	SlB	156	(0.11,0.28)	SlB
157	(0.11,0.39)	SlB	158	(0.28,0.11)	SlB	159	(0.17,0.11)	SlB

Table 2: Table showing visual signals and dominant C2 behaviours in distinct visual regions of world for wall height 5. LrB=backwards rotation about left wheel; RrB= likewise about right wheel; SlB= straight line backwards; LrRrb= noisy oscillation between LrB and RrB; LrBsl= noisy oscillation between LrB and SlB.

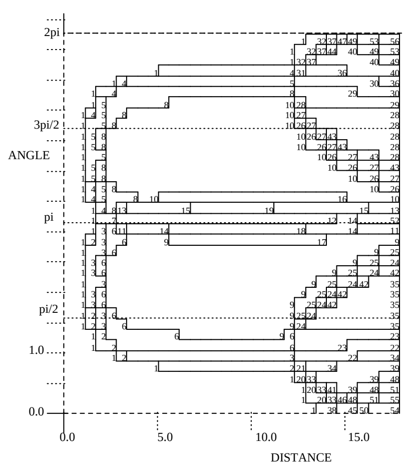

Figure 16: Distinct visual regions for C2 for wall height 15, shown in $r\phi$ space.

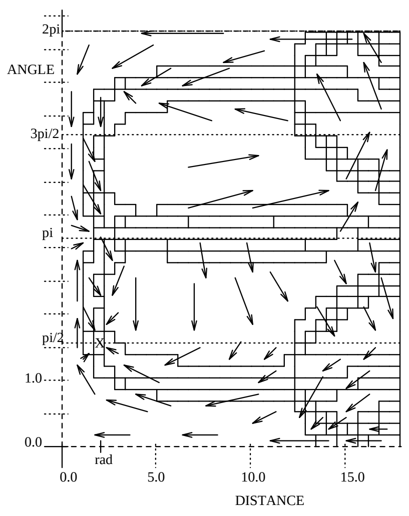

Figure 17: Constructed phase portrait for C2 at wall height 15.

Of course it is not possible to give any sort of definitive answer to this question, but there are a number of issues that appear to be highly pertinent. A major concern is in finding the right level of abstraction to allow a meaningful analysis with manageable state spaces. Two techniques to help in this direction were used here: collapsing the network dynamics into feedback loop properties, and collapsing motor output dynamics into behaviour categories. The global picture afforded by using the $r\phi$ space to construct a visualisation of the robot's visual world was made possible by exploiting the simplicity and symmetries of the world. For more complex environments, especially dynamic environments with changing lighting conditions, that kind of global picture will just not be possible. Some kind of statistical categorisation will probably be needed to produce a manageable visual-inputs state space. This will get harder, and will probably necessitate more abstract spaces, as the number of visual receptive fields increases. However, work with real robots reported in [24] shows that it is certainly possible to extend the kind of analysis reported in the present paper to more complex environments and tasks. Useful *analyses* of existing mechanisms can often adopt higher levels of abstraction than can be tolerated in the *design* of the mechanisms. So although we may need artificial evolution to help develop complex systems, this does not necessarily mean that we will not be able to understand its products.

This paper has been concerned with ways of analysing visually guided behaviours from a dynamical systems perspective. Other researchers have been treading related paths, but have not been specifically concerned with visuo-motor behaviours. In particular, Randy Beer and his students have analysed a number of behaviour generating network controllers in dynamical

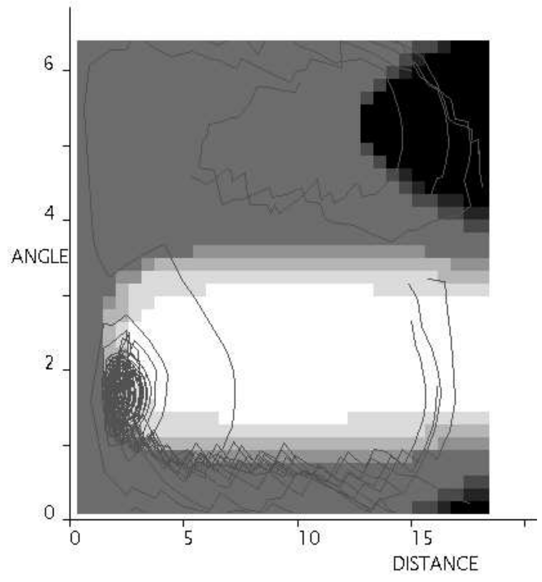

Figure 18: C2 trajectories for wall height 15 shown in $r\phi$ space and superimposed over right eye grey levels.

systems terms. Beer and Gallagher analysed recurrent dynamical network locomotion controllers for an artificial insect [3]; while Yamauchi and Beer have shown how the dynamics of a network shaped by a genetic algorithm are capable of generating sequential behaviour and learning without recourse to an explicit lifetime learning algorithm and without an a priori discretization of states or time [35]. Tim Smithers has argued for agent environment systems to be viewed from a dynamical systems perspective [28]. For some time Steels has advocated a dynamical systems framework for autonomous agents, particularly with reference to emergent functionality in such systems [29, 30].

15 Animate vision

Animate vision [1] is an approach that advocates that visual perception should be considered as a dynamic behaviour that an animal performs in order to achieve its goals in a dynamic environment. This is in contrast to traditional approaches to computer vision which tended to take snapshots of a stationary world, and then analyse them. It follows from this concept of animate vision that sensing and motor behaviour must interact closely with each other. In contrast, the advocates of functional decomposition assumed that vision was one task, and action something different; research on one, it was thought, should be done in a different room from research on the other.

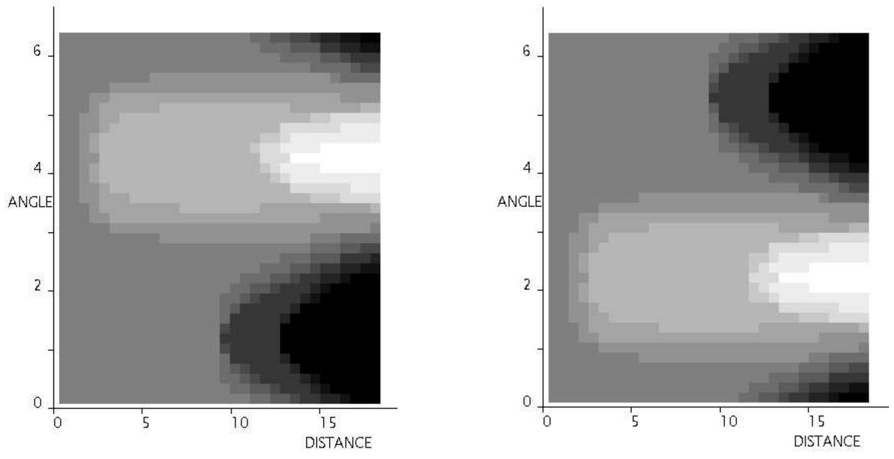

Figure 19: Grey level 0-1 (black-white) $r\phi$ representations of left and right visual signals for C2 for wall height 5.

From our viewpoint of cognition, that a robot or animal should be seen as a whole system with its own internal dynamics, coupled to a dynamical world, the tenets of animate vision appear perfectly natural. Furthermore, the behaviour of the successful robots in the circular arena supports an interpretation in terms of primitive animate vision. Any individual snapshot that the robot could be considered to take would consist of a single grey-scale value for the one pixel of its one effective photorector; on its own, this could be no basis for creating an internal model of the world at that moment in time, and no basis for a decision whether to circle left, right, or go straight ahead. Yet the action that the robot takes of continually moving, switching between straight line motion and circular motions with a greater or lesser radius, allows the variation in light input to be used as a basis for 'deciding' when to increase or decrease the radius or move in a straight line; and this results in the robot reaching the centre and remaining there, as encouraged by the evaluation function. The dynamics of the overall system inevitably force it to the centre and then hold it there; if it strays away it is very quickly sucked back.

16 Further work: moving into the real world

The experiments discussed in this paper were intended to test the plausibility of our approach. However, the simulated visual environment was very simple and computational costs will increase dramatically as the visual environment becomes more complex. Indeed, even ignoring computational costs, the plausible modelling of visual inputs in such circumstances is highly problematic. Hence we have developed an experimental setup that allows the whole evo-

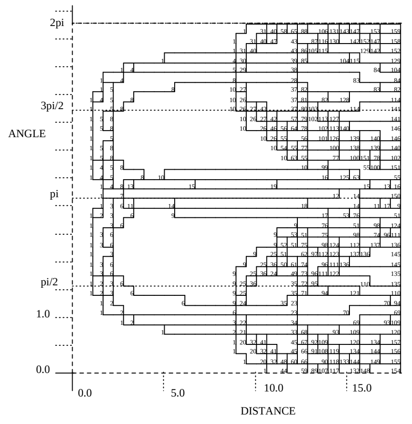

Figure 20: Distinct visual regions for C2 for wall height 5, shown in $r\phi$ space.

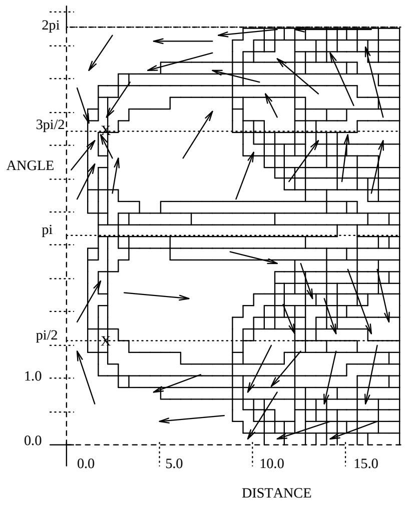

Figure 21: Constructed phase portrait in $r\phi$ space for C2 at wall height 5.

lutionary process to work with a real robot, moving in the real world, and without recourse to simulated vision. This apparatus is briefly discussed below, see [17] for full details and an account of early experimental results achieved with it.

16.1 The gantry-robot

A gantry runs along the top of a frame about 1.5m by 1m. The gantry is moved in the X-direction by a stepper motor, and a platform is moved across the gantry in the Y-direction by a second stepper motor (see Figure 23). Supported below this, on the bottom of a vertical pole, is our 'part-real, part-virtual' robot. This has a camera facing in a vertical direction at an angled mirror which gives horizontal vision. The camera does not rotate, but rotation of the mirror, together with adjustment of the software subsampling the CCD image allows vision in any direction. Around the circumference of the camera is an array of whiskers (touch-sensors).

This robot is thus 'real' in the sense that it uses real light for its vision, real touch for its whiskers, and can move around in the Toytown-scale environments that we can build for it. It is partly 'virtual' in the sense that it has virtual left and right motors, modeling those of the original mobile robot, which are aligned in the direction of gaze given by the angled mirror; motion commands sent to these virtual motors are in practice converted to the appropriate movements of the X and Y stepper motors, for horizontal translation, and of the angled-mirror mounting, for rotation. It is also 'virtual' in the sense that the (genetically specified) virtual photoreceptors used

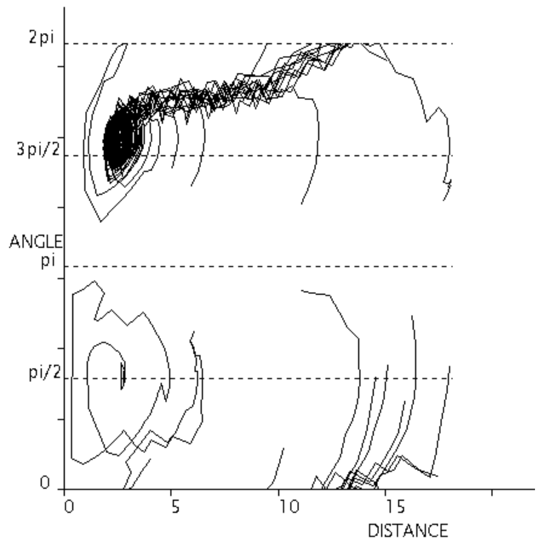

Figure 22: C2 trajectories translated into $r\phi$ space for wall height 5.

for vision are implemented through sub-sampling the CCD image.

The genetically specified control networks for this robot are run off-board on a fast PC communicating via an umbilical cable. Through the use of stepper motors for X and Y motion, a supervisory program always knows where the robot is, this knowledge being used to evaluate its performance — even though the control system for the robot has no such privileged knowledge. This setup, with off-board computing and avoidance of tangled umbilicals, means that the apparatus can be run continuously for long periods of time — making artificial evolution feasible. A top-level program automatically evaluates, in turn, each member of a population of control systems. A new population is produced by selective interbreeding and the cycle repeats. The gantry-robot is being used in this way to further our understanding of how to concurrently evolve visual morphologies and control networks for visually guided robots.

Another advantage of this apparatus is that it is possible to automatically collect many forms of data for use in analyses like those given earlier in this paper. See [24] for such an analysis of behaviours evolved on the gantry-robot.

17 Conclusions

We have shown it is possible to evolve dynamical network controllers plus visual morphologies that produce simple robust autonomous behaviours across a range of related environments.

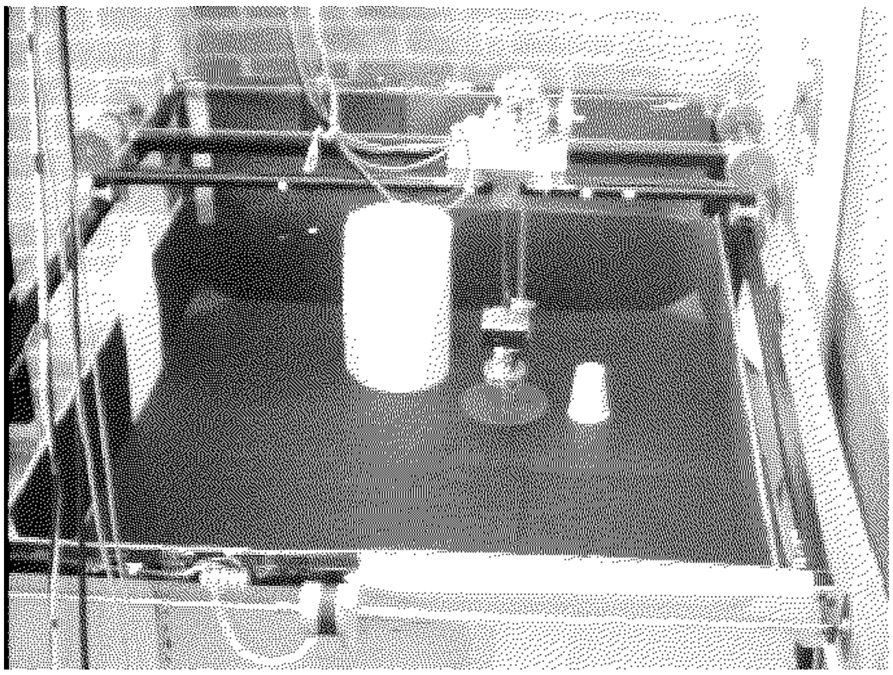
Figure 23: *The Gantry-Robot.*

We have shown that these systems can be thoroughly understood, and introduce one strand of our current research where the simulation of the agent environment coupling is abandoned.

Acknowledgements

Work on the Sussex evolutionary robotics project is funded by EPSRC grant GR/J18125. Thanks to Michael Wheeler and Tim Smithers for useful discussions on some of the material covered in this paper.

References

[1] D. H. Ballard. Animate vision. *Artificial Intelligence*, 48:57–86, 1991.

[2] R.D. Beer. A dynamical systems perspective on autonomous agents. Technical Report CES-92-11, Case Western Reserve University, Cleveland, Ohio, 1992.

[3] R.D. Beer and J.C. Gallagher. Evolving dynamic neural networks for adaptive behavior. *Adaptive Behavior*, 1(1):91–122, 1992.

[4] R. Belew and L. Booker, editors. *Proceedings of the Fourth International Conference on Genetic Algorithms*. Morgan Kaufmann, 1991.

[5] R. A. Brooks. Coherent behavior from many adaptive processes. In D. Cliff, P. Husbands, J.-A. Meyer, and S.W. Wilson, editors, *From Animals to Animats 3: Proceedings of The Third International Conference on Simulation of Adaptive Behavior*, pages 22–29. MIT Press/Bradford Books, Cambridge, MA, 1994.

[6] R.A. Brooks. Intelligence without reason. In *Proceedings IJCAI-91*, pages 569–595. Morgan Kaufmann, 1991.

[7] R.A. Brooks. Intelligence without representation. *Artificial Intelligence*, 47:139–159, 1991.

[8] Rodney A. Brooks. Artificial life and real robots. In F. J. Varela and P. Bourgine, editors, *Proceedings of the First European Conference on Artificial Life*, pages 3–10. MIT Press/Bradford Books, Cambridge, MA, 1992.

[9] D. Cliff, I. Harvey, and P. Husbands. Explorations in evolutionary robotics. *Adaptive Behavior*, 2(1):73–110, 1993.

[10] D. Cliff, P. Husbands, and I. Harvey. Evolving visually guided robots. In H. Roitblat J. Meyer and S. Wilson, editors, *From Animals to Animats 2: Proceedings of the 2nd International conference on the Simulation of Adaptive Behaviour*, pages 374–383. MIT Press/Bradford Books, 1993.

[11] D. T. Cliff, P. Husbands, and I. Harvey. Analysis of evolved sensory-motor controllers. In *Proceedings of Second European Conference on Artificial Life, ECAL93*. Brussels, May 1993, 1993.

[12] M. Dorigo and U. Schnepf. Genetic-based machine learning and behavior-based robotics: A new synthesis. *IEEE Transactions on Systems, Man, Cybernetics*, 23(1):141–154, 1993.

[13] S. Forrest, editor. *Proc. 5th Int. Conf. on GAs*. Morgan Kaufmann, 1993.

[14] David E. Goldberg. *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison-Wesley, Reading, Massachusetts, USA, 1989.

[15] I. Harvey. The saga cross: The mechanics of recombination for species with variable length genotypes. In R. Manner and B. Manderick, editors, *Parallel Problem Solving from Nature,2*, pages 269–278. North-Holland, 1992.

[16] I. Harvey, P. Husbands, and D. Cliff. Issues in evolutionary robotics. In H. Roitblat J. Meyer and S. Wilson, editors, *From Animals to Animats 2: Proceedings of the 2nd International conference on the Simulation of Adaptive Behaviour*, pages 364–373. MIT Press/Bradford Books, 1993.

[17] I. Harvey, P. Husbands, and D. Cliff. Seeing the light: Artificial evolution, real vision. In D. Cliff, P. Husbands, J.-A. Meyer, and S. Wilson, editors, *From Animals to Animats 3, Proc. of 3rd Intl. Conf. on Simulation of Adaptive Behavior, SAB'94*, pages 392–401. MIT Press/Bradford Books, 1994.

[18] Inman Harvey. Species adaptation genetic algorithms: The basis for a continuing SAGA. In *Proceedings of the First European Conference on Artificial Life*, pages 346–354. MIT Press/Bradford Books, Cambridge, MA, 1992.

[19] Inman Harvey. Evolutionary robotics and SAGA: the case for hill crawling and tournament selection. In C. Langton, editor, *Artificial Life III*, pages 299–326. Santa Fe Institute Studies in the Sciences of Complexity, Proceedings Vol. XVI, Addison-Wesley, Redwood City CA, 1994.

[20] John Holland. *Adaptation in Natural and Artificial Systems*. University of Michigan Press, Ann Arbor, USA, 1975.

[21] P. Husbands. Genetic algorithms in optimisation and adaptation. In L. Kronsjo and D. Shumsheruddin, editors, *Advances in Parallel Algorithms*, pages 227–277. Blackwell Scientific Publishing, Oxford, 1992.

[22] P. Husbands and I. Harvey. Evolution versus design: Controlling autonomous robots. In *Integrating Perception, Planning and Action, Proceedings of 3rd Annual Conference on Artificial Intelligence, Simulation and Planning*, pages 139–146. IEEE Press, 1992.

[23] P. Husbands, I. Harvey, D. Cliff, and G. Miller. The use of genetic algorithms for the development of sensorimotor control systems. In P. Gaussier and J-D. Nicoud, editors, *Proceedings of From Perception to Action Conference*, pages 110–121. IEEE Computer Society Press, 1994.

[24] P. Husbands, I. Harvey, N. Jakobi, and D. Cliff. Evolved network controllers for mobile robots. In preparation.

[25] Stuart Kauffman. Adaptation on rugged fitness landscapes. In Daniel L. Stein, editor, *Lectures in the Sciences of Complexity*, pages 527–618. Addison Wesley: Santa Fe Institute Studies in the Sciences of Complexity, 1989.

[26] J. Koza. *Genetic Programming: On the programming of computers by means of natural selection.* MIT Press, 1992.

[27] J. D. Schaffer, editor. *Proceedings of the Third International Conference on Genetic Algorithms*, San Mateo, California, 1989. Morgan Kaufmann.

[28] Tim Smithers. On why better robots make it harder. In D. Cliff, P. Husbands, J.-A. Meyer, and S. Wilson, editors, *From Animals to Animats 3, Proc. of 3rd Intl. Conf. on Simulation of Adaptive Behavior, SAB'94*, pages 54–72. MIT Press/Bradford Books, 1994.

[29] L. Steels. Towards a theory of emergent functionality. In J.-A. Meyer and S.W. Wilson, editors, *From Animals to Animats: Proceedings of The First International Conference on Simulation of Adaptive Behavior*, pages 451–461. MIT Press/Bradford Books, Cambridge, MA, 1991.

[30] L. Steels. The artificial life roots of artificial intelligence. *Artificial Life*, 1(2):75–110, 1994.

[31] A. Thompson. Mr. chips technical reference manual. Technical report, School of Cognitive and Computing Sciences, University of Sussex, 1994.

[32] Tim van Gelder. What might cognition be if not computation. Technical Report 75, Indiana University Cognitive Sciences, 1992.

[33] F. Varela, E. Thompson, and E. Rosch. *The Embodied Mind.* MIT Press, 1991.

[34] B. Yamauchi and R. Beer. Integrating reactive, sequential, and learning behavior using dynamical neural networks. In D. Cliff, P. Husbands, J.-A. Meyer, and S. Wilson, editors, *From Animals to Animats 3, Proc. of 3rd Intl. Conf. on Simulation of Adaptive Behavior, SAB'94*, pages 382–391. MIT Press/Bradford Books, 1994.

[35] Brian M. Yamauchi and Randall D. Beer. Sequential behavior and learning in evolved dynamical neural networks. *Adaptive Behavior*, 2(3):219–246, 1994.