Extract from: "Artificial Evolution: A New Path for Artificial Intelligence?" P. Husbands, I. Harvey, D. Cliff, and G. Miller (Sussex). BRAIN AND COGNITION 34, 130–159 (1997). Full version also available (gives extra intro on evolutionary robotics, and some reasonable but outdated speculations about future directions).

~~the help of automated techniques, of which artificial evolution is a front runner.~~

5. AN EXAMPLE EXPERIMENT

This section makes much of the surrounding discussion more concrete by focusing on a particular experiment to evolve a network-based control system for a mobile robot engaged in visually guided tasks of increasing complexity.

There are many different ways of realizing each stage of the cycle shown in Fig. 1. A crucial decision is whether or not to use simulation at the evaluation stage, transferring the end results to the real world. Since an evolutionary approach potentially requires the evaluation of populations of robots over many generations, a natural first thought is that simulations will speed up the process, making it more feasible. Despite initial scepticism (Brooks, 1992), it has recently been shown that control systems evolved in carefully constructed simulations, with an appropriate treatment of noise, transfer extremely well to reality, generating almost identical behaviors in the real robot (Jakobi, Husbands, & Harvey, 1995; Thompson, 1995). However, both of these examples involved relatively simple robot–environment interaction dynamics. Once even low-bandwith vision is used, simulations become altogether more problematic. They become difficult and time consuming to construct and computationally very intensive to run. Hence evolving visually guided robots in the real world becomes a more attractive option. The case study described in this section revolves around a piece of robotic equipment specially designed to allow the real-world evolution of visually guided behaviors—the Sussex gantry robot.

5.0.1. Concurrent evolution of visual morphologies and control networks. Rather than imposing a fixed visual sampling morphology, we believe a more powerful approach is to allow the visual morphology to evolve along with the rest of the control system. Hence we genetically specify regions of the robot's visual field to be subsampled, these provide the only visual inputs to the control network. It would be desirable to have many aspects of the robot's morphology under genetic control, although this is not yet technically feasible.

5.0.2. The gantry robot. The gantry robot is shown in Fig. 3. The robot is cylindrical, some 150 mm in diameter. It is suspended from the gantry frame with stepper motors that allow translational movement in the *X* and *Y* directions, relative to a coordinate frame fixed to the gantry. The maximum *X* (and *Y*) speed is about 200 mm/sec. Such movements, together with appropriate rotation of the sensory apparatus, correspond to those which would be produced by left and right wheels. The visual sensory apparatus consists of a CCD camera pointing down at a mirror inclined at 45° to the vertical (see Fig. 4). The mirror can be rotated about a vertical axis so that its orienta-

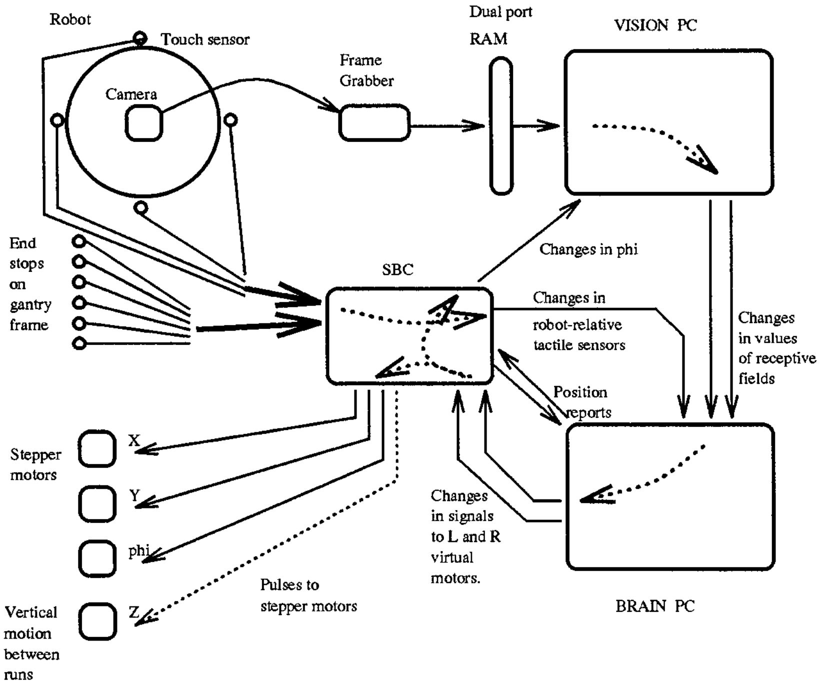

FIG. 2. The different roles of the Vision computer, the Brain computer and the SBC.

FIG. 3. The Gantry viewed from above. The horizontal girder moves along the side rails, and the robot is suspended from a platform which moves along this girder.

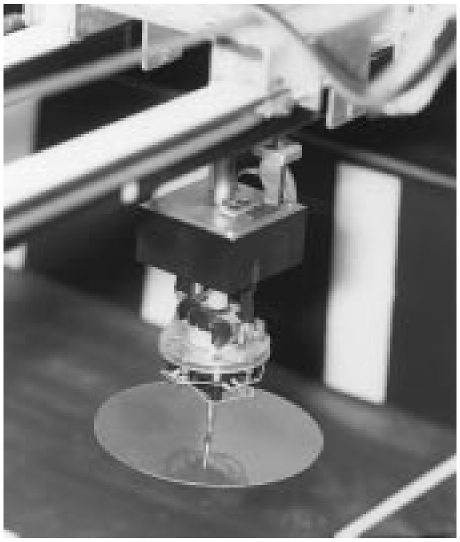

FIG. 4. The gantry robot. The camera inside the top box points down at the inclined mirror, which can be turned by the stepper motor beneath. The lower plastic disk is suspended from a joystick, to detect collisions with obstacles.

tion always corresponds to the direction the ‘‘robot’’ is facing. The visual inputs undergo some transformations en route to the control system, described later. The hardware is designed so that these transformations are done completely externally to the processing of the control system.

The control system for the robot is run off-board on a fast personal computer, the ‘‘Brain PC.’’ This computer receives any changes in visual input by interrupts from a second dedicated ‘‘Vision PC.’’ A third (single-board) computer, the SBC, sends interrupts to the Brain PC signaling tactile inputs resulting from the robot bumping into walls or physical obstacles. The only outputs of the control system are motor signals. These values are sent, via interrupts, to the SBC, which generates the appropriate stepper motor movements on the gantry.

The roles of the three computers are illustrated in Fig. 2. Continuous visual data is derived from the output of the small monochrome CCD camera. A purpose-built Frame Grabber transfers a 64 × 64 image at 50 Hz into a high-speed 2 K CMOS dual-port RAM, completely independently and asynchronously relative to any processing of the image by the Vision PC. The Brain PC runs the top-level genetic algorithm and during an individual evaluation, it is dedicated to running a genetically specified control system for a fixed period. At intervals during an evaluation, a signal is sent from the Brain PC to the SBC requesting the current position and orientation of the robot. These are used in keeping score according to the current fitness function. The Brain

PC receives signals, to be fed into the control system, representing sensory inputs from the Vision PC and the SBC. The visual signals are derived from averaging over genetically specified circular receptive patches in the camera's field of view.

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. For full technical details of the system see Harvey et al. (1994).

5.0.3. The artificial neural networks. The artificial neurons used have 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). 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 below. Noise was added to provide further potentially interesting and useful dynamics. The noise was uniformly distributed in the real range $[-N, +N]$.

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

The networks' continuous nature was modeled by using very fine time slice techniques. In the experiments described in this paper the following neuron parameter setting were used: $N = 0.1$, $T_v = 0.75$, $T_1 = 0.0$, and $T_2 = 2.0$. The networks are hardwired in the sense that they do not undergo any architectural changes during their lifetime, they all had unit weights and time delays on their connections. These networks are just one of the class we are interested in investigating.

5.0.4. The genetic encoding. Two ‘‘chromosomes’’ per robot are used. One of these is a fixed length bit string encoding the position and size of three visual receptive patches as described above. Three eight-bit fields per patch are used to encode their radii and polar coordinates in the camera's circular field of view. The other chromosome is a variable-length character

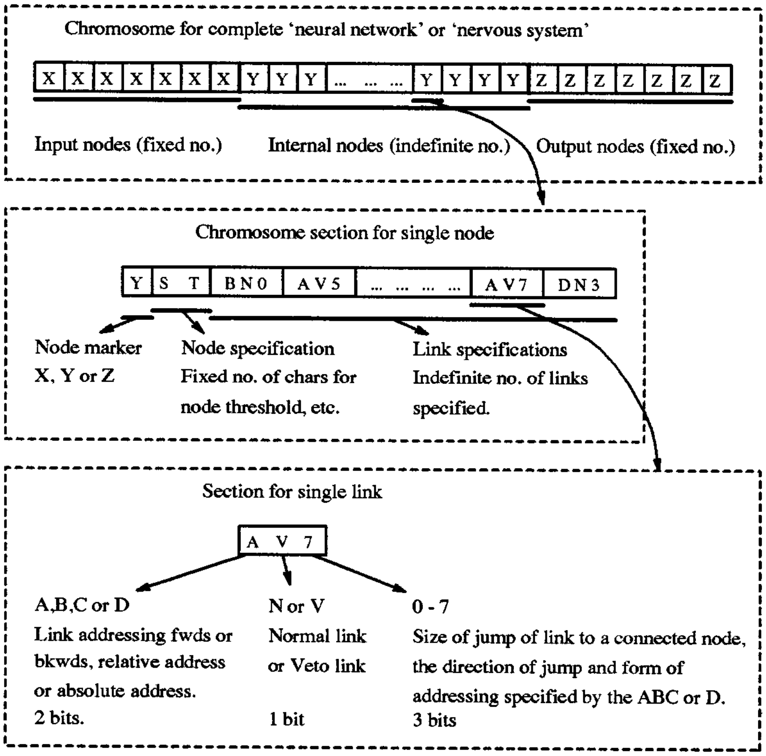

FIG. 5. The genetic encoding scheme.

string encoding the network topology. The genetic encoding used for the control network is illustrated in Fig. 5.

The network chromosome is interpreted sequentially. First the input units are coded for, each preceded by a marker. For each node, the first part of its gene can encode node properties such as threshold values; there then follows a variable number of character groups, each representing a connection from that node. Each group specifies whether it is an excitatory or veto connection, and then the target node is indicated by jump type and jump size. In a manner similar to that used in Harp and Samad (1992), the jump type allows for both relative and absolute addressing. Relative addressing is provided by jumps forward or backward along the genotype order; absolute addressing is relative to the start or end of the genotype. These modes of addressing mean that offspring produced by crossover will always be legal. There is one input node for each sensor (three visual, four tactile).

The internal nodes and output nodes are handled similarly with their own identifying genetic markers. Clearly this scheme allows for any number of internal nodes. The variable length of the resulting genotypes necessitates a careful crossover operator which exchanges homologous segments. In keeping with SAGA principles, when a crossover between two parents can result in an offspring of different length, such changes in length (although allowed) are restricted to a minimum (Harvey, 1992a). There are four output neurons, two per motor. The outputs of each pair are differenced to give a signal in the range $[-1,1]$.

5.0.5. Experimental setup. In each of the experiments a population size of 30 was used with a genetic algorithm employing a linear rank-based selection method, ensuring the best individual in a population was twice as likely to breed as the median individual. Each generation took about 1.5 hr to evaluate. The most fit individual was always carried over to the next generation unchanged. A specialised crossover allowing small changes in length between offspring and parents was used (Cliff et al., 1993). Mutation rates were set at 1.0 bit per vision chromosome and 1.8 bits per network chromosome.

With the walls and floor of the gantry environment predominantly dark, initial tasks were navigating toward white paper targets. In keeping with the incremental evolutionary methodology, deliberately simple visual environments are used initially, as a basis to moving on to more complex ones. Illumination was provided by fluorescent lights in the ceiling above, with the gantry screened from significant daylight variations. However, the dark surfaces did not in practice provide uniform light intensities, neither over space nor over time. Even when the robot was stationary, individual pixel values would fluctuate by up to 13%.

5.1. Results

5.1.1. Big target. In the first experiment, one long gantry wall was covered with white paper. The evaluation function ϵ_1, to be maximized, implicitly defines a target locating task, which we hoped would be achieved by visuomotor coordination

$$\epsilon_1 = \sum_{i=1}^{i=20} Y_i, \tag{2}$$

where Y_i are the perpendicular distances of the robot from the wall opposite that to which the target is attached, sampled at 20 fixed-time intervals throughout a robot trial which lasted a total of about 25 sec. The closer to the target the higher the score. For each robot architecture four trials were run, each starting in the same distant corner, but facing in four different partially random directions, to give a range of starts facing into obstacle walls as well as toward the target. As the final fitness of a robot control architecture was based on the *worst* of the four trials (to encourage robustness), and since in this case scores accumulated monotonically through a trial, this allowed later trials among the four to be prematurely terminated when they bettered previous trials. In addition, any control systems that had

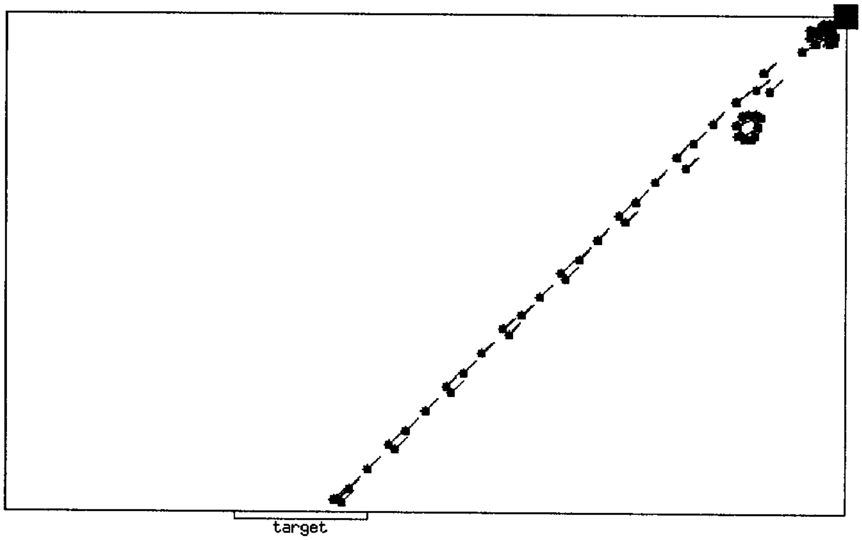

FIG. 6. Behavior of the best of a later generation evolved under second evaluation function. The dots and trailing lines show the front of the robot and its orientation. Coarsely sampled positions from each of four runs are shown, starting in different orientations from the top right corner.

not produced any movement by ⅓ of the way into a trial was aborted and given zero score.

The run was started from a converged population made entirely of clones of a single randomly generated individual picked out by us as displaying vaguely interesting behavior (but by no means able to do anything remotely like locate and approach the target). In two runs using this method very fit individuals appeared in less than 10 generations. From a start close to a corner, they would turn, avoiding contact with the walls by vision alone, then move straight toward the target, stopping when they reached it.

5.1.2. Small target. The experiment continued from the stage already reached, but now using a much narrower target placed about ⅔ of the way along the same wall the large target had been on, and away from the robot's starting corner (see Fig. 6), with evaluation ϵ_2

$$\epsilon_2 = \sum_{i=1}^{i=20} (-d_i), \qquad (3)$$

where d_i is the distance of the robot from the center of the target at one of the sampled instances during an evaluation run. Again, the fitness of an individual was set to the worst evaluation score from four runs with starting conditions as in the first experiment. The initial population used was the 12th

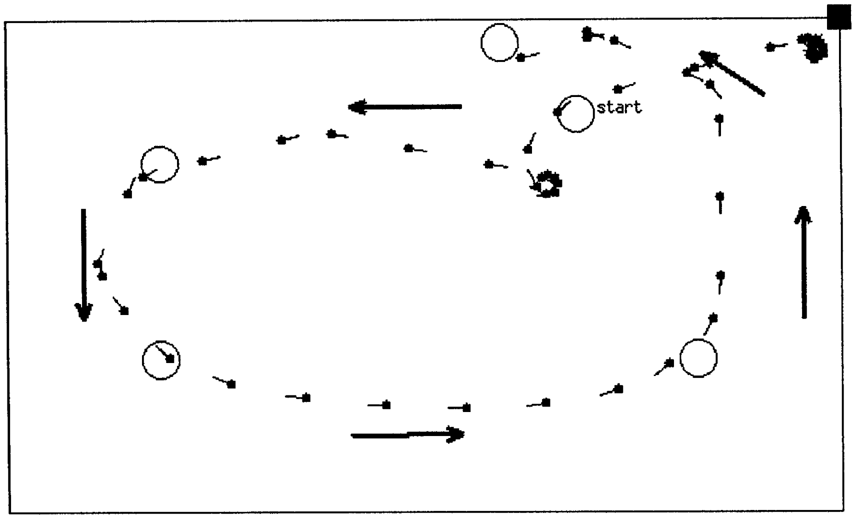

FIG. 7. Tracking behavior of the control system that generated the behavior shown in Fig. 6. The unfilled circles show the position of the target at a number of points on its path (starting position indicated). The arrows roughly indicate the path of the target.

generation from a run of the first experiment (i.e., we incrementally evolved on top of the existing behaviors).

Within six generations a network architecture and visual morphology had evolved displaying the behavior shown in Fig. 6. This control system was tested from widely varying random starting positions and orientations, with the target in different places, and with smaller and different-shaped targets. Its behavior was general enough to cope with all these conditions for which it had not explicitly been evolved. It was also able to cope well with moving targets as shown in Figs. 7 and 8.

5.1.3. Rectangles and triangles. The experiment continued with a distinguish-between-two-targets task. Two white paper targets were fixed to one of the gantry walls: one was a rectangle, the other was an isosceles triangle with the same base width and height as the rectangle. The robot was started at four positions and orientations near the opposite wall such that it was not biased toward either of the two targets. The evaluation function ϵ_3, to be maximized, was

$$\epsilon_3 = \sum_{i=1}^{i=20} [\beta(D_{1_i} - d_{1_i}) - \sigma\,(D_{2_i}, d_{2_i})], \tag{4}$$

where D_1 is the distance of target 1 (in this case the triangle) from the gantry origin; d_1 is the distance of the robot from target 1; and D_2 and d_2 are the corresponding distances for target 2 (in this case the rectangle). These are

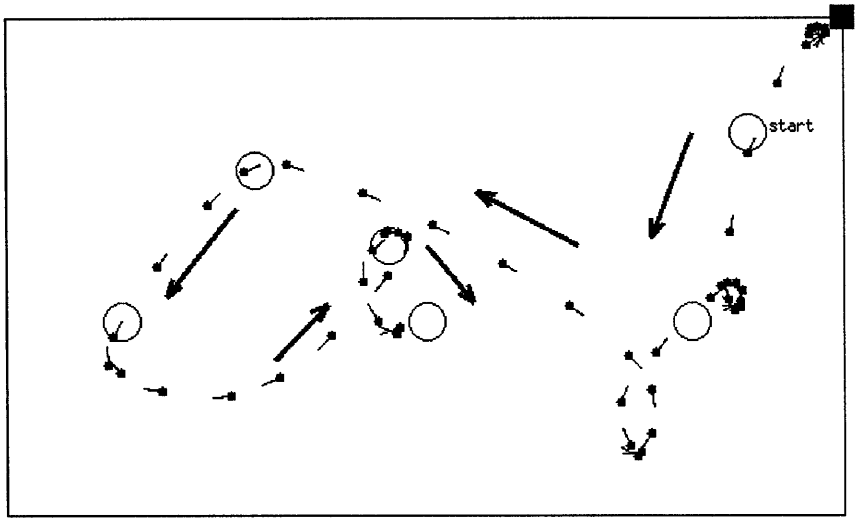

FIG. 8. Further tracking behavior of the control system that generated the behavior shown in Fig. 7.

sampled at regular intervals, as before. The value of β is $(D_1 - d_1)$ unless d_1 is less than some threshold, in which case it is $3 \times (D_1 - d_1)$. The value of σ (a penalty function) is zero unless d_2 is less than the same threshold, in which case it is $I - (D_2 - d_2)$, where I is the distance between the targets; I is more than double the threshold distance. High fitnesses are achieved for approaching the triangle but ignoring the rectangle. It was hoped that this experiment might demonstrate the efficacy of concurrently evolving the visual sampling morphology along with the control networks.

After about 15 generations of a run using as an initial population the last generation of the incremental small target experiment, fit individuals emerged capable of approaching the triangle, but not the rectangle, from each of the four widely spaced starting positions and orientations. The behavior generated by the fittest of these control systems is shown in Fig. 9. When started from many different positions and orientations near the far wall, and with the targets in different positions relative to each other, this controller repeatedly exhibited very similar behaviors to those shown.

The active part of the evolved network that generated this behavior is shown in Fig. 10. The evolved visual morphology for this control system is shown in the inset. Only receptive fields 1 and 2 were used by the controller.

Detailed analyses of this evolved system can be found in Harvey, Husbands, & Cliff (1994) and Husbands (1996). To crudely summarize, unless there is a difference in the visual inputs for receptive fields 1 and 2, the robot makes rotational movements. When there is a difference it moves in a straight

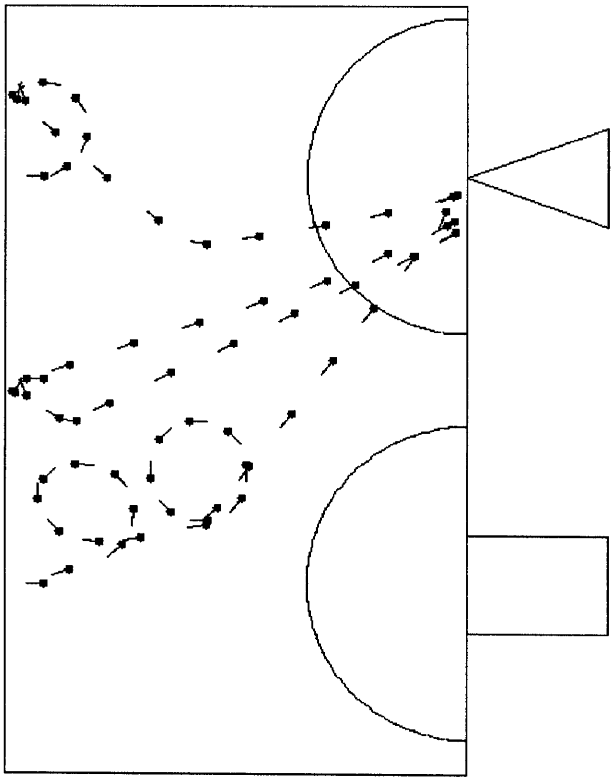

FIG. 9. Behavior of a fit individual in the two target environment. The rectangle and triangle indicate the positions of the targets. The semicircles mark the ''penalty'' (near rectangle) and ''bonus score'' (near triangle) zones associated with the fitness function. In these four runs the robot was started directly facing each of the two targets, and twice from a position midway between the two targets: once facing into the wall and once facing out.

line. The visual sensor layout and network dynamics have evolved such that it fixates on the sloping edge of the triangle and moves toward it.

The case study described above has been included to provide a concrete focus to the issues discussed in this paper. However, this is only one experiment of many, making use of one particular type of network, genetic encoding, and experimental setup. The rest of this paper introduces other aspect of such research.

6. GENETIC ENCODINGS AND DEVELOPMENTAL SCHEMES

Once the decision to evolve network-based systems has been taken, the question of how to encode the networks on an artificial genotype becomes crucially important. Without a suitable encoding scheme little progress can

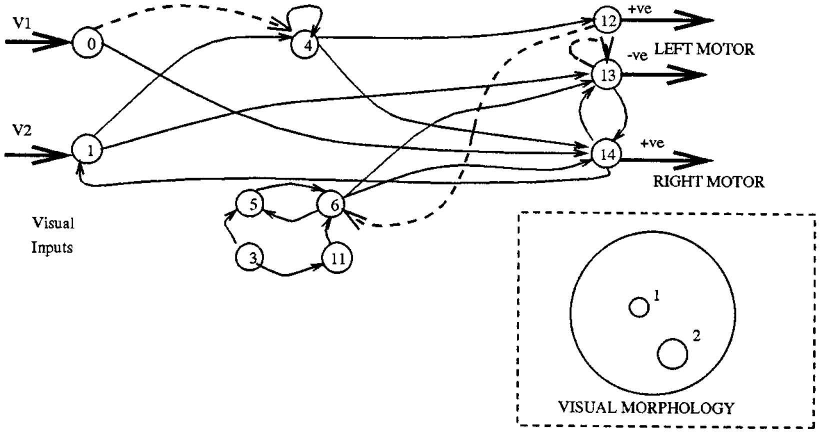

FIG. 10. Active part of the control system that generated fit behavior for the rectangle and triangle experiment. Visual morphology shown in the inset.

be made. In the simplest schemes the genotype is a direct description of the network wiring. An example of that kind of scheme is the genetic encoding used with the gantry robot and described in the previous section. Such encodings will necessarily be restrictive. Much more powerful approaches, allowing complete open-endedness and modularity through the repeated use of genotype sections, must involve a more complex interpretive process.[1] This can be thought of as being loosely analogous to the developmental processes that occur in nature to produce a phenotype from a genotype. Since we regard encoding issues as being central to evolutionary development of control systems, this and the following section concentrate on this area.

Gruau (1992) defines seven properties of genetic encoding of neural networks that should be considered. These include: *completeness,* any NN architecture should be capable of being encoded; *compactness,* one encoding scheme is more compact than the second if for any NN architecture the first genetic encoding is shorter than that given by the second; *closure,* implies that any genotype encodes some architecture; *modularity,* a genetic encoding would be modular if parts of the genotype specify subnetworks of the complete network, and other parts specify the connections between such subnetworks, this decomposition could be recursive. We endorse all these considerations, especially modularity which would seem necessary for sensorimotor systems employing vision. Additional points are: *smooth interaction with*

---

[1] In this context modularity refers to a developmental process analogous to the use of subroutines in programs. For instance, the left limbs and right limbs of animals will not be independently ‘‘coded for’’ in DNA, but rather generated by the same genetic information expressed more than once.

Craig Reynolds (1993) uses genetic programming to create control programs which enable a simple simulated moving vehicle to avoid collisions. He comments that these solutions are brittle, vulnerable to any slight changes or to noise. In further work where the fitness-testing includes noise, he reports that the brittleness problem is overcome, and only compact robust solutions survive (Reynolds, 1994).

Floreano and Mondada (1994) were able to run a GA on a real robot in real time, rather than a simulation. The GA set the weights and thresholds in a simple recurrent network where every sensory input was connected to both motor outputs. The task was to traverse a circular corridor while avoiding obstacles, and this work demonstrates that with well-designed equipment it is possible to avoid the problems associated with simulations.

9. SOME COMMON OBJECTIONS

One common objection to the use of artificial evolution is the amount of time it is likely to take to evolve anything useful. This is difficult to answer. However, work done so far has shown that it is possible to evolve simple control systems in simulation in a matter of 2 or 3 hr (Jakobi et al., 1995) and in the real world in about 1 day (Harvey et al., 1994). It is too early to say how things will scale up as more complex tasks are used.

Another complaint is that the entire morphology of the robot, as well as its control system, should be evolved. This is a valid criticism. Successful adaptive behavior depends on harmonious relationships between body morphology and nervous system dynamics. However, some progress is being made in this direction in the work of Harvey, Husbands, and others where the visual morphology of a real robot is concurrently evolved along with a control network (Harvey et al., 1994). As described earlier, this is done by allowing the subsampling pattern (position and size of receptive fields) of a video camera image to be under evolutionary control (see Section 5).

Sometimes it is stated that as more complex tasks are investigated, it will become extremely difficult to design evaluation functions. Opinion is divided over this issue. One of the implicit assumption of the field is that it is generally much easier to produce a criteria for deciding how well a robot achieved a task than it is to specify how the task should be achieved. Evaluation functions can be very implicit. For instance, tasks such as exploration and foraging can be set up as straight survival tests. Only those robots that maintain viability for sufficiently long get a chance to breed. Maintaining viability will involve finding and exploiting energy sources. However, issues relating to evaluation, both explicit and implicit, are likely to become increasingly important as attempts are made to evolve more complex behaviors.

Finally, a common assumption is that the evolved systems will be impossible to understand. There are two answers to this. The first is that there is no evidence to suggest that the systems will be impenetrable. The second is that `[overpage] even if they are, so what. [discuss!]`