Evolving Robot Morphology and Control

Craig Mautner
Computer Science and Engineering
University of California, San Diego
La Jolla, CA. 92093-0114

Richard K. Belew
Computer Science and Engineering
University of California, San Diego
La Jolla, CA. 92093-0114

Abstract

Most robotic approaches begin with a fixed robot hardware design and then experiment with control structures. We take a different approach that considers both the robot hardware and control structure as variables in the evolutionary process. This paper reports the results of experiments which explore the placement of sensors and effectors around the perimeter of a simulated agent's body, and the Neural Network (NNets) that controls them.

1 Introduction

Evolutionary algorithms have been used in the design of robot controllers for some time and with great success. Almost all work on evolving autonomous agents and robots has focused on evolution of the control structure, often a NNet. This work has taken as a basic assumption that the agent body is immutable. This is a consequence of the flexibility of software over that of hardware: in the past it has not been feasible to explore the space of robot morphology. In contrast to this, the course of natural evolution shows a history of body, nervous system and environment all evolving simultaneously in cooperation with and in response to each other. The research proposed below investigates the interaction between co-evolved body and control structures.

The most successful agents we know of are those found in real life. These agents are well adapted to their environment and can handle many small and large surprises to their world and themselves without failures. Because of this we look to biology for much of our inspiration for this work. However, in contrast to some biorobotic models, we are not trying to reflect the results of our work back to the biology that inspires it.

The methods by which we investigate this is to specify the agent's body and NNet using a grammar. Grammars offer modularity in terms of encompassing detailed structures at various levels of granularity. That is, a grammar can provide a compact representation for complicated and repeated structures. By using grammars we are able to build hierarchical solutions to problems based on the solutions found at lower levels. These grammars are then evolved using common Genetic Algorithm techniques based on the performance of the instantiated agent.

Many aspects of this research have been investigated in isolation by others. Examples of evolved robots that have implemented NNet controllers include six legged walking controllers (Whitley [3], Kodjabachian [8]), maze following (Floreano [5]), predator-prey behavior (Floreano [6]), and food tracking (Angeline [1]).

An early experiment applying grammatical models to the construction of feed-forward NNets is due to Kitano [7]. Other researchers who have used grammars to develop NNets are Whitley and Gruau [3] and Lucas [10].

Very little prior work in evolving morphology exists. Menczer and Belew [12] investigated the nature of sensor usage by providing their agents with an evolvable NNet connected to sensors and effectors. Marks, Polani, and Uthmann [11] explored eye types and positioning. Sims [13] demonstrated a simulation where the complete morphology of the individuals was involved. Sims created an artificial world in which each agent was grown from a genome that defined both the physical structure and the control structure. Eggenberger [4] has developed an evolutionary system that simulates the growth of a body based on differential gene expression. Lee et. al. [9] have also worked on evolving both control structures and body plans.

2 Background

We have carried out several experiments based on the paradigms listed above. The experiments begin with a population of genome strings which are converted to individual grammars. The grammars are used to generate agent bodies and their associated NNet control

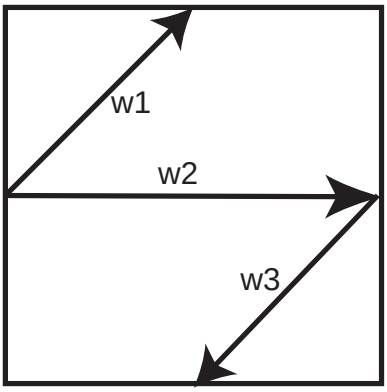

Figure 1: Terminal Cell Example

systems. Through a development process that transforms a single undivided cell (the gamete) into a body consisting of several interconnected cells.

The body/controller pair is then evaluated in the context of an environment and the genomes for the more successful individuals are preferentially selected for reproduction in the next generation.

The process of cell division, neural network extraction, and fitness evaluation are described in the next sections.

2.1 Production Rules

The grammar consists of an alphabet of terminal and non-terminal symbols, a set of production rules, and a single starting symbol from the set of non-terminals. In our experiments the terminals are the hexadecimal characters {0-9a-f}, and the non-terminals are a subset of the uppercase characters {A-Z}.

Each production rule is made up of a left side and a right side. The left side is a single non-terminal symbol; the right side is a specification for the creation of one or two cells. A non-terminal may appear as the left side of more than one production rule.

The specification of each cell defines it to be either a single non-terminal symbol or a terminal cell which is a list of directed, weighted edges from the sides of the cell to each other (See Figure 1). The production rule also contains the orientation of the cell (whether it is horizontally or vertically flipped). Specifying a cell to be a non-terminal allows it to be subject to the application of further production rules.

A production rule also specifies whether the non-terminal produces one or two cells, and if two, the relative position (i.e. above, below, to the left of, or to the right of) of each.

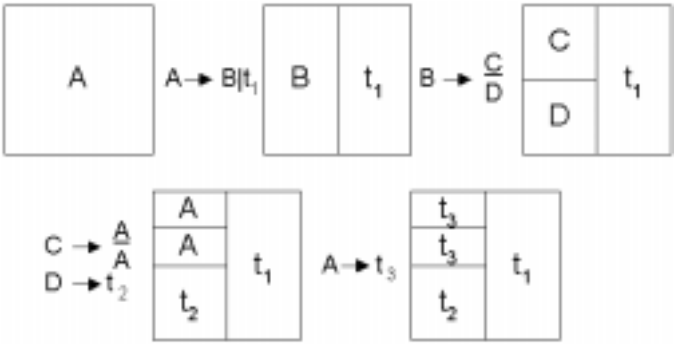

Figure 2: Development Process

2.2 Cell Division

As described in the previous section, each cell is either labeled with a non-terminal symbol or is a terminal cell. Initally the gamete is labeled with the starting non-terminal symbol of the grammar and cell differentiation proceeds by selecting and applying the rules of the grammar. For each cell labeled with a non-terminal, a rule is found whose left side matches the non-terminal. The cell is then replaced with the one or two cells specified by the right side of the rule. If there is no matching rule, the cell is replaced with a terminal cell with random weights and edges.

The process continues replacing non-terminals with terminals and non-terminals until there are only terminals left or a maximum depth of replacements have occurred. The rules are applied only a limited number of times to keep rules of the form A→A from generating an infinite regress. In our experiments, the maximum number of cell divisions was set to 6. This permits a body to have at most 64 cells (one cell divided in half 6 times produces $2^6 = 64$ cells). A derivation that continues beyond the sixth rule application will replace the cell by a terminal cell with random weights and edges.

A derivation that takes four generations is shown in Figure 2. In this example the gamete is labeled with the starting symbol, A. The production rule A→B|t_1 indicates that the non-terminal A is converted into two cells. The first cell, a non-terminal B, is to be placed to the left of the second which is the terminal cell, t_1.

2.3 Neural Network Interpretation

Once the cell division is complete, the body consists of a set of cells that have within them directed, weighted edges. The cells and edges are interpreted as sensors, effectors and the neural networks that connect them.

Each side of a cell has associated with it two values,

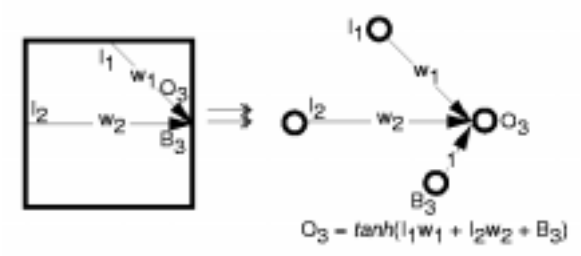

Figure 3: NNet edges formed from within cell connections

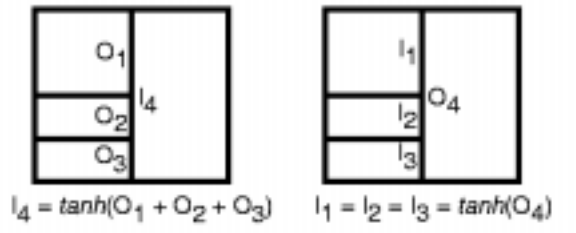

Figure 4: NNet edges formed from connections between adjacent cells

an input value and an output value. The input value is dependent on the output values of all adjacent cells. The output value of a side is dependent on the input value of all sides and the internal weighted edges between the sides. Both the input and output values are described in detail below.

2.3.1 Network Edges Within a Cell

There are three types of edges that can exist within the cell: Corner, Cross, and Tonic. Corner edges run from one side to an adjacent side. Cross edges run from a side to the opposite side and Tonic edges are interpreted as bias weights. The three types are shown in Figure 3. The edge from I_1 is of corner type with weight w_1, I_2 is of cross type with weight w_2, and T_3 is of type tonic providing activation of value T_3.

The output of a cell is the squashed sum of the weighted inputs to that cell. The weighting is only done across those edges that are defined for that cell. In Figure 3 there are two edges and one tonic defined for the cell pictured. The output of the cell on the right edge is the squashed sum of the products $I_1 w_1$, $I_2 w_2$ and the Tonic weight T_3. For all of our experiments the squashing function is the hyperbolic tangent function: $\tanh(x)$.

2.3.2 Edges Between Cells

When one or more cells are adjacent to the side of a cell, the outputs of the incident cell are combined to form the inputs to the adjacent cell. This is demonstrated in Figure 4.

In this case the inputs labeled I1, I2, and I3 are adjacent to the output labeled O4. Similarly, the input labeled I4 is adjacent to the outputs labeled O1, O2 and O3. A neural network layer is formed by setting the input value on an edge of a cell equal to the squashed sum of the outputs of the cells adjacent to that edge.

Figures 7, 9, 10 and 11 illustrate some final agents' terminal cell divisions and the neural networks that result. In these diagrams the semicircles pointing outwards are effectors and those semicircles pointing inwards are sensors.

2.3.3 Sensors and Effectors

Sensors and effectors are defined by the edges of the terminal cells. Any directed edge that originates from a cell side that is on the perimeter of the body becomes a sensor or input node. Any directed edge that terminates on a cell side that is on the perimeter of the body becomes an effector or output node.

Sensor nodes detect signals of the environment. They provide the input that is propagated through the NNet of the body. Effector nodes are the outputs of the NNet.

Effector nodes provide propulsion to the agent's body. The force of this propulsion is proportional to their output activation. The direction of propulsion of the agent is a result of summing all of these forces based on their position on the body. The vector sum of all effector outputs is broken into two forces: The first is a pressure, which acts through the center of the body and translates the body through the environment. The second force is torque which acts perpindicular to a line through the center of the body and causes the body to rotate. The net effect is demonstrated in Figure 5.

2.4 Application in a Simple Environment

In order to judge the effectiveness of a given body plan, and hence the effectiveness of the genome from which the body plan developed, the mature agent is placed within an environment. For our experiments the environment consists of a 500x500 world that has at its center a source of reward which produces a de-

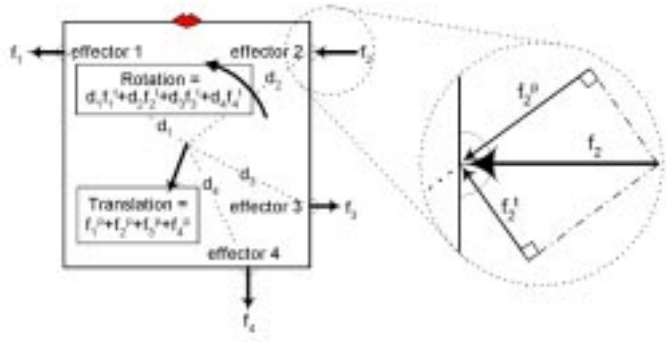

Figure 5: Conversion of Effector Outputs to Rotation and Translation

tectable signal that falls off as the inverse-square of the distance.

All agents' bodies are set to be 20 units on each edge. The body has a mouth placed at its top. The fitness is measured from the mouth. The perfect agent would turn it's mouth toward the center of the world and approach the center in as few time steps as possible.

The agents are placed at a random location in the world and then allowed to wander freely (or to just sit in the case of many agents) for 30 time steps. After 30 time steps the agent is moved to another random location. This is repeated for 10 placements for a total of 300 time steps. All agents in a given generation are started from the same set of locations. If an agent gets within a body length of the center of the world then it is moved the next location.

The fitness, f_i, at time step i is shown below where d is the distance from the agent's mouth to the center of the world.

$$f_i = \begin{cases} \frac{1}{d^2} & \text{if } d \geq \text{agent body length} \\ 1 & \text{otherwise} \end{cases}$$

The fitness, F, of an individual is the sum of the fitnesses at each time step over its lifetime.

2.5 The Braitenberg Vehicle

A classic design in the field of robotic control is the Braitenberg Vehicle 2b described in [2]. This agent has two sensors on the front and two effectors on the back. The agent's body is bi-laterally symmetric with each sensor connected via a positive weight to the effector on the opposite side. The effect of this connectivity is to steer the agent to the side with the stronger sensor. The effectiveness of this design has been demonstrated in a number of robots.

Figure 6: Grammar for Generating Braitenberg Vehicle

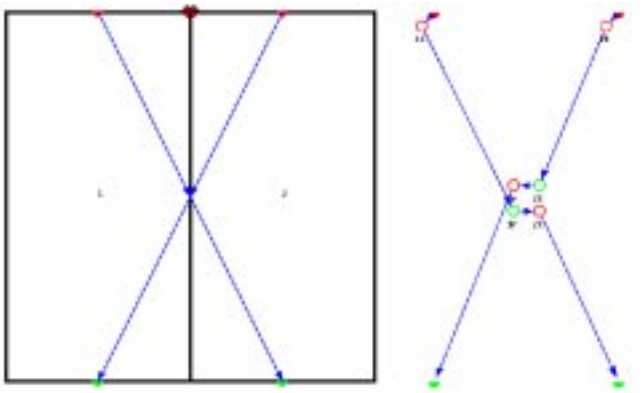

Figure 7: Braitenberg Vehicle with Cell and NNet Representations

A simple grammar that generates a complete Braitenberg body is shown in Figure 6. The grammar consists of two rules. The first rule rewrites the undifferentiated cell body (start symbol A) into two nonterminal B cells one of which is horizontally flipped relative to the other. The second rule converts a B nonterminal cells into a terminal cell with two edges defined. Figure 7 shows Braitenberg's Vehicle 2b alongside the one generated by this hand-designed grammar. The parsimonious nature of the grammar that generates the Braitenberg Vehicle under our system shows the representational adequacy of the grammar system.

2.6 Preliminary Results

Using the experimental platform described above we ran several experiments to determine the effectiveness of our approach. Three such results are described in the graph of Figure 8. The results show the average of the best agent over ten different runs with the same starting conditions. The first experiment is our hand-designed Braitenberg agent which is run in the environment for 100 generations. Its performance is shown by the short line around a fitness of 26. This experiment provides a baseline with which we may compare other results.

The top line of the graph begins with the same Braitenberg agents of the first experiment, but in this

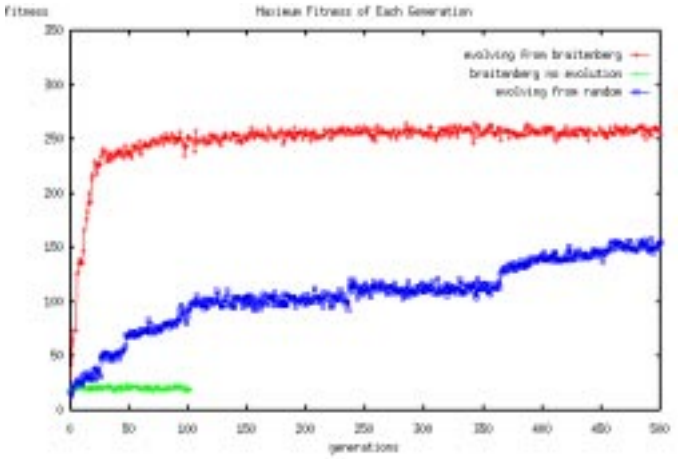

Figure 8: Average of Fitnesses (Best of Each Generation)

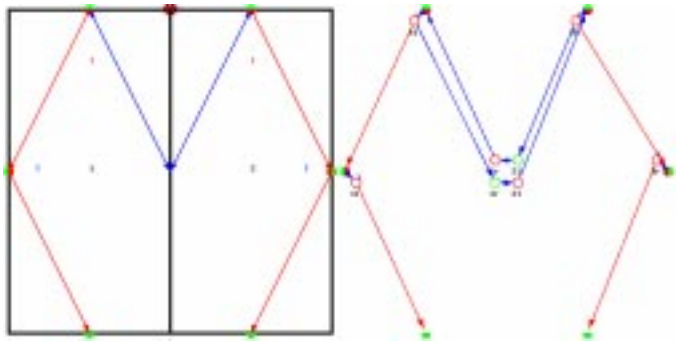

Figure 9: Agent and NNet evolved from Braitenberg

experiment they are allowed to evolve. Their performance shows significant improvement when compared to the original run of Braitenberg agents. All ten populations converged by the end of the run, although they each converged on different solutions. One commonality of all solutions was the preservation of the original *bi-lateral symmetry* that was part of the Braitenberg population prototype. However in about half of the final populations the connections were positive and crossed as in the Braitenberg but, surprisingly, the remaining populations converged on solutions that had negatively weighted connections straight down the body rather than across it. Most of the solutions also added two negative bias weights to the front of the agent which acted as tractor effectors pulling the body forward constantly. This left new sensors and effectors on the side free to steer the body towards the goal. A typical solution is shown in figure 9.

The final experiment (middle line) initialized the

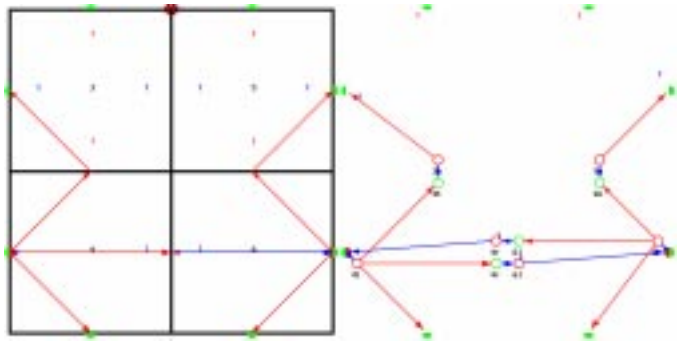

Figure 10: Agent and NNet evolved from Random Genome(Ex. 1)

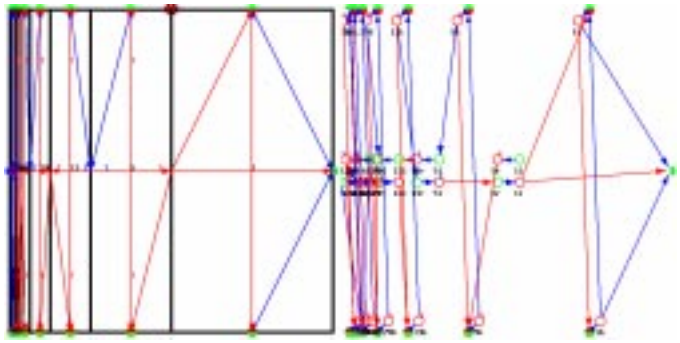

Figure 11: Agent and NNet evolved from Random Genome (Ex. 2)

population with random genomes. In this case results varied widely. Some runs were able to find steadily improving solutions within a few hundred generations while others took quite a bit longer. Some solutions found were Braitenberg-like in that they discovered bi-lateral symmetry (figure 10). Other solutions produced bizarre body types that performed very well (figure 11).

3 Summary

The conclusion that we can draw from these results is that even in this very simple world a wide range of potential solutions exist. Many of these solutions are superior to the known good *a priori* Braitenberg Vehicle solution. It was shown that we can evolve agents whose body and control structures are tightly coupled in order to solve the simple environment presented here. The fact that the system found so many solutions is encouraging as we apply it to more complex environments.

Our current work involves applying these tech-

niques to real robots including Kheperas and LEGOBots whose body shapes and sensor and effector placement will be determined by the evolved grammars.

References

[1] P. Angeline, G. Saunders, and J. Pollack. An evolutionary algorithm that constructs recurrent networks. *IEEE Trans. on Neural Networks*, 5:54–65, 1994.

[2] V. Braitenberg. *Vehicles: Experiments in Synthetic Psychology*. MIT Press, Cambridge, 1984.

[3] F. Gruau D. Whitley and L. Pyeatt. Cellular encoding applied to neurocontrol. In L. Eshelman, editor, *International Conference on Genetic Algorithms*. Morgan Kaufmann, 1995.

[4] P. Eggenberger. Evolving morphologies of simulated 3d organisms based on differential gene expression. In Phil Husbands and Inman Harvey, editors, *Fourth European Conference on Artificial Life*, pages 205–213. MIT Press, 1997.

[5] D. Floreano and F. Mondada. Automatic creation of an autonomous agent: Genetic evolution of a neural network driven robot. In J.-A. Meyer D. Cliff, P. Husbands and S. Wilson, editors, *From Animals to Animats III*, Cambridge, MA, 1994. MIT Press.

[6] D. Floreano and S. Nolfi. Adaptive behavior in competing co-evolving species. In Phil Husbands and Inman Harvey, editors, *Fourth European Conference on Artificial Life*. MIT Press, 1997.

[7] H. Kitano. Designing neural networks using genetic algorithms with graph generation system. *Complex Systems*, 4(4), 1990.

[8] J. Kodjabachian and J. A. Meyer. Evolution and development of neural networks controlling locomotion, gradient following, and obstacle avoidance in artificial insects. http://www.biologie.ens.fr/fr/animatlab/perso/-kodjaba/kodjaba.html, 1997.

[9] W. Lee, J.C. Hallam, and H.H. Lund. Hybrid gp/ga approach for co-evolving controllers and robot bodies to achieve fitness-specified tasks. In *Proceeding of IEEE 3rd International Conference on Evolutionary Programming*, pages 384–389, New York, 1996. IEEE Press.

[10] S. M. Lucas. Growing adaptive neural networks with graph grammars. In *Proceedings of European Symposium on Artificial Neural Networks (ESANN '95)*, pages 235–240, 1995. http://esewww.essex.ac.uk/ sml/papers.html.

[11] Alexandra Mark, Daniel Polani, and Thomas Uthmann. A framework for sensor evolution in a population of braitenberg vehicle-like agents. In Christoph Adami, Richard K. Belew, Hiroaki Kitano, and Charles E. Taylor, editors, *Artificial Life VI*, pages 428–432, Cambridge, Mass, 1998. MIT Press.

[12] F. Menczer and R. K. Belew. Evolving sensors in environments of controlled complexity. In R. Brooks, editor, *Proc. Fourth Conf. on Artificial Life*, 1994.

[13] K. Sims. Evolving 3d morphology and behavior by competition. In R. Brooks and P. Maes, editors, *Proceedings of the International Conference Artificial Life IV*, Cambridge MA, 1994. MIT Press.