Combining Genetic Algorithms and Neural Networks: Evolving the Vision System for an Autonomous Robot

Matthias Scheutz and **Thomas Naselaris**
Indiana University Bloomington, Indiana

Abstract

In this paper we describe the vision system of a robot which has to accomplish a path following task. The vision system combines three different learning methods: reinforced, competitive, and supervised. A genetic algorithm is used to determine the regions of the visual field essential for discriminating the path from its surroundings. These regions, in turn, serve as input for a neural network that categorizes the information present in the visual stimulus. The output of the network is used by a motor system to navigate the robot on the path.

Introduction

Robot models have recently gained much attention in cognitive science for many reasons, probably the most important one being that they are the "ultimate" touchstone for cognitive theories. In many important respects, the life of a robot mimics much more faithfully the life of the human/animal than does the life of a simulated agent or experimental subject. Even in the performance of a simple task, robots and other physical agents must be able to cope with a variety of challenges which the agent in a simulated environment may able to ignore. For a robot to behave appropriately in an environment that was not carefully designed by an experimenter, it must first be able to identify information in its environment that can be exploited to provide a useful basis for action. It must also be able to reliably detect this information in the face of changing circumstances and imperfect stimuli. The situation is quite different for an agent which operates in a simulated environment that is well-behaved and fully-described, that is, an environment which provides perfectly accessible information, all of which relates directly to the agent's goals. An agent which exhibits perfect behavior in such an environment might completely fail in the "real world", where the outcome of a robot's actions may be affected by a multitude of factors that the robot's designer understands but poorly and cannot control.

Another one of the problems which physical agents must contend with is the successful integration of perception, cognition, action, and learning. It is possible, in the laboratory or simulated environment, to ignore the dependencies among these activities, and study each of them in isolation. Yet it is clear that for an agent to perform any non-trivial task in the real world, these activities cannot take place independently, but must constrain one another in essential ways. The robot model thus provides an arena in which theories of perceptual, cognitive, motor or learning processes may each be evaluated in the context of the other. It is unarguably one the primary goals of artificial intelligence (AI) to understand how these processes interact to produce interesting behavior.

There are many other theoretical arguments which could be made in favor of physical agents. However, there is a final one that we would like to underscore in this context, which, although very often neglected by people who are merely concerned with simulations, we believe to be one of the crucial criteria in modeling cognitive systems: a dynamic environment requires an agent to function under real-time constraints![1] And these constraints exclude a major bulk of learning and programming techniques because of their time-intense nature.

In the following, we describe the control system for a robot that is designed using a combination of a genetic algorithm and neural networks. The robot operates in the "real world" (a forest, in this case), and solves a path-navigating task that brings to bear on the robot each of the challenges just discussed. We present results of tests of the system's performance in categorizing raw visual stimuli, and in autonomously navigating a path. Finally, we discuss related work by others, and some possible extensions our own work.

[1] The important consequences of this fact are described in (vanGelder & Port 1995).

The path following task

Motivated by our interest in the effect of real world constraints on cognitive systems, we wanted to choose a task for a physical agent, which, while being manageable from an engineering and programming point of view,

1. is computationally feasible
2. is at the same time biologically plausible
3. integrates different learning methods
4. lends itself naturally to further investigations of higher level cognitive faculties

In view of these criteria, we have defined a path following task for our robot model, in which the robot is required to navigate on red brick paths leading through a small forest in front of the computer science department at Indiana University, Bloomington (IUB). (see figure 1). The main challenge for the robot in this task lies in processing the "real world" visual input: not only does the path have no uniform color, but depending on various circumstances (such as the weather, the time of the day, the foliage on the path, etc.) the path looks very different, and what is part of the path or just part of the surrounding forest cannot be determined by simple rules (which could be easily done in a simulation!). It is thus obvious that certain parts of the visual field are more important and relevant for detecting the path than others (depending on the perspective, the position of the camera above the path, etc.). Therefore, the visual system needs to be able to compute a *reliable estimate* of which part of the visual field counts as path in order to allow the robot to navigate successfully through the woods.

Figure 1: The path

The robot

The robot, which we are using for our project, is currently under development by a group of graduate students at IUB. It is intended as a multi-purpose platform for and sponsored by the IUB cognitive science program to test cognitive models that need a physical agent. It consists of a three-wheeled body, carrying a PC as well as the batteries that supply two motors for locomotion. A head with a camera and two microphones is mounted on the body allowing the robot to look around and localize sounds. In addition, speakers will be added later to permit speech production.

The model

In this section, we will first give a brief overview over the program that controls the robot and then describe the vision system in detail.

The basic structure

The program, guiding the robot's actions, consists of two major parts: the vision system and the behavioral (i.e., motor control) system. Both systems run in parallel and asynchronously (Brooks 1991). The motor system processes the output of the visual system, leading to motor actions, which in turn change the input to the visual system, thereby establishing a coupling between the robot and the environment.

The vision system

Visual input is received from a camera mounted above the robot's two front wheels. While navigating, the camera collects approximately 3 to 4 images per second, which–encoded as matrices of gray-scale values–are sent to the on-board PC. There, the images are used as inputs to an array of "virtual" sensors. Each sensor is responsible for reporting information about a particular region of the robot's visual field, and so attends to a single connected subset of the image matrix. Thus, a sensor array is determined by a set of vectors $S = \{(x_1, y_1, r_1), \ldots, (x_n, y_n, r_n)\}$, where x_i and y_i determine the image-matrix entry about which the i-th sensor is centered and r_i, the radius, determines the size of the region (around (x_i, y_i)) covered by the i-th sensor. The output of the sensor array for a given image matrix is then a vector $v = (mean_1, std_1, \ldots, mean_n, std_n)$, where std_i is the standard deviation of gray-scale values for matrix entries covered by the i-th sensor, and $mean_i$ is the average gray-scale value over entries covered by the i-th sensor. The standard deviation of gray-scale values is used to provide a rough model of texture differences between regions of the robot's visual field. The path which the robot navigates is composed entirely of red brick, so

that differences in pixel values about a small neighborhood of any point on the path are almost always negligible. Surrounding the path is a forest, where a rich variety of shapes and colors serve to increase the possibility of deviation within a given region significantly.

The output vectors of the sensor array serve as input to a 3-layer, feed-forward neural network. The hidden layer uses a clustered competitive learning algorithm to determine regularities in the visual field. It learns to attend to "blobs" in the visual field using the mean and standard deviations of neighboring regions to constrain sensor data. A "backprop" layer, in turn, uses a back-propagation learning algorithm (hence the name) to "translate" the blobs of the hidden layer into a few general categories such as "right turn", "straight path", "junction", etc.

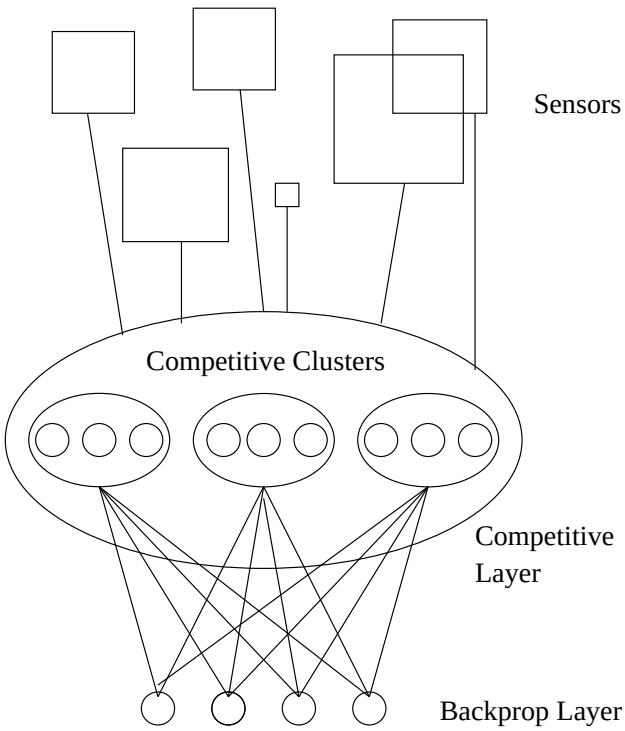

Figure 2: The structure of the vision system

Motor system

Input from the backprop layer provides for a simple, non-adaptive scheme for generating motor commands. As mentioned, the backprop layer translates visual stimuli into simple categories that can be used to provide prescriptions (e.g., turn right, turn left, stop) for the robot's movement. Given the proper encoding, these prescriptions can be sent directly to the motors to produce an action. Although a variety of possible commands can be tested over different trials, the robot can only select from among a fixed set of commands while actually navigating.

Evolving the vision system

Learning in the system splits into two separate phases: and off-line genetic learning phase, during which the visual system is "evolved", and an on-line adaption phase, during which the robot learns to generate behaviors that will allow it to stay on the path. In the following, we will focus only on the vision system.

The goal of the genetic algorithm is to evolve a sensor configuration (together with the neural network processing it) such that the motor system can use the output to generate behaviors that will allow the robot to stay on the path. This aspect of the design is in keeping with ecological principles (Gibson 1974), as it equips the agent with a perception system that is well adapted to its environment, and particularly well-suited to the task for which it is used (Beer 1990). The choice is also sound from an engineering standpoint, for although we have set up a "hand-made" sensor array to test the behavioral system, we were not convinced that this array, being merely the result of analyzing the constraints of the robot, is the best possible.

The challenging problem in our case was to come up with a fitness function which assesses the functionality of the sensor array. The best possible would be, of course, an actual test run on the robot, but this is obviously not feasible. So a neural network is used with real images of the path together with teacher values that describe (in human categories) all possible directions that the robot could move in. The following steps have to be taken for every generation in the GA:

1. The GA generates a population of sensor arrays. The initial population is generated by assigning random values to all sensor parameters (array size is not allowed to vary, it is fixed for each complete run of the GA).

2. A 3-layer neural network with small random weights is created for every sensor in each array. Then the average grey value and the standard deviation are computed for every image. In our current implementation, these values are paired with "teacher" values, each corresponding to one of three movement categories: "right" and "left".

3. The input patterns to the network are computed depending on the particular sensor array of the individual. Every input pattern is paired with a predefined teacher pattern to train the network.

4. The hidden layer of the network is trained for a given number of epochs using competitive learning in clusters.

5. The output layer of the network is trained for a given number of epochs using back-propagation. The teacher patterns serve to determine the error signal which is then used to adjust the weights from the hidden layer to the output layer. After training, the network is tested on these patterns and the difference in output between the predefined categories and the ones learned by the network are used to determine the fitness of the sensor array. Specifically, fitness is given by

$$fitness = 1/\sum_{pat=1}^{k}\sum_{i=1}^{n}(teacher_{pat,i} - output_{pat,i})$$

where k is the number of images and n the number of output nodes.[2]

6. When a fitness value has been generated for each member of the population, the GA constructs a new population using three standard genetic operators: selective reproduction, crossover, and mutation. Parameter values for the "fittest" sensor-array and its associated network are recorded. When the GA halts, the array with the highest fitness value over every generation is selected for testing.

This method of evolving sensor arrays which allow for network classification of inputs contributes to the robot's success mainly via its effect on the input space. By adjusting the placement and size of individual sensors, the GA manages to locate those areas of the visual field which consistently contain salient, stable information about the robot's environment. By fixing sensors in these areas, the GA constrains the robot's perception of the world in much the same the way as an attention mechanism does in other organisms. The competitive layer then searches for regularities in the restricted input space, thus further reducing the volume of input data. In absence of these constraints, the sheer number of states of the world to be interpreted by the robot might well preclude the possibility of its generating useful actions on the basis of sensory input.

[2]At first glance, this seems to be an instance of the "credit assignment problem". It turns out, though, that if the hidden layer classification is "good enough", back-propagation can always learn to map it onto the predefined categories. Therefore, we concluded, that those sensor arrays where the representation in the hidden layer are useful, but where backprop fails to achieve a reasonable mapping, can be ignored without too great a loss.

Results

We now describe the performance of several of the evolved arrays when used to categorize visual input, and to navigate the robot down the path. We provide specific examples of the interactions between certain configurations of the sensors, and the ability of the network to correctly classify inputs given these configurations.

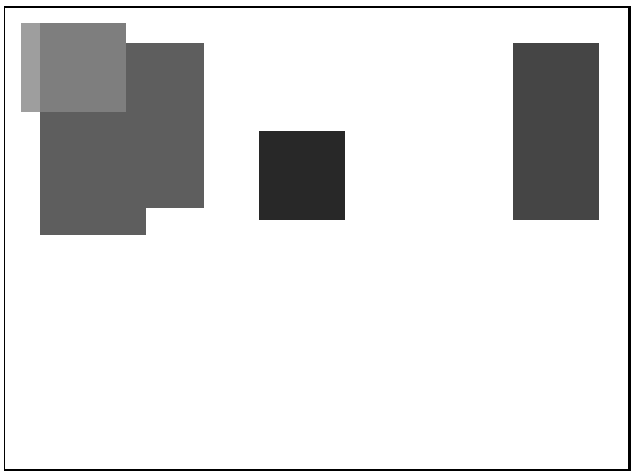

Figure 3: The fittest sensor array

In order to assess the potential effectiveness of the sensor arrays prior to their use in the navigation task, "movies" of the path environment were recorded while driving the robot manually down the path. Movies were taken in a variety of light conditions, and consisted entirely of images which were not present in the training set of the evolved array. These images were then paired with appropriate teacher values, and fed to a network simulator which displayed the output of a trained network for a given image when coupled to an evolved array.

Figure 3 depicts an evolved array which had highest overall fitness after a 10,000 generation run of the GA on a 20-image training set.[3] The teacher values used during the run were "right turn" and "left turn". A network coupled with this sensor array correctly classified 360 separate images.

The array depicted in figure 4 is a "hand-coded" array. Networks coupled to this array managed to classify roughly one third of the movie-file images correctly. These results clearly indicate the network's dependence on an "intelligent" selection of inputs from the input space. Note that the majority of the sensors in the evolved array are clustered in the upper left and right hand corners of the visual field–precisely the location

[3]Notice that individual sensors can overlap.

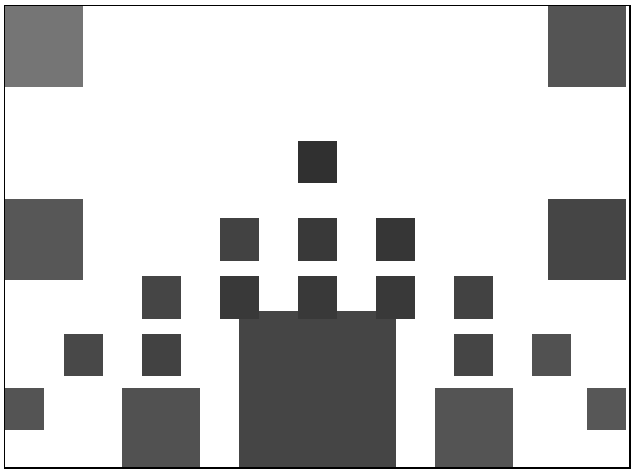

Figure 4: The hand-coded sensor array

where the presence or absence of vegetation indicates, to the human eye at least, the dividing line between forest and path. It also not unreasonable to hypothesize that the lack of centrally-placed sensors reflects a response to the "noisiness" (in the form of glare, see figure 5) of the central region of the visual field.

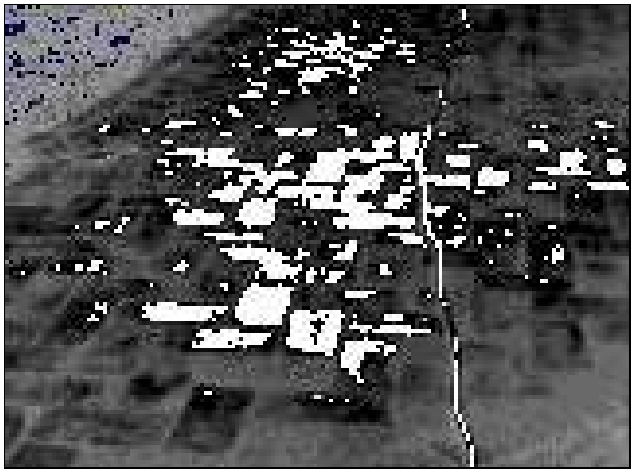

Figure 5: Glare on the path

But the proof of the pudding, so to speak, is demonstrated by the robot's ability to perform the task for which it was designed. Coupled with the array in figure 3, a network that was trained for 1,000 epochs on a 20-image training set was installed on the robot's on-board PC. Under the control of this coupled array/network, the robot successfully traversed the forest path, covering some 150 yards, in 2 out of 5 separate trials. This performance can be compared to tests run with the "hand-coded" sensor array, which, when coupled with a network given the same amount of training, failed to navigate the path for more than 30 feet on any one of five trials.

Related work

The application of genetic algorithms to problems in robotic control has generated a number of successful results (Floreano & Mondada 1996; Beer 1990; Colombetti, Dorigio, & Borghi 1996), particularly where a GA has been used in conjunction with a neural network. Our use of a GA/neural network combination differs from many previous applications in three notable respects: firstly, we have used a GA not to optimize connection weights of the network (Floreano & Mondada 1996; Beer 1990; Law & Miikkulainen 1994; Cliff, Husbands, & Inman 1992) but rather to constrain the network's input space, thus optimizing the effectiveness of further network learning procedures. A second distinction, shared by (Cliff, Inman, & Husbands 1996) and (Cliff, Husbands, & Inman 1992), lies in our utilizing the GA in the construction of a sensory system that is well-suited to the robot's task, as opposed to the construction of a motor system that must accomplish a given task, using a representation of the environment that is fixed by the designer. Finally, the network training algorithm we employ is distinctive in it's integration of self-organizing and supervised learning procedures.

A comprehensive overview of current practices and approaches in evolutionary robotics can be found in (Mataric & Cliff 1996) and (Nolfi *et al.* 1994). Convincing arguments for the necessity of evolving robot controllers for complex tasks are given by (Cliff, Husbands, & Inman 1992). For further examples and discussions of adaptive robotics, broadly construed, see (Kaelbling 1993) and (Nehmzow & Mitchell 1995). Finally, the important links between robotics and other branches of AI and Cognitive Science are explored in (Harnad 1995) and (Brooks 1991).

Conclusions

A major goal of our "path-following project" was not only to explore the effectiveness of a novel learning application, but also, in accordance with the ideas discussed in the introduction, to further tighten the links between research in robotics and the rest of AI. An agent which can navigate autonomously in an environment as complex as the one we have chosen would provide an ideal platform from which to launch investigations of higher-level cognitive activities in the "real world". Yet our pursuit of such an agent has already generated some results which might be of some interest to the concerns of cognitive science. In particular, we have demonstrated that, at least in this instance, an

agent's perception of its environment cannot be arbitrary, but must be crafted by an adaptive process to conform to the task at hand. As was shown above, an appropriate vision system is *conditio sine qua non* for the robot, because all behaviors, i.e., motor commands, hinge upon its interpretation of the world.

In the future, we intend to replace the primitive motor system used in these experiments with one that will allow the robot to adaptively generate motor commands. Once we can elicit reliable, independent path-following, we may begin making progress towards an overarching goal: the evaluation and design of cognitive theories for agents in the "real world".

Acknowledgments

The genetic algorithm used in our experiments was constructed from "Matthew's GA Library", to be found at *http://lancet.mit.edu/GA*. Funding for the robot used in these experiments was provided by the IU Cognitive Science Department. We would also like to thank the IU Robot Group for their ideas and technical support.

References

Beer, R. 1990. *Intelligence as Adaptive Behavior: an experiment in computational neuroethology.* San Diego: Academic Press.

Brooks, R. A. 1991. Artificial life and real robots. In *Proceedings of the Workshop on Evolution and Chaos in Cognitive Processing (IJCAI-91).*

Cliff, D.; Husbands, P.; and Inman, H. 1992. Evolving visually guided robots. In *Proceedings of the Second International Conference on Simulation of Adaptive Behavior.* Cambridge: MIT Press Bradford Books.

Cliff, D.; Inman, H.; and Husbands, P. 1996. Artificial evolution of visual control systems for robots. *Forthcoming.*

Colombetti, M.; Dorigio, M.; and Borghi, G. 1996. Behavior analysis and training: A methodology for behavior engineering. In *Transactions on Systems, Man, and Cybernetics*, volume 26(6).

Floreano, D., and Mondada, F. 1996. Evolution of homing navigation in a real mobile robot. *IEEE Transactions on Systems, Man, and Cybernetics* 26.

Gibson, J. 1974. *The Perception of the Visual World.* Westport, Conn.: Greenwood Press.

Harnad, S. 1995. Grounding symbolic capacity in robotic capacity. In Steels, L., and Brooks, R., eds., *The "artificial life" route to "artificial intelligence". Building Situated Embodied Agents.* New Haven, Conn.: Lawrence Erlbaum.

Kaelbling, L. P. 1993. *Learning in Embedded Systems.* Cambridge, Mass.: MIT Press.

Law, D., and Miikkulainen, R. 1994. Grounding robot control with genetic networks. Technical Report AI94-223, Dept of Computer Sciences, Univ. of Texas at Austin.

Mataric, M., and Cliff, D. 1996. Challenges in evolving controllers for physical robots. *Forthcoming.*

Nehmzow, U., and Mitchell, T. 1995. The prospective student's introduction to the robot learning problem. Technical Report UMCS-95-12-2, University of Manchester, Manchester.

Nolfi, S.; Floreano, D.; Miglino, O.; and Mondada, F. 1994. How to evolve autonomous robots: Different approaches in evolutionary robotics. Technical Report PCIA-94-03, Dept of Cognitive Processes and Artificial Intelligence, Viale Marx.

vanGelder, T., and Port, B. 1995. *Mind as Motion.* Cambridge: MIT Press.