BIOLOGICALLY INSPIRED AUTONOMOUS SYSTEMS
Computation, Cognition and Control with applications to Robotics, Imaging and Large Scale Networks

Duke University – March 4, 5 and 6, 1996

Co-Sponsors: Triangle Area Neural Networks Society, International Federation of Information Processing Societies WG 6.3 and WG 7.4, ITC Mini-Seminar Series, Office of the Vice-Provost for Academic and International Affairs at Duke, and the Duke University Departments of Electrical and Computer Engineering, Psychology - Experimental, Biomedical Engineering, Neurobiology, and the NSF-ERC

Sunday, March 3 – 6:00pm Washington Duke Inn
Registration and cash bar

Monday, March 4 – 8:00- 8:45 Registration
8:45- 9:00 Erol Gelenbe and Nestor Schmajuk - Welcome

9:00- 9:30 Jean-Arcady Meyer (ENS, Paris)
From Natural to Artificial Life

9:30- 10:00 Heinz Muehlenbein (GMD, Bonn)
Inspiration from Nature vs. Copying nature. Lessons Learned from Genetic Algorithms

10:00- 10:50 Stephen Grossberg (Boston U)
Are there Universal Principles of Brain Computation?

10:50-11:30 Discussion and Coffee

11:30-12:00 Anil Nerode (Cornell, Ithaca)
Hybrid Systems as a Modelling Substrate for Biological and Cognitive Systems

12:00- 12:30 Daniel Mange (EPFL, Lausanne)
Von Neumann Revisited: a Turing Machine with Self-Repair and Self-Reproduction Properties

12:30- 1:30 Lunch

Robotics and Autonomous Systems

1:30- 1:50 Lynne Parker (ORNL, Oak Ridge)
From Social Animals to Teams of Cooperating Robots

1:50-2:10 Takashi Gomi and Ann Griffiths (Applied AI Systems, Toronto)
Bio-Robots for Real World Applications

2:10-2:30 Bengt Carlsson (Karlskrona U. Sweden)
The War of Attrition Strategy in Multi-Agent Systems

2:30-2:50 Claudio Cesar de Sa (IMA, Brasil)
Architecture for a Mobile Agent

2:50-3:10 A.N. Stafylopatis (NTU, Athens)
Autonomous Vehicle Navigation Using Evolutionary Reinforcement Learning

3:10-3:30 Jun Tani (Sony, Tokyo)
Cognition from Dynamical Systems Perspective: Robot Navigation Learning

3:30-4:00 Discussion and Coffee

Models and Measurements

4:00-4:20 Erol Gelenbe (Duke Univ.)
Genetic Algorithms which Learn

4:20-4:40 Petr Lansky (CTS, Prague University), Jean-Pierre Rospars (INRA)
Stochastic Models of the Olfactory System

4:40-5:00 Vladimir Protopopescu (ORNL, Oak Ridge, Tenn.)
Learning Algorithms Based on Finite Samples

5:00-5:20 Ivan Havel (CTS, Prague University)
Interaction of Processes at Different Time Scales

5:20-5:40 Miguel Nicolelis (Duke University)
Dynamic and Distributed Processing of Sensory Information by Multi-layer Neuronal Networks in Mammals

Discussions and/or Walk Around Duke Forest

7:00 Workshop Dinner

Autonomous Vehicle Navigation Using Evolutionary Reinforcement Learning (Extended Abstract)

A. Stafylopatis and K. Blekas
National Technical University of Athens
Department of Electrical and Computer Engineering
157 73 Zographou, Athens, Greece

1 Introduction

Typically a trainable adaptive controller architecture facing supervised learning consists of a teacher, the trainable controller and the plant to be controlled. The teacher may be automated as a linear or nonlinear control law, or it may be a human expert. However, in some learning tasks arising in control neither a designed nor a human expert is available. In these situations it may be possible to improve plant performance over time by means of on-line methods performing *reinforcement learning* [6, 7], i.e. to adjust a control rule as a function of a measure that in some way evaluates the overall behavior of the plant. Based on the received scalar reinforcement values, the learning system has to discover the most appropriate actions corresponding to the state of the environment as it is perceived through a set of sensors or detectors. Reinforcement learning has been effectively applied to control tasks and, in particular, to the problem of autonomous vehicle navigation in unknown environments [3, 4, 5], assuming no prior knowledge about the task and the system to be controlled.

Our concern in this paper is the use of a reinforcement learning strategy to drive an autonomous vehicle through simulated paths made of straight segments as well as left and right turns. The task of providing the proper control commands, so that the vehicle stays on the road and avoids collision, is assigned to a *learning classifier system (LCS)* [1, 2, 8, 9]. An LCS is a learning system in which a set of condition-action rules called classifiers compete to control the system and gain credit based on reinforcement provided by the environment. Classifiers are generated and modified in an evolutionary process using a genetic algorithm (GA). Survival and evolution depend on the fitness (strength) of individuals, which in turn is based on the cumulative credit of classifiers.

Previous work of ours on the autonomous navigation problem has been based on the use of pure connectionist reinforcement learning schemes assuming no a priori knowledge, as well as on fuzzy-neural approaches. In the scheme proposed here, a classifier system is trained according to an immediate reinforcement algorithm, while the coverage of the rule-space is achieved by means of an appropriate genetic mechanism.

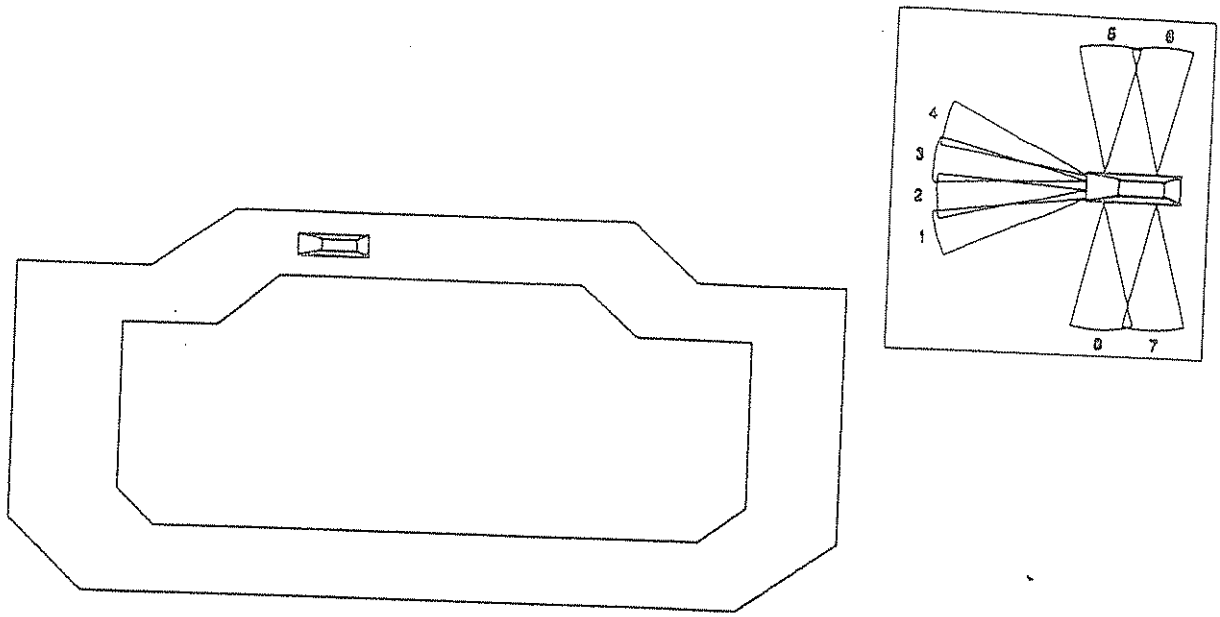

Figure 1: A typical ground and the vehicle

2 The Autonomous Vehicle Application

Collision-free autonomous navigation of a vehicle in uknown grounds constitutes an interesting control problem. The objective is to give the proper driving commands as a response to the current state of the vehicle, so that it moves in a course without collisions. This task represents a sensor-motor association problem in the control area. Conventional approaches to solve the problem use artificial intelligence methods (expert systems), which are time-consuming and require much knowledge of the environment in which the vehicle is moving. In our approach we assume that very little is known about the vehicle's environment.

The autonomous vehicle uses a number of sensors to perceive its environment. The number and the configuration of the sensors play a very important role in the control task. On the one hand they cannot be too many, and on the other hand their number must be sufficient to provide the necessary information.

Eight sensors seem to be adequate to describe the state of the vehicle at each time instant. Figure 1 illustrates a typical ground and shows the location of the sensors on the vehicle, where four of them are placed at the front and two at each side. Each sensor can detect the presence of an obstacle situated within a conic space in front of it and provides a measure of the distance of the obstacle. Although the sensors provide a real-valued distance measure, we have considered for encoding purposes a discrete representation of the distance, by partitioning the cone of each sensor into 3 logarithmically sized areas, numbered from 0 to 2. A smaller area number corresponds to obstacles at a closer distance. An additional distance value of 3 indicates that there is no obstacle in the range covered by a sensor. Thus, the distinction of 4 different distance values is possible (very close, close, far, very far), requiring 2 bits for each sensor value. Therefore, the environmental state of the vehicle is described by a 16-bit vector.

During navigation, the vehicle is able to perform one of five possible actions in response to the current state. These driving commands are:

1 : Ahead
2 : 30 degrees right
3 : 60 degrees right
4 : 30 degrees left
5 : 60 degrees left

The movement of the vehicle is governed by a set of rules that relate states to actions. Each rule i can be represented as a pair (c_i, a_i), where the condition part c_i and the action part a_i are encoded as described above.

After each movement, the environment provides an evaluation of the action performed at the given state of the system. The evaluation should be based on how probable it is for the vehicle to collide and go off the road. This probability can be estimated in terms of the positions of obstacles as they are detected by the sensors. In our system, the scalar reinforcement signal r, which is the outcome of the evaluation, is obtained as an average of the partial reinforcements r_k, $(k = 1, \ldots, 4)$, computed by the four front sensors. These partial reinforcements are computed according to the rule $r_k = \xi_k/27$ where the ξ_k's take values from the set $\{0, \ldots, 27\}$ and are obtained through a different discretization of the detected distances. The value $\xi_k = 0$ means that sensor k has detected an obstacle right in front of it, while larger values of ξ_k correspond to obstacles at a longer distance. The case of failure is treated in a special manner. If at least one of the sensors satisfies the condition $\xi_k = 0$, the global reinforcement assumes a zero value. It is obvious from the above description of the evaluation mechanism that the reinforcement signal takes values in the interval $[0, 1]$ and that large reinforcements correspond to actions leading to improved performance of the controlled system.

3 The Learning Classifier System

A schematic representation of the proposed learning classifier system is given in Figure 2, where the interaction with the environment is performed via sensors for state detection and actuators for motor actions. Each classifier (rule) i has a condition part c_i built upon the binary alphabet plus the "don't care" symbol #, and an action part a_i encoding one of the possible actions. There is a scalar strength S_i^t associated with each classifier i in the current set of working classifiers at time step t. The system starts with a random set of classifiers initialized to some default strength. Training of the classifier system consists of a sequence of cycles, where each cycle begins with the vehicle at the same initial position and ends with a failure (collision) signal.

At each step during operation, the condition of each classifier in the current set is compared with the state vector. If a classifier's condition matches the state vector in all non-# positions, the classifier is included in the current *match set* M. Next, an action is selected from among those recommended by the classifiers in M. The selection can be done based on the sum of strengths of members of M supporting the action a_i, denoted by S_{M_i}, either using a roulette wheel method (with probability proportional to S_{M_i} for each action a_i) or using a more flexible Boltzmann probability

$$\text{Prob}(a_i) = e^{S_{M_i}/T} / \sum_j e^{S_{M_j}/T}$$

following some "annealing" parameter T. Then, all members of M advocating the selected action become members of the current *action set* A. The selected action is performed by the actuators and a reinforcement signal r $(0 \leq r \leq 1)$ is received from the environment.

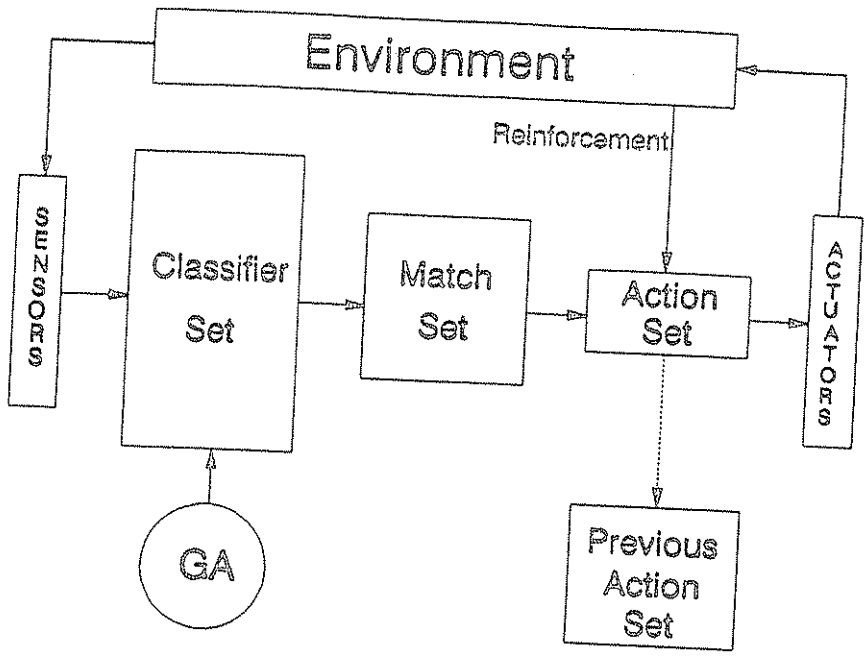

Figure 2: The classifier system architecture

The operation step is concluded by appropriately doing *credit assignment* to the active classifiers. The algorithm adopted in our approach borrows from a variant of the "Bucket Brigade" algorithm of Holland [2, 9] and from techniques used in immediate reinforcement learning schemes [3, 7]. Let S_A^t, S_A^{t+1} denote the total strength of the classifiers present in the action set at time steps t and $t+1$ respectively. Credit assignment is performed by means of the following update equation, where modification of the strength of a set implies apportionment of the increment to all its members:

$$S_A^t \longleftarrow (1-\beta)S_A^t + \eta(r-b)S_A^t + \gamma\beta S_A^{t+1}$$

According to the above algorithm a fixed fraction β $(0 < \beta < 1)$ of the current strength of the action set is removed and added to the strength of the action set of the previous time step discounted by a factor γ $(0 < \gamma < 1)$. On the other hand, the strength of the current set is modified by an amount depending on the received reinforcement r, specifically the offset of r with respect to some baseline value b, with a learning rate η. In this way, the reinforcement signal r is rewarding if it is greater than the baseline and is penalizing if it is less than the baseline. For the baseline either a fixed value (e.g. $b = 0.5$) can be used or a trace $\overline{r}$ of past values of the reinforcement signal, computed as an exponentially weighted average. In the latter case, the trace can be viewed as a prediction of the expected reinforcement value and can be computed for the classifier system as a whole or for each individual classifier separately. In the case of rewarding, the strengths of classifiers in the set difference $M - A$ are decreased by a fixed small fraction δ, to enhance strength differences while preventing unbounded growing of strength.

The *genetic algorithm component* of the classifier system is responsible for generating a number of new classifiers by evolving the existing ones through crossover and mutation, while keeping the total number of classifiers constant. The GA is invoked with a fixed period within training cycles, as well as at cycle ending instants (failure). We have considered both a "panmictic" and a "niching" mechanism to drive the population evolution. The effect of the GA component is supplemented by a *covering* operation, that is by the insertion of appropriately constructed new rules each time the match set is empty.

4 Experimental Results

The performance of the proposed system has been investigated through computer simulation experiments, each experiment consisting of a sequence of training cycles. Several training grounds of varying difficulty have been employed. Statistical results of the effectiveness of learning during each experiment were obtained by computing a moving average of the number of steps per cycle. The experimental investigation of the system's behaviour allowed us to evaluate the effect of the various options considered during design. The performance of vehicle movement under the proposed evolutionary approach has been compared with that of other approaches (such as neural and fuzzy-neural schemes based on reinforcement learning) that had been applied to the same benchmark application in previous work of ours. The results have shown that the evolutionary reinforcement learning mechanism offers a particularly promising alternative in adaptive movement control.

References

[1] R. Abu Zitar and M.H. Hassoun, "Neurocontrollers Trained with Rules Extracted by a Genetic Assisted Reinforcement Learning System", IEEE Trans. on Neural Networks, Vol. 6, No. 4, pp. 859–879, July 1995.

[2] D. E. Goldberg, "Genetic Algorithms in Search, Optimization & Machine Learning", Reading, MA: Addison-Wesley, 1989.

[3] D. Kontoravdis and A. Stafylopatis, "Reinforcement Learning Techniques for Autonomous Vehicle Control", Neural Network World, Vol. 3-4, pp. 329–346, 1992.

[4] D. Kontoravdis, A. Likas, K. Blekas and A. Stafylopatis, "A Fuzzy Neural Network Approach to Autonomous Vehicle Navigation", in Proc. Europ. Robotics and Intelligent Systems Conf., pp. 243–252, Malaga, Spain, 1994.

[5] J. del R. Millan and C. Torras, "A Reinforcement Connectionist Approach to Robot Path Finding in Non-Maze-Like Environments", Machine Learning, Vol. 8, pp. 363–395, 1992.

[6] R.S. Sutton, A.G. Barto and R.J. Williams, "Reinforcement Learning in Direct Adaptive Optimal Control", IEEE Control Systems Mag., Vol. 12, No. 2, pp. 19–22, 1992.

[7] R.J. Williams, "Simple Statistical Gradient-Following Algorithms for Connectionist Reinforcement Learning", Machine Learning, Vol. 8, pp. 229–256, 1992.

[8] S.W. Wilson, "Classifier Systems and the Animat Problem", Machine Learning, Vol. 2, pp. 199-228, 1987.

[9] S.W. Wilson, "ZCS: A Zeroth Level Classifier System", Evolutionary Computation, Vol. 2, No. 1, pp. 1–18, 1994.