Adaptive Behavior of Fuzzy System Optimized by Genetic Algorithm

Sung-Bae Cho and Seung-Ik Lee
Dept. of Computer Science, Yonsei University
134 Shinchon-dong, Sudaemoon-ku, Seoul 120-749, Korea

Abstract— **The problem of automatically adapting the behavior of a mobile robot in a changing environment is recognized as a very difficult task. Towards a promising approach to this problem, we have developed a genetic fuzzy controller for a mobile robot, and showed the potential by applying to a simulated robot called Khepera. The robot gets input from eight infrared sensors and operates two motors according to the fuzzy inference based on the sensory input. This paper attempts to analyze the adaptive behaviors of the controller by using automata, which indicates the emergence of several strategies to make the robot to navigate the complex space without bumping against walls and obstacles.**

I. Introduction

It is difficult to program an autonomous robot so that it reliably acts in a dynamic environment. This is due to such problems as missing necessary information at design stage, the unpredictability of the environment dynamics, and the inherent noise of the sensors and actuators [7]. Clearly, an autonomous robot that can acquire knowledge by interaction with the environment and subsequently adapt and change its behavior in the run time could greatly simplify the work of its designer. As a promising approach to this learning autonomous robot, the behavior-based robotics has recently appeared [2], [6].

One of the key points of this approach is not to give the robot information about the environment but to let the robot find the knowledge by itself. With this approach, a number of researchers have successfully employed an evolutionary procedure to develop the control system of simulated robots [1], [3], [9], [13]. The rich variety of structures that have appeared during evolution and the large number of evolved behaviors have empirically demonstrated the power and generality of the evolutionary algorithms. However, this approach suffers from the difficulty of analyzing the control system evolved, which prohibits the designer from fully exploiting domain knowledge to design the control system by an evolutionary approach.

To work out this problem, we proposed a fuzzy system for a behavior-based robot, and presented an evolutionary approach to determine the parameters in the fuzzy controller [4]. In this paper, we attempt to analyze the genetic fuzzy controller developed to control the simulated robot called Khepera. We also show that the adaptive behaviors result from the interaction of several primitive low-level strategies acquired through the evolutionary process.

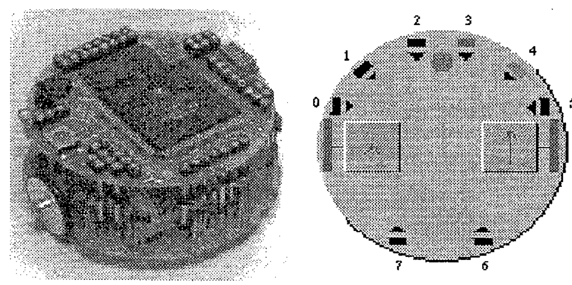


Fig. 1. Khepera robot and the simulated representation.

II. Autonomous Robot

For the simulation, we have used the Khepera robot that is circular, compact and robust as shown in Fig. 1. This is a miniature robot that has diameter of $55mm$, height of $30mm$, and weight of $70g$. The robot is supported by two wheels and two small Teflon balls placed under its platform. The wheels are controlled by two DC motors with an incremental encoder (12 pulses per mm of robot advancement) and can rotate in both directions. The geometrical shape and the motor layout of Khepera make the robot to navigate in sophisticated environment even when its control system is immature.

It is provided with eight infrared proximity sensors placed around its body which are based on emission and reception of infrared light. Each receptor can measure both the ambient infrared light and the reflected infrared light emitted by the robot itself. Several new single sensors and complete modules, such as a stereo-vision module and a gripper module, can be easily added, due to the hardware and software modularity of the system.

Dedicated to Khepera, the simulated mobile robot [12] includes eight infrared sensors allowing it to detect by reflection (small rectangles) the proximity of objects in front of it, behind it, and to the right and left sides

0-7803-4869-9/98 $10.00©1998 IEEE

of it. Each sensor returns a value ranging between 0 and 1023 represented in gradual color levels. 0 means that no object is perceived whereas 1023 means that an object is very close to the sensor (almost touching the sensor). Intermediate values may give an approximate idea of the distance between the sensor and the object. Each motor can take a speed value ranging between -10 and $+10$. The size of arrows on the motors in Fig. 1 indicates the amount of speed.

III. Genetic Fuzzy System

In order to operate the robot introduced at the previous section, we have developed a fuzzy controller in which genetic algorithm determines the internal parameters. A fuzzy inference system provides a computing framework based on the concepts of fuzzy sets, fuzzy if-then rules, and fuzzy reasoning. The basic structure consists of a fuzzy rulebase, a reasoning mechanism, and a defuzzification. A fuzzy rulebase is a set of fuzzy rules that are expressed as follows:

(Rule 1) If (x_1 is A_1^1) and ... and (x_n is A_n^1), then y is B^1
(Rule 2) If (x_1 is A_1^2) and ... and (x_n is A_n^2), then y is B^2
...
(Rule m) If (x_1 is A_1^m) and ... and (x_n is A_n^m), then y is B^m

where x_j ($1 \leq j \leq n$) are input variables, y is output variable, and A_j^i and B^i ($1 \leq i \leq m$) are fuzzy sets which are characterized by the membership functions. In our simulation, the numbers of input and output variables are eight and two, respectively.

In order to facilitate the controller of Khepera robot, we use the following four fuzzy sets for the input and output parameters:

Input : 8 values from infrared sensors ($0 \sim 1023$)
Fuzzy set : I = {VF, F, C, VC}
VF (Very Far)
F (Far)
C (Close)
VC (Very Close)

Output : 2 values from motors ($-10 \sim +10$)
Fuzzy set : O = {BH, B, F, FH}
BH (Backward High)
B (Backward)
F (Forward)
FH (Forward High)

Triangular shapes specify the membership function. A parameter value divides the range ($0 \sim 1023$ for input and $-10 \sim +10$ for output) by 10 equidistance segments. Fig. 2 shows the membership functions used

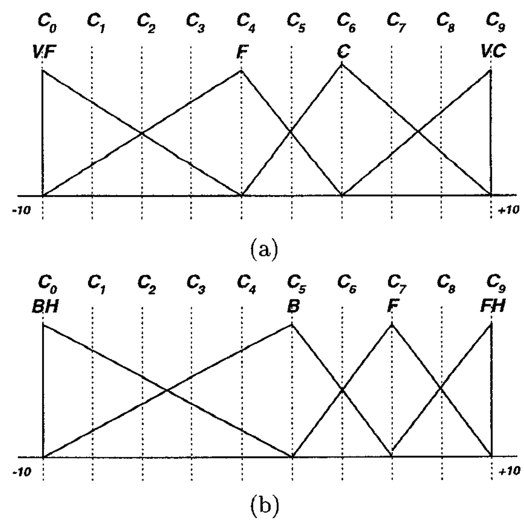


Fig. 2. Membership functions for (a) input; (b) output.

for input and output values, respectively.

For fuzzy inference is used correlation minimum method, which truncates the consequent fuzzy region at the truth of the premise [10]. The inference value, μ_i, of the ith rule is defined as follows.

$$\mu_i = \min(I_{i0}(x_0), I_{i1}(x_1), \cdots, I_{ij}(x_j)),$$
$$j = \text{no. of input variables} \qquad (1)$$

Finally, centroid defuzzification method is adopted to yield the expected value, y_l^*, of the solution fuzzy region, as follows.

$$y_l^* = \frac{\sum_{i=0}^{m} \mu_i \overline{y_i}}{\sum_{i=0}^{m} \mu_i} \qquad (2)$$

In order to robustly determine the shape and number of membership functions in fuzzy rules, genetic algorithm has been utilized. This approach reduces the burden of human operators to decide the structure of fuzzy rules. Genetic algorithm (GA) is considered as an effective method for optimization [8], and several hybrid methods with fuzzy logic have been recently proposed [5], [11]. Fig. 3 shows the overall diagram of the proposed system. The system parameters in the fuzzy system are represented as a gene, and the performance with the Khepera simulator decides whether it can produce offsprings with the genetic operators. In the figure, four genes of number 1, 2, 4 and 6 are selected as candidates for the next generation, and the crossover is applied to them.

To get a success in the application of genetic algorithm, it is quite important to devise a gene coding

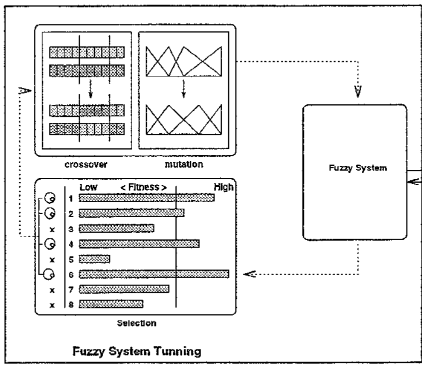


Fig. 3. Schematic diagram of the genetic fuzzy system.

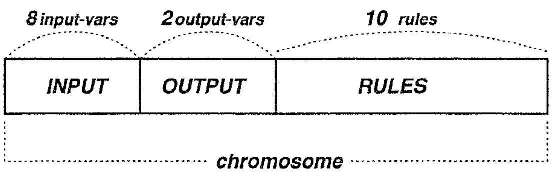


Fig. 4. Gene code for encoding the fuzzy system.

scheme appropriate to the problem. For our problem, we should incorporate the input and output membership functions and the rules as a gene code as shown in Fig. 4 which encodes the eight input parameters, two output parameters and maximum 10 rules. For details on the gene encoding scheme, refer to the recent publication made by the authors [4]. In this paper, we have used the same encoding method, and tried to analyze the adaptive behaviors of the robot evolved.

Another important issue in the application of genetic algorithm is to determine a proper fitness measure for the problem. In this paper we make the fitness function to decrease as the robot bumps against the walls, and to increase as it moves farther from the start point. In addition, a couple of factors are included to induce the compact fuzzy system by preferring to the smaller number of rules and membership functions. The fitness function is as follows.

$$\begin{aligned} \text{fitness} &= \alpha \times \text{no. of collisions} \\ &+ \beta \times \text{distance moved} \\ &+ \gamma \times \text{no. of rules} \\ &+ \delta \times \text{no. of membership functions} \\ &+ \epsilon \times \text{no. of check points reached,} \end{aligned} \tag{3}$$

where $\alpha = -3$, $\beta = 1$, $\gamma = -100$, $\delta = -10$, and $\epsilon = 500$.

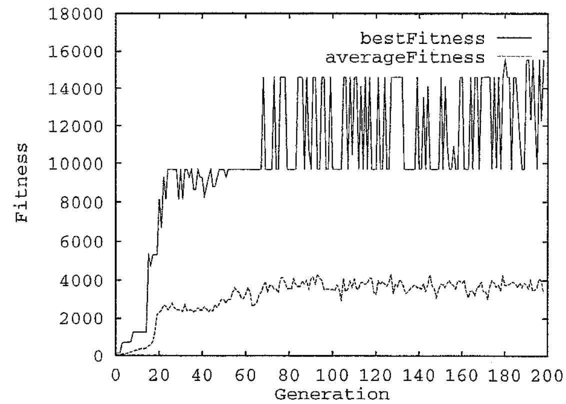


Fig. 5. Fitness change.

The coefficients might be determined by another optimization technique, but in this paper we have just selected them by trial-and-error. The fitness would increase as the robot goes farther from the start point while passing by more check points. The fitness would decrease as the robot collides with the walls or the numbers of rules and membership functions get larger. In order to expedite the evolution, we put several check points along with the pathways which will be removed later.

IV. Simulation

The Khepera simulator was written in C++ [12], and the simulation was conducted in SUN Sparc 10 workstation. We initialized 200 chromosomes at random, each of which was developed to a fuzzy controller for the robot. Each robot operates within 5000 sensor sampling time, and produces the performance value according to the fitness function.

Fig. 5 shows the best and average fitness changes in the course of simulation. As the figure depicts, the performance increases gradually as the generation goes, and a robot navigated successfully at the given environment has been obtained at less than 100 generations. It can be seen that the fitness is radically increased at the beginning stage, but there is nearly no change after 90 generations. Around the $67th$ generation the best individuals already perform a near optimal behavior. They navigate so smoothly that they do not bump into walls and corners, and maintain a straight trajectory when possible.

Fig. 6 shows the trajectories that the robot has made during the simulation. These results are highly reliable and have been replicated in many runs of the experiment. In the beginning of the evolution the individuals evolved a frontal direction of motion, corresponding to

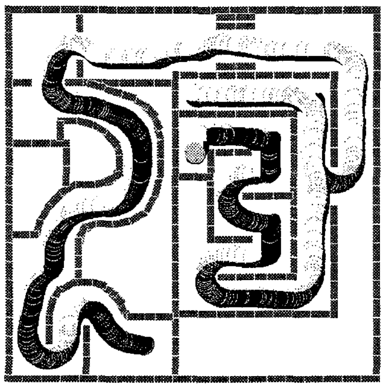


Fig. 6. Trajectories of the robot.

the side where more sensors are available. Those individuals that moved in the other direction got stuck in a corner without being able to detect it, and soon disappeared from the population. The controller for this robot consists of only seven effective rules, which are generated through the evolutionary process as follows.

(Rule 1) If $(x_2 = C)$ and $(x_5 = VF)$ and $(x_7 = VC)$
then $(y_0 = BH)$ and $(y_1 = B)$
(Rule 2) If $(x_4 = VF)$
then $(y_0 = FH)$ and $(y_1 = F)$
(Rule 3) If $(x_1 = VC)$ and $(x_2 = F)$ and
$(x_4 = C)$ and $(x_7 = VC)$
then $(y_0 = BH)$ and $(y_1 = B)$
(Rule 4) If $(x_2 = F)$ and $(x_3 = F)$ and $(x_6 = VC)$
then $(y_0 = F)$ and $(y_1 = FH)$
(Rule 5) If $(x_4 = VC)$
then $(y_0 = BH)$ and $(y_1 = F)$
(Rule 6) If $(x_2 = VF)$ and $(x_4 = F)$ and $(x_6 = VC)$
then $(y_0 = F)$ and $(y_1 = FH)$
(Rule 7) If $(x_0 = VF)$ and $(x_4 = F)$ and $(x_5 = C)$
then $(y_0 = BH)$ and $(y_1 = F)$

Even though we did not give any hints to the system, several effective rules to control the mobile robot appropriately at a number of different cases have emerged through the evolution. The overall behavioral model can be depicted as Fig. 7. The rule 2 triggers the state of "Obstacle Avoidance," the rules 2 and 7 cooperatively induces the state of "Wall Following," and the rule 5 activates the state of "Impact Avoidance." This result dictates that the evolutionary approach is quite useful to design a flexible and efficient fuzzy systems to control mobile robot.

For instance, Fig. 8 shows the snapshots of the robot that escapes from the closed corridor. When the robot arrives at the closed corridor the internal state of the

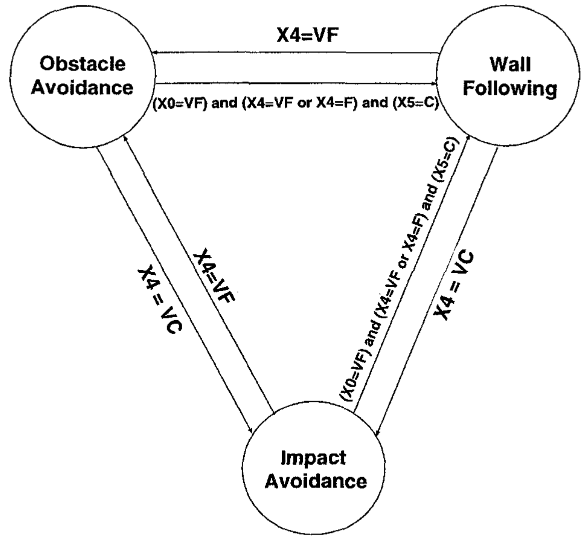


Fig. 7. Behavior model for the robot evolved.

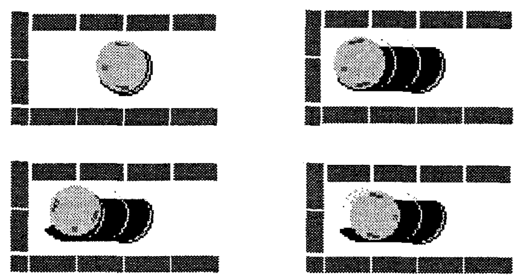
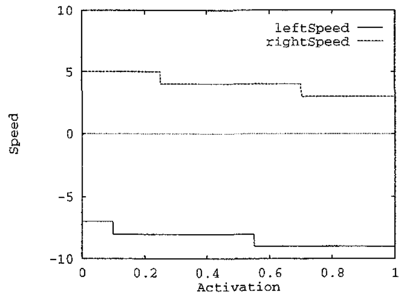


Fig. 8. Snapshots of the robot escaping from the closed corridor.

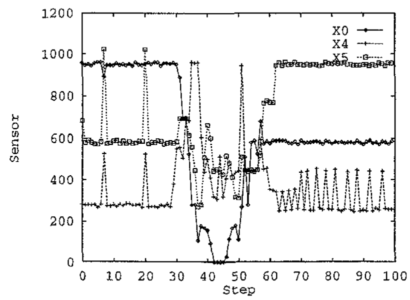
robot changes to "Impact Avoidance" which is governed by rule 5, while the usual "Wall Following" state is activated by rules 2 and 7. Fig. 9(a) depicts the speed of the two motors with respect to the activation levels of rule 5. As can be seen from this figure, the robot turns left as soon as the rule 5 is activated. Fig. 9(b), (c) and (d) show the changes of the sensor values, the activation levels of the rules, and the speed of left and right motors, respectively.

V. Concluding Remarks

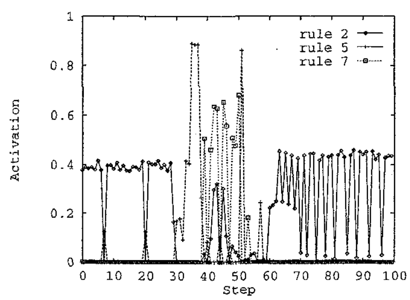
In this paper, we have presented a fuzzy system to control a mobile robot, and utilized genetic algorithm to optimize the internal parameters in the system. A successful controller generated consists of only seven effective rules, which shows the evolution finds out the optimal set of rules to control the robot. An analysis of the simulation results dictates that the evolutionary



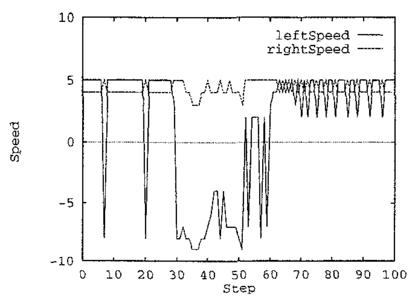
(a) speed of two motors with respect to the activation of rule 5



(b) changes of the sensor values



(c) activation levels of the rules



(d) speed of left and right motors

Fig. 9. Internal states of the robot in the closed corridor.

approach is quite useful to design a flexible and efficient fuzzy systems to control mobile robot.

Acknowledgements

This work was supported in part by a grant from the Korea Research Foundation and the Ministry of Science and Technology in Korea.

References

[1] R.D. Beer and J.C. Gallagher, "Evolving dynamical neural networks for adaptive behavior," *Adapt. Beh.*, vol. 1, pp. 91–122, 1992.

[2] R.A. Brooks, "A robust layered control system for a mobile robot," *IEEE Trans. Robotics and Automation*, vol. 2, no. 1, pp. 14–23, 1986.

[3] D. Cliff, I. Harvey and P. Husbands, "Explorations in evolutionary robotics," *Adapt. Beh.*, vol. 2, pp. 73–110, 1993.

[4] S.-B. Cho and S.-I. Lee, "Hybrid evolutionary learning of fuzzy logic and genetic algorithm," *Lecture Notes in Artificial Intelligence*, Springer Verlag, vol. 1285, pp. 206–215, 1997.

[5] O. Cord, F. Herrera and M. Lozano, "A classified review on the combination fuzzy logic–genetic algorithms bibliography," *Technical Report DECSAI-95129*, Dept. of Computer Science and A.I., University of Granada, 1996.

[6] M. Dorigo and U. Schnepf, "Genetic-based machine learning and behavior based robotics: A new synthesis," *IEEE Trans. Syst. Man, Cybern.*, vol. 23, pp. 141–154, 1993.

[7] M. Dorigo, "Introduction to the special issue on learning autonomous robots," *IEEE Trans. Syst. Man, Cybern.*, vol. 26, pp. 361–364, 1996.

[8] D.E. Goldberg, *Genetic Algorithms in Search Optimization & Machine Learning*, Addison-Wesley, 1989.

[9] J. Kodjabachian and J.A. Meyer, "Evolution and development of control architectures in animats," *Robotics and Autonomous Systems*, vol. 16, pp. 161–182, 1995.

[10] B. Kosko, *Neural Networks and Fuzzy Systems*, Prentice-Hall, 1992.

[11] D. Leitch and P. Probert, "Genetic algorithms for the development of fuzzy controllers for mobile robots," *Lecture Notes in Artificial Intelligence 1011*, pp. 148–172, Springer, 1995.

[12] O. Michel, *Khepera Simulator Version 1.0 User Manual*, 1995.

[13] D. Parisi, F. Cecconi and S. Nolfi, "Econets: Neural networks that learn in an environment," *Network*, vol. 1, pp. 149–168, 1990.