1999 IEEE International Fuzzy Systems Conference Proceedings
August 22-25, 1999, Seoul, Korea

Rule-based Integration of Multiple Neural Networks Evolved based on Cellular Automata

Geum-Beom Song and Sung-Bae Cho
Department of Computer Science, Yonsei University
134 Shinchon-dong, Sudaemoon-ku, Seoul 120-749, Korea
Tel: +82-2-361-2720 Fax: +82-2-365-2579
(E-mail: goldtiger@candy.yonsei.ac.kr, sbcho@csai.yonsei.ac.kr)

Abstract

There has been extensive research of developing the controller for a mobile robot. Especially, several researchers have constructed the mobile robot controller that can avoid obstacles, evade predators, or catch moving prey by evolutionary algorithms such as genetic algorithm and genetic programming. In this line of research, we also presented a method of applying CAM-Brain, evolved neural networks based on cellular automata (CA), to control a mobile robot. However, this approach has a limitation to make the robot to perform appropriate behavior in complex environments. In this paper, we have attempted to solve this problem by combining several modules evolved to do a simple behavior by rule-based approach. Experimental results show that this approach has possibility to develop a sophisticated neural controller for complex environments.

1.Introduction

There are many studies of constructing mobile robot controller with different approaches such as evolving neural network by genetic algorithm [1], using genetic programming [2], combining fuzzy controller with genetic algorithm [3] and programming [4]. In previous work [5], we introduced CAM-Brain, evolved neural networks based on cellular automata [5, 6], and applied it to controlling a mobile robot.

However, the controller composed of one module has a difficulty to make the robot to perform complex behavior. To overcome this shortcoming, some researchers combine several modules evolved or programmed to a simple behavior such as "go straight", "avoid obstacles", "seek object", and so on. They expect the controller combined with several modules can do complex behaviors [4, 7, 8].

In this paper, we also attempt to combine several neural networks for solving this problem. Each neural network can be evolved or programmed. Evolved neural network is based on CAM-Brain model, and a programmed module controls the robot directly. We apply rule-based method to combine modules and control a mobile robot in several environments.

The rest of this paper briefly introduces CAM-Brain model and basic behaviors, and presents the combining method in detail. The detailed description of simulation follows, and the results of simulation is given.

2.Neural Networks Evolved on CA

CAM-Brain is a model based on CA which can show complicated behavior by combining simple rules, and developed by its own chromosome that has information about CA-cell structure. One chromosome is mapped to exactly one neural network module. Therefore, with genetic algorithm working on this chromosome, it is possible to evolve and adapt the structure of the neural network for a specific task. It is the basic idea of CAM-Brain that brain-like system can be made by combining many neural network modules of various functions [5, 6]. This section illustrates a design of CA-space for developing a neural network module.

2.1.CoDi Model

CAM-Brain's neural network structure composed of blank, neuron, axon and dendrite is grown inside 2-D or 3-D CA-space by state, neighborhoods and rules encoded by chromosome. If cell state is blank, it represents empty space and cannot transmit any signals. Neuron cell collects signals from surrounding dendrite cells which are accumulated. If the sum of collected signals is greater than threshold, neuron cells send them to surrounding axon cells. Axon cell sends signals received from neurons to the neighborhood cells. Dendrite cell collects signals from neighborhood cells and passes them to the connected neuron in the end [5, 6].

2.2.Growth Phase

The growth phase organizes neural structure and makes the signal trails among neurons. First, a chromosome is randomly made and the states of all cells are initialized as blank. At this point some of the cells are specified as neuron with some probability. Then, a neuron cell sends axon and dendrite growth signals to the direction decided by chromosome. Axon growth signal is sent to two directions and dendrite growth signal is sent to the other remaining directions. Next the blank cell received growth signal changes to axon or dendrite cell according to the type of growth signal. It sends the signals received from other cells to the direction determined by chromosome. Finally, repeating this process, the neural network is obtained when the state of every cell changes no longer.

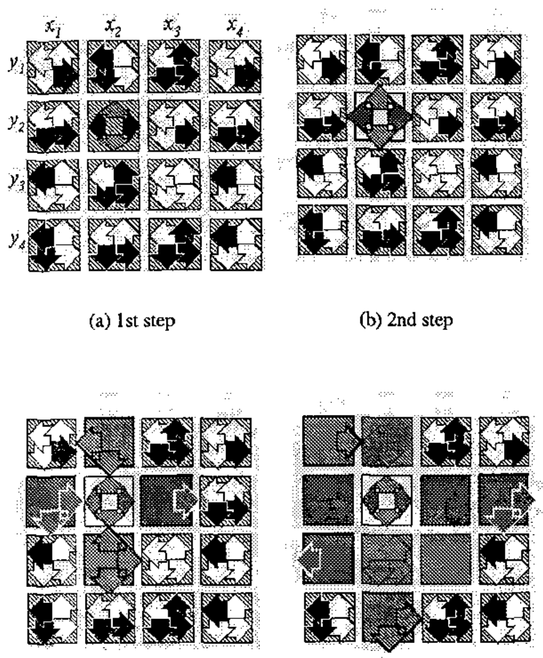


Figure 1: Growth phase. (a) Black arrow represents signaling direction determined by chromosome, and a neuron is located in (x_2, y_2). (b) The neuron sends growth signals. (c) The cell state is decided according to the type of growth signals. (d) Propagating growth signals, blank cells become axon or dendrite.

Figure 1 shows the growing process in 4 × 4 2-D CA-space. In this figure, the cell which has oblique lines is blank cell, and the black arrows show the direction of signaling decided by chromosome. Figure 1(a) shows the process of seeding a neuron in blank cells, where a neuron is located in (x_2, y_2). Figure 1(b) shows that the neuron cell sends growth signal to surrounding cells. Figure 1(c) shows the cell state is changed by growth signal. Figure 1(d) shows that blank cells grow into axon or dendrite. In a neuron, the dendrite collects signals and sends to the neuron, and the axon distributes signals originated from the neuron.

2.3.Signaling Phase

Signaling phase transmits the signal from input to output cells continuously. The trails of signaling are transmitted with evolved structure at the growth phase. Each cell plays a different role according to the type of cells. If the cell type is neuron, it gets the signal from connected dendrite cells and gives the signal to neighborhood axon cells when the sum of signals is greater than threshold. If the cell type is dendrite, it collects data from the faced cells and eventually passes them to the neuron body. If the cell type is axon, it distributes data originating from the neuron body.

The position of input and output cells in CA-space is decided in advance. At first, if input cells produce the signal, it is sent to the faced axon cells, which distribute that signal. Then, neighborhood dendrite cells belonged to other neurons collect and send this signal to the connected neurons. The neurons that have received the signal from dendrite cells send it to axon cells. Finally, dendrite cells of output neuron receive and send this signal to the output neurons. Output value can be obtained from output neurons. During signaling phase, the fitness is evaluated by the output in this process. Figure 2 shows the process of signaling after neuron, axon and dendrite have been made.

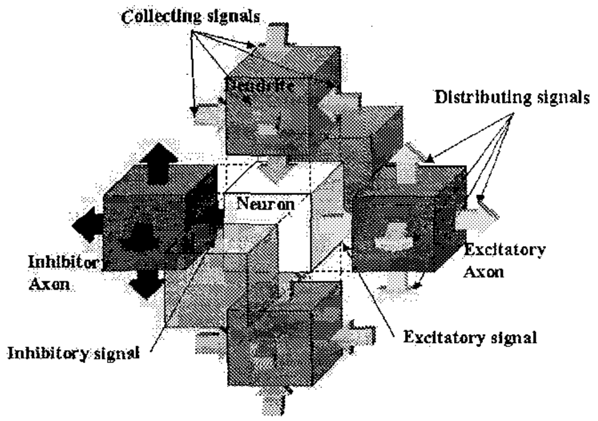


Figure 2: Signaling phase.

2.4.Evolution of CAM-Brain

In general, simple genetic algorithm generates the population of individuals and evolves them with genetic operators such as selection, mutation, and crossover [9]. We have used the genetic algorithm to search the optimal neural network. At first, a half of the population that has better fitness value is selected to produce new population. Two individuals in the new population are randomly selected and parts of them are exchanged by one-point crossover. The crossover is occurred at the same position in the chromosomes to maintain the same length in chromosomes. Mutation is operated in the segment of chromosome. The genetic algorithm generates a new population from the fittest individuals on the given problem. Figure 3 shows the evolution process of CAM-Brain.

3.Integration of Neural Networks

High and complex behaviors are made by combining low and simple behaviors. In this section, basis behaviors and the IF-THEN rules of combining them are described.

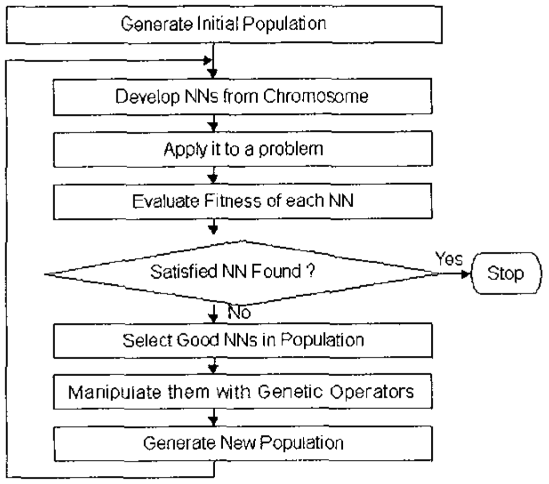

Generate Initial Population
Develop NNs from Chromosome
Apply it to a problem
Evaluate Fitness of each NN
Satisfied NN Found ?
Yes
Stop
No
Select Good NNs in Population
Manipulate them with Genetic Operators
Generate New Population

Figure 3: The evolution process of CAM-Brain.

3.1.Basis Behaviors

In this paper, four basis behaviors are defined as follows.

- **Battery Recharge** : If a robot arrives at battery recharge area, battery is recharged. This module enables the robot to operate as long as possible.
- **Follow Light** : The robot goes to stronger light. It must operate this module to go to the "Battery Recharge" area because the light source exists in that area.
- **Avoid Obstacles** : If the obstacles exist around the robot, it avoids obstacles without bumping against them. This module enables it to go to "Battery Recharge" area safely by avoiding obstacles.
- **Go Ahead** : If there is nothing around the robot, it goes ahead. This module makes it to move continuously without stopping.

These basis behaviors are needed to adapt to given environments shown in Figure 4. In order to live for a long time in a given environment the robot avoids bumping against obstacles and goes to the "Battery Recharge" area for recharging battery occasionally.

3.2.Combining with IF-THEN Rules

Biology gives us a reason of combining the modules. Hence, we use IF-THEN rules for combining the four basis behaviors properly. Using IF-THEN rules, we can decide an operating module according to situation of the robot which is judged by sensor values of it. We expect that this approach adapts to a given environment if the rules are defined well.

The rules used in this paper are as follows.

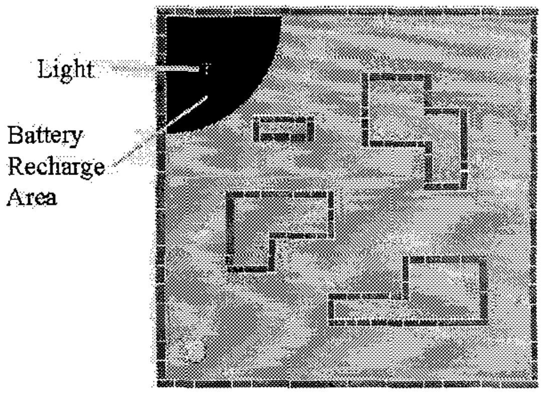


Figure 4: Simulation environments.

```
IF ("Battery Recharge" area)
    Execute Battery Recharge module
ELSE
    IF (Battery sensor $< \alpha$) AND
        (Minimum value of light sensors $<= \gamma$)
        IF (Maximum value of distance sensors $<= \beta_1$)
            Execute Follow Light module
        ELSE
            Execute Avoid Obstacles module
    ELSE
        IF (Maximum value of distance sensors $<= \beta_2$)
            Execute Go Ahead module
        ELSE
            Execute Avoid Obstacles module
```

The process of operating modules works well because these rules are very simple and clear. In these rules, constants are defined as follows.

- α : If battery sensor value is less than α, battery is needed to recharging.
- β_1 and β_2 : If the maximum values of distance sensors are larger than β_1 and β_2, the robot perceives obstacles.
- γ : If the minimum values of light sensor is less than γ, the robot perceives light.

The robot moves differently according to these constant values. For example, the robot moves around "Battery Recharge" area if α is large, while the robot consumes the battery before reaching the "Battery Recharge" area if it is small. Also, if β is large, the robot bumps against the obstacles frequently and if γ is very small, the robot far from light source cannot perceive "Battery Recharge" area.

4.Simulation Results

4.1.Behavior-based Robot: Khepera

Khepera robot contains 8 infrared sensors to detect by reflection the proximity of objects in front of it, behind it, and to the right and the left sides of it, and to measure the level of ambient light all around the robot. Also, the robot has two motors to control the left and right wheels. Khepera simulator also features the ability to drive a real Khepera robot, so that we can very easily transfer the simulation results to the real robot [2, 3, 4]. We have modified the Khepera simulator for the purpose of our experiments. Figure 5 shows the khepera robot.

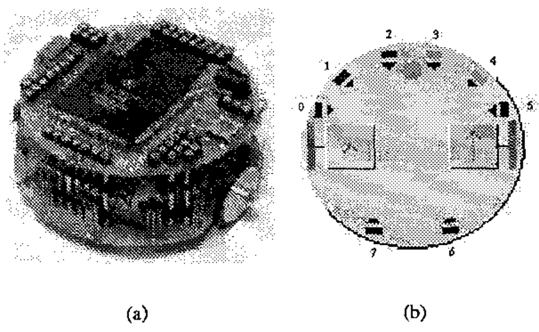


Figure 5: Khepera robot. (a) robot. (b) simulator.

4.2.Basis Modules

The modules are two kinds. One is programmed modules such as "Battery Recharge" and "Go Ahead", and the other is evolved CAM-Brain modules such as "Follow Light" and "Avoid Obstacles".

4.2.1."Battery Recharge" Module

"Battery Recharge" module is programmed because this behavior is very simple. Algorithm of this module is that the agents recharge battery if it is in the "Battery Recharge" area shown in Figure 4. We can decide whether it is in "Battery Recharge" area or not by measuring the distance from the light source to the location of robot. This method has a problem in real world because x-y coordinate values are not available. In this case, we can use another sensor which can recognize floor painted black [1].

4.2.2."Go Ahead" Module

This module is also programmed. If there is nothing around the robot, it goes ahead. This module is made by controlling the velocity of the left and right wheels. The velocity of left and right wheels is fixed as 5. Figure 6 shows the trajectory of the robot when this module operates.

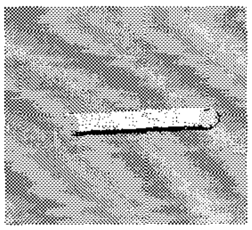


Figure 6: The trajectory of the robot when "Go Ahead" module operates.

4.2.3."Follow Light" Module

"Follow Light" module is evolved to go to the light source. The light source tells the robot about "Battery Recharge" area. Input values of CAM-Brain module are light sensor values of the robot. The fitness value is defined as follows.

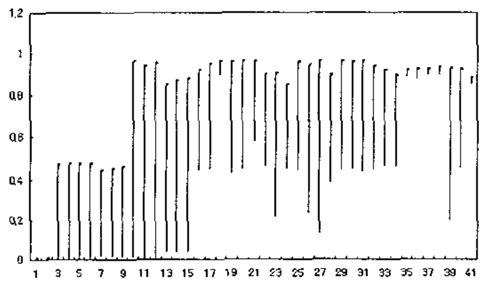
$$Fitness = S * (1 - \sqrt{V}) * (c * D) \qquad (1)$$

S : Average speed of the two wheels
V : Difference between the velocity of two wheels
c : 1, if the robot does not bump against obstacles
1/2, otherwise
D : Distance from the robot to the goal position

The fitness evaluation leads the robot to go straight quickly and reach nearby the light source. Population size is 50 and the fitness of individual is computed by the average of four experiments. Of course, the direction of the robot is different in each time. Figure 7(a) and (b) show the change of the best and average fitnesses respectively. Figure 8 shows the trajectory of the evolved robot. Figure 8(a), (b), (c) and (d) show the trajectory of the robot when the angle is 0, 90, 180 and 270 degrees, respectively.

4.2.4."Avoid Obstacles" Module

We evolve this module incrementally, because the behavior, avoid bumping against obstacles, is very complex and difficult to evolve. Incremental evolution approach is expected to evolve controller to do complex behavior efficiently. We use this module developed in our previous work [5]. Figure 9 shows the trajectory of this module when the robot avoids bumping against various obstacles.


(a)

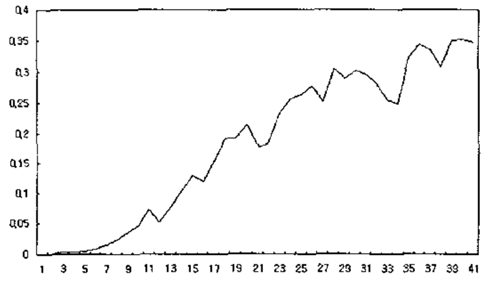
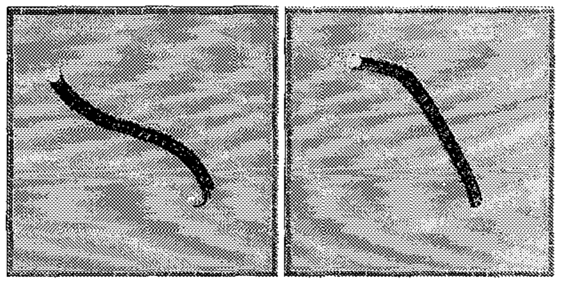

(b)

Figure 7: The change of the fitness. (a) Best fitness. (b) Average Fitness.


(a) (b)

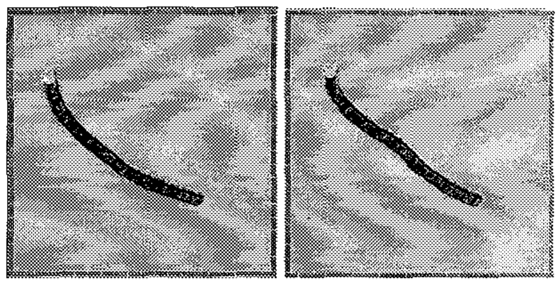

(c) (d)

Figure 8: The trajectory of the successful robot. (a) 0 degree. (b) 90 degrees. (c) 180 degrees. (d) 270 degrees.

4.3.Combined Modules

Combining four basis behaviors using IF-THEN rules explained at previous section, the controller shows more complex behaviors. We simulate this approach in modified Khepera simulator. We select the operating module by rules and sensor values of the robot. After selected basis behavior is operated, another module is selected by them. Repeating this process the robot moves. Figure 10 shows the trajectory of the robot by combined controller in a given environment. According to starting position of the robot, the trajectory and moving behavior are different. Battery decreases whenever the robot moves and battery becomes 2500 when it is recharged : the robot can move without recharging for 2500 times. α is 2/3 of the maximum battery, β_1 is 200, and β_2 is 250. Figure 10(a), (b), and (c) move for 5000, 14000, and 10000 time steps, respectively.

5.Concluding Remarks

This paper has attempted to combine several modules evolved or programmed to simple basis behavior and apply it to control a mobile robot in a complex environment. It is difficult to evolve controller to adapt to a complex environment. Therefore, this paper has combined multi-modules evolved using IF-THEN rules and showed the result of simulation. However, this approach is very simple and insufficient. Moreover, it has difficulty to determine the optimal rules varying the constant values. In summary, this paper shows the feasibility of combining multiple CAM-Brain modules evolved and programmed.

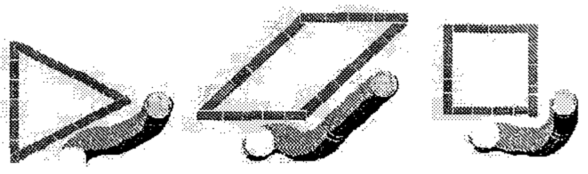
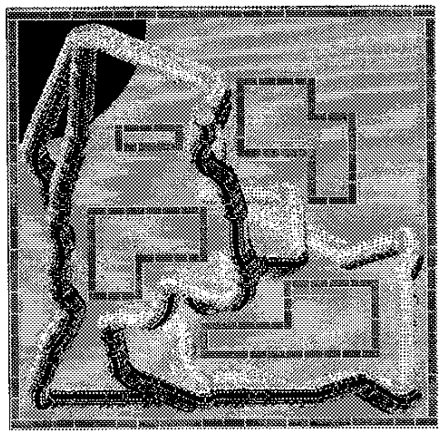
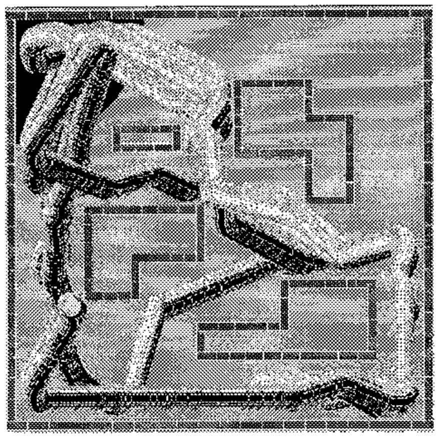

(a) (b) (c)

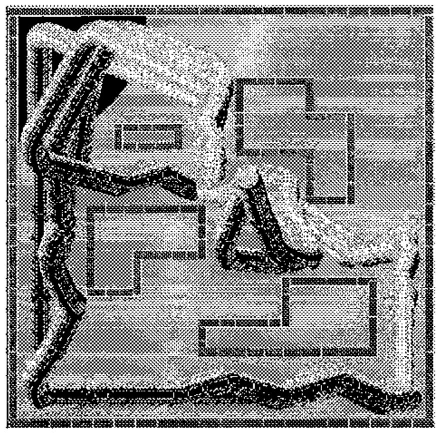
Figure 9: The trajectory of the robot when avoiding various obstacles.



(a)



(b)



(c)

Figure 10: The trajectory of the robot according to starting position of the robot.

References

[1] D. Floreano and F. Mondada, "Evolution of homing navigation in a real mobile robot," *IEEE Trans. Systems, Man, and Cybernetics*, Vol. 26, No. 3, pp. 396-407, June 1996.

[2] P. Nordin and W. Banzhaf, "Real time control of a Khepera robot using genetic programming," *Cybernetics and Control*, Vol. 26, No. 3, pp. 533-561, 1997.

[3] S.-B. Cho and S.-I. Lee, "Evolutionary learning of fuzzy controller for a mobile robot," *Proc. Int. Conf. on Soft Computing*, pp. 745-748, Iizuka, Japan, 1996.

[4] M. J. Mataric, "Designing and understanding adaptive group behavior", *Adaptive Behavior*, Vol. 4, No.1, pp. 51-80, 1995.

[5] G.-B. Song and S.-B. Cho, "Incremental evolution of CAM-Brain", *Proc. AROB99*, pp.297-300, Beppu, Japan, 1999.

[6] F. Gers, H. de Garis and M. Korkin, "CoDi-1Bit: A simplified cellular automata based neuron model," *Proc. Conf. on Artificial Evolution*, Nimes, France, October 1997.

[7] W. Banzhaf, P. Nordin, and M. Olmer, "Generating adaptive behavior using function regression within genetic programming and a real robot," *Int. Conf. on Genetic Programming*, pp. 35-43, 1997.

[8] T. Tyrrell, "An evaluation of Maes's bottom-up mechanism for behavior selection", *Adaptive Behavior*, Vol. 2, pp. 307-348, 1994.

[9] D. E. Goldberg, *Genetic Algorithms in Search, Optimization, and Machine Learning*, Addison-Wesley Publishing Company, 1989.