Proceedings of 2001 IEEE International Symposium on Computational Intelligence in Robotics and Automation July 29 - August 1, 2001, Banff, Alberta, Canada

Dynamic Selection of Evolved Neural Controllers for Higher Behaviors of Mobile Robot

Kyung-Joong Kim and Sung-Bae Cho
Department of Computer Science
Yonsei University, Seoul 120-749, Korea
Email: uribyul@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 have also developed a method of applying CAM-Brain, evolved neural networks based on cellular automata (CA), to control a mobile robot. However, the direct evolution has a difficulty that the controller cannot generalize well to new environments. We attempt to solve it by incremental evolution, which starts with simpler environments and gradually develops the controller with more general and complex environments. We combine several behaviors evolved or programmed by dynamic selection mechanism for higher behaviors. In this paper, we evaluate the performance of robot using Khepera simulator. Simulation results show the possibility of easily developing higher behaviors by integrating CAM-Brain behavior modules.

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 presented CAM-Brain, evolved neural networks based on cellular automata [5,6], and applied it to controlling a mobile robot.

However, the controller obtained had a difficulty to adapt in changing environment. We attempt to devise a sophisticated method based on incremental evolution for solving this problem. Incremental evolution does not evolve controller directly to do goal behavior in an environment, but starting with simpler environments gradually develops the controller with more general and complex environments [7, 8].

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 do a simple behavior such as "going straight," "avoiding obstacles," "seeking object," and so on. They expect the controller combined with several modules can do complex behaviors [4,9,10].

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 Pattie Maes's Action Selection Mechanism (MASM) [10] to combine modules and control a mobile robot in simulated environments. The rest of this paper introduces CAM-Brain model, incremental evolution and basic behaviors, and presents the integration method in detail. The detailed description of simulation follows, and the results of simulation are given.

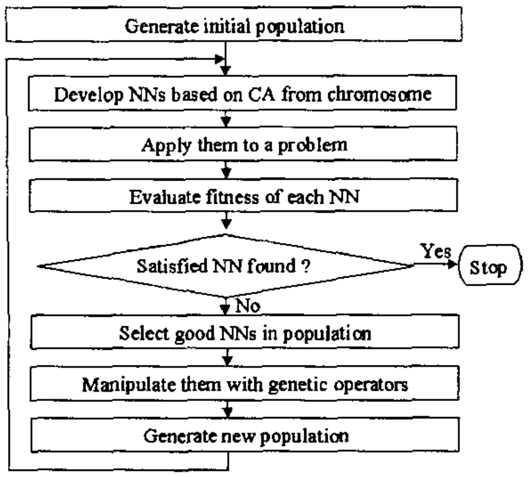
Generate initial population
Develop NNs based on CA from chromosome
Apply them to a problem
Evaluate fitness of each NN
Satisfied NN found ?
Yes
Stop
No
Select good NNs in population
Manipulate them with genetic operators
Generate new population

Figure 1: The evolution process of CAM-Brain.

2. CAM-Brain

CAM-Brain is a model based on CA which can perform complicated behavior by combining simple rules, and developed by its own chromosome: 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. Figure 1 shows the evolution process of CAM-Brain. One chromosome corresponds to one neural network. CAM-Brain finds optimal neural network structure using genetic algorithm. It is the basic idea of CAM-Brain that brain-like system can be constructed 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 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].

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. 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 2 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 blank arrows show the direction of signaling decided by chromosome. Figure 2(a) shows the process of sending a neuron in blank cells, where a neuron is located in (x_2, y_2). Figure 2(b) shows that the neuron cell sends growth signal to surrounding cells. Figure 2(c) shows that the cell state is changed by growth signal. Figure 2(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.

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. 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 neurons connected. 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 3 shows the process of signaling after neuron, axon and dendrite have been made.

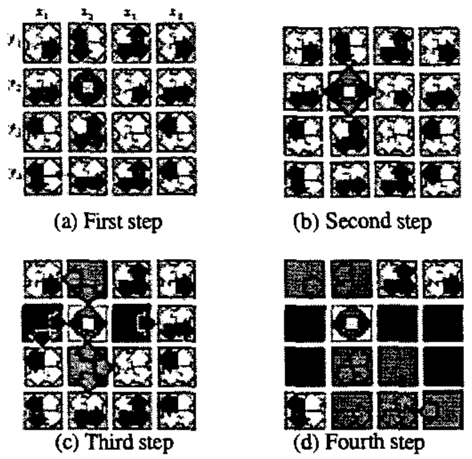

Figure 2: Growth phase: (a) Black arrow represents a neuron located in (x_2, y_2). (b) The neuron sends growth signals. (c) The cell state is decided according to the type of growth signals. (d) Propagating growth signals, blank cells become axon or dendrite.

2.2 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. 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 for the given problem.

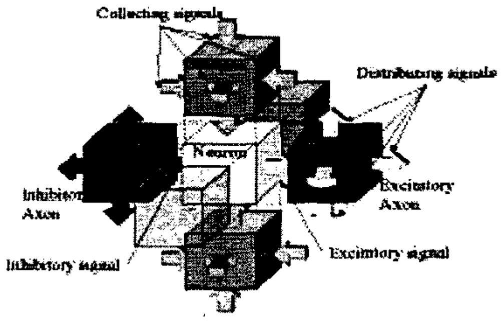

Figure 3: Signaling phase.

3. Incremental Evolution

Evaluation tasks $\{t_1, t_2, t_3, \ldots, t_n\}$ are derived by transforming a goal task in incremental evolution, where n is the number of tasks and t_n is the goal task. In this set, t_i is easier task than t_{i+1} for all i: $0<i\leq n$. Thus, population is evaluated in task t_i and then task t_{i+1} and it does in goal task, t_n, finally [7]. It is expected to produce complex and general behaviors which can adapt in changing environment. Figure 4 shows the incremental evolution of CAM-Brain. In this process, task is changed into more difficult one and new population is created from successful individuals when satisfied controller for the task is found.

The robot controller is evolved incrementally by starting with simpler environments and gradually evolving the controller with more general and complex environments. The environments get more sophisticated from straight movement to left and right turn movements. Consequently, the robot controller that can move straight and turn left and right can be obtained.

After the CAM-Brain module is evolved in the environment intended to go straight, successful chromosomes are copied to the next population. Then it is evolved in the environment intended to go straight and turn right. Progressing this process the controller evolves to go straight and turn left and right. Efficient evolution is expected because of the reduced search space by incremental evolution [8]. Figure 5 shows the trajectories of the successful robot in each environment. Environments are gradually bended from (a) to (f) and generality of successful controller are improved step by step.

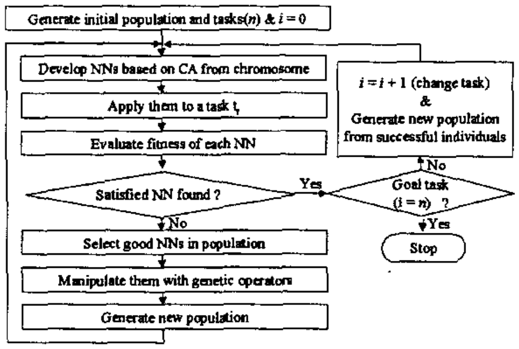
Generate initial population and tasks(n) & i = 0
Develop NNs based on CA from chromosome
Apply them to a task t_i
Evaluate fitness of each NN
Satisfied NN found ?
Yes
No
Select good NNs in population
Manipulate them with genetic operators
Generate new population
$i = i + 1$ (change task) & Generate new population from successful individuals
Goal task ($i = n$) ?
No
Yes
Stop

Figure 4: Incremental evolution of CAM-Brain.

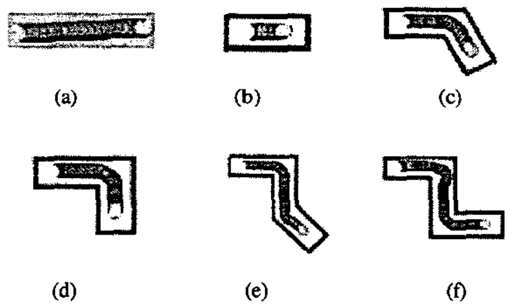

Figure 5: Trajectories of the successful robot in each environment.

4. Action Selection Mechanism

Maes's Action Selection Mechanism (MASM) was proposed originally as an improvement over previous approaches to action selection in the field of AI. In particular, it was proposed as an improvement on traditional planning systems and reactive systems [10].

4.1 Components

MASM is composed of nodes, internal links and external links. Each node has a set of preconditions. The preconditions are logical conditions about the environment that are required to be true in order for the node to be executable. The add list consists of conditions about the environment that the node is likely to make true. The delete list consists of conditions that are likely to be made false by the execution of the node. The final two components of the node are the activation level and the code that gets run if the node is executed. The internal and external links are specified in table 1.

Table 1: Internal and external links.

Internal links	
Predecessor link	If (proposition X is false) and (proposition X is a precondition of node A) and (proposition X is in the add list of node B), then there is an active predecessor link from A to B.
Successor link	If (proposition X is false) and (proposition X is in the add list of node A) and (proposition X is a precondition of node B) and (the node A is executable), then there is an active successor link from A to B.
Conflictor link	If (proposition X is true) and (proposition X is a precondition of node A) and (proposition X is in the delete list of node B), then there is an active conflictor link from A to B.
External links	
From sensors of the environment	If (proposition X about the environment is true) and (proposition X is a precondition of node A), then there is an active link from the sensor of the proposition X to node A.
From goals	If (goal Y has an activation greater than zero) and (goal Y is in the add list of node A), then there is an active link from the goal Y to node A.
From protected goals	If (goal Y has an activation greater than zero) and (goal Y is in the delete list of node A), then there is an active link from the goal Y to node A.

4.2 Procedure of Action Selection

The procedure used to select a node to execute at each time step is as follows:

1. Calculate the excitation coming in from the environment and the motivations.
2. Spread excitation along the predecessor, successor, and conflictor links.
3. Normalize the node activations so that the average activation becomes equal to the constant π.
4. Check to see if there are any executable nodes and, choose and execute the one with the highest activation.
5. If there is no executable node, reduce the global threshold and go to step 1.

A node is executable if all its preconditions are true and if its activation is greater than the global thresh old. If more than one node is executable, the one w ith the highest activation is chosen.

5. Experimental Results

In order to show that higher behaviors can emerge by combining lower simple behaviors, we have conducted the simulation with Khepera robot.

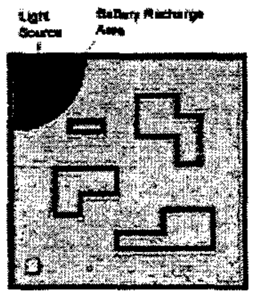

Figure 6: Simulation environment (simple environment).

5.1 Basic Behaviors

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

- Recharging Battery : If a robot arrives at battery recharge area, battery is recharged. This module enables the robot to operate as long as possible.
- Following Light : The robot goes to stronger light. This module must be operated to go to the battery recharge area because the light source exists in that area.
- Avoiding Obstacle : If the obstacles exist around the robot, it avoids them without bumping against them.
- Going Straight : If there is nothing around the robot, it goes ahead. This module takes it to move continuously without stopping.

Basic behaviors are programmed or evolved on CAM-Brain. Recharging Battery and Going Straight modules are programmed. Avoiding Obstacle and Following Light are evolved on CAM-Brain. Figure 6 shows the simulation environment. In this environment, black fan-shaped area represents "Battery Recharge Area" and robot can recharge battery only in this area. Light source exists in "Battery Recharge Area" and guides the robot to the black area.

5.2 Module Integration

In this section we apply the action selection mechanism to the robot control. Our environment requires 5 states, such as "In battery recharge area," "Obstacles are close," "Near battery recharge area," "Light is low," and "Nothing around the robot." They are set as follows :

- "In battery recharge area" : Check if robot is in battery recharge area.
- "Obstacles are close" : Check if the maximum value of distance sensors is larger than 700.
- "Near battery recharge area" : Check if the distance from robot to light source is less than 800.
- "Light is low": Check if the minimum value of light sensors is larger than 400.
- "Nothing around the robot" : Check if the maximum value of distance sensors is less than 700.

We set 2 goals such as "Full battery," and "Not zero battery." Because robot's battery decreases while robot moves, the robot intends to maintain high battery value to live long. They are set as follows :

- "Full battery" :

$$c = \frac{m-n}{m}$$

c : Value of "Full battery"
m : Maximum battery
n : Robot's battery

"Not zero battery" : Check if battery is less than half of the maximum battery.

5 states are binary-valued and 2 goals are of continuous values. Our MASM model is composed of 4 nodes, 5 states, 2 goals and their relationships. Figure 7 shows our MASM model. In this figure, each circle represents basic behavior module and each rectangle represents sensor or goal. Lines represent the relationship among components of ASM.

Each node has preconditions. Node must fulfill all preconditions to be executed. Table 2 describes preconditions and add lists of the nodes. Each node has one or two preconditions. The relationships among nodes are decided by successor links or predecessor links. If predecessor link is from A node to B node, successor link from B node to A node exists. If node A and node B are connected, they exchange their activation values while activation spreads.

Table 2: Preconditions and add lists of nodes

	Preconditions
Recharging Battery	In battery recharge area
Following Light	Light is low, Near battery recharge area
Avoiding Obstacle	Obstacles are close
Going Straight	Nothing around the robot
	Add lists
Recharging Battery	Full Battery, Not zero battery
Following Light	In battery recharge area
Avoiding Obstacle	Nothing around the robot
Going Straight	Obstacles are close, In battery recharge area, Near battery recharge area

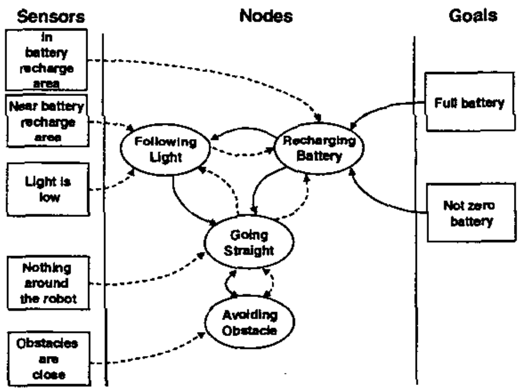

Figure 7: MASM model. Solid lines denote goal or predecessor connections, and dashed lines denote sensor or successor connections.

5.3 Result Analysis

Robot moves during 9329 steps. Figure 8 shows the robot trajectory in simple environment. This figure shows the robot navigates the environment without bumping and recharges battery. When robot is in battery recharge area, the state of "In battery recharge area" becomes true, and recharging battery module can be selected, because "In battery recharge area" is a precondition of recharging battery module. When battery is low, the corresponding goals increase and following light module is selected more frequently than other behavior modules.

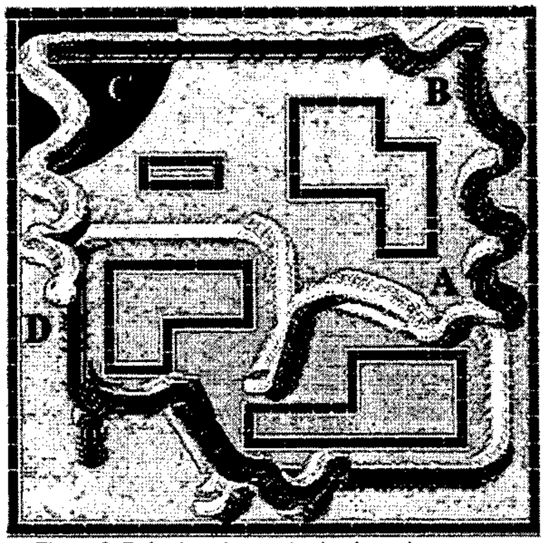

Figure 8: Robot's trajectory in simple environment.

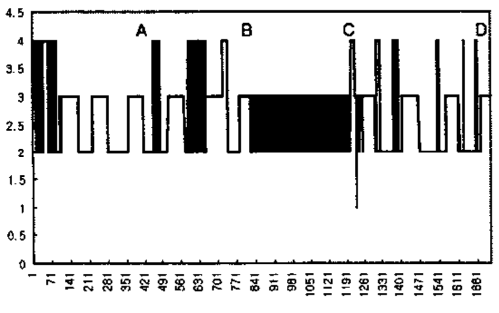

Figure 9: Action selection of robot (0=Recharging Battery 1=Following Light 2=Avoiding obstacle 3=Going Straight).

Robot selects avoiding obstacle and following light modules successively before battery recharge area. This makes robot go battery recharge area without bumping obstacles. By selecting lower level behaviors, robot can perform higher behavior which we cannot make easily. Figure 9 shows the sequence of the action selection in the environment. Robot selects frequently "Following Light," "Going Straight," and "Avoiding Obstacles."

6. Conclusions

In this paper, we apply Maes's Action Selection Mechanism to dynamically controlling a robot in simple environment. Robot selects basic behaviors from behavior networks. MASM gets signals from environments and internal motivations, and spreads the activation among internal links. We have four basic behaviors which are evolved on CAM-Brain or programmed. The robot controller is evolved incrementally by starting with simpler environments and gradually evolving the controller with more general and complex environments. Incremental evolution can evolve CAM-Brain module more efficiently than direct evolution, and the controller evolved by this method adapts to new environments. For higher behavior, several basic behavior modules are combined by action selection mechanism. Robot achieves goals by selecting basic behaviors. Contribution of this research is applying action selection mechanism to combining incrementally evolved neural network mobile robot controllers.

7. References

[1] D. Floreano and F. Mondana, "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] S.-B. Cho and G.-B. Song, "Evolving CAM-Brain to control a mobile robot," *Applied Mathematics and Computation*, vol. 111, pp. 147-162, May, 2000.

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

[7] F. Gomez and R. Miikkulainen, "Incremental evolution of complex general behavior," *Adaptive Behavior*, Vol. 5, pp 317-342, 1997.

[8] I. Harvey, P. Husbands and D. Cliff, "Seeing the light: Artificial evolution, real vision," *Proc. of 3rd Int. Conf. on Simulation of Adaptive Behavior*, pp. 392-401, MIT Press/Bradford Books, 1994.

[9] 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.

[10] P. Maes, "How to do the right thing," *Connection Science Journal*, Vol 1, No. 3, pp 291-323, 1989.