Proceedings of the 32nd ISR(International Symposium on Robotics), 19-21 April 2001

Robot Action Selection for Higher Behaviors with CAM-Brain Modules

Kyong-Joong Kim and Sung-Bae Cho
Department of Computer Science
Yonsei University
134 Shinchon-dong Sudaemoon-ku, Seoul 120-749, Korea
E-mail : uribyul@candy.yonsei.ac.kr, sbcho@csai.yonsei.ac.kr

Abstract

CAM-Brain is a neural network based on cellular automata, which model complex phenomenon by simple rules, and optimized by genetic algorithm. Like many evolutionary approaches to robot control such as neural network evolved by genetic algorithm and fuzzy controller optimized by genetic algorithm, CAM-Brain can be applied to robot control. Behavior modules such as avoiding obstacles and following light are evolved on CAM-Brain. They are evolved incrementally by starting with simpler environment needed simple behavior and gradually making it more complex and general for complex behaviors. Because evolving higher behaviors directly is difficult, we combine several basic behaviors by action selection mechanism. Robot selects one of the basic behavior modules evolved or programmed at each time. We evaluate the performance of robot using Khepera simulator and modify simulator interface for visualization of the action selection procedure. Simulation results show the possibility of the action selection method for higher behaviors with CAM-Brain 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, combining fuzzy controller with genetic algorithm [2] and programming. In previous work [3], we presented CAM-Brain, evolved neural networks based on cellular automata [3,4], 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 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 [5,6,7].

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) [5,6,7] to combine modules and control a mobile robot in simulated environments. The rest of this paper introduces CAM-Brain model and basic behaviors, and presents the integration method in detail. The detailed description of simulation follows, and the results of simulation are given.

2. CAM-Brain

CAM-Brain is a neural network based on cellular automata which model complex phenomenon by simple rules, and optimized by genetic algorithm. Figure 1 shows the evolution of CAM-Brain.

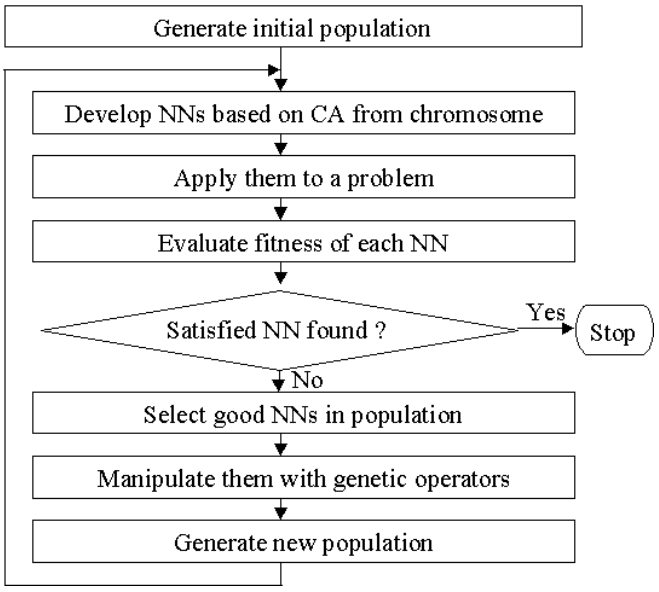


Figure 1: Evolution of CAM-Brain.

2.1. Neural Networks based on CA

CA are population of interacting cells, each of which is itself a computer (automaton) and can represent many kinds of complex behavior by building appropriate rules into it. CA forms either a 1-dimensional string of cells, a 2-D grid or a 3-D solid. Mostly the cells are arranged as a simple

rectangular grid. CA has the three essential features of state, neighborhood, and program. Its state is a variable that takes a different separate for each cell. The state can be either a number or a property. Its neighborhood is the set of cells that it interacts with. In a grid these are normally the cells physically closest to the cell. Its program is the set of rules that define how its state changes in response to its current state, and that of its neighborhood.

CAM-Brain's neural network structure composed of blank, neuron, axon and dendrite are grown inside 2-D or 3-D CA-Space by state, neighborhoods and rules encoded by chromosome. Roles of each cell are as follows.

- Blank: If cell state is blank, it represents empty space and cannot transmit any signals.
- Neuron: It 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: It sends signals received from neurons to the neighborhood cells.
- Dendrite: It collects signals from neighborhood cells and passes them to the connected neuron in the end.

Neighborhood cells of one cell mean surrounding cells (North, South, West, and East in 2-D CA space and Top and Bottom added to them in 3-D CA space).

The information encoded in a chromosome determines a neural network architecture. To represent the whole structure of a neural network, a chromosome has the same number of segments with the cells in CA-space and each segment has information of each cell. A segment can change blank cell to neuron cell (NS bit) and decides the directions of sending received signals to neighborhood cells (N, S, E, W, T and B bits). The signals can be only sent to the direction in which the bit corresponds to 1. Figure 2 shows the chromosome of CAM-Brain.

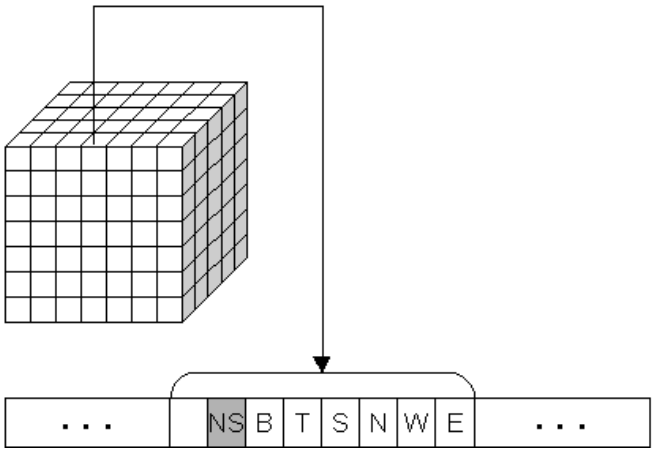


Figure 2: Chromosome of CAM-Brain.

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

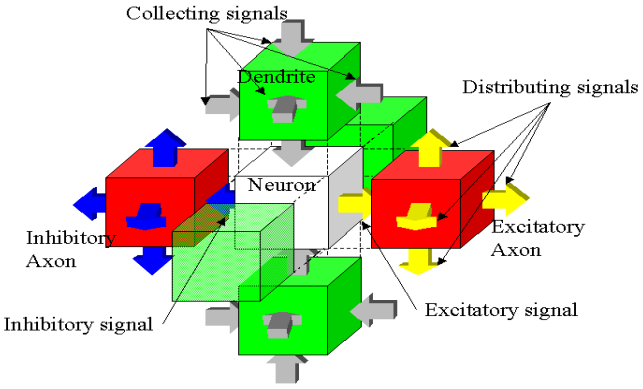


Figure 3: Signaling phase.

2.2. 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 [8]. It is expected to produce complex and general behaviors which can adapt in changing environment. Figure 4 shows the procedure of the incremental evolution with CAM-Brain.

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. Figure 5 shows the trajectories of the successful robot in each environment.

3. Action Selection Mechanism

In behavior-based robotics the control of a robot is shared between a set of purposive perception-action units, called behaviors. Based on selective sensory information, each behavior produces immediate reactions to control the robot with respect to a particular objective, i.e., a narrow aspect of the robot's overall task such as obstacle avoidance or wall following. Behaviors with different and possibly incommensurable objectives may produce conflicting actions that are seemingly irreconcilable. Thus a major issue in the design of behavior-based control systems is the formulation of effective mechanisms for coordination of the behaviors' activities into strategies for rational and coherent behavior. This is known as the action selection problem (also refereed to as the behavior coordination problem) [7]. We use activation networks proposed by Pattie Maes to combine basic behavior modules evolved on CAM-Brain for controlling a mobile robot.

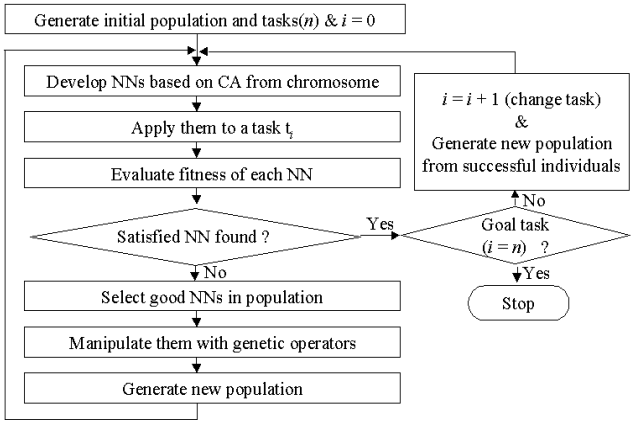


Figure 4: Incremental evolution of CAM-Brain.

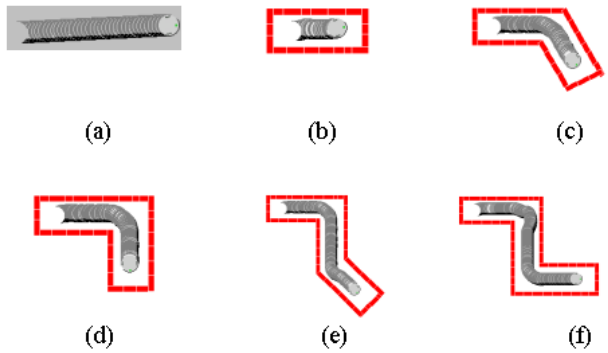


Figure 5: Trajectories of the successful robot in each environment.

3.1. Activation Networks

An activation network 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 links are specified in Table 1. The external links providing input to the network are specified in Table 2. Table 1 and 2 describe the links.

Table 1: Internal links.

Predecessor link	If proposition X is false, 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, proposition X is in the add list of node A, 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, 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.

Table 2: 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.

3.2. Procedure of Action Selection

Action selection procedure of an activation network 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 whether any nodes are executable and, if so, choose the one with the highest activation, execute it, and finish.
5. If no node is executable, reduce the global threshold and repeat the cycle.

4. Experimental Results

We evaluate the performance of robot using Khepera simulator and modify simulator interface for visualization of the action selection procedure. At first, the battery level of robot is 2500. Because robot uses battery to go around the environment, robot has to recharge battery at low battery level. Recharging battery behavior is conducted only when robot is inside battery recharge area. To solve this problem, we combine several basic behaviors evolved on CAM-Brain or programmed by activation network. Figure 6 shows the simulation environment.

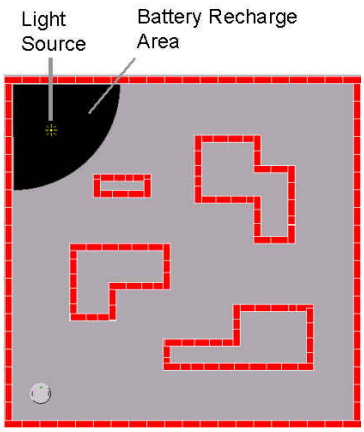


Figure 6: Simulation environment.

4.1. Experimental Environment

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 incrementally evolved on CAM-Brain.

4.2. Action Selection Model

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 attempts to maintain high battery value to operate 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 continuous values. Our MASM model is composed of 4 nodes, 5 states, 2 goals and their relationships. Figure 7 shows our action selection model. Each node has preconditions, and must fulfill all preconditions to be executed. Table 3 describes preconditions of the nodes. Each node has one or two preconditions.

Table 3: Preconditions of nodes.

Node	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

Table 4: Add lists of nodes.

Node	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

Table 5: Relationships between nodes.

Predecessor link
Recharging Battery → Following Light Recharging Battery → Going Straight Following Light → Going Straight Going Straight → Avoiding Obstacles Avoiding Obstacles → Going Straight
Successor link
Following Light→ Recharging Battery Going Straight → Following Light Going Straight → Recharging Battery Going Straight → Avoiding Obstacles Avoiding Obstacles → Going Straight

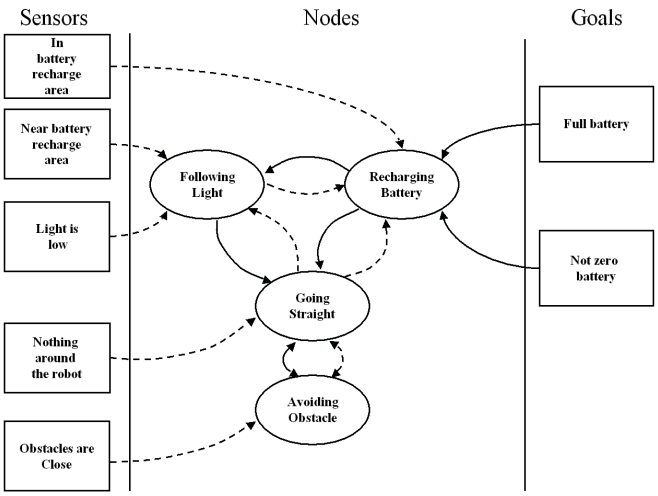


Figure 7: Action selection model. Solid lines denote goal or predecessor connections, and dashed lines denote sensor or successor connections.

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 4 describes add lists of nodes and Table 5 describes the relationships between nodes.

4.3. Simulation Results

We use the Khepera robot simulator to evaluate the performance of the proposed action selection model. We modify original Khepera robot simulator to display which behavior is selected and how the activation is exchanged. Figure 8 shows the interface of the simulation system.

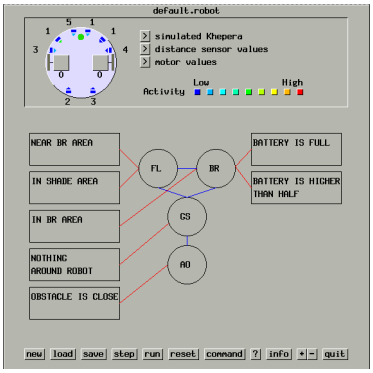


Figure 8: Action selection model in simulator.

Figure 9 shows the simulation result in simulation environment. Robot selects one behavior at each time and executes that module to achieve goals. Robot moves 4690 times while recharging battery behavior is executed 13 times. Because robot has 2500 level of battery at first, robot must recharge battery to live long period. To solve this problem, we assign two goals of activation network: "Full battery", and "Not zero battery." Activation network helps robot to select battery recharge behavior when robot is inside battery recharge area. Figure 10 shows the action selection of robot for 4690 times and the change of battery levels. This shows how the robot can survive when robot's battery level is very low. Robot selects avoiding obstacle and going straight for the most part.

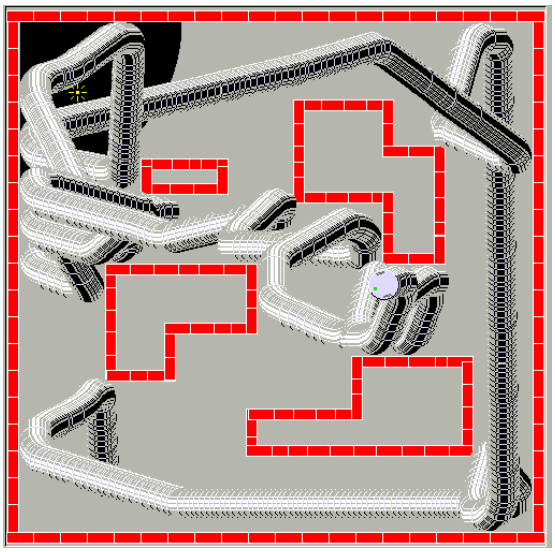
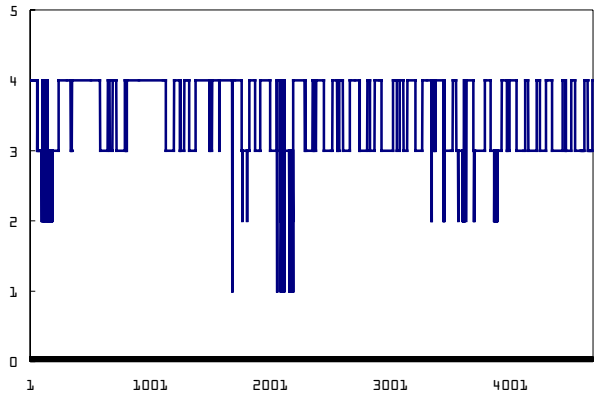
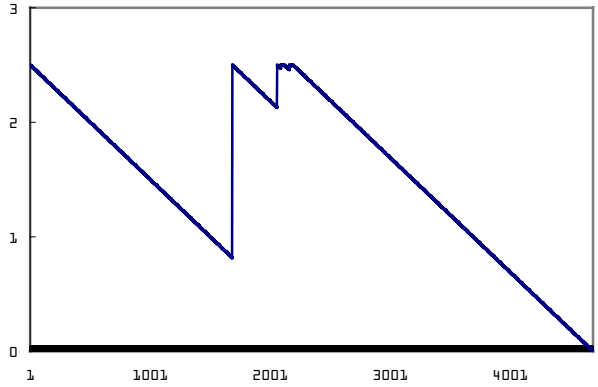


Figure 9: Robot's trajectory.



(a)



(b)

Figure 10: Action selection of robot (a) selected action at each time (1=Recharging battery, 2=Following light, 3=Avoiding Obstacle, 4=Going Straight) (b) battery level change (1 means 1000 battery level).

Figure 11 shows the action sequence of robot when it is in left-bottom corner of simulation environment. Before the obstacle is close to the robot, robot chooses going straight. To avoid the obstacle, robot chooses avoiding obstacle mainly. After that, robot chooses going straight and following light for the most part. Because another obstacle comes, robot chooses avoiding obstacle and following light for the most part. The reason that robot chooses following light is the increase of light. After avoiding the first obstacle, robot has no obstacle with light source.

5. Conclusions and Future Works

We have developed several behavior modules that conduct complicated action using neural networks based on cellular automata. To find optimal behavior module, we use incremental evolution approach. This helps to find optimal solution in the effective way. To achieve higher behaviors, we combine several behavior modules evolved or programmed by using activation networks which coordinate behavior modules. Simulation results and some analyses show the usefulness of this approach. For the complex higher behaviors, we will have to use more behavior modules than four and combine them using activation network.

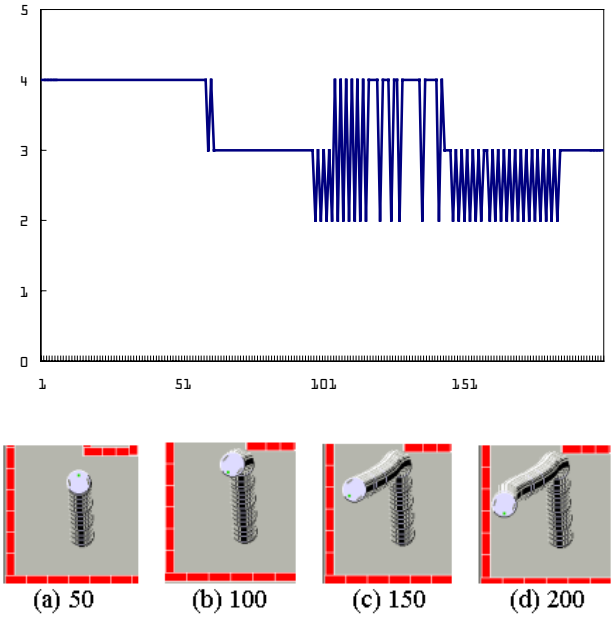


Figure 11: Action sequence of robot in left-bottom corner of simulation environment.

6. 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] S.-B. Cho and S.-I. Lee, "Evolutionary learning of fuzzy controller for a mobile robot," *Proc. Int. Conf. on Soft Computing*, vol. 2, pp. 745-748, Oct, 1996.

[3] 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.

[4] 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.

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

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

[7] P. Pirjanian, "Behavior coordination mechanism–state-of-the-art," *Tech-report IRIS-99-375*, Institute for Robotics and Intelligent Systems, School of Engineering, University of Southern California, Oct, 1999.

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