Proceedings of the 2003 IEEE
International Conference on Robotics & Automation
Taipei, Taiwan, September 14-19, 2003

Design and Implementation of a Behavior-Based Control and Learning Architecture for Mobile Robots

Il Hong Suh[1], Sanghoon Lee[1], Bong Oh Kim[1], Byung Ju Yi[1], and Sang Rok Oh[2]
[1]Department of Electrical Engineering and Computer Science
Hanyang University, Seoul, KOREA
Tel : +82-2-2290-0392, E-mail : ihsuh@hanyang.ac.kr
[2]Intelligent System Control Research Center, Korea Institute of Science and Technology,
P. O. Box 131, Cheongryang, Seoul 130-650, KOREA

Abstract **– A behavior-based control and learning architecture is proposed, where reinforcement learning is applied to learn proper associations between stimulus and response by using two types of memory called as short Term Memory and Long Term Memory.**
In particular, to cope with delayed-reward problem, a knowledge-propagation (KP) method is proposed, where well-designed or well-trained S-R(stimulus-response) associations for low-level sensors are utilized to learn new S-R associations for high-level sensors, in case that those S-R associations require same objective such as obstacle avoidance. To show the validity of our proposed KP method, comparative experiments are performed for the cases that (i) only a delayed reward is used, (ii) some of S-R pairs are preprogrammed, (iii) immediate reward is possible, and (iv) our KP method is applied

I. INTRODUCTION

For implementation of autonomous and intelligent system, a lot of research works have been done in many areas including cognition, reasoning, and learning. When compared with level of human intelligence, behaviors of low-level animals could not be considered as intelligent one. But, recently, it has been understood that those behaviors should be counted as sufficiently intelligent when considering the environment of the low-level animals. Some robot control systems involving behavior-based architectures of such low-level animals have been proposed [1]-[4].

A distinct feature of behavior-based control architecture can be described as follows: an action is selected without use of cognitive models, reasoning, and planning, among possible actions of a robot associated with a stimulus (or state) when an environmental state is given to a robot. Here, there are two representative behavior-based control architectures; Subsumption [4] and Schema [3].

In the Subsumption architecture, an action with higher priority can always subsume other possible actions associated with the state. And in schema-based architectures, all actions associated with the state would be combined. In those architectures, if associations between set of behaviors and set of stimulus are fixed, then the robot shows same behaviors under same environment. But, for the adaptation on uncertain and dynamic environment, it is necessary to change associations; strengthening or weakening current connections between sensor state and actions, and linking a new sensor state with proper actions (or behaviors). For such a purpose, reinforcement learning techniques have been employed [5]-[8].

When designing an intelligent robot by using a behavior-based control architecture involving reinforcement learning, several factors should be taken into account. For example, all necessary state-action pairs (or stimulus-response behaviors), and their relations should be defined and designed, where relations could be setup in a hierarchical or a parallel fashion [2].

On the other hand, it usually takes a long time to learn some necessary associations between stimuli and behaviors by reinforcement learning technique [11][12]. There are two types of rewards; immediate reward and delayed reward.

Delayed rewards have to be used to evaluate the suitability of past course of action of the robot. While this process is theoretically possible, it tends to be unacceptably inefficient: feedback information is often too rare and episodic for an effective learning process to take place in realistic robotic applications. A way to bypass this problem is to use a trainer to continuously monitor the behavior of a robot and provide immediate reinforcements. To produce an immediate reinforcement, the trainer must be able to judge how well each single robot action fits into the desired behavior pattern.

In this paper, a behavior-based control and learning architecture is proposed, where a behavior is selected among behaviors associated with a given sensor state by considering internal desires, and also reinforcement learning is applied to learn proper associations between sensor states and behaviors. In our design, two types of memory are employed for the learning. One is short term memory (STM) in which stimulus-response (SR) pairs are recorded along the time. The other is long term memory (LTM) to which stimulus-response pairs are moved from STM together with their reliability, when a reward is received. And in particular, to solve delayed-reward problem, a knowledge-propagation (KP) method is suggested, where well-designed or well-trained S-R associations for low-level sensors such as ultra sonic sensors are utilized to learn new S-R associations for high-level sensors such as CCD camera, in case that those S-R associations could require same objective such as obstacle avoidance.

To show the validity of our proposed KP method, comparative experiments are performed for the cases that (i) only a delayed reward is used, (ii) some of S-R pairs are preprogrammed, (iii) immediate reward is possible, and (iv) our KP method is applied. From such experiments, we will show that KP method could be more effective in learning S-R behaviors in delayed reword environment than other methods.

0-7803-7736-2/03/$17.00 ©2003 IEEE

II. A Behavior-based Control and Learning Architecture

A. General Architecture

Block diagram of our proposed behavior-based control and learning architecture is represented as in Fig. 1 In our architecture, there are 5 modules; Sensor Module, Perception Module, Memory Module, Motor Module, and Behavior Selection Module. Sensor Module is composed of physical sensor and logical sensor. Physical sensor is attached into the robot, and transforms physical quantity into electrical signals. And, logical sensor can be considered as processing part of the physical sensor signals to detect stimulus from the environment. Explicit use of the logical sensor can enhance the openness of our behavior-based architecture in the sense that any physical sensors can be easily attached into our software system thru our logical sensors by simply defining necessary information into pre-specified DB protocols. Perception Module (PM) plays a role of transferring output of logical sensors to the behavior modules, where each logical sensor information is properly weighted by stimulus recognition filters. Memory module consists of short-term memory (STM) and long-term memory (LTM).

STM is used to record stimulus-response pair at every tick (sampling time) until a new reward or reinforcement is enforced. Then, the stored information is transferred to LTM with reliability index, where S-R pairs are analyzed to learn current association between stimulus and response.

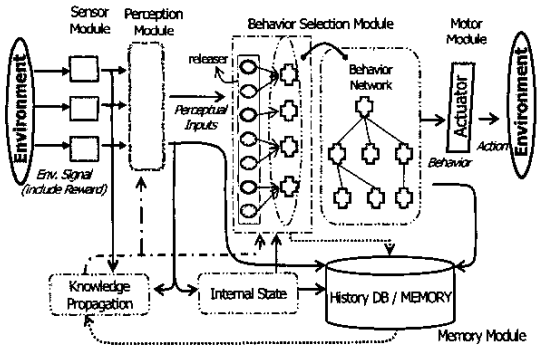


Fig. 1 Behavior-based control architecture.

Motor module plays a role of controlling actuators to perform selected behaviors. Behavior Selection Module includes releaser which activates behaviors, behavior network which explains relations among behaviors, and associations between releasers and behaviors. Output of PM is transferred to all releasers, and every releaser computes how much current stimulus coming from PM is related or associated with its own behavior, where one releaser handling one stimulus is connected with one behavior to form an S-R behavior [2]. Behavior value is then determined by using only releaser output, or by using releaser output as well as internal state values [5]-[8]. Behavior selection is done in such a way that maximum behavior value is chosen [14], or some behaviors [15], where values are greater than a pre-specified threshold value, are competed by means of a competition network. Maximum value method lets the system perform the action right after action selection. But, competition method requires competition among behaviors with values above a threshold, which relatively cause complexity problem [16]. On the other hand, priority-based behavior selection can be also applied. All such behavior selection techniques do not employ cognitive reasoning and/or planning to select a behavior responding to incoming stimulus. In this sense, we call such techniques as behavior-based technique.

In our work, a competition-based action-selection is utilized. And, to learn appropriate association between stimulus and its possible responding behaviors, reinforcement learning will be applied. Specifically, our architecture is designed such that new S-R behaviors can be inserted into the behavior network whenever strength of association between a new stimulus and some behavior gets higher than a threshold.

B. Memory for Learning

Behavior exploration is executed to find a proper behavior for a new stimulus (or environmental state), and to learn optimal behavior which is better than current behavior.

To find a proper behavior for a new stimulus, an arbitrary behavior is performed. And rewards may be received or not. Such an arbitrary behavior to respond to a new stimulus is recorded in STM along the time regardless of reward signal. If a reward is received, then past history of S-R behaviors are transferred from STM to LTM. In this process, if some S-R behaviors are already registered in LTM, reliability of those S-R behaviors are updated by

$$V_{ij} = V_{ij(t-1)} + \eta \frac{1 - V_{ij(t-1)}}{d_k} \qquad (1).$$

where

η : learning rate,

V_{ij} : reliability of between stimulus and behavior at time t,

d : time difference on ST memory,

i : index of stimulus,

j : index of behavior,

k : index of ST memory.

And, new S-R behaviors whose reliability value is above a threshold are registered in the behavior network. Exploration for optimal behavior can be performed by ε-greedy policy [17] or by skill learning [18] which generates a new behavior by changing motor command parameters. Here, ε-greedy policy is employed.

In Figs. 2 and 3, operational procedure and its pseudo codes of STM and LTM are summarized.

It is remarked that our learning as explained above can be considered as an associative learning between stimulus and behaviors using STM and LTM. There are two types of associative learning; one is classical conditioning and the other is operant conditioning. Classical conditioning is an association forming process by which a stimulus that previously did not elicit a response comes to elicit a

response. Operant conditioning is a process through which the consequences of a response increase or decrease the likelihood that the response will occur again. In our case, operant conditioning is employed for learning.

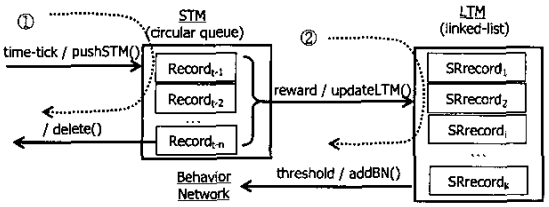


Fig. 2 STM and LTM operations.

```
① if received<timetick> then
   Begin thread
    if( query(stimulus,behavior) ) // to PM and BSM
    {
      makeRecord()              // stimulus and behavior data
      pushToSTM( data );
    }
   End thread

② if received<reward> then
   Begin thread
    do
    {
      popFromSTM()              // pop the data from STM
      if( find( stimulus, behavior) ) // in LTM
        updateReliability() // update reliability in LTM using Eq. (1)
      else
      {
        insertData()            // insert data to LTM
        updateReliability()
      }
    }while( isEmptySTM() )
   End thread
```

Fig. 3 Pseudo codes for STM and LTM operations.

III. Knowledge Propagation

A. Behavior Learning by Delayed Reward

In general, it is difficult for the robot to get rewards or reinforcement immediately just after a behavior in applying reinforcement learning technique. This is due to a difficulty that a robot cannot evaluate how well its behavior is fitted for the goal or for the satisfaction of its own motivation. Such as evaluation may be possible, if a trainer is available. In this case, the trainer should watch out the robot until it completes necessary reinforcement learning which might require a long time operation. Thus, it is practically difficult to apply. After all, delayed-rewards have to be received. Delayed reward greatly slow down the learning process, because reward signal is often too rare and episodic for an effective learning process to take place in realistic applications. Several techniques can be considered to bypass such a delayed reward problem [19].

First, robot is driven to completely learn necessary S-R behaviors. In this case, it cannot be known when learning is completed. In fact, a complete learning may not be guaranteed due to insufficient number of episodes.

Second, reinforcement program (RP) can be constructed to replace a human trainer. This artificial trainer could provide the robot with immediate reward, and thus robot can be effectively trained by reinforcement learning. However, it is not clear how we can construct such a reinforcement program, since we should know all possible stimuli coming from robot sensors, and should understand what to do for those stimuli. Unfortunately, such a perfect understanding may be unrealistic. If wrong RP is applied, incorrect S-R behaviors may be generated, which we would like to avoid in the design of robot actions.

Third, direct programming (DP) can be applied, if we exactly know what behavior should be matched with a given stimulus. DP can reduce the number of S-R behaviors to be learned. But, all S-R behaviors cannot be directly preprogrammed, since there could be a lot of unexpected stimuli to the robot. Thus, even in DP case, learning is still required as in other cases. It is expected that DP can enhance speed of reinforcement learning owing to reduction of number of S-R pairs to be learned.

In this work, to partially cope with delayed-reward problem of reinforcement learning, a novel type of knowledge propagation (KP) is proposed. To be specific, suppose that a robot is perfectly trained to respond to a stimulus from a low level of robot sensor, which let the robot achieve an internal desire or motivation. Also suppose that a robot gets a new stimulus from a different or high level of robot sensors, and robot should respond to the new stimuli to achieve the same type of internal desire or motivation of existing S-R behaviors made by the low level of robot sensors. Then, new S-R behavior made by the high level sensor can be expected to be learned by using the existing S-R behaviors made by low-level sensor.

For example, suppose that obstacle avoidance behavior by sonar sensors (low-level sensors) be innately programmed or pre-learned in a mobile robot. Then if a new vision sensor (high-level sensor) is mounted to the mobile robot, an obstacle can be detected as a new stimulus from the vision sensor, while the robot could avoid the obstacle by the obstacle avoidance behavior learned by sonar sensors. For this case, a new obstacle-avoidance behavior can be effectively learned to respond to the obstacle stimulus coming from the vision sensor by employing such an existing obstacle-avoidance behavior of the robot. We will call this type of learning as knowledge propagation (KP). Fig. 4 shows the conceptual diagram of our proposed KP module.

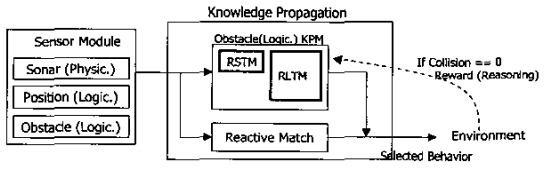


Fig. 4 Knowledge Propagation Module.

B. Knowledge Propagation Algorithm

As shown in Fig. 4, two types of memories, RSTM and RLTM, are included in the module. Recall that in STM, every selected behavior and its associated stimulus is recorded at every time tick. But, in RSTM, a stimulus-

response for a sensor (e.g., sonar sensor) is recorded, only if the stimulus for the sensor is considered to be associated with other stimulus from a different sensor (e.g., vision sensor). When a reward is given, history of S-R behaviors for the sensor (usually, relatively low-level sensor) is transformed to RLTM together with reliability index computed by the similar equation of Eq. (1). Detailed memory operation for RSTM and RLTM is omitted since it is similar with that for STM and LTM in Sec2.

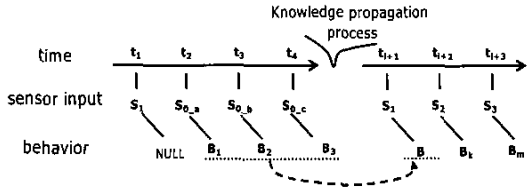


Fig. 5 Knowledge Propagation Process.

To be more specific, consider the timing chart in Fig. 5, when at time t1, a new stimulus S1 (obstacle) from high-level sensor like CCD camera is applied to the robot. And at time t2, t3, and t4, the robot receives stimulus S0_a, S0_b, and S0_c, respectively, from a low-level sensor like sonar sensor, and thus executes innately preprogrammed avoidance behaviors B1, B2, and B3, respectively. If objectives of S1-R and S0-R behavior are known as the same, then KP process begins to analyze B1, B2, and B3 to make a new behavior B for the stimulus S1. Here, B is made by vector sum of B1, B2, and B3, for the robot to rapidly avoid obstacle when receiving the stimulus S1. After all, a new S1-B pair is made by the knowledge propagated from S0_a-B1, S0_b-B2, and S0_c-B3 pairs, and is then available to the robot. This process is done in reasoningKPM() of the pseudo code in Fig. 6.

```
sensorModuleUpdate()
perceptionModuleUpdate()
reactiveMatch()
selectBehavior()

if( existKPMPercept() )
  saveKPMMemory( KPMPercept, currBehvaior )

if( receivedReward() )
{
  reasoningKPM()        // start reasoning
  updateKPMMemory() // update reliability in RLTM
}
```

Fig. 6 Pseudo Code of Knowledge Propagation.

It is remarked that KP method would be better than DP and KP methods owing to the use of RSTM. To be specific, recall that each logical sensor is connected with an RSTM. Then, we can understand that when delayed reward is received, KP process gets to know what behaviors should be rewarded by searching for RSTM or by analyzing RSTM. This process can be considered as replacing delayed reward with immediate reward. Thus, KP can show enhanced learning speed. And also, a behavior is chosen generated by referring to those rewarded behaviors, which reduces exploration trials. But, RP and DP methods have to explore behaviors until a robot receives a positive reward for the behavior on each stimulus.

IV. EXPERIMENTS

A. AmigoBot System

AmigoBot of ActivMedia Robotics Company [22] is employed for our experiments as shown in Fig. 7. The robot has 8 sonar sensors and one CCD camera. Pentium PC(233MHz) is used to control the robot, where 900 MHz RF modem is utilized to communicate with robot, and 2.2 GHz A/V receiver is used to get the video data of CCD camera.

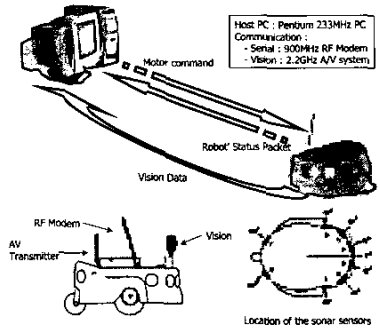


Fig. 7 Control System and Robot.

B. Learning Experiments

To show the validity of our proposed architecture and KP algorithm, experimental set-up is organized as in Fig. 8, where successful robot task is defined as running along the circular track without violation of traffic signal and without collision to the walls as well as an obstacle.

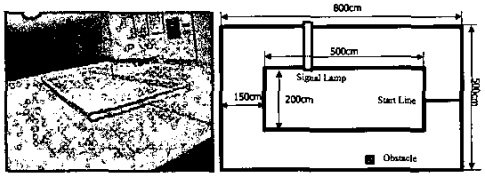


Fig. 8 Environment.

When a run is considered successful, the robot will get a reward. Here, it is assumed that physical properties of the track such as, length and width should not be changed during whole experiments. But a traffic signal in the track will be arbitrarily changed from red to green or from green to red, and an obstacle is given at any place in the track.

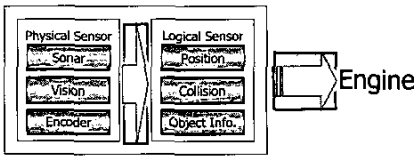
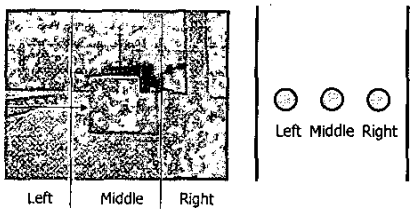
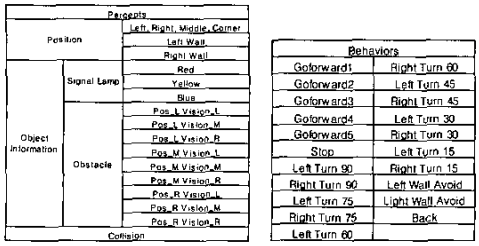


Fig. 9 Sensor Module



(a) Vision Physical Sensor (b) Position Logical Sensor

Fig. 10 Obstacle Logical Sensor



(a) Percepts (b) Behaviors

Fig. 11 Percepts and Behaviors

TABLE I
Innate Knowledge

Stimulus	Behavior
Corner	LeftTurn90
Left Wall	LeftWallAvoid
Right Wall	RightWallAvoid
Collision	Back

TABLE II
Experiment Results in case Robot has only innate knowledge

	Simulation	Experiment
Success	5	2
Episodes	250	100

Sensor Module of the architecture in Fig. 1 is designed as in Fig. 9. And 9 different stimuli for obstacle detection are employed as in Fig. 10, and percepts and behaviors are designed as in Fig. 11. Initial stimulus-response pairs are given in TABLE I as innate knowledge. TABLE II shows how the robot could behave in the track only with innate knowledge in TABLE I. As shown in TABLE II, only 2% of 100 real episodic robot runs could be counted successful. This implies that no further learning is possible by means of reinforcement learning due to shortage of number of successful episodes which cause delayed-reward problem.

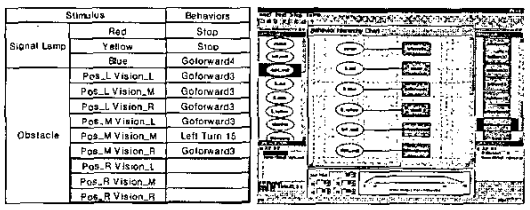


Fig. 12 S-R editor tool for DP method.

Now, to show the effectiveness of KP method, same experiment as above is performed for the DP, RP, and KP methods. In the DP, correct responses to the traffic signal, and correct behaviors to same obstacle-stimuli are directly programmed by using our developed S-R editor tool as shown in Fig. 12. Here, remaining 3 S-R behaviors are to be learned by reinforcement learning technique. For the RP, a computer program is written and a human trainer is employed to give proper rewards to the responses for the traffic signals and an obstacle. And finally, for the KP, S-R behaviors for the sonar sensors and position logical sensors are designed. Fig. 13 shows an exemplar design of such S-R behaviors.

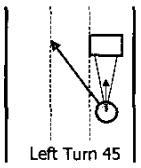


Fig. 13 An Example of S-R Behaviors for sonar sensor and logical position sensor.

TABLE III
Reliability of LTM

Behavior ID	Obstacle Logical Sensor														
	M-M			M-R			R-L			R-M			R-R		
	DP	RP	KP	DP	RP	KP	DP	RP	KP	DP	RP	KP	DP	RP	KP
goforward1	0	0	0	0	0.13	0	0	0	0	0	0	0	0	0	0
goforward2	0	0	0	0	0.17	0	0	0	0	0	0	0	0	0	0
goforward3	0	0	0	1	0.18	0.73	0	0	0	0	0	0	0	0	0
goforward4	0	0	0	0	0.33	0	0	0	0	0	0	0	0	0	0
goforward5	0	0	0	0	0.13	0	0	0	0	0	0	0	0	0	0
stop	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
left turn 90	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
right turn 90	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
left turn 75	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
right turn 75	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
left turn 60	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
right turn 60	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
left turn 45	0	0	0	0	0	0	0.08	0.1	0	0.12	0.14	0	0	0	0
right turn 45	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
left turn 30	0	0.12	0	0	0	0	0.29	0.42	0.67	0.17	0.36	0.48	0.04	0.29	0.33
right turn 30	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
left turn 15	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
right turn 15	1	0.18	0.27	0	0	0	0	0	0	0	0	0	0	0	0
left wall avoid	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
right wall avoid	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
back	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

50 episodic trials are performed for each method. TABLE III shows the reliability index values for S-R behaviors of each method. Here, we omitted S-R behaviors for traffic signals and other stimuli. Reliability index values show how much a stimulus is strongly connected to a behavior. It is here observed from TABLE III that RP showed 5 behaviors such as goforward1, goforward2, goforward3, goforward4, and goforward5 for the case of M-R. M-R is the case from Fig. 10 that robot is in the middle and obstacle is in the right. On the other hand, for M-R, KP method showed an appropriate behavior, goforward3, with the reliability value higher than RP. It can be also observed from TABLE III that reliability values of S-R behaviors of M-M, R-L, R-M, and R-R cases for KP method are higher than those for DP and RP methods. This implies that for the same episodic trial numbers, KP method enable the robot to learn necessary S-R behaviors more reliable than other two methods in delayed reward environment.

It is remarked that fast learning and high reliability of KP method would come from the use of RSTM which is connected with a logical sensor. Specifically, when a delayed reward is received, KP process can immediately know what behaviors should be rewarded by searching for

RSTM or by analyzing RSTM. That is, KP process can be considered as replacing delayed reward with immediate reward. Thus, KP can show an enhanced learning speed. But, RP and DP methods have to explore behaviors until a robot receives a positive reward for the behavior on each stimulus. Thus, for RP and DP methods, it may take a relatively long time to learn correct S-R behaviors.

V. CONCLUSIONS

In this paper, a behavior-based control and learning architecture was proposed, where reinforcement learning was applied to learn proper associations between sensor states and behaviors. To solve delayed-reward problem, a knowledge-propagation (KP) method was proposed. And to show the validity of our proposed KP method, comparative experiments were performed for the cases that (i) only a delayed reward was used, (ii) some of S-R pairs were preprogrammed, (iii) immediate reward was possible, and (iv) our KP method was applied. From the experiments, we showed that KP method enabled the robot to learn necessary S-R behaviors faster and more reliable than RP and DP methods.

Acknowledgement

This work has been supported in part by Next-Generation New Technology Development Program entitled as Control/Recognition Technology for Personal Robots.

VI. REFERENCES

[1] R.C. Arkin, Behavior-Based Robotics, MIT Press, 1998.

[2] R.R. Murphy, Introduction to AI Robotics, MIT Press, 2000.

[3] R.C. Arkin, "Towards cosmopolitan robots: Intelligent navigation in extended man-made environments," Ph.D. Thesis, COINS Tech, Rpt., 97-80, Univ. of Massachusetts, Dept. of Computer and Information Science, pp. 143-177, 1987.

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

[5] S. D. Touretzky, and L.M. Saksida, "Skinnerbots," Proc. Int. Conf. Simulation of Adaptive Behavior (SAB96), pp. 285 – 294, 1996.

[6] B. Blumberg, "Old Tricks, New Dogs: Ethology and Interactive Creatures," Ph.D. Thesis, The Media Lab, MIT, Cambridge, 1996.

[7] S.Y. Yoon, "Affective Synthetic Characters," Ph.D. Thesis, The Media Lab, MIT, Cambridge, 2000.

[8] J. Pauls, "Pigs and People," Project Report, Division of Information, University of Edinburgh, 2001.

[9] P. Maes, "The dynamics of action selection," Proc. Int. Joint Conf. Artificial Intelligence, Detroit, MI, pp. 991-997, 1989.

[10] A. Saffiotti, K. Konolige, and E. Ruspini, "A multivalued logic approach to integrating planning and control," Artificial Intelligence 76, pp. 481-526, 1995.

[11] A.F.R. Araujo, and A.P.S. Braga, "Reward-Penalty Reinforcement Learning Schema for Planning and Reactive Behavior," Proc. IEEE Int. Conf. System, Man, and Cybernetics, vol. 2, pp. 1485-1490, 1998.

[12] R. Genov, S. Madhavapeddi, and G. Cauwengerghs, "Learning to Navigate from Limited Sensory Input: Experiments with the Khepera Microrobot," Proc. Int. Conf. Neural Networks, vol. 3, pp. 2061-2064, 1999.

[13] C.J.C.H. Watkins, "Learning from delayed rewards," Ph.D. Thesis, Cambridge University, Cambridge, England, 1989.

[14] K. Lorenz, "The comparative method in studying innate behavior patterns," Symposia of the Society for Experimental Biology, 4, pp. 221-268, 1950.

[15] P. Maes, "How to do the right thing," Connection Science, 1, 291-323, 1989.

[16] A. Ludlow, "The evolution and simulation of a decision maker," In: Analysis of Motivational Process, Academic Press, 1980.

[17] R.S. Sutton, and A.G. Barto, Reinforcement Learning, MIT Press, 1988.

[18] L.S. Crawford, "Learning Control of Complex Skills," Ph.D. Thesis, Biophysics, University of California, Berkeley, September 1998.

[19] M. Dorigo, and M. Colombetti, Robot Shaping: An Experiment in Behavior Engineering, MIT Press, 1998.

[20] I.P. Pavlov, Selected works, Foreign Languages Publishing House, Moscow, 1950.

[21] B.F. Skinner, The behavior of organisms: An experimental analysis, Englewood Cliffs, NJ: Prentice Hall, 1938.

[22] AmigoBot User's Guide, 2000.

[23] R.C. Arkin, and J. Diaz, "Line-of-sight constrained exploration for reactive multiagent robotic teams," 7th Int. Workshop on Advanced Motion Control, pp. 455-461, 2002.

[24] M. Likhachev, M. Kaess, and R.C. Arkin, "Learning behavioral parameterization using spatio-temporal case-based reasoning," Proc. Int. Conf. Robotics and Automation, vol. 2, pp. 1282-1289, 2002.

[25] J.B. Lee, M. Likhachev, and R.C. Arkin, "Selection of behavioral parameters: integration of discontinuous switching via case-based reasoning with continuous adaptation via learning momentum," Proc. Int. Conf. Robotics and Automation, vol. 2, pp. 1275-1281, 2002.

[26] M. Likhachev, and R.C. Arkin, "Spatio-temporal case-based reasoning for behavioral selection," Proc. Int. Conf. Robotics and Automation, vol. 2, pp. 1627-1634, 2001.

[27] J.B. Lee, and R.C. Arkin, "Learning momentum: integration and experimentation," Proc. Int. Conf. Robotics and Automation, vol. 2, pp. 1975-1980, 2001.