2005 IEEE/RSJ International Conference on Intelligent Robots and Systems

Learning of Action Patterns and Reactive Behavior Plans via a Novel Two-Layered Ethology-Based Action Selection Mechanism

Il Hong Suh, Sanghoon Lee
Graduate School of Information and Communications
Hanyang University, Seoul, Korea
ihsuh@hanyang.ac.kr, shlee@incorl.hanyang.ac.kr

Woo Young Kwon, and Young-Jo. Cho
Electronics and Telecommunications Research Institute
Daejun, Korea
{wykwon,youngjo}@etri.re.kr

***Abstract*— The two most important abilities for a robot to survive in a given environment are selecting and learning the most appropriate actions in a given situation. Historically, they have also been the biggest problems in robotics. To help solve this problem, we propose a two-layered action selection mechanism (ASM) which designates an action pattern layer and a reactive behavior plan layer. In the reactive behavior plan layer, a task is selected by comparing behavior motivation values that, in an animal, correspond to external stimuli as well as internal states due to hormones. After a task is selected, its corresponding reactive behavior plan is executed as a set of sequential dynamic behavior motivations (DBMs), each of which is associated with an action pattern. In the action pattern layer, each action pattern can be functionally decomposed into primitive motor actions. Shortest Path-based Q-Learning (SPQL) is incorporated into both the reactive behavior plan and action pattern layers. In the reactive behavior plan layer, relationships between perceptions and action patterns are learned to satisfy a given motivation, as well as the relative priorities of these relationships. In the action pattern layer, the relations between sensory states and primitive motor actions can be learned. To establish the validity of our proposed ASM, experiments with our real designed robot will be illustrated together with simulations.**

***Index Terms*— Action Selection Mechanism, Reinforcement Learning, Ethology**

I. INTRODUCTION

An autonomous robot has to select the most appropriate action based on both its internal state and its external environment. This is why robots require sensors to understand a given situation as well as effectors to interact with its environment to accomplish its given task or mission. Moreover, the robot must also include an action selection mechanism or behavior coordination mechanism to decide on the proper sequence of actions [1], [2]. Brooks suggested a new architecture called 'Behavior-based Architecture', which is composed of independently-operating competence modules that have their own sensors and effectors [3]. When two or more competence modules work simultaneously, modules with high priority inhibit modules with low priority. The priority of these competence modules are fixed and predesigned in the subsumption architecture. This makes it difficult to design robots to complete missions in complex environment. To make early stage of behavior-based AI architecture more useful for a robot in a real environment, several researchers have proposed ethologically-inspired Action Selection Mechanisms(ASMs) for action selection [1] [4] [5]. Those models have showed better results in that they better imitate real-life behavior. However, they were mostly performed by simulated robots in simulated environments, not by real robots in real environments. Many ethology-based models are primarily focused on the cause or motivation for a particular behavior, because the actual action of an animal is much more complex to analyze and imitate. Thus, in those models, the abstraction level of primitive behavior is artificially high. However, designing primitive behaviors in real robots with high abstraction levels is extremely difficult.

On the other hand, in both behavior-based AI approach and ethology-based approach, planning sequential behaviors is quite difficult. In the behavior-based approach, it is extremely difficult to see how these plans could be expressed. In other words, in the behavior-based approach, it is very hard to express plans as we know them. In the ethology-based model studies, planning sequential behaviors was not even considered. Bryson considered planning sequences of behaviors to be "reactive plans," a formalized expression which is often used in behavior-based appoaches [6]. The reactive plan is a more complicated plan structure used for circumstances in which the exact ordering of steps cannot be predetermined, and consists of three elements: priorities, preconditions and actions. Most of the approaches presented above were tested in simple environments, where the world was assumed to be completely understood. However, it is hard to give a completely understanding of real environments to a robot. Moreover, if the environment changes, formerly "complete" knowledge of the environment may become completely invalid. Therefore, a robot must have the ability to adapt to its environment; to do this, a computational architecture is required to process and store information about the environment. Reinforcement learning is a learning technique used widely in autonomous robots to select which action is the most appropriate in a particular situation [7] [8]. However, such reinforcement learning techniques may present some difficulties when they are applied to reality. For instance, rewards may not be given immediately after a behavior has been completed. Delayed rewards makes learning very time-consuming, even to the point of impracticality. To

apply reinforcement learning into robot's action selection, this slow convergence problem must first be solved.

Recently, several researchers have suggested ethologically-inspired models of action selection that incorporate learning [9]–[12]. But much work remains to be done to cope with several shortcomings such as the models's inability of generate sequential behavior.

In our previous work, we suggested a single-layered action selection mechanism(ASM) where primitive behavior was stored in fixed action pattern(FAP) that could not be changed by learning [13]. When FAPs are used to regulate primitive behaviors in a complex environment, the number of behavior sequences required grows to be too large. To avoid this shortcoming, we propose a two-layered action selection mechanism that designates an action pattern layer and a reactive behavior plan layer. In addition, Shortest Path-based Q-Learning(SPQL) was incorporated into both reactive behavior plan layer and action pattern layer. In the reactive behavior plan layer, relationships between perception and action patterns are learned to satisfy a given motivation, as well as the relative priorities of these relationships. In action pattern layer, relations between sensor and motor action can be learned.

II. Action Selection Mechanism

One of primitive behavior design concepts in our previous approach [13] was the fixed-action pattern(FAP), inspired by ethology. The term of 'fixed-action patterns' is used by ethologists to describe stereotyped, species-specific complex motor actions that are triggered by genetically preprogrammed releasing stimuli. In this case, 'fixed' implies that the sequences of motor actions is not changed, regardless of the locally changing situation. In a complex environment, satisfying a particular behavioral motivation would require too many sequential pre-defined actions. To avoid this problem, we will consider each behavior as an adaptive action pattern. Here 'adaptive' implies that motor actions could be adaptively modulated according to a locally changing situation. In order to effectively apply this idea, we propose a two-layered ASM consisting of a reactive behavior plan layer and an action pattern layer.

In our ASM, a Perception Filter(PF) filters the particular stimuli that can release action patterns. Following filtration, Behavior Motivation(BM) is employed as it was in [14] [15]. Here, BM is divided into Statical Behavior Motivation(SBM) and Dynamic Behavior Motivation(DBM). SBM is a valued competence connective used to link a task with its situational context, affected by internal drive and external situation. DBM is another valued connective used to link a particular stimulus with a particular action pattern.

In a reactive behavior plan layer, task coordination can be accomplished with SBM as in Tinbergen approach [15]. Specifically, a task is selected by comparing SBM values determined by both external stimuli and internal drive. Once an SBM is selected, a DBM is activated until the SBM is archieved. A DBM is a primitive node used to evaluate the necessity of an association between a PF and an action pattern. A sequence of DBMs implies a reactive behavior plan. Here, 'reactive' means selection of a DBM is not fixed, but adaptively modified to its changing sensory state. Specifically, according to the DBM values, the highest-valued DBM is selected for a particular releasing stimulus, and then the action pattern associated with the DBM is fired into the action pattern layer's action pattern repository. The SBM value is computed by

$$V_{SBM_{i(t+1)}} = \Sigma V_{IV_{jt}} + \Sigma V_{PF_{kt}} - \Sigma_{allSBM}(V_{SBM_{lt}} I_{il}) + effect^{i}_{DBM}, \quad (1)$$

where
i : index of where the i th SBM,
j : index of related IS,
k : index of related PF,
I_{il}: inhibitory gain that SBM_i applies against SBM_l,
l : index of same level SBM(inhibition),
$effect^{i}_{DBM}$: feedback value from DBM under the i th SBM.

Here, $effect^{i}_{DBM}$ is given as

$$effect^{i}_{DBM} = w_i V_{DBM_m},$$

where
m: index of maximum valued DBM under i th SBM,
w_i: weight of feedback value from DBM under the i th SBM.

The DBM value is computed by

$$V_{DBM_i} = V_{DBM_parent} + V_{PF_i} - \epsilon, \quad (2)$$

where
i: index of this node
j: index of related PF
ϵ: constant value; e.g. 0.001

Selecting DBMs for a particular stimulus generates a sequence of behaviors, similar to reactive plans [6]. The main idea behind a reactive plan is to dynamically generate a sequence of behaviors according to changes in the environment. The three elements of the reactive plan are priority, the releaser, and the action. The major difference between our ASM and Bryson's reactive plan in generating sequential behavior is the method used to coordinate behaviors. In our ASM, a dynamic priority value is used to coordinate DBMs. However, in reactive plans, fixed-priorities with Boolean values(TRUE or FALSE) are used. A robot can show more intelligent behavior in a multi-task environment when dynamic priority-based coordination is used.

In summary, our ASM shows some distinct characteristics when compared with previous approaches in [1], [3], [9]. Firstly, our ASM is supported by dynamic priority-based action selection. This allows the robot to adapt easily to its changing environment and its own changing desires. Secondly, in our ASM, the action pattern, a sequence of primitive behaviors, can be adaptively turned by a reinforcement learning technique. This eliminates the need for robot designers to program specific code on a case-by-case basis to avoid a lengthy sequence of behaviors. It also

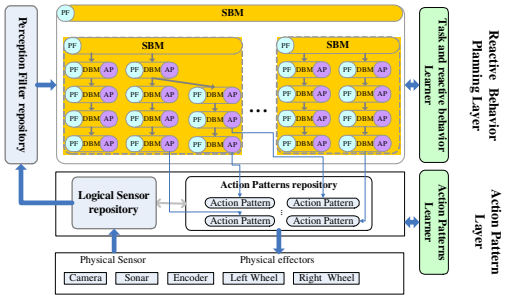


Fig. 1. Proposed robot control architecture

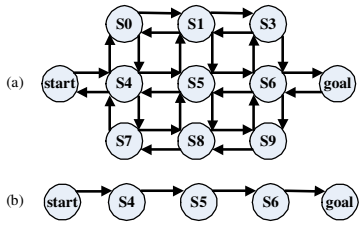


Fig. 2. An Example of shortest path finding

Initialize $Q(s,a)$ and LTM(s,a,s')
Do forever;
choose action a and state s with respect to Q value (ε-greedy or softmax)
execute action a
observe resultant state s' and reward r
$Q(s,a) \leftarrow Q(s,a) + \alpha\left[\gamma \max_{a'} Q(s',a') - Q(s,a)\right]$
update LTM(s,a,s')
IF reward r exceed *threshold*
goal state $g = s'$
find_ShortestPath$(g) \leftarrow$ shortest path from all state to g
repeat for all s,a
IF s,a are in shortest path
$Q(s,a) \leftarrow (1+\eta)Q(s,a)$

Fig. 3. Pseudo-code of SPQL

allows us to define a small number of DBMs for completion of an SBM, which in turn allows us to more easily compute the cost of an SBM's(in other words, a task's) completion. Because of this properly, SBM selection in Eq.(1) can be computed more effectively.

III. Learning

A. Shortest Path-based Reinforcement Learning(SPRL)

In this section, a Shortest Path-based Reinforcement Learning(SPRL) algorithm first developed by in [16] is briefly reviewed. A robot or software agent must learn associations between stimulus and response in a relatively small number of trials. Exploration for learning usually has a high cost, and may endanger the system. On the other hand, internal reasoning may cause computational or algorithmic complexity, but its cost is relatively low when compared with cost of exploration. The learning speed performance can thus be imporved on by fusing the two methods.

In our previous work, we proposed a Monte-Carlo-based SPRL method as an effective learning technique, where internal reasoning is incorporated to increase probability of selecting the most appropriate behavior [16]. In the present research, Shortest Path-based Q-learning(SPQL)is employed instead of the Monte-Carlo technique is shown in Fig.3.

In Fig.3, Long Term Memory(LTM) consists of s, a, s' and v; where s is a current state, a is a current action, s' is a next state that is obtained by action a in a state of s, and v is a reliability value. LTM is updated whenever a learning episode is completed. Note that the LTM may include several paths from an initial state to a goal state in which a reward will be obtained. When all elements of the LTM are represented on a direct acyclic graph, we can find the shortest path among all such possible paths. It is expected that learning speed can be increased if the Q-values of State-Action (S-A) pairs lying on the shortest-path are increased. These increased Q-values drastically reduce the number episodes required for learning. To be more specific, consider an acyclic direct graph or S-A pairs as shown Fig.2. Fig.2(b) shows the shortest path for the graph in Fig.2(a). Here, Dijkstra's algorithm [17] is employed to find the shortest path. Once shortest path is found, important S-A pairs lying on the most optimal path to the goal will have their Q-values raised.

However, it should be noted that to increase the probability of finding the most optimum behaviors, searching space should not be strongly constrained. Thus, the amount that the Q-values should be increased for the S-A pairs on the shortest path must be carefully determined. Fig.3 shows the pseudo-code of our proposed SPQL process.

Our proposed SPQL will be also utilized to learn action pattern as well as reactive behavior plans.

B. Learning of Reactive Behavior Plan by SPQL

When the current ASM fails to accomplish a task, reactive behavior plan learning will be performed. Failures in reactive behavior plan layer can be classified into following three cases;

Case (i): The selected SBM does not contain any DBMs. This implies that there is no knowledge on the way how to accomplish the SBM.

Case (ii): The selected SBM does not include a DBM that can be matched to the current situation. This implies that there is no knowledge of how to cope with given situation.

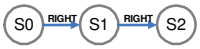
Case (iii): A DBM is selected several times without accomplishing the goal. This implies that the knowledge used to cope with the given situation is incorrect.

In reactive behavior plan layer of our proposed ASM, when a failure is reported, exploration of new state-space and updating of LTM values is initiated. The process of exploration and updating of LTM values is given in the following:

1) A Perception Filter(PF) is randomly selected among all PFs that have a value above the threshold.
2) An LTM element relevant to the selected PF is chosen by considering the extracted reliability values from

State	Action	Reliability
S0	LEFT	0.5
S0	RIGHT	1.0
S1	LEFT	0.5
S1	RIGHT	1.0
S2	LEFT	0.5
S2	RIGHT	1.0

(a)



(b)

Fig. 4. An inner structure of action patten

selected LTM elements containing one action pattern.

3) After a PF and action pattern is selected, the said PF and action pattern is performed for a fixed period of time.
4) Reliability values in the LTM are updated by the SPQL method. A PF of the past is mapped into s, an action pattern of the past is mapped into a, and the current PF is mapped into s'.
5) The Process is repeated from 1 to 4 by time-out or until task is completed.

As the updating of LTM values is iterated, some LTM elements may have higher reliability values. Among those LTM elements, LTM elements with reliability values exceeding a certain threshold is added to the DBM linked with the PF corresponding to s', or the location just below the SBM reporting the failure.

C. Learning of Action Patterns by SPQL

In our proposed ASM, an action pattern is a sequence of primitive motor actions that can be changed by learning. The inner structure of an action pattern shown in Fig.4 consists of tuples; sensory states, motor actions, and reliability values. A SPQL-based learning module is also included in the action pattern layer to aid learning speed.

Processes in the action pattern layer can be divided into two modes: 'run mode' and 'learning mode'. Processes of 'run mode' are given as follows:

1) A sensory state is received from a logical sensor.
2) If sensory state in action pattern tuples is the same as the sensory state just received by the logical sensor, the action pattern tuples are selected.
3) If number of selected action pattern tuples is one, a motor action of the action pattern tuples is performed.
4) If number of selected action pattern tuples is given as two or more, the motor action with higher reliability is activated.

On the other hand, the process of the 'learning mode' is as follows:

1) -3) Process of learning mode is the same as 'run mode'
2) 4) If number of selected action pattern tuples is given as two or more, a motor action is selected by an exploration strategy (e.g. epsilon greedy, or softmax) by comparing the reliability values of those action pattern tuples.
3) 5) If there is a reward signal, the reliability value is updated by applying SPQL method.

Fig. 5. Photos of our designed mobile robot

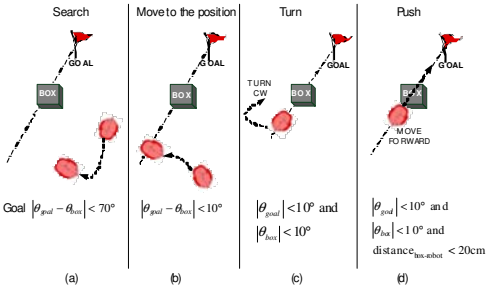


Fig. 6. A task description for box-pushing task

IV. Experimental Results

A. Our Mobile Robot System

Our own mobile robot was designed and employed for our experiments as shown in Fig.5. The robot had been designed to have a single CCD camera with pan/tilt guide, and to be controlled by a two-wheeled differential drive system and an embedded PC(VIA C3).

B. Experimental Objectives

To show the validity of our proposed ASM integrating reinforcement learning, the experimental set-up for a pushing-box-into-a-goal(PBIG) task was organized as shown in Fig.6.

In this experiment, a mobile robot was shown to learn each action pattern needed to push a box into a goal via our SPQL. To do this, the robot was rewarded when it completed the PBIG task successfully.

C. Implementation Notes

1) Object Extraction from a CCD Camera Image: In order for an image of an object to be recognizable, it needs to be separated from its background. HSV representation is employed to minimize the effects of variation in light intensity in the color images's processing. After separating the image of the object from the background image, the center of gravity is computed. Then, the center of gravity is mapped to a relative object angle and the area is mapped to a relative object direction.

2) Bayesian Filter-based Design of Logical Sensors: For a PBIG task, a robot must recognize the locations of objects such as the box and the goal. It should be recalled that our robot has only a single camera with which to find objects. Due to a restricted field of view, our robot could not recognize the locations of all objects around itself

repeat
 init t=0
 scanning image and getting position of objects(s_0)
 set $Belief_0 = initialSensorError$
 repeat unil $Belief_t = threshold$
  execute a_t
  $Bel_{t+1} = P(s_{t+1} \mid a_t, s_t) Bel_t$
  $s_{t+1} = \arg\max_{all\ i} P\left(s_{t+1} \mid a_t, s_t\right)$
  $t \leftarrow t+1$
$P(s_{t+1} \mid a_t, s_t)$

Fig. 7. A pseudo code of Bayesian filter-based logical sensor

Name	forward movement (mm)	Heading (degree)
TURN_L	0	-5
TURN_R	0	5
MOVE_FL	20	-5
MOVE_FFL	40	-5
MOVE_F	40	0
MOVE_FFR	40	5
MOVE_FR	20	5
BACK	-40	0
BACK_L	-20	-5
BACK_R	-20	5

(a) primitive actions

$state := \langle \alpha, \beta, \gamma, \delta \rangle$

$\alpha = 0$ when $\left|\theta_{box} - \theta_{goal}\right| < threshold$

$\alpha = 1$ when a box is left of a goal

$\alpha = 2$ when a box is right of a goal

$\beta = 0$ when $\left|\theta_{box} - \theta_{goal}\right| < 10°$

$\beta = 1$ when $\left|\theta_{box} - \theta_{goal}\right| < 70°$

$\beta = 2$ when $\left|\theta_{box} - \theta_{goal}\right| < 120°$

γ: discrete angle of a box(total 11 state)

δ : discrete angle of a goal(total 11 state)

(b) states

Fig. 8. primitive actions and states

at any single time. Therefore, our robot needed to have a logical sensor to process perceptual state information, especially to estimate locations for out-of-sight-objects. A grid-based Bayesian Filter was used to estimate the locations of multiple objects. The basic idea behind a Bayesian Filter is to estimate the posterior probability density over the state-space condition on the data [18]. Details on the design of our Bayesian filter are shown in Fig.7

3) State Organization and primitive actions for action patterns: Action patterns are sequences of motor actions used to satisfy a specified motivation. Relative positions between the robot and objects such as the box and the goal are used as perceptual states for the action pattern. State organization and primitive actions for action patterns for a box-pushing task are shown in Fig.8.

D. The Learning of Action Patterns

The robot was shown to learn a sequence of motor actions for implementing action patterns. To learn an action pattern, 10 trials at two different initial positions were performed as shown in Fig.9. Because the robot did not have any initial knowledge of its surroundings, it often escaped from 2m $\times$ 2m experimental playground. Such an escape from playground was regarded as a failure, and failed trials were excluded from learning action patterns. Because a human observer cannot perceive state in exactly the same way as a robot does, a reward signal for a successful action pattern was self-generated by its own reward state. Four action patterns were designed and learned in this experiment in order to accomplish the PBIG task. Of the four, three action patterns, excluding the 'search' action pattern, were learned through a sequence of successfully performed PBIG task. Specifically, when a successful action sequence is obtained for a 'move-to-the-position' action pattern, this action sequence will get a reward signal, and is considered an action pattern. Then, perception filter value for a 'turn' action pattern is increased, which makes subsequent action sequence learning the 'turn' action pattern. Each action pattern can be learned in one trial in the same manner. DBM trees were pre-designed to allow the robot learn all action patterns in one trial. In other words, perception filters and action patterns were tightly coupled. To acquire a complete knowledge of the environment, many experiments are required. However, two initial positions were selected as shown in Fig.9. The learning performance of action patterns using SPQL is compared with that of Q-learning for the same action patterns in Fig.10.

TABLE I
NUMBER OF STEPS IN ACTION PATTERNS

	hand-coded	**SPQL**	**Q-learning**
move to	24	32	31
turn	20	20	24
push	11	17	18
total	55	65	70

The number of primitive behaviors($\cong$ step) for each action pattern after 10 trials of learning are shown in Table I for three methods: case-by-case hand-coding, SPQL ,and Q-learning. It can be seen that the hand-coded method requires 55 steps, and SPQL and Q-learning require 65 steps and 70 steps, respectively. From this experiment, SPQL methods perform better than Q-learning with respect to learning speed.

A clear difference between hand-coded and learning methods is observed mostly in the learning of 'move-to' action pattern. In action patterns obtained by learning experiments, the combination of two primitive actions 'moveF' and 'moveFFL'(defined as in Fig.8(a)) was observed often as shown in Fig.11. It should be noted that in the optimal action pattern, 'moveF' will be observed as a lack in number of trials and to a limited search space in the Q-learning and SPQL methods, respectively. In the case of fast-RL, it is restriction of search space that is the limiting factor. As the number of trials is increased, the performance obtained by hand-coded and learning methods should become very similar.

E. Learning of Reactive Behavior Plans

After learning of action patterns is completed, another learning process can be activated to learn about associations between perception filters and action patterns, and sequences of these associations; known as learning of DBMs.

Box-pushing tasks and batter-charging tasks were performed in a simulation. The two corresponding SBMs were innately defined; however, the DBMs necessary to complete these tasks were not attached to the two SBMs.

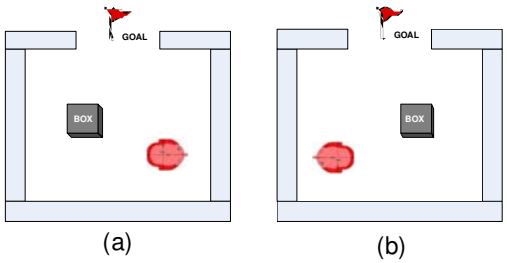


Fig. 9. Two relative locations of a robot with respect to the box

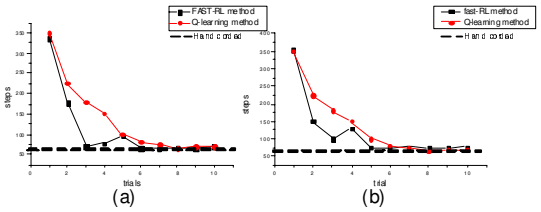


Fig. 10. Number of step in action patterns

Thus, DBMs needed to be learned to satisfy the two pre-defined SBMs. Fig.12 shows the simulation environment for the box-pushing and battery-charging tasks. The Internal State(IS) value for box-pushing was given as a constant value, while the IS value for batter-charging increased as battery voltage became low. Thus, when the battery voltage became low, the value of the SBM affected by the IS value for battery-charging became high and thus selected the SBM for batter charging. When the batter voltage was high, however, the value of the batter-charging SBM became low, thus selecting the SBM for the box-pushing task. After one of two SBMs was selected, learning must then be performed because there were no DBMs to satisy the SBM. Here, the necessary action patterns were assumed to be available. SBM and DBM structures before and after learning are shown in Fig.13(a) and (b), respectively. One learning trial is completed when both tasks are satisfied. SPQL was applied here. The number of action patterns is shown for each learning trial in Table II. II Shows that 45 action patterns were required for the first learning trial, but only 7 action patterns were necessary for the 4th and 5th learning trials.

Owing to our proposed SPQL method, learning of sequential behaviors was completed within a relatively small number of learning trials. Note that since pre-learned necessary action patterns were assumed to be available, SBMs(in other words, tasks) could be satisfied by learning only a relatively small number of relations between perception filter(PF) and action patterns(AP) that would be considered as more abstracted rules.

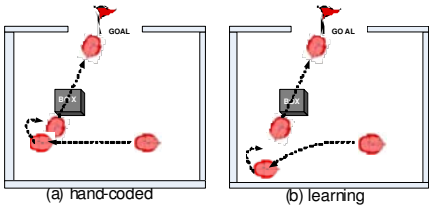


Fig. 11. Trajectory of a robot

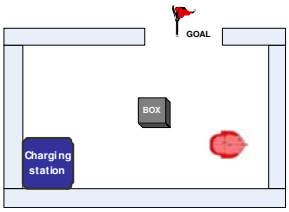


Fig. 12. Experimental setup for multi-task coordination

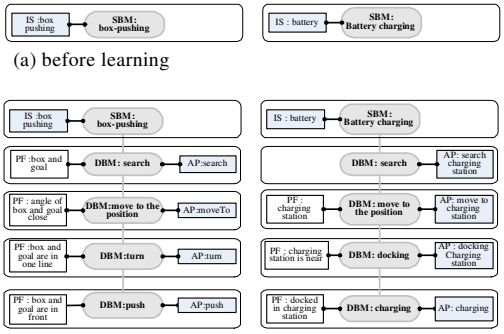


Fig. 13. Structures of SBM and DBM before and after learning

V. Conclusion

In this paper, we proposed a two-layered action selection mechanism including an action pattern layer and a reactive behavior plan layer. Shortest Path-based Q-Learning(SPQL) was incorporated into both layers to learn dynamic behavior motivations(DBMs) to satisfy static behavior motivations(SBMs){i.e. tasks} and learn action patterns to support the DBMs.

The characteristics of our proposed ASM can be summarized as follows:

1) Our ASM was supported by dynamic priority-based action selection. This allowed the robot to effectively select the appropriate behaviors to satisfy its own desires.
2) The action pattern in our ASM, which is a sequence of primitive behaviors, could be adaptively turned by a reinforcement learning technique. This would allow robot designers not to program all specific code on a case-by-case basis.

TABLE II
Number of Steps in Action Patterns

Learning Trials	Trial1	Trial2	Trial3	Trial4	Trial5
No. of action patterns	45	12	8	7	7

Experiments in learning action patterns of primitive behaviors in order to accomplish the box-pushing task were performed by our SPQL method. Moreover, simulation of the reactive behavior plan was also performed, where sequences of DBMs were satisfactorily learned.

ACKNOWLEDGMENT

The work reported in this paper was supported in part by "Embedded Component Technology and Standardization for URC" project of Ministry of Information and Communication.

REFERENCES

[1] P. Maes, "A Bottom-Up Mechanism for Behavior Selection in an Artificial Creature," In From Animals to Animats, First International Conference on Simulation of Adaptive Behaviour, MIT Press, Cambridge, 1991.

[2] P. Pirjanian, "An Overview of System Architectures for Action Selection in Mobile Robotics," Tech-report, Laboratory of Image Analysis, Aalborg University 1997.

[3] R. Brooks, "A Robust Layered Control System For a Mobile Robot," IEEE Journal of Robotics and Automation, pp.14-23, April, 1986.

[4] J. K. Rosenblatt and D. W. Payton, "A Fine-Grained Alternative to the Subsumption Architecture for Mobile Robot Control," Proceedings of the IEEE/INNS International Joint Conference on Neural Networks, Vol.II, pp.317-323, 1989.

[5] E. Spier and D. McFarland, "A Finer-Grained Motivational Model of Behaviour Sequencing," In From Animals to Animats 4 : Proceedings of SAB96, 1996.

[6] Joanna J. Bryson, Intelligent by design,PhD thesis, MIT, 2001. 83.

[7] I.H. Suh, S.H. Lee, B.O. Kim, B.J. Yi, S.R. Oh, "Design and Implementation of a Behavior-Based Control and Learning Architecture for Mobile Robots" Proc. of IEEE Conf. on ICRA, 2003.9.

[8] Dorigo M. , M. Colombetti, "Robot Shaping: Developing Autonomous Agents through Learning", Artificial Intelligence, 71, 2, 321-370, 1994.

[9] B.M. Blumberg, "No Bad Dogs : Ethological Lessons for Learing in Hamsterdam and Interactive Createures," MIT ,1997.

[10] B. M. Blumberg, "Old Tricks, New Dogs," Ethology and Interactive Createures, MIT ,1997.

[11] C. M. Witkowski, "Schemes for Learning and Behaviour: A New Expectancy Model," Ph.D. Thesis, University of London, 1997.

[12] M. Humphrys, "Action Selection methods using Reinforcement Learning.," Ph.D. Thesis, University of Cambridge, England, 1996.

[13] Il Hong Suh, Woo Young Kwon, and Sanghoon Lee, "A Novel Action Selection Mechanism for Intelligent Service Robots," International Conference on Control, Automation and Systems 2003, pp.22-25, October, 2003.

[14] K. Lorenz, Foundations of Ethology, Springer verlag, 1981.

[15] N. Tinbergen, The Study of Instinct. Clarendon Press,1951.

[16] W. Y. Kwon, S. H. Lee, and I. H. Suh, "A Reinforcement Learning Approach Involving a Shortest Path Finding Algorithm," Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems, pp.27-31, 2003.

[17] R.E. Neapolitan, Foundation of algorithms : using C++ pseudocode, Jones and Bartlett Publishers, 1998

[18] A. Doucet, N. de Freitas, and N. Gordon,eds., Sequential Monte Carlo in Practice,Springer-Verlag, 2001.