International Journal of
Soft Computing And Software Engineering (JSCSE)
e-ISSN: 2251-7545
Vol.3,No.2, 2013
Published online: Feb 25, 2013

DOI: 10.7321/jscse.v3.n2.1

Use of the Knowledge of Perceptual State Transition in Reinforcement Learning

[1]Kentarou Kurashige, [2]Yoshiki Miyazaki
[1,] *Muroran Institute of Technology, Muroran, Japan*
[*2,] *NEC Software Hokkaido, Ltd., Sapporo, Japan*
Email: [1]kentarou@csse.muroran-it.ac.jp, [2]mydks.public@gmail.com

Abstract. **Reinforcement learning(RL) is one of the machine learning and is often used for actual robot. A reward which indicates a task for robot is most important information on RL. But how to get a reward will change by change of task or environment. So learning performance of RL will be worse for multiple tasks or dynamic environment. To cope with multiple task and dynamic environment, we focus on a knowledge which is independent on reward and propose a learning system based on reinforcement learning and the knowledge. In this paper, we perform two kinds of experiments with simulation. One is for multiple tasks under a static environment and another is for a task under a dynamic environment. We will show the validity of proposed system by these simulations.**

Keywords: *Reinforcement Learning, Environmental Knowledge, Multi-tasks, Dynamic environment*

* Corresponding Author:
Kentarou Kurashige,
Department of Computer Science & Systems Engineering, Muroran Institute of Technology,
Muroran, Japan,
Email: kentarou@csse.muroran-it.ac.jp Tel:+81-143-46-5400(ext.5489)

1. Introduction

Now, robot plays an active part in many fields[1]. And various researches about robot are studied[2-8]. Real environment in which we want to throw a robot is very complicated and dynamic so it is more difficult to design proper actions of robot for such environment. Instead of action design, a method to adapt to environment automatically become to be desirable recently and various studies about machine learning have been studied to realize such desire[9]. There are some general techniques about machine learning[10-13]. Reinforcement Learning(RL) is one of them and attracts attention as technique which is often used in actual machines[14,15]. In this study, we pay attention to RL.

RL uses reward which expresses a task on an environment for a robot and stores rewards as knowledge to achieve given task on an environment which robot faces. This learning based on reward has an advantage that can treat unknown environment without prior knowledge[16,17].

But there is the problem that learning becomes to be unstable when a robot is given multi-tasks or faces dynamic environment. The reason is that a reward function which indicates a task on an environment changes along to each task and each situation. This means that a value of reward becomes different with the same robot action. This prevents RL from learning the meaning of action and make learning efficiency worse.

To overcome this problem, there are studies that a robot has some learning space to cope with various tasks[18-20]. In this method, a robot cope with various tasks by

International Journal of
Soft Computing And Software Engineering (JSCSE)
e-ISSN: 2251-7545
Vol.3,No.2, 2013
Published online: Feb 25, 2013
DOI: 10.7321/jscse.v3.n2.1

preparing learning space for every task. So each learning task has good learning efficiency as good as normal reinforcement learning for single task and it is possible to learn various tasks. But in these studies, the number of learning spaces is needed as same as the number of tasks and must be prepared manually.

Our objective is a learning system which can cope with multi-tasks and adapt to dynamic environment without preparing learning space for each task. To realize it, we focus on learning of environmental model and use it with task learning. We consider learning of environmental model as learning which is independent on task, so it is dependent on a reward function. We put it in normal RL and make RL to simulate environment and to guess the sequence of robot action toward the goal of new task when the way to get reward changed, which indicates the change of task. By acquiring the candidate of action sequence, we make task learning on RL better and faster.

In this paper, we pay attention to state transition[21] of environment and we propose Perceptual State Transition(PST) as state transition composed by robot sensor. We propse the method that a robot acquires PST with its experience as the learning of environmental model. And we put this method in normal RL and construct total learning system, we show the system can adapt to the change of task or dynamic environment. We experiment multi-goals maze task and single-goal maze task on dynamic environment and show the validity of proposed method.

2. Problem on reinforcement learning

2.1. Framework of reinforcement learning

The reinforcement learning is a learning technique to adapt environment by trial and error. The main elements constituting the reinforcement learning is the following elements. And we show a conceptual diagram of the reinforcement learning consists of these elements in fig.1

- Learner
- Environment
- Reward function
- Value function
- Learning method
- Action selection method

The learner takes an action by action selection method. The learner can get a reward and a state that the learner faces changes to other state. Then the learner learns based on a reward in learning method. Usually in reinforcement learning, the learner evaluates a pair of a state and an action. The learner uses a reward to evaluate this pair for given task. Reward is a scalar value and expresses how good a pair of a state and an action is. The learner learns to get more reward. Therefore, it is necessary to set a reward correctly so that the learner achieves a task. Information evaluated by reward is saved as knowledge in value function. The learner chooses an action based on value function in action selection method and takes action. The learner can achieve a task by repeating this cycle.

The learner has value function. And value function expresses a result of learning based on reward. And the learner has learning method and action selection method. In learning method, the learner updates a value function by reward. In action selection method, the learner decides an action by value function. There are some techniques each part. For example, Q-learning is famous

International Journal of
Soft Computing And Software Engineering (JSCSE)
e-ISSN: 2251-7545
Vol.3,No.2, 2013
Published online: Feb 25, 2013
DOI: 10.7321/jscse.v3.n2.1

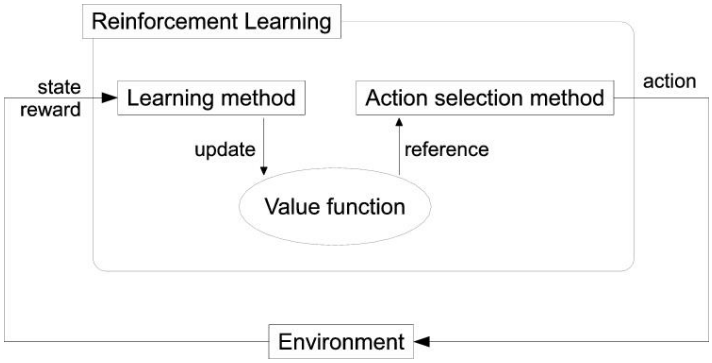


Figure 1. The conception diagram of RL

technique in learning method, and greedy method is famous technique in action selection method.

2.2. Problem caused by reward

The problem we paid attention to is that it takes much time to learn when a reward function which shows how to get reward changes. Change of a reward function is caused by change of given task and change of environment. In reinforcement learning, a result of learning is directly affected by a reward function. Therefore the learner ignores some information that is not shown by a reward function. Because the learner takes actions based on a result of last task, the learner takes much time to re-learn for new task when given task changed. In addition, when the task changed into a task that does not resemble the last task, the problem becomes remarkable. In the case that tasks are totally different, because the knowledge that the learner get by learning is completely different, the learner cannot utilize the knowledge for the new task effectively.

When the learner takes an action that unrelated to a task, the learner evaluates the action as low value. And the learner holds this information as knowledge which is low value.

However, it is possible that this knowledge is the useful information, when the task changed.

Especially for the case of environment with a few changes, there is effective information in other task.

So we paid attention to the information that can use various tasks and focus on PST.

3. Definition of perceptual state transition

In this paper, we define PST of environment which a robot perceive as reward-independent knowledge.We define the unit of PST in eq. (1).

$$k_m := (s_i, {}^{\exists}a_h) \rightarrow (s_j) \tag{1}$$

Here, s_* is a state which a robot faces and a_* is an action which a robot takes. k_m means *m*-th PST and consists of a pair of s_i, and a_h and s_j which is the state by taking a_h on s_i.

The unit of PST is gathered and stored at each action a robot takes. We define knowledge table *K* to treat whole PST as follows. We define the knowledge table in eq. (2) when a robot has *n* units of PST.

$$\mathrm{K} = \{k_m | m = 1, \cdots, n\} \tag{2}$$

4. Proposed system based on RL using perceptual state transition

4.1. The outline of the system

In this section, we explain the outline of a proposed system. The system is consists of RL and the proposed method using PST and each part works independently. RL learns and decides a robot's action

International Journal of
Soft Computing And Software Engineering (JSCSE)
e-ISSN: 2251-7545
Vol.3,No.2, 2013
Published online: Feb 25, 2013
DOI: 10.7321/jscse.v3.n2.1

based on value function which is knowledge related with reward. The proposed method gathers and stores knowledge table about PST by trial and error. When given task or environment changes and a structure of reward function changes, the proposed method affects value functions using knowledge table and inclines a robot to take actions for new task or environment. We shows the outline of the proposed system in fig. 2.

4.2. Acquisition of perceptual state transition

A robot selects actions by RL and sense states for RL. So a point of time *t*, the system can treat a state at *t* which a robot faces and a pair of an action and a state at *t*-1. Regarding this state transition by an action as unit of PST (eq. (1)), the system looks up this unit in knowledge table. If there is same one, add this unit to knowledge table.

4.3. Prediction of action for new task

The proposed system makes a robot to cope with new task quickly by prediction of proper action for new task when a robot recognizes a change of a task. In this section, we express the details of this process.

At first, we express how to recognize a change of a task. RL recognizes a task by a way to get reward. Here, a change of a task causes a change of a way to get reward. The system can detect a change of a task by monitering a reward which a robot gets. In this paper, a change of a task indicates a change of a value of a reward when a robot gets at goal.

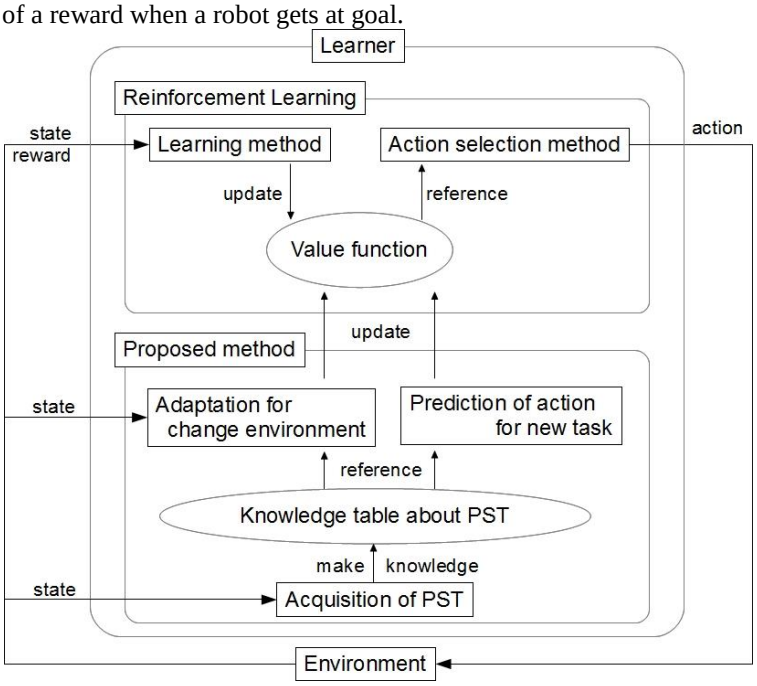


Figure 2. The conception diagram of proposed system

When the system recognizes a change of a task, it searches all routes to new goal state with knowledge table which is the list of acquisition of PST. The system has the information represented by eq. (1), so it can search routes by back propagation starting from new goal. The system stops to search when the state is initial state of a robot or is the pair of a state and an action which was already found as the part of route(we name this pair "cross state"). In the second case, the system choices shorter route and longer route is ended at cross state (fig. 3).

International Journal of
Soft Computing And Software Engineering (JSCSE)
e-ISSN: 2251-7545
Vol.3,No.2, 2013
Published online: Feb 25, 2013
DOI: 10.7321/jscse.v3.n2.1

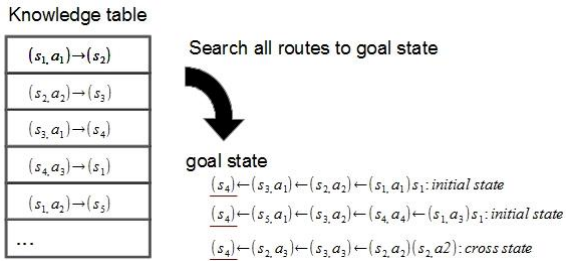


Figure 3. The example of searching all routes to goal state with knowledge table

After searching all routes to new goal state, the system updates value function as a robot inclines to take actions for new goal. The all routes are searched based on a robot's experience, so RL complement uncertainty by trial and error. In this paper, value function is updated by eq. (3) and eq. (4).

$$\mathrm{Q}(s_d, a) \leftarrow \mathrm{Q}(s_d, a) + (f \cdot \gamma)^{d-1} \cdot r \cdot SG(s_d) \tag{3}$$

$$\mathrm{SG}(s_d) = \begin{cases} 1.0 & s_d \ is \ on \ a \ route \\ 0.5 & s_d \ is \ not \ on \ a \ route \end{cases} \tag{4}$$

Here, Q is value function and s_d is the pair of a state and an action which is on a route from initial state to goal. SG is the function to judge a pair of a state and an action is on a route or not. d expresses the number of steps from goal to s_d in knowledge table. γ is discount rate for reward and the same value used in Q-learning. And γ expresses the reward which a robot got as new task. In addition, f is the parameter expressing reliability of PST and takes the range as $0 \leq f \leq 1$. The influence of PST becomes big along with f.In this paper, the system updates by proposed method only when the system recognizes a change of a task.

4.4. Adaptation for change environment

The knowledge table is the information about environment. And this is acquired by a robot's trial and error. By comparing PST with the pair of a state and an action as the result of a robot's action, the system can detects a change of environment and update knowledge table. We show the algorithm for them as follows.

- Get the s_i as the result of a robot action a_h at s_i.
- Search the pair of s_i and a_h and get the unit of PST, $(s_i, a_h) \rightarrow (s_j')$.
  - If there is no PST, $(s_i, a_h) \rightarrow (s_*')$, that means no change of environment and just acquisition of new PST.
- Compare s_j and s_j'.
  - $s_j = s_j'$: Detects no change of environment.
    - Do nothing.
  - $s_j \neq s_j'$: Detects the conflict between environment and PST.
    - Superscribe new PST, $(s_i, a_h) \rightarrow (s_j)$.
    - Initialize $\mathrm{Q}(s_i, a_h)$.

5. Experiment with maze problem

5.1. Outline of experiment

In this paper, we perform two experiments with maze problem by simulation. One is the experiment for multi-task in static environment. In maze problem, a task is arrival at a goal position from a start position. We set multi-task on maze problem by changing a goal position periodically. Anothor is the

International Journal of
Soft Computing And Software Engineering (JSCSE)
e-ISSN: 2251-7545
Vol.3,No.2, 2013
Published online: Feb 25, 2013
DOI: 10.7321/jscse.v3.n2.1

experiment for single-task in dynamic environment. We make dynamic environment by putting or removing obstacles in this paper. As the common setting through experiments, we express the setting of maze, the setting of a robot actions and the setting of RL.

A maze size is different between experiments but a start position is same. We set the x-axis as the horizontal axis and the y-axis as the vertical axis, and set the origin at the upper left. In this case, we set a start position at the origin.

A robot in a maze can move four direction: up, down, right and left. If a robot reaches a goal position, a robot go back to a start position. We define this cycle as one trial.

As the setting of RL, we use Q-learning as learning method and ϵ-greedy method as action selection method. Q-learning updates Q-value expressed in eq. (5).

$$\mathrm{Q}(s_t, a_t) \leftarrow \mathrm{Q}(s_t, a_t) + \alpha\left\{r_{t+1} + \gamma \cdot \max_k Q(s_{t+1}, k) - Q(s_t, a_t)\right\} \quad (5)$$

Here, s_t is a state at time t and a_t is an action a robot took at time t. And $\mathrm{Q}(s_t, a_t)$ expresses Q-value about a_t on s_t. r_{t+1} is a reward which a robot got by taking a_t on s_t. α is learning rate and the range is $(0 \leq \alpha \leq 1)$. γ is discount rate. It is used too in eq. (3).

ϵ-greedy method is the action selection method which chooses a random action in probability of ϵ or choose an greedy action in probability of $1 - \epsilon$. Greedy action is the action which makes the Q-value max value.

We prepare the three robots to compare results. Robot-A does not use PST, so Robot-A learns only with reinforcement learning. Robot-B and robot-C use PST. The parameter f which shows the rate of use of PST equals 0.5 on Robot-B. And the parameter f equals 1.0 on Robot-C.

5.2. Simulation for multi-tasks on static environment

In this simulation, we will show that a robot can cope with multi-tasks by proposed system. To represent multi-tasks, we make goal position changing every constant trial. The size of the maze used in this simulation is 64×64 shown in fig.4. The start position is always the same place; it is (0,0), upper left of the maze. And we show the goal positions in table 1. The number of trials which change g o a l

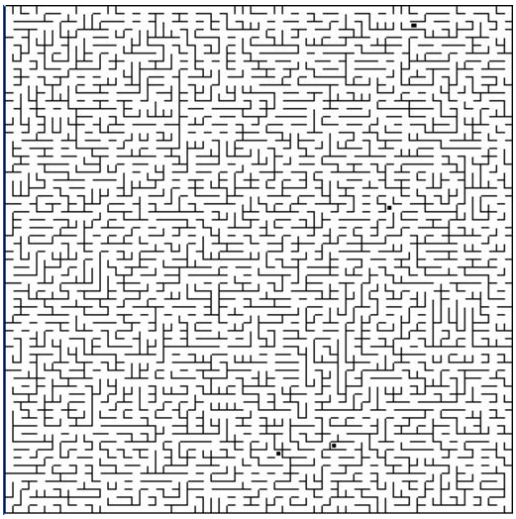


Figure 4. The maze and goals used in the simulation.

Table 1. Goal position

International Journal of
Soft Computing And Software Engineering (JSCSE)
e-ISSN: 2251-7545
Vol.3,No.2, 2013
Published online: Feb 25, 2013
DOI: 10.7321/jscse.v3.n2.1

G1	*G2*	G3	G4
(34,56)	(51,2)	(48,25)	(41,55)

Table 2. Simulation setting for static environment

Reward(only at a goal)	100
Number of trials for each goal	250
Initial value of Q-value	0.001
The size of maze	64 x 64
A	0.5
Γ	0.8
ϵ	0.05

position is 250 trials for each goal in this paper. We set reward only at goal state and make a robot restart from start position after getting reward. We show the parameter in table 2.

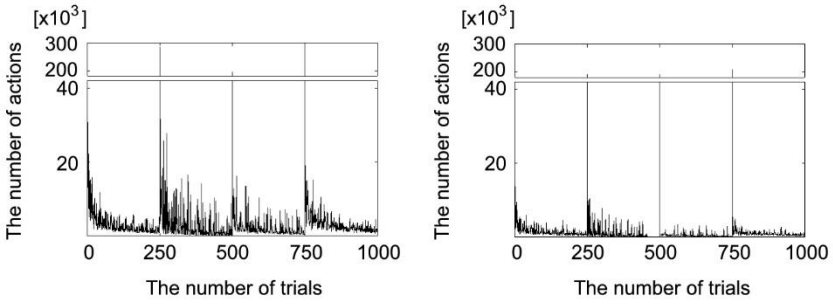
5.3. Result of the simulation for multi-tasks on static environment

We show the transition of the actions for each goal in fig.5. Figure 5(a)-(c) show the number of the actions for reaching the goal every trial. We cut the y-axis from about 40000 to about 200000 to be easy to look. In these graphs, the number of the actions decreases expresses that the robot learns. Here, the number of the actions increases suddenly every 250 trials. This means the goal position changes. Focusing on the trials after changing goal position, fig. 5(b) and fig. 5(c) show less the number of actions than fig. 5(a). This means Robot-B and Robot-C which are proposed method adapt to new goal more quickly than Robot-A which is ordinary method. Figure 5(d) shows the score of each robot for each goal. We use the remainder of the number of actions from the number of optimum action as the score. Here, the score of the actions which Robot-B and Robot-C took for goal 2 and Robot-C took for goal 3 is zero. This means these actions equal optimum actions. The number of the actions through all trials is shown in fig. 6. This graphs shows the number of the actions for Robot-B and Robot-C are less than Robot-A.

By these results, we could show the validity of proposed system for multi-tasks.

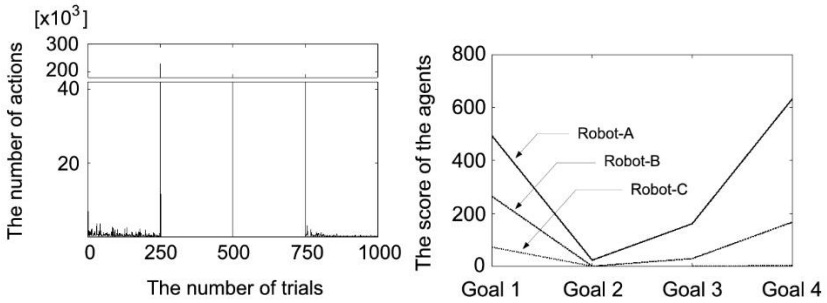
5.4. Simulation for dynamic maze

In this simulation, we show that a robot can cope with change of environment quickly. There are the changes in environment and a goal does not change in this simulation. Here, we put the obstacles in environment and change the position of the obstacles every constant trials. A robot can not go to the



(a) Result of Robot-A (b) Result of Robot-B

International Journal of
Soft Computing And Software Engineering (JSCSE)
e-ISSN: 2251-7545
Vol.3,No.2, 2013
Published online: Feb 25, 2013
DOI: 10.7321/jscse.v3.n2.1



(c) Result of Robot-C (d) The score of the robots

Figure 5. The results of the simulation for multi-tasks

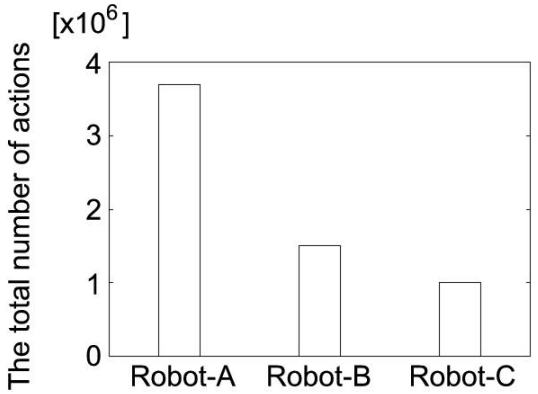


Figure 6. Comparison of all robots

position which exists the obstacle. By the change of the positions of the obstacles, an optimal path to the goal changes. We make dynamic environment by the change of the positions of the obstacles. The

Table 3. Simulation setting for dynamic environment

Reward(only at a goal)	100
Number of trials for the goal	100
Number of trials for change of environment	30
Number of obstacle	51
Initial value of Q-value	0.001
The size of maze	16 x 16
A	0.5
Γ	0.8
ϵ	0.05

International Journal of
Soft Computing And Software Engineering (JSCSE)
e-ISSN: 2251-7545
Vol.3,No.2, 2013
Published online: Feb 25, 2013
DOI: 10.7321/jscse.v3.n2.1

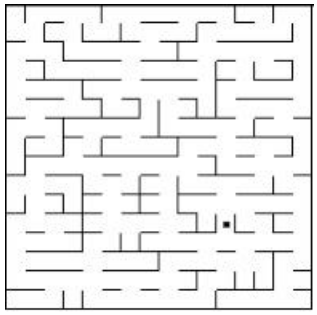


Figure 7. The maze and goal used in the simulation

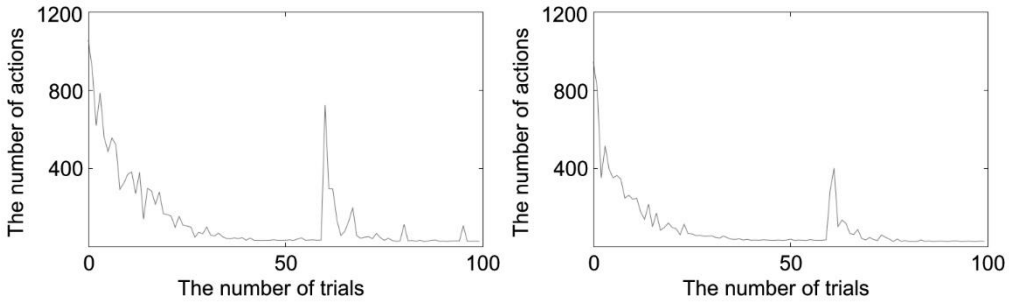
positions of the obstacles will be set at random. When a goal path does not exist by random setting of the positions of the obstacles, we redo it until the path exists.

The size of the maze used in this simulation is 16×16 shown in fig. 7. The start position is (0,0) and the goal position is (11,11). We put 51 obstacles in the maze, which is about 20\% of all states. We show the setting of the parameters in table 3.

5.5. Result of the simulation for dynamic maze

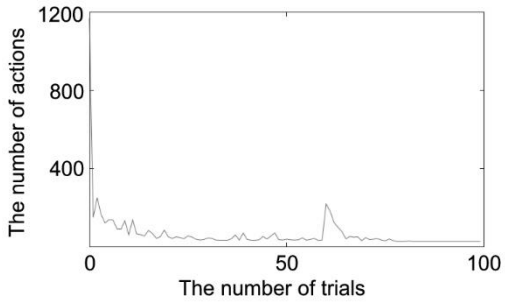
We show the result of the simulation in fig. 8 and fig. 9. Focusing on trials less than 30th trials in fig. 8(a)-(c), robots learn for reaching the goal. Especially, tendency of learning is same as static simulation's result.

Next, we focus on the trials around 30th/60th/90th. In these points, the change of environment occurs. As a results, the route to the goal which robots learned and acuired become unusable and the number of the actions increases for all robots. Here, the influences of the change of environment were comparatively small at 30th and 90th trials. This reason is that a few obstacles appeared on route which robots learned, so robots could cope with this change of environment by little time. However we can see that the number of actions of robots increase and the performances of robots become bad at 60th trial. This reason is that



(a) Result of Robot-A (b) Result of Robot-B

International Journal of
Soft Computing And Software Engineering (JSCSE)
e-ISSN: 2251-7545
Vol.3,No.2, 2013
Published online: Feb 25, 2013
DOI: 10.7321/jscse.v3.n2.1



(c) Result of Robot-C

Figure 8. The results of the simulation on dynamic environment

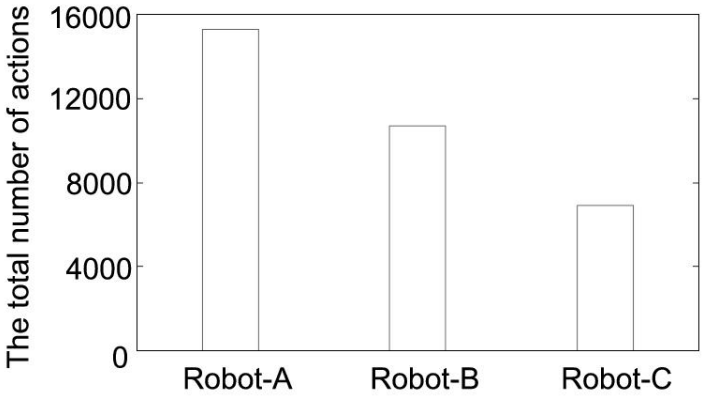


Figure 9. Comparison of all robots

many obstacles appeared on route which robots learned and acquired. In all case which the change of environment occured, Robot B and Robot C which used proposed knowledge could cope with this change quicker than Robot A which used ordinary RL.

In addition to this result, we show the total number of the actions through trials in fig. 9. This shows the number of actions which Robot B and Robot C took is less than the one Robot A took. It indicate robots which used proposed knowledge adapted for the change of environment quicker than robot which used ordinary RL.

6. Conclusion

In this paper, we focus on the information which is independent on reward and proposed perceptual state transition(PST) and knowledge table with PST. Firstly, we defined how to get and store PST on knowledge table. Next, we defined how to use this knowledge as prediction of actions for multi-tasks and adaptation of change of environment. And we proposed the system using this knowledge based on reinforcement learning.

To show the validity of proposed system, we experiment two kinds of simulation. One is about multi-tasks and another is on the change of environment. Both is about maze problem which aim to reach a goal from initial position. So, one has goals which change every constant trials, and another has obstacles which appear or disappear at random every constant trials. We compare proposed system with ordinary reinforcement learning. Proposed system got better results in all experiments and showed quicker and more effective adaptation on these maze problems.

As the future works, we want to use proposed method for actual robot on real environment. We must consider noises in that case. We must improve PST and make it robust to noises. In addition to

International Journal of
Soft Computing And Software Engineering (JSCSE)
e-ISSN: 2251-7545
Vol.3,No.2, 2013
Published online: Feb 25, 2013
DOI: 10.7321/jscse.v3.n2.1

that, we will apply PST to other learning method, especially the method for discrete environment. PST is independent on concrete learning method so we can use it for evolutionary computation or other soft-computing mehtods.

References

[1] M. Hirose and K. Ogawa, "Honda humanoid robots development", Phil. Trans. of THE ROYAL SOCIETY A, Royal Society Publishing, Vol.365, pp.11-19, 2007.

[2] H. Kawamoto and S. Lee and S. Kanbe and Y. Sankai, "Power assist method for HAL-3 using EMG-based feedback controller", IEEE Int. Conf. on Systems, Man and Cybernetics, Vol.2, pp.1648-1653, 2003.

[3] K. Kurashige, "A simple rule how to make a reward for learning with human interaction.", Proc. of the 2007 IEEE Int. Symp. on Computational Intelligence in Robotics and Automation, pp.202-205, 2007.

[4] K. Kurashige and Y. Onoue, "The robot learning by using "sense of pain"", Proc. of Int. Symp. on Humanized Systems 2007, pp.1-4,2007.

[5] Y. Kishima and K. Kurashige, "Growth of individual intelligence using communication", Joint 4th Int. Conf. on Soft Computing and Intelligent Systems and 9th Int. Symp. on advanced Intelligent Systems, pp.287-292, 2008.

[6] J. P. Desai and J. Ostrowski and V. Kumar, "Controlling formations of multiple mobile robots", Proc. Of IEEE Int. Conf. on Robotics and Automation, Vol.4, pp.2864-2869, 1998

[7] C. Tseng and Y. Huang and J. Hu, "ESAIR: A Behavior-Based Robotic Software Architecture on Multi-Core Processor Platforms", Int. J. of Automation and Smart Technology, Chinese Institute of Automation Engineers, Vol.3, pp.47-56, 2013

[8] J. Zhen and H. Aoki and E. Sato-shimokawara and T. Yamaguchi, "Obtaining Objects Information from a Human Robot Interaction using Gesture and Voice Recognition", IWACIII 2011 Proceedings, pp.101_GS1_1, 2011

[9] T. Mitchell, Machine Learning, McGraw-Hill Science/Engineering/Math, 1997.

[10] R. S. Sutton and A. G. Barto, Reinforcement Learning, MIT Press, Cambridge, MA, 1998.

[11] S. Haykin, Neural Networks: A Comprehensive Foundation, Prentice Hall,1999.

[12] H. Mo and Z. Li, "Bio-geography based Differential Evolution for Robot Path Planning", Proc. of 2012 International Conference on Information and Automation, pp.1-6, 2012

[13] S. F. Arnold, R. Suzuki and T. Arita, "Modelling Mental Representation as Evolved Second Order Learning", Proc. of the 17th Int. Symp. on Artificial Life and Robotics, pp. 674-677, 2012

[14] M. Asada and S. Noda and S. Tawaratsumida and K. Hosoda, "Purposive Behavior Acquisition for a Real Robot by Vision-Based Reinforcement Learning", Machine Learning, Springer, Vol.23,pp.279-303, 1996.

[15] H. Kimura and T. Yamashita and S. Kobayashi, "Reinforcement learning of walking behavior for a four-legged robot", Proc. of the 40th IEEE Conf. on Decision and Control, Vol.1, pp.411-416, 2001.

[16] M. Hara and N. Kawabe and N. Sakai and J. Huang and H. Bleuler and T. Yabuta, "Consideration on Robotic Giant-swing Motion Generated by Reinforcement Learning", IEEE/RSJ Int. Conf. on Intelligent Robots and Systems,pp.4206-4211,2009.

[17] M. Kearns and S. Singh, "Near-Optimal Reinforcement Learning in Polynomial Time", Machine Learning, Springer, Vol.49, No.2-3, pp.209-232, 2002.

[18] K. Samejima and K. Doya and M. Kawato, "Inter-module credit assignment in modular reinforcement learning", Neural Networks, Elsevier Ltd., Vol.16, No.7, pp.985-994, 2003.

International Journal of
Soft Computing And Software Engineering (JSCSE)
e-ISSN: 2251-7545
Vol.3,No.2, 2013
Published online: Feb 25, 2013
DOI: 10.7321/jscse.v3.n2.1

[19] E. Uchibe and M. Asada and K. Hosoda, "Behavior Coordination for a Mobile Robot Using Modular Reinforcement Learning", Proc. of the 1996 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, Vol.3, pp.1329-1336, 1996.

[20] H. Valizadegan and R. Jin and S. Wang, "Learning to Trade Off Between Exploration and Exploitation in Multiclass Bandit Prediction", Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining, pp.204-212, 2011

[21] D. W. Stroock, An Introduction To Markov Processes, Springer, 2005.