STNS-R: A LEARNING METHOD FOR SEAMLESS TRANSPLANTATION FROM A VIRTUAL AGENT TO A PHYSICAL ROBOT

ATSUSHI UENO*, HIROAKI SOEDA*[1], HIDEAKI TAKEDA** * and MASATSUGU KIDODE*

* Graduate School of Information Science, Nara Institute of Science and Technology, Ikoma, Nara, Japan 630-0101
** Intelligent Systems Research Division, National Institute of Informatics, Tokyo, Japan 101-8430

Abstract

In this paper, we are concerned with the problem of how a physical robot can get an appropriate internal representation to its task and environment. Learning from experience is effective for the problem, but it is very time-consuming to learn a representation from the beginning in a real environment. On the other hand, the representation learned only in a simulated environment has the risk of not serving the purpose in a real environment because of the uncertainty in sensors, actuators, and the environment. In order to have the best of both worlds, it is effective to transplant the learned state representation of a virtual agent to a physical robot. For this purpose, we improved our developed incremental learning architecture for use in the real environment and developed a new architecture, called *STNS-R*. In this architecture, inappropriate negative instances caused by uncertainties are found on the basis of the distribution of instances and removed in order to correct the distorted shapes of the states. The effectiveness of STNS-R is shown in the experimental results.

Keywords

Categorical learning, uncertainty in the real environment, reactive agent, reinforcement learning.

1 Introduction

Machine learning is a promising and necessary method to realize an autonomous agent because it does not need a priori knowledge. Since most of learning methods require many trials to learn the given environment, they are usually performed in a simulated environment. It means that they are not applicable to robots in a real world. We need a seamless learning method that is applicable either in a simulated or in a real environment, i.e., the same algorithm is applicable and acquired knowledge is valid in both environments.

A physical robot must face various situations and cope with a huge amount of information. Learning from experience is effective for such a robot in acquiring an appropriate state representation for the current task. It is, however, very time-consuming to learn a representation from the beginning in a real environment. On the other hand, the state representation learned only in a simulated environment has the risk of not serving the purpose in a real environment because of the uncertainty in sensors, actuators, and the environment. In order to have the best of both worlds, it is effective to transplant the learned state representation of a virtual agent to a physical robot.

Incremental learning methods are suitable for this type of learning task because they can adapt seamlessly to changes of the environment. In our previous work [1, 2], we proposed *Situation Transition Network System (STNS)*, an architecture that performs categorical learning and behavioral learning incrementally in parallel with task execution. In categorical learning, it makes a state representation and modifies it according to the results of behaviors. Simulation results showed that it could learn efficiently and adapt to unexpected changes of the environment such as sensor trouble or actuator trouble.

For a preparatory experiment, we tested our method with a physical robot that started to learn with a STNS learned enough in a simulated environment. In the experiment, it could maintain the state representation and behaviors, but its performance got worse. Then, we improved STNS, and developed a new architecture by which a state representation learned in a simulated environment can be transplanted seamlessly and adapted to a physical robot. This paper proposes the improved STNS for the real world, called *STNS-R*.

The remainder of this paper is structured as follows: In the next section, we review our previous work, including explanations for the research area we tackled. Next, we show the seamless learning method to learn an appropriate state representation either in a simulated or in a real environment, and finally, we

[1] Currently, he is with H.Q.S CORPORATION.

0-7803-7087-2/01/$10.00 © 2001 IEEE

show the experimental results and give a discussion with concluding remarks.

2 The Domain and Our Previous Work

2.1 Categorical Learning in Reactive Agents

Physical robots are usually designed with fixed categories for recognizing their environments. However, human-designed categories are not always suitable for them. Learning the categories from experiences is effective in adapting to the environment.

We think that categories should be created on the criterion how well the robot can act. Several methods have been proposed which segment the state space enclosing input vectors from which the robot reaches the same reward by the same sequence of behaviors in the same state (*segmentation-based-on-behaviors*) [1-6]. This policy is suitable for reactive agents that assign one feasible behavior to each state because input vectors at which the robot should take the same behavior put together into the same state by this policy. Thus, this policy makes highly abstracted state representation specialized to the current task.

There are four previous works on this policy except for ours. Asada et al. [3] proposed a method that divides the state space by hyper-ellipsoids that enclose input vectors from which the robot achieves the goal or already acquired state by a variable sequence of one kind action primitive. Albus et al. [4] proposed a method that divides the state space using recursively a standard clustering algorithm. Yairi et al. [5] proposed a method that segments the state space using Bayesian classifier. Ishiguro et al. [6] proposed a method that divides states by hyper-planes in which the robot receives different rewards (or delayed rewards) when executing the same action.

In our previous work [1, 2], we developed this type of categorical learning to a new architecture, STNS, which performs segmentation-based-on-behaviors by incremental learning: the system extracts states and maintain them while executing the task on the basis of experiences of task-executing behaviors. This type of incremental learning contributes to autonomy, adaptability and efficiency of the learning system, and it is suitable for agents that continue task execution without breaking down against changes of the environment. In the next section, we explain this architecture in outline.

2.2 STNS

As shown in Fig. 1, STNS consists of a situation classifier, a situation transition network (STN), and several behavior modules. In each behavior step, the system identifies the current situation (or state) where the current input should be included, then makes a plan on the STN, and at last activates a behavior module according to the plan.

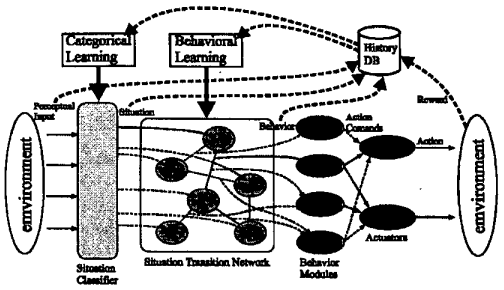


Fig. 1: The Structure of STNS

Categorical learning and behavioral learning are performed on the data that consists of the input, the corresponding situation, the selected behavior, and the received reward. This data puts into a history database. It keeps the data for a fixed period and deletes it after that. Therefore, STNS can adapt flexibly to the changes of environment and the internal representation without being misguided by outdated data.

In categorical learning, the system constructs a state representation and maintains it by segmentation-based-on-behaviors policy. Reinforcement learning of a behavior policy is performed on this state representation: the system constructs an *MDP* (Markov Decision Problem) model of the environment which consists of the transition probabilities between situations and the expectations of immediate rewards accompanying transitions by the maximum likelihood estimation. It decides an appropriate behavior by the *policy iteration* algorithm [7] and planning on the model in order to maximize discounted sum of rewards received over a period of time. In each behavior step, the system adjusts the shapes of situations in categorical learning and modifies the MDP model in behavioral learning reflecting the last data. If big modification is needed, the system extracts or eliminates situations in categorical learning and modifies the MDP model reflecting the new state representation in behavioral learning on the data in the history database. In this way, these two learning processes are performed in parallel while executing the task.

In the next section, we explain the state representation used in STNS. For further details of the whole architecture, please see [1] or [2].

2.3 State Representation in STNS

STNS segments the state space into some situations. In STNS, a situation is defined as a set of input vectors from which the system can meet with the specific result by the specific behavior. The specific behavior is called the *condition behavior* of the situa-

tion. The specific results are divided into two types, i.e., *R-situation* and *T-situation*. In a situation based on immediate rewards called R-Situation, the result is to receive a specific big reward (called the *goal reward*). In a situation based on situation transitions called T-Situation, the result is to transit to a specific situation (called the *parent situation*). If every chain of T-Situations is anchored to an R-Situation, every situation is guaranteed to lead to a goal reward by the same sequence of behaviors.

In incremental learning, the system should be able to decide rough shapes of situations from a limited amount of data, and to decide finer shapes as data increase. For this purpose, we contrived the *bitten hyper-ellipsoid* representation. In this representation, each situation is shaped by the positive instances and the negative instances that are decided based on the definition of the situation. As shown in Fig. 2, this representation is a mixture of macroscopic recognition and microscopic recognition. In macroscopic recognition, the boundary of a situation is a contour of Mahalanobis' distance from the population of the positive instances. This boundary forms a hyper-ellipsoid. This recognition is quick and rough that can be formed even from very few data. Microscopic recognition is realized by the Nearest Neighbor methods and grows finer as data increase. By mixing these two types of recognition, a fine and flexible recognition is realized.

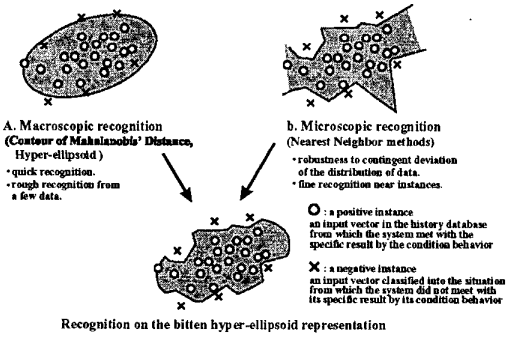


Fig. 2: The Bitten Hyper-Ellipsoid Representation

3 Preparatory Experiment

3.1 Task and Environment

The task for a mobile robot is to reach a goal area as shown in Fig. 3. Fig. 4 shows a picture of the physical robot with an omnidirectional vision sensor with a hyperboloidal mirror. It can find a goal target from everywhere in the plane and calculate its coordinates on the coordinate system fixed to the robot.

3.2 Results

First, we made a virtual agent with STNS and let it learn enough in a simulated environment corresponding to the real environment. Then, we transplanted the well-learned STNS to a physical robot and let it continue learning in the real environment. This procedure was planned in consideration of practical use where learning in a real environment is very time-consuming and should be transferred to a simulated environment as much as possible. Fig. 5a shows the state space of the transplanted STNS after 2000 behavior steps learning in the simulated environment. Learning converged thoroughly until then. Relearning in the real environment was performed until 1000 behavior steps. We repeated this experiment five times. Table 1 shows the average numbers of behaviors that are needed to reach the goal. All numbers of behaviors to the goal was averaged in each experiment. The average of these numbers is 2.53 and the standard deviation is 0.05.

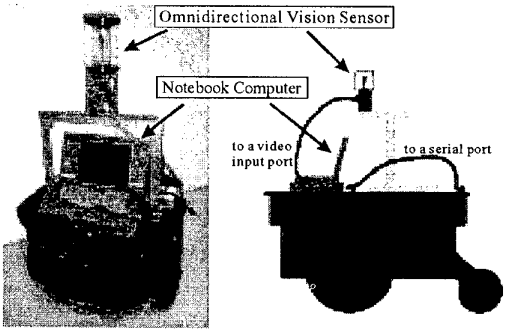


Fig. 4: Our Physical Robot

Task:
To navigate the center of a 0.44m x 0.33m rectangle robot from an arbitrary position into the goal area (a circle whose radius is 0.50m) in a 3.00m x 3.00m square plane.
The goal area is settled at an arbitrary position in the plane.
There is no obstacle but the wall around the room.

Perceptual Input (2-dimensional):
The real-valued (x, y) coordinates of the center of the goal on the coordinate system fixed to the robot.
The origin is settled at the center of the robot, and the x-axis points to the front of the robot.

Rewards:
1. Arrival at the goal: +10
2. Collision with the wall: -1
3. Trying to rotate over 90 degrees: -1
(The robot can look any direction by at most 90 degrees rotation because of the symmetry of actions.)

Behavior:
a sequence of the same simple actions which is continued until one of the following stopping conditions is fulfilled.
Actions: forward movement, backward movement, clockwise rotation, counterclockwise rotation.
(The robot is assumed to be able to make collision-free rotation.)

Stopping Conditions of Behaviors:
1. Getting some reward.
2. Arrival at the current target situation in the plan.

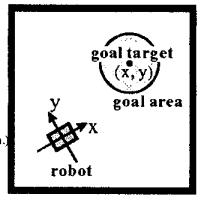
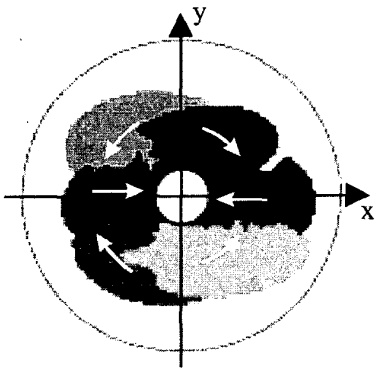


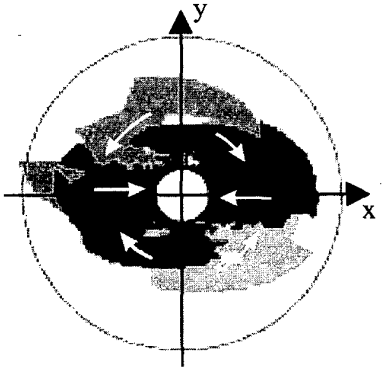
Fig. 3: The Navigation Problem

As a comparative experiment, the virtual agent continued learning until additional 1000 behavior

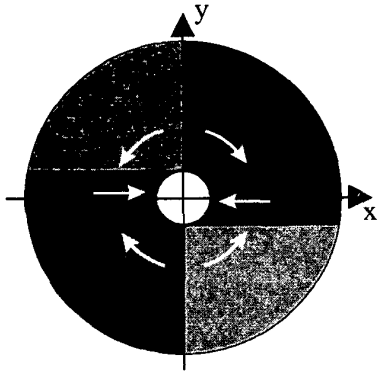
steps. We repeated this experiment ten times. The



a. Before relearning



b. After relearning



c. Optimum state space

condition behaviors
forward move
backward move
clockwise rotation
counterclockwise rotation

Fig. 5: State Spaces before and after Relearning and the Optimum State Space

Table1: Results of the Original STNS

Experiment #	Ave. # of Behaviors to the Goal
1	2.52
2	2.46
3	2.53
4	2.57
5	2.57

total average of the numbers of behaviors to the goal is 2.04. Compared with this number, the result of STNS is 23.4 % worse.

Fig. 5b shows an example of state spaces after 1000 behavior steps relearning in the real environment. The small circle (the radius is 50cm) at the center of each space denotes states in which the rover arrives at the goal. The big circle (the radius is about 283cm) surrounding each space denotes the longest distance between the rover and the goal. Compared with the state space before relearning (Fig. 5a) or the optimum state space (Fig. 5c), its shape is rather distorted.

4 The Improved STNS for the Real World

4.1 Problems with Our Previous Work

The preparatory experiment showed that learning in the real environment was worse than in the simulated one in performance. We suspect that incorrect distortion of state shapes causes the low performance of STNS in the real environment. As shown in Fig. 6, causes of the low performance are connected with each other and form a vicious circle. Distortion of state shapes is related to all causal relationships denoted by solid arrows in this figure. By reducing this distortion, the performance will be improved to a considerable extent.

This incorrect distortion of state shapes is caused by experience in the real environment that has various types of uncertainty as shown by broken arrows in Fig. 6. In STNS, each state is represented by using

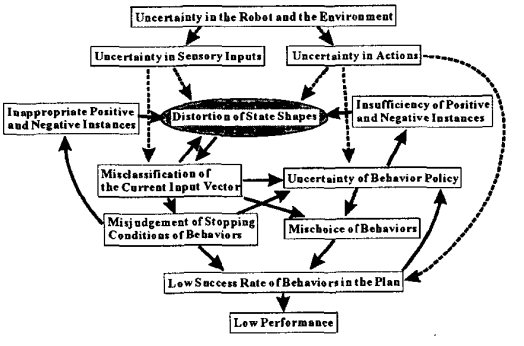


Fig. 6: Vicious Circle Causative of Low Performance

positive instances and negative instances collected while executing the task as shown in Fig. 2. It works well in a simulated environment where every instance is represented exactly, but it may have problems in a real environment where instances have uncertainty in their representation.

4.2 The Method

The proposed method so-called STNS-R is an improved version of STNS to adapt to uncertainty in sensors, actuators, and the environment in the real world. In order to correct the distorted shapes of the states, STNS-R removes inappropriate negative instances caused by the uncertainty from the state representation. It judges appropriateness of instances from their distribution using the following heuristics: the more positive instances exist around a negative instance, the more suspicious the validity of the instance is. Removal of inappropriate positive instances is not performed because positive ones may be much more than negative ones and it is difficult to judge their appropriateness from fewer negative ones around them. Negative instances are apt to be short because collecting many negative instances is in conflict with achieving high performance in task execution.

Fig. 7 shows how to discriminate between inappropriate negative instances and appropriate ones in STNS-R. It calculates all visual angles between adjacent positive instances in view of a negative instance. If the maximum angle is less than a threshold, the surrounded negative instance is judged inappropriate. The threshold is fixed empirically to 90 degrees in this task. We chose not the minimum distance between a positive instance and a negative instance but the maximum visual angle between adjacent positive instances as an attribute for discrimination because the former cannot discriminate in the neighborhood of boundaries of states.

This discrimination process is performed when a new negative instance or a new positive one is added to a state. When a new negative one is added, the system checks it with all positive instances in the state. If it is judged inappropriate, it is removed from the state. When a new positive one is added, the system checks all negative instances one at a time with all positive ones including the newest. All negative instances judged inappropriate are removed from the state.

The advantage of the method is that it is based only on the distribution of instances and needs neither a priori knowledge on uncertainty of sensors, actuators, and the environment nor redundant sensory information. It is good at maintaining a state that has a convex shape. On the other hand, it cannot maintain a ring-shaped state or a state divided into several areas. Therefore, it is unsuitable for problems in which the optimum state representation has such states. In the problem to which it applied, the ideal loci, that is the loci without uncertainties, of the condition behaviors in the state space should fulfill both of the following two conditions:

1. Loci are continuous.
2. If two loci pass two sufficiently neighboring points, they are neighboring in their lower courses.

These conditions are broken in many problems in which behavior modules make complicated behaviors. Fig. 8 shows three examples of such cases. In the simple problem tackled in this paper (shown in Fig. 3), these conditions are fulfilled.

The proposed method is limited to a problem with a two-dimensional state space. We are tackling the problem of extending the idea of discriminating with the maximum visual angle to a multidimensional state space.

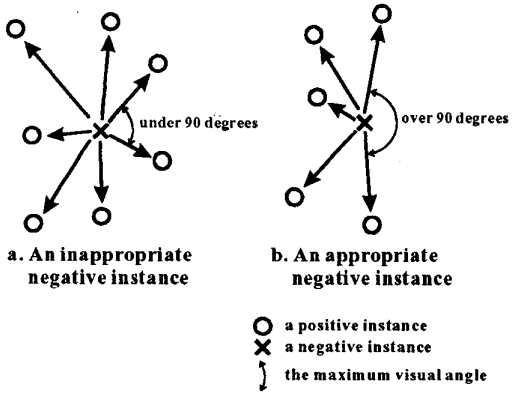


Fig. 7: An Inappropriate Negative Instance and an Appropriate One

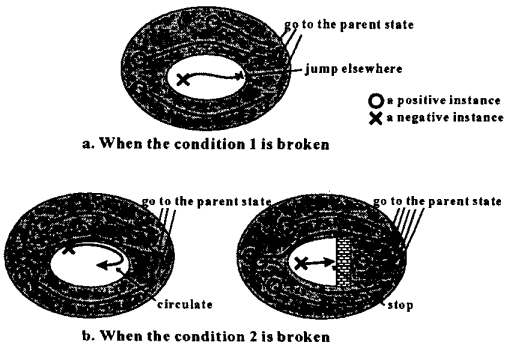


Fig. 8: Unsuitable Cases for Our Method

5 Experimental Results

In order to test the improved STNS, STNS-R, we conducted the same experiment as the original STNS: we transplanted a state representation learned enough in a simulated environment (Fig. 5a) to the physical robot and let it continue learning until 1000 behavior steps. We repeated this experiment five times. With STNS-R, it could adapt seamlessly to the real environment and maintain the state representation, behaviors, and high performance all through the experiment. Table 2 shows the average numbers of behaviors that are needed to reach the goal in the same form as Table 1. The average of these numbers is 2.28 and the standard deviation is 0.02. Compared with the average of the virtual agent in an ideal environment without uncertainties, 2.04 as mentioned above, the deterioration of performance were limited to 11.8 %. This results means that STNS-R prevented 47.9 % of unnecessary deterioration in performance as compared with the original STNS.

Table2: Results of STNS-R

Experiment #	Ave. # of Behaviors to the Goal
1	2.25
2	2.30
3	2.28
4	2.27
5	2.30

6 Conclusions

We have proposed a new architecture, called STNS-R, by which a state representation learned in a simulated environment can be transplanted seamlessly and adapted to a physical robot. In this architecture, inappropriate negative instances caused by uncertainties in the real environment are found on the basis of the distribution of instances and removed in order to correct the distorted shapes of the states. The effectiveness of STNS-R was shown in the experimental results. The advantage of STNS-R is that we can switch the learning environment from a simulated one to a real one at anytime because of the incrementality of learning. Even if the transplanted state representation should not be mature, it would be adapted by learning in the real environment.

STNS-R is the only learning method, as far as we know, which can transplant seamlessly a state representation learned in a simulated environment to a physical robot. Any other learning method needs a large number of data of random behaviors in a real environment in order to construct an appropriate state representation. Transplanting a state space from a virtual agent to a physical robot can accelerate learning very much.

STNS-R is effective also in learning from the beginning in a real environment because it can select appropriate data among a corrupt set of data because of uncertainty in the real environment.

As mentioned in Section 4.2, there is limitation in the problem area to which the proposed method for discriminating inappropriate instances can be applied. We are tackling the problem of improving the definition of state in order to simplify state shapes, as well as extending the state representation to be more expressive.

References

1. U. Ueno, H. Takeda and T. Nishida, Learning of the Way of Abstraction in Real Robots, Proc. of the IEEE International Conference on Systems, Man and Cybernetics (SMC'99) (CD-ROM), vol. 1, pp. II746-II751, 1999.
2. U. Ueno and H. Takeda, Cooperation of Categorical and Behavioral Learning in a Practical Solution to the Abstraction Problem, New Generation Computing, vol. 19, no. 3, pp. 257-282, 2001
3. M. Asada, S. Noda and K. Hosoda, Action-Based Sensor Space Categorization for Robot Learning, Proc. of the 1996 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 96), pp. 1502-1509, 1996.
4. J. Albus, A. Lacaze and A. Meystel, Multiresolutional Intelligent Controller for Baby Robot, Proc. of the 10th International Symposium on Intelligent Control, 1995.
5. T. Yairi, S. Nakasuka and K. Hori, Sensor Fusion for State Abstraction Using Bayesian Classifier, Proc. of the IEEE International Conference on Intelligent Engineering Systems (INES'98), pp. 99-104, 1998.
6. H. Ishiguro, R. Sato and T. Ishida, Robot Oriented State Space Construction, Proc. of the 1996 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 96), pp. 1496-1501, 1996.
7. S. Russell and P. Norvig, Artificial Intelligence: A Modern Approach, Prentice Hall, 1995.