Reasonable Performance in Less Learning Time by Real Robot Based on Incremental State Space Segmentation

Yasutake Takahashi, Minoru Asada and Koh Hosoda

Dept. of Mech. Eng. for Computer-Controlled Machinery
Osaka University, 2-1, Yamadaoka, Suita, Osaka 565, Japan
yasutake@robotics.ccm.eng.osaka-u.ac.jp

Abstract

Reinforcement learning has recently been receiving increased attention as a method for robot learning with little or no a priori knowledge and higher capability of reactive and adaptive behaviors. However, there are two major problems in applying it to real robot tasks: how to construct the state space, and how to reduce the learning time. This paper presents a method by which a robot learns purposive behavior within less learning time by incrementally segmenting the sensor space based on the experiences of the robot. The incremental segmentation is performed by constructing local models in the state space, which is based on the function approximation of the sensor outputs to reduce the learning time and on the reinforcement signal to emerge a purposive behavior. The method is applied to a soccer robot which tries to shoot a ball into a goal. The experiments with computer simulations and a real robot are shown. As a result, our real robot has learned a shooting behavior within less than one hour training by incrementally segmenting the state space.

1 Introduction

Reinforcement learning has recently been receiving increased attention as a method for robot learning with little or no a priori knowledge and higher capability of reactive and adaptive behaviors[1]. However, there are two major problems in applying it to real robot tasks.

1. Selection of the sensor information to describe the state of robots and their environment. If one uses all the sensor outputs, the amount of the data the robot has to deal with will exceed the capability of the robot (memory and processing power).

2. Even though the sensor information is well selected for the given task, the segmentation problem will remain. The state space designed by the programmer is not guaranteed as an optimal one for the robot to perform the task. The coarse segmentation will cause so-called "perceptual aliasing problem"[2] by which the robot cannot discriminate the states important to accomplish the task at hand. On the other hand, the fine segmentation to avoid the perceptual aliasing problem will produce too many states to generalize the experiences. Since the learning time increases exponentially with the number of states, the robot needs enormous amount of learning time.

For the former, Whitehead and Ballard [2] proposed a selection method of the sensor information in order to avoid the perceptual aliasing. Tan [3] proposed a method of sensor selection that reduces the sensing cost. Chapman and Kaelbling [4] proposed an algorithm based on recursive splitting of the state space based on statistical measures of the differences in reinforcements received. However, they have dealt with the discrete state space, therefore, these methods cannot be directly applied to continuous state space.

For the latter, roughly speaking, there are two approaches for continuous state space: learning the value function with a method of function approximation or with segmentation of continuous state space.

Boyan et al. [5] reported on the method for the function approximation that the combination of dynamic programming and function approximation had shown poor performances even for benign cases. Then, they proposed Grow-Support algorithm for the function approximation. However, they need the environmental model and can cope with only deterministic worlds. Sutton [6] used CMAC[7][8] as a method of the function approximation. CMAC has its own problem of quantization (segmentation). Also, Saito and Fukuda [9] used CMAC to estimate the Q values. However, the sensor space was huge and they needed enormous learning time, therefore they reduced the searching space by using the initial controller.

As a method for state space segmentation, Kröse and Dam [10], and Dubrawski and Reignier [11] used the reinforcement signal to divide the state space, therefore the space far from the states given the reinforcement signal have not been segmented. Asada et al. [12] proposed a method which cuts off regions from the state space as states recursively from the goal state. In order to construct the state space suitable for the robot to perform the given task, they need a sufficient of uniformly sampled data.

Further, since the reinforcement learning generally begin to updates the action values from the state given the reinforcement, the experiences before the reinforcement is given or far from the states where the reinforcements are given might be vain. Connel and Mahadevan [1] decomposed the whole task into subtasks each of which can be separately learned. However, task decomposition and behavior switching are designed by the programmer.

In this paper, we propose a method by which a robot learns purposive behavior within less learning time by incrementally segmenting the sensor space based on the experiences of the robot. Here, we do not deal with the sensor selection problem. The incremental segmentation is performed by constructing local models in the state space, which is based on the function approximation of the sensor outputs to reduce the learning time and the reinforcement signal to emerge a purposive behavior. The method is applied to a soccer robot which tries to shoot a ball into a goal. The experiments with computer simulations and a real robot are shown. As a result, our real robot has learned a shooting behavior within less than one hour training by incrementally segmenting the state space.

The remainder of the article is structured as follows: In the next section, we briefly review the reinforcement learning, and then explain the basic idea of the method and the algorithm. Next, we give the task and assumptions. Finally, we show the experimental results by the computer simulations and the real robot system, and give conclusions.

2 Basics of Reinforcement Learning

Before getting into the our method, we briefly review the basics of the reinforcement learning.

We assume that the robot can discriminate the set $\boldsymbol{S}$ of distinct world states, and can take an action from the action set $\boldsymbol{A}$. The world is modeled as a Markov process, making stochastic transitions based on its current state and the action taken by the robot. For each state-action pair (s, a), the reward $r(s, a)$ is defined.

The general reinforcement learning problem is typically stated as finding a policy that maximizes discounted sum of the reward received over time. Watkins' Q-learning algorithm [13] gives us elegant method for doing this.

In the Q-learning algorithm, the robot takes an action $a \in \boldsymbol{A}$ in a state $s \in \boldsymbol{S}$ and transits to the next state $s' \in \boldsymbol{S}$, then it updates the action-value function $Q(s, a)$ as follows.

$$Q(s,a) \Leftarrow (1-\alpha)Q(s,a) + \alpha(r(s,a) + \gamma \max_{a' \in \boldsymbol{A}} Q(s', a')) \tag{1}$$

where α is a learning rate and γ is a discounting factor.

After a sufficient number of trials, the action a which maximize the $Q(s, a)$value is the optimal decision policy at the state s.

3 Basic Idea

In this paper , we focus on the state space segmentation, and reduction of learning time. As a basic idea coping with these problems, we adopt the incremental segmentation of the state space by which the state space is autonomously segmented, and we expect the reduction of the learning time and the capability of coping with dynamic change of the environment.

A key issue is to find the basic policy to segment the state space so as to realize the desirable features described above. The following two policies can be considered.

A: Segment the state if the prediction of sensor outputs is incorrect.

B: Segment the state if the same action causes the desirable or undesirable result (ex., transition to the goal states or non-goal states) even though the prediction itself is correct.

According to the first policy, the robot can discriminate the world situations with as few states as possible based on the experiences until the current time. This contributes to the followings:

1. as long as the prediction of sensor outputs is correct, tedious exploration process can be eliminated, and therefore
2. reinforcement learning converges immediately. Further
3. the robot can cope with dynamic change of the environment due to its incrementality of the segmentation process.

However, the policy **A** does not care where the goal state is. On the other hand, the policy **B** contributes to the emergence of the purposive behavior. Even though the prediction is correct, it would be nonsense if the same action from the same state resulted in different situations. This state should be separated so that the same action can always cause the desirable transition.

From the above arguments, the policy **A** is related to the world model construction by coarse mapping between states and actions far from the good states. While, the policy **B** is related to the the goal oriented segmentation based on the reinforcement signals. As a result fine mapping between states and actions near the goal states is obtained.

4 Algorithm

Fig.1 shows the rough flow of the proposed method. First, the robot acquires sensor outputs as data. If the data are consistent with the current local models, the robot updates the local models. Else, the robot builds new local models, and initialize the action value function by reusing the knowledge obtained by the past experiences. Then, it learns the policy using reinforcement learning, and returns the beginning. The robot iterates this cycle forever.

4.1 Action Space and Data Structure

In the conventional reinforcement learning methods, an "action" is defined as an execution of motor command per fixed sampling interval. In real situation, this definition often causes "state-action deviation problem" as pointed out by Asada et al.[14]. They defined such an action as an action primitive, and a

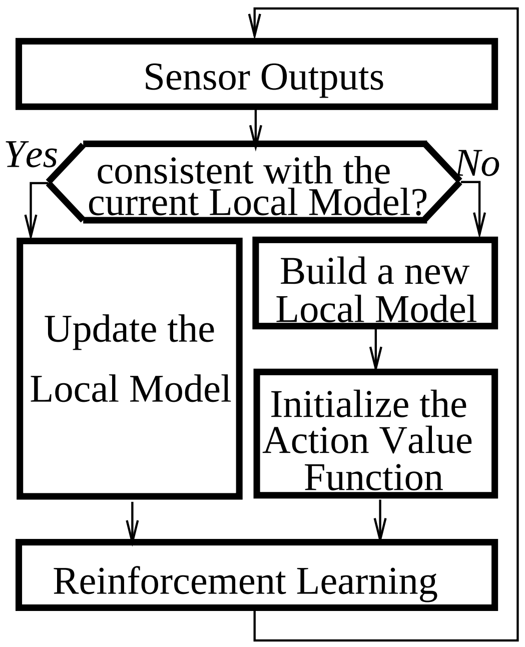


Figure 1: The rough flow of the proposed method

action is defined as a sequence of action primitives until the current state changes. Here, we follow their definition.

We define a data set $d_i \in D$, $(i = 1, 2, \cdots)$ as a triplet of action primitive $m_i \in M$, sensor output $\boldsymbol{s}_i \in S$ and its gradient $\dot{\boldsymbol{s}}_i \in \dot{S}$.

If the robot stores the all data of its experiences, the amount of data will exceed the capacity of the robot. Therefore, it is not practical to store the all data. Further, the robot often receives incorrect data because of sensor noise, change of the environment, and the uncertainty of motor commands. Then, we update the gradient of inputs vector $\dot{\boldsymbol{s}}_i$, when the robot receives a new data set d_j.

if

$$|\boldsymbol{s}_i - \boldsymbol{s}_j| < \epsilon \ \ and \ \ m_i = m_j$$

then

$$\dot{\boldsymbol{s}}_i = (1 - \beta)\dot{\boldsymbol{s}}_j + \beta \dot{\boldsymbol{s}}_i$$

else

register $\boldsymbol{s}_j$ as a new datum.

Here, $0 < \beta < 1$ and ϵ stands for a similarity threshold. $|\cdot|$ means weighted Euclidean norm.

4.2 Local Model Construction

We first explain the method of local model construction by using a linear model of the the gradient of sensor outputs, that is,

$$\dot{\boldsymbol{s}} = \boldsymbol{A}\boldsymbol{s} + \boldsymbol{b}.$$

The algorithm for local model construction and segmentation is as follows:

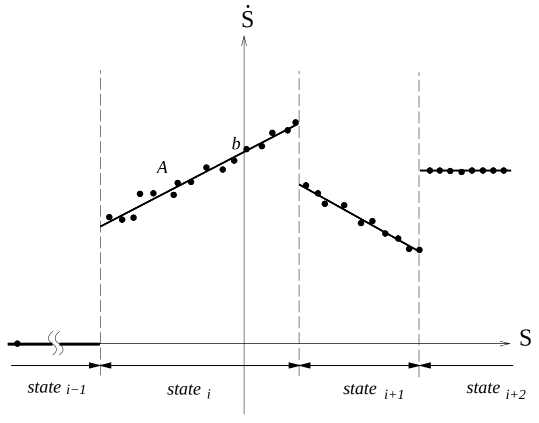


Figure 2: The construction of local model and the segmentation of sensor space

1. Gather data sets which have the same action primitive.
2. Apply the weighted linear regression method to fit a linear model to the data sets.
3. Divide the data into two with a method of cluster analysis using weighted Euclidean norm as similarity and return 2 if the unbiased variance of the residual exceeds a certain threshold, else stop.

Fig.2 shows an example of the construction of local model and the segmentation of sensor space in case of one dimension of the sensor output.

Near the goal state, the segmented region obtained by the above process is not always appropriate because multiple transitions (success in the reaching the goal state or failure), from a same pair of the sensor outputs in the same region and the action primitive can be often observed. Then, we use the reinforcement signals to divide the segmented region so that the same action primitive from the divided region can reach the unique state (the goal state or others).

The segmented regions obtained by the above process are regarded as "states" for the reinforcement learning method. Each segmented region has several data sets d_i, and let the $\boldsymbol{s}_i$ $(i = 1, 2, \cdots)$ be the representatives of the region. A new sensor outputs $\boldsymbol{s}_q$ is classified into one of the states by finding a representative in the corresponding state based on NN(nearest neighbor) methods.

4.3 Action Generation

As we stated in the section 4.1, we define "action" as "a sequence of several action primitives until the current state changes". The sequence of action primitives with the local model is generated as follows.

One can calculate the desired gradient of sensor outputs $\dot{\boldsymbol{s}}_d$ from the current sensor outputs $\boldsymbol{s}_j$ and desired sensor outputs $\boldsymbol{s}_d$, that is,.

$$\dot{\boldsymbol{s}}_d = \boldsymbol{s}_d - \boldsymbol{s}_j.$$

Since the linear model parameters have been obtained in each local model, we can predict a desirable action to satisfy the above equation. The robot carries out the action primitive m_d which is closest to the desired gradient of sensor outputs.

$$m_d = arg \min_{m_i} (\dot{\boldsymbol{s}}_d - \dot{\boldsymbol{s}}_{m_i})^2 \tag{2}$$

We assumed the continuity of sensor space. However, if the robot cannot observe the objects in the environment, the robot cannot obtain the information about the objects from sensors. Therefore, there is a case that equation (2) cannot be applied. In such cases, however, an action for a state transition is needed, then we adopt a sequence of the same action primitive as one action until the current state changes.

4.4 Reuse of the Knowledge Obtained by Experiences

Theoretically, the action value function should be reset every time the new state space is constructed by the incremental segmentation of the state space. This prevents the knowledge obtained by the past experiences from being used efficiently in the learning process. Then, we consider to reuse the knowledge by calculating the new action value function for the new segmented state space from the old state space and its action-value function.

Basic idea is to adopt a new action value function calculated by weighted sum of the old action value function as the initial knowledge for reinforcement learning. The weights are calculated based on the numbers of the sensor output representatives in both the new and old states. Concrete procedure is given as following.

S^{old} and S^{new} denote the old and new state spaces, respectively. $\boldsymbol{s}_k(k = 1, 2, \cdots, n)$, $state_j{}^{old}(j = 1, 2, \cdots, n^{old})$ and $state_i{}^{new}(i = 1, 2, \cdots, n^{new})$ denote the sensor output of stored data d_i, a state of the old state space and a state of the new state space. We prepare a $n^{old} \times n^{new}$ matrix $T(state^{old}, state^{new})$ of which component $t(state_i^{old}, state_j^{new})$ represents the number of sensor output representatives $\boldsymbol{s}_k$ that are classified into $state_j^{new}$ from $state_i^{old}$. Then, we can calculate the action-value function of the new state space $Q(state_i^{new}, a)$ as follows.

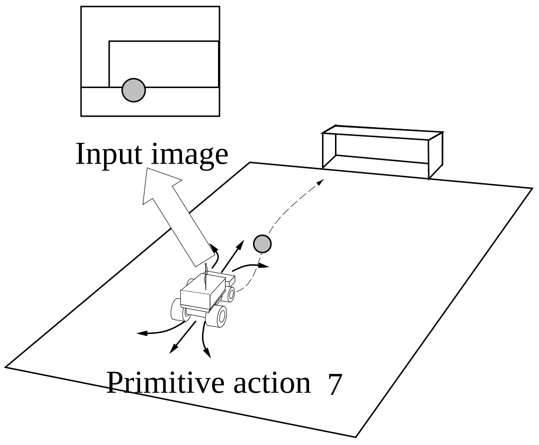
$$Q(state_i^{new}, a) = \sum_{j=1}^{n^{old}} \omega_{ji} Q(state_j^{old}, a), \tag{3}$$

where

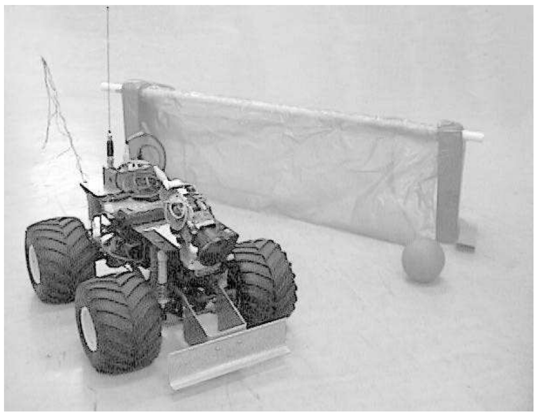
$$\omega_{ji} = \frac{t(state_j^{old}, state_i^{new})}{\sum_{l=0}^{n^{old}} t(state_l^{old}, state_i^{new})}. \tag{4}$$

5 Task and Assumptions

Only one assumption we need is continuity of the sensor space. This makes local model construction efficient, and therefore contributes to eliminate unnecessary exploration.



(a) The task is to shoot a ball into the goal



(b) A picture of the radio-controlled vehicle with a ball and a goal

Figure 3: A task and our real robot

We apply the method to shooting behavior acquisition by a soccer robot as an example of robot tasks. The task for a mobile robot is to shoot a ball into a goal as shown in Fig.3(a).

We assume that the environment consists of a ball and a goal, that the mobile robot has a single TV camera and can get the primitive features of the ball and the goal, and that the robot does not know the location and the size of the goal, the size and the weight of the ball, any camera parameters such as focal length and tilt angle, or kinematics/dynamics of itself. Fig.3(b) shows a picture of the real robot, the ball and the goal used in the experiments.

The sensor information the robot discriminates consists of five features of the ball and the goal. The features are the size and the position of both the ball and the goal on the image. In addition to them, the goal has its orientation as the fifth feature. These features are obtained by the principal component analysis for image data taken by the robot. The robot often loses the ball and/or goal because of its narrow angle of view (65°). In such a case, there are no feature values of ball and/or goal. However since the robot knows into which direction it lost the ball and/or the goal by memorizing the previous state, large absolute constants (opposite signs) are assigned to these lost states. As the result, the local models for these states are obtained with their gradients equal zeros (The left side of Figure 2 indicates such a case).

6 Simulation

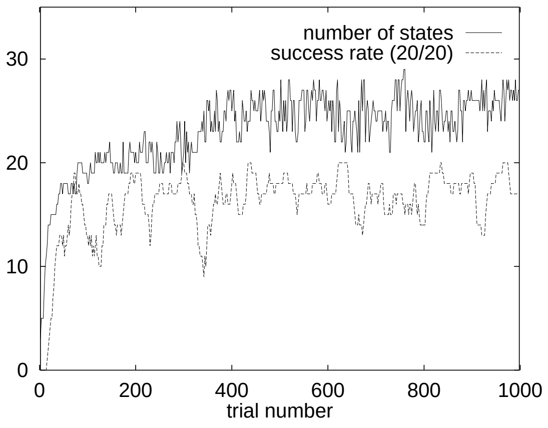


Figure 4: The success rate and the number of states

The number of action primitives which the robot can take is seven as shown in Fig.3(a). We assign a reward value 1 when the ball was kicked into the goal or -0.1 otherwise. 90% of the time the robot selects the action specified by its optimal policy, the remaining 10% of the time it takes a random action.

Fig.4 shows the success rate and the number of states during the incremental state space segmentation and the processes of shooting behavior acquisition. Here, the success rate indicates the number of successes in the last twenty trials.

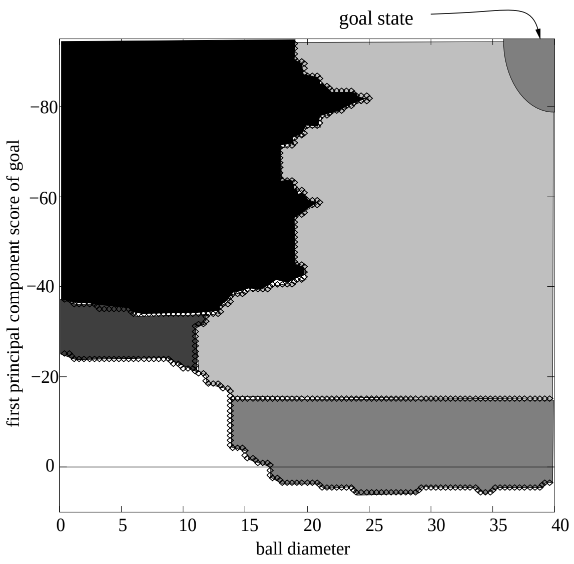


Figure 5: Result of state space construction

Fig.5 shows a projection of the state space after 1,110 trials, where the state space in term of ball size and goal size is indicated when the position of the ball and the goal are center of the screen and the orientation of the goal is frontal.

Fig.6 shows the success rate and the number of states in the case that the ball diameter suddenly became twice at the 500th trial. It suggests the proposed method can deal with dynamic change of the environment.

7 Experiment on the Real Robot

Fig.7 shows a configuration of the real mobile robot system. Fig.8(a) and (b) show the images taken by a TV camera mounted on the robot and processed by Datacube MaxVideo 200, a real-time pipeline video image processor. The image processing and the vehicle control system are operated by VxWorks OS on MC68040 CPU which are connected with host Sun workstations via Ether net. The result of image processing are sent to the host CPU to decide an optimal action against the current state. The sampling time is about 30ms.

Fig.9 shows the state space after 72 trials. The state space in term of ball size and goal size is indicated when the position of the ball and the goal are center of the screen and the orientation of the goal is frontal. The numbers of acquired states and data are 18 and 151, respectively.

Fig.10 shows how the robot tries to shoot a ball into the goal. Because of the sensor noise and the uncertainty of the motor commands, the robot often misunderstands the states, and takes wrong actions, therefore it fails to do the task. ① indicates that the

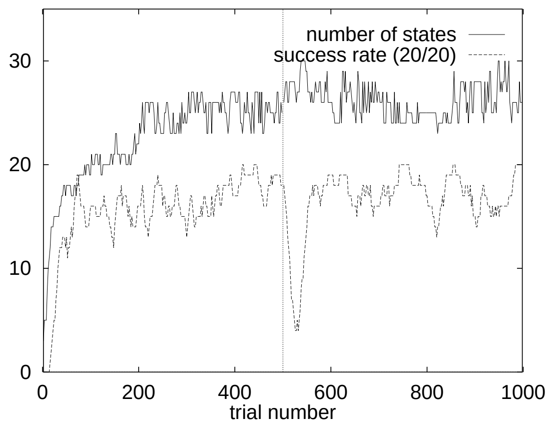


Figure 6: The success rate and the number of states in the case that environment change one the way

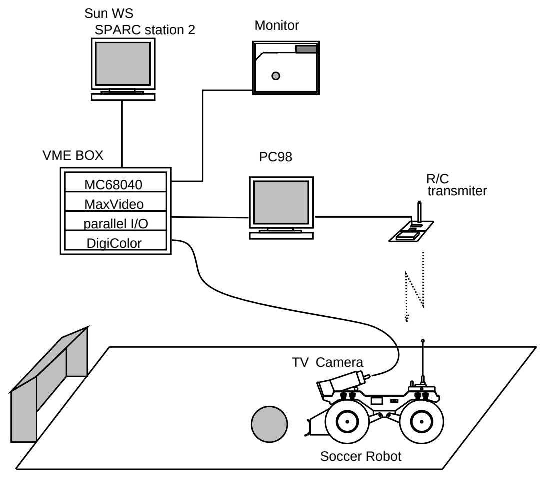


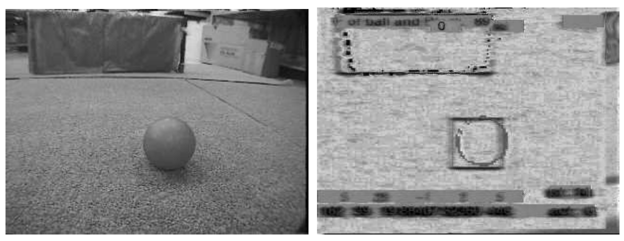
Figure 7: A configuration of the real robot

robot is going to shoot a ball into the goal and move forward. But it fails to kick the ball at ② because the speed is too hight to turn. The ball is occluded by the robot in ②. Then, it goes left back so that it can shoot a ball at ③. But it fails again at ④. Then it goes left back again at ⑤. After all, the robot does the shooting task successfully at ⑥.

8 Conclusion and Future Work

This paper presented a method of incremental segmentation of sensor space based on the experiences of the robot, by which the robot learns purposive behavior within reasonable learning time.

Let us discuss to what extent the proposed method can be scaled up. In the linear local model, we will be able to easily cope with reaching multiple stationary goals or avoiding stationary obstacles, because the gradient of the sensor outputs can be reasonably explained by the linear model. However, the more complicated task such as reaching the goal with obstacle avoidance, seems difficult because the state space suitable for multiple tasks is difficult to build by the current linear local model. Further, in case of the environment including other agents, collaboration/competition with them are the focused task to the robot, and actions of other agents seem difficult to be explained by the current model because they are not simply related to the actions of the robot. Behaviors of collaboration/competition might have much more complicated relationship to the robot behavior and a method being able to cope with these highly complicated relationship between the robot actions and other agents' behaviors should be developed.



(a) input image (b) detected image

Figure 8: Detection of the ball and the goal

Acknowledgment

We like to thank Masateru Nakamura for his support for the real robot experiments.

References

[1] Jonalthan H. Connell and Sridhar Mahadevan. *ROBOT LEARNING*. Kluwer Academic Publishers, 1993.

[2] Steven D. Whitehead and Dana H. Ballard. Active perception and reinforcement learning. In *MLWS*, pages 179–188, 1990.

[3] Ming Tan. Learning a cost-sensitive internal representation for reinforcement learning. In *Proceedings Eighth International Workshop on Machine Learning*, pages 358–362, 1991.

[4] David Chapman and Leslie Pach Kaelbling. Input generalization in delayed reinforcement learning: An algorithm and performance comparisons. In *AAAI'91*, pages 726–731, 1991.

[5] Justin Boyan and Andrew Moore. Generalization in reinforcement learning: Safely approximating the value function. In *Proceedings of Neural Information Processings Systems 7*. Morgan Kaufmann, January 1995.

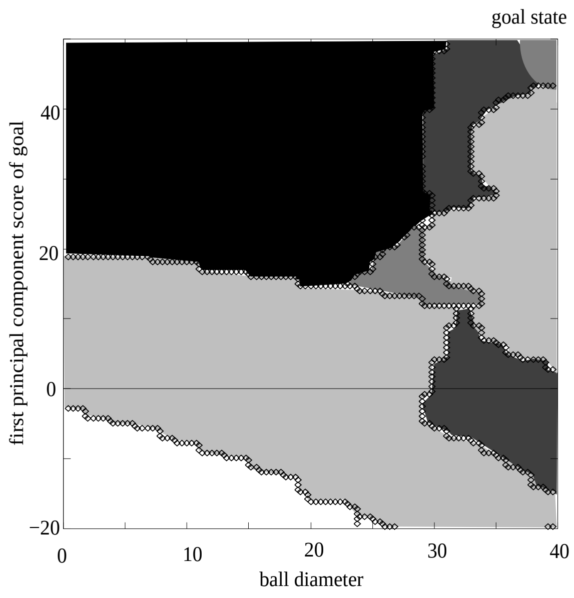


Figure 9: state space construction of real robot experiment

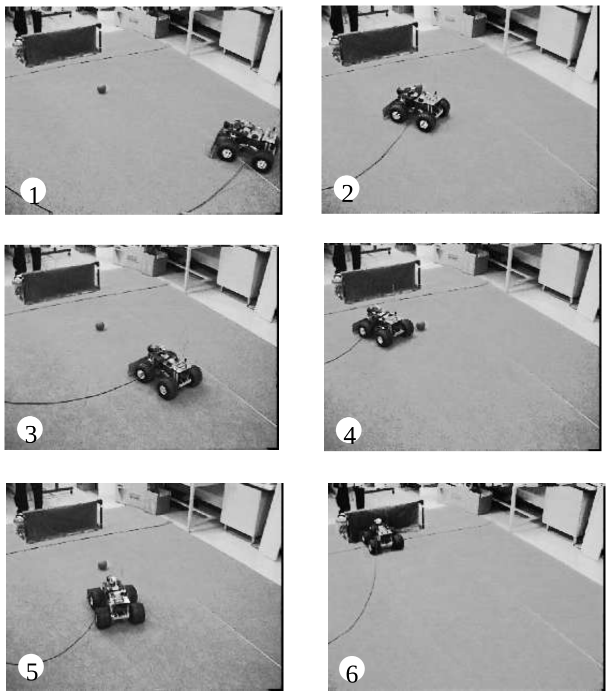


Figure 10: The robot succeeded in shooting a ball into the goal

[6] Richard S. Sutton. Generalization in reinforcement learning: Successful examples using sparse coarse coding. In *Proceeding of Neural Information Processing Systems(NIPS 95)*, 1995.

[7] J. S. Albus. A new approach to manipulator control: The cerebbellar model articulation controller (cmac). *Journal of Dynamic Systems, Measurement, and Control, Trans. ASME*, 97(3):220–227, Sept 1975.

[8] J. S. Albus. Data storage in the cerebellar model articulation controller (cmac). *Journal of Dynamic Systems, Measurement, and Control, Trans. ASME*, 97(3):227–233, Sept 1975.

[9] Fuminori Saito and Toshio Fukuda. Two-link-robot brachiation with connectionist q-learning. In *From Animals to Animats 3*.

[10] Ben J. A. Kröse and Joris W. M. van Dam. Adaptive state space quantisation for reinforcement learning of collision-free navigation. In *Proceedings of the 1992 IEEE/RSJ International Conference on Intelligent Robots and Systems*, volume 2, pages 1327–1331, 1992.

[11] Artur Dubrawski and Patrick Reignier. Learning to categorize perceptual space of a mobile robot using fuzzy-art neural network. In *Proceedings of the IEEE/RSJ/GI International Conference on Intelligent Robots and Systems*, volume 2, pages 1272–1277, September 1994.

[12] Minoru Asada, Shoichi Noda, and Koh Hosoda. Non-physical intervention in robot learning based on lfe method. In *Proc. of Machine Learning Conference Workshop on Learning from Examples vs. Programming by Demonstration*, 1995.

[13] C.J.C.H.Watkins. *Learning from delayed rewards.* PhD thesis, King's College, University of Cambridge, May 1989.

[14] Minoru Asada, Shoichi Noda, Sukoya Tawaratsumida, and Koh Hosoda. Vision-based reinforcement learning for purposive behavior acquisition. In *Proceedings of 1995 IEEE International Conference on Robotics and Automation*, volume 1, pages 146–153, 1995.