Continuous Valued Q-learning for Vision-Guided Behavior Acquisition

Yasutake Takahashi, Masanori Takeda, and Minoru Asada
Dept. of Adaptive Machine Systems
Graduate School of Engineering Osaka University
Suita, Osaka 565-0871, Japan

yasutake,takeda,asada@er.ams.eng.osaka-u.ac.jp

Abstract

Q-learning, a most widely used reinforcement learning method, normally needs well-defined quantized state and action spaces to converge. This makes it difficult to be applied to real robot tasks because of poor performance of learned behavior and further a new problem of state space construction.

This paper proposes a continuous valued Q-learning for real robot applications, which calculates contribution values for estimate a continuous action value in order to make motion smooth and effective. The proposed method obtained the better performance of desired behavior than the conventional real-valued Q-learning method, with roughly quantized state and action.

To show the validity of the method, we applied the method to a vision-guided mobile robot of which task is to chase the ball. Although the task was simple, the performance was quite impressive. Further improvement is discussed.

1 Introduction

Reinforcement learning has been receiving increased attention as a method with little or no a priori knowledge and higher capability of reactive and adaptive behaviors through such interactions [1]. Asada et al. have presented a series of works on soccer robot agents which chase and shoot a ball into the goal or pass it to another agent. In their reinforcement learning methods, the state and action spaces are quantized by the designer [2, 3] or constructed through the learning process [4, 5, 6] in order to make Q-learning, a most widely used reinforcement learning method [7], applicable. That is, well-defined and quantized state and action spaces are needed to apply Q-learning to real robot tasks. This causes two kinds of problems:

- Performance of robot behavior is not smooth, but jerky due to quantized action commands such as forward and left turn.
- State space construction which satisfies Markovian assumption is a new problem as noted in [4, 5, 6]

In this paper, we propose a continuous valued Q-learning for real robot applications. There were several related works so far. Boyan and Moore reported that the combination of dynamic programming and parameterized function approximation had shown poor performances even for benign cases [13]. Saito and Fukuda[11] and Sutton[12] proposed to use sparse-coarse-coded function approximator (CMAC) for Q-value estimation. However CMAC has its own problem of quantization and generally need a lot of learning data. This means their method takes long learning time.

On the other hand, the proposed method interpolates continuous values between roughly quantized states and actions. This contributes to realize smooth motions with much less computational resources.

To show the validity of the method, we applied the method for a vision-guided mobile robot of which task is to chase the ball. Although the task was simple, the performance was quite impressive.

The rest of this article is structured as follows: first, Q-learning is briefly described, then our method is explained. The method is applied to the domain of soccer robot, RoboCup [8], where a learning robot attempts to approach a ball. Finally, the real robot learning results are shown and a discussion is given.

2 An Overview of Q-Learning

Before getting into the details of our method, we will briefly review the basics of Q-learning, a most widely used reinforcement learning algorithm.

Q-learning is a form of model-free reinforcement learning based on stochastic dynamic programming.

It provides robots with the capability of learning to act optimally in a Markovian environment. We briefly explain one-step Q-learning algorithm. We assume that the robot can discriminate the set $\boldsymbol{S}$ of distinct world states, and can take the set $\boldsymbol{A}$ of actions on the world.

1. Initialize $Q(s,a)$ to 0 for all state s and action a.
2. Perceives current state s.
3. Choose an action a according to action value function.
4. Carry out action a in the environment. Let the next state be s' and immediate reward be r.
5. Update action value function from s, a, s', and r,

   $$\begin{aligned} Q^{t+1}(s,a) &= (1-\alpha)Q^t(s,a) \\ &\quad +\alpha(r+\gamma \max_{a'\in\boldsymbol{A}} Q^t(s',a')) \end{aligned} \quad (1)$$

   where α is a learning rate parameter and γ is a fixed discount factor between 0 and 1.
6. Return to 2.

3 Continuous Valued Q-Learning

3.1 State and Action Representation

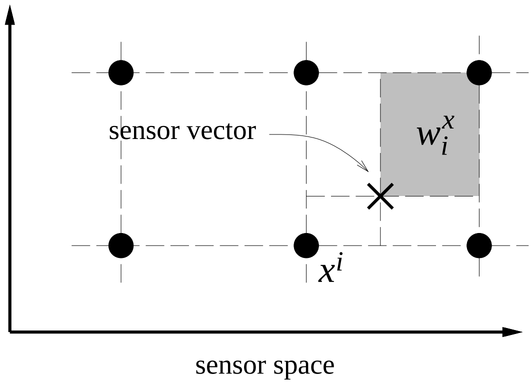


Figure 1: Calculation of contribution value w_i for the representative state s_i in the case of two-dimensional state space

The basic idea for continuous value representation of state, action, and reward in Q-learning is to describe them as contribution vectors of representative states, actions, and rewards. For the readers' understanding, here we assume that the n-dimensional sensory information directly construct the n-dimensional state space, and the motor space has m-dimensions. The robot perceives the current sensory information as a state vector $\boldsymbol{x} = (x_1, x_2, \cdots x_n)$, and executes motor command $\boldsymbol{u} = (u_1, u_2, \cdots u_m)$. The following explanation is for the state representation, but it is also the case for action representation.

First, we tessellate the state space into n-dimensional hyper cubes[1]. The vertices of all hyper cubes can be the representative state vectors $\boldsymbol{x}^i = (x_1^i, x_2^i, \cdots x_n^i)$ $i = 1, \cdots, N$ (here, N denotes the number of the vertices), and we call each vertex the representative state s_i. The contribution value $w_i^{\boldsymbol{x}}$ for each representative state s_i when the robot perceives the input $\boldsymbol{x} = (x_1, x_2, \cdots x_n)$ is defined as follows:

1. Specify a hyper cube including the input $\boldsymbol{x} = (x_1, x_2, \cdots x_n)$.
2. Tessellate the cube into 2^n hyper boxes based on the input $\boldsymbol{x}$ (see Fig.1 for the two dimensional case)
3. Calculate the volume of each hyper box.
4. Assign the volume $w_i^{\boldsymbol{x}}$ of the box diagonal to the state s_i.
5. If the input $\boldsymbol{x}$ is on the surface of the hyper cube, the volume can be reduced to the area or the length.
6. Any other contribution values for the states which do not compose the above cube are all zeros.

Mathematical formulation of the above process is given by

$$w_i^{\boldsymbol{x}} = \prod_{k=1}^{n} l_i(x_k), \quad (2)$$

where

$$l_i(x_k) = \begin{cases} 1 - |x_k^i - x_k| & (\text{if } |x_k^i - x_k| \le 1) \\ 0 & (\text{else}) \end{cases} \quad (3)$$

Fig.1 shows the case of two-dimensional sensor space. The area $w_i^{\boldsymbol{x}}$ is assigned as a contribution value for state s_i. The summation of contribution values $w_i^{\boldsymbol{x}}$ for the input $\boldsymbol{x}$ is one, that is,

$$\sum_{i=1}^{N} w_i^{\boldsymbol{x}} = 1 \quad (4)$$

Thus, the state representation corresponding to the input $\boldsymbol{x}$ is given by a state contribution vector $\boldsymbol{w}^{\boldsymbol{x}} =$

[1] the unit length is determined by normalizing the length of each axis appropriately

$(w^{\boldsymbol{x}}_1, \cdots, w^{\boldsymbol{x}}_N)$. Similarly, the action representation corresponding to the output $\boldsymbol{u}$ is given by an action contribution vector $\boldsymbol{w}^{\boldsymbol{u}} = (w^{\boldsymbol{u}}_1, \cdots, w^{\boldsymbol{u}}_M)$, where M denotes the number of the representative actions a_j in the tessellated action space.

3.2 Modified Learning Algorithm

Since the state and the action are represented by the definition mentioned above, we have to modify the standard Q-learning algorithm as follows:

The Q-value when executing the representative action a_j at the representative state s_i is denoted by $Q_{i,j}$. A Q-value at any state and action pair $(\boldsymbol{x}, \boldsymbol{u})$ is given by:

$$Q(\boldsymbol{x}, \boldsymbol{u}) = \sum_{i=1}^{N} \sum_{j=1}^{M} w^{\boldsymbol{x}}_i w^{\boldsymbol{u}}_j Q_{i,j}. \tag{5}$$

Given the representative state s_i, the optimal representative action is calculated by $\arg\max_j Q_{i,j}$. The optimal action contribution vector $\boldsymbol{w}^{\boldsymbol{x}*}$ for any state $\boldsymbol{x}$ is given by:

$$\boldsymbol{w}^{\boldsymbol{x}*} = \sum_{i=1}^{N} w^{\boldsymbol{x}}_i \boldsymbol{e}(\arg\max_j Q_{i,j}), \tag{6}$$

where $\boldsymbol{e}(k)$ denotes an M-dimensional vector of which k-th component is one and of which others are zeros.

Mapping from the optimal action contribution vector $\boldsymbol{w}^{\boldsymbol{x}*}$ to motor command $\boldsymbol{u}^*$ is given by:

$$\boldsymbol{u}^* = \sum_{j=1}^{M} w^{u^*}_j \boldsymbol{u}^j, \tag{7}$$

where $w^{u^*}_j$ denotes the j-th component of $\boldsymbol{w}^{\boldsymbol{u}*}$.

In order to obtain the optimal action based on eq.(6), $\max Q$ is calculated by:

$$\max Q = \sum_{i=1}^{N} \sum_{j=1}^{M} w^{\boldsymbol{x}}_i w^{\boldsymbol{u}^*}_j Q_{i,j} \tag{8}$$

Then, the action value when choosing an action $\boldsymbol{u}$ at the current state $\boldsymbol{x}$, and transiting the next state $\boldsymbol{x}'$ given reward r is updated by:

$$Q^{t+1}_{i,j} = (1 - \alpha w^{\boldsymbol{x}}_i w^{\boldsymbol{u}}_j) Q^t_{i,j} + \alpha w^{\boldsymbol{x}}_i w^{\boldsymbol{u}}_j (r + \gamma \max Q^{t'}), \tag{9}$$

where $\max Q^{t'}$ denotes Q value when choosing the optimal action at the next state.

3.3 Time to Update Action Values

One of the problems with the rough quantization of the state space is that one action does not always corresponds to one state transition. For example, in the case of a mobile robot with a TV camera mounted on it, one physical action (ex. forward motion command) may result in a small change in the camera image, which means the action is not sufficient to cause one state transition. The Q-value updating just after taking a physical actions without state transition causes an underestimate of Q-value for the state-action pair, and the learner cannot acquire any appropriate policy. Asada et.al. called this "state-action deviation problem"[2] and re-defined one action as a series of one kind physical action primitives which causes one state transition. That is, one physical action primitive is repeated until a state transition.

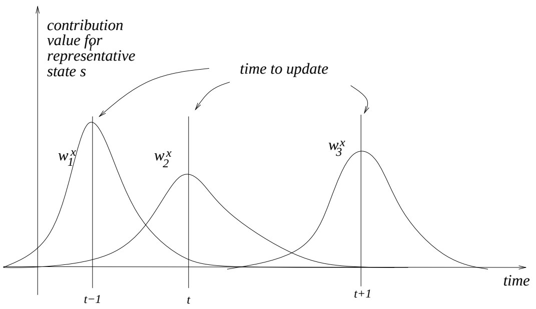


Figure 2: sampling time

To avoid this problem, we update Q value using eq.(9) at not every physical fixed time step, but sampled time step considering the time to update. We specify the time to update Q value as follows: The time to update is the moment when the contribution value of the representative state which has maximum value reaches the maximum (see Fig.2), and the moment when the situation doesn't change for a specific period. This manner has the same policy as that in [2], that is "once the state has changed, the learner update the action value function".

4 Experimental Results

4.1 Task of real mobile robot

In order to show the validity of the proposed method, we apply the method to a mobile robot of which task is to chase a ball, one of the vision-guided behavior acquisition. Fig.3 shows a picture in which the mobile robot we have designed and built and the ball to chase are shown. A simple color image processing (Hitachi IP5000) is applied to detect the ball area in the image in real-time (every 33ms).

The robot has a TV camera of which visual angles are 35 degs and 30 degs in horizontal and vertical directions, respectively. The camera is tilted down 23.5 degs to capture the ball image as large as possible.

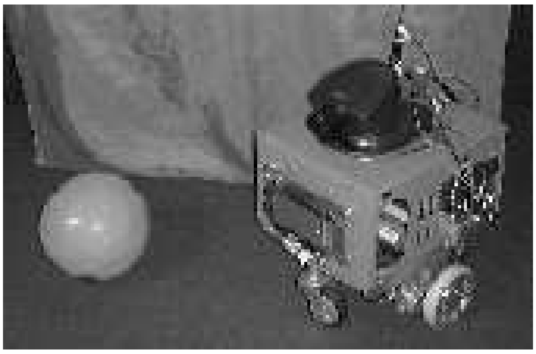


Figure 3: A mobile robot and a ball

The image area is 256 by 220 pixels, and the state space is constructed in terms of the centroid of the ball image. The driving mechanism is PWS (Power Wheeled System), and the action space is constructed in terms of two torque values to be sent to two motors corresponding to two wheels. These parameters of the robot system are unknown to the robot, and it tries to estimate the mapping from sensory information to appropriate motor commands by the method.

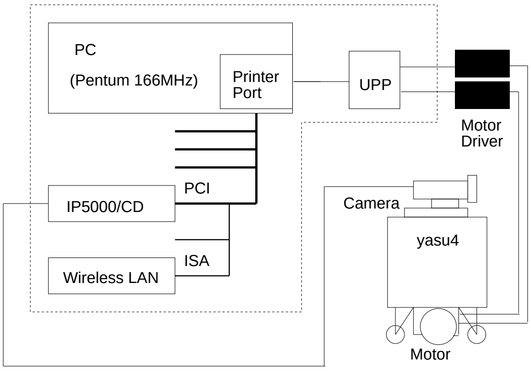


Figure 4: An overview of the robot system

The state and action spaces are two-dimensional ones, and tessellated into 9 by 9 and 5 by 5 grids, respectively. The learning rate α and the discount factor γ are 0.2 and 0.9, respectively. The parameters are constant during the learning.

Action selection during the learning was random and the goal state is a situation that the robot captures the ball region at the center of the image and its size is pre-specified value so that the ball is located just in front of the robot (about 50cm apart). In other words, the goal state is that the coordinates of the centroid of the ball region is (128,110).

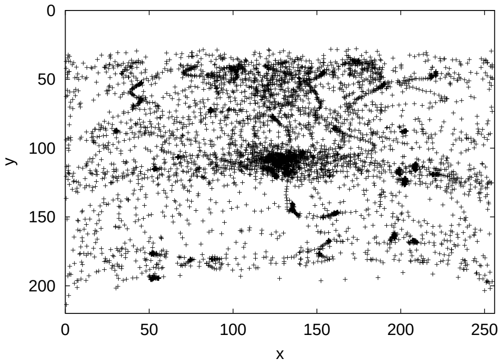


Figure 5: The distribution of the centroid of the ball image

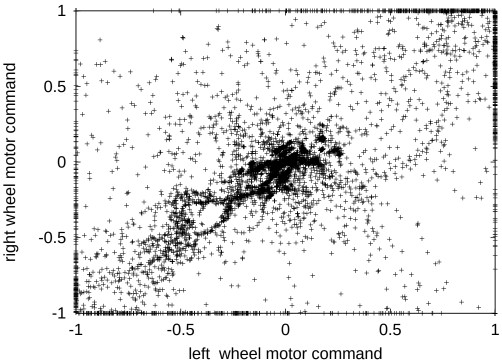
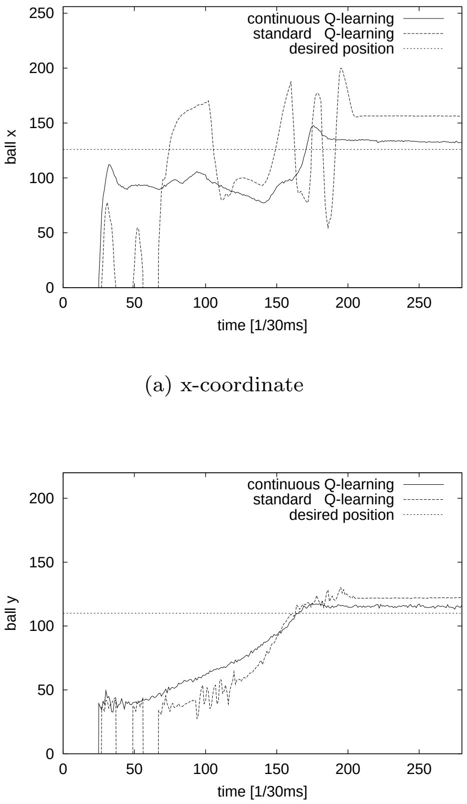


Figure 6: The distribution of the action data: the left (horizontal) and right (vertical) torques sent to the robot

To show the efficiency of the exploration and smoothness of the motion, we compare the results with standard Q-learning based on the quantized state and action spaces. We prepared the same data of experiences, and give them to the both methods. Fig.5 displays the distribution of the centroid of the ball image. Fig.6 shows the distribution of the action data corresponding to Fig.5.

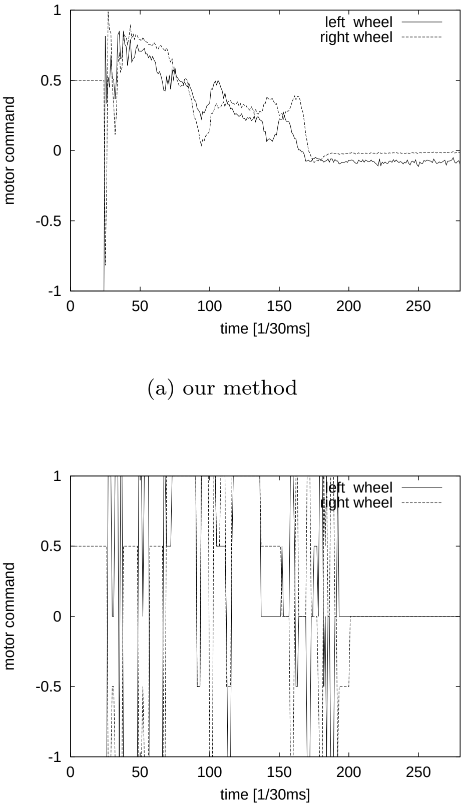
Next, we show the difference in the step responses of the robot behavior after the learning. The ball is positioned 3m far from the robot. Fig.7 shows the step responses, where the behavior based on the proposed method successfully reaches the goal state with smaller error and smoother motion than the standard Q-learning method. The standard Q-learning method sometimes lost the ball and oscillated left and right before reaching the goal state. Fig.8 indicates the motor commands for the step response. The vertical axis



(a) x-coordinate

(b) y-coordinate

Figure 7: Step responses with/without our method



(a) our method

(b) standard Q-learning

Figure 8: Motor commands for the step response

indicates the torque ratio for the left and right wheels, respectively. Fig.8 (a) shows the change of motor commands by our method, where the robot turns (the opposite torque ratio can be seen) for the first ten steps, then goes straightforward to the ball. While, Fig.8 (b) shows that the motor commands by the standard Q-learning oscillated.

Fig.9 shows a picture in which a sequence of action results is shown. First, the ball was at the top left corner, then it was pushed into the top right corner. We can see the robot approached the ball, and then turned right to chase the ball.

5 Discussion

In this paper, we have proposed a continuous valued Q-learning, which could obtain the much smoother behavior than the standard Q-learning. Rough quantization of the state and action spaces often causes a bad performance, or a divergence of the learned policy. On the other hand, fine quantization of the state space needs much computational resources, therefore a lot of learning data are required. This is one the most serious issues to apply reinforcement learning to real robot tasks. Many researchers have attacked this problem by constructing the state space through the experiences. However, our method could obtain the better performance with less computational resources.

We are now investigating the theoretical formulation of our approach and planning to apply more complicated tasks. As a future work, we will make some comparisons between the proposed method and the other ones which use parameterized function approximations or memory-based methods.

Acknowledgement

This research was supported by the Japan Society for the Promotion of Science, in Research for the Future Program titled Cooperative Distributed Vision for Dynamic Three Dimensional Scene Understanding

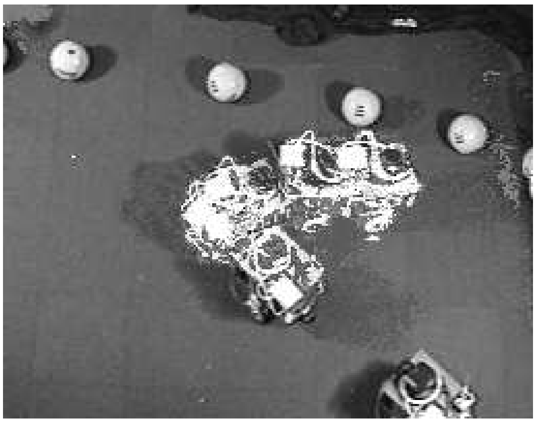


Figure 9: An image sequence of robot's chasing a ball

(JSPS-RFTF96P00501)[15].

References

[1] J. H. Connel and S. Mahadevan, editors. *Robot Learning.* Kluwer Academic Publishers, 1993.

[2] M. Asada, S. Noda, S. Tawaratumida, and K. Hosoda. Purposive behavior acquisition for a real robot by vision-based reinforcement learning. *Machine Learning*, 23:279–303, 1996.

[3] E. Uchibe, M. Asada, and K. Hosoda. Behavior coordination for a mobile robot using modular reinforcement learning. In *Proc. of IEEE/RSJ International Conference on Intelligent Robots and Systems 1996 (IROS96)*, pages 1329–1336, 1996.

[4] Minoru Asada, Shoichi Noda, and Koh Hosoda. Action based sensor space segmentation for soccer robot learning. *Applied Artificial Intelligence*, 12(2-3):149–164, 1998.

[5] Y. Takahashi, M. Asada, and K. Hosoda. Reasonable performance in less learning time by real robot based on incremental state space segmentation. In *Proc. of IEEE/RSJ International Conference on Intelligent Robots and Systems 1996 (IROS96)*, pages 1518–1524, 1996.

[6] E. Uchibe, M. Asada, and K. Hosoda. "state space construction for behavior acquisition in multi agent environments with vision and action". In *Proc. of ICCV 98*, pages 870–875, 1998.

[7] C. J. C. H. Watkins and P. Dayan. "Technical note: Q-learning". *Machine Learning*, 8:279–292, 1992.

[8] H. Kitano, M. Asada, Y. Kuniyoshi, I. Noda, E. Osawa, and H. Matsubara. "robocup: A challenge problem of ai". *AI Magazine*, 18:73–85, 1997.

[9] 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, 1975.

[10] 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, 1975.

[11] F. Saito and T. Fukuda. "Learning Architecture for Real Robotic Systems – Extension of connectioninst Q-Learning for Continuous Robot Control Domain". *Proceedings of IEEE International Conference on Robotics and Automation*, pages 27–32, 1994.

[12] R.S. Sutton. "Generalization in Reinforcement Learning: Successful Examples Using Sparse Coarse Coding". *Advances in Neural Information Processing Systems 8*, pages 1038-1044, 1996.

[13] J. Boyan and A. Moore. "Generalization in Reinforcement Learning: Safely Approximating the Value Function". *Proceedings of Neural Information Processing Systems 7*, 1995.

[14] S. D. Whitehead and D. H. Ballard. "Active perception and reinforcement learning". In *Proc. of Workshop on Machine Learning-1990*, pages 179–188, 1990.

[15] T. Matsuyama. "Cooperative Distributed Vision - Dynamic Integration of Visual Perception, Action, and Communication". In *Proc. of Image Understanding Workshop*, Monterey CA, 1998.