14

Complex-Valued Reinforcement Learning: A Context-based Approach for POMDPs

Takeshi Shibuya[1] and Tomoki Hamagami[2]
[1]*University of Tsukuba*
[2]*Yokohama National University*
[1,2]*Japan*

1. Introduction

Reinforcement learning (RL) algorithms are representative active learning algorithms that can be used to decide suitable actions on the basis of experience, simulations, and searches (Sutton & Barto, 1998; Kaelbling et al., 1998). The use of RL algorithms for the development of practical intelligent controllers for autonomous robots and multiagent systems has been investigated; such controllers help in realizing autonomous adaptability on the basis of the information obtained through experience. For example, in our previous studies on autonomous robot systems such as an intelligent wheelchair, we used RL algorithms for an agent in order to learn how to avoid obstacles and evolve cooperative behavior with other robots (Hamagami & Hirata, 2004; 2005). Furthermore, RL has been widely used to solve the elevator dispatching problem (Crites & Barto, 1996), air-conditioning management problem (Dalamagkidisa et al., 2007), process control problem (S.Syafiie et al., 2008), etc.
However, in most cases, RL algorithms have been successfully used only in ideal situations that are based on Markov decision processes (MDPs). MDP environments are controllable dynamic systems whose state transitions depend on the previous state and the action selected. On the other hand, because of the limited number of dimensions and/or low accuracy of the sensors used, real-world environments are considered to be partially observable MDPs (POMDPs). In a POMDP environment, the agent faces a serious problem called perceptual aliasing, i.e., the agent cannot distinguish multiple states from one another on the basis of perceptual inputs. Some representative approaches have been adopted to solve this problem (Mccallum, 1995; Wiering & Schmidhuber, 1996; Singh et al., 2003; Hamagami et al., 2002). The most direct representative approach involves the use of the memory of contexts called episodes to disambiguate the current state and to keep track of information about the previous state (Mccallum, 1995). The use of this memory-based approach can ensure high learning performance if the environment is stable and the agent has sufficient memory. However, since most real-world environments belong to the dynamic class, the memory of experience has to be revised frequently. Therefore, the revised algorithm often becomes complex and task-dependent.
Another approach for addressing perceptual aliasing involves treatment of the environment as a hierarchical structure (Wiering & Schmidhuber, 1996). In this case, the environment is divided into small sets without perceptual aliasing, so that the agent can individually learn each small set. This approach is effective when the agent knows how to divide the environment into sets with non-aliasing states. However, the agent must learn to divide the

environment accurately and to apply a suitable policy to each divided set. This process is time-consuming.
In this paper, we introduce a new RL algorithm developed by using a complex-valued function that represents each state-action value as a complex value. The most important advantage of using complex values in RL is to use time series information. This simple extension allows for compensation of the perceptual aliasing problem and ensures that mobile robots exhibit intelligent behavior in the real world. The complex-valued RL algorithm has two practical advantages. First, it can be easily combined with conventional simple algorithms such as Q-learning and Profit Sharing; this is because the proposed algorithm uses the same framework as do the aforementioned conventional algorithms. Second, the proposed algorithm does not memorize episodes of experience directly, i.e., the learning/control system does not require a large memory space.
In the remainder of this paper, we describe the basic idea of complex-valued reinforcement. In this study, we extend this idea to conventional RL algorithms such as Q-learning and Profit Sharing. We also perform simulation experiments to demonstrate the effectiveness of the method in environments involving perceptual aliasing problems.

2. Reinforcment learning with complex-valued function

2.1 Basic idea

Complex-valued neural networks (CVNNs) (Hirose, 2003) are extensions of conventional real-valued neural networks. In a CVNN, the inputs, outputs, and parameters such as weights and thresholds are expressed as complex numbers, and hence, the activation function is inevitably a complex-valued function. The advantage of such a network is that it allows one to represent the context dependence of information on the basis of both amplitude and phase structure. The amplitude corresponds to the energy, while the phase represents time lag and time lead.
The idea of complex-valued RL has been proposed on the basis of the abovementioned features of CVNNs. In other words, similar to the case of a CVNN, the complex values used in complex-valued reinforcement learning (CVRL) are expansions of the real values in used ordinary RL. One of the reasons for introducing complex values in RL is to compensate for perceptual aliasing by employing the phase representing the context in which the behavior is observed.
In CVRL, the value function or action-value function is defined as a complex value. Although we discuss only action-value functions in the following sections, the same idea may be applied to state-value functions as well. These complex values are revised by the proposed learning algorithm, whose framework is almost the same as that of a conventional algorithm. The main difference between the proposed algorithm and a conventional algorithm is that the former shifts the phase during the exploration of the environment.
The proposed RL algorithm does not select the best action but returns the most appropriate action for the given context. This properness is evaluated by using a complex-valued action-value function and an internal reference value that holds the context of the agent. The context is a series of observations and actions which the agent has obtained and taken. A complex-valued action value function is a function of state and action but not of time. The internal reference value is time-dependent and is updated step-by-step. Namely, our algorithm separates a time-dependent value function into two parts: a complex-valued action value function and an internal reference value.

In the following sections, we describe more specific algorithms that use the complex-valued function based on Q-learning and PS.

2.2 Basic implementations

2.2.1 Complex-valued Q-learning

Q-learning is a typical RL algorithm that learns the manner in which an environment state can be identified (C.J.C.H.Watkins, 1989). $Q(s,a)$ indicates the expected value of the reward when an agent selects action a at state s. In the learning phase, $Q(s,a)$ is revised as follows:

$$Q(s_t,a_t) \leftarrow (1-\alpha)Q(s_t,a_t) + \alpha(r + \gamma \max_{a'} Q(s_{t+1},a')) \tag{1}$$

where α denotes the learning rate; γ, the discount rate; and r, the reward.
The agent decides the next action on the basis of a function named policy. Policy is defined as a probability distribution over the action set for a given agent state. In this study, the Boltzmann policy, which gives probability of selecting action according to the Boltzmann distribution, is employed as a softmax policy.

$$\pi(s_t,a) = \frac{\exp(Q(s_t,a)/T)}{\sum_{a' \in \mathcal{A}(s_t)} \exp(Q(s_t,a')/T)} \tag{2}$$

where T denotes a scale parameter that controls randomness of the policy.
As mentioned above, the agent cannot observe its state directly in POMDPs. The agent can obtain the observation value x instead of the state s. Therefore, the value function is defined as a function of a pair of x and a.
The proposed algorithm named $\dot{\text{Q}}$-learning causes the action-value function to expand to a complex value as follows:

$$\begin{aligned} \dot{Q}(x_t,a_t) &\leftarrow (1-\alpha)\dot{Q}(x_t,a_t) + \alpha(r_{t+1} + \gamma \dot{Q}^{(t)}_{\max})\dot{\beta} && (3)\\ \dot{Q}^{(t)}_{\max} &= \dot{Q}(x_{t+1},a) && (4)\\ a &= \underset{a' \in \mathcal{A}(x_{t+1})}{\operatorname{argmax}} \left(Re\left[\dot{Q}(x_{t+1},a')\overline{\dot{I}_t}\right]\right) && (5) \end{aligned}$$

The *dot* mark ($\dot{}$) indicates that the value is complex. $\dot{\beta}$ is the degree of rotation of the phase. $\dot{Q}^{(t)}_{\max}$ is the most appropriate complex-valued action-value in the given context. $\overline{\dot{I}_t}$ indicates the complex conjugate of the internal reference value $\dot{I}_t$. In this study, we assume that the agent receives a positive reward if it reaches the terminal state.
We employ the following equation to determine the value of $\dot{I}_t$.

$$\dot{I}_t = \dot{Q}(x_t,a_t)/\dot{\beta} \quad t \geq 0 \tag{6}$$

(6) shows that $\dot{Q}$-value and the rotational amount give the internal reference value used in the next action selection. When this equation is employed, the phase difference between the internal reference value and the selected action value becomes zero in the MDP environment. For the first action selection, we cannot determine the internal reference value since there is no previously selected action value. Therefore, we employ the following heuristics to determine the internal reference value.

$$\begin{aligned} \dot{I}_{-1} &= \dot{Q}(x_0,a) && (7)\\ a &= \underset{a' \in \mathcal{A}(x_0)}{\operatorname{argmax}} \left|\dot{Q}(x_0,a')\right| && (8) \end{aligned}$$

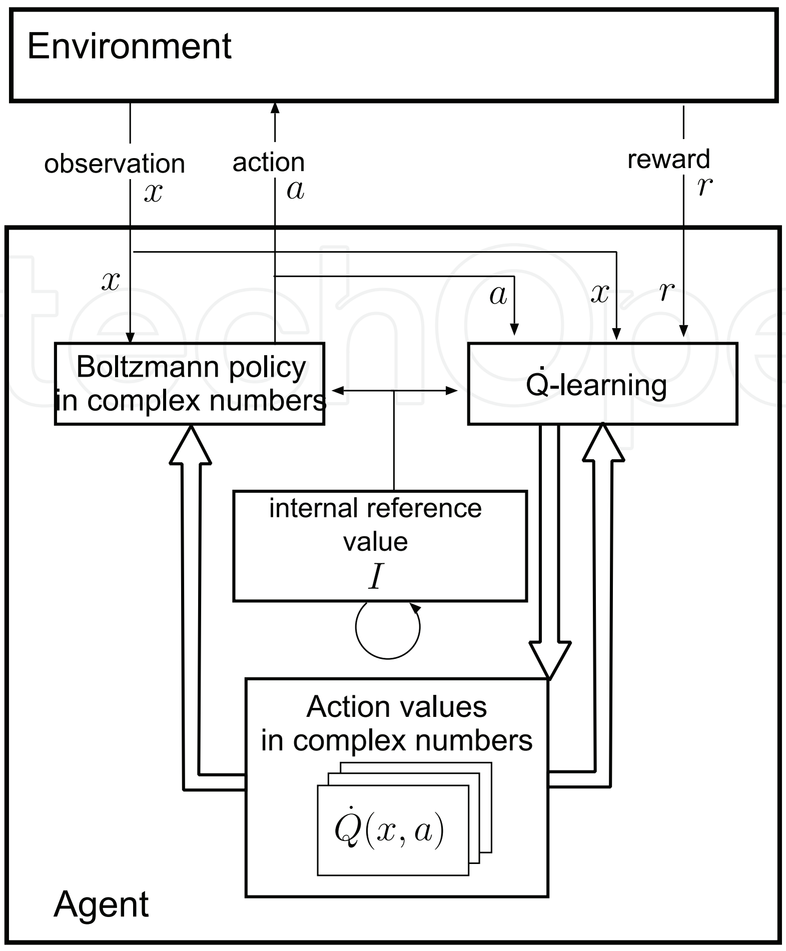

Fig. 1. Complex-valued reinforcement learning using Q̇-learning and Boltzmann policy.

Furthermore, in order to improve the learning efficiency, we use the concept of eligibility trace (A.G.Barto et al., 1983).

$$\dot{Q}(x_{t-k}, a_{t-k}) \leftarrow (1-\alpha)\dot{Q}(x_{t-k}, a_{t-k}) + \alpha(r_{t+1} + \gamma \dot{Q}_{\max}^{(t)})\dot{u}(k) \tag{9}$$

where $\dot{u}(k)$ denotes an eligibility parameter. In this study, we employ

$$\dot{u}(k) = \dot{\beta}^{k+1}. \tag{10}$$

where $k = 0, 1, \cdots$, and $N_e - 1$ is the index used for updating the action-values. N_e is the number of traces. Equation (9) is reduced to (3) if $N_e = 1$.

In order to select an action, the agent evaluates the action according to the stochastic policy, as follows:

$$\pi_{\dot{I}_{t-1}}(x_t, a) = \frac{\exp\left(Re\left[\dot{Q}(x_t, a)\overline{\dot{I}_{t-1}}\right]/T\right)}{\sum_{a' \in \mathcal{A}(x_t)} \exp\left(Re\left[\dot{Q}(x_t, a')\overline{\dot{I}_{t-1}}\right]/T\right)} \tag{11}$$

With this Boltzmann selection, there is a higher probability of the agent choosing the action corresponding to the $\dot{Q}$-value, which not only has a greater norm $|\dot{Q}|$ but is also closer in phase to $\dot{I}_{t-1}$. Figure 1 shows an overview of our algorithm.

2.2.2 Complex-valued profit sharing

based on the method of learning experience reinforcement. The basic idea of this learning is to share profits on the basis of the evaluation value for action selection $v(s,a)$ when the agent receives a reward and to reinforce the successful episodes.

$$v(s_t,a_t) \leftarrow v(s_t,a_t) + f_t \quad (t = W-1, W-2, \cdots, 0) \tag{12}$$

$$f_t = \begin{cases} r \quad (t = W-1) \\ \gamma f_{t+1} \quad (t = W-2, W-3, \cdots, 0) \end{cases} \tag{13}$$

where f_t denotes the reinforcement function; W, the length of the episode; and γ, the discount rate.
Then, simple complex-valued profit sharing(scPS) is defined as the extension of these functions, as follows:

$$\begin{aligned} \boldsymbol{v}(x_t,a_t) \quad \leftarrow \quad & \boldsymbol{v}(x_t,a_t) + \boldsymbol{f}_t \\ & (t = W-1, W-2, \cdots, 0) \end{aligned} \tag{14}$$

$$\boldsymbol{f}_t = \begin{cases} r \ (t = W-1) \\ \gamma e^{j\omega_t} \boldsymbol{f}_{t+1} \quad (t = W-2, W-3, \cdots, 0) \end{cases} \tag{15}$$

where W is the length of each episode. In this section, variables represented in bold face denote that they are complex.
Fig. 2 shows a comparison of the conventional function f_t and the complex-valued function $\boldsymbol{f}_t$. Similar to f_t, $\boldsymbol{f}_t$ is attenuated, and the plot of the complex-valued function is a spiral around the time axis. The continuity of the phase on the spiral is expected to represent a context in which the agent receives a reward. In other words, depending on whether the context phase is continuous or not, the agent can identify identical state-action pairs that would cause perceptual aliasing. After learning, the following probability provides the agent's action. We employ a roulette selection scheme $V(x_t,a) = \mathrm{Re}\left[\boldsymbol{v}(x_t,a)\bar{\boldsymbol{i}}(t)\right]$. Since in the roulette selection scheme, every weight of the action is considered to be positive, a subset of the action set is considered:

$$\mathcal{A}_{sub}(x_t) = \{a \in \mathcal{A}(x_t) | V(x_t,a) \geq 0\} \tag{16}$$

and the policy is defined as follows:

$$P(x_t,a) = \begin{cases} \dfrac{V(x_t,a)\delta(x_t,a)}{\sum_{a' \in \mathcal{A}_{sub}(x_t)} \left(V(x_t,a')\right)} & for \quad |\mathcal{A}_{sub}(x_t)| > 0 \\ \dfrac{1}{|\mathcal{A}(x_t)|} & \text{otherwise} \end{cases} \tag{17}$$

$$\delta(x,a) = \begin{cases} 1 \quad \text{if} \quad a \in \mathcal{A}_{sub}(x_t) \\ 0 \quad\quad \text{otherwise} \end{cases} \tag{18}$$

Equation (17) provides a roulette selection scheme in which actions s.t. $V(x,a) < 0$. If all the actions are ignored, the agent chooses an action with the equal probability.

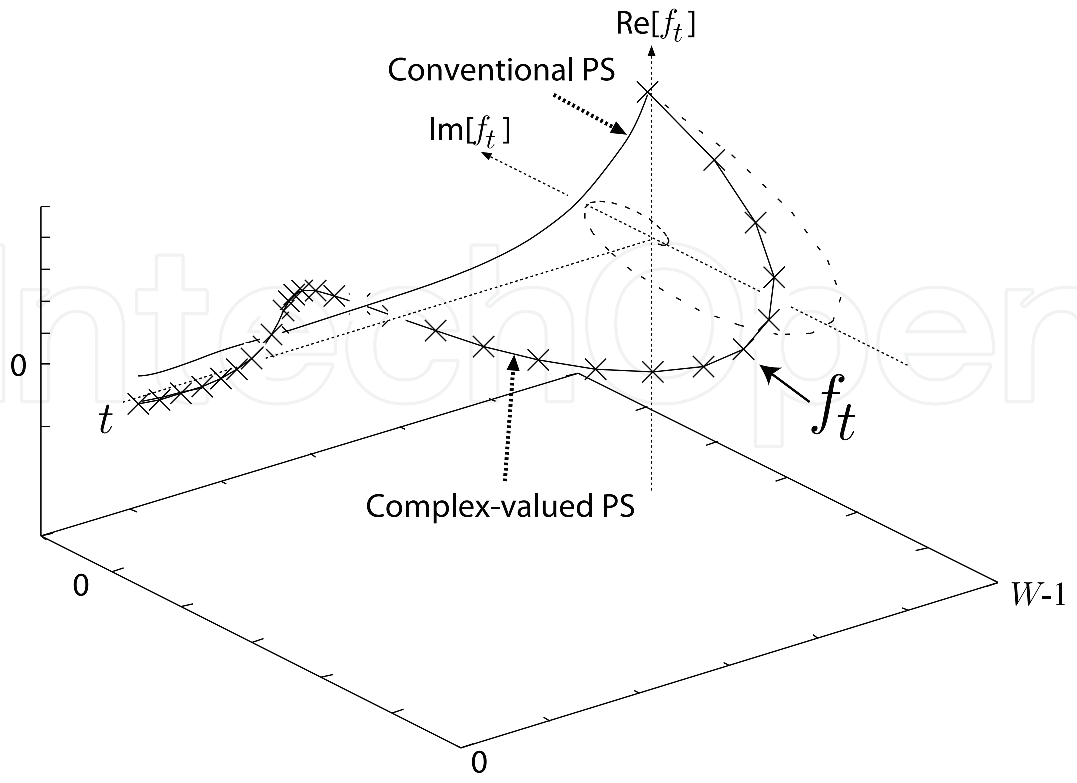

Fig. 2. An example of the complex-valued reinforcement function $\boldsymbol{f}$ for scPS.

$\boldsymbol{i}$ represents an internal reference value that is expressed as follows:

$$\boldsymbol{i}(t) = \begin{cases} \exp\left(j\theta_0\right) & (t = 0) \\ \exp\left(j\left(\theta_0 + \sum_{k=1}^{t}(-\omega_{k-1})\right)\right) & \\ & (t = 1, 2, \cdots, W-1) \end{cases} \tag{19}$$

where θ_0 represents the initial phase $\boldsymbol{i}(0)$.

2.3 Advanced implementations

2.3.1 Multiple action values

The aim of CVRL is to solve the perceptual aliasing problem by using the context of the agent. As described below, some of the experimental results obtained support the fact that CVRL can be successfully used to solve the perceptual aliasing problem. However, it is not possible for the agent to re-visit given a state many times and take the same action every time in that state. This is because that the internal reference value changes from the ideal value. Figure 3 illustrates this problem from the point of view of the action values. Figure 3(a) shows an example of the complex-valued action value corresponding to an observation. In Fig.3(b), the horizontal axis represents the phase of the internal reference value and the vertical axis represents effectiveness of the action values in the context. We assume that action a_1 is the suitable action. When the internal reference value is in area A or C, the probability of action a_1 being selected is high. However, as mentioned above, the internal reference value is so revised that the phase of the value varies along the horizontal axis in each step. In this case, the agent chooses the undesired action a_0 when the phase of the internal reference value is in area B.

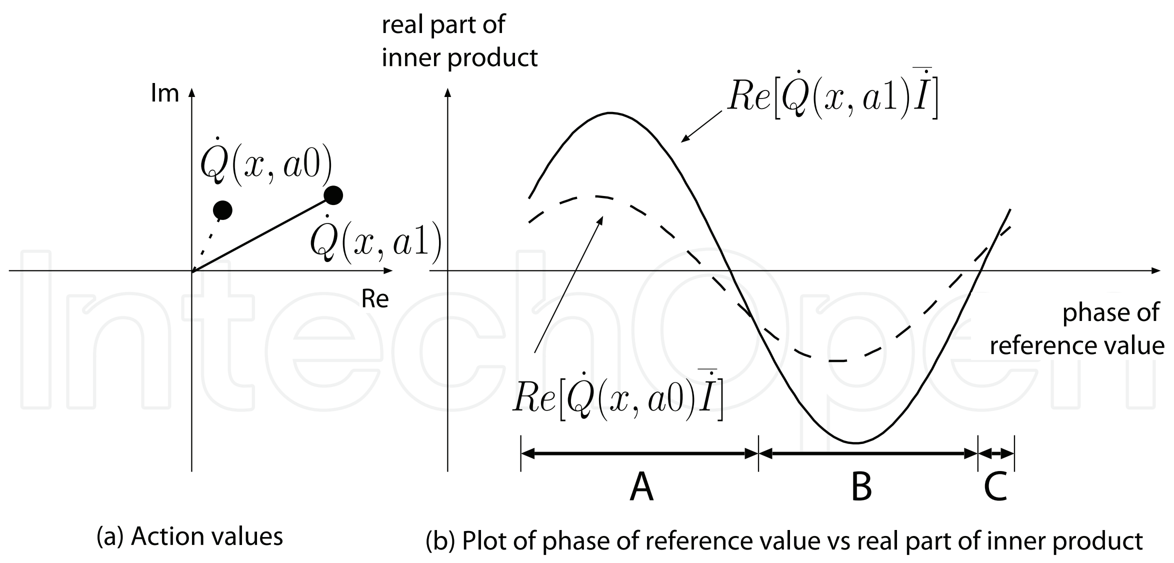
Im
$\dot{Q}(x, a0)$
$\dot{Q}(x, a1)$
Re
(a) Action values
real part of inner product
$Re[\dot{Q}(x, a1)\bar{\dot{I}}]$
$Re[\dot{Q}(x, a0)\bar{\dot{I}}]$
phase of reference value
A
B
C
(b) Plot of phase of reference value vs real part of inner product

Fig. 3. Relation between conventional complex-valued action values and internal reference value ($|\dot{I}|$ is normalized)

In order to resolve this problem, the use of multiple action values has been proposed (Shibuya & Hamagami, 2009). In this case, we assign multiple values to each action. Formally, we consider a virtual action set $\mathcal{A}^+(x)$ for the original action set $\mathcal{A}(x)$ and assume that there exists a corresponding action in $\mathcal{A}(x)$ for every action in $\mathcal{A}^+(x)$. We define action values for the virtual actions. The agent uses the virtual action set $\mathcal{A}^+(x)$ for computing the value function. Namely, the value function is defined on the $\mathcal{X} \times \mathcal{A}(x)$. The agent executes an original action with respect to the virtual action. Although $\mathcal{A}^+(x)$ can be learnt from the agent-environment interaction, we set $\mathcal{A}^+(x)$ by using a priori knowledge. In this study, we multiply the action values by an experimental parameter named *multiple degree*. For example, when we use 2 as the multiple degree, the agent uses the virtual action set $\mathcal{A}^+(x) = \{a_{00}, a_{01}, a_{10}, a_{11}\}$ for the original action set $A(x) = \{a_0, a_1\}$. Figure 4 shows the case

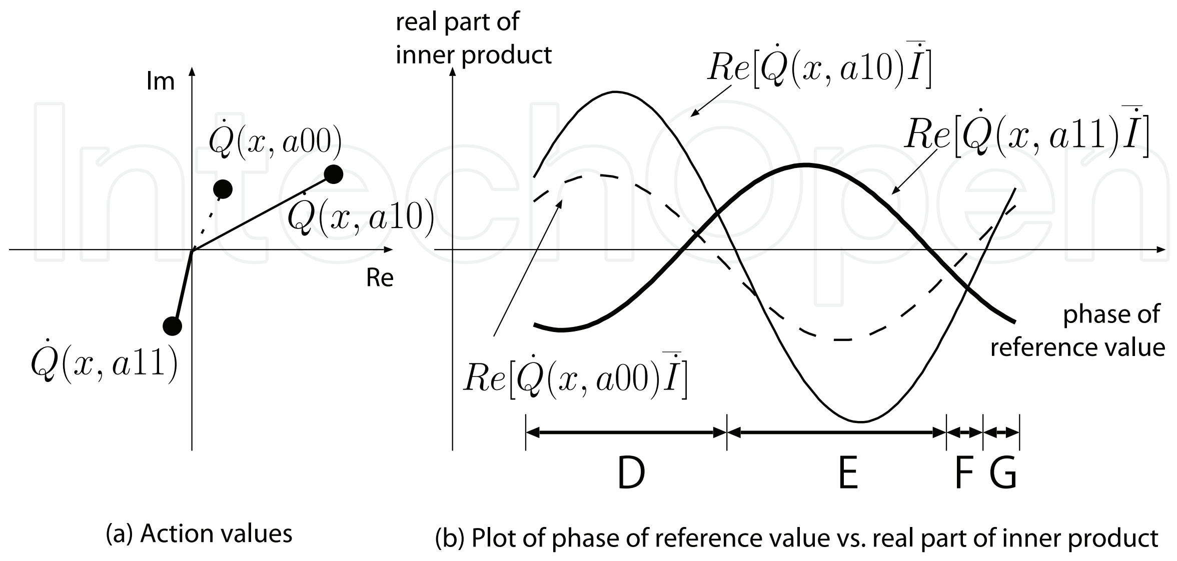
Im
$\dot{Q}(x, a00)$
$\dot{Q}(x, a10)$
Re
$\dot{Q}(x, a11)$
(a) Action values
real part of inner product
$Re[\dot{Q}(x, a10)\bar{\dot{I}}]$
$Re[\dot{Q}(x, a11)\bar{\dot{I}}]$
$Re[\dot{Q}(x, a00)\bar{\dot{I}}]$
phase of reference value
D
E
F
G
(b) Plot of phase of reference value vs. real part of inner product

Fig. 4. Relation between multiple complex-valued action values and the internal reference value ($|\dot{I}|$ is normalized)

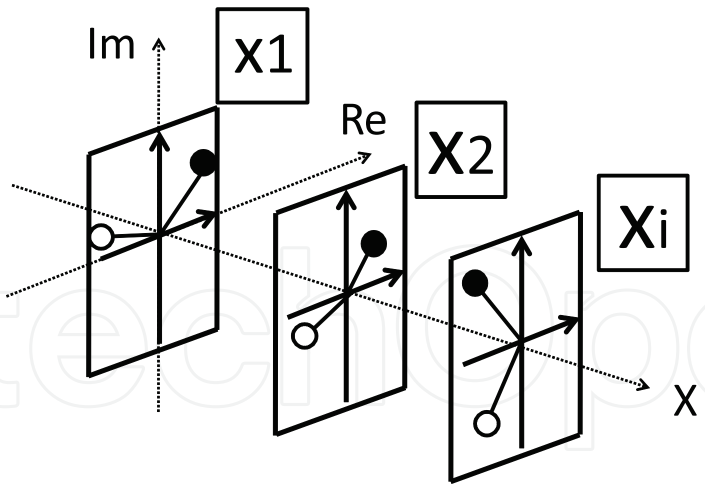

Fig. 5. Idea of conventional complex-valued reinforcement learning.

of that multiple action value is employed. In Fig. 4(b), horizontal and vertical axis is same as Fig.3(b). Action a_{10} is taken in the area D or G. Action a_{11} is taken in the area E. In this way, a method of multipe actoin value contributes to widen area where a_1 is taken.
An important advantage of using multiple action values is that it is not necessary to update the method of obtaining the action values, and hence, only virtual action values need to be determined.

2.3.2 Complex-valued RBF network for continuous environment

For using RL in real-world applications, it is important that continuous state spaces are efficiently handled. Fuchida et al. showed that in the case of (real-valued) RL, continuous state spaces can be efficiently handled without the need for mesh size parameters if the continuous action value function is approximated by using the radial basis function (RBF) (Fuchida et al., 2000; Samejima & T.Omori, 1999; Li & Duckett, 2005). In this section, approximation of the complex-valued action value function for continuous action spaces with perceptual aliasing is discussed, as shown in Fig.6. The aforementioned approximation method enables the treatment of the continuous state space without discretization in complex-valued RL.
In this section, we assume that an observation is a vector in the vector space $\mathbb{R}^M$. We employ

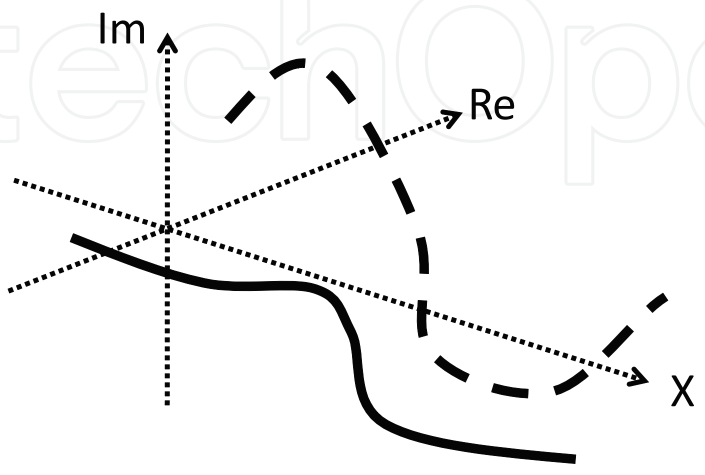

Fig. 6. Idea of complex-valued reinforcement learning using a complex-valued RBF network.

two RBF networks corresponding to two components of $\dot{Q}(\boldsymbol{x},a)$: a real part and an imaginary part. The complex-valued action value function is then represented as follows:

$$\dot{Q}(\boldsymbol{x},a) = Q_{Re}(\boldsymbol{x},a) + jQ_{Im}(\boldsymbol{x},a) \tag{20}$$

$$Q_{Re}(\boldsymbol{x},a) = \sum_{n=1}^{N_a} \omega_{Re,a,n}\phi_{Re,a,n}(\boldsymbol{x}) \tag{21}$$

$$\phi_{Re,a,n}(\boldsymbol{x}) = \prod_{m=1}^{M} \exp\left(-\frac{(x_m - \mu_{Re,a,n,m})^2}{\sigma^2_{Re,a,n,m}}\right) \tag{22}$$

$$Q_{Im}(\boldsymbol{x},a) = \sum_{n=1}^{N_a} \omega_{Im,a,n}\phi_{Im,a,n}(\boldsymbol{x}) \tag{23}$$

$$\phi_{Im,a,n}(\boldsymbol{x}) = \prod_{m=1}^{M} \exp\left(-\frac{(x_m - \mu_{Im,a,n,m})^2}{\sigma^2_{Im,a,n,m}}\right) \tag{24}$$

Note that n is the index of the RBF, and m is the dimension index; $\phi_{Re,a,n}$ and $\phi_{Im,a,n}$ are the RBFs. N_a is the number of RBFs in each component. $\omega_{Re,a,n}$ and $\omega_{Im,a,n}$ are the weight parameters. $\sigma_{Re,a,n}$ and $\sigma_{Im,a,n}$ are the variance parameters. $\mu_{Re,a,n,m}$ and $\mu_{Im,a,n,m}$ are the mean parameters. In this section, variables represented in bold face denote that they are vector.

$$\dot{\delta}_t = (r_{t+1} + \gamma\dot{Q}^{(t)}_{\max})\dot{\beta} - \dot{Q}(\boldsymbol{x}_t,a_t) \tag{25}$$

$$= \delta_{Re,t} + j\delta_{Im,t} \tag{26}$$

1. Real part

$$\omega_{Re,a,n} \leftarrow \omega_{Re,a,n} + \alpha_\omega\delta_{Re,t}\phi_{Re,a,n}(\boldsymbol{x}_t) \tag{27}$$

$$\sigma_{Re,a,n,m} \leftarrow \sigma_{Re,a,n,m} + \alpha_\sigma\delta_{Re,t}\omega_{Re,a,n} \times \frac{(x_{t,m} - \mu_{Re,a,n,m})^2}{\sigma^3_{Re,a,n,m}}\phi_{Re,a,n}(\boldsymbol{x}_t) \tag{28}$$

$$\mu_{Re,a,n,m} \leftarrow \mu_{Re,a,n,m} + \alpha_\mu\delta_{Re,t}\omega_{Re,a,n} \times \frac{x_{t,m} - \mu_{Re,a,n,m}}{\sigma_{Re,a,n,m}}\phi_{Re,a,n}(\boldsymbol{x}_t) \tag{29}$$

2. Imaginary part

$$\omega_{Im,a,n} \leftarrow \omega_{Im,a,n} + \alpha_\omega\delta_{Im,t}\phi_{Im,a,n}(\boldsymbol{x}_t) \tag{30}$$

$$\sigma_{Im,a,n,m} \leftarrow \sigma_{Im,a,n,m} + \alpha_\sigma\delta_{Im,t}\omega_{Im,a,n} \times \frac{(x_{t,m} - \mu_{Im,a,n,m})^2}{\sigma^3_{Im,a,n,m}}\phi_{Im,a,n}(\boldsymbol{x}_t) \tag{31}$$

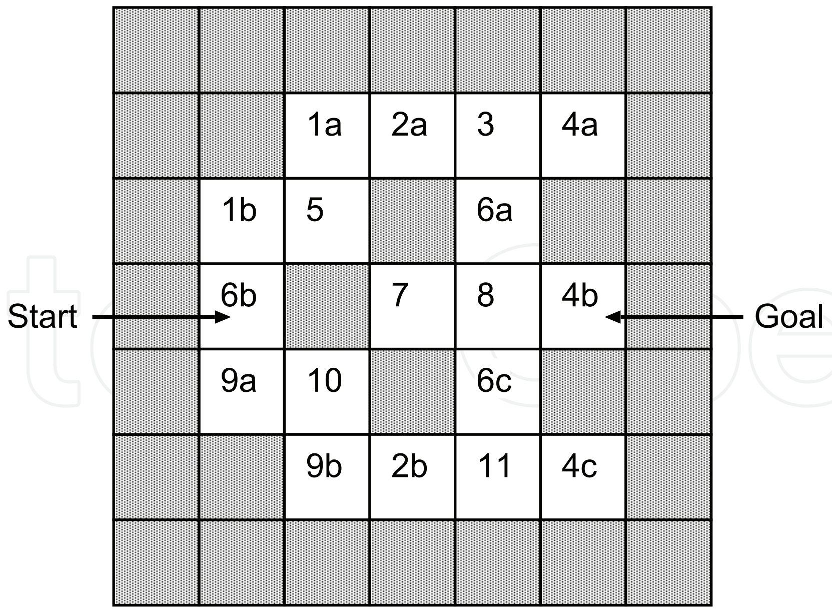

Fig. 7. Maze problems involving perceptual aliasings. Each cell is a state of the agent. The number on a cell indicates the observation. That is, the agent cannot distinguish between state 2a and 2b.

$$\begin{aligned} \mu_{Im,a,n,m} \leftarrow\ & \mu_{Im,a,n,m} \\ & + \alpha_\mu \delta_{Im,t} \omega_{Im,a,n} \\ & \times \frac{x_{t,m} - \mu_{Im,a,n,m}}{\sigma_{Im,a,n,m}} \phi_{Im,a,n}(\boldsymbol{x}_t) \end{aligned} \tag{32}$$

The method of updating the complex-valued RBF network is the same as that of updating the RBF network.

3. Experiments and results

Three simple experiments are conducted to evaluate the algorithms by using the proposed complex-valued functions. First, $\dot{\mathrm{Q}}$-learning is used to solve simple maze problems associated with perceptual aliasing Then, Miyazaki's environment (Miyazaki & Kobayashi, 2003) is learnt in the case of scPS. Finally, an experiment in a chained state environment which is useful type of environment for evaluating various intervals of perceptual aliasings is conducted for the multiple action values.

3.1 Maze problems

Figure 7 shows the maze environment involving a certain degree of perceptual aliasing. An agent starts from the cell marked "Start" and attempts to reach the cell marked "Goal." However, the agent cannot distinguish between some of the cells in the maze because the sensors detect only the existence of the walls around the agent. Thus, the agent faces a serious problem of having to change its action according to the context.

Q-learning, Q(λ)(Sutton & Barto, 1998), $\dot{\mathrm{Q}}$-learning ($N_e = 1$) and $\dot{\mathrm{Q}}$-learning ($N_e = 2$) are evaluated in the maze. The parameter conditions are shown in Table 1. The agent can choose an action from north, east, south, or west but cannot choose the action that makes it move

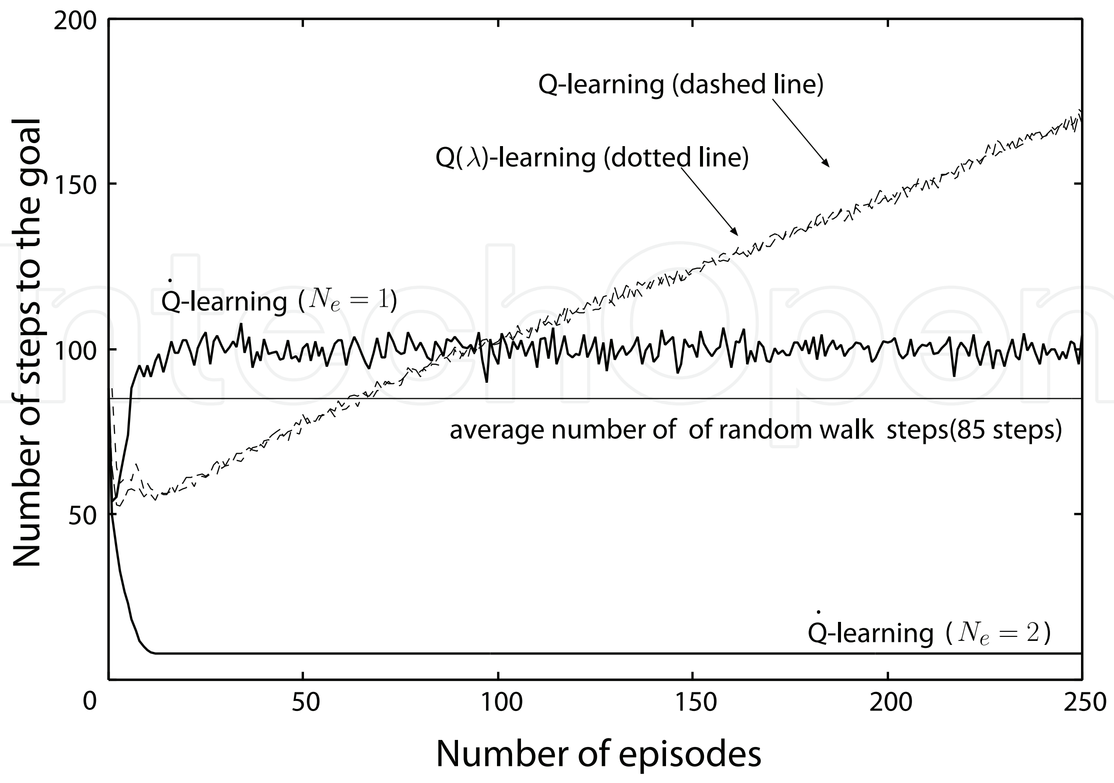

Fig. 8. Result in the POMDPs maze. This graph shows average steps over 1000 learning.

toward the walls. The agent obtains an *r* value of 100 if it achieves the goal. Each learning action consists of 100 episodes, and each episode consists of a series of steps from the start to the goal. If the agent cannot achieve the goal within 5000 steps, we restart the learning. Figure 8 shows the average number of steps in the POMDP environment shown in Fig.7. None of the algorithms are restarted in this case. In Q-learning and Q(λ), the average number of steps begins to increase after approximately 30 episodes because of the effect of perceptual aliasing. A greater number of steps are required in $\dot{\text{Q}}$-learning ($N_e = 1$) than in random walk. In contrast to these algorithms, the $\dot{\text{Q}}$-learning ($N_e = 2$) curve shows a sharp descent, and the number of learning steps converges to 12.

Q-learning, Q(λ)	
α	0.001 $\times$ (500 - episode)
γ	0.9
T	100 $\times$ (1.0/(1+episode))
λ	0.9 (for Q(λ))
$\dot{\text{Q}}$-learning	
α	0.25
$\dot{\beta}$	$\exp(j\pi/6)$
γ	0.9
T	20
N_e	1 (eligibility trace is not used)
	2 (eligibility trace is used)

Table 1. experimental parameters for the maze tasks.

Figure 9 shows the acquired typical action sequence from the point of view of using the context. Figure 9(a) shows the case of using Q-learning. The two state-action pairs in the initial state cannot reach the goal because of the perceptual aliasing between 6a, 6b, and 6c. For instance, if the agent has learnt to choose north in the initial state, the agent tries to choose north in the state 6a. However, the north action in this state 6a is not suitable. The agent has to learn this contradiction. Figure 9(b) shows the case of using $\dot{Q}$-learning . In contrast to the case of Q-learning, both the state-action pairs in the initial state in $\dot{Q}$-learning can reach the goal. Furthermore, we can see that the number of state-action pairs that can reach the goal is greater in $\dot{Q}$-learning than in Q-learning. Since $\dot{Q}$-learning employs the idea of context, action selection depends on the history of the state-action pair. Context-based action selection involves the use of the internal reference value shown in (6) and (7) and depends on the previous state-action pair only. Thus, the proposed method enables the agent to take a context-dependent action to achieve the goal even if several perceptual aliasing problems exist.

3.2 Miyazaki's environment

We compare scPS, PS-r*, and $\dot{Q}$-learning in Miyazaki's simulation experiment(Miyazaki & Kobayashi, 2003), as shown in Fig.10. The agent obtains the same observation Z in four states Z_a, Z_b, Z_c, Z_d. Thus, we conclude that the agent obtains identical observations in n states $S_1, S_2, \cdots, S_n$. In other words, n in the figure indicates the number of states that the agent can distinguish between. When the agent takes action a in state X, it reaches state S_1 with probability p or state X with probability $1 - p$. The agent repeatedly takes an action from the initial state X to the terminal state Z_d. The agent obtains the reward 100 iff the agent take action b at the state Z_d. We employ the parameters $n = 7$ and $p = 0.9$. We set the minimum number of steps from the start to the goal as 12.

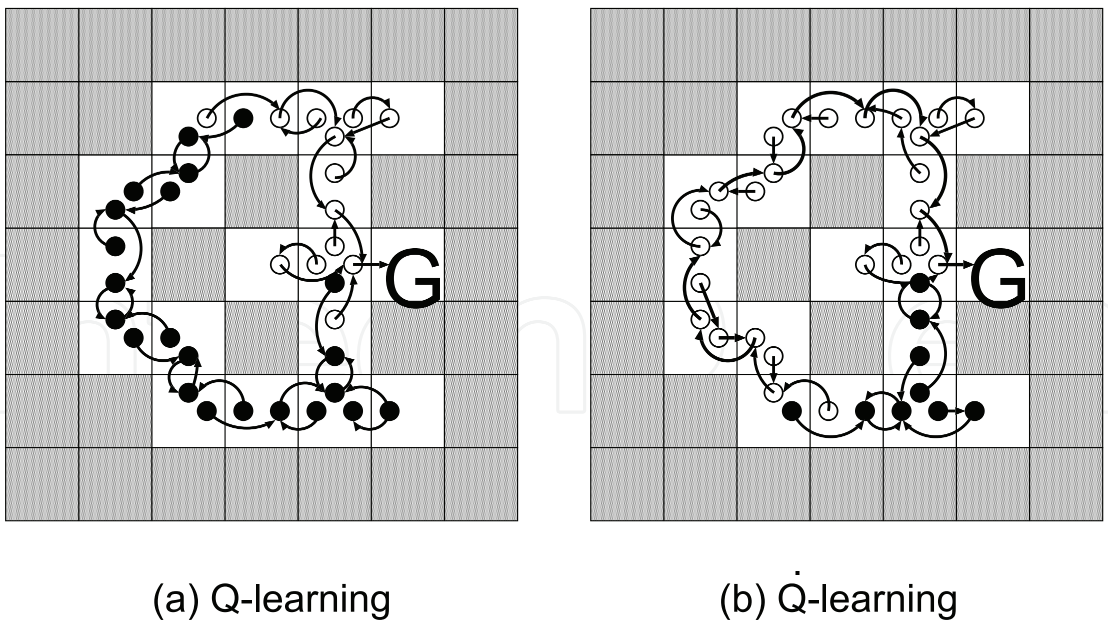

Fig. 9. Acquired action sequences. White and black bullets show the state-action pairs that can or cannot reach the goal, respectively. The location of a bullet is directly indicative of the corresponding action. For example, a bullet located in the upper region of a cell indicates the state-action pair of the cell and north action.

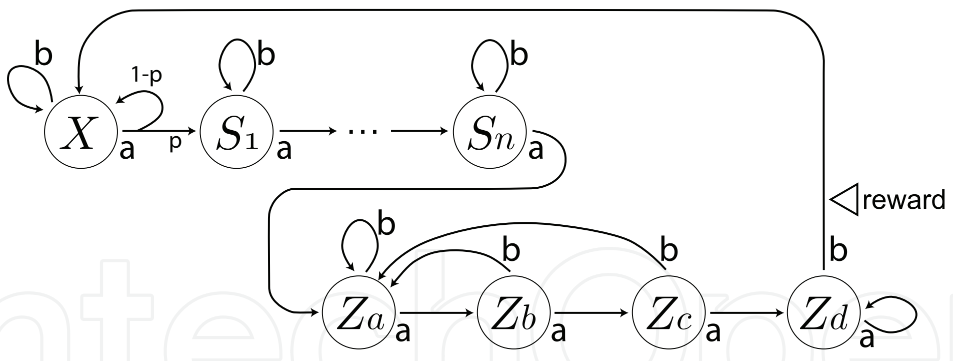

Fig. 10. Miyazaki's simulation environment (Miyazaki & Kobayashi, 2003).

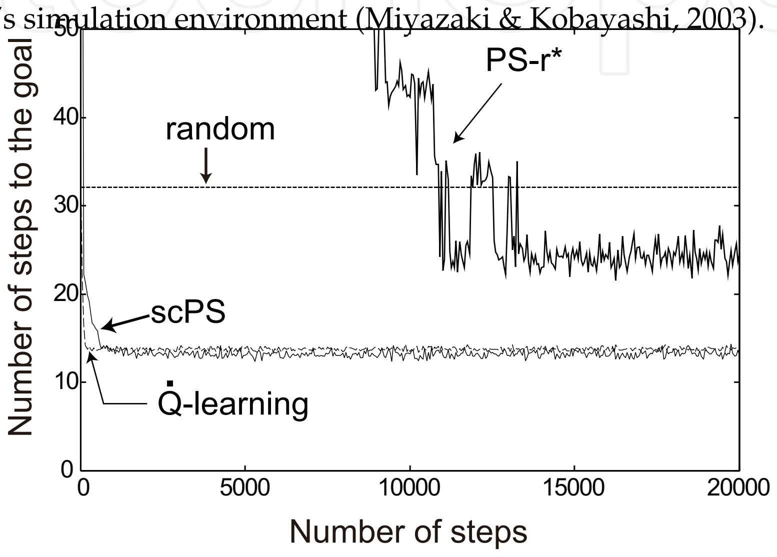

Fig. 11. Learning performance in Miyazaki's environment. The horizontal axis indicates the number of action selection. The vertical axis indicates the average number of steps if the agent uses the policy in each step.

We perform a total of 50 experiments, each of which consists of 20000 action selection steps. The experimental parameters are shown in Table 2.

scPS	
initial evaluation values	1.0×10^{-10}
discount factor γ	0.5
phase change ω[deg]	30
PS-r*	
initial evaluation values	0.8
significance level α	0.05
$\dot{Q}$-learning	
initial action values	0.0
learning rate α	0.8
phase rotation $\dot{\beta}$	$e^{j\pi/6}$
discount factor γ	0.9
number of traces N_e	3

Table 2. Parameter setting for Miyazaki's experiment.

The results of our experiments are shown in Fig.11. From the results, scPS a acquires suitable policy for a short term. Table 2 shows that state transition and series of selected actions acquired by the policy in the case of scPS. Between states X and S_7, the phase of the selected complex-valued evaluation value rotates by 30 degree. The phases of the selected evaluation values are almost equal to the phase of the internal reference value at a given time. From Z_a to Z_c, the selected evaluation values remain unchanged since the same evaluation value is selected. However, the internal reference value changes continuously. This causes the agent to take another action in state Z_d. In this experiment, the agent learns $V(Z,a) > V(Z,b)$ for $Z = Z_a, Z_b, Z_c$ and $V(Z,a) < V(Z,b)$ for $Z = Z_d$.

3.3 Chained state problems

Figure 12 shows an example of a simulation environment performed to evaluate the multiple action values. In Fig. 12, $s_0, s_1, \cdots, s_7$ denote the states of the agent. States s_0 and s_7 are the initial state and terminal state, respectively. The agent's objective is to learn to go from the start s_0 to the goal s_7. a_0 and a_1 denote the actions. Note that the agent is required to take action a_1 in all the states in order to achieve the goal s_7.
The agent obtains a reward r if it achieves the goal s_7. We assume that the state transition and observation are both deterministic. Additionally, we assume the state space and observation space to be both finite and countable. We compare the performance of $\dot{Q}$-learning with and without the multiple complex-valued action value. The multiplexing degree is employed is 2. In each experiment, we consider all possible types of aliasing. Namely, each experiment comprises a series of sub-experiments ranging from no aliasing (the agent can distinguish all the states.) to full aliasing (the agent is confused and cannot distinguish between any of

x	a	arg$\boldsymbol{v}$ [deg]	arg$\boldsymbol{i}$ [deg]
X	a	337.5	337.5
S_1	a	309.5	307.5
S_2	a	279.5	277.5
S_3	a	249.5	247.5
S_4	a	219.6	217.5
S_5	a	189.4	187.5
S_6	a	159.4	157.5
S_7	a	128.6	127.5
Z_a	a	50.2	97.5
Z_b	a	50.2	67.5
Z_c	a	50.2	37.5
Z_d	b	0.7	7.5

Table 3. Comparison between the phase of complex-valued evaluation values and the phase of the internal reference value.

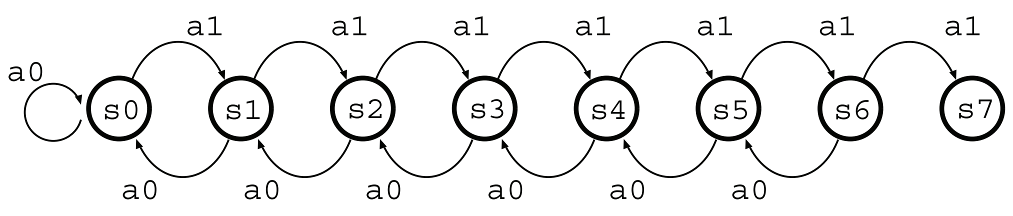

Fig. 12. An eight-state environment.

the states). In order to identify a task, we define certain observation patterns. An observation pattern is a set of relationships between the states and the observations and is represented by a sequence of numbers from the observation of the initial state to the observation of the terminal state. The appearance of the same sequence of numbers is indicative of perceptual aliasing. We ignore the numerical values in the pattern and simply focus on whether the values are the "same or different." For instance, the observation pattern 0-1-2-3-4-5-6-7 indicates that there is no perceptual aliasing, while 0-0-0-0-0-0-0-0 implies that the agent confuses between all the states; further, 0-1-2-3-4-0-5-6 implies that the agent confuses between s_0 and s_5. We perform five learning experiments in each observation pattern. Each learning action consists of 100 episodes, each of which consists of a series of steps from the start to the goal; here, each transition is called a step. The other experimental parameters are shown in Table 4. It is difficult for the $\dot{Q}$-learning agent to take the same action in several states where the obtained observation is the same. For example, we assume that the agent cannot distinguish between s_0 and s_5. In other words, the agent can distinguish between all the states except for the s_0/s_5 pair. The internal reference value in the s_0 stage differs from that in the s_5 stage because either (6) or (7) is used to move the internal reference value in the complex plane at each step. Because of this difference, the agent takes different actions. When the same action is suitable for the abovementioned two states, CVRL is effective. However, when the actions suitable for the two states are different, CVRL is ineffective.

Figure 13 shows the average learning curves in each environment. The number of steps in the conventional method converges to 15, while the number of steps in the multiple action value method converges to approximately 9.1. These results reveal that fewer steps are required in the case of $\dot{Q}$-learning with the multiple action value method than in the case of $\dot{Q}$-learning without the multiple action value method. We use the average number of steps in the final episode. We define the scores of the proposed method and conventional method as f_m and f_s, respectively. We evaluate the learning performance on the basis of the ratio f_m/f_s.

Figure 14 shows the experimental results. The vertical axis on the left indicates the final number of average steps f_m & f_s, while the vertical axis on the right indicates the ratio f_m/f_s. The horizontal axis indicates the tasks sorted on the basis of the ratio f_m/f_s. We group the observation patterns under three classes on the basis of the fm/fs value. Classes 1, 2, and 3 are sets of observation patterns for which $f_m/f_s < 1$, $f_m/f_s = 1$, and $f_m/f_s > 1$, respectively. We show the typical examples for each class in Table 5. Classes 1, 2, and 3 account for 32%, 59%, and 9.1% of the total number of observation patterns, respectively.

In class 1, the proposed method is superior to the conventional method. The observation patterns in this class are characterized on the basis of the time interval between the perceptual aliasing states. Every third or fourth state appearing in a given time interval is found to be a perceptual aliasing state. The results show that the use of multiple action values enables the agent to show a suitable behavior.

$\dot{Q}$-learning	
α	0.25
$\dot{\beta}$	$\exp(j\pi/6)$
γ	0.9
T	20
N_e	2
r	100.0

Table 4. Experimental parameters for chained state environment.

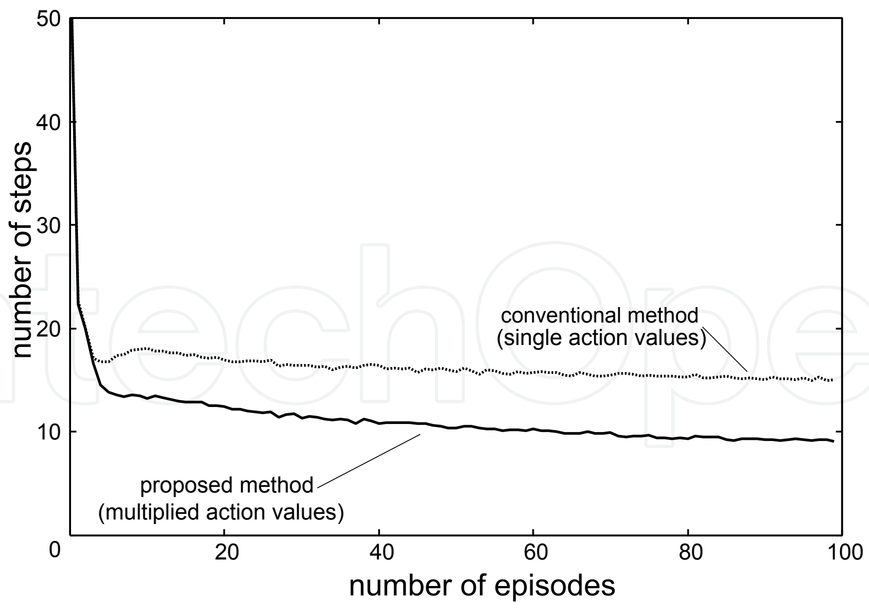

Fig. 13. Simulation results obtained for eight-state environment.

In class 2, the proposed method is as efficient as the conventional method. An observation pattern 0-1-2-3-4-5-6-7, which represents no perceptual aliasing, is included in this class. Another observation pattern 0-0-0-0-0-0-0, which represents that the agent confuses all the states, is also included in this class. Consequently, the performance of $\dot{Q}$-learning depends on the time intervals between the perceptual states and not on the number of perceptual states alone.

In class 3, the conventional method is superior to the proposed method. In the case of the conventional method, the observation patterns require a greater number of episodes to acquire a suitable behavior in class 3 than in class 2. We conducted an additional experiment with 500

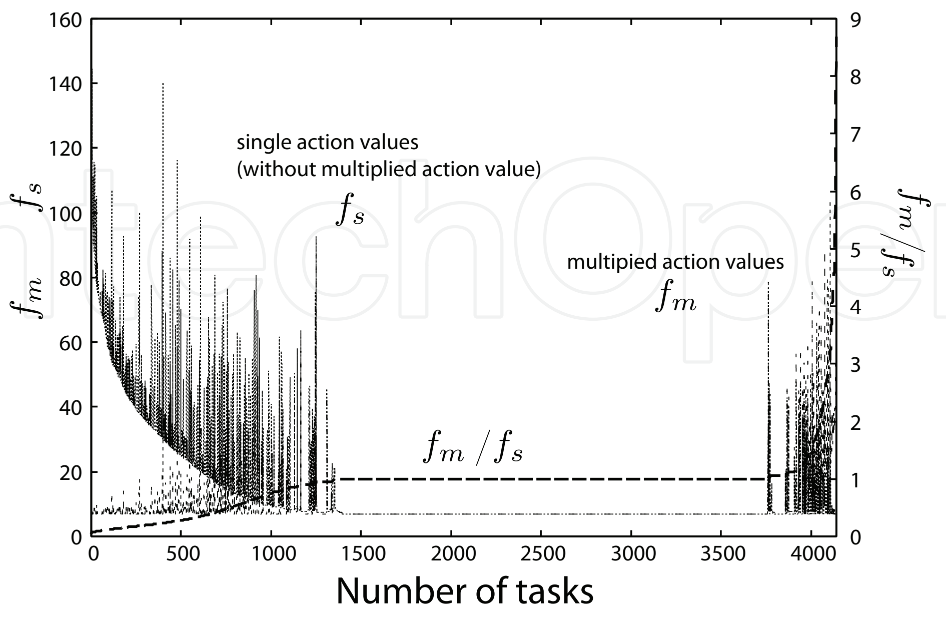

Fig. 14. Rules for interaction between complex-valued action values and internal reference values.

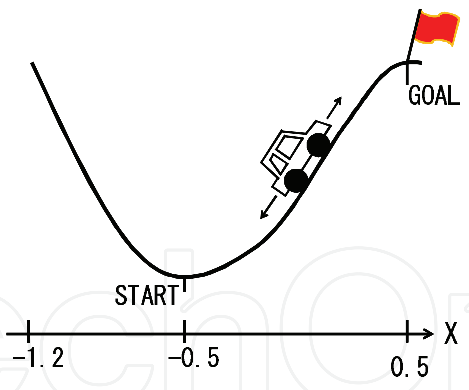

Fig. 15. Mountain car task.

episodes in the observation patterns of this class (shown in Table 5). The experimental results show that $f_m = 7.0$ in the three observation patterns. Therefore, we assume that the proposed method requires a greater number of episodes than does the conventional method; this is probably because of the greater number of action values in the former case.
As mentioned above, the proposed method can be used to improve the learning performance. On the other hand, it causes to increase the number of episodes.

3.4 Mountain car task with perceptual aliasing

We compare the method based on the complex-valued RBF network with three conventional methods, that is, random walk, dQ-learning without the RBF network, and Q-learning with the RBF network. We conduct a simulation experiment involving a mountain car task. The agent that uses $\dot{Q}$-learning without the RBF network divides the state space into 10 equal parts.

	observation pattern	f_{m}	f_{s}	$f_{\mathrm{m}}/f_{\mathrm{s}}$
	0-1-2-3-0-4-0-4	7.0	147.8	0.047
class 1	0-1-2-3-4-1-5-4	7.0	125.0	0.056
	0-1-2-3-4-0-5-5	7.0	118.6	0.059
	0-1-2-0-3-4-2-4	7.0	7.0	1.0
class 2	0-0-0-0-0-0-0-0	7.0	7.0	1.0
	0-1-2-3-4-5-6-7	7.0	7.0	1.0
	0-0-1-2-3-1-1-2	54.0	7.0	7.7
class 3	0-1-2-3-2-1-0-1	57.2	7.0	8.2
	0-1-2-1-0-0-0-3	60.8	7.0	8.7

Table 5. Comparison of performance of the proposed method and that of the conventional method.

discount rate γ	0.7
reward r	100
Boltzmann temperature T	150/(1+episode)
learning rate of μ α_μ	0.001
learning rate of σ α_σ	0.001
learning rate of ω α_ω	0.001

Table 6. Parameters for Q-learning with RBF network.

discount rate γ	0.7
reward r	100
Boltzmann temperature T	0.5
rotational value of phase $\dot{\beta}$	exp(1j[deg])
learning rate α	0.1
discreted state number D	10 at even intervals

Table 7. Parameters for $\dot{\text{Q}}$-learning

Figure 15 shows the mountain car task. The possible actions for an agent that controls the car are forward(a=+1), neutral(a=0) and reverse(a=-1). The agent aims to reach the goal from the start. However, the output power is not sufficiently high for the car to climb up the anterior mountain in one trip. Therefore, the car should climb a posterior mountain and climb the anterior mountain swiftly. The agent observes only the position of the car. The subsequent state is derived from the following equation:

$$v \leftarrow v + 0.001a - 0.0025\cos(3x) \tag{33}$$

$$x \leftarrow x + v \tag{34}$$

Note that $x(-1.2 \le x \le 0.5)$ and $v(-0.07 \le v \le 0.07)$ indicate the position and velocity of the car, respectively. The agent obtains a reward if and only if the car reaches the goal. Tables 6, 7 and 8 show the parameters for Q-learning with the RBF network, $\dot{\text{Q}}$-learning , and complex-valued RBF network, respectively. Each step corresponds to an action of the car, and each episode consists of a series of steps from the start to the goal. One learning action consists of 300 episodes.

discount rate γ	0.7
reward r	100
Boltzmann temperature T	0.5
rotational value of phase $\dot{\beta}$	exp(1j[deg])
learning rate of μ α_μ	0.001
learning rate of σ α_σ	0.001
learning rate of ω α_ω	0.01

Table 8. Parameters for complex-valued RBF network

Figure 16 shows the average number of steps for 20 learnings. The horizontal axis indicates the number of episodes, and the vertical axis indicates the number of steps from the start to the goal. Since Q-learning cannot be used to address perceptual aliasing, the results show that the learning behavior is better in the case of $\dot{\text{Q}}$-learning without the RBF network than in the case of Q-learning with the RBF network. The result also shows that the use of the complex-valued RBF network in these methods enables the agent to learn the best behavior quickly. When the RBF network is used, the learning efficiency is improved because the value function is extrapolated to the unsampled states.

4. Conclusion and future plans

A new reinforcement learning algorithm that uses complex-valued functions is proposed. This algorithm can be used to expand typical learning algorithms such as Q-learning and

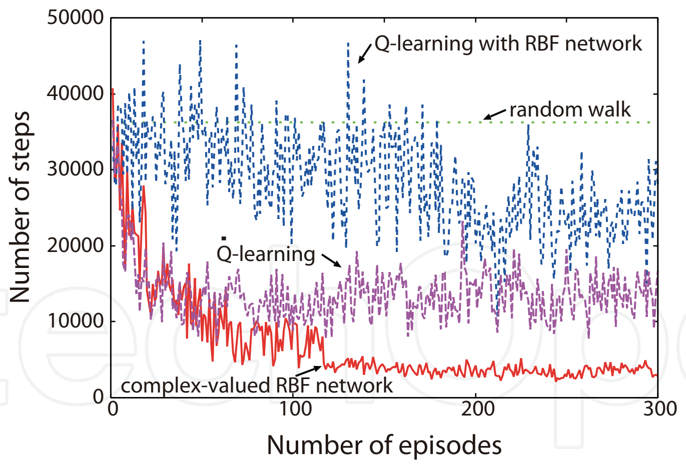

Fig. 16. Simulation result obtained for mountain car task with perceptual aliasing.

profit sharing and can allow for an agent to deal with time series or context. From the results of simulation experiments, we confirm that the algorithms generate redundant contexts to compensate for perceptual aliasing.
In the future, we plan to expand the proposed learning algorithm and compare the performance of this algorithm with that of other reinforcement learning algorithms. We also plan to make the following improvements to the proposed learning method:

– Implementation of the phase of the internal reference value as a time-varying function so that the method can be used in a dynamic environment
– Extension of the method to more complex and high-dimensional space
– Improvement of the method so that it can be applied to real autonomous robots in an uncertain environment

5. References

A.G.Barto, R.S.Sutton & C.W.Anderson (1983). Neuronlike elements that can solve difficult learning control problems, *IEEE Transactions on Systems, Man, and Cybernetics* 13: 835–846.

C.J.C.H.Watkins (1989). *Learning from Delayed Rewards*, PhD thesis, Cambridge University.

Crites, R. H. & Barto, A. G. (1996). Improving elevator performance using reinforcement learning, *Advances in Neural Information Processing Systems 8* pp. 1017–1023.

Dalamagkidisa, K., Kolokotsab, D., K.Kalaitzakisc & Stavrakakisc, G. (2007). Reinforcement learning for energy conservation and comfort in buildings, *Building and Environment* 42(7): 2686–2698.

Fuchida, T., Maehara, M., Mori, K. & Murashima, S. (2000). A learning method of RBF network using reinforcement leaning method(in Japanese), *IEICE technical report. Neurocomputing* 99(684): 157–163.

Hamagami, T. & Hirata, H. (2004). Development of intelligent wheelchair acquiring autonomous, cooperative, and collaborative behavior, *Proceedings of IEEE International Conference on Systems, Man and Cybernetics*, pp. 3235–3530.

Hamagami, T. & Hirata, H. (2005). State space partitioning and clustering with sensor alignment for autonomous robots, *Proceedings of IEEE International Conference on Systems, Man and Cybernetics*, pp. 2655–2660.

Hamagami, T., Koakutsu, S. & Hirata, H. (2002). Reinforcement learning to compensate for perceptual aliasing using dynamic additional parameter: Motivational value, *Proceedings of IEEE International Conference on Systems, Man and Cybernetics*, pp. 1–6.

Hirose, A. (ed.) (2003). *Complex-Valued Neural Networks : Theories and Applications*, Series on Innovative Intelligence, World Scientific Publishing Co. Pte. Ltd.

Kaelbling, L. P., Littman, M. L. & Cassandra, A. R. (1998). Planning and acting in partially observable stochastic domains, *Artificial Intelligence* 101: 99–134.

Li, J. & Duckett, T. (2005). Q-learning with a growing RBF network for behavior learning in mobile robotics, *Proceedings of the IASTED International Conference on Robotics and Applications*.

Mccallum, R. A. (1995). Instance-based utile distinctions for reinforcement learning with hidden state, *Proceedings of the 12th International Conference on Machine Learning*, pp. 387–395.

Miyazaki, K. & Kobayashi, S. (2003). An extention of profit sharing to partially observable markov decision processes : Proposition of ps-r* and its evaluation, *Transactions of the Japanese Society for Artificial Intelligence* 18(5): 286–296.

Samejima, K. & T.Omori (1999). Adaptive internal state space construction method for reinforcement learning of a real-world agent, *Neural Networks* 12: 1143–1155.

Shibuya, T. & Hamagami, T. (2009). Multiplied action values for complex-valued reinforcement learning, *Proceedings of the International Conference on Electrical Engineering*, pp. I9FP0491_1–6.

Singh, S., Littman, M. L., Jong, N. K., Pardoe, D. & Stone, P. (2003). Learning predictive state representations, *Proceedings of the 20th International Conference on Machine Learning*, pp. 712–719.

Syafiie, S., Tadeo, F., Martinez, E., Weber, C., Elshaw, M. & Mayer, N. M. (eds) (2008). Model-free learning control of chemical processes, *in Reinforcement Learning*, I-Tech Education and Publishing, Austria, chapter 16.

Sutton, R. S. & Barto, A. G. (1998). *REINFORCEMENT LEARNING: An Introduction*, MIT Press.

Wiering, M. & Schmidhuber, J. (1996). HQ-learning, *Adaptive Behavior* 6(2): 219–246.