LETTER

Dependency of Parameter Values in Reinforcement Learning for Navigation of a Mobile Robot on the Environment

Keiji Kamei and Masumi Ishikawa

Department of Brain Science and Engineering, Graduate School of Life Science and Systems Engineering
Kyushu Institute of Technology
Kitakyushu, Fukuoka, 808-0196, Japan
E-mail: kamei-keiji@edu.brain.kyutech.ac.jp

(Submitted on April 21 and July 22, 2006; Accepted on August 8, 2006)

Abstract—Reinforcement learning is suitable for navigation of a mobile robot due to its learning ability without supervised information. Reinforcement learning, however, has difficulties. One is its slow learning, and the other is the necessity of specifying its parameter values without prior information. We proposed to introduce sensory signals into reinforcement learning to improve its learning performance, and to optimize its parameter values in reinforcement learning by a genetic algorithm with inheritance. The latter has to specify the parameter values for every new environment, which is impractical due to huge computational time. In this paper, we propose to analyze the dependency and sensitivity of the values of parameters on the environment for predicting the values of parameters for a novel environment without optimization. Computer experiments clarify the dependency of the values of parameters on the environment and their sensitivities.

Keywords—reinforcement learning, genetic algorithm, navigation of a mobile robot, parameter dependency

1. Introduction

Reinforcement learning(RL)[1] has frequently been used for navigation of a mobile robot because of its effectiveness in obstacle avoidance and navigation[2]. However, RL is known to suffer from large computational cost. The conventional methods do not use sensory signals in RL, but some do. [3][4][5][6] use sensory signals only for localization, hence with no contribution to the acceleration of RL. Our proposal, on the contrary, introduces sensory information directly into RL, aiming at accelerating the learning speed of RL.

Another difficulty in RL is that we have to specify values of parameters such as a discount rate and a learning rate without prior information. We proposed to optimize the values of parameters in RL with the help of a genetic algorithm(GA) due to its ability in global search[7].

The previous studies which combine RL and GA are only for the improvement of a GA with the help of RL[8][9][10]. In contrast to these, Eriksson et al. proposed to improve the learning performance of RL by optimizing parameter values in RL with the help of a GA[11]. Our proposal is in line with this. The difference is the followings. We optimized seven parameters in RL including rewards and penalties in contrast to only two parameters of theirs[12]. Another difference is that we introduce inheritance into a GA to drastically reduce the computational cost.

The reduced computational cost for optimizing the values of parameters is still too large to compute them for every new environment[13][14]. Therefore, we propose to clarify the dependency and sensitivity of optimized parameter values in RL. This is for predicting appropriate values of parameters in RL for a novel environment based on a set of optimized parameter values for a small number of environments. For this purpose, we propose a hypothesis on the complexity measures of the environment in this paper.

In the subsequent section we explain RL and its revision. This is followed by a GA with inheritance. Section 4 presents our hypothesis on the complexity measures of the environment. Section 5 describes an autonomous mobile robot used here. Section 6 presents experimental results. Section 7 concludes the paper.

2. Reinforcement learning

We adopt the Q-learning, which is one of most popular reinforcement learning (RL) methods. Q-learning estimates a Q-value, $Q(s,a)$, as a function of a pair of a state and an action, which we think is suitable for a mobile robot. The conventional Q-learning iteratively updates a Q-value as,

$$Q(s,a) \leftarrow Q(s,a) + \alpha\left[r + \gamma \max_{a'} Q(s',a') - Q(s,a)\right] \tag{1}$$

where s is the current state, a is the corresponding action, s' is the next state, a' is the corresponding action, α is a learning rate, γ is a discount rate, and r is a reward or a penalty from the environment. A penalty, which is a negative reward, is also referred to as a reward for simplicity. The state s is composed of the location and orientation.

Major reason for slow learning in RL is that a mobile robot learns only at the current state based on a reward from the environment. To accelerate learning, we propose to directly reduce Q-values on a line segment between an obstacle and the current location of a mobile robot, in addition to the modification in Eq.(1).[3]. An additional idea is to suppress a detour and enable a mobile robot to pass through a narrow corridor by restricting an area of reduction of Q-values. The combined reduction of Q-values on the line segment is defined by,

$$\begin{aligned} Q(s'',a) &\leftarrow Q(s'',a) - P_o \cdot \frac{\|\mathbf{x} - \mathbf{x}_o\|}{\|\mathbf{x}_r - \mathbf{x}_o\|} \\ \mathbf{x} &= \mathbf{x}_o + \lambda(\mathbf{x}_r - \mathbf{x}_o), \quad 0 \le \lambda \le \frac{\theta}{\|\mathbf{x}_r - \mathbf{x}_o\|} \le 1 \end{aligned} \tag{2}$$

where P_o is the amount of penalty at the obstacle, $\mathbf{x}_r$ is the location of a mobile robot, $\mathbf{x}_o$ is the location of an obstacle, λ is the normalized distance from an obstacle, and θ represents the interval for the reduction of Q-values as in Figure 1. The amount of reduction of a Q-value is assumed to be linearly decreasing as the distance from the obstacle increases.

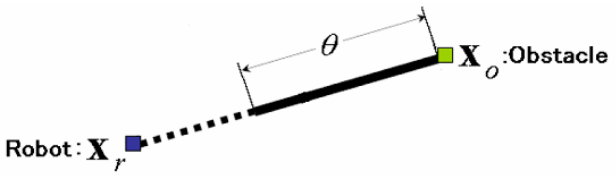


Figure 1. The bold line segment indicates the area of reduction of Q-values.

3. Genetic algorithm

A genetic algorithm (GA) is inspired by evolution of living things; search involves genetic operations such as selection, crossover and mutation in a probabilistic way frequently observed in the real evolutionary processes. Each individual has its chromosome and is evaluated by a fitness function.

RL has various parameters such as a learning rate, a discount rate and rewards. The values of parameters in RL are not known in advance. We optimize the values of parameters in RL with the help of a GA.

A chromosome is coded in binary, and its length is *42 bits*, with *6 bits* for each parameter. It is composed of a discount rate, a learning rate, a threshold for restricting the area of reduction of Q-values, and penalties for collision and detection of an obstacle, forward action and turning action. The reward for goal is set to 1.0 without loss of generality. A discount rate is coded in a logarithmic scale as,

$$\gamma = 1 - 10^{-kx}$$

where γ is a discount rate, x is an integer from 0 to 63, and k (= 0.1) is a scaling parameter. The reason for the logarithmic scale is to give a discount rate close to 1.0 sufficiently high resolution. All other parameters are coded in a linear scale.

In this paper, 50 individuals are generated initially, for each of which the fitness is evaluated. We then generate 25 new individuals in addition to the original 50 individuals. Out of 75 individuals, 50 individuals with higher fitness constitute the next generation. The value of fitness of each individual in the initial generation is evaluated by 500-episode learning. In later generations, a fitness function of a child individual is calculated by

the learning of 500-episodes starting from the final Q-values of the individual with the best matching chromosome in the previous generation and omission of RL for individuals with large fitness. This is what we call "inheritance", and is introduced here for drastically decreasing computational cost. Calculation of a fitness function of an individual requires computation of a few hours even if we use the inheritance. Therefore, computation of a GA here would not have been realized without the inheritance.

Figure 2 illustrates the procedure for optimization of parameters in RL by a GA with inheritance. The innermost loop is computation of RL. In the outer loops, computation of a GA over individuals and over generations is carried out, which provides optimized parameter values in reinforcement learning.

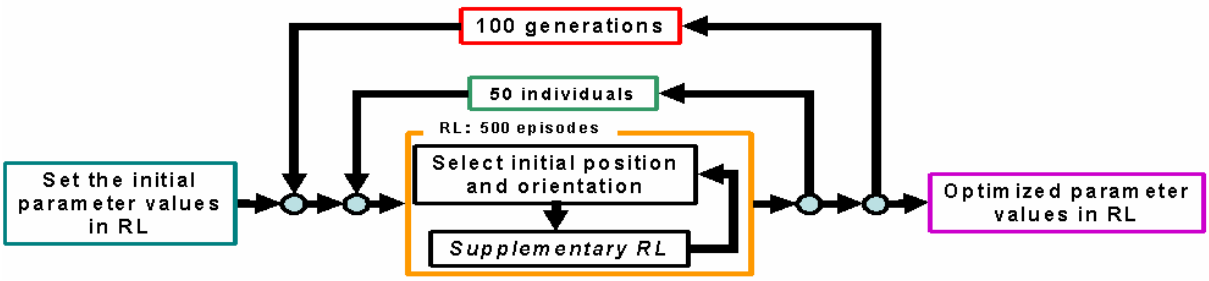


Figure 2. Procedure of the proposed GA with inheritance. The innermost loop is computation of RL, and in the outer loops computation of a GA over of individuals and generations is carried out.

In the innermost loop, RL is carried out for 500 episodes. One episode, here, is either a sequence of actions to a given goal, that to an obstacle, or that of 250 actions without getting to a goal or an obstacle. A fitness is calculated by the average over 500 episodes to diminish random fluctuations of initial states by the law of large numbers. The fitness is defined by

$$f = wg\frac{N_g}{N_E} + \left(1.0 - \frac{N_{axts}}{N_{\max}}\right)$$

where N_{acts} is the average number of actions in successful episodes, $N_{\max}$ is the upper bound for the number of actions in an episode, i.e., 250, N_E is the number of total episodes, i.e., 500, N_g is the number of successful episodes, and w_g is the goal weight.

3.1. Softmax action selection

We adopt the softmax function in selecting actions in RL[1]; the probability of selecting an action, $p(a)$, for the Q-value, $Q_t(a)$, is defined by,

$$p(a) = \frac{e^{Q_t(a)/\tau}}{\sum_{a=1}^{n} e^{Q_t(a)/\tau}}$$

where $\tau(t)$ is a temperature parameter defined by,

$$\tau(t) = \frac{B}{1 + At}$$

$$t = N_{gen}N_E + N_{epi}$$

where A (= 0.002) and B (= 0.1) are constant parameters, N_{gen} is a generation number, N_E is the upper bound for the number of actions in an episode, and N_{epi} is an episode number of an individual.

4. Dependency of optimized parameter values on the environment

We succeeded in optimizing parameter values in RL by a GA with inheritance for two kinds of environment in our previous studies[11][13][14]. Generally speaking, the optimized parameter values depend on the environment. Since it is quite time consuming to optimize such parameter values, it is almost meaningless to obtain them every time a new environment is given. In this paper, we clarify the relationship between parameter values and the type of the environment.

For a systematic analysis of the relationship, it is necessary to define measures representing the complexity of the environment. We propose a hypothesis on the complexity measures of the environment: the number of possible states and the ratio of the number of rotation actions to the number of total actions in an episode. The former assumes that the larger the number of possible states is, the more complex the environment becomes. The latter assumes that the larger the ratio is, the more complex the environment becomes. Based on this hypothesis, we design the four types of environment in Figure 3.

The area of the environment is *4m* x *4m*, and is composed of *20* x *20* grids, each with *20cm* x *20cm*. Figure 3(e) illustrates the complexity measures map for 4 types of environment. The upper-right corner of the figure is assumed to be the most complex under the above hypothesis.

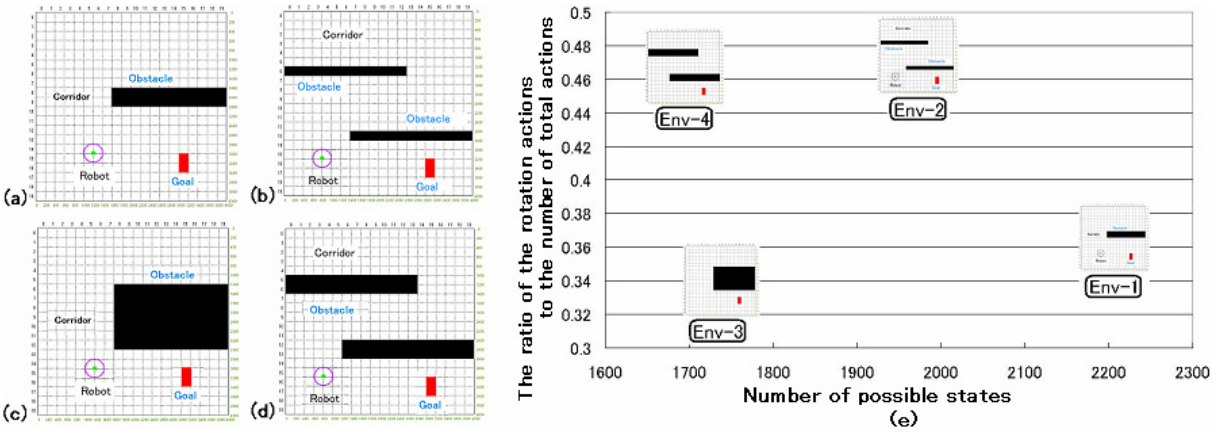


Figure 3. (a) Environment 1(abbreviated as Env-1); (b) Environment 2(Env-2); (c) Environment 3(Env-3); (d) Environment 4(Env-4); (e) Map of complexity measures of the environment. A black object stands for an object, and the red rectangle stands for the goal.

5. Experimental conditions

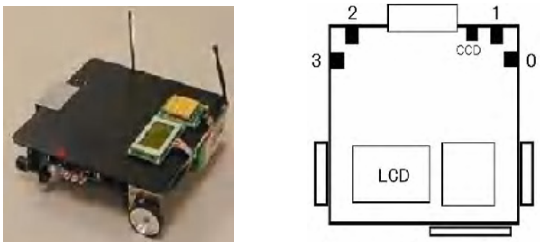
Figure 4(a) illustrates the mobile robot, ***TRIPTERS mini***, and Figure 4(b) depicts the positions of sensors. The mobile robot has 1 free wheel and 2 independent driving wheels. It cannot rotate on the spot, because the axle between the 2 driving wheels does not pass through the center of the robot.

In computer experiments in Section 6, we use 3 primitive actions, i.e., moving forward by 100*mm*, turning right by10°, and turning left by 10°. A multiple of 100*mm* or 10° can easily be realized by a sequence of the corresponding primitive. As these primitive values become smaller, the resulting path becomes more precise, but the computational cost increases. Taking this tradeoff into account, we adopt the above 3 primitive actions. The ***TRIPTERS mini*** has ultrasonic and infrared (IR) sensors. The ultrasonic sensors on ***TRIPTERS mini*** can measure the distance to an obstacle not exceeding 800 *mm*. In contrast to this, outputs of IR sensors on ***TRIPTERS mini*** are binary; the output is 1 if the distance is less than 700 *mm*, and 0 otherwise. We use only ultrasonic sensors here, because of its ability of measuring the distance.

The state of the mobile robot is defined by its location (one of *20* x *20* grids) and orientation (one of 8 sectors). An assumption adopted here is that the mobile robot knows its randomly selected state, i.e., location and orientation, at the start of each episode. It is also assumed that the mobile robot knows its state thereafter based on odometry information. Although the state of the mobile robot is discretized, its exact state is also preserved for their calculation at later steps. An episode terminates, provided a mobile robot reaches a goal, collides with obstacles, or the number of actions reaches the upper limit of 250.

6. Results of Computer Experiments

We optimize the values of parameters in RL by a GA with inheritance. We examine how optimized parameter values are affected by the goal weight in the fitness function. We also clarify the dependency of parameter values in RL on the environment.

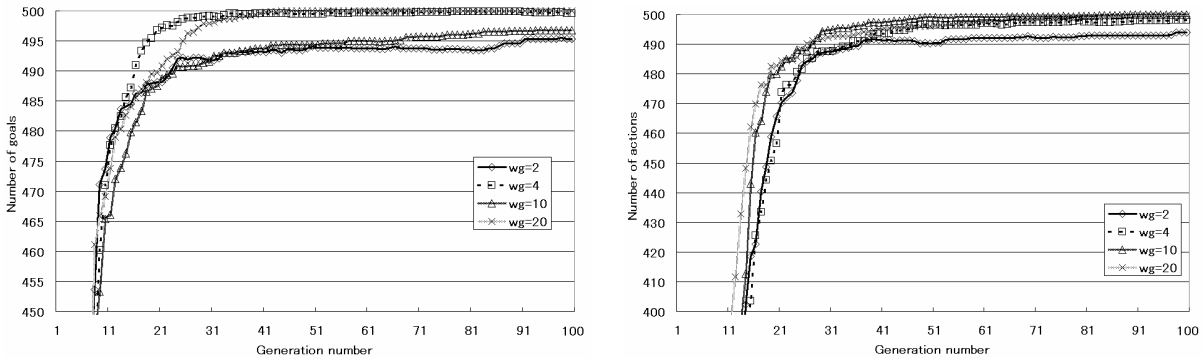


(a) (b)

Figure 4. (a) Overview of ***TRIPTERS mini***; (b) positions of sensors.

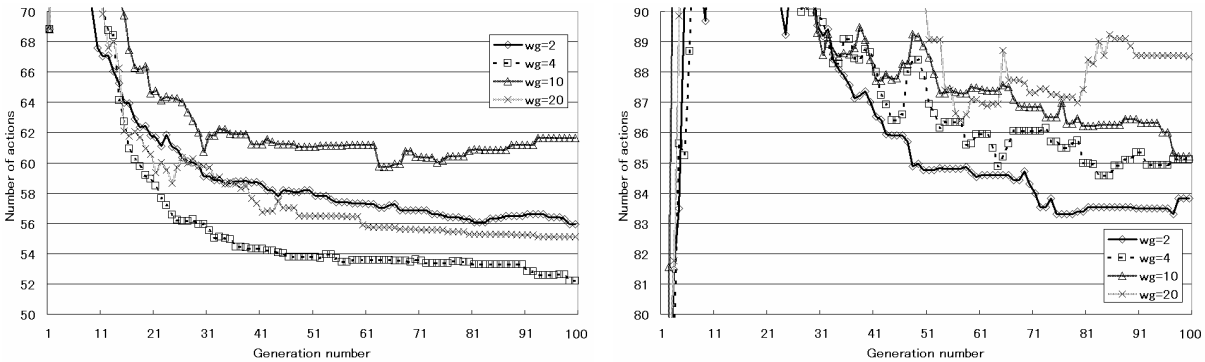
6.1. Determination of parameter values in RL by a GA with inheritance

Experiments are done for 4 different goal weights, i.e., 2, 4, 10, and 20. Figure 5 illustrates the number of goals reached averaged over top 10 individuals among a set of individuals in each generation in Env-1 and Env-2. It indicates that the number of goals reached is not monotonic with respect to the goal weight in Env-1. In contrast to this, the number of goals reached is monotonic with respect to the goal weight in Env-2. Figure 6 illustrates that the number of actions in an episode averaged over top 10 individuals in Env-1 and Env-2. The figure shows the maximum difference in the number of actions among different goal weights is about 10. We, therefore, examine how optimized parameter values are affected by goal weights in later experiments.



(a) (b)

Figure 5: The number of goals reached averaged over top 10 individuals. (a) Env-1 (b) Env-2. w_g stands for goal weight.



(a) (b)

Figure 6: The number of actions averaged over top 10 individuals. (a) Env-1 (b) Env-2. w_g stands for the goal weight.

The GA with inheritance tends to optimize the performance in the steady state. The best fit individual in this case experiences supplementary learning only bout 10 times which is equivalent to learn 5000 episodes. For this reason, to evaluate the performance in the steady state, we additional 10000-episode learning is carried out starting from the final Q-values of the best fit individual. Table 1 shows the results of additional learning. It shows that the number of goals reached and the number of actions do not differ much for different goal weights. As the result of this experiment, we use the goal weight of 10 in the subsequent experiments.

Table 1. Results of Additional Learning of 10000 Episodes (w_g stands for the goal weight)

		The goal weight, w_g			
		2	4	10	20
Number of goals reached	Env-1	249.05	249.08	249.55	248.68
	Env-2	248.40	248.75	248.65	248.33
Number of actions	Env-1	52.60	60.90	52.00	55.88
	Env-2	87.78	87.08	81.80	84.78

6.2. Dependency of parameter values in RL on the environment

In addition to Env-1 and Env-2, we create Env-3 and Env-4 to clarify the dependency of parameter values in RL on the environment. Table 2 shows the optimized parameter values. Generally speaking, optimized parameter values differ for different types of environment. A crucial issue is how the difference in optimized parameter values affects the performance in terms of goals reached and the number of actions. For this reason we carry out a sensitivity analysis for each parameter.

Table 3 shows the results of additional learning of 2000 episodes using the optimized values of parameters by a GA. Table 4 shows the results of additional learning of 2000 episodes under the perturbation of +20% or -20% from its standard value for each parameter. Excepting the discount rate, the perturbation does not affect much to the performance.

Table 2. Optimized Parameter Values in RL for 4 Types of Environment

	Reward for				Discount Rate	Learning Rate	Threshold for Sensors
	Forward	Rotation	Collision	Obstacle			
Env-1	-0.16	-1.43	-114.29	-65.08	0.999996	0.43	76.19
Env-2	-0.26	-1.59	-53.97	-20.63	0.999997	0.36	38.10
Env-3	-0.16	-1.43	-88.89	-3.17	0.999992	0.58	0.00
Env-4	-0.48	-4.76	-38.10	-57.14	0.996838	0.32	114.29

Table 3. Results of Additional Learning of 2000 Episodes with the Optimized Parameter Values by a GA.

	Env-1	Env-2	Env-3	Env-4
The number of goals reached	249.00	247.63	248.13	246.75
The number of actions	53.50	58.50	81.25	82.63

7. Conclusions and Discussions

In this paper, we propose to clarify the dependency and the sensitivity of the values of parameters in RL on the environment. To this end, we create four types of environment based on a hypothesis on the complexity measures of the environment, i.e., the number of possible states and the ratio of the number of rotation actions to the number of total actions. Firstly, we clarify how optimized parameter values are affected by goal weights in the fitness function of a GA. The result indicates that the number of goals reached is not monotonic with respect to the goal weight in Env-1. In contrast to this, the number of goals reached is monotonic with respect to the goal weight in Env-2. The maximum difference in the number of actions among different weights is about 10 in both environments. We change the goal weight for the fitness function in a GA, and we make additional learning from the final Q-values for the best fit individual in a GA. The results indicate that the goal weight hardly affect the learning performance of RL.

Secondly, we clarify the sensitivity of optimized values of parameters in RL. Excepting the discount rate, the perturbation does not affect much to the performance of RL. This result shows that the values of a discount rate should be determined with much care due to its large sensitivity.

Thirdly, we evaluate the dependency of parameter values in RL on the environment. In this experiment, we change the parameter values in RL with other environment. The result indicates that the two types of complexity measures affect learning performance. We have to take into consideration of these measures in determining of the values of parameters in RL, and the complexity measures distinctive the environment. As results of these experiments, we can predict the values of parameters in RL using the complexity measures.

As a future work, we will predict the values of parameters in RL based on the complexity measures for a novel environment.

Table 4. Results of Performance due to Perturbation of Parameter Values.

	Perturbation Values	Reward for Forward		Reward for Rotation		Reward for Collision	
		#goals	#actions	#goals	#actions	#goals	#actions
Env-1	+20%	248.50	53.38	249.38	53.50	249.00	53.38
	-20%	248.75	52.75	249.38	53.13	249.00	53.38
Env-2	+20%	247.00	60.25	247.63	59.13	247.63	58.50
	-20%	247.75	57.88	247.14	58.29	247.63	58.50
Env-3	+20%	247.63	81.10	247.88	82.13	248.13	81.25
	-20%	247.88	81.13	248.00	81.13	248.13	81.25
Env-4	+20%	246.63	82.75	245.63	84.75	246.75	83.00
	-20%	246.25	82.63	245.86	81.57	245.38	83.13
		Reward for Obstacle		Discount Rate		Learning Rate	
		#goals	#actions	#goals	#actions	#goals	#actions
Env-1	+20%	249.00	53.50	249.00	53.50	248.63	53.75
	-20%	249.00	53.50	248.13	53.13	249.50	53.50
Env-2	+20%	247.63	58.50	247.63	58.63	247.88	58.75
	-20%	247.63	58.50	79.13	59.38	247.50	59.00
Env-3	+20%	248.13	81.25	248.13	81.25	247.88	81.00
	-20%	248.13	81.25	247.38	80.00	248.00	81.13
Env-4	+20%	246.88	83.00	247.13	83.00	247.13	84.50
	-20%	246.88	83.00	202.38	64.50	245.63	83.25

Acknowledgment

This research was supported by the 21st Century Program by MEXT, Japan, and by Grant-in-Aid for Scientific Research(C)(15500140) by MEXT, Japan.

References

[1] R. S. Sutton and A. G. Barto, *Reinforcement Learning*, MIT Press, 1998

[2] C. Unsal, P. Kachroo, and J. S. Bay, "Multiple Stochastic Learning Automata for Vehicle Path Control in an Automated Highway System," *Proceedings of the 1999 IEEE Trans. Systems, Man,* and *Cybernetics*, Part A: Systems and Humans, 1999, vol. 29, pp120-128.

[3] K. Kamei and M. Ishikawa, "More Effective Reinforcement Learning by Introducing Sensory Information," *International Joint Conference on Neural Networks*, pp3185-3188, 2004

[4] K. Samejima, T. Omori, "Adaptive internal state space construction method for Reinforcement learning of a real-world agent," *Neural Networks*, 12, pp1143-1155, 1999.

[5] S. Zilberstein, R. Washington, D.S. Bernstein, A.I. Mouaddib, "Decision-Theoretic Control of Planetary Rovers," *Plan-Based control of Robotic Agents, LNAI*, 2002, No. 2466, pp270-289.

[6] E. Zalama, J. Gomez, M. Paul, J. Peran, "Adaptive Behavior Navigation of a Mobile Robot," *Proceedings of 2002 IEEE Trans. Systems, Man, and Cybernetics*, *Part A: Systems and Humans*, 2002, vol. 32, pp160--169.

[7] R. Pfeifer and C. Scheier, "Understanding Intelligence," MIT Press, 1999

[8] J.E. Pettinger and R.M. Everson, "Controlling Genetic Algorithms with Reinforcement Learning," Department of Computer Science, School of Engineering and Computer Science, University of Exeter. EX4 4QF. UK, 2003

[9] S. Calderoni and P. Marcenac, "MUTANT: a MultiAgent Toolkit for Artificial Life Simulation," *IEEE. Published in the Proceedings of TOOLS-26'98*, August 3-7, 1998 in Santa Barbara, California.

[10] M. R. Lee and H. Rhee, "The effect of evolution in artificial life learning behavior," *Journal of intelligent and robotic systems*, 2001, Vol. 30, pp399-414.

[11] K. Kamei and M. Ishikawa, "A genetic approach to optimizing the values of parameters in reinforcement learning for navigation of a mobile robot," *International Conference on Neural Information Processing,* pp1148-1153, 2004

[12] A. Eriksson, G. Capi, and K. Doya, "Evolution of meta-parameters in reinforcement learning algorithm," *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2003

[13] K. Kamei and M. Ishikawa, "Improvement of Performance of Reinforcement Learning by Introducing Sensory Information and a GA with Inheritance," *International Conference on Neural Information Processing*, pp743-748, 2005

[14] K. Kamei and M. Ishikawa, "Reduction of Computational Cost in Optimization of Parameter Values in Reinforcement learning by a Genetic Algorithm," *The Second International Conference on Brain-inspired Information Technology,* pp. 83, 2005

Keiji Kamei received his BS in Computer Science from the Department of Control Engineering and Science, Faculty of Computer Science and Systems Engineering, Kyushu Institute of Technology(KIT) in 2002. In 2004 he obtained MS in Computer Science from the Department of Brain Science and Engineering, KIT. His research interest includes computer science, neural computation and robotics.

Masumi Ishikawa received his BS and MS in Electrical Engineering from the University of Tokyo in 1969 and 1971, respectively. In 1974 he obtained his PhD in Electrical Engineering from the University of Tokyo. He become a research associate at the Electrotechnical Laboratory, Ministry of International Trade and Industry, Japan, in 1974.He was promoted to a senior scientist at the Electrotechnical Laboratory in 1979. From 1986 to 1987 he was a visiting scholar at the Institute for Cognitive Science, University of California, San Diego. In 1990 he moved to the Faculty of Computer Science and Systems Engineering, Kyushu Institute of Technology(KIT), as a full professor. Since 2000, he has been a professor, Department of Brain Science and Engineering, KIT. His major research interests include neural computation and its applications to mobile robots. (Home page: http://www.brain.kyutech.ac.jp/~ishikawa)