ADAPTIVE SWARM BEHAVIOR ACQUISITION BY A NEURO-FUZZY SYSTEM AND REINFORCEMENT LEARNING ALGORITHM

TAKASHI KUREMOTO

Graduate School of Science and Engineering, Yamaguchi University,
755-8611, Tokiwadai, 2-16-1, Ube, Japan
wu@yamaguchi-u.ac.jp[*]
http://www.nn.csse.yamaguchi-u.ac.jp/home/wu

MASANAO OBAYASHI

Graduate School of Science and Engineering, Yamaguchi University,
755-8611, Tokiwadai, 2-16-1, Ube, Japan
m.obayas@yamaguchi-u.ac.jp[†]
http://www.nn.csse.yamaguchi-u.ac.jp/obayashi

KUNIKAZU KOBAYASHI

Graduate School of Science and Engineering, Yamaguchi University,
755-8611, Tokiwadai, 2-16-1, Ube, Japan
koba@yamaguchi-u.ac.jp[‡]
http://www.nn.csse.yamaguchi-u.ac.jp/k

Received (Day Month Year)
Revised (Day Month Year)
Accepted (Day Month Year)

Purpose – A neuro-fuzzy system with a reinforcement learning algorithm (RL) for adaptive swarm behaviors acquisition is presented. The basic idea is that each individual (agent) has the same internal model and the same learning procedure, and the adaptive behaviors are acquired only by the reward or punishment from the environment. The formation of the swarm is also designed by RL, e.g., TD-error learning algorithm, and it may bring out a faster exploration procedure comparing with the case of individual learning.
Design/Methodology/Approach – The internal model of each individual composes a part of input states classification by a fuzzy net, and a part of optimal behavior learning network which adopting a kind of reinforcement learning methodology named *actor-critic* method. The membership functions and fuzzy rules in the fuzzy net are adaptively formed online by the change of environment states observed in the trials of agent's behaviors. The weights of connections between the fuzzy net and the action-value functions of *Actor* which provides a stochastic policy of action selection, and *Critic* which provides an evaluation to state transmission, are modified by temporal difference error (TD-error).
Findings - Simulation experiments of the proposed system with several goal-directed navigation problems were accomplished and the results showed that swarms were successfully formed and optimized routes were found by swarm learning faster than the case of individual learning.
Originality/value – Two techniques i.e. fuzzy identification system and reinforcement learning algorithm are fused into an internal model of the individuals for swarm formation and adaptive behavior acquisition. The proposed model may be applied to multi-agent systems, swarm robotics, metaheuristic optimization and so on.

Keywords: neuro-fuzzy net, swarm behavior, reinforcement learning, multi-agent system, *actor-critic* algorithm, goal-directed navigation problem.
Style: research paper

1. Introduction

Animals survive to the nature replying not only on the adaptive ability of individuals but also on the life style of the species. Individuals, such as birds, fishes, ants, bees and so on, gather to form their swarms to adapt to the dynamic environment more efficiently. It can be considered that the swarm behavior means a kind of intelligence of a species. To search optimal solution by metaheuristic methods, mathematical swarm models, such as Particle Swarm Optimization (PSO) (see (Kennedy and Eberhart, 1995), (Kennedy *et al*., 2001)) which swarm consists of a number of "particles" searching multidimensional space with different velocity, Ant Colony Optimization (ACO) (see (Dorigo *et al*., 1996), (Dorigo and Caro, 1999) which swarm consists of a number of "ants" finding the shortest route with pheromone, are presented and work well to the optimization problems, such as Traveling Salesperson Problem (TSP), machine Job-shop Scheduling Problem (JSP), etc. To solve those problems in a large, complex, open system such as the internet, Multi-Agent Systems (MASs) which using a number of modular components (agents) dealing with problems of reliability, computational efficiency, responsiveness, maintainability, etc., are also effective techniques (Sycara, 1998). All of those swarm models use cooperative behaviors between individuals and global optimal exploration is evaluated iteratively to improve temporary or local solutions.

However, these models usually are designed with a prior assumption that particles or individuals have existed in some kind of swarm state, and the swarm behaviors are given by the structure of mathematical models. The process of how a swarm is formed in the environment is often neglected. The motivation of an individual's action is not considered deeply, i.e., rules of actions are designed to response external stimuli directly. For example, a particle in PSO flies according to observing the positions and the directions of other individuals, visual information is used directly to decide the direction and the velocity of the particle in the next step. For another example, an artificial ant in ACO moves to find a good solution according to pheromone concentration of available paths, olfactory information is used stochastically to decide the action of the ant. And as same as inherent problems need to be challenged in MASs (Sycara, 1998), there are also foundational issues in swarm intelligence should be faced: "How do we formulate, describe, decompose, and allocate problems and synthesize results among a group of intelligent agents? How do we enable agents to communicate and interact? How do we ensure that agents act coherently in making decisions or taking action, and avoiding harmful interactions? How do we enable individual agents to represent and reason about the actions, plans, and knowledge of other agents to coordinate with them? How do we recognize and reconcile disparate viewpoints and conflicting intentions among a collection of agents trying to coordinate their actions? And how do we design technology platforms and development methodologies for swarm models?"

This paper intends to tackle all those issues of swarm intelligence mentioned above. The foundational principle and the methodology are the following:

First, we propose that the motivation of individuals' behaviors is given by positive or negative rewards from the external dynamic environment and emotional response. The reward is a scalar parameter which realizes learning scheme and it may be confirmed concerning with dopamine in a real brain (see (Schultz *et al*., 1997) and (Schultz, 1998, 2001)). Dopamine neurons exist in a wide spread of midbrain, basal ganglia and frontal cortex corresponding to plural sensory inputs such as visual, auditory and somatosensory signals. The reward responses appear not only when the brain accepts environmental stimulus but also when "prediction error" occurs, and this mechanism may motivate learning process to acquire "adaptive behavior" (Waelti *et al*., 2001).

Second, to form a swarm, to make agents act coherently in the decision process, and to avoid harmful interactions, we design an internal model of each individual (agent) which is able to acquire adaptive behaviors through its self-exploration. The proposed internal model consists of a neuro-fuzzy network with a reinforcement learning algorithm (RL). Agents explore the unknown environment and receive different reward values from the different states. For an example, the rewards can be calculated by the values of events such as obstacle crashing, goal arrival, and near collision or excessive separation from other agents in a goal-directed navigation problem. Adaptive and coordinative behaviors are yielded by the state value function and the action value function which are modified in the process of a stochastic exploration of agents. In detail, the adaptive action is given by *Actor* with a stochastic policy which can be modified by *Critic* and temporal difference error (TD-error) of state values. *Critic* calculates the value of a state accepting the reward (positive or negative scalar value) from the environment. And rewards given according to the Boids rules (Craig Reynolds, 1986) play a crucial role to form a swarm and obtain the adaptive swarm behaviors. So the proposed internal model and its learning algorithm provide a novel platform to study swarm intelligence including behavior learning, swarm control and swarm formation.

Third, to confirm the performance of our learning system, 2 kinds of goal-directed navigation problems were used in computer simulation experiments. Individual learning and swarm learning methods were compared, while a simple environment and a complex environment (i.e., a maze-like environment) were also tested. Furthermore, discrete state space and continuous state space were both employed to investigate the effectiveness of the proposed model.

This paper has the following form. In Section 2 an internal model of individuals with a structure of neuro-fuzzy network and a reinforcement learning algorithm for adaptive swarm behaviors are described. In Section 3, the description and results of computer simulation experiments are given and conclusion of this research is in Section 4.

2. A Neuro-Fuzzy Network with Reinforcement Learning Algorithm

Architecture of a neuro-fuzzy reinforcement learning system is shown in Fig.1. The system is designed for each individual (agent) to swarm exploration and swarm behavior formation. It composes with a part of Fuzzy net and a part of *actor-critic* learning network. An agent observes states $\mathrm{x}(t) \in \mathrm{R}^{\mathrm{n}}$ from the environment and classifies the inputs into *k* classes $\phi_k(\mathrm{x}(t))$ with its Fuzzy net. The agent executes an action which may change its state according to a policy function given by *Actor*. *Critic* receives rewards from the environment or external evaluators, connects to the Fuzzy net with synapses (weights). The value of each state is calculated by a state value function $V(\mathrm{x}(t))$, and its temporal change results temporal difference error (TD-error). TD-error is used to adjust policy function through the action value function $A(\mathrm{x}(t))$ given by *Critic*. Each agent has the same architecture and there is no any communications

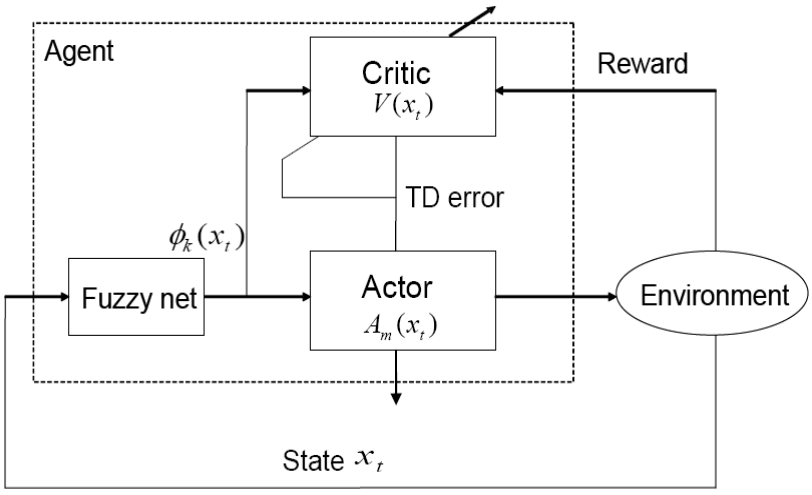


Fig.1. Structure of a neuro-fuzzy learning system for adaptive swarm behaviors is shown. The internal model of agents has a part of Fuzzy net and a part of *actor-critic* learning network. The reward used in calculating TD-error is given by not only the environment but also the distance with other agents.

between the individuals. When swarm behaviors are evaluated with positive rewards according some conditions, agents learn to form swarm and acquire adaptive behaviors, i.e. act more efficiently than individual situation.

2.1. *Fuzzy net*

A Fuzzy net which uses self-multiplication Gaussian membership functions and rules to classify state of the environment (Kuremoto, *et al*., 2008b, 2008c) is adopted into the internal model of individuals (Fig. 2). The Fuzzy net has an RBF-like architecture and powerful ability to classify continuous input and give continuous output (Jouffe, 1998). It has been successfully adopted in cart-pole balance control (Wang, X. S., et al., 2007), adaptive behavior learning of autonomous robots (Samejima and Omori, 1999), (Perez-

Uribe, 2001), and nonlinear prediction systems in our previous work (Kuremoto, *et al*., 2003, 2007, 2008a).

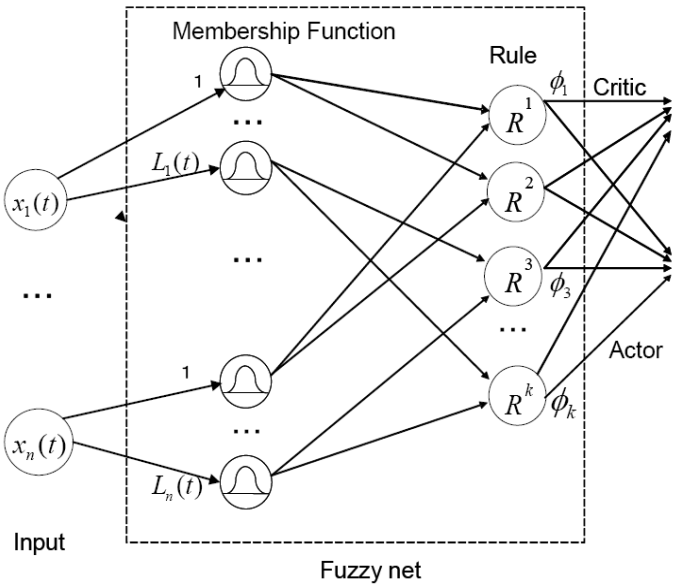


Fig.2. Fuzzy net uses self-multiplication Gaussian membership functions dealing with continuous input space (states of agents) and output space (the value of state and action).

For an *n*-dimension input state space $\mathbf{x}(x_1(t), x_2(t),\ldots,x_n(t))$, the fuzzy inference net is designed with a hidden layer of fuzzy membership functions $B_i^k(x_i(t))$ to categorize the input states.

$$B_i^k(x_i(t)) = \exp\left\{-\frac{\left(x_i(t) - c_i^k\right)^2}{2\sigma_i^{k\,2}}\right\} \quad . \tag{1}$$

Here c_i^k , σ_i^k denote the mean and the deviation of *i*th membership function which is a kind of radial basis function (RBF) in *k*th fuzzy rule corresponding to *i*th input $x_i(t)$, respectively. $i = 1,2,\ldots,n$.

Let $K(t)$ be the largest number of fuzzy rules R^k ($k = 1,2,\ldots,K$), then we have Equation (2) for R^k :

if ($x_1(t)$ is $B_1^k(x_1(t))$, ,... , $x_n(t)$ is $B_n^k(x_n(t))$) then

$$\phi_k(\mathrm{x}(t)) = \prod_{i=1}^{n} B_i^k(x_i(t)). \quad (2)$$

Where $\phi_k(\mathrm{x}(t))$ means the fitness of the rule R^k for the input $\mathbf{x}(t)$.

The number of membership functions and rules of Fuzzy net are important for a fuzzy inference system. We proposed a self-organized fuzzy neural network (SOFNN) which constructed adaptive membership functions and rules using training data and thresholds previously (Kuremoto, *et al*., 2003, 2007, 2008a), (Obayashi, *et al*., 2008). Wang, Cheng, and Yi proposed a structure learning algorithm for adding and merging units using TD-error distribution recently (Wang, X. S., *et al*., 2007). Here we use a simpler self-multiplication algorithm to decide the size of Fuzzy net. The structure decision of Fuzzy net is as following.

Only one membership function is generated by the first input data (for example, the position of agent) of each input state vector. The value of its center equals to the value of input, and the value of width (standard deviation) of each Gaussian function units is fixed to an empirical value. The number of rule functions is one at first, and the output of the rule R^1 equals to $\phi_1(x(1)) = \prod_{i=1}^{n} B_i^1(x_i(1))$ according to Equation (2).

For the next input state $\mathbf{x}(x_1(t), x_2(t), \ldots, x_n(t))$, a new membership function is generated if Equation (3) is satisfied.

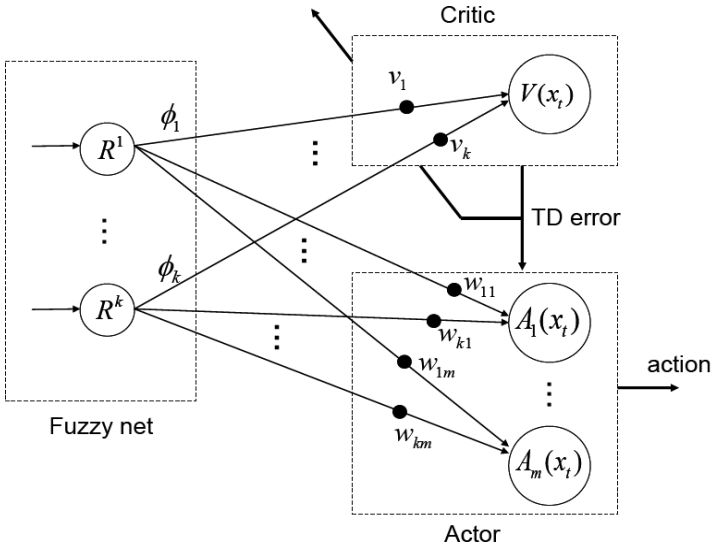


Fig.3. An *actor-critic* learning network is used for acquiring optimal behaviors dealing to characterized states by Fuzzy net. *Actor* generates an action of an agent according to a stochastic policy which concerning with the output of Fuzzy net. *Critic* calculates temporal difference error (TD-error) using the reward of the states from the environment.

$$\max_{s} B_{i,s}(x_i(t)) < F \,. \tag{3}$$

Here $B_{i,s}(x_i(t))$ denotes the value of existed membership functions calculated by (1) and $s = 1,2,...,L_i(t)$ indicates the *s*th membership function, the maximum number is $L_i(t)$. *F* denotes a threshold value of whether an input state is evaluated enough by existing membership functions. A new rule is generated automatically corresponding to the new membership function. According to above membership function and rule generation criteria, Fuzzy net is completed to adapt to features of input states in time.

2.2. *An actor-critic Learning Nework*

Reinforcement learning algorithm (RL) is defined as learning by trial-and-error from a learner's performance and its feedback from the environment or an external evaluator (Sutton and Barto, 1998). The mechanism of RL is considered from origins in the psychology of animal learning and hypothesized existing in the brain (Doya, 2002). As a computational theory of acquisition of goal-directed behaviors, RL has been successfully applied to many study fields such as robust control (Wang *et al*., 2007), autonomous robot design (Samejima and Omori, 1999), (Perez-Uribe, 2001), nonlinear prediction (Kuremoto *et al*., 2003, 2007, 2008a), and so on. For cooperative behaviors learning in swarm robotics, IIma and Yasuaki proposed a swarm RL algorithm using information sharing of agents (IIma and Yasuaki, 2006), and Kawakami *et al*. showed the effectiveness of actor-critic algorithm for co-operation behaviors learning (kawakami *et al*., 2005). However, a serious problem exists when RL is used in multi-agent systems is that the explosion of state space (curse of dimension). To overcome this hurdle, approaches using fuzzy inference systems (Jouffe, 1998), (Wang, X. S., et al., 2007), adaptive global radial basis function (GRBF) network (Samejima and Omori, 1999) are proposed, and recently classifying states using ART (Kobayashi *et al*., 2005), Neuro-GAS-like technique (Kobayashi *et al*., 2008) or hierarchical structure model (Wang, B. N. *et al*., 2007) are also given effectively. In this study, individuals are assumed without any knowledge of the environment, and they act according to a stochastic policy of the internal model temporally. To acquire adaptive behaviors, RL can be considered as a suitable learning algorithm for each individual (agent). Fuzzy net described by section 2.1 can be adopted to deal with the explosion of state space for its ability of pattern classification.

The *actor-critic* methods in RL are applied successfully on the fields of adaptive control and autonomous systems (Sutton and Barto, 1998). There are usually 2 structures used in the methods called *actor* and *critic*. The *actor* corresponds to a probabilistic function called selection policy which mapping states to actions. The *critic* corresponds to a value function which is used to improve the selection policy by the reward value received from the environment or internal signal.

The part of *actor-critic* learning network for acquiring optimal behaviors is shown in Fig.3. The weighted outputs of Fuzzy net are used to calculate the value of states and actions according to Equation (4) and Equation (5), respectively.

$$V(\mathrm{x}(t)) = \frac{\sum_k v_k(t)\phi_k(\mathrm{x}(t))}{\sum_k \phi_k(\mathrm{x}(t))}. \tag{4}$$

$$A_j(\mathbf{x}(t)) = \frac{\sum_k w_{kj}(t)\phi_k(\mathbf{x}(t))}{\sum_k w_{kj}(\mathrm{x}(t))}. \tag{5}$$

Here v_k, w_{kj} are the weighted connections between Fuzzy net output $\phi_k(\mathrm{x}(t))$ and state value function $V(\mathrm{x}(t)) \in R^1$ in the *Critic*, action value function $A_j(\mathrm{x}(t)) \in R^m$ in the *Actor*. A_j denotes values of *j*th action a_j (behavior) selected by the agent according to a stochastic policy Equation (6) dealing with the input state x(*t*). *j* = 1, 2, …, *m*.

$$P(a_t = a_j \mid \mathrm{x}(t)) = \frac{\exp(A_j(\mathrm{x}(t))/T)}{\sum_m \exp(A_m(\mathrm{x}(t))/T)}. \tag{6}$$

Here *T* is the temperature of Boltzmann distribution. Higher *T* causes more active exploration, and lower temperature causes more greedy action to the goal.

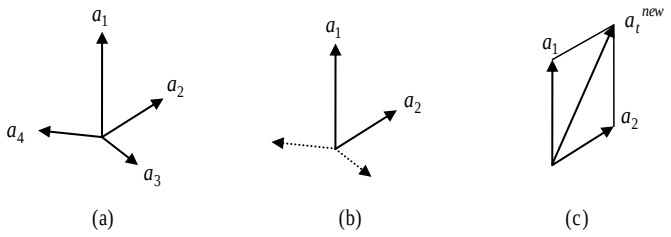


Fig.4. A new action is created in the continuous action space. The length of the arrow lines shows the different value of action-value function $A_m(\mathrm{x}(t))$. (a) A 4-dimension action space is shown. (b) A new action space is obtained using higher value dimensions a_1 and a_2. (c) A new action a_t^{new} is created by a linear combination operation.

The action value function $A_j(\mathrm{x}(t)) \in R^m$ in *Actor* belongs to an *m*-dimension real number space corresponding to *m* actions $a_t \subset (a_1, a_2, ..., a_m)$. In our previous study, only discrete actions were permitted for the simulation of goal-exploration problem (Kuremoto, *et al*, 2008b). Agents moved 1 grid by 1 step, one and only one of 4

candidature directions is limited for the selection of action. When the information of state space is given by position of agent, and expressed by coordinates, state space also comes to a discrete system according to the discrete actions. Recently, we proposed to change the *m*-dimension of actions to one arbitrary dimension by linear combination (Kuremoto, *et al*., 2008c). In fact, policy function Equation (6) decides one action which has highest probability in *m* candidates conventionally. The stochastic selection neglects *m*-1 actions for their lower probabilities. Now, let *h* enough high probabilities $P_t(a_t = \sum_{i=1}^{h} a_i \mid \mathrm{x}(t))$ express the probability vector of action a_t:

$$P_t(a_t = \sum_{i=1}^{h} a_i \mid \mathrm{x}(t)) = \sum_{j=1}^{h} P_t(a_t = a_j \mid A_m(\mathrm{x}(t))) \le 1.0 . \tag{7}$$

Where $h \le m$.

So a new action set ($a_1, a_2, \ldots a_h$) becomes a dominant candidate. *m* minus *h* actions which have lower values are neglected. A new action a_t^{new} may be generated by linear combination in the dominant action vector space as shown as Equation (8). The value of the new action $A_{new}(\mathrm{x}(t))$ is given by Equation (9) which concerns with all values of dominant actions.

$$\vec{a}_t^{new} = k_1 \vec{a}_1 + k_2 \vec{a}_2 + \ldots + k_h \vec{a}_h . \tag{8}$$

$$A_{new}(\vec{a}_t^{new} \mid \mathrm{x}(t)) = \| \vec{a}_t^{new} \|_z = (\sum_{i=1}^{h} k_i^z)^{\frac{1}{z}} . \tag{9}$$

Where $h \le m$, coefficients $k_1, k_2, \ldots, k_n$ are scalar values given by action value function Equation (5) and actions $a_1, a_2, \ldots, a_h$ are the basis of action vector space, $z = 2$ when the action vector is in Euclidean space. The process of a new continuous action is shown by Fig.4.

Critic owns its value and provides the temporal difference error (TD-error) using reward from the environment resulted by *Actor*'s action. TD-error $\varepsilon_{TD}(\mathrm{x}(t))$ is given by Equation (10).

$$\varepsilon_{TD}(\mathrm{x}(t)) = r_t + \gamma V(\mathrm{x}(t+1)) - V(\mathrm{x}(t)) . \tag{10}$$

Here r_t is the value of reward, γ is a discount.

TD-error obtained from Equation (10) is reduced by adjusting the connection weights of *actor-critic* network in learning iterations, i.e. Equation (11) and Equation (12), respectively.

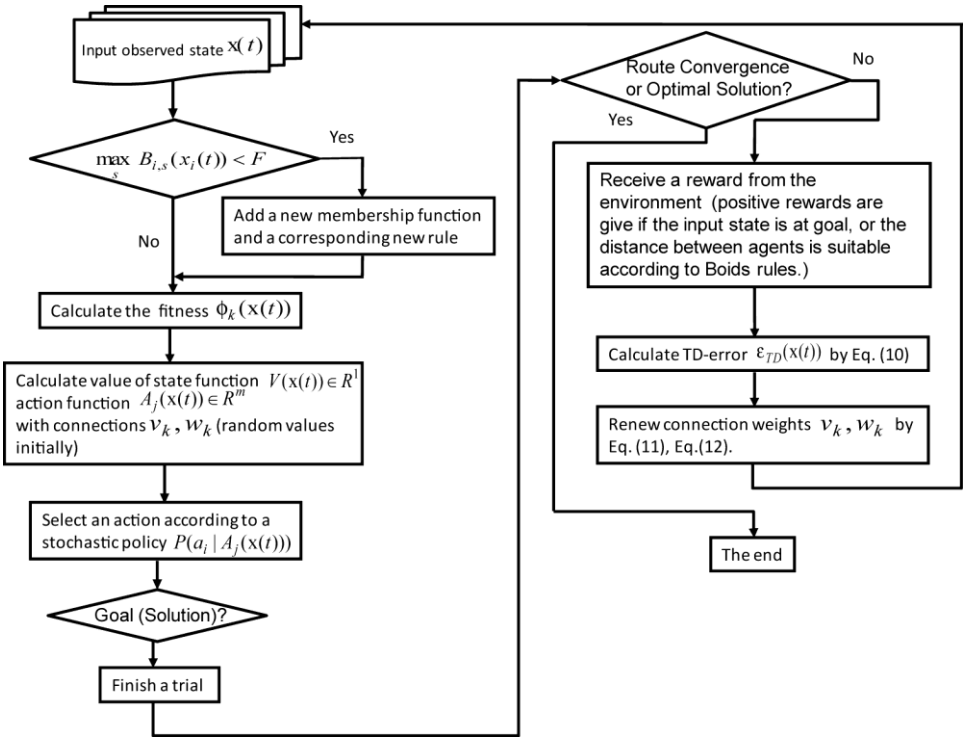
$$v_k(t+1) = v_k(t) + \beta_v \varepsilon_{TD}(\mathrm{x}(t)) \phi_k(\mathrm{x}(t)) . \tag{11}$$

$$w_{kj}(t+1) = w_{kj}(t) + \begin{cases} \beta_w \varepsilon_{TD}(\mathrm{x}(t)) \phi_k(\mathrm{x}(t)) & a_t = a_j \\ 0 & otherwise \end{cases} \tag{12}$$

Here β_v, β_w denote rate of learning.

By modifying the connection weights (synapses) between Fuzzy net and *Actor* and *Critic* according to Equation (10), Equation (11) and Equation (12), the system learns to select an optimal action by the modified stochastic policy (6) corresponding to the input state $\mathrm{x}(t)$.

A learning algorithm and a flow chart that summarized the proposed learning procedure are shown as the following:



Step 1 Observe the state $\mathbf{x}(x_1(t), x_2(t), \ldots, x_n(t))$ of each agent at discrete time *t*.

Step 2 Generate fuzzy membership functions and rules according to self-multiplication algorithm described in Section 2.1.

Step 3 Calculate the fitness of fuzzy net from Equation (1) and Equation (2).

Step 4 Calculate the value of action function in *Actor* and the value of state function in *Critic* by defuzzification given by Equation (4) and Equation (5).

Step 5 Select a valuable action (including the new action given by Equation (8) if necessary) according to a probability distribution given by Equation (6).

Step 6 Calculate TD-error $\varepsilon_{TD}(\mathrm{x}(t))$ by Equation (10) when the reward is obtained from the environment and the next state is observed.

Step 7 Renew the weights of connections between Fuzzy net and *Critic*, *Actor* v_k, w_k (which initial values are random) by Equation (11) and Equation (12) to reduce TD-error.

Step 8 Finish one trial if the state is at the goal.

Step 9 Return to Step 1 if the number of steps in trials is not convergence.

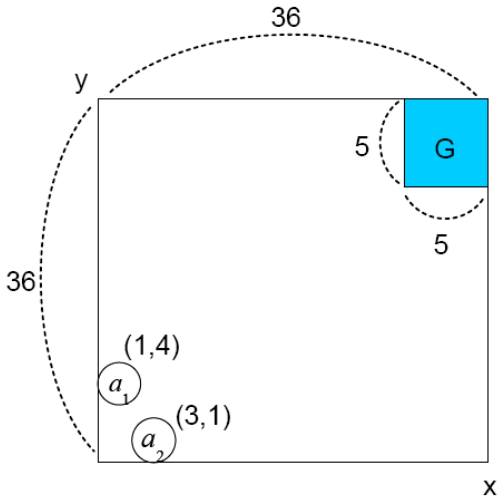


Fig.5. A simulation environment for two agents goal-directed searching experiment. The search area was set to be a square with a size of 36x36. Agent a_1 started from (1, 4) and agent a_2 started from (3, 1) and square G (31-36, 31-36) was set as the goal area.

2.3. Rewards for swarm

Swarm behavior is defined as the collection behaviors of independent agents each responding to local stimuli without supervision (Kennedy and Eberhart, 1995), (Dorigo *et. al*., 1996). To form swarm and acquire adaptive swarm behaviors by reinforcement learning process here, agents need to obtain rewards when they arrive at the goal, crash to obstacles or themselves, or whether a swarm is formed. In fact, Craig Reynolds proposed a famous model to simulation bird flocks or fish schools (Reynolds, 1986). The basic rules are called *Boids* which named from bird oid. *Separation*, *Alignment* and *Cohesion* are defined as the rules of *Boids*. If each individual is designed takes action according to the *Boids* rules, then a swarm is able to be formed and the swarm behaviors appears dynamically. Here we give positive/negative rewards to *Cohesion/Separation* to each agent and *Alignment* is adopted indirectly by a sufficient positive reward to the goal of all agents.

Table I Parameters used in the simulation described in Section 3.1.

Description	Symbol	Quantity
Dimension of input vectors	n	2
The number of actions	m	4
Standard deviation of membership	σ_i^k	0.1
Threshold of fitness	F	0.4
Initial weight between rules and *Critic*	v_k	1.0
Initial weight between rules and *Actor*	w_{kj}	0.25
TD learning coefficient for *Critic*	β_v	0.3
TD learning coefficient for *Actor*	β_w	0.3
Discount of TD-error	γ	0.9
Temperature of Bolzmann distribution	T	0.1
Reward for goal arrived	r_{goal}	100.0
Reward for wall or agent crashed	r_o	-1.0
Reward for swarm formed	r_{swarm}	1.0
Reward for swarm unformed	$r_{a\text{-}swarm}$	-1.0
Minimum distance between agents	min_*dis*	1.5
Maximum distance between agents	max_*dis*	3.0

Let $D(\mathrm{x}^p(t), \mathrm{x}^q(t))$ express the Euclidean distance between two arbitrary agents p and q, i.e.:

$$D(\mathrm{x}^p(t), \mathrm{x}^q(t)) = \sqrt{\sum_{i=1}^{n} (x_i^p(t) - x_i^q(t))^2} \ . \tag{13}$$

Then we give a positive reward r_{swarm} in Equation (15) to those agents satisfied Equation (14), or a negative reward in the opposite. This means that only by the consideration of swarm reward, agents are endowed to learn to select swarm behaviors by themselves.

$$\min\_dis \le D(\mathrm{x}^p(t), \mathrm{x}^q(t)) \le \max\_dis. \tag{14}$$

$$r_{t+1} = r_t \pm r_{swarm} \ . \tag{15}$$

Here $\min\_dis, \max\_dis$ denote near limit distance and far limit distance, respectively. r_{t+1} in Equation (15) is substituted into TD-error calculation Equation (10).

3. Computer Simulation Experiments

Computer simulations to solve goal-directed navigation problems were executed to confirm the performance of the proposed reinforcement learning system for swarm formation and adaptive swarm behaviors acquisition. The specification of personal computer is with a 2.4GHz CPU and a 512MB memory. C language was used in the simulations. Using coordinate information to describe the state space of agents, discrete

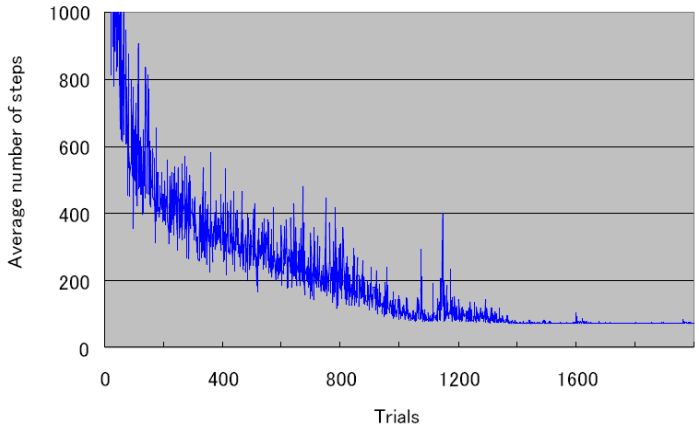


Fig.6. A case of one agent (Simulation (a)): The average number of steps (of 10 episodes) to arrive at the goal position reduced to less than 100 after 1,400 trials (learning iterations).

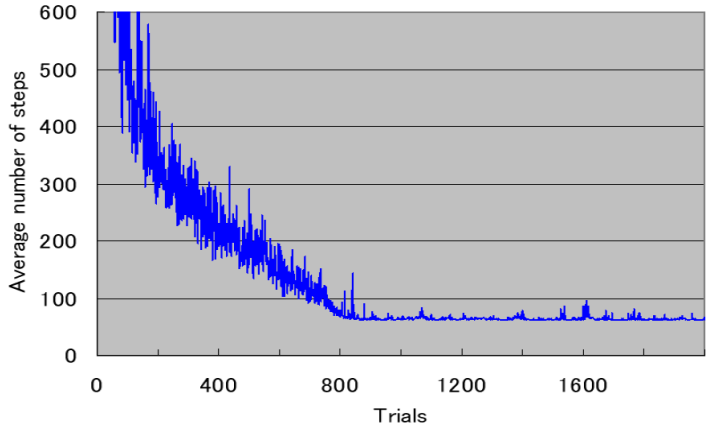


Fig.7. A case of two agents (Simulation (b)): The average number of steps (of 10 episodes) to arrive at the goal area reduced to less than 100 after 850 trials (learning iterations).

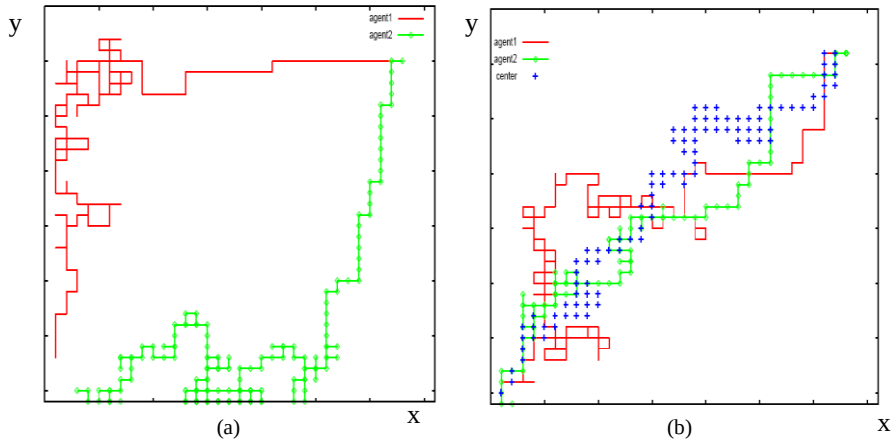


Fig.8. Comparison between individual leaning result (a) and swarm learning result (b) (Simulation (b)). 2 agents routes at 300 iterations of training are shown. "+" in (b) presents the center between 2 agents that is used to decide different value of reward in swarm learning. In (a) agents found goal area with lengthy steps and swarm formation could not be confirmed. In (b), agents found goal area with comparative optimized steps and swarm formation could be confirmed.

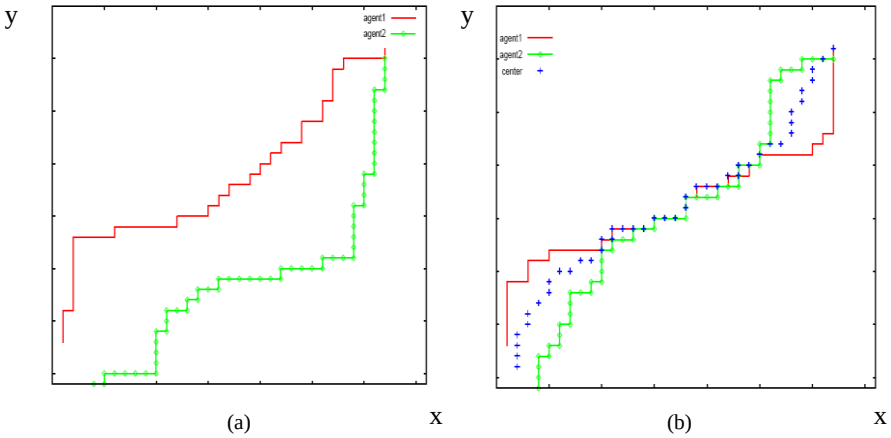


Fig.9. Comparison between individual leaning result (a) and swarm learning result (b) (Simulation (b)). 2 agents routes at 2,000 iterations of training are shown. "+" in (b) presents the center between 2 agents that is used to decide different value of reward in swarm learning. In (a) agents found goal area with comparative optimized steps but swarm formation could not be confirmed. In (b), agents found goal area with optimized steps and swarm formation was elevated.

action case and continuous action case were investigated, and the details are reported in Section 3.1 and Section 3.2 respectively.

3.1. *Goal Exploration in Discrete Space*

A square with a size of 36x36 was used as exploration space, and there were walls on the four sides, no obstacles and goal area was fixed as shown in Fig.5. An agent observed its position and position of others in the square, and moved one step to 4 probable directions: up, down, left and right. Agents did not have any information of the goal position before they arrived at it.

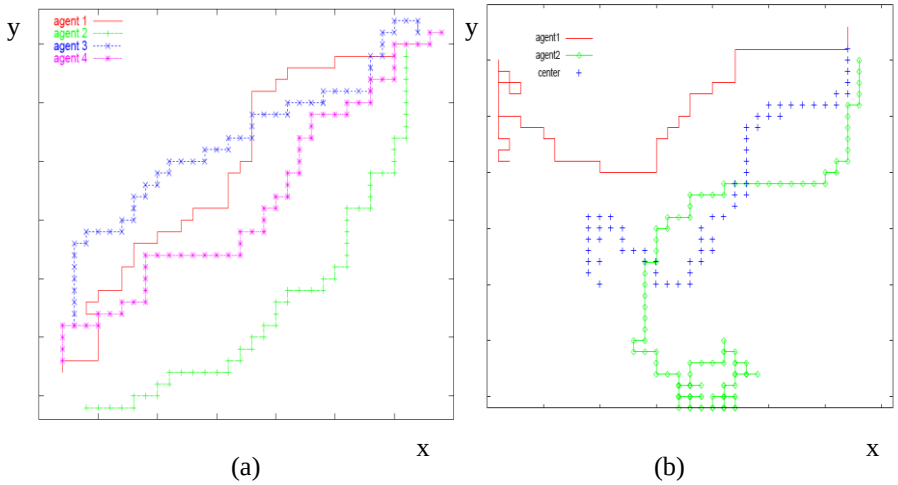
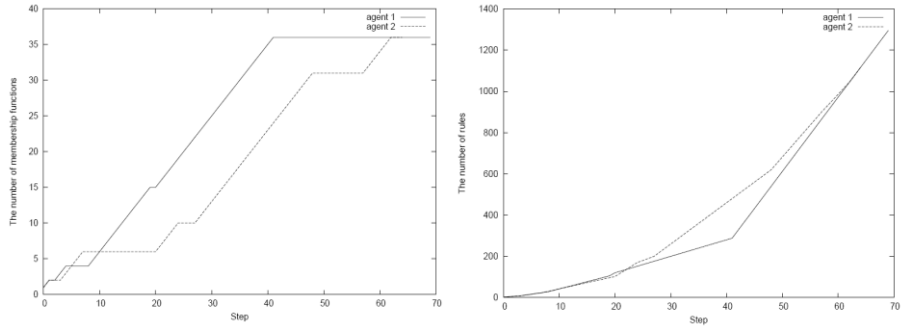


Fig.10. More investigations were done. (a) Swarm learning (Simulation (c)): routes of 4 agents after training. Optimized routes and swarm formation also can be confirmed. (b) Robustness of the system (Simulation (d)): the start position of each agent was chosen arbitrarily after training, the routes shown here are in a case of agent 1 started from (1, 22) and agent 2 started from (19, 1), optimized solutions can be confirmed respectively ("+" presents the center of 2 agents).

Three kinds of simulations were performed: (a) one agent goal-directed learning; (b) two or more agents goal-directed and swarm-behaviors-encouraged learning; (c) robustness examination of system after training. The values of parameters used in these simulations were shown in Table I. The number of learning iterations for one episode (or one trial: an exploration from the start to the goal) was set to be 2,000 in all kinds of simulations, and to observe average performance of the proposed stochastic learning system, the learning curves were given by results of 10 times simulations in each experiments.

Simulation (a): one agent goal-directed learning simulation was executed to test the internal model proposed here. One agent started from (1, 1) and tried to arrived to the goal (36, 36) by learning iterations. The 10 times average number of total steps of one exploration episode (an exploration episode means an agent state transmit from the start position to the goal area.) reduced with learning iterations as shown as in Fig.6. The optimized route of agent after 2,000 learning iterations had 70 steps which equals to the

shortest number of steps in theory. These results proved that the proposed system is effective to adapt to the environment and acquire adaptive behaviors in the goal-directed problem.



(a) (b)

Fig.11.The change of the number of internal nodes of Fuzzy net in Simulations with discrete space is shown. (a) The number of membership functions for x-dimension of input space grew with steps of agents in training (36 membership functions were yielded at last). (b) The number of rules for input states grew with steps of agents in training. For the 36x36 square, 1.296 rules were vielded at last.

Simulation (b): swarm formation and adaptive swarm behavior simulation was executed to test the RL method proposed here. Two agents stated from (1, 4) and (3, 1), respectively, tried to arrive at the goal area (31-36, 31-36). In this simulation, swarm reward was not considered at first, we call it "individual learning", i.e, each agent explored its optimal route independently. Then "swarm learning" was set by two kinds of reward providing: 1) the action which caused swarm states was encouraged with reward r_{swarm} = 1.0, where he minimum and maximum distance between two agents was set to 1.5 and 3.0 respectively; 2) the opposite situations which were agents too closed and too departed were discouraged with reward $r_{a\text{-}swarm}$ = -1.0 as values as shown in Table I. The number of total steps of one exploration episode reduced with learning iterations in swarm learning is shown in Fig.7. The optimized route of each agent after 2,000 learning iterations had 62 steps which equals to the shortest number of steps in theory in both of learning methods. The traces of agents on the 300^{th} trials (learning iterations) were shown in Fig.8. Two agents moved to the goal with redundant steps more separately in the case of individual learning (Fig.8 (a)) comparing with the case of swarm learning (Fig.8 (b)). Furthermore, the traces after 2,000 training, when learning has come to convergence, showed the optimal routes were selected in both methods, however, agents moved to the goal area keeping swarm form in the case of swarm learning (Fig.9).

Simulation (c): More agents case and the robustness after learning were confirmed. We also simulated swarm learning using 3 and 4 agents in the similar conditions of two agents, the similar learning results were obtained comparing with the case of two agents simulation results (Fig.10 (a)). These results proved the proposed system is effective to

multiple agents adapt to the environment and acquire optimal swarm behaviors in goal-directed problem.

Simulation (d): To examine robustness system after swarm learning, two agents restarted from positions (1, 22) and (19, 1) which were different from the start positions in learning. Agents still moved to the goal area with a swarm form though wandering at the early periods after left their restarted positions. The trajectories were shown in Fig.10 (b).

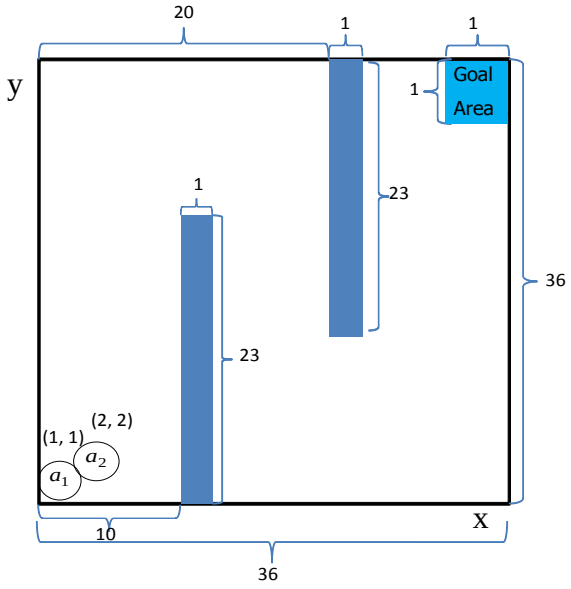


Fig.12. A maze-like environment for two agents of goal-directed exploration simulation experiment is shown. The search area was set to be a square with a size of 36x36. Agent a_1 started from (1, 1) and agent a_2 started from (2, 1) and Goal Area is in (35-36, 36-35).

The part of the Fuzzy net in the proposed system executed states categorization successfully in all of simulation experiments. The number of membership functions and rules in simulation (b) was shown in Fig. 11. It suggests that the self-multiplication algorithm proposed here was able to generate adaptive fuzzy net structure dealing to discrete input space.

3.2. *Goal Exploration in Continuous Space*

An environment as shown as Fig.5 and a maze-like environment as shown as Fig.12 were used for two agents of goal-directed exploration in the continuous space simulation experiments. Both search area were set to be a square with a size of 36x36. Agent a_1 started from (1, 1) and agent a_2 started from (2, 1) and Goal Area is in (35-36, 36-35) in each exploration area.

Here each agent observed its position and position of others in the square, and moved 1.0 length per step toward an arbitrary direction while only 4 probable directions was allowed in the conventional system simulations, i.e., up, down, left and right. The method to decide the direction of one time movement was to choose 2 orthogonal directions whose value of action function were higher than others. According to (7) and (9) in Section 2.2, the new action direction and its value were decided by 2 dominant action directions which had higher values than others. Agents did not have any information of the goal position before they arrived at it.

Table II. Parameters used in the simulations described in Section 3.2.

Description	Symbol	Quantity
Dimension of input vectors	n	**2**
The number of actions	m	2
Standard deviation of membership	σ_i^k	0.1
Threshold of fitness	F	0.4
Initial weight between rules and critic	v_k	1.0
Initial weight between rules and actor	w_{kj}	0.25
TD learning coefficient for critic	β_v	0.3
TD learning coefficient for actor	β_w	0.3
Discount of TD error	γ	0.9
Temperature of Bolzmann distribution	T	0.1
Reward for goal arrived	r_{goal}	100.0
Reward for wall or agent crashed	r_o	-10.0
Reward for corner crashed	r_c	-20.0
Reward for swarm formed	r_{swarm}	1.0
Reward for swarm unformed	$r_{a\text{-}swarm}$	-|*dis*|
Minimum distance between agents	min_*dis*	1.5
Maximum distance between agents	max_*dis*	3.0

Two kinds of simulation experiments using continuous input and output were performed in the different environments respectively. The values of parameters used in these simulations were shown in Table I. The number of learning iterations for one episode (or

one trial: an exploration from the start to the goal) was set to be not over 2,000 in all kinds of simulations.

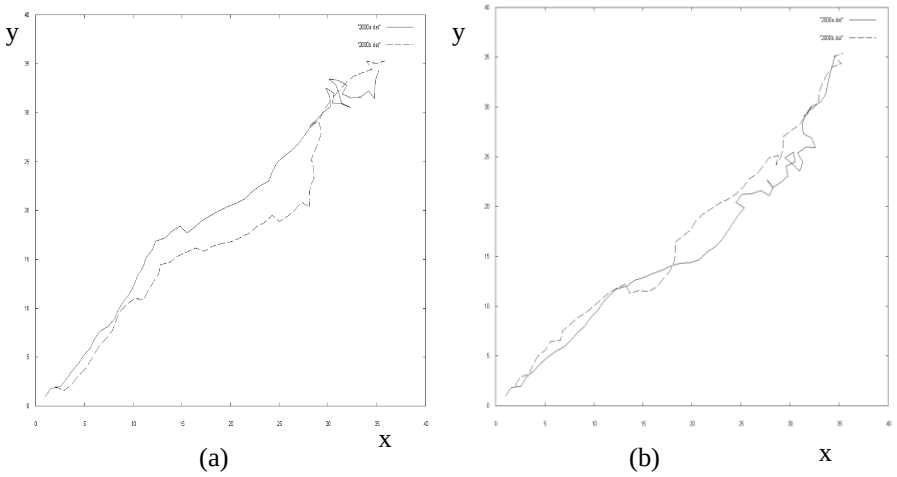


Fig.13. Results of Simulation I using continuous space of input and output. (a) Routes of 2 agents after 2,000 trials (learning iterations) by *individual learning* (Simulation I (i)). (b) Routes of 2 agents after 2,000 trials (learning iterations) by swarm *learning* (Simulation I (ii)).

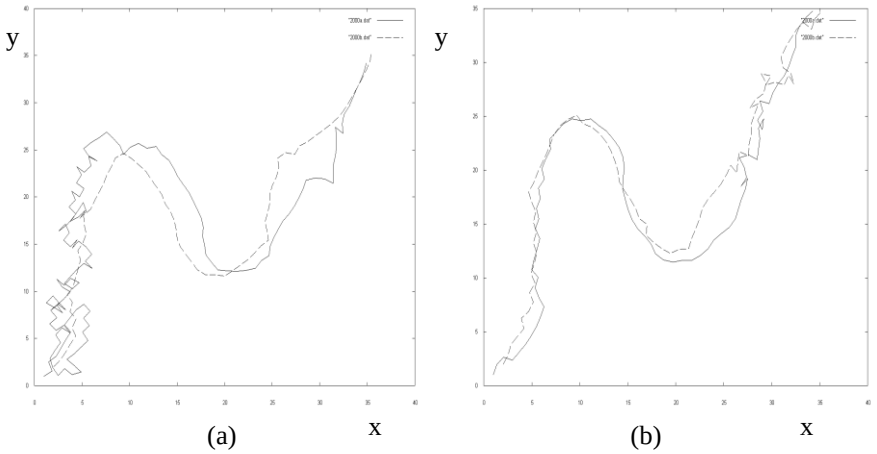


Fig.14. Results of Simulation II using continuous space of input and output. (a) Routes of 2 agents after 2,000 trials (learning iterations) by *individual learning* (Simulation II (i)). (b) Routes of 2 agents after 2,000 trials (learning iterations) by swarm *learning* (Simulation II (i)).

Simulation I: In a non-obstacle exploration environment, multiple agents learn to search the goal as fast as they can, but using 2 methods (i) without any swarm reward (individually) (*individual learning*); (ii) with swarm reward (*swarm learning*).

Simulation II: In a maze-like environment, multiple agents learn to search the goal as fast as they can, but using 2 methods: (i) without any swarm reward (individually) (*individual learning*); (ii) with swarm reward (*swarm learning*).

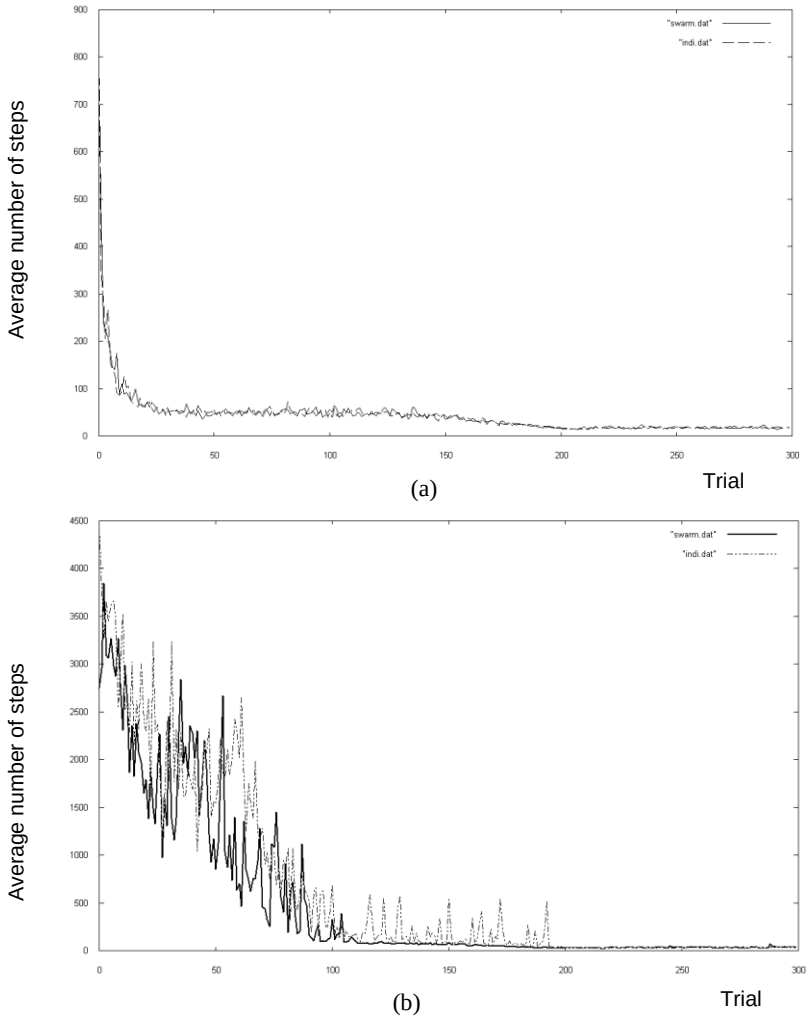


Fig.15. Comparisons of learning convergence using continuous space of input and output in the different environments. (a) The case of *individual learning* (broken line) and *swarm learning* (solid line) in the non-obstacle environment as shown in Fig.5 (in Simulation I). (b) The case of *individual learning* (broken line) and *swarm learning* (solid line) in the maze-like environment as shown in Fig.12 (in Simulation II).

The results of Simulation I are shown in Fig.13, where Fig.13 (a) corresponds to Simulation I (i) and Fig.13 (b) corresponds to Simulation I (ii) respectively. From the comparison, it can be conclude that the proposed system was effective to the continuous space, and *swarm learning* had higher performance than *Individual learning* in the non-obstacle environment, because of the shorten routes which are considered adaptive behaviors in the goal-directed exploration problem.

The results of Simulation II are shown in Fig.14, where Fig.14 (a) corresponds to Simulation II (i) and Fig.14 (b) corresponds to Simulation II (ii) respectively. From the comparison, it can be conclude that the proposed system was effective to complex environment too, and *swarm learning* had more higher performance than *Individual learning* in the more complicated environment, because of the shorten routes which are considered adaptive behaviors in the goal-directed exploration problem.

Fig.15 shows how the number of exploration steps reduced and converged with learning iterations. Fig.15 (a) shows the case of Simulation I (i) and Simulation I (ii) which used the non-obstacle environment. There was no obvious difference between *individual learning* method and *swarm learning* method. Fig.15 (b) shows the case of Simulation II (i) and Simulation II (ii) which used the maze-like environment. Comparing the gradient of learning curves in Fig.15 (a) and (b), it is suggested that s*warm learning* method gave a faster convergence than *individual learning* in the exploration of complicate environment.

The number of membership functions and rules in the continuous state-action space was yielded larger than in the case of discrete space simulations. In Simulation I, 105 membership functions and 2756 rules were generated, and in Simulation II, they were 109 and 2970.

All results of simulations suggest that: (a) the proposed neuro-fuzzy learning system is able to solve state explosion problem of multi-agent system even in the continuous space; (b) swarm formation and adaptive swarm behaviors are able to be acquired just by adding simple reward rules in reinforcement learning algorithm without more formula calculations.

4. Conclusions

To form a swarm and acquire adaptive swarm behaviors in the dynamic unknown environment, a neuro-fuzzy reinforcement learning system was proposed as the internal model of individuals (agents). In the proposed system, the part of Fuzzy net realizes fuzzy inferences to identify the pattern of input state, and the part of *actor-critic* network with TD-error learning algorithm yields adaptive swarm behaviors using rewards concerning with the *Boids* rules. Simulation experiments with different environments of the goal-directed problems were performed and the results showed the effectiveness of Fuzzy net dealing with the unknown states adaptively, and the effectiveness of the proposed learning method with good solution convergence. Robustness of the proposed system was also investigated by the simulations. The future work of this study is expected to avoid using absolute coordinate information but local perspective

information in the environment. This problem is also considered as a Partially Observable Markov Decision Process (POMDP) which is defined that the system dynamics are determined by an MDP but agents can not directly observe the underlying states. This problem may be solved by adopting *eligibility trace*, which is a temporary record of the occurrence of an event such as the visiting of a state or the taking of an action (Sutton and Batto, 1998), into the proposed learning system.

Acknowledgements

We would like to thank Mr. Adachi H. and Mr. Yoneda K. for their work in experiments, and a part of this study was supported by JSPS-KAKENHI (No.18500230, No.20500277 and No.20500207).

References

Craig Reynolds (1986). Boids Background and Update, http://www.red3d.com/cwr/boids/.

Dorigo, M. and Caro, G. D. (1999). Ant Colony Optimization: A New Meta-heuristic, *Proc. 1999 Congress on Evolutionary Computation*, 1470-1477.

Dorigo, M., Maniezzo, V., Colorni, A. (1996). Ant System: Optimization by a colony of cooperating agents, *IEEE Transactions on Systems, Man, and Cybernetics-Part B*, **26(1)**:29-41.

Doya, K. (2002). Metalearning and Neuromodulation, *Neural Networks*, **15(4)**: 495-506.

Iima, H. and Yasuaki, K. (2006). Swarm Reinforcement Learning Algorithm Based on Exchanging Information Among Agents, *Transaction of SICE (in Japanese)*, **42(11)**: 44-1251.

Jouffe, L. (1998). Fuzzy Inference System Learning by Reinforcement Learning, *IEEE Transactions on System, Man and Cybernetics-Part B*, **28(3)**: 338-355.

Kaelbling, L. P. and Littman, M. L. (1996). Reinforcement Learning: A Survey, *Journal of Artificial Intelligence Research*, **4**: 237-285.

Kawakami, T., Kinoshita, M., Watanabe, M., Takatori, N. and Furukawa, M. (2005). An Actor-Critic Approach for Learning Cooperative Behaviors of Multi-agent Seesaw Balancing Problems, *Proc. IEEE International Conference on System, Man and Cybernetics*, 109-114.

Kennedy, J., and Eberhart, R. C. (1995). Particle Swarm Optimization, *Proc. of the IEEE Int. Conf. Neural Networks*, IEEE Press, 1942-1948.

Kennedy, J., Eberhart, R. C., and Shi, Y. (2001). *Swarm Intelligence*. San Francisco, Morgan Kaufmann Publishers.

Kobayashi, K., Mizuno, S., Kuremoto, T., and Obayashi, M. (2005). A Reinforcement Learning System Based on State Space Construction Using Fuzzy ART, *Proc. SICE Annual Conference*, 3653-3658.

Kobayashi, K., Nakano, K., Kuremoto, T., and Obayashi, M. (2008). A State Predictor Based Reinforcement Learning System, *IEEJ Transactions on EIS*, 128(8), 1303-1311.

Kuremoto, T., Obayashi, M., and Kobayashi, K. (2007). Forecasting Time Series by SOFNN with Reinforcement Learning. *Proceedings of the 27th Annual International Symposium on Forecasting* (*ISF2007*), 99

Kuremoto, T., Obayashi, M., and Kobayashi, K. (2008a) Neural Forecasting Systems. *In Reinforcement Learning, Theory and Applications (ed. Cornelius Weber, Mark Elshaw and Norbert Michael Mayer)*, Advanced Robotic Systems, IN-TECH, 1-20.

Kuremoto, T., Obayashi, M., Kobayashi, K., Adachi, H., and Yoneda, K. (2008b). A Reinforcement Learning System for Swarm Behaviors, *Proc. IEEE World Congress on Computational Intelligence (WCCI / IJCNN 2008)*, 3710-3715.

Kuremoto, T., Obayashi, M., Kobayashi, K., Adachi, H., and Yoneda, K. (2008c). A Neuro-Fuzzy Learning System for Adaptive Swarm Behaviors Dealing with Continuous State Space, *Proc. International Conference on Intelligent Computing (ICIC 2008)*, Springer, *LNAI 5227*, 675-683.

Kuremoto, T., Obayashi, M., Yamamoto, A. and Kobayashi, K. (2003). Predicting Chaotic Time Series by Reinforcement Learning, *Proc. 2nd International Conference on CIRAS*, CD-ROM.

Obayashi, M., Kuremoto, T. and Kobayashi, K. (2008). A Self-Organized Fuzzy-Neuro Reinforcement Learning System for Continuous State Space for Autonomous Robots, *Proc. of International Conference on Computational Intelligence for Modeling, Control and Automation (CIMCA 2008)*, 552–559.

Pérez-Uribe, A. (2001). Using A Time-Delay Actor-Critic Neural Architecture with Dopamine-Like Reinforcement Signal for Learning in Autonomous Robots, *LNAI 2036*, 522-533.

Pyeatt, L. D. and Howe, A. E. (2001). Decision Tree Function Approximation in Reinforcement Learning, Proc. 3rd International Symposium on Adaptive Systems: Evolutionary Computation & Probabilistic Graphical Models, 70-77.

Schultz, W. (1998). Predictive Reward Signal of Dopamine Neurons. *The Journal of Neurophysiology*, 80, 1–27.

Schultz, W., Dayan, P., & Montague, R. P. (1997). A Neural Substrate of Prediction and Reward. *Science*, 275, 1593–1599.

Schultz, W. (2001). Reward Signal by Dopamine Neurons. *Neuroscientist*, **7(4)**, 293–302.

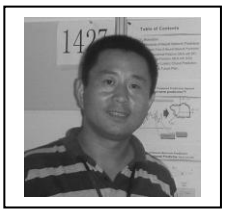
Sutton, R. S. and Barto, A. G. (1998). *Reinforcement Learning: An Introduction*, The MIT Press, Cambridge.

Sycara, K. P. (1998). Multiagent Systems, *Artificial Intelligence Magazine*, Summer 1998, 79-92B.

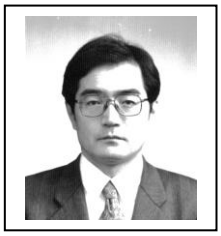
Wang, N., Gao, Y., Chen, Z. Q., Xie, J. Y., and Chen, S.F. (2007). A Two-layered Multi-agent Reinforcement Learning Model and Algorithm, *Journal of Network and Computer Applications*, **30(4)**, 1366-1376.

Wang, X. S., Cheng, Y. H. and Yi, J. Q. (2007). A Fuzzy Actor–Critic Reinforcement Learning Network, *Information Sciences*, **177**: 3764-3781.

Waelti, P., Dickinson, A., & Schultz, W. (2001). Dopamine Responses Comply with Basic Assumptions of Formal Learning Theory. *Nature*, 412, 43–48.



TAKASHI KUREMOTO, received the B.E. degree in System Engineering at University of Shanghai for Science and Technology, China in 1986, and M.E. degree in Computer Science and Engineering at Graduate School of Science and Engineering of Yamaguchi University, Japan in 1996. He worked as a system engineer at Research Institute of Automatic Machine of Beijing from 1986 to 1992 and is currently an assistant professor in Division of Computer Science & Design Engineering, Graduate School of Science and Engineering at Yamaguchi University, Japan. His research interests include bioinformatics, machine learning, complex systems, forecasting and swarm intelligence. He is a member of IEICE, IEEE and IIF. Email: wu@yamaguchi-u.ac.jp HP: *http://www.nn.csse.yamaguchi-u.ac.jp/home/wu*



MASANAO OBAYASHI, received the B.S. degree in electrical engineering from Kyushu Institute of Technology, Japan, in 1975, the M.S. and the PhD degrees in engineering from Kyushu University, Japan, in 1977 and 1997, respectively. From 1977 to 1989, he was with Hitachi Ltd.. From 1989 to 1998, he was with Kyushu University, where he was a Research Associate and an Associate Professor. From 1998 he is with Yamaguchi University, Japan, since July 2001 he has been a Professor in the Faculty of Engineering, Yamaguchi University. He is now with the Graduate school of Science and Engineering of the same university. His current research interests are intelligent computing and intelligent control. Dr. Obayashi is a member of SICE, IEEJ, JSAI, JSFTII and IEEE. Email: m.obayas@yamaguchi-u.ac.jp HP: *http://www.nn.csse.yamaguchi-u.ac.jp/obayashi*

KUNIKAZU KOBAYASHI, received B.E. and M.E. degrees in electronic engineering and computer science from Yamaguchi University, Japan in 1991 and 1993, respectively. He is currently an assistant professor in Division of Computer Science & Design Engineering, Graduate School of Science and Engineering at Yamaguchi University. His research interests include wavelet neural networks, Bayesian method, reinforcement learning, and multi-agent system. He is a member of IEEE and IEICE. Email: koba@yamaguchi-u.ac.jp HP: *http://www.nn.csse.yamaguchi-u.ac.jp/k*