18

Multi-robot Information Fusion and Coordination Based on Agent

Bo Fan and Jiexin Pu
Electronic Information Engineering College
Henan University of Science & Technology, 471003 Luoyang
P.R.China

1. Introduction

With the rapid progress of modern science and technology, robots' development and application extend ceaselessly. For different tasks and environment, especially to large complex tasks and environment, the problems of capability limitation of single robot become more and more obvious, such as information gathering and processing, controlling and so on. Therefore, the multi-robot coordination and cooperation are required to improve these abilities.

Nowadays, the multi-robot system has been paid much attention. It has the prominent property of multidisciplinary intercrossing and fusion. Based on multi-agent system, and with the introduction of multi-agent system's architecture, coordination and cooperation, this dissertation gives a thorough and systematic research on the information fusion of multi-robot system, the tasks distribution and programming of multi-robot coordination, and the multi-robot under oppositional environment.

2. Multi-agent decision fusion based on evidence reasoning and its application to multi-robot system

The overall goal of the research described in this section is to investigate a problem of distributed artificial intelligence introducing evidence theory for decision making in uncertain environment. Such problems arise, for example, in multi-sensor information fusion systems, and in multi-agent decision systems and in the situation assessment problem with distributed artificial intelligence.

The evidence reasoning is applied effectively to intelligent decision systems and specialist systems, which not only accords with people's reasoning manner and decision process but also gives the rational explanation based on information theory to reasoning. In this paper, evidence theory is introduced to the multi-agent system and the Multi-Agent Decision System Based on the Evidence Reasoning is presented. This system is composed of agents, fusion center and decision center. The distributed agents do not interact with environment and do not communicate with each other but only with a fusion center. Each agent extracts the feature information and educes the basic beliefs about local environment. The fusion center combines all agents' beliefs and acquires the beliefs about global environment. The decision center produces the pignistic probabilities that are used to make decision.

2.1 Distributed decision system

a. System architecture

A general multi-agent distributed system consists of a group of distributed intelligent agents that have to coordinate their knowledge, goals, skills, and plans in order to make decisions, take actions, and solve problems. Agents in a distributed system may have different areas of expertise, specific a priori knowledge, and different decision functions. They may be able to observe only certain characteristics of the environment; they may observe different spatio-temporal regions of the environment. Since no one agent has complete information about the environment and observations, they have to cooperate to achieve their goals. There have been many coordination schemes developed in the field of distributed artificial intelligence (Galina Rogova & Pierre Valin., 2005).

In this paper, we investigate a problem of decision making in a hierarchical multi-agent system. Agents do not communicate with each other but only with a fusion center. Each agent has its own internal structure including domain knowledge, a set of hypotheses to be considered, and the basic belief assignment for each hypothesis. They transmit these beliefs to the fusion center. The fusion center combines each agent's beliefs and produces the combined beliefs about the global environment. The decision center acquires pignistic probabilities of the hypotheses under consideration and maps them into actions. Based on this idea, we present the distributed decision system architecture in Figure 1.

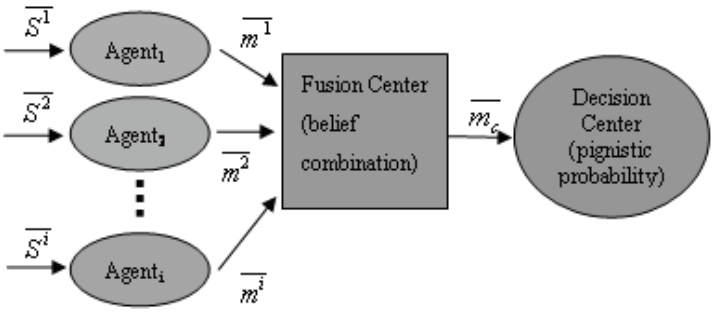


Fig. 1. Distributed decision system architecture

b. Agent model

Each agent i ($1 \le i \le I$, I is the number of agent) is able to observe states of environment and extracts a particular type of information from it, which represented by a feature vector $\overline{S}^i = (s_1^i, s_2^i, ..., s_{N_i}^i)$ (Ni is the dimension of a feature vector. Let $\Theta = \{\theta 1, \theta 2, ,..., \theta n\}$ be a frame of discernment, where θ_k (k=1,…,n) is the element of the frame of discernment and belongs to class k. of the hypothesis.

For the feature vector which each agent i extract from environment and each class k, a proximity measure function $\Phi(\overline{S}^i, \theta_k)$ is defined to map the evidence (feature vector) to the basic belief assignment of the element of Θ. $\Phi(\overline{S}^i, \theta_k)$ is a decreasing function, $0 \le \Phi(\overline{S}^i, \theta_k) \le 1$, which can be considered as a simple support function with focal element θ_k.

$$m_k^i(\theta_k) = \Phi(\overline{X^i}, \theta_k) \tag{1}$$

$$m_k^i(\Theta) = 1 - \Phi(\overline{X^i}, \theta_k) \tag{2}$$

$$m_k^i(A) = 0 \qquad \forall A \neq \theta_k \subset \Theta \tag{3}$$

Combining all the m_k^i according to the Dempster rule of combination (Dempster, A. P., 1967), the basic belief assignment of agent i can be obtained (Shafer, G., 1976):

$$m^i(\theta_k) = \frac{m_k^i \prod_{j \neq k} (1 - m_j^i)}{\sum_k m_k^i \prod_{j \neq k} (1 - m_j^i) + \prod_j (1 - m_j^i)} \tag{4}$$

Each agent i extracts the feature vector from environment as the input, produces the simple support function of each hypothesis of Θ by the measure function Φ, and acquires its basic belief assignment using (4) as the output: $m^i(\theta_1), m^i(\theta_2), ..., m^i(\theta_n), m^i(\Theta)$. This is shown in Figure 2.

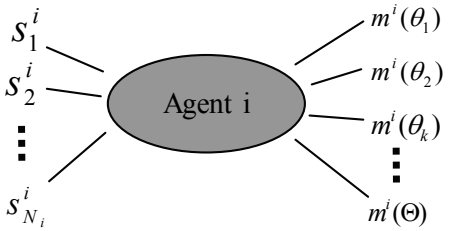


Fig. 2. Agent model

c. Fusion Center

The function of the fusion center is to combine beliefs of all the agents in all the hypotheses with the Dempster rule of combination. However, the fusion center confront with two problems: how to deal with the agents' beliefs when they have the different internal structure, which means that the agents have the different and compatible frame of discernment; and how to resolve the relative reliability of different agents.

In our system (Figure 1.), the fusion center is able to combine the various kinds of the evidence. However, the Dempster rule of combination is based on the single frame of discernment, which obviously can handle one part of the information. In some situation, we have to appeal to the different conception and change the frame of discernment for evidence reasoning in order to solve the some special kinds of evidence. Coarsening and refinement are the transform methods to satisfy this demand. If the agents have different internal structure, i.e. the different and compatible frame of discernment, we can resolve their beliefs combination by coarsening and refinement.

$Agent_i$ and $agent_j$ are assumed to own the different and compatible frame of discernment $\Theta = \{\theta_1^i, \theta_2^i, ..., \theta_n^i\}$ and $\Omega = \{\theta_1^j, \theta_2^j, ..., \theta_m^j\}$. The map, $\sigma: 2^\Theta \to 2^\Omega$, is a refining from Θ to Ω, then: $\forall A \subset \Theta, B \subset \Omega \quad \sigma(A) = B$

Let m_1 and m_2 are the basic belief assignments of Θ and Ω, they can be transformed and transferred as follow:

$$\forall B \subset \Omega \qquad m_2(B) = \sum_{A \subset \Theta : \sigma(A) = B} m_1(A) \tag{5}$$

where the sum is 0 when no A satisfies the constraint. m_2 is called the vacuous extension of m_1 on Ω (Philippe Smets., 2005).

Reliability, i.e. our opinion about the 'value' of a agent, various from agent to agent. The idea is to weight more heavily the information produced by the 'best' agents and conversely for the 'bad' ones. For α [0, 1], let (1-α) be the degree of 'confidence' we assign to the agent. It can be encoded into a basic belief assignment defined on the set {reliable, unreliable} such that (Elouedi, Z. et al., 2004):

$$m(reliable) = 1 - \alpha \tag{6}$$

$$m(unreliable) = \alpha \tag{7}$$

Suppose m is the basic belief assignment on Θ. The result of combining m with the 'confidence' is a new belief, defined as:

$$m^{\alpha}(A) = (1-\alpha) \cdot m(A) \qquad for \qquad A \subset \Theta \tag{8}$$

$$m^{\alpha}(\Theta) = \alpha + (1-\alpha) \cdot m(\Theta) \tag{9}$$

This operation is called a discounting by Shafer and the coefficient α is called the discounting factor. The larger α, the closer mα is from the vacuous belief function.

The fusion center combines the all agent's basic belief assignment to acquire the global environment information. With the coarsening and refinement, the basic belief assignments between the different and compatible frames of discernment are transformed and transferred. Moreover, the fusion center considers the relative reliability of agents, so it distributes the discounting to each agent's basic belief assignment. Let m_i is the basic belief assignment of agent i, and is transferred on the same frame of discernment. Let αi is the discounting factor of agent i. The Dempster rule of combination (Rogova G., 2003) is used to combine the all agent's beliefs:

$$\begin{aligned} m_c(\theta_k) &= c \sum_{\substack{A_i \subset \Theta \\ \bigcap_{i=1}^{I} A_i = \theta_k}} \underset{i=1,2,\cdots,I}{m^{1,\alpha_1}(A_1) \cdot m^{2,\alpha_2}(A_2) \cdots m^{I,\alpha_I}(A_I)} \\ c &= \Big(\sum_{\substack{A_i \subset \Theta \\ \bigcap_{i=1}^{I} A_i \neq \varnothing}} \underset{i=1,2,\cdots,I}{m^{1,\alpha_1}(A_1) \cdot m^{2,\alpha_2}(A_2) \cdots m^{I,\alpha_I}(A_I)} \Big)^{-1} \end{aligned} \tag{10}$$

So the fusion center produces the beliefs in all hypothesis of the frame of discernment $\Theta = \{\theta_1, \theta_2, \ldots, \theta_k\}$: $\{m_c(\theta_1), m_c(\theta_2), \ldots, m_c(\theta_k)\}$ and $m_c(\Theta)$.

d. Decision Center

The agents shown in Figure 1 extract the environment information and derive their corresponding beliefs. The fusion center combines the ensemble of agent beliefs into a set of combined basic probability assignments defined over the frame of discernment. From the

fusion center's output, decision center adopts pignistic probability transformation to compute pignistic probabilities of each hypothesis. The transformation is based on the generalized Insufficient Reason Principle according to which $\forall A \subseteq \Theta$ the mass of belief m(A) is distributed equally among the elements of A (Philippe Smets., 2005).

$$BetP_k = BetP(\theta_k) = \sum_{\substack{\theta_k \in A \\ A \subseteq \Theta}} \frac{m_c(A)}{|A|} \tag{11}$$

2.2 Experiments and results

The approach described in the papers is problem independent and does not impose any restrictions on the kind of features or information used by each agent. We use SimuroSot, one of FIRA simulation game, to evaluate the performance of this decision system.

SimuroSot is FIRA's robot football simulation match. The simulation system provides playground information: robots' position and rotation information of both sides and ball's position information. The strategy system assesses the situation based on the information and makes the responsive game (shown in Figure 3). From analyzing SimuroSot, we think that the strategy system must focus on two types of important information: the state of the ball and the strategy of opponent. The multi-agent decision system based on evidence reasoning of this paper is applied to the match situation assessment of SimuroSot, aiming at the ball state information and opponent's strategy information.

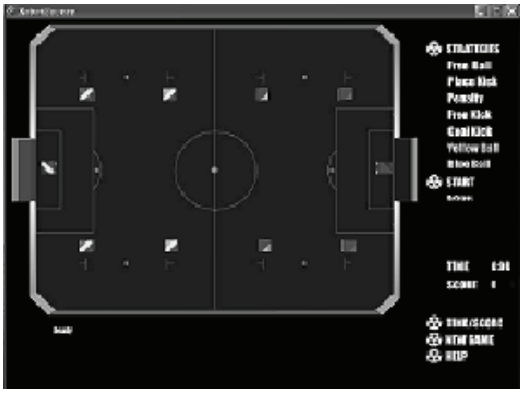


Fig. 3. the interface of SimuroSot platform

The ball state information is observed by agent p and agent d respectively. The two agents have different internal structure.

Agent p extracts ball's position features, the feature vector is ball's position coordinates: $\overline{S_p} = \{x, y\}$, and produces beliefs in each hypothesis of Θ_p = {threat, sub-threat, sub-good, good}, which is the frame of discernment of agent p. Base on the playground's property, there are four representative vector defined: $\omega_p^1, \omega_p^2, \omega_p^3, \omega_p^4$. So the measure function is as follow:

$$\Phi(\overline{S_p}, \theta_p^k) = \exp(-\gamma^k(d^k)) \tag{12}$$

where k=1,2,3,4, $\gamma^k > 0$, $d^k = \left\| \overline{S_p} - \omega_p^k \right\|$

Agent d extracts ball's movement direction features, the feature vector is ball's direction angle: $\overline{S_d} = \{\phi\}$, and produces beliefs in each hypothesis of Θ_d = {towards us, backwards us}, which is the frame of discernment of agent d. The goals of both sides are considered as reference point. Let the angle from ball's direction to goal home is φh, $0^o \le \phi_h \le 180^o$, and to goal opponent is φo, $0^o \le \phi_o \le 180^o$. We can define the measure function:

$$\Phi(\overline{S_d}, \theta_k^d) = \tfrac{1}{2}(\cos\phi_k + 1) \tag{13}$$

The fusion center takes the two agent's beliefs as input. The fusion center confirms the combination beliefs frame of discernment, Θf = {threat, sub-threat, sub-good, good}. So the beliefs of agent d must be transferred. Let $\sigma_1 : 2^{\Theta_d} \to 2^{\Theta_f}$ is a refining from Θ_d to Θ_f. During the transform, several rules are added according to the need of application, which is based on the area (area1, area2, area3) of ball to judge.

if ball is area1,then $\sigma_1(\{towards\ us\}) = \{threat\}$ *and* $\sigma_1(\{backwards\ us\}) = \{sub\text{-}threat, sub\text{-}good, good\}$,
if ball is area2, then $\sigma_1(\{towards\ us\}) = \{threat, sub\text{-}threat\}$ *and* $\sigma_1(\{backwards\ us\}) = \{sub\text{-}good, good\}$,
if ball is area3, then $\sigma_1(\{towards\ us\}) = \{threat, sub\text{-}threat, sub\text{-}good\}$ *and* $\sigma_1(\{backwards\ us\}) = \{good\}$,
also $\sigma_1(\Theta_d) = \Theta_f$.

The fusion center also considers the two agents' belief reliability. Let αp and αd are the discounting factor of agent p and agent d, respectively. During the simulation, we design that, when ball is in middle-ground, $\alpha_p = 0.6$ and $\alpha_d = 0.1$; when ball is near the base line, $\alpha_p = 0.1$ and $\alpha_d = 0.8$

With the Dempster rule of combination, the fusion center leads to the basic probability assignment for hypotheses under consideration. The decision center computes pignsitic probabilities of each hypothesis based on combined beliefs. We can use these probabilities to judge the ball states.

Figure 4 describes the five different game states information. If the right side is us, the decision of game state is made by pignistic probability: $BetP_1$(threat), $BetP_2$(sub-threat), $BetP_3$(sub-good) and $BetP_4$(good). Table 1 shows that only agent p is used to observe the match and receive the game state. Table 2 shows the game state with the fusion of the observation of agent p and agent d. We can find that the distributed system makes the fusion of each agent's information and produces the more accuracy and more effective game state.

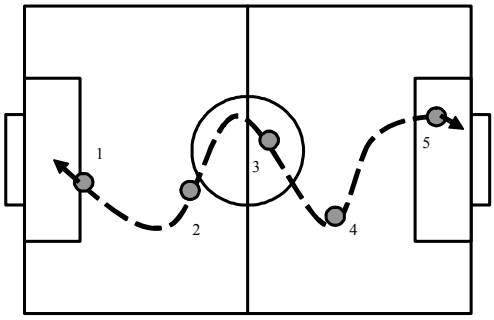


Fig. 4. Ball state: five cases (● is ball)

Game State	Pignistic probability			
	$BetP_1$	$BetP_2$	$BetP_3$	$BetP_4$
case 1	0.072856	0.159931	0.224461	0.542753
case 2	0.058579	0.220646	0.617479	0.103296
case 3	0.021759	0.856901	0.103349	0.017991
case 4	0.203953	0.457638	0.251437	0.086973
case 5	0.639409	0.172916	0.126939	0.060736

Table 1.

Game State		Pignistic probability			
		$BetP_1$	$BetP_2$	$BetP_3$	$BetP_4$
case 1	L	0.051437	0.108067	0.147387	0.693109
	R	0.083047	0.183288	0.258486	0.475180
case 2	L	0.078714	0.081742	0.617399	0.222144
	R	0.216245	0.505598	0.139411	0.138746
case 3	L	0.119005	0.123862	0.579351	0.177782
	R	0.054978	0.837930	0.054657	0.052435
case 4	L	0.125516	0.125686	0.506824	0.241973
	R	0.241554	0.596728	0.081055	0.080663
case 5	L	0.506455	0.240708	0.172812	0.080025
	R	0.636418	0.176342	0.127544	0.059696

Table 2. (L and R are two direction)

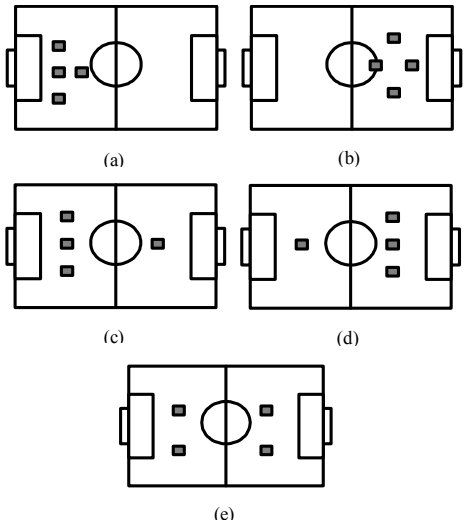


Fig. 5. Five kinds of opponent strategy formation(is opponent robot)

Opponent's strategy is the other focus. Each robot's information of opponent team has to be obtained. With the information, opponent's formation and decision are estimated. This is the important foundation of our decision-making. On basis of application to SimuroSot, the four agents are designed to observe the opponent's four robots except the goalkeeper. These agents have a common internal structure including domain knowledge, a set of hypothesis to be considered, and a procedure for assigning a level of belief to each hypothesis. Let Θr = {attack, balance, defense} be the frame of discernment. Each agent produces beliefs from opponent's robot position information, the feature vector is opponent robot's position coordinates: $\overline{S_i} = \{x_i, y_i\}$. The fusion center combines beliefs of the four agents in all the hypotheses with the Dempster rule and computes combined beliefs. Based on them, the decision center produces pignistic probabilities of each hypothesis, so that opponent strategy is estimated.
The five typical strategy formations is shown in Figure 5. If the right side is us, we can judge the opponent's strategy using decision center's output, pignistic probability: $BetP_1$(attack), $BetP_2$(balance) and $BetP_3$(defense), shown in Table 3.

Opponent strategy formation	Pignistic probability		
	$BetP_1$	$BetP_2$	$BetP_3$
formation a	0.150022	0.849978	0.849978
formation b	0.814607	0.185393	0.185393
formation c	0.282619	0.717381	0.717381
formation d	0.634057	0.365943	0.365943
formation e	0.470250	0.529750	0.529750

Table 3.

2.3 Summary

The method of distributed decision based on TBM is presented in this paper. In the system architecture, the agents may be homogeneous, or may be heterogamous and do not communicate with each other. Each agent is independent and does not be imposed any restrictions on the kind of features or information used by itself. The fusion center integrates the information provided by all agents and computes the global environment beliefs. The decision center uses the probability transformation to produce the final conclusion.
In the application, we find that there are two key factors of the system operation performance: how to choose the frame of discernment and how to extract the features from the environment. Therefore, the multi-agent decision system of this paper should depend on the practicality characters to design agent's extracting features and to choose appropriate frame of discernment. The decision system must aim at the property of application and assign the rational discount factors to agents to improve the accuracy and effectively of this system.

3. Multi-agent learning reward function based on knowledge

Markov games and reinforcement learning are main research methods to Multi-Agent System (MAS) ((M. L. Littman., 2001) & (Bowling M.; Veloso M., 2004)). Markov game can

fit for dealing with MAS coordination and building the dynamic model of multi-agent's interaction. Reinforcement learning is an interactive learning. Q-learning (C. J. C. H. Watkons & P. Dayan., 1992), one of reinforcement learning, is the dynamic programming learning based on Markov Decision Process (MDP). Applied to multi-agent system, reinforcement learning is extended to Markov games. Although reinforcement learning is focused on widely by its convergence and biology relativity, it does not work well in practice, especially to application to robot.

Reinforcement learning does not provide the mapping of state and action. After choosing an action, an agent is signaled the effect but do not know which is the optimal. Therefore, the agent depends on the interacting with environment to collect states, actions, the transition of states and reward in order to get the optimal policy. However, in practice, the reward received from environment is not immediate, but delayed, so learning becomes more difficult within the time-limited. Currently, how to design the reinforcement function, which maybe the most difficult to reinforcement learning, is seldom discussed. M. J. Matalic designs the reinforcement function by reinforcing multiple goals and using progress estimators (M. J. Mataric, 2001). Kousuk INOUE presents reinforcement learning is accelerated by using experience information to distill the general rules (Kousuke INOUE, et al., 2000). L. P. Kaelbling decomposes the learning task and combines the prior knowledge to direct reinforcement learning (W. D. Smart & L. P. Kaelbling., 2002).

Most research on reinforcement function is depended on application environment. Based on environment's property, learning system reduces complexity and introduces relative information to enrich reinforcement function so that the learner can obtain more knowledge about environment and itself to process reinforcement learning by. We have the same opinion. We design reinforcement function including two aspects: the information of goal state and the agent's action effect. The former is provided by environment, which is the interaction information between an agent and environment for accomplishing the special task, the latter is that the agent evaluates action effect, which depends on its domain knowledge of action ability. In this paper, the simulation game of Robot Soccer is applied.

3.1 Multi-agent reinforcement learning

In this section some basic principle of MDP is reviewed and then the formalisms of Markov games, which is an extension of MDPs. Q-learning algorithm is also presented, which is used to solve MDPs. Minmax-Q algorithm is proposed to solve Markov games.

a. MDP and Q-learning Algorithm

Let us consider a single agent interacting with its environment via perception and action. On each interaction step the agent senses the current state s of the environment, and chooses an action to perform. The action alters the state s of the environment, and a scalar reinforcement signal r (a reward or penalty) is provided to the agent to indicate the desirability of the resulting state.

Formally, a MDP is represented by a 4-tuple <S, A, r, T>: S is a set of states, A is a set of actions, r is a scalar reinforcement function, r: $S \times A \rightarrow R$, T is a state transition function, T: $S \times A \rightarrow S$.

The goal of the agent in the most common formulation of the reinforcement learning problem is to learn an optimal policy of actions that maximizes an expected cumulative sum of the reinforcement signal for any starting state. The task of an agent with RL is thus to learn a policy π: $S \rightarrow A$ that maps the current state s into the desirable action a to be performed in s.

The action policy π should be learned through trial-and-error interaction of the agent with the environment, which means that the learner must explicitly explore its environment.
There is at least one optimal policy π* that is stationary and deterministic. One strategy to learn the optimal policy π* when the model (T and r) is not known in advance is to allow the agent to learn the evaluation function Q: S×A→R. Each Q(s, a) value (or action value for pair s, a) represents the expected cost incurred by the agent when taking action at state s and following an optimal policy thereafter.
Q-learning algorithm iteratively approximates Q, provided the system can be modeled as a MDP, the reinforcement function is bounded, and actions are chosen so that every state-action pair is visited an infinite number of times. Q-learning rules (C. J. C. H. Watkons & P. Dayan, 1992) are:

$$Q(s,a) \leftarrow Q(s,a) + \alpha(r + \beta \max_{a'} Q(s',a') - Q(s,a)) \tag{14}$$

Where s is the current state, a is the action performed in s, r(s, a) is the reinforcement received after performing a in s, s′ is the new state, β is a discount factor ($0 \le \beta < 1$) and a is the learning rate ($\alpha > 0$).

b. MDP and Q-learning Algorithm Markov games and Minmax-Q Algorithm

Let us consider a specialization of Markov games, which consists of two agents performing actions in alternating turns, in a zero-sum game. Let A be the set of possible actions that the playing agent A can choose from, and O be the set of actions for the opponent player agent O. r(s, o, a) is the immediate reinforcement agent A receives for performing action a∈A in state s∈S when its opponent agent O performs action o∈O.
The goal of agent A is to learn an optimal policy of actions that maximizes its expected cumulative sum of discounted reinforcements. However, learning this policy is very difficult, since it depends critically on the actions the opponent performs. The solution to this problem is to evaluate each policy with respect to the opponent's strategy that makes it look the worst [1]. This idea is the core of Minmax-Q algorithm, which is essentially very similar to Q-learning algorithm with a minmax replacing the max function in the definition of the state value.
For deterministic action policies, the value of a state s∈S in a MG is:

$$V(s) = \max_{a \in A} \min_{o \in O} Q(s,a,o) \tag{15}$$

And Minmax-Q learning rule is:

$$Q(s,a,o) \leftarrow Q(s,a,o) + \alpha[r(s,a,o) + \beta V(s') - Q(s,a,o)] \tag{16}$$

where s is the current state, a is the action performed by agent A in s, o is the action performed by agent O in s, Q(s, a, o) is the expected discounted reinforcement for taking action a when Agent O performs o in state s, and continuing the optimal policy thereafter, r(s, o, a) is the reinforcement received by agent A, s′ is the new state, β is a discount factor ($0 \le \beta < 1$) and α is the learning rate ($\alpha > 0$).
For non-deterministic action policies, a more general formulation of Minmax-Q has been formally defined elsewhere.

3.2 Reinforcement function based on knowledge

Reinforcement learning systems learn a mapping from situations to actions by trial-and-error interactions with a dynamic environment. The goal of reinforcement learning is defined using the concept of a reinforcement function, which is the exact function of future reinforcements the agent seeks to maximize. In other words, there exists a mapping from state/action pairs to reinforcements; after performing an action in a given state the learner agent will receive some reinforcement (reward) in the form of a scalar value. The agent learns to perform actions that will maximize the sum of the reinforcements received when starting from some initial state and proceeding to a terminal state.

Perhaps, design of reinforcement is the most difficult aspect of setting up a reinforcement learning system. The action performed by the learner not only receives an immediate reward but also transits the environment to a new state. Therefore, the learning has to consider both the immediate reward and the future reinforcement caused by the current action. Nowadays, much of reinforcement learning work uses two types of reward: immediate, and very delayed. In reinforcement learning system, immediate reinforcement, when available, is the most effective. And delayed reinforcement requires introduction of sub-goal so that learning is performed within time-limited. Reinforcement learning is a feedback algorithm, so it is not good for long-term goal but more effective to the near goals. A learner can introduce medium-term goals and distributing task so as to accelerate the learning rate and increase the learning efficiency.

In traditional reinforcement learning, the agent-environment interaction can be modeled as a MDP, in which agent and environment are synchronized finite state automata. However, in real-world, the environment and agent states change asynchronously, in response to events. Events take various amounts of time to execute: the same event (as perceived by the agent) can vary in duration under different circumstances and have different consequences. Reinforcement learning gives the reward to what are caused by and in control of the agent.

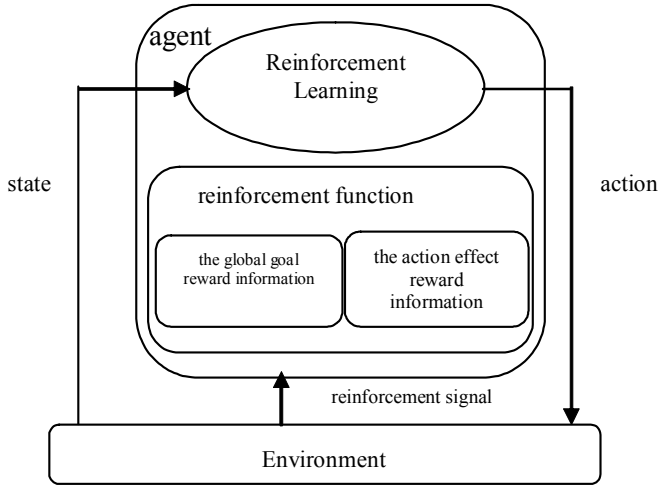


Fig. 5. Learning Model

Instead of encoding knowledge explicitly, reinforcement learning hides it in the reinforcement function which usually employs some ad hoc embedding of the semantics of

the domain. We divide this reinforcement information involving domain knowledge into two types:

- The global goal's reward;
- The agent action effect's reward.

In each state, a learner agent chooses an appropriate action and performs it to environment, which is transited to a new state. On the one hand, the agent depends on the task to judge the environment and receives the global goal reward; on the other hand, based on the agent's domain knowledge about its action ability, it compares performance effect and obtains the reward of action. The reinforcement function that we design combines the global environment's reinforcement and the agent action's reinforcement. The reinforcement learning model with this reinforcement function is shown in Figure 5.

3.3 Experiments

a. Experimental Environment

We adopted the Robot Soccer to perform our experiments. Robot Soccer is a typical MAS: robots is the agents, the playground and ball are thought as environment.

State, actions of multi-agent learning are design as follows:

- S = {threat, sub-threat, sub-good, good};
- Home agents' A = {Shoot, Attack, Defend, More-defend};
- Opponent agents' O = {Shoot, Attack, Defend, More-defend}

In traditional reinforcement learning, reinforcement function is usually developed that reward is +1 if home team scored; reward is -1 if opponent team scored. Instead of it, we design the reinforcement function including two kinds of information: game goal reinforcement and robot's action effect reinforcement.

In play, game goal reinforcement information is the reward received by score of both sides. The rewards signal rs, is defined as:

$$r_s = \begin{cases} c, & our\ team\ scored \\ -c, & opponent\ team\ scored \\ 0, & otherwise \end{cases} \quad c > 0 \tag{17}$$

And, reinforcement information of the robot's action effect is that, after performing each action, the robot receives the reward signal ra, which involves the robot's domain knowledge about each action and evaluates the action effect.

$$r_a = \begin{cases} d & action\ success \\ 0 & action\ unsuccesss \end{cases} \quad d > 0, \tag{18}$$

The combination reinforcement function considers the two kinds of reward and sums them with weighting their values constants appropriately.

$$\begin{aligned} &R = \omega_s \bullet r_s + \omega_a \bullet r_a \\ &\omega_s, \omega_a \geq 0, \quad (\omega_s + \omega_a) = 1 \end{aligned} \tag{19}$$

Thus, the robot agent evaluates its policy using comprehensive reinforcement. With learning continually, the agent improves its policy and increases its ability.

b. Experimental Result

We use SimuroSot, one of simulation match provided by FIRA [11] to conduct the experiments. In experiments, In order to evaluate the effectiveness of the reinforcement function presented in this paper, we compare its performance against traditional reinforcement function.

traditional reinforcement function:

$$R = \begin{cases} +1 & \text{home team scored} \\ -1 & \text{opponent team scored} \end{cases} \tag{20}$$

reinforcement function based on knowledge:

$$R = \omega_s \bullet r_s + \omega_a \bullet r_a \tag{21}$$

There are two group experiments. In experiment 1, the home team uses the conventional Q-learning. In experiment 2, the home team uses the Minmax-Q algorithm of Markov Games. The opponent team uses fix strategy. The team size is 1.

The parameters used in the algorithms were set at: $\beta = 0.9$, initial value of $\alpha = 1.0$, α decline = 0.9. In Q-learning, initial value of Q-table = 0. In Minmax-Q algorithm, initial value of Q-table = 1.

In experiment, we define that the appropriate policy is: $s1 \rightarrow a1, s2 \rightarrow a2, s3 \rightarrow a3, s4 \rightarrow a4$. For Q-learning algorithm, we save several Q-values, which are $Q(s_1,a_1)$, $Q(s_2,a_2)$, $Q(s_3,a_3)$, $Q(s_4,a_4)$. And for Minmax-Q algorithm, we save Q-values, which are $\underset{o\in O}{Min}Q(s_1,a_1,o)$, $\underset{o\in O}{Min}Q(s_2,a_2,o)$, $\underset{o\in O}{Min}Q(s_3,a_3,o)$ and $\underset{o\in O}{Min}Q(s_4,a_4,o)$.

During more than 1000 steps learning, we analyze their results. Q-learning with the two kinds of reinforcement function all can converge to appropriate policy, but the former needs long time. In Minmax-Q algorithm, it can get to appropriate policy with knowledge-base reinforcement function, while it does not learn appropriate policy with traditional reinforcement function even if spending too long time. We choose the same Minmax-Q value to observe.

The results of Q-learning are shown in Figure 6. The results of Minmax-Q are shown in Figure 7. Thereinto, Figure 5(a) and Figure 6(a) are respectively the learning with traditional reinforcement function; Figure 5(b) and Figure 6(b) are respectively the learning with

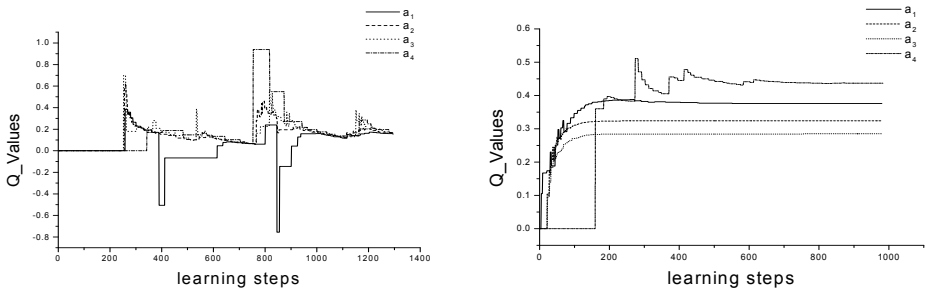


Fig. 6 (a). Q-learning algorithm with the traditional reinforcement function; (b). Q-learning algorithm with the knowledge-base reinforcement function

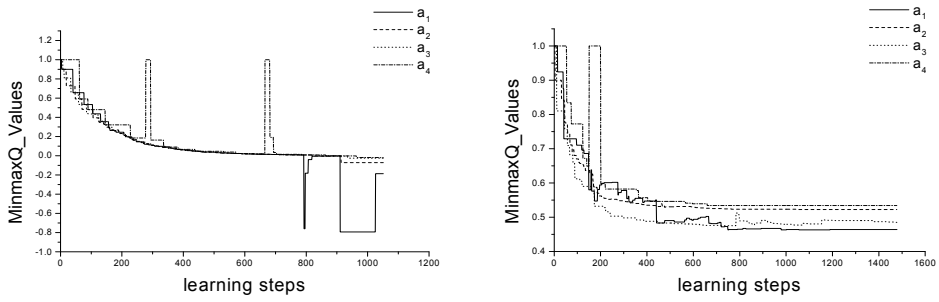


Fig. 7 (a). Minmax-Q algorithm with the traditional reinforcement function; (b). Minmax-Q algorithm with the knowledge-base reinforcement function

knowledge-base reinforcement function. Obviously, we can observe that learning with traditional reinforcement function has worse convergence and still has many unstable factors at end of experiment, while the learning with knowledge-base reinforcement function converges rapidly and it gets to stable value about half time of experiment. Therefore, with the external knowledge (environment information) and internal knowledge (action effect information), multi-agent learning has better performance and effectivity.

3.4 Summary

When Multi-agent learning is applied to real environment, it is very important to design the reinforcement function that is appropriate to environment and learner. We think that the learning agent must take advantage of the information including environment and itself domain knowledge to integrate the comprehensive reinforcement information. This paper presents the reinforcement function based on knowledge, with which the learner not only pays more attention to environment transition but also evaluates its action performance each step. Therefore, the reinforcement information of multi-agent learning becomes more abundant and comprehensive, so that the leaning can converge rapidly and become more stable. From experiment, it is obviously that multi-agent learning with knowledge-base reinforcement function has better performance than traditional reinforcement. However, we should point out, how to design the reinforcement must depend on the application background of multi-agent learning system. Different task, different action effect and different environments are the key factors to influence multi-agent learning. Hence, differ from traditional reinforcement function; the reinforcement function is build by the characteristic based on real environment and learner action.

4. Distributed multi-agent reinforcement learning and its application in multi-robot

Multi-agent coordination is mainly based on agents' learning abilities under distributed environment ((Yang, X. M. Li, & X. M. Xu, 2001), (Y. Chang, T. Ho, & L. P. Kaelbling, 2003), (Kok, J. R. & Vlassis, N., 2006)). In this section, a multi-agent coordination based on distributed reinforcement learning is proposed. In this way, a coordination agent decomposes the global task of system into several sub-tasks and applies the central reinforcement learning to distribute these sub-tasks to task agents. Each task agent uses the individual reinforcement learning to choose its action and accomplish its sub-task.

4.1 Distributed reinforcement learning of MAS

Currently, research on distributed reinforcement learning of MAS mainly includes the central reinforcement learning (CRL), the individual reinforcement learning (IRL), the group reinforcement learning (GRL) and the social reinforcement learning (SRL) (Zhong Yu; Zhang Rubo & Gu Guochang, 2003).

The CRL aims at the coordinating mechanism of MAS and adopts the standard reinforcement learning algorithm to accomplish an optimal coordination. The distributed problem of the system is focused on and resolved by learning centrally. In a CRL, the whole state of MAS is the input and the action assignment of every agent is the output. The agents in CRL system are not the learning unit but an actuator unit to perform the orders of the learning unit passively. The structure of CRL is shown in Figure 8.

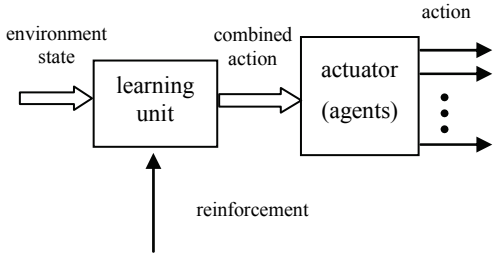


Fig. 8. the structure of CRL

In IRL, all agents are the learning units. They perceive the environment state and choose the actions to receive the maximized reward. An IRL agent does not care about other agents' states and only considers its reward to choose the action, so it is selfish and the learning system has difficulty in attaining the global optimal goal. However, the IRL has strong independence and is easy to add or reduce the agents dynamically. Also the number of agents has less effect on learning convergence. The structure of IRL is shown in Figure 9.

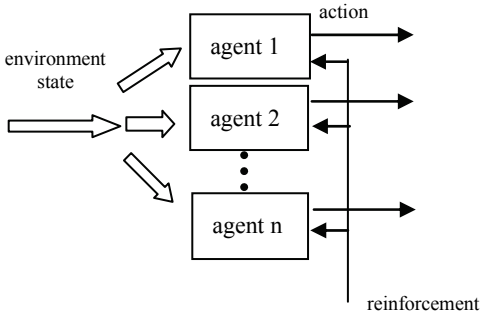


Fig. 9. the structure of IRL

The GRL regards all agents' states and actions as the combined states and actions. In a GRL, the Q-table of each agent maps the combined states into the combined actions. A GRL agent must consider other agents' states and choose its action based on the global reward. The GRL has an enormous state space and action space, so it would learn much more slowly as the number of agents grew, which is not feasible. The structure of GRL is shown in Figure 10.

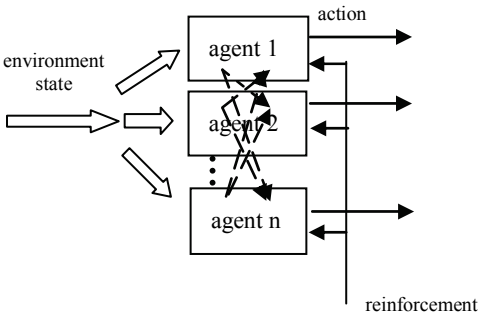


Fig. 10. the structure of GRL

SRL is thought as the extension of IRL. It is the combination of IRL, social models and economical models. The SRL simulates the individual interaction of human society and builds the social model or economical model. In SRL, the methodology of management and sociology is introduced to adjust the relation of agents and produces more effective communication, cooperation and competition mechanisms so as to attain the learning goal of the whole system.

4.2 Multi-agent coordination based on reinforcement learning

In this section, the multi-agent coordination based on distributed reinforcement learning is proposed, which is shown in Figure 11. This coordination method is a hierarchical structure: coordination level and behavioral level. The complicated task is decomposed and distributed to the two levels for learning.

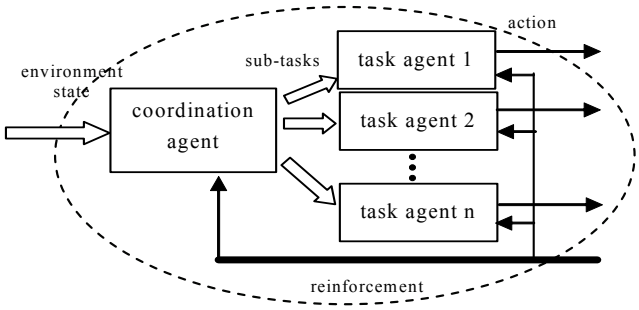


Fig. 11. the structure of multi-agent coordination based on distributed reinforcement learning

a. Coordination Level

Coordination level decomposes the complicated task into several sub-tasks firstly. Let $P=\{p_1,p_2,\ldots,p_m\}$ be a set of strategies of coordination agent, where $p_i\ (1\le i\le m)$ is the element of the set of strategies and corresponds to the assignment of sub-tasks. Based on the environment state, coordination agent adopts CRL to choose the appropriate strategy and distributes the sub-tasks to task agents. The update for coordination agent's Q function can be written:

$$Q_p(s,p) \leftarrow (1-\alpha_p)Q_p(s,p) + \alpha_p\left[r_p + \beta \max_{p' \in P} Q_p(s',p')\right] \tag{22}$$

where s is the current state, p is the strategy chosen by coordination agent in s, rp is the reward signal received by coordination agent, s′ is the next state, α_p is the learning rate of coordination agent, β is the discount factor.

b. Behavioral Level

In behavioral level, all task agents have a common internal structure. Let A be the action set of task agents. Each sub-task corresponding an action sub-set, $SA_k \subseteq A$, is assigned to a task agent. According to the sub-task, each task agent k ($1 \le k \le n$) adopts the IRL to choose its action, $a^k \in SA_k$, and performs it to environment. The update for Q function of task agent k is written:

$$Q^k(s,a^k) \leftarrow (1-\alpha^k)Q^k(s,a^k) + \alpha^k\left[r^k + \beta \max_{a^{k\prime} \in st_k} Q_p(s',a^{k\prime})\right] \tag{23}$$

where s is the current state, ak is the action performed by task agent k in s, rk is the reinforcement signal received by task agent k, s′ is the next state, αk is the learning rate of task agent k, β is the discount factor.

c. Reinforcement assignment

The reinforcement assignment is that the reinforcement signal received from environment is assigned to all agents in distributed system according to the effective method. In this paper, we design a heterogeneous reinforcement function: global task reinforcement and sub-tasks' coordination effect reinforcement.

Coordination agent is responsible to decide the high-level strategies and focuses on the global task achievement. Simultaneously, it arranges the sub-tasks to all task agents. So its reinforcement information includes both the global task and sub-tasks' coordination effect. All task agents coordinate and cooperate so as to take their actions to accomplish the high-level strategies. So their learning is evaluated by sub-tasks' coordination effect.

4.3 Experiments and results

The SimuroSot simulation platform [10] is applied to the evaluation of our proposed method. In this simulation platform, the simulation system provides the environment information (ball's and all robots' position information), from which the strategic system makes decision to control each robot's action and perform it to the game.

In the distributed reinforcement learning system, the state set is defined to S = {threat, sub-threat, sub-good, good}. In the coordination level, the strategy set of coordination agent is defined to H = {hard-defend, defend, offend, strong-offend}. In the behavioral level, the action set of task agents is defined to A = {guard, resist, attack, shoot}.

The global goal of games is to encourage home team's scoring and avoid opponent team's scoring. The reward of global goal is defined:

$$r_g = \begin{cases} c, \text{ our team scored} \\ -c, \text{ other team scored} \\ 0, \quad \text{otherwise} \end{cases} \qquad (24)$$
$$c > 0$$

The reinforcement of sub-tasks' coordination effect is to evaluate the home team's strategies, which includes the domain knowledge of each strategy. It is defined:

$$r_a = \begin{cases} d & \text{strategy success} \\ 0 & \text{strategy unsuccess} \end{cases} \qquad (25)$$
$$d > 0,$$

Coordination agent sums the two kinds of reinforcement, weighting their values constants appropriately, so its reinforcement function, R_c, is defined:

$$R_c = \omega \bullet r_g + \upsilon \bullet r_a \qquad (26)$$
$$\omega, \upsilon \geq 0, \quad (\omega + \upsilon) = 1$$

Task agents cooperate and take their actions to accomplish the team strategies. Their reinforcement function, Rm, is defined: $R_m = r_a$

The parameters used in the algorithm are set at : $\beta = 0.9$, initial value of $\alpha = 1.0$, α decline = 0.9, initial value of Q-table = 0.

There are two groups in experiments. The conventional reinforcement learning (group 1) and our proposed distributed reinforcement learning (group 2) are applied to the home team respectively. The opponent team uses random strategy. The team size is 2.

The results of group 1 are shown in Figure 12a and Figure 12b respectively. During the simulation, the convergence of Q-learning has worse performance. Two Robots cannot learn the deterministic action policies.

In group 2, Figure 13a shows the Q-value of the coordination agent, which convergent rapidly. From the Q's maximum, coordination agent can get the effective and feasible result. Figure 13b and Figure 13c describe two Robots' Q values respectively, which are convergent. Robots can get deterministic policy to choose actions.

4.4 Summary

With agents' coordination and cooperation, MAS adopts multi-agent learning to accomplish the complicated tasks that the single agent is not competent for. Multi-agent learning provides not only the learning ability of individual agent, but also the coordination learning of all agents. Coordination agent decomposes the complicated task into sub-tasks and adopts the CRL to choose the appropriate strategy for distributing the subtasks. Task agents adopt the IRL to choose the effective actions to achieve the complicated task. With application and experiments in robot soccer, this method has better performance than the conventional reinforcement learning.

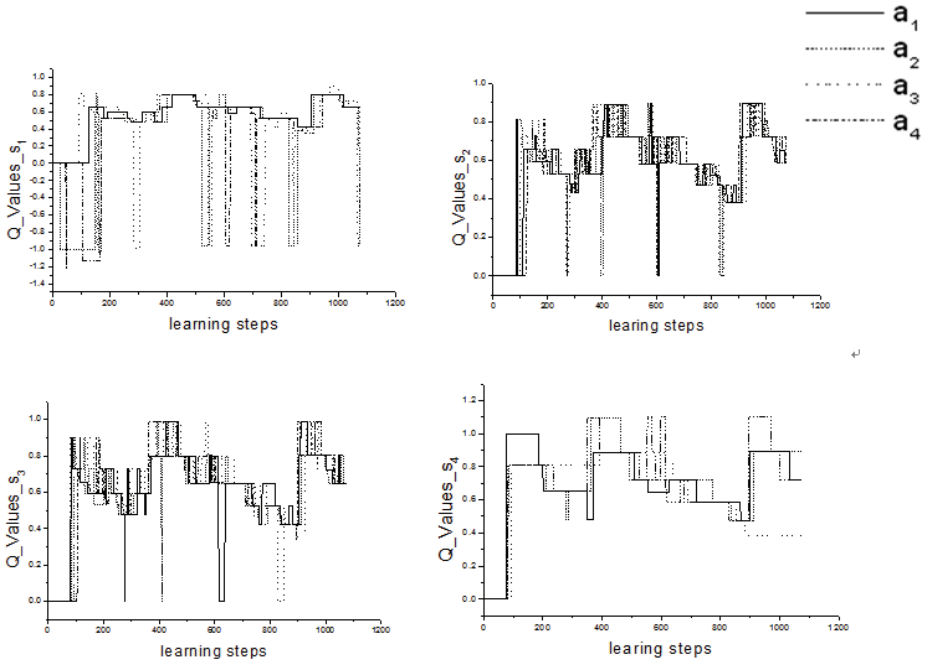


Fig. 12a. Q-values of Robot 1 in group 1

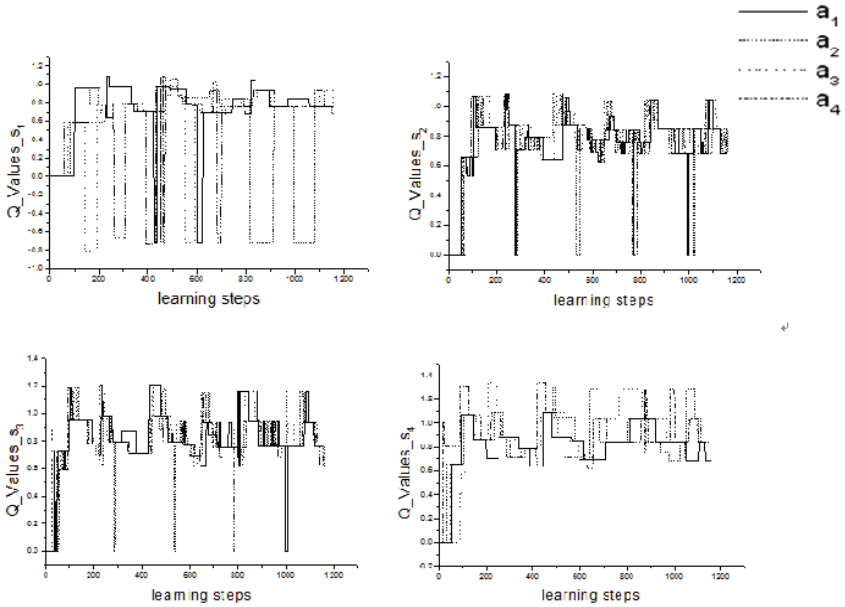


Fig. 12b. Q-values of Robot 2 in group 1

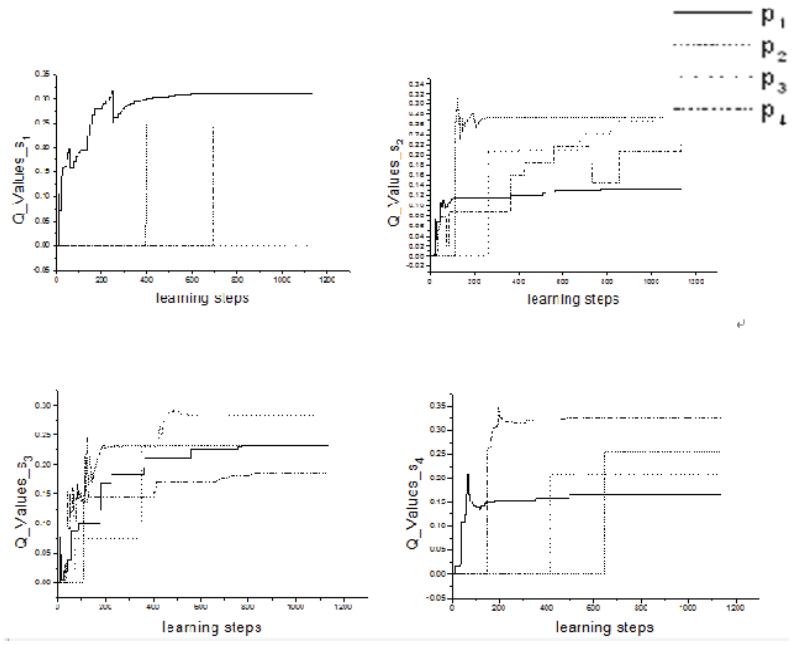


Fig. 13a. Q-values of coordination agent in group 2

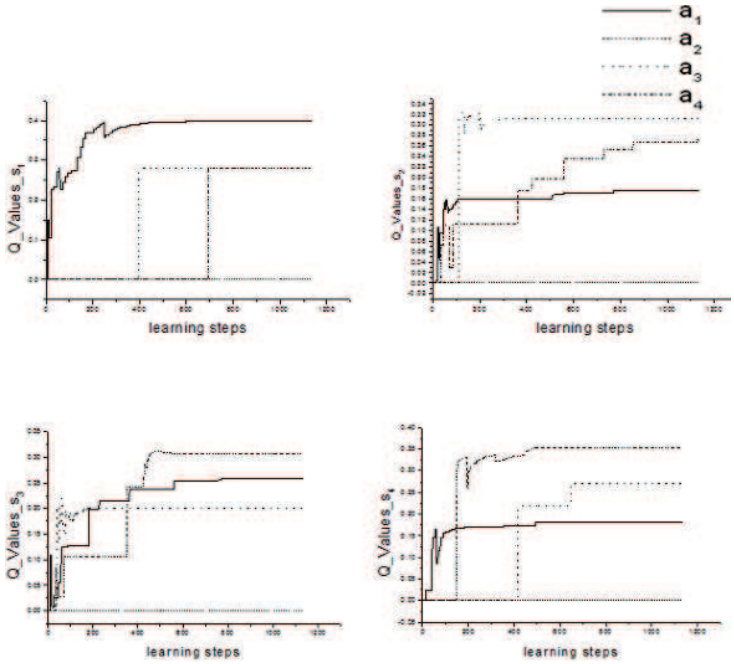


Fig. 13b. Q-values of Robot 1 in group 2

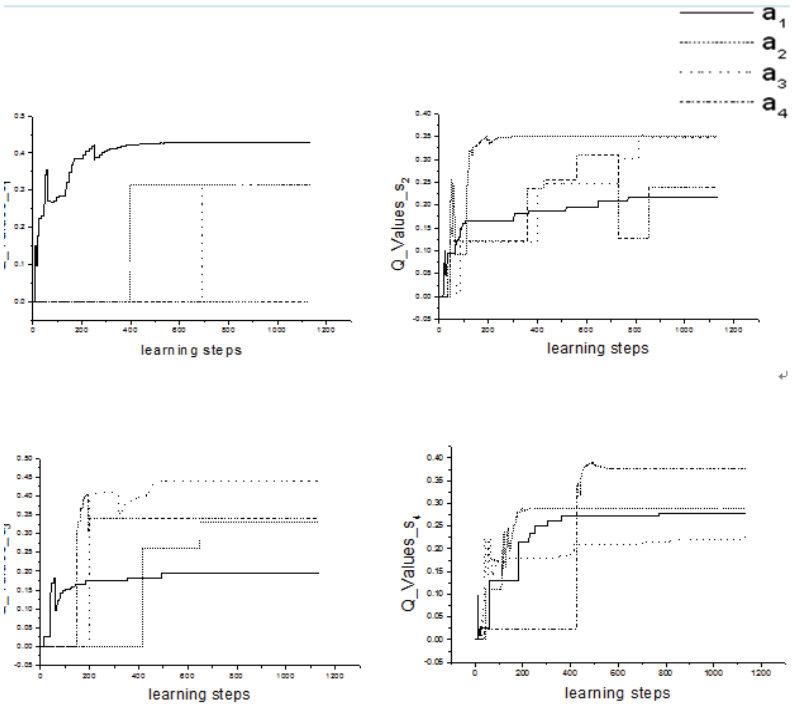


Fig. 13c. Q-values of Robot 2 in group 2

5. Multi-robot coordination framework based on Markov games

The emphasis of MAS enables the agents to accomplish the complicated tasks or resolve the complex problems with their negotiation, coordination and cooperation. Games and learning are the inherence mechanism of the agents' collaboration. On the one side, within rational restriction, agents choose the optimal actions by interacting each other. On the other side, based on the information of environment and other agents' actions, agents adopt the learning to deal with the special problem or fulfill the distributed task.

At present, research on multi-agent learning lacks the mature theory. Littman takes the games as the framework of multi-agent learning (M. L. Littman, 1994). He presents the Minmax Q-learning to resolve the zero-sum Markovgames, which only fit to deal with the agents' competition. The coordination of MAS enables the agents not only to accomplish the task cooperatively, but also to resolve the competition with opponents effectively. On the basis of Littman's multi-agent game and learning, we analyze the different relationship of agents and present a layered multi-agent coordination framework, which includes both their competition and cooperation.

5.1 Multi-agent coordination based on Markov games

Because of the interaction of cooperation and competition, all agents in the environment are divided into several teams. The agents are teammates if they are cooperative. Different agent teams are competitive. Two kinds of Markov games are adopted to cope with the

different interaction: zero-sum games are used to the competition between different agent teams; team games are applied to the teammates' cooperation.

a. Team level: zero-sum Markov games

Zero-sum Markov games are a well-studied specialization of Markov games in which two agents have diametrically opposed goals. Let agent A and agent O be the two agents within zero-sum game. For a∈A , o∈O (A and O are the action sets of agent A and agent O respectively) and s∈S (S is the state set), R1(s, a, o) = - R2(s, a, o). Therefore, there is only a single reward function R1, which agent A tries to maximize and agent O tries to minimize. Zero-sum games can also be called adversarial or fully competitive for this reason.
Within a Nash equilibrium of zero-sum game, each policy is evaluated with respect to the opposing policy that makes it look the worst. Minmax Q-learning (M. L. Littman, 1994) is a reinforcement learning algorithm specifically designed for zero-sum games. The essence of minimax is that behave so as to maximize your reward in the worst case. The value function, V(s), is the expected reward for the optimal policy starting from state s. Q(s, a, o) is the expected reward for taking action a when the opponent chooses o from state s and continuing optimally thereafter.

$$V(s) = \max_{\pi \in PD(A)} \min_{o \in O} \sum_{a \in A} Q(s,a,o)\pi_a \tag{27}$$

The update rule for minimax Q-learning can be written:

$$\begin{aligned} Q(s,a,o) \leftarrow (1-\alpha)Q(s,a,o) \\ + \alpha(r + \beta V(s)) \end{aligned} \tag{28}$$

In MAS, there are several competitive agent-teams. Each of teams has a team commander to be responsible for making decision. Therefore, two teams' competition simplifies the competition between two Team-commanders, which adopt the zero-sum Markov games.

b. Member level: team Markov games

In team Markov games, agents have precisely the same goals. Supposed that there are n agents, for a1∈A1, a2∈A2,…, an∈An, and s∈S, R1(s, a1, a2,…,an) = R2(s, a1, a2,…,an) = …. Therefore, there is only a single reward function R1, which all agents try to maximize together. Team games can also be called coordination games or fully cooperative games for this reason.
Team Q-learning (Michael L. Littman., 2001) is a reinforcement learning algorithm specifically designed for team games. In team games, because every reward received by agent 1 is received by all agents, we have that Q1=Q2=…=Qn. Therefore, only one Q-function needs to be learned. The value function is defined:

$$V_1(s) = \max_{a_1,...a_n} Q_1(s,a_1,...,a_n) \tag{29}$$

The update rule for team Q-learning can be written:

$$\begin{aligned} Q_1(s,a_1,...,a_n) \leftarrow (1-\alpha)Q_1(s,a_1,...,a_n) \\ + \alpha(r_1 + \beta V_1(s)) \end{aligned} \tag{30}$$

In MAS, an agent team consists of the agents that have the same goal. Because of cooperation in a team, agents adopt team Markov game to cooperate each other to accomplish the task.
We present the multi-agent coordination framework shown in Figure 14. Based on the environment information and opponent information, Team commander applies zero-sum Markov game to make decision of the team level. According to team commander's strategies, member agents use the team Markov game to make the decision of member level, performing their actions to environment.

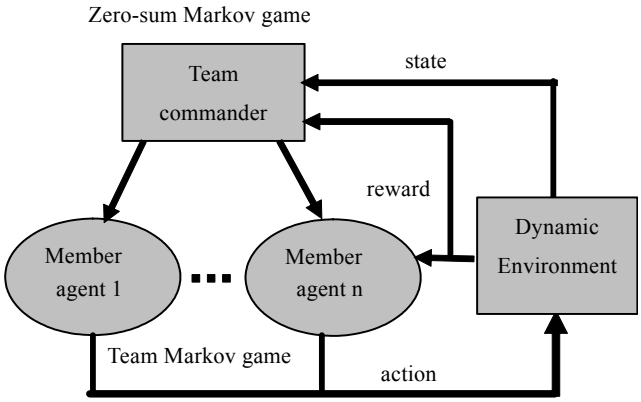


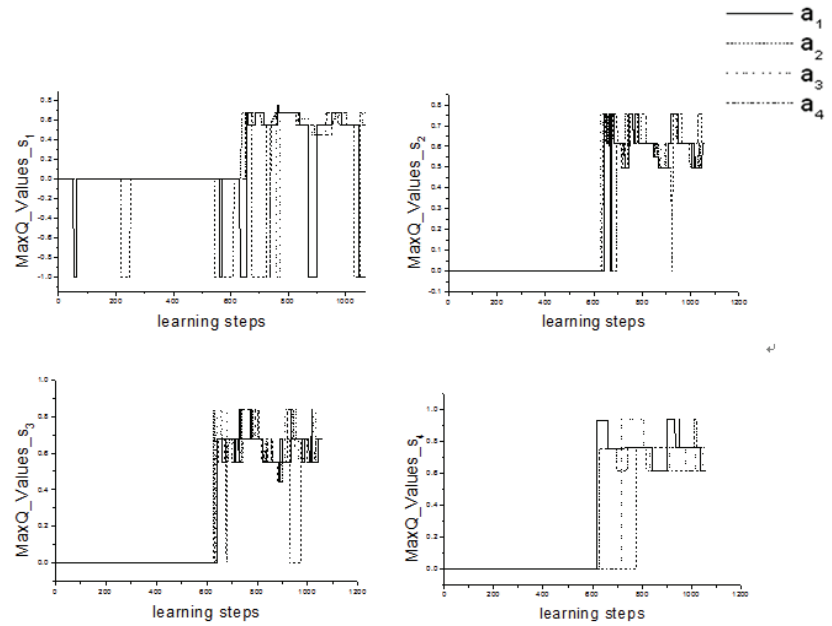
Fig. 14. Multi-agent coordination framework

Team commander's strategies aim at the environment and opponent team. Also, these strategies arrange different actions' choice scope to all member agents. Team commander decomposes the complex task to several strategies. Each of them divides member agents into different roles, which are according to basic skills of member agents. Each member agent carries out its skill by learning.
The decomposition of task and arrangement of roles are designed based on application system and domain knowledge. How to make decision and accomplish task is learned by multi-agent coordination framework.

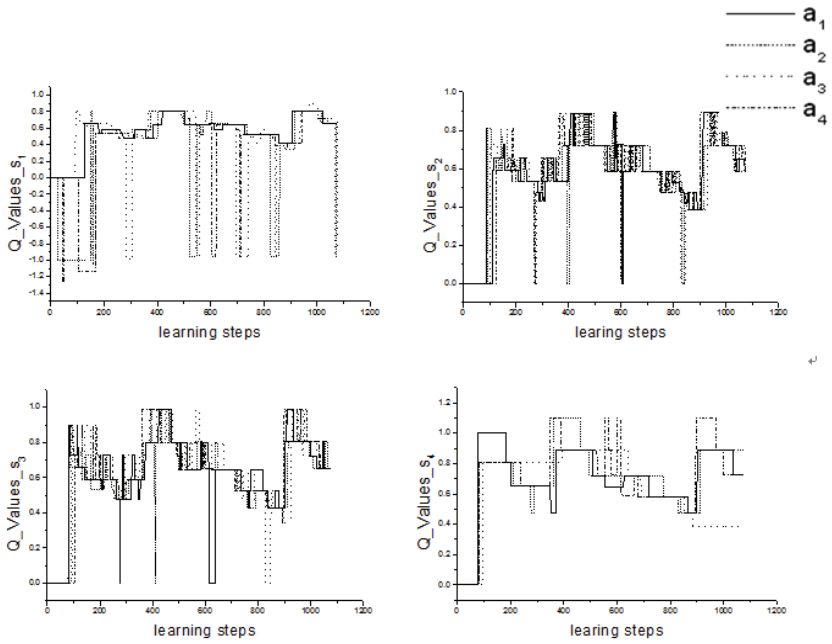
5.2 Experiment and results

a. Experiment setup

Robot soccer is a typical MAS. SimuroSot simulation platform is applied to evaluate our proposed method. Ball and playground is environment. Robots are agents. We define the state set, S = {threat, sub-threat, sub-good, good}. The opponent team situation is defined to O = {hard-defend, defend, offend, strong-offend}. In the team commander, there is a team-level strategy set, H = {hard-defend, defend, offend, strong-offend}. Each member agent has the action set, A = {guard, resist, attack, shoot}. Each team level strategy corresponds to a team formation and arranges the roles of all member agents.
In multi-agent learning, traditional reinforcement function is usually developed that reward is +1 if the home team scored; reward is -1 if the opponent team scored. In order to accelerate learning, we design a heterogeneous reinforcement function, which reinforces multiple goals including global and local goals.



(a)



(b)

Fig. 15. Q-values of Robot 1, 2 in experiment 1

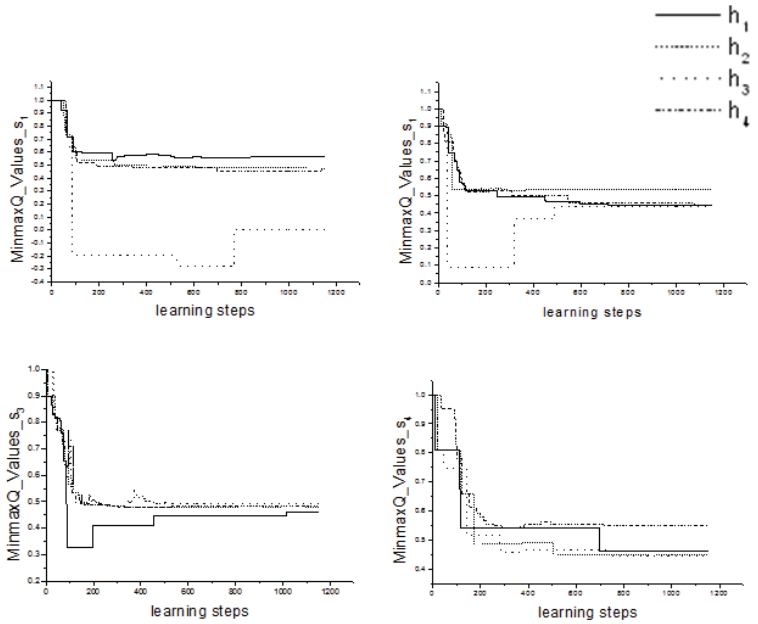
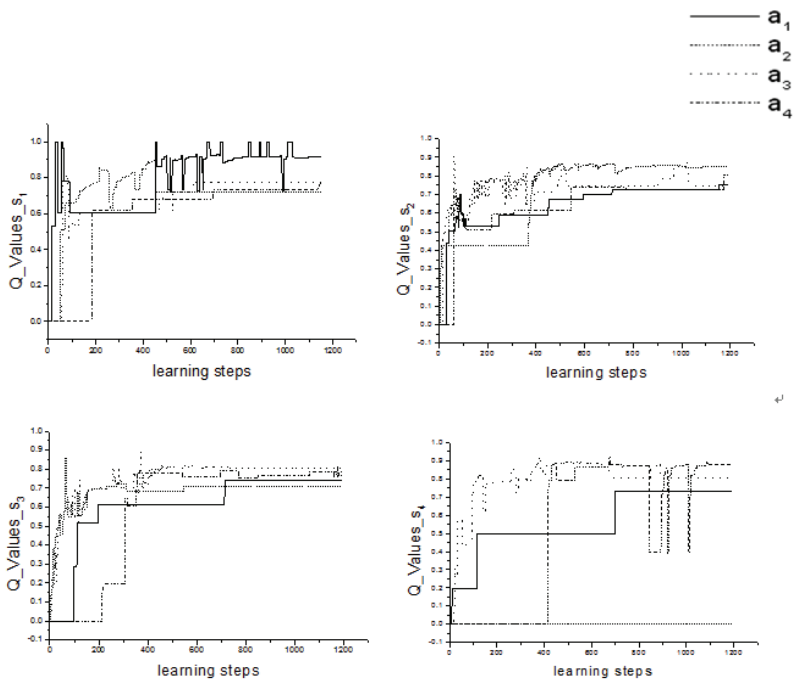
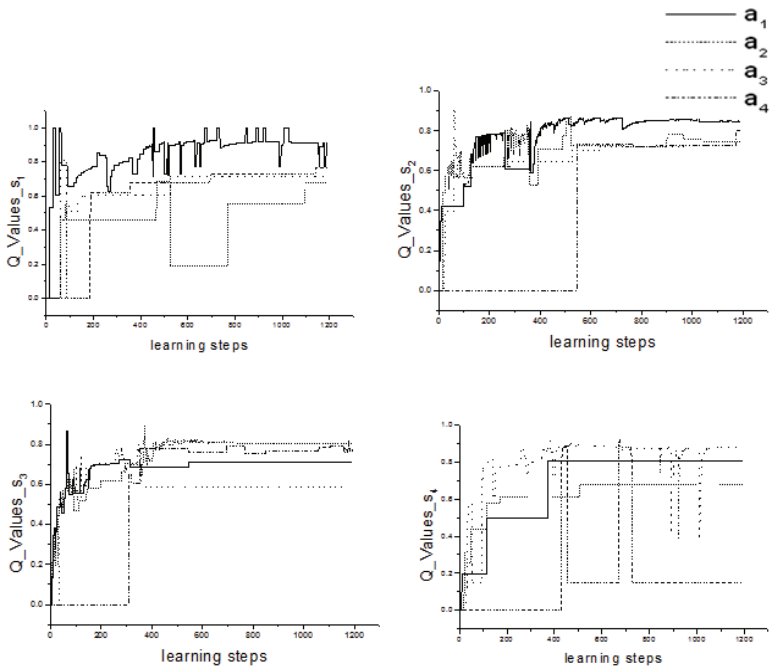


Fig. 16a. Q-values of Team commander in experiment 2



(b)



(c)

Fig. 16b-c. Q-values of Robot 1, 2 in experiment 2

The global goal of match is to encourage home team's scoring and avoid opponent team's scoring. The reward of global goal is defined:

$$r_g = \begin{cases} c, & our\ team\ scored \\ -c, & other\ team\ scored \\ 0, & otherwise \end{cases} \qquad (31)$$

$$c > 0$$

The local goals are to achieve home team's cooperative strategies. This reinforcement includes the domain knowledge and evaluates member agents' cooperative effect. It is defined:

$$r_a = \begin{cases} d & strategy\ success \\ 0 & strategy\ unsuccess \end{cases} \qquad (32)$$

$$d > 0,$$

Team commander sums the two kinds of reinforcement, weighting their values constants appropriately, so its reinforcement function, Rc, is defined:

$$R_c = \omega \bullet r_g + \upsilon \bullet r_a \qquad (33)$$

$$\omega, \upsilon \geq 0, \quad (\omega + \upsilon) = 1$$

In member level, team games focus on the cooperation of member agents. Its reinforcement function, Rm, is defined:

$$R_m = r_a \tag{33}$$

b. Results

There are two group experiments. In experiment 1, the home team uses the conventional Q-learning. In experiment 2, the home team uses our proposed method. The opponent team uses fix strategy. The team size is 2.

The results of experiment 1 are shown in Figure 4a and Figure 4b respectively. The learning of two Robots has worse convergence and still has many unstable factors at the end of experiment. In the results of experiment 2, Figure 5a shows zero-sum game performance of the team commander. The values of $\underset{o\in O}{Min}Q(s_i,h_j,o)$ (i, j = 1, 2, 3, 4) are recorded. They are convergent rapidly. Team commander gets the effective and rational strategy. Figure 5b and Figure 5c describe two Robots' Q values, $\underset{a\in A}{Min}Q(s_i,a_j,a)$ and $\underset{a\in A}{Min}Q(s_i,a,a_j)$ (i, j = 1, 2, 3, 4) respectively, which are convergent. Robots can get deterministic policy to choose actions.

5.3 Summary

In multi-agent environment, neglecting the agents' interaction of competition and cooperation, multi-agent learning can not acquire the better performance. This paper proposed a multi-agent coordination framework based on Markov game, in which team level adopts zero-sum game to resolve competition with opponent team and member level adopts team game to accomplish agents' cooperation. By applying the proposed method to Robot Soccer, its performance is better than the conventional Q-learning. However, this paper only discusses two agent teams' relationship. How to deal with the games and learning of multiple agent teams in multi-agent environment will confront with more challenges and difficulties.

6. References

Galina Rogova & Pierre Valin. (2005). Data fusion for situation monitoring, incident detection, alert and response management, Amsterdam, Washington, D.C.: IOS Press.

Dempster, A. P. (1967). Upper and Lower Probabilities Induced by a Multivalued Mapping. Ann. Math. Statist., vol. 38, pp. 325-339.

Shafer, G. (1976). A Mathematical Theory of Evidence, Princeton University Press.

Elouedi, Z.; Mellouli, K. & Smets, P. (2004). Assessing sensor reliability for multisensor data fusion within the transferable belief model. In: IEEE Transactions on Systems, Man, and Cybernetics, Vol: 34, Issue 2, Feb, pp.782- 787

Philippe Smets. (2005). Decision making in the TBM: the necessity of the pignistic transformation. International Journal of Approximate Reasoning, Vol 38, Issue 2, February, pp. 133-147.

Rogova G. (2003) Adaptive decision fusion by reinforcement learning neural network. In Distributed Reinforcement Learning For Decision-making In Cooperative Multi-agent Systems, Part 1, CUBRC Technical report prepared for AFRL, Buffalo, NY.

M. L. Littman. (2001). Value-function reinforcement learning in Markov games. Journal of Cognitive Systems Research, Vol. 2, pp.55-66,

Bowling M.; Veloso M. (2004). Existence of Multiagent Equilibria with Limited Agents. J of Artificial Intelligence Research, Vol. 22, Issue 2, pp.353-384

C. J. C. H. Watkons & P. Dayan. (1992). Q-leanign. Machine Learning, Vol. 8, Issue 3, pp.279-292.

M. J. Mataric (2001). Learning in behavior-based multi-robot systems: policies, models, and other agents. Journal of Cognitive Systems Research, Vol. 2, pp. 81-93,

Kousuke INOUE; Jun OTA; Tomohiko KATAYAMA & Tamio ARAI. (2000). Acceleration of Reinforcement Learning by A Mobile Robot Using Generalized Rules, Proc. IEEE int.Conf. Intelligent Robots and Systems, pp.885-890

W. D. Smart & L. P. Kaelbling. (2002). Effective reinforcement learning for mobile robots. in Proceedings of the IEEE International Conference on Robotics and Automation, Vol. 4, pp. 3404-3410.

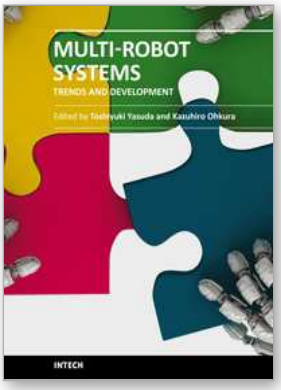
Yang, X. M. Li & X. M. Xu (2001). "A suery of technology of multi-agent cooperation", Information and Control, Issue. 4, pp.337-342

Y. Chang; T. Ho & L. P. Kaelbling. (2003). Reinforcement learning in mobilized ad-hoc networks. Technical Report, AI Lab, MIT,

Kok, J. R. & Vlassis, N. (2006). "Collaborative multiagent reinforcement learning by payoff propagation", Journal of Machine Learning Research 7, pp.1789 –1828.

Zhong Yu; Zhang Rubo & Gu Guochang. (2003). Research On Architectures of Distributed Reinforcement Learning Systems. Computer Engineer and Application., Issue 11, pp.111-113 (in Chinese).

M. L. Littman. (1994). Markov Games as a Framework for Multi-agent Reinforcement Learning. Machine Learning , Vol. 11, pp.157-163.



Multi-Robot Systems, Trends and Development

Edited by Dr Toshiyuki Yasuda

ISBN 978-953-307-425-2
Hard cover, 586 pages
Publisher InTech
Published online 30, January, 2011
Published in print edition January, 2011

This book is a collection of 29 excellent works and comprised of three sections: task oriented approach, bio inspired approach, and modeling/design. In the first section, applications on formation, localization/mapping, and planning are introduced. The second section is on behavior-based approach by means of artificial intelligence techniques. The last section includes research articles on development of architectures and control systems.

How to reference

In order to correctly reference this scholarly work, feel free to copy and paste the following:

Bo Fan and Jiexin Pu (2011). Multi-Robot Information Fusion and Coordination Based on Agent, Multi-Robot Systems, Trends and Development, Dr Toshiyuki Yasuda (Ed.), ISBN: 978-953-307-425-2, InTech, Available from: http://www.intechopen.com/books/multi-robot-systems-trends-and-development/multi-robot-information-fusion-and-coordination-based-on-agent

INTECH
open science | open minds

InTech Europe
University Campus STeP Ri
Slavka Krautzeka 83/A
51000 Rijeka, Croatia
Phone: +385 (51) 770 447
Fax: +385 (51) 686 166
www.intechopen.com

InTech China
Unit 405, Office Block, Hotel Equatorial Shanghai
No.65, Yan An Road (West), Shanghai, 200040, China
中国上海市延安西路65号上海国际贵都大饭店办公楼405单元
Phone: +86-21-62489820
Fax: +86-21-62489821