A Modified Average Reward Reinforcement Learning Based on Fuzzy Reward Function

Zhenkun Zhai, Wei Chen, Xiong Li, Jing Guo

***Abstract*—The purpose of this paper is to propose a fuzzy reward function for improving the learning efficiency of reinforcement learning. Reinforcement learning is a sort of on-line learning approach. During the learning process, learning system (often called an agent) learns how to operate in the environment, basing upon the effect of action—reward signal. Actually, the reward signal is always represented with a reward function, of which the role is to evaluate whether the agent acts well or not. According to the distribution of rewards in the space of states, reward function can be classified as two basic forms, dense function and sparse function. Parse function has the advantage of the easy implementation, but its learning efficiency is relatively low. As for dense function, it is very difficult to design when the size of state space is very large. In response to these defects, we present a methodology for designing a fuzzy reward functions by the use of fuzzy inference system. As a result of this solution, not only generalization ability of reward function can be improved, but also we are apt to embed expert experience in reinforcement learning system. Through applying the fuzzy reward function to R-learning and making simulation experiment, we verify that the learning efficiency for R-learning is effectively improved.**

***Index Terms*—reinforcement learning, fuzzy inference system, reward function, R-learning.**

I. Introduction

A. Reinforcement Learning

Reinforcement learning is a kind of learning that can be seen as located between the completely supervised and the unsupervised paradigms[1]. The learning system (often called an agent) receives feedback (rewards and punishments) from the world.That means the agent learns behavior through trial-and-error interactions with a dynamic environment[2]. The learning task is that the agent should try to maximize its long term reward by performing actions and receiving rewards. The challenge of the agent is to understand how a current action will affect future rewards. A good way to model this task is Markov Decision Process (MDP), which has become the dominating approach in reinforcement learning. In general, MDP model can be expressed with a multicomponent (S, A, T, r) [3]. A MDP consists of a set of states S, and a set of actions $A=(a_1, a_2 .....a_n)$ for $n\geq 2$. Associated with each state-action pair is a state transition function $T: S\times A\rightarrow T$.There is also an immediate reward function $r: S\times A\rightarrow r$, where $r(s,a)$ is the expected reward for doing action a in state s. At each discrete time step t, the agent senses the current state s_t, and chooses an action a_t from A, and performs it. The environment responds by giving the agent an immediate reward $r(s,a)$ and transferring to the next state $s_t^{'}$. Here the state transition function $T(s_t, a_t)$ represents the probability of moving from state s_t to $s_t^{'}$ under action a_t. So we can use $P_{s_t, s_t^{'}}(a_t)$ in place of $T(s_t, a_t)$ .In uncertainty MDP,the state transition function and immediate reward function are all nondeterministic. The The task of the agent is to learn a policy, $\pi : S\rightarrow A$ which can produce the greatest possible cumulative reward for the agent over time.

On the other hand, the general framework of reinforcement learning can be divided into two forms, discounted reward reinforcement learning and average reward reinforcement learning. In different domains they have their own merits. This paper undertakes a detailed examination of the latter—average reward reinforcement learning.

B. Average Reinforcement Learning

The principle of average reward reinforcement learning as follows, firstly, the average cumulative reward of reinforcement learning system is calculated (or estimated) ; secondly, in each policy agent will learn the value function, which is relative to the sum of average reward; finally, according to the value function and Bellman optimal theorem, we can get the optimal policy. In average reward reinforcement learning, the average reward $\rho^{\pi}(s_t)$ associated with a particular policy π at a state s_t is defined as,

$$\rho^{\pi}(s_t)=\lim_{N\rightarrow\infty}\frac{E\left[\sum_{i=0}^{N-1} r(s_{t+i}, \pi(s_{t+i}))\right]}{N} \quad (1)$$

Where $r(s_{t+i}, \pi(s_{t+i}))$ is the reward received from state s_{t+i}, and actions are chosen using policy π . We define a gain-optimal policy π^* as one that maximizes the average

This work is supported by National Nature Science Foundation under Grant 60843001.

Zhenkun Zhai is with the Faculty of Automation, Guangdong University of Technology, Guangzhou, 510006, China (corresponding author to provide phone: +08613631327659; email: smartzhai@163.com).

Wei Chen is with the Faculty of Automation, Guangdong University of Technology, Guangzhou, 510006, China (email: 0tongxin3@163.com)

Xiong Li, Jing Guo is with the Faculty of Automation, Guangdong University of Technology, Guangzhou, 510006, China.

reward over all states, that is $\rho^{\pi^*}(s) \geq \rho^{\pi}(s)$ over all policies π and states s. For unichain MDP, the average reward of any policy is state independent. That is, $\rho^{\pi}(s_i) = \rho^{\pi}(s_j) = \rho^{\pi}, \forall s_i, s_j \in S$.

Due to the fact that average reward reinforcement learning has neglected the relative importance of the short-term reward and the forward reward, gain-optimal policy is not always the optimal policy. In order to get the optimal policy which has shortest time or step, bias optimal policy is defined as,

$$V^{\pi}(s) - V^{\pi}(s_0) = \lim_{N\to\infty}\{E\left[\sum_{t=0}^{N-1} R_t^{\pi}(s)\right] - E\left[\sum_{t=0}^{N-1} R_t^{\pi}(s_0)\right]\} \quad (2)$$

A policy π^* is termed bias-optimal if it is gain-optimal, and it also maximizes bias value, that is $V^{\pi^*}(s) \geq V^{\pi}(s)$, over all $s \in S$ and policies π.

II. FUZZY INFERENCE SYSTEM

In recent years, fuzzy inference system in dynamical systems has attracted increasing attention since it's applicable in the case of uncertainty. Fuzzy inference systems employ a mode of approximate reasoning that resembles the decision making process of humans[4]. The behavior of fuzzy inference system is easily understood by a human expert as knowledge is expressed by means of intuitive, linguistic rules. On the other hand, fuzzy inference system is a rule based system. It consists of a set of rules, which are applied to the actual controller input to infer the controller output. Fig. 1 shows the basic configuration of the fuzzy system considered in this paper.

Basically a fuzzy inference system is composed of five functional blocks[5],

a *rule base* containing a number of fuzzy if-then rules;

a *data base* which defines the membership functions of the fuzzy sets used in the fuzzy rules;

a *decision-making* unit which performs the inference operations on the rules;

a *fuzzifier* which transforms the crisp inputs into degrees of match with linguistic values;

a *defuzzifier* interface which transform the fuzzy results of the inference into a crisp output.

Usually, the rule base and the database are jointly referred to as the knowledge base.

The steps of fuzzy reasoning (inference operations upon fuzzy if-then rules) performed by fuzzy inference systems are,

a. Compare the input variables with the membership functions on the premise part to obtain the membership values (or compatibility measures) of each linguistic label. (This step is often called fuzzification).

b. Combine (through a specific T-norm operator, usually multiplication or min.) the membership values on the premise part to get firing strength (weight) of each rule.

c. Generate the qualified consequent (either fuzzy or crisp) of each rule depending on the firing strength.

d. Aggregate the qualified consequents to produce a crisp output. (This step is called defuzzification.)

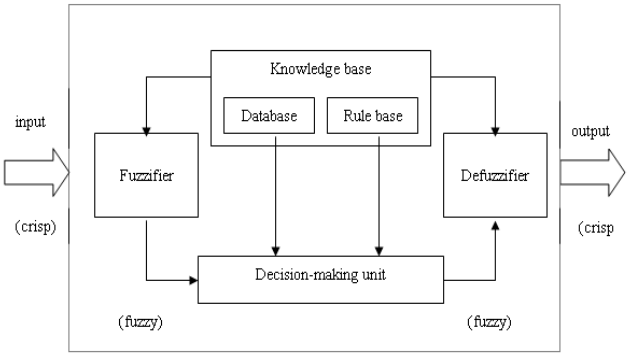

Fig.1 Fuzzy inference system

A. Design of Fuzzy Reward Function Based on Fuzzy Inference System

In reinforcement learning system, reward signal transmits from the environment to the system. In order to expediting the learning process, reward signal should describe the process accurately and in time. And it is always represented with a reward function $r(s,a)$, which holds constant or changes through time. According to the distribution of rewards in the space of states, reward function can be classified as two basic forms, dense function and sparse function. Parse function takes advantage of the easy implementation, but its learning efficiency is relatively low. As for dense function, it is very difficult to design when the size of state space is very large. In response to these defects, we present a methodology for designing a fuzzy reward functions by the use of fuzzy inference system. The design idea is as following, state vector in reinforcement learning is regarded as the input vector of the fuzzy inference system, that is $S=(S_1,S_2,.....S_n)$ and reward signal r is seen as the output of the fuzzy inference system. Here assume that a crisp input vector $S=(S_1,S_2,.....S_n)$ maps to a fuzzy set A_x, where A_x is label of the fuzzy set such as "small", "medium", "large", etc. As for output r, there is a mapping from a crisp r to a fuzzy set B. And we assume that there are M rules, and lth rule is

R^l: *IF* S_1 *is* A_1^l *and* S_2 *is* A_2^l *.....and* S_n *is* A_n^l *THEN r is* B^l

l=1,2,.....M, where $S=(S_1,S_2,.....S_n)$ and r are the crisp input and output of the fuzzy system, respectively. And here we use the sum-product inference and the center-average defuzzifier.So the fuzzy system can be expressed as,

$$r(x) = \frac{\sum_{l=1}^{M} r^l \prod_{i=1}^{n} \mu_{A_i^l}(x_i)}{\sum_{l=1}^{M} \prod_{i=1}^{n} \mu_{A_i^l}(x_i)} \quad (3)$$

where $r(x)$ is the crisp output of fuzzy system, $\mu(x_i)$ is the membership degree of the input S_i to fuzzy set A_x, r is the point at which the membership function of fuzzy set B achieves its maximum value.

B. R-learning Algorithm Based on Fuzzy Reward Function

R-learning is a model-free average reward reinforcement learning method, and R-learning employs the action value

representation. The action value $R^{\pi}(s,a)$ represents the average adjusted value of doing an action a in state s once, and then following policy π subsequently [3]. That is,

$$R^{\pi}(s,a) = r(s,a) - \rho^{\pi} + \sum_{s'} P(s' \mid s,a) V^{\pi}(s') \tag{4}$$

Where $V^{\pi}(s') = \max_{a \in A} R^{\pi}(s',a)$, and ρ^{π} is the average reward of policy π. R-learning consists of the following steps,

a. Let time step t=0. Initialize all the $R_t(s,a) = 0$, let the current state be s.

b. Choose the action a that has the highest $R_t(s,a)$ value with some probability, else let a be a random exploratory action.

c. Carry out action a. Let the next state be s', and the reward be $r_t(s,a)$. Update the R values and the average reward ρ using the following rules,

$$R_{t+1}(s,a) \leftarrow R_t(s,a)(1-\beta) + \beta(r_t(s,a) - \rho_t^{\pi} + \max_{a' \in A} R_t(s',a')) \tag{5}$$

$$\rho_{t+1} \leftarrow \rho_t(1-\alpha) + \alpha[r_t + \max_{a' \in A} R_t(s',a) - \max_{a \in A} R_t(s,a)] \tag{6}$$

d. Set the current state to s' and go to step b.

Here $0 \le \beta \le 1$ is the learning rate controlling how quickly errors in the estimated action values are corrected, and $0 \le \alpha \le 1$ is the learning rate for updating ρ. And ρ is only updated when a non-exploratory action is performed.

In conventional R-learning algorithm, reward function is always designed to parse function. Considering as improvement, we introduce fuzzy reward function instead of parse function. Through making simulation experiment, we verify the performance of the improved algorithm.

III. THE APPLICATION OF THE IMPROVED R-LEARNING IN ROBOCUP SIMULATION LEAGUE(2D)

A. Analysis of The Kick Problem in Robocup Simulation League(2D)

Robocup (Robot Soccer World Cup) is a standard problem so that various theories, algorithms and architectures can be evaluated.[6] It is a suitable domain for evaluating and comparing different AI techniques[7]. In Robocup Simulation League(2D),the Soccer Server program provides a virtual test platform. Simulation game is a dynamic real-time system,and the server parameter simulation_step which in current server version has a value 100ms[8]. In each simulation_step, the server receives commands from the players of both teams, calculates the positions and movements of all objects on the field, sends visual and audible information to all players and acts as a referee. In a soccer match, whether to be skillful on completing a kicktask,such as pass and shoot, is very critical. In Soccer Server,the agent and ball occupy a circle.In Soccer Server,each agent occupies a circle. When the ball's distance from the center of the agent is less than kick radius, this agent can accelerate the ball by sending to the server a kick command accompanied with two parameters angle and power.And the equations of the ball's movement are as following[9],

$$(\vec{u}_x^{\,t+1}, \vec{u}_y^{\,t+1}) = (\vec{v}_x^{\,t}, \vec{v}_y^{\,t}) + (\vec{a}_x^{\,t}, \vec{a}_y^{\,t}) + (\tilde{\Upsilon}_{r\max} + \tilde{\Upsilon}_{r\max}) \tag{7}$$

$$(\vec{v}_x^{\,t+1}, \vec{v}_y^{\,t+1}) = ball\_decay * (\vec{u}_x^{\,t+1}, \vec{u}_y^{\,t+1}) \tag{8}$$

$$(\vec{P}_x^{\,t+1}, \vec{P}_y^{\,t+1}) = (\vec{P}_x^{\,t}, \vec{P}_y^{\,t}) + (\vec{v}_x^{\,t+1}, \vec{v}_y^{\,t+1}) \tag{9}$$

Here V represents velocity, P means position, t means simulation time, u represents movement vector , and $\tilde{\Upsilon}_{r\max}$ means random noise. The equations of how the kick command works on the ball's acceleration are as following,

$$ep = kick\_power * kick\_powerrate \tag{10}$$

$$ep = ep * (1 - 0.25 * \frac{dir\_ball}{180^0} - 0.25 * \frac{dist\_ball}{margin}) \tag{11}$$

Because of the limit of the ball's acceleration and existence of the ball's initial speed, one kick command often fails to accelerate the ball directly to the desired speed. In other words, a kick task possibly needs a series of kick commands to be completed. On the other hand, the essence of first-order MDP is that the transition probability and reward between current state and next state only depends on the state and action at current time, but has nothing to do with the history state and action. So, Robocup soccer simulation can be seen as a first-order MDP. Based on the above improved R-learning algorithm, in this paper the decision-making of kick-task means that players will plan out a series of kick commands completed in the following cycles to accelerate the ball to the desired speed,according to the initial state in current cycle.

B. Establishment of The Algorithm Model

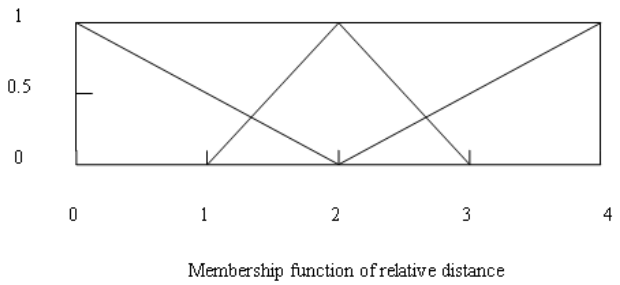

Fig 2 membership function of the relative distance

$S(d_1,d_2,d_3,d_4,d_5,a_1,a_2,a_3,a_4)$ and output r, we use triangular membership function and divide the input and output vector into three fuzzy sets. Figure 2 shows the membership function of the relative distance, which is similar with the relative angle and reward signal.

When there is a input vector $S(d_1,d_2,d_3,d_4,d_5,a_1,a_2,a_3,a_4)$ through the following fuzzy IF-THEN rules, we can get the fuzzy set of the reward signal.

$$R: IF\ S_1\ is\ A_1\ and S_2\ is\ A_2 \ldots\ldots and\ S_n\ is\ A_n\ THEN\ r\ is\ B$$

And then the defuzzifier maps the fuzzy set to a crisp value r, after which the reward signal is sent to the agent.

On the other hand, according to expert experience we choose Boltzmann Exploration as the exploration strategies. The Boltzmann exploration function assigns the probability of doing an action a in state x as,

$$p(x,a) = \frac{e^{\frac{U(x,a)}{T}}}{\sum_a e^{\frac{U(x,a)}{T}}} \quad (12)$$

Where T is a "temperature" parameter that controls the degree of randomness.. In our simulation experiment, in the early stage of the learning process we choose smaller T, and gradually increase T value along with learning process.

IV. EXPERIMENTAL RESULTS

In this section we present a simulation experimental study of the improved R-learning. Firstly, we give the R-Value curve of the improved R-learning in the simulation game. And then we compare the performance of the improved R-learning with Q-learning. Finally we give the average goal different in two soccer teams which use the improved R-learning and expert experience respectively. As a specific example, there are also two illustrations for a successful through-pass kick.

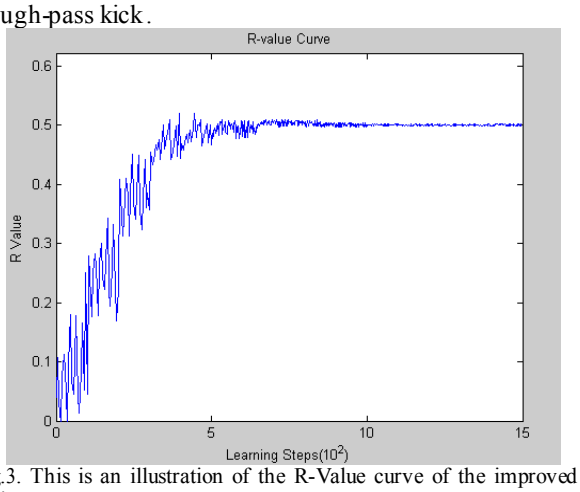

Fig.3. This is an illustration of the R-Value curve of the improved R-learning

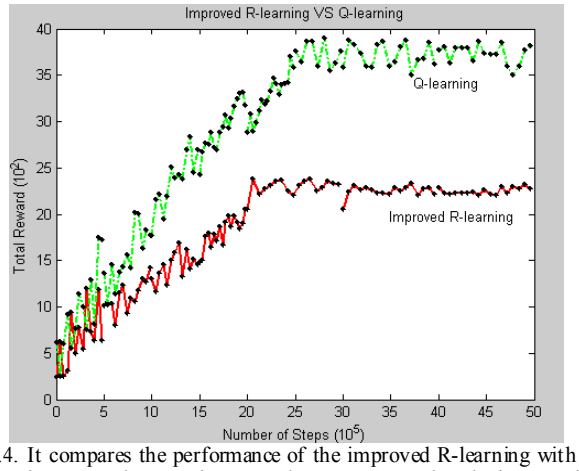

Fig.4. It compares the performance of the improved R-learning with the Q-learning. From the experiment results, we can see that the improved R-learning is clear outperforming than the Q-learning in the simulation game. Furthermore the former is quickly to reach its optimal performance than the latter

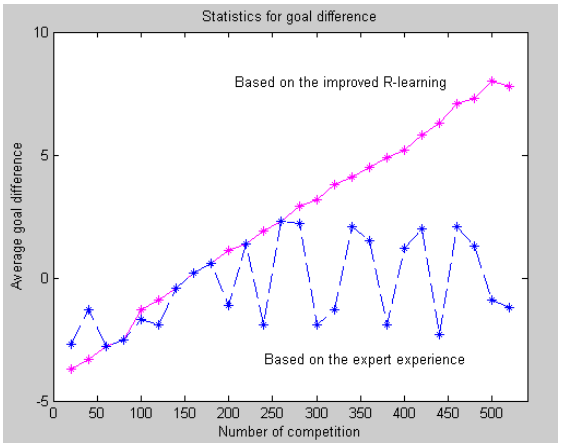

Fig.5. In figure 5 it indicates that average goal difference in soccer team based on the improved R-learning gradually increases. In the early stages of the learning process, the performance of the two policies is similar. Following with the number of competition increase, the soccer team based on the improved R-learning has the remarkable advantages.

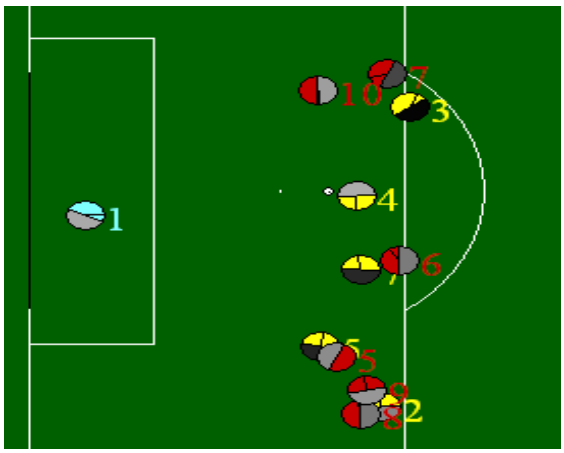

Fig.6. There is an example of the process of a successful through-pass kick and goal in the real-time simulation game. In Fig 6, we can see that the No.6 player on red side observes his teammate No.10 lying in a good position, and he passes the ball immediately.

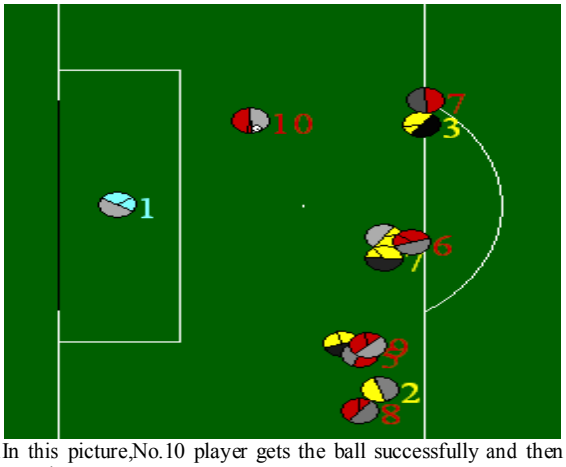

Fig.7.In this picture,No.10 player gets the ball successfully and then he scores a goal.

REFERENCES

[1] Micheal Kaiser, Markus Riepp. Learning from undiscount delay rewards. GI Fachgruppentreffen Maschinelles Lernen,Chemnitz,1996,8.

[2] Leslie Pack Kaelbling, Michael L. Littman. Reinforcement Learning: A Survey. Journal of Artificial Intelligence Research 4,1996 237-285.

[3] MAHADEVAN, S. Average Reward Reinforcement Learning: Foundations, Algorithms, Empirical Result Machine Learning, 1996,22,159-195.

[4] Frank Hoffmann. Evolutionary Algorithms for Fuzzy Control System. Design. Proceeding of the IEEE,2001.

[5] Jyh-Shing Roger Jang. ANFIS: Adaptive-Network-Based Fuzzy Infe-. Rence System. System, Man and Cybernetics, IEEE Transaction on, 1993.

[6] Hiroaki Kitana. RoboCup:A Challenge Problem for AI and Robotics. In Robocup-97.

[7] Mihal Badjonski, Kay Schroter, Jan Wendler, HansDieter Burkhard. Learning of Kick in Artificial Soccer. In Robocup-2000.

[8] Boer R D, Kok J. The Incremental Development of Synthetic Multi-ent System: TheUVA Trileam 2001Robotic Soccer Simulation Team. Master's thesis, University of Amsterdam, Netherlands, 2002.

[9] Chen Mao, Ehsan Forouhgi, Fredrik Heintz, Xianghua Zhang, Spiros Kapetanakis, Kostas Kostiadis,Johan Kummeneje, Ituski Noda,Oliver Obst, Pat Riley,Timo steffens,Wang Yi and Yin Xiang,Robocu Soccer Soccer Server (User Manual). Available in http://sourceforge.net/projects/sserver/

[10] Jinyi Yao, Jiang Chen and Zengqi Sun. An Application in RoboCup CombiningQ-learning with Adversarial Planning, The World Congress on Intelligent Control and Autumation, 2002.