ELSEVIER

Information Sciences 145 (2002) 45–68

INFORMATION SCIENCES
AN INTERNATIONAL JOURNAL

www.elsevier.com/locate/ins

Robot learning with GA-based fuzzy reinforcement learning agents

Changjiu Zhou *

School of Electrical and Electronic Engineering, Singapore Polytechnic, 500 Dover Road, Singapore 139651, Singapore

Received 4 July 2001; received in revised form 8 October 2001; accepted 28 November 2001

Abstract

How to learn from both expert knowledge and measurement-based information for a robot to acquire perception and motor skills is a challenging research topic in the field of autonomous robotic systems. For this reason, a general GA (genetic algorithm)-based fuzzy reinforcement learning (GAFRL) agent is proposed in this paper. We first characterize the robot learning problem and point out some major issues that need to be addressed in conjunction with reinforcement learning. Based on a neural fuzzy network architecture of the GAFRL agent, we then discuss how different kinds of expert knowledge and measurement-based information can be incorporated in the GAFRL agent so as to accelerate its learning. By making use of the global optimization capability of GAs, the GAFRL can solve the local minima problem in traditional actor-critic reinforcement learning. On the other hand, with the prediction capability of the critic network, GAs can evaluate the candidate solutions regularly even during the periods without external feedback from the environment. This can guide GAs to perform a more effective global search. Finally, different types of GAFRL agents are constructed and verified using the simulation model of a physical biped robot.
© 2002 Elsevier Science Inc. All rights reserved.

Keywords: Robot learning; Reinforcement learning; Genetic algorithms; Neural fuzzy systems; Learning agents; Biped robot

* Fax: +65-6772-1974.
E-mail addresses: zhoucj@sp.edu.sg, changjiu.zhou@ieee.org (C. Zhou).

1. Introduction

Building a robot that learns to perform a task has been acknowledged as a difficult problem, due to sensory and effector uncertainty, partial observability of the robot's environment, and non-stationary of the environment [3,4,11,14]. For a learning problem, there are many alternative learning methods that can be chosen, either from neural networks, the statistical or the machine learning literature. In neural learning methods, supervised learning (SL) is more efficient than the reinforcement learning (RL) [1,15] when the input–output training data are available. However, many robot control problems require selecting control actions whose consequences emerge over uncertain periods for which input–output data are not really available. In such a case, RL can be used to learn the unknown desired outputs by providing an agent with suitable evaluation of its performance. The RL agent allows autonomous systems to learn from their experiences instead of exclusively from knowledgeable teachers. Therefore, the RL agent is more appropriate than the SL agent for practical systems such as robot learning and control. However, for real-time applications, especially for a movement system with many degrees of freedom, there is an exponential explosion in the number of actions that can be taken in every state. It is difficult to search such a huge space for what constitutes a good action, and learning is therefore too slow. However, human beings can always deal with perceptual and action uncertainty, non-stationarity, and real-time constraints. Furthermore, human beings can utilize perception-based information to guide their learning on those parts of the state–action space that are actually relevant to the task. Hence, we conduct a research aimed at improving robot learning through incorporation of both expert knowledge and measurement-based information.

In order to incorporate expert knowledge in the RL agent, a fuzzy RL (FRL) agent with a neural fuzzy architecture is a naturally good choice. For the FRL agent [2,9,23], neural fuzzy networks are used to replace the neuron-like adaptive elements in the RL architecture. Therefore, the expert knowledge can be directly built into the FRL agent as a starting configuration so as to speed up its learning. However, the main drawback of the actor-critic architectures is that they usually suffer from the local minima problem in learning as the gradient descent learning method is required. Because genetic algorithms (GAs) do not require or use derivative information, we integrate the FRL architectures and GAs into a GA-based FRL (GAFRL) agent in this paper. It can solve the local minima problem in the actor-critic architecture by using the global optimization capability of GAs. This method is similar to the TD (temporal difference) and GA-based reinforcement (TDGAR) learning method proposed by Lin and Jou [10]. It uses a critic network to provide a more informative internal reinforcement signal for the action network. Since the internal reinforcement signal can be used to formulate as the fitness function of the GA, so the GA can evaluate the candidate solutions regularly even during the periods without external reinforcement feedback from the environment. Hence, the GA can proceed to new generations regularly without waiting the arrival of the external reinforcement signal.

The rest of the paper is organized as follows. In the following section, we give a brief introduction on robot learning. Some robot learning problems are also addressed in conjunction with reinforcement learning. In Section 3, based on different kinds of expert knowledge and measurement-based information available for assisting robot learning, four types of GA-based reinforcement learning agents and their learning methods (TD + GA) are constructed. We then demonstrate how the GAFRL agent can be used for biped robot learning in Section 4. In Section 5, the effectiveness of the proposed GAFRL agent is verified using the simulation model of a physical biped robot. This is followed by some concluding remarks.

2. Robot learning

The goal of learning in a robot is to prepare it to deal with unpredictable situations and circumstances in its environment. Robot learning refers to the process acquiring a sensory-motor control strategy for a particular movement task and movement system by trial-and-error. Based on the type of information that is learned and its effect on the robot's action in the environment, the robot learning approaches can be divided into four main categories [3]: (1) learning numerical functions for calibration or parameter adjustment; (2) learning about the world; (3) learning to coordinate behaviors; (4) learning new behaviors. In this paper, we will discuss how the GAFRL agent can be used for the first category of robot learning, i.e., to acquire a task dependent control policy π that maps the continuous valued state vector x of a control system and its environment to a continuous valued control vector u:

$$u = \pi(x, \alpha, t), \tag{1}$$

where the parameter vector α contains the problem specific parameters in the policy π that need to be adjusted by the learning agent. Since the controlled system can be expressed as a nonlinear function $\dot{x} = f(x, u)$, the combined system and controller dynamics result in

$$\dot{x} = f(x, \pi(x, \alpha, t)). \tag{2}$$

From Eq. (2), we can see that learning control means finding a function π that is adequate for a given desired behavior and movement system. The general control diagram for a robot learning control system is shown in Fig. 1.

2.1. The robot learning problem

The robot learning problem can be examined from different aspects [4]. For example, we may discuss where the training data come from. We may consider

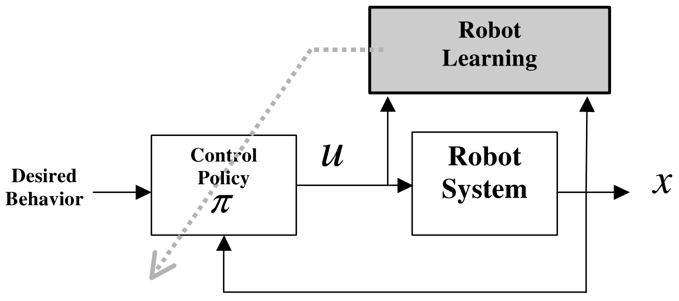

Fig. 1. General diagram for robot learning and control.

what types of feedback the robot might receive from the environment to help it to learn a task. We may also study what kinds of knowledge need to be learned. If the training data are available, the robot is being passively guided through the task. However, a more practical and challenging situation is that a robot attempts to learn a task is an unsupervised mode without the active guidance of a teacher. In this case, the robot has to collect its own training data. Supervised and unsupervised learning both play important roles in robot learning. Supervised learning is useful because training time with real robots is short. However, it is often difficult to generate a rich enough sample space of data. Based on the above consideration, trial-and-error unsupervised learning is more applicable for robot learning.

Another important issue should be addressed is the different types of feedback available to a robot. For a robot to learn something, it must be given some feedback concerning its performance. For example, the robot can be directly told what action to be performed with the control feedback. With the control feedback (desired behavior), it can eliminate a lot of unnecessary trial-and-error effort by the robot. On the other hand, scalar feedback provides the robot a reward signal telling it how well it is currently doing on the task. The ideal situation is that the robot receives scalar feedback at every step. However, the most common situation is that the only feedback occurs when the robot fails its task, such as in pole balancing and biped walking [19]. At one stream, some robots may receive no feedback at all during most of their exploratory trial-and-error learning. In view of this, how to provide a more informative feedback signal for robot learning, even during the periods without external feedback from the environment, has been considered as one of the major challenges facing the field of robot learning. In this paper, we will use a TD (temporal difference) [15] + GAs approach to tackle this problem in the field of reinforcement learning.

We should also notice that some types of feedback are somewhat oversimplified. For example, for scalar feedback, only the sign of the reward is usually used to indicate whether the robot's actions are good (positive reward) or bad

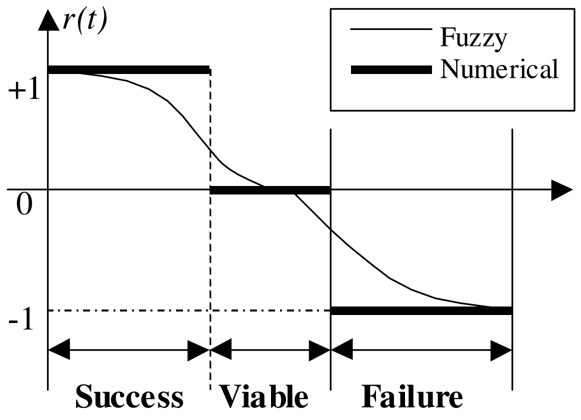

Fig. 2. Comparison of the fuzzy evaluative feedback signal with the numerical (crisp) one.

(negative reward). However, using such a crisp scalar feedback signal for learning is surely not the biologically plausible. For example, for human walking, we usually use the linguistic scalar signal, such as "near fall down", "almost success", "slower", "faster" and etc., rather than the crisp scalar signal like "stand" (positive reward) and "fall down" (negative reward), to evaluate the walking gait. To provide more informative scalar feedback to robot learning, we proposed an FRL agent with fuzzy evaluative feedback [20]. The comparison of the oversimplified crisp feedback with fuzzy feedback is illustrated in Fig. 2. In the following section, we will discuss how to make use of fuzzy evaluative feedback, rather than oversimplified crisp scalar feedback, in the GAFAR agent in order to speed up its learning.

2.2. Reinforcement in robot learning

The robot learning problem is easily formulated in the reinforcement learning (RL) framework as seeking the control policy (π in Eq. (1)) that maps states to behaviors so as to maximize received reward over the lifetime of the agent. RL covers techniques of learning by trial-and-error through evaluating performance feedback. It evaluates the behavior but dose not indicate correct behavior. This type of situation is common in robotics, and RL can therefore be applied to many robotics tasks [12,16].

Reinforcement can certainly play a role in the development of autonomous robotic systems. However, the prominence of that role depends on the extent to which it can be scaled to complex robot learning tasks. In the following, we shall look at some robot learning methods [14] in conjunction with the concept of RL.

(1) *Learning the control policy directly.* This method can learn the control policy π directly. The desired behavior needs to be expressed as an optimization criterion $r(t) = r(x(t), u(t))$ to be optimized over a certain temporal horizon t, resulting in a cost function: $J(x(t)) = \int_{t=0}^{T} r(x(s), u(s))\, \mathrm{d}s$. For RL, $r(t)$ corresponds to the "immediate reward", and $J(x(t))$ is called the "long-term reward". The policy π is acquired with RL by learning the optimal function $J(x(t))$ for every state x, and then deducing the policy π as the control action u in every state x that leads to the best future payoff. There are many variations of RL techniques, including methods that avoid estimating the optimization function $J(x(t))$ [14].

(2) *Learning the control policy in a modular way.* The RL agent usually requires a large amount of exploration of all actions and states for proper convergence of learning as well as appropriate representation for the function $J(x(t))$. However, for a high-dimensional movement system, this strategy will lead to an explosion of lookup table cells. A possible way to reduce the computational complexity of learning a control policy comes from modularizing the policy. It can learn sub-policies $\pi_1, \pi_2, \ldots, \pi_n$, instead of learning the entire policy π. This approach can then complete control policy from such sub-policies. RL has recently begun to formulate an approach to this problem.

(3) *Indirect learning on control policies.* This approach can decompose the learning problem into several independent learning problems. The motor commands here are not directly generated based on the information of the state of the world x, i.e., from the policy function π. Most commonly, the control policy is decomposed into a trajectory planning and an execution stage. RL can be used for different modules.

Robot learning is a very complex problem. It needs to discuss how to learn from rewards (possibly delayed), how to deal with very high-dimensional learning problems, how to embed all elements in a control system with real-time performance. The main difference between robot learning and many other machine learning methods is that robot learning addresses the question of making a robot perform certain tasks in the world successfully while machine learning addresses the problem of how to obtain one well-defined goal state from the current well-defined system state.

3. GA-based fuzzy reinforcement learning (GAFRL) agents

3.1. Structure of the GAFRL agent

The RL agent is usually formulated mathematically as an optimization problem with an objective of finding an action, or a strategy for producing control actions, that is, optimal in some well-defined ways. The proposed GA-based FRL (GAFRL) agent is shown in Fig. 3. It is based on an actor-critic

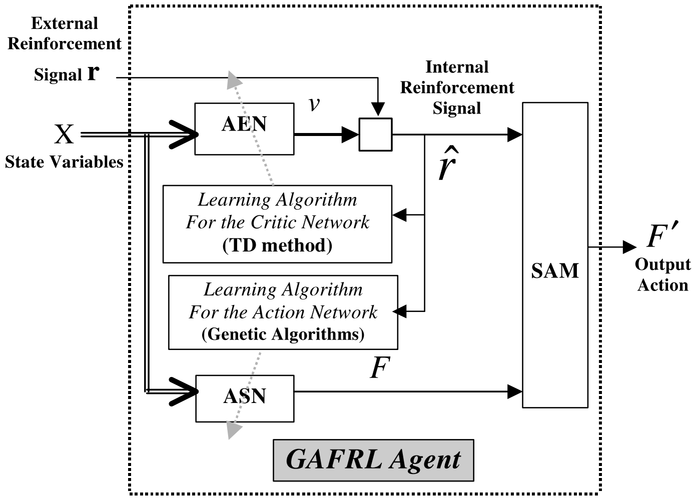
External Reinforcement Signal r
X
State Variables
AEN
v
Internal Reinforcement Signal
$\hat{r}$
Learning Algorithm For the Critic Network (TD method)
Learning Algorithm For the Action Network (Genetic Algorithms)
SAM
F′
Output Action
F
ASN
GAFRL Agent

Fig. 3. The GAFRL agent and its learning system.

architecture [1]. An adaptive critic network provides an internal reinforcement signal to an actor, which learns a policy for controlling the process. It has three components [2]: the action selection network (ASN), the action evaluation network (AEN), and the stochastic action modifier (SAM). The ASN maps a state vector into a recommended action F using fuzzy inference. The AEN maps a state vector and a failure signal into a scalar score that indicates state goodness. It models the environment so as to perform a multistep prediction of the external reinforcement signal. With the multistep prediction, the AEN can provide the ASN with a more informative internal reinforcement signal $\hat{r}$. The ASN can then determine a better action for the next time step according to the current environment state and $\hat{r}$. This strategy can solve the long delay reward problem with RL. The SAM uses both F and $\hat{r}$ to produce a desired control action F' which is applied to the plant. The details of this process were discussed in [2,19,23]. However, it should be noted that the learning element of the GAFRL agent is based on the TD approach and the GA, rather than the gradient descent learning method.

Based on the GAFRL architecture shown in Fig. 3 and the different kinds of expert knowledge available for assisting reinforcement learning, we can construct different types of GAFRL agents. The structural comparison of the GARL agent with three different types of GAFRL agents is given in Table 1. Since the GARL agent starts learning from scratch, it usually needs a large number of trials to learn its desired control action. For three different types of GAFRL agents, they use neural fuzzy networks to implement their critic networks and action networks. Therefore, a priori knowledge can be directly built into them in different ways. Since the GAFRL agent starts its learning

Table 1
The structural comparison of different types of GA-based reinforcement learning agents

	GARL agent	GAFRL agent (type A)	GAFRL agent (type B)	GAFRL agent (type C)
Action network	A 3-layer neural network	A 5-layer neural fuzzy network	A 5-layer neural fuzzy network	A 5-layer neural fuzzy network
Critic network	A 3-layer neural network	A 3-layer neural network	A 5-layer neural fuzzy network	A 5-layer neural fuzzy network
Evaluation feedback	Numerical	Numerical	Numerical	Fuzzy
Remarks	Both critic and action networks are initialized randomly.	The action network starts from an initial setting, but action-state evaluation starts from scratch.	Both action and critic networks can be initialized by available expert knowledge.	The fuzzy evaluative feedback is expected to provide a more informative external reinforcement signal.

Table 2
Use of different types of a priori knowledge in GA-based reinforcement learning agents

	GARL agent	GAFRL agent (type A)	GAFRL agent (type B)	GAFRL agent (type C)
Control knowledge	No	Yes	Yes	Yes
Evaluation knowledge	No	No	Yes	Yes
External reinforcement evaluation	No	No	No	Yes

from an initial configuration, it is expected to achieve faster learning than the GARL agent does. Table 2 shows how different kinds of a priori knowledge could be utilized by the GARL agent and the GAFRL agent.

3.2. Genetic algorithms

For solution of optimization problems, genetic algorithms (GAs) have been investigated recently and shown to be effective at exploring a large and complex space in an adaptive way, guided by the equivalent biological evolution mechanisms of *reproduction*, *crossover*, and *mutation* [5–7]. The mathematical framework of GAs was developed in the 1960s and was presented in Holland's pioneering book [7]. There are two major areas of GAs' application: optimization and machine learning. Some references on GAs and their implementations and applications can be found in [5,6]. This paper focuses on the study of GAs as an tool in the structure and parameter learning of the fuzzy reinforcement learning (FRL) agent.

As a general purpose stochastic optimization method for search problems, the GA is different from normal optimization and search procedures used in neural networks in several ways. (1) It works with a population of strings, searching many peaks in parallel. By using genetic operators, it exchanges information between the peaks, and hence lower in the possibility of ending at a local minimum and missing the global minimum. (2) It works with a coding of the parameters, not the parameters themselves. (3) The transition rules are probabilistic rather than deterministic. The randomized search is guided by the fitness value of each string and how it compares to others. Using the operators on the chromosomes which are taken from the population, the algorithm efficiently explores parts of the search space where the probability of finding improved performance is high. (4) The GA only needs to evaluate the objective function to guide its search, and there is no requirement for derivatives or other auxiliary knowledge. The only available feedback from the systems is the value of the performance measure (fitness) of the current population. Therefore, the most appropriate applications are for those problems where gradient information is unavailable or costly to obtain, such as the fuzzy reinforcement learning agent.

From the network learning point of view, since GAs only need the suitable evaluation of the network performance to yield the fitness values for evaluation, they are suitable for reinforcement learning problems [18]. However, the pure GA-based RL agent only uses external reinforcement signal without the predictions (internal critics) of the critic network. In this case, GAs cannot proceed to the next generation until the arrival of the external reinforcement signals. This causes the pure GA-based reinforcement learning converge rather slow, since an external reinforcement signal may only be available at a time long after a sequence of actions has occurred in the RL problems. In the following section, we will utilize an internal reinforcement signal predicated by the critic network, rather than an external reinforcement signal to guide the GA. By using the internal reinforcement signal as the fitness function, the GA can evaluate the candidate solutions (chromosomes) regularly even during the periods without external feedback from the environment. This idea is illustrated in Fig. 4.

3.3. Learning in the GAFRL agent

The proposed learning system for the GAFRL agent is constructed by integrating two feedforward multiple-layer networks. One neural network acts as the critic network (AEN) for guiding the learning of the action network (ASN). Another neural network is a neural controller (GARL agent) or a neural fuzzy controller (GAFRL agent) for determining the actions to be taken. The general structure of the proposed GAFRL learning system is shown in Fig. 3. The critic network can provide a more informative internal reinforcement signal to

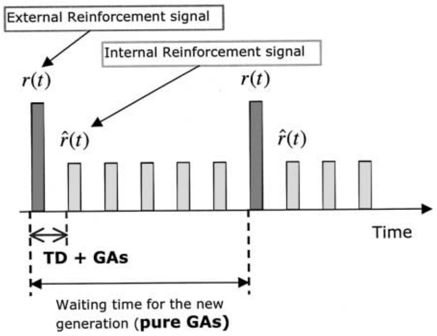
External Reinforcement signal
Internal Reinforcement signal
$r(t)$
$r(t)$
$\hat{r}(t)$
$\hat{r}(t)$
Time
TD + GAs
Waiting time for the new generation (pure GAs)

Fig. 4. Comparison of the TD + GA based reinforcement learning procedure with the pure GA-based reinforcement learning.

the action network. The action network can then determine a better action for the next time step based on the current environment state and the internal reinforcement signal. The internal reinforcement signal from the critic network enables both the action network and the critic network to learn without waiting for an external reinforcement, which may only be available at a time long after a sequence of actions has occurred. This can speed up the learning of both neural networks.

3.3.1. Learning algorithm for the critic network

The critic network evaluates the action recommended by the action network and generates the internal reinforcement signal.

$$\hat{r}[t+1] = \begin{cases} 0, & \text{start state,} \\ r[t+1] - v[t,t], & \text{failure state,} \\ r[t+1] + \gamma v[t,t+1] - v[t,t] & \text{otherwise,} \end{cases} \tag{3}$$

where $\hat{r}[t+1]$ is the internal reinforcement at time $t+1$, it plays the role of an error measure in the update of the output weights (since it is impossible to measure error directly). $\gamma \in [0,1]$ is the discount rate. $v[t,t]$ is an estimation of the state (a prediction of the external reinforcement signal). $r[t+1]$ is an external reinforcement signal.

The goal of training the critic network is to minimize the prediction error. It is similar to a reward/punishment scheme for neural networks. If $\hat{r}$ is positive, the value of the weights of the critic network should be rewarded by being changed in the direction which increase its contribution to the total sum. Otherwise, if $\hat{r}$ is negative, its weights will be punished. For both 3-layer critic network (non-fuzzy) and 5-layer critic network (neural fuzzy), we use a technique based on the temporal difference (TD) method [15] to train the critic networks. Because it is based on multistep prediction, this can ensure that both the critic network and the action network can update their parameters during the periods without any evaluative feedback (external reinforcement signals) from the environment.

The critic network architecture and training algorithms for the GARL agent and the GAFRL agent (type A) are the same as those in [2,19]:

$$b_i[t+1] = b_i[t] + \beta\hat{r}[t+1]x_i[t], \tag{4}$$

$$c_i[t+1] = c_i[t] + \beta\hat{r}[t+1]y_i[t,t], \tag{5}$$

$$a_{ij}[t+1] = a_{ij}[t] + \beta_h\hat{r}[t+1]y_i[t,t](1-y_i[t,t])\mathrm{sgn}(c_i[t])x_j[t], \tag{6}$$

where $\beta > 0$ and $\beta_h > 0$ are constants, x_i and y_i are input and output of the critic network, respectively, b_i is the weight on the links connecting the nodes in the input layer directly to the nodes in the output layer, c_i is the weight on the links between the hidden layer and the output layer, and a_{ij} is the weight from the jth input node to the ith hidden node.

For the GAFRL agent (types B and C), its critic network is a 5-layer neural fuzzy network. In this paper, its architecture and training algorithms are the same as those in [23].

3.3.2. Learning algorithm for the action network

The GA is used to train the action network (ASN) by using the internal reinforcement signal $\hat{r}$ from the critic network as the fitness function. To accelerate the GA's searching so as to speed up the GAFRL agent's learning, following methods have been considered in this paper.

(1) *Encoding of the action network.* The initial population of the GA is initialized by domain expert knowledge (fuzzy control rules) rather than generated randomly. Bit string encoding is the most common encoding technique [6]. However, with the real-value string encoding the GA can achieve much shorter searching time [18]. Hence we use the real-value encoding scheme instead of the traditional binary string encoding scheme in this paper. From Table 1, we can see that the GARL agent's action network is a 3-layer neural network while all the GAFRL agents' action networks are 5-layer neural fuzzy networks. Therefore, for the GARL agent, the encoding scheme of its action network is the same as the real-value-encoded GA in neural networks [18]. For the GAFRL agent's action network, an individual of the population represents one trial set of fuzzy membership functions [10]. As an example, if the action network has two inputs x_1 and x_2 with five linguistic terms, respectively, and one output F with seven linguistic terms (see Fig. 5), its real-value-encoding on chromosomes with population size p can be expresses as

- Chromosome 1: $(a_1^1, \ldots, a_5^1, b_1^1, \ldots, b_5^1, c_1^1, \ldots, c_7^1)$,
- Chromosome i: $(a_1^i, \ldots, a_5^i, b_1^i, \ldots, b_5^i, c_1^i, \ldots, c_7^i)$,
- Chromosome p: $(a_1^p, \ldots, a_5^p, b_1^p, \ldots, b_5^p, c_1^p, \ldots, c_7^p)$.

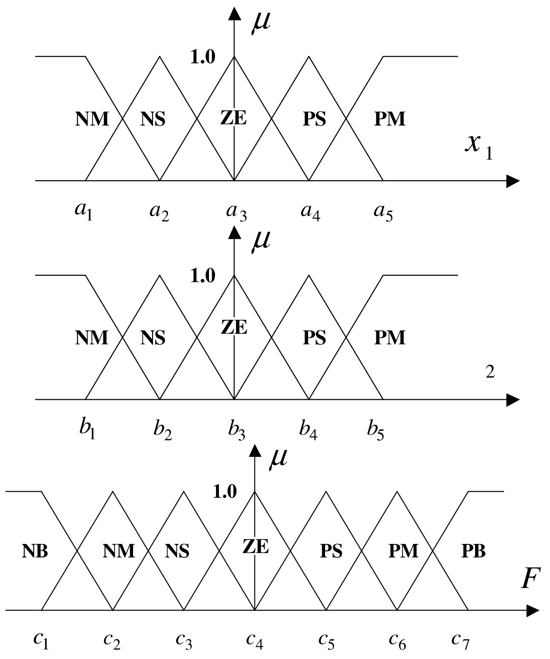
μ
1.0
NM
NS
ZE
PS
PM
x_1
a_1 a_2 a_3 a_4 a_5
μ
1.0
NM
NS
ZE
PS
PM
2
b_1 b_2 b_3 b_4 b_5
μ
1.0
NB
NM
NS
ZE
PS
PM
PB
F
c_1 c_2 c_3 c_4 c_5 c_6 c_7

Fig. 5. Examples of fuzzy membership functions for a 2-input 1-output action network.

In this way, the fuzzy rules are encoded in chromosomes. For each chromosome, a gene is represented by a floating point number or an integer. After a new generation is created, an interpreter takes the chromosomes and uses them to set the parameters in the action network. Then the action network suggests an action to control the environment for a fixed time or until a failure occurs. At the same time, the critic network predicts the external reinforcement signal from the controlled environment and provides an internal reinforcement signal to indicate the "fitness" of the action network.

(2) *Fitness assignment*. In this paper, the internal reinforcement signal $\hat{r}$ has been used to define the fitness function as follows [10]:

$$\mathrm{FIT(t)} = \frac{1}{|\hat{r}(t)|}. \tag{7}$$

It reflects the fact that smaller internal reinforcement values mean higher fitness of the action network. The string with the largest fitness value in the last generation is selected and decoded into the final action network. Using $\hat{r}(t)$ can usually accelerate the GA learning because it can evaluate the candidate solutions regularly during the periods without external feedback form the environment.

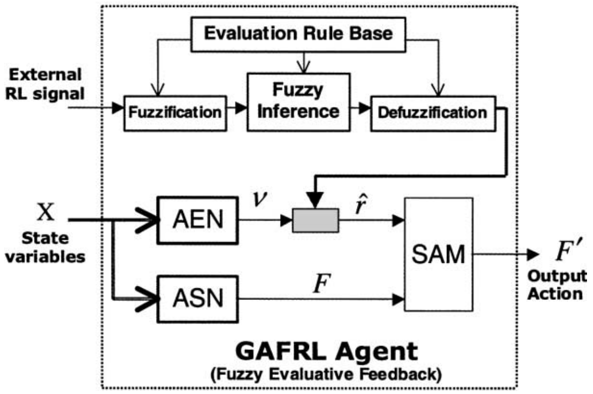

Fig. 6. The architecture of the GAFRL agent with fuzzy evaluative feedback (type C).

3.4. The GAFRL agent with fuzzy evaluative feedback

Using a crisp numerical (scalar) reinforcement signal for the RL agent is surely not the biologically plausible. Instead of using a numerical external reinforcement signal $r(t)$, we consider the reinforcement signal as a fuzzy number $R(t)$. We also assume that $R(t)$ is a fuzzy signal available at time step t and caused by the input and action chosen at time step $t-1$ or even affected by earlier inputs and actions. Let us assume the fuzzy reinforcement signal is expressed by a fuzzy number such that

$$R(t) \in \{R_1, R_2, \ldots, R_n\}. \tag{8}$$

It should satisfy the following inequality relation:

$$-1 \leqslant \text{defuzzifier}(R_1) \leqslant \text{defuzzifier}(R_2) \leqslant \cdots \leqslant \text{defuzzifier}(R_n) \leqslant 1, \tag{9}$$

where defuzzifier(R_i) represents the discrete degree of reward or penalty.

Based on a modified architecture of the GAFRL agent with numerical evaluative feedback as shown in Fig. 3, we propose a GAFRL agent with fuzzy evaluative feedback (type C). Its architecture is given in Fig. 6. It can generate the external reinforcement signal using fuzzy evaluation rule base by means of a fuzzy inference system which is similar to a fuzzy controller. Since the fuzzy evaluative feedback is a kind of continuous signal rather than a discrete one (see Fig. 2). Hence, it can provide more detail information about the environments.

4. Biped learning using GAFRL agents

There is no unique problem solution for the biped dynamic walking control. Any trajectory through the state space (with constrains) that does not result in a failure signal (fall down) is acceptable. To synthesize the biped walking motion, it is required to take a workspace variable p from an initial position p_i at time t_i to a final position p_f at time t_f. The motion trajectory for $p(t)$ can be obtained by solving a dynamic optimization problem. To achieve the dynamic balance, the zero moment point (ZMP) [17] is usually used as a criterion, therefore, we can determine the biped motion to minimize the following performance index:

$$\text{minimize} \int_{t_i}^{t_f} \|P_{\text{zmp}}(t) - P^d_{\text{zmp}}(t)\|^2 \,\mathrm{d}t \tag{10}$$

subject to the boundary conditions of both $p(t)$ and $\dot{p}(t)$ at time t_i and t_f, where P_{zmp} is the actual ZMP, and P^d_{zmp} is the desired ZMP position. The control objective of the gait synthesis for the biped dynamic balance can be described as

$$P_{\text{zmp}} = (x_{\text{zmp}}, y_{\text{zmp}}, 0) \in S, \tag{11}$$

where $(x_{\text{zmp}}, y_{\text{zmp}}, 0)$ is the coordinate of the ZMP with respect to O–XYZ. S is the domain of the supporting area.

In order to reduce the complexity of the biped dynamic balancing problem, many researchers assume that the balance in the sagittal and frontal planes is independent [8]. Thus, we consider the dynamic balance in each plane separately. The proposed biped dynamic balance and learning scheme is shown in Fig. 7. There are two independent GAFRL agents, namely, GAFRL Agent-x and GAFRL Agent-y for sagittal and frontal planes, respectively. Each GAFRL agent only searches its relevant state–action space so as to speed up biped learning.

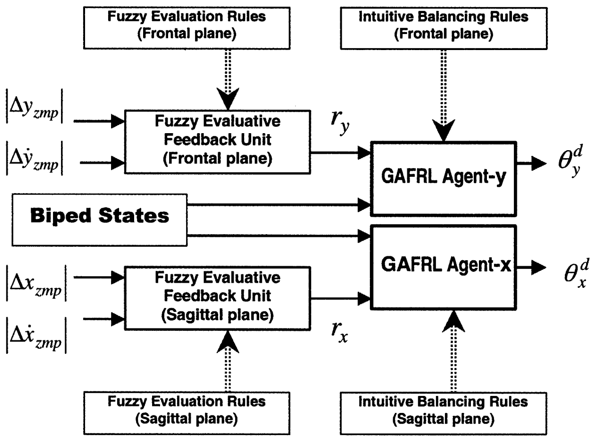
Fuzzy Evaluation Rules (Frontal plane)
Intuitive Balancing Rules (Frontal plane)
$|\Delta y_{zmp}|$
$|\Delta \dot{y}_{zmp}|$
Fuzzy Evaluative Feedback Unit (Frontal plane)
r_y
GAFRL Agent-y
θ^d_y
Biped States
GAFRL Agent-x
θ^d_x
$|\Delta x_{zmp}|$
$|\Delta \dot{x}_{zmp}|$
Fuzzy Evaluative Feedback Unit (Sagittal plane)
r_x
Fuzzy Evaluation Rules (Sagittal plane)
Intuitive Balancing Rules (Sagittal plane)

Fig. 7. The block diagram of the biped dynamic balancing control using two independent GAFRL agents.

5. Simulation results

In this section, we apply the multi-GAFRL-agent-based biped dynamic balance control method (see Fig. 7) to a simulated biped robot. Its physical structure is shown in Fig. 8. Fig. 9 illustrates its three-dimensional link structure. The total mass is about 12 kg. The maximum hip height is about 70 cm. It is headless and armless. Each leg has five active DOF of which two DOF are available at the hip, one at the knee and two at the ankle joint. Each of the feet has four force sensors (two at the toe and two at the heel) to provide the contact forces between the feet and the ground.

5.1. The configuration of the GAFRL agent

For the GAFRL agent in Fig. 7, each action network has five layers. There are two inputs in the Layer 1, 10 units in the Layer 2 (there are five antecedent

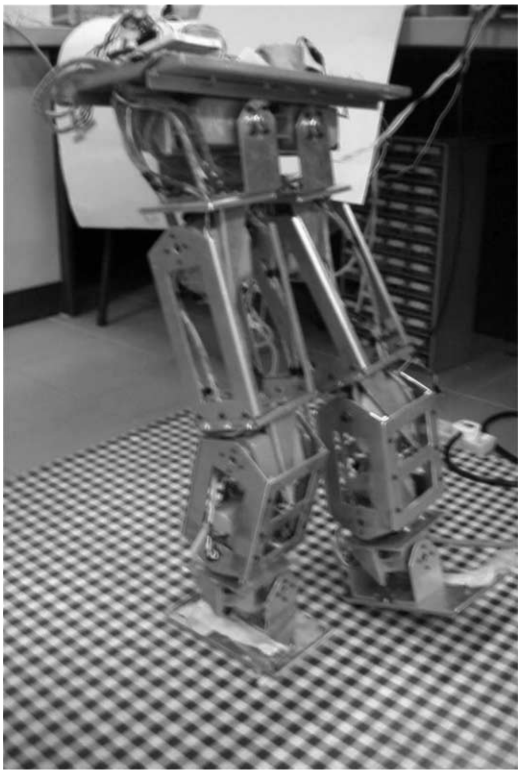

Fig. 8. The physical structure of the biped robot.

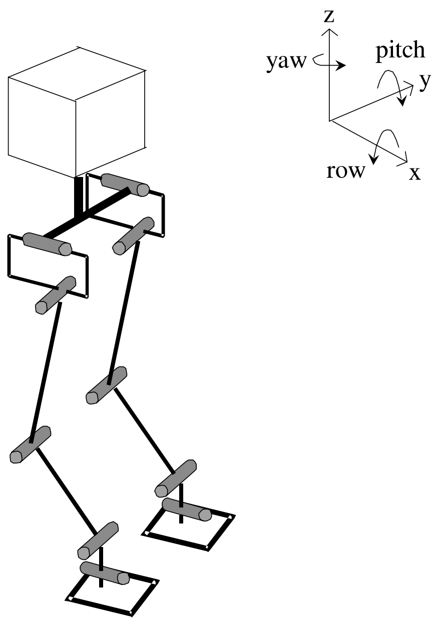

Fig. 9. The link structure of the biped robot.

labels, negative medium (NM), negative small (NS), zero (ZE), positive small (PS), and positive medium (PM) for each input). There are 25 units in the Layer 3 (the number of the balancing rules), seven units in the Layer 4 (there are seven consequent labels, negative big (NB), negative medium (NM), negative small (NS), zero (ZE), positive small (PS), positive medium (PM), and positive big (PB) for the output). For the Layer 5, there is only one output unit to compute the desired control action. The critic network has three layers for the GARL agent and five layer for the GAFRL agent. For the 3-layer network, it has two input units, a bias input unit, five hidden units and an output unit. The input state vector is normalized so that the error of the ZMP positions lie in the range $[-1, 1]$. The weights of the network are initialized randomly to values in $[-0.1, 0.1]$. The discount rate, γ, used in the calculation of Eq. (3), is 0.9. The β and β_h in Eqs. (4)–(6) are set as 0.2. For the GAFRL agent, the critic network can be initialized by the fuzzy evaluation rules. Its configuration is similar to that of the action network. The fuzzy evaluative feedback unit in Fig. 7 is configured as a fuzzy controller according to the fuzzy rules for evaluating the biped dynamic balance in the sagittal or frontal plane.

In GA-based reinforcement learning, an individual of the population represents one trial set of all the fuzzy membership functions. For the GAFRL Agent-x in Fig. 7, there are two input variables $x_1 = \Delta x_{\mathrm{zmp}}$ and $x_2 = \Delta \dot{x}_{\mathrm{zmp}}$, and one output variable $F = \theta_x^d$. The types of membership functions used for the input and output variables are shown in Fig. 5. It can be seen that there are 17 parameters (5 for x_1, 5 for x_2 and 7 for θ_x^d) which are concatenated into a real-value string as an individual for GA learning. There are also another 17 parameters in the GAFRL Agent-y for biped dynamic balance in the frontal plane. As mentioned before, these two agents can be trained independently. Therefore, each string contains 17 gens (parameters) rather than 34 gens. Fig. 10 shows an example of the learned membership functions for the GAFRL Agent-x's output. Where the part of the sting in the initial individual for GA learning can be expressed as $(\ldots, -10, -6.6, -3.3, 0.0, 3.3, 6.6, 10, \ldots)$. After GA learning, the above string was changed to $(\ldots, -8.90, -4.06, -1.78, 0.0, 3.31, 7.78, 9.91, \ldots)$. The fitness functions for the GAFRL Agent-x and Agent-y are defined as $\mathrm{FIT}_x(t) = 1/|\hat{r}_x(t)|$ and $\mathrm{FIT}_y(t) = 1/|\hat{r}_y(t)|$, respectively. The learning parameters used in the GA-based reinforcement learning are set as follows [10,18]. (1) The crossover probability is set as 0.9 and the crossover sites are randomly selected. (2) The mutation probability is set as 0.1 and the mutation site is random selected. A random value with range ± 1 is added to the chosen site. (3) the population size is set as POP = 100. A learned strategy would be considered to be successful if the individual with maximum fitness and the ZMP is always within the supporting area for certain time. Otherwise the evolution of the next generation will be started again.

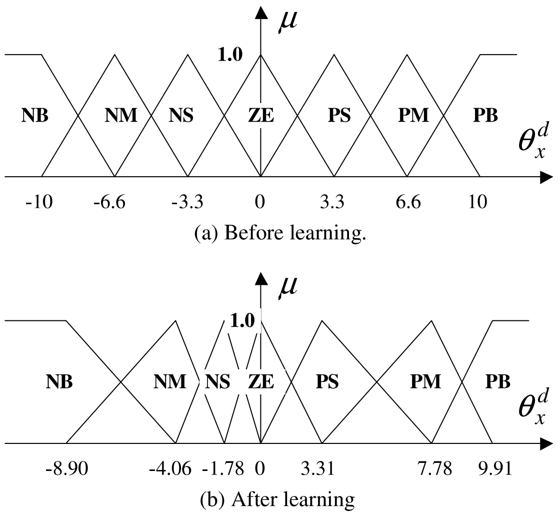

Fig. 10. Membership functions used for the GAFRL Agent-x's output θ_x^d. (a) Initial setting of the membership functions. (b) The learned membership functions after GA-based reinforcement learning.

5.2. Use of intuitive knowledge to assist biped learning

(1) *Intuitive biped balancing knowledge.* There are many methods to derive fuzzy rules for the biped learning and control [19,21,23]. Based on intuitive balancing knowledge, 25 fuzzy rules for the sagittal plane are obtained as shown in Table 3. They can be used to initialize the GAFRL Agent-x, thus it does not start learning from scratch. As an example, the following is a fuzzy rule to balance the biped robot in the sagittal plane:

IF Δx_{zmp} is **Positive Medium**

AND $\Delta \dot{x}_{\text{zmp}}$ is **Negative Small**

THEN the body swings in **Negative Small**

where $\Delta x_{\text{zmp}} = x_{\text{zmp}} - x^d_{\text{zmp}}$ is the displacement of the ZMP, x_{zmp} is the actual ZMP in the sagittal plane, and x^d_{zmp} is the desired ZMP. The balancing rules for the frontal plane are similar to the rules in Table 3.

(2) *Evaluation knowledge on biped dynamic balance.* To evaluate biped dynamic balance in the sagittal plane, a penalty signal should be given if the biped robot falls down in the sagittal plane, that is, x_{zmp} is not within the supporting area in the sagittal plane:

$$r_x = \begin{cases} 0 & \text{if } x_{\text{zmp}} \in S_x \text{ and } |\Delta \dot{x}_{\text{zmp}}| \leqslant v_{\text{xu}}, \\ -1 & \text{otherwise} \end{cases} \tag{12}$$

where S_x stands for the supporting area in the sagittal plane, v_{xu} is used to limit the velocity of the ZMP in the sagittal plane.

The above two-value evaluative feedback signals can only describe the simple “go – no go” or “fall down – stand” walking states. It is less informative than the human evaluation on walking. To provide more detailed information on the evaluative feedback, we provide some fuzzy rules as shown in Table 4 to evaluate the biped dynamic balance in the sagittal plane (the fuzzy rules to evaluate the biped dynamic balance in the frontal plane are similar to the rules in Table 4). As an example, the following fuzzy rules can be used to evaluate the biped dynamic balance in the sagittal plane [20]:

Table 3
The fuzzy rules for biped dynamic balancing in the sagittal plane

θ^d_x		$\Delta \dot{x}_{\text{zmp}}$				
		NM	NS	ZE	PS	PM
Δx_{zmp}	NM	PB	PB	PM	PS	ZE
	NS	PB	PM	PS	ZE	NS
	ZE	PM	PS	ZE	NS	NM
	PS	PS	ZE	NS	NM	NB
	PM	ZE	NS	NM	NB	NB

Table 4
The fuzzy rules for evaluating the biped dynamic balance in the sagittal plane

r_x		$\|\Delta\dot{x}_{\mathrm{zmp}}\|$		
		Zero	Small	Big
$\|\Delta x_{\mathrm{zmp}}\|$	Zero	Very Good	Good	OK
	Small	Good	OK	Bad
	Big	Bad	Very Bad	Very Bad

Rule 1: *IF* $|\Delta x_{\mathrm{zmp}}(t)|$ is **Zero** *AND* $|\Delta\dot{x}_{\mathrm{zmp}}(t)|$ is **Small** *THEN* $r_x(t)$ is **Good**
Rule 2: *IF* $|\Delta x_{\mathrm{zmp}}(t)|$ is **Small** *AND* $|\Delta\dot{x}_{\mathrm{zmp}}(t)|$ is **Big** *THEN* $r_x(t)$ is **Bad**
where $r_x(t)$ is the external reinforcement signal for the FRL Agent-x. $|\Delta x_{\mathrm{zmp}}(t)|$ is the displacement of the ZMP in the sagittal plane.

5.3. Results and discussion

In the following, we will illustrate how to incorporate different kinds of expert knowledge mentioned in above section in the GAFRL agent so as to accelerate the biped learning. We will also demonstrate how to use the GAFRL agent to the biped dynamic balance control and hence to improve the biped gait. According to the different kinds of a priori knowledge available to assist biped learning, we conduct a simulation study in four cases as shown in Table 5.

In the simulation analysis, we observe that the learning rate of the biped robot can be improved tremendously when the expert knowledge is applied to the GAFRL agent. From Fig. 11 and Table 5, we found that the learning rate is comparable in terms of different kinds of reinforcement learning agents. In case I, the GARL agent with numerical evaluative feedback is used. There is no expert knowledge available for its initialization. Therefore, it starts learning from a random setting. It achieves 120 s of walking at the 88th iteration (each iteration corresponds to each attempt of the biped to walk until a failure

Table 5
The different kinds of knowledge for biped learning

	Learning agents	The knowledge available for biped learning
Case I	GARL	No expert knowledge is available. Only numerical reinforcement signal is used to train the gait synthesizer.
Case II	GAFRL (type A)	Only the intuitive biped balancing knowledge is used as the initial configuration of the gait synthesizer.
Case III	GAFRL (type B)	Both the intuitive biped balancing knowledge and walking evaluation knowledge are utilized.
Case IV	GAFRL (type C)	Besides all the knowledge used in case III, the fuzzy evaluative feedback, rather than numerical evaluative feedback, is included.

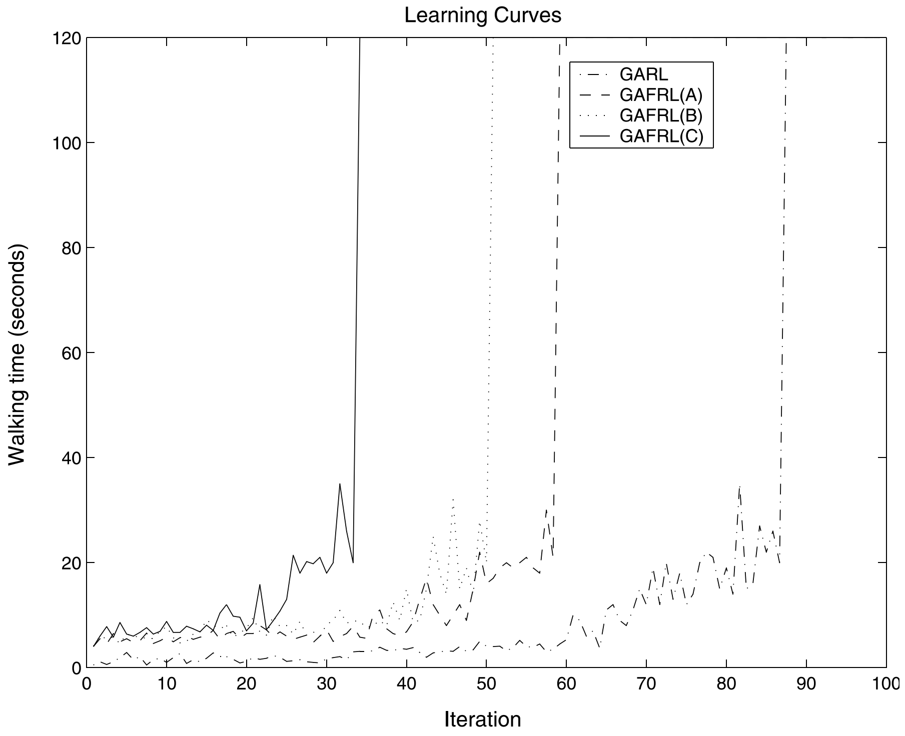

Fig. 11. The learning curves of different types of reinforcement learning agents for the simulated biped robot.

condition is encountered). In case II, the GAFRL agent (type A) is chosen. Its action network is initialized by a set of fuzzy control rules given in Table 3. Starting from this initial configuration, the biped can walk a few seconds before starting its training. After 59 iterations, it can achieve 120 s walking. Thus, we can deduce that the expert knowledge on human balance dose improve the biped learning rate. In case III, we use the GAFRL (type B) to incorporate both balancing knowledge and walking evaluation knowledge (see Table 4). It can be seen that the biped can walk up to 120 s after 50 iterations learning. Furthermore, in case IV, the GAFRL agent (type C) is chosen to make use of both balancing knowledge and walking evaluation knowledge and fuzzy evaluative feedback. It only needs 33 iterations learning to make the biped walk at least 120 s. We may conclude that the incorporation of expert knowledge into the GAFRL agents can improve the biped robot learning rate significantly.

We also conduct a simulation to compare the GAFRL agent with the FRL agent. The configuration and implementation of the FRL agent for biped learning can be found in [23]. For both simulations, the same simulated biped

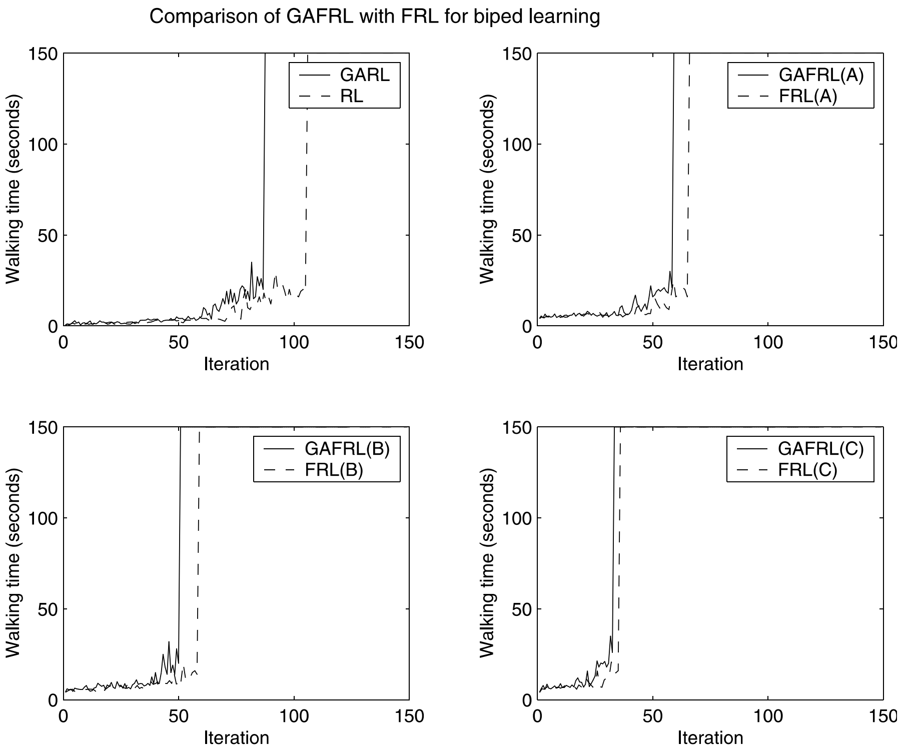

Fig. 12. Comparison of fuzzy reinforcement learning with GA-based fuzzy reinforcement learning for the simulated biped robot.

robot as shown in Figs. 8 and 9 is used. The result is given in Fig. 12. We can find that the performance of the GAFRL agent is better than that of the FRL agent in all the four cases as shown in Table 5. We may conclude that the GA learning dose speed up the reinforcement learning. Note that the learning rates are almost same for the GAFRL agent (type C) and the FRL agent (type C). This may be because the starting setting of the agent is quite close to the desired solution since different kinds of a priori knowledge have been utilized. In this situation, the main drawback of the gradient-based reinforcement learning, i.e., local minima problem, dose not obviously exist.

To demonstrate how the fuzzy evaluative feedback can be used to assist the biped learning, the comparison of the ZMP tracking results is given in Figs. 13 and 14. The desired ZMP trajectory is obtained from the human locomotion studies. At the beginning, the ZMP is under the heel, then it moves to the foot center, and finally it is shifted under the toes. At the end of a half step, the ZMP jumps so that it is under the other foot, which is in contact with the ground. After 50 trials, the ZMP tracking result using GAFRL agent with

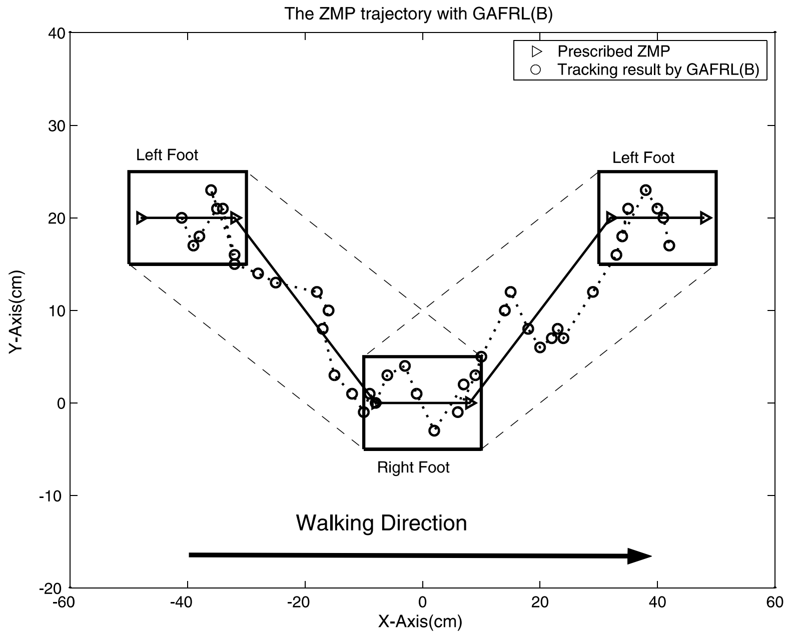

Fig. 13. The ZMP tracking result using GAFRL agent (type B) with numerical evaluative feedback.

numerical evaluative feedback (type B) is shown in Fig. 13 while Fig. 14 illustrates the ZMP tracking result using the GAFRL agent with fuzzy evaluative feedback (type C). It can be seen that the ZMP is much closer to the desired trajectory by using the GAFRL agent (type C), and the smoother biped balancing control can be achieved. This is probably because the learning objective of the GAFRL agent (type B) is to ensure the ZMP is always inside the supporting area (see Eq. (12)), while the ZMP must track the prescribed trajectory as near as possible to get "Good" or "very Good" credit for the GAFRL agent (type C) (see Table 4). In this sense, the ZMP tracking result in Fig. 13 should be considered very good because the ZMP is always inside the supporting area.

6. Concluding remarks

We propose a general GA-based fuzzy reinforcement learning (GAFRL) architecture for robot learning. It can incorporate different kinds of expert knowledge in the GAFRL agent so as to accelerate its learning rate. By making use of the global optimization capability of GAs, the GAFRL can solve the local minima problem in traditional actor-critic reinforcement learning. On

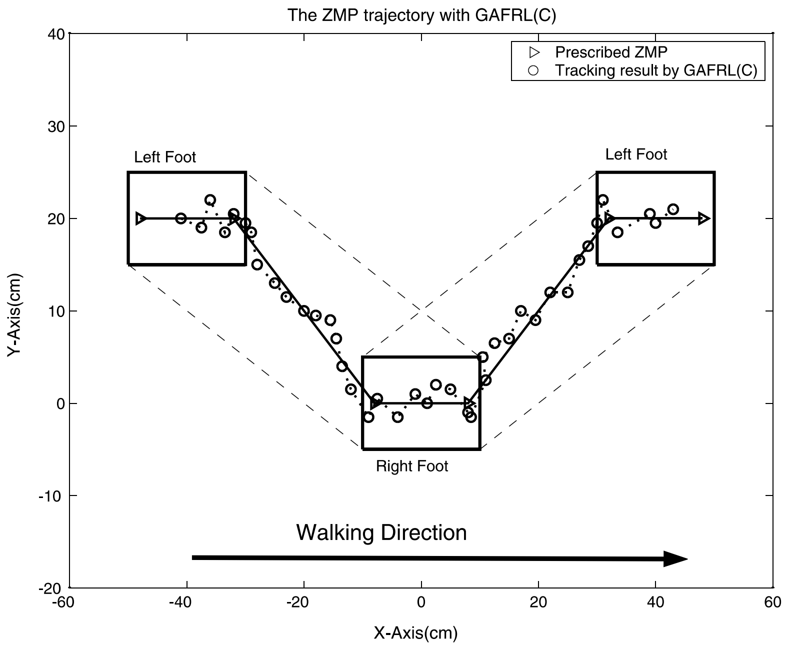

Fig. 14. The ZMP tracking result using FRL agent (type C) with fuzzy evaluative feedback.

the other hand, since the fitness function of the GA is constructed by using the internal reinforcement signal, it can provide more informative information to speed up the GA's learning. The simulation analysis demonstrates that it is possible for a robot to start with heuristic knowledge and to learn to refine it with the GAFRL agents proposed in this paper.

Humans have remarkable learning capability without using any measurements and any computations in the uncertain and unpredictable environments. In performing such learning tasks, for the most of cases, humans use perceptions rather than measurements. Hence, how to incorporate different perception-based information [22] to the GAFRL agent and other machine learning methods, such as imitation learning [13], to speed up robot learning will be our future research.

Acknowledgements

We would like to thank J. Kanniah, S. H. Ker and S. C. Oh at the Singapore Robotic Games (SRG) Lab of Singapore Polytechnic for their support in biped hardware development. The research described in this paper was made possible by the support of the Singapore Tote Fund.

References

[1] A.G. Barto, Reinforcement learning, in: O. Omidvar, D.L. Elliott (Eds.), Neural Systems for Control, Academic Press, New York, 1997, pp. 7–30.

[2] H.R. Berenji, P.S. Khedkar, Learning and tuning fuzzy logic controllers through reinforcements, IEEE Trans. Neural Networks 3 (1992) 724–740.

[3] R.A. Brooks, M.J. Martaric, Real robots, real learning problems, in: J.H. Connell, S. Mahadevan (Eds.), Robot Learning, Kluwer Academic Publishers, Dordrecht, 1993.

[4] J.H. Connell, S. Mahadevan, Introduction to robot learning, in: J.H. Connell, S. Mahadevan (Eds.), Robot Learning, Kluwer Academic Publishers, Dordrecht, 1993.

[5] L. Davis, Handbook of Genetic Algorithms, Van Nostrand Reinhold, New York, 1991.

[6] D.E. Goldberg, Genetic Algorithms in Search, Optimization and Machine Learning, Addison-Wesley, Reading, MA, 1989.

[7] J.H. Holland, Adaptation in Natural and Artificial Systems, University of Michigan, Ann Arbor, 1975.

[8] A.L. Kun, T.W. Miller, Control of variable speed gaits for a biped robot, IEEE Robot. Automat. Mag. 63 (1999) 19–29.

[9] C.-T. Lin, M.-C. Kam, Adaptive fuzzy command acquisition with reinforcement learning, IEEE Trans. Fuzzy Systems 6 (1998) 102–121.

[10] C.-T. Lin, C.-P. Jou, GA-based fuzzy reinforcement learning for control of a magnetic bearing system, IEEE Trans. Syst., Man Cybern. B 30 (2000) 276–289.

[11] M.J. Mataric, Learning in behavior-based multi-robot systems: policies, models, and other agents, Cognitive Syst. Res. 2 81–93.

[12] A.W. Salatian, K.Y. Yi, Y.F. Zheng, Reinforcement learning for a biped robot to climb sloping surfaces, J. Robot. Syst. 14 (1997) 283–296.

[13] S. Schaal, Is imitation learning the route to humanoid robots, Trends Cognitive Sci. 3 (1999) 1040–1046.

[14] S. Schaal, Robot learning, in: M. Arbib (Ed.), Handbook of Brain Theory and Neural Networks, second ed., MIT Press, Cambridge, MA (in press).

[15] R.S. Sutton, A.G. Barto, Reinforcement Learning: An Introduction, MIT Press, Cambridge, MA, 1998.

[16] C.F. Touzet, Neural reinforcement learning for behavior synthesis, Robot. Auton. Syst. 22 (1997) 251–281.

[17] M. Vukobratovic, B. Borovac, D. Surla, D. Stokic, Biped Locomotion: Dynamics, Stability, Control and Application, Springer, Berlin, 1990.

[18] D. Whitley, S. Dominic, R. Das, C.W. Anderson, Genetic reinforcement learning for neurocontrol problems, Mach. Learning 13 (1993) 259–284.

[19] C. Zhou, Neural fuzzy gait synthesis with reinforcement learning for a biped walking robot, Soft Comput. 4 (2000) 238–250.

[20] C. Zhou, Q. Meng, Reinforcement learning with fuzzy evaluative feedback for a biped robot, in: Proceedings of the IEEE International Conference on Robotics and Automation, 2000, pp. 3829–3834.

[21] C. Zhou, Incorporation of expert knowledge in robot learning using GA-based fuzzy reinforcement learning agents, in: Proceedings of the 4th Asian Conference on Robotics and its Application, 2001, pp. 152–174.

[22] C. Zhou, Y. Yang, X. Jia, Incorporating perception-based information in reinforcement learning using computing with words, Lect. Notes Comput. Sci. 2085 (2001) 476–483.

[23] C. Zhou, Q. Meng, Dynamic balance of a biped robot using fuzzy reinforcement learning agents, Fuzzy Sets and Systems, to appear.