2007 IEEE International Conference on
Robotics and Automation
Roma, Italy, 10-14 April 2007

# A Reinforcement Learning Based Dynamic Walking Control

Yong Mao, Jiaxin Wang, Peifa Jia, Shi Li, Zhen Qiu, Le Zhang and Zhuo Han

*State Key Laboratory of Intelligent Technology and Systems,*
*Department of Computer Science and Technology, Tsinghua University,*
*Beijing 100084, China*

*maoyong97@mails.tsinghua.edu.cn*

***Abstract* - A quasi-passive dynamic walking robot is built to study natural and energy-efficient biped walking. The robot is actuated by MACCEPA actuators. A reinforcement learning based control method is proposed to enhance the robustness and stability of the robot's walking. The proposed method first learns the desired gait for the robot's walking on a flat floor. Then a fuzzy advantage learning method is used to control it to walk on uneven floor. The effectiveness of the method is verified by simulation results.**

***Index Terms* - *passive dynamic walking, reinforcement learning, fuzzy advantage learning, biped robot.***

## I. Introduction

Passive Dynamic Walking (PDW) research was originally proposed by McGeer [1]. His PDW robots were able to walk down an inclined floor without any actuators. Compared with biped robots based on zero moment point (ZMP) stable criterion and mainstream trajectory tracking methods, PDW robots walk in a more natural and energy-efficient way and are regarded as an important way to study the underlying principles of human walking. Instead of trying to control the robot to keep balance according to the ZMP criterion, stable passive dynamic walking is viewed as a periodical motion. It is understood as a continuous passive fall, only intermittently interrupted by a change of foot contact [2]. During this process the passive dynamics of the robots are fully made use of, through which natural and energy-efficient walking is achieved.

After McGeer, several PDW and quasi-PDW robots were built [3]. Although natural and energy-efficient walking was achieved easily in PDW, the stability and robustness of the robots remained impractical. The stability of the PDW robot relies on the initial condition and the limited cycle of the mechanism. And the quasi-PDW robots controlled by simple feedback controllers are sensitive to disturbances when walking in rough terrain.

To enhance the stability and robustness of quasi-PDW robots, reinforcement learning controllers were introduced to control the robots' walking [4,5]. Morimoto et al. [5] demonstrated a poincaré-map based reinforcement learning. However, this method uses a human walking pattern as the target trajectory for the robot. It is difficult to decide whether this pattern is appropriate for robots, because of marked physical disparities between humans and robots. In this paper we proposed a method which automatically learns a target walking pattern for our planar quasi-PDW robot. Then, based on the learned gait and control strategy, a fuzzy advantage learning (FAL) method is used to control the robot's walking in uneven terrain. In this method, with the strong generalization ability of fuzzy inference systems (FIS), the stability and robustness of the robot is enhanced.

This paper is organized as follows. Section 2 describes the quasi-PDW robot and its dynamics model. Section 3 proposes the reinforcement learning method which learns the target walking gait for the robot and the FAL method which controls the robot's walking based on the learned gait. Simulation results are shown in section 4. Finally in section 5, conclusions and future works are given.

## II. Robot Prototype and Dynamics Model

### *A. Mechanical Prototype*

The quasi-PDW walker shown in Fig.1 has 2 legs and a trunk. Each leg has a thigh, a shin and a curved foot which is connected to the shin without an ankle joint. The 2 hip joints and 2 knee joints are actively actuated. An angle-bisecting mechanism [6] is used to keep the trunk always on the two legs' angle bisector. Thus the robot has 3 internal degrees of freedom (DOF) - at the hip and the knees. Each knee joint has a knee lock and a knee cap. When the swing leg is fully extended, or when the leg is in supporting mode, the knee lock is active and keeps the leg straight. The knee cap prevents the knee from over-extension. The mechanical parameters are listed in Table.1.

The robot is actuated by MACCEPA actuators [7]. MACCEPA actuator is made up of two RC motors and a spring. The joint actuated by this kind of actuators can be viewed as driven by a torsional spring with independently controlled stiffness and equilibrium position. At each joint a rotational potentiometer is mounted as a sensor to read the joint angle. Two detecting sensors are mounted on each foot to give out the exact moments of collisions and foot clearances. A LF2407 DSP (digital signal processor) is used to control the RC motors and process the signals of the sensors.

### *B. Dynamics Model*

A simulation system is built for the robot. The dynamic model is shown in Fig.2. The dynamics equations of the biped robot are derived by Newton-Euler equations and the TMT method [2]. It has the following form:

1-4244-0602-1/07/$20.00 ©2007 IEEE.

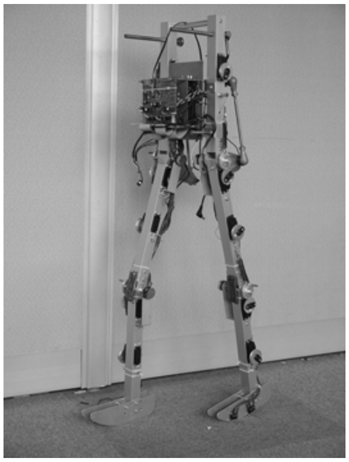

Fig.1 Robot prototype

TABLE I
MECHANICAL PARAMETERS OF THE ROBOT

| | M /kg | L/m | I/(kg·m2) | R/m |
|---|---|---|---|---|
| Trunk | 0.5 | 0.35 | 0.005 | - |
| Thigh | 0.5 | 0.3 | 0.004 | - |
| Shin | 0.5 | 0.3 | 0.004 | - |
| Foot | 0.1 | - | - | 0.2 |

$$M(\theta)\ddot{\theta} + C(\theta,\dot{\theta})\dot{\theta} + g(\theta) = \tau\,, \tag{1}$$

where $\theta$ is the generalized coordinate vector, $\dot{\theta}$ is the generalized velocity vector, $\ddot{\theta}$ is the generalized acceleration vector, $M(\theta)$ is the generalized mass matrix, $C(\theta,\dot{\theta})$ is the matrix of centrifugal and Coriolis terms, $g(\theta)$ is the gravity terms, $\tau$ is the torque vector applied by the MECCAPA actuators which is derived by the virtual work principle as follows:

$$\tau = [0,0,-(\tau_1-\tau_3)/2,(\tau_1-\tau_3-2\times\tau_4)/2,\tau_4,0]^T\,.$$

The definition of the generalized coordinates is changed during the walking process. When leg1 is the supporting leg and the knee joint of leg1 is locked, the generalized coordinates are chosen as $\theta = [x_h, y_h, \theta_1, \theta_3, \theta_4, \theta_5]^T$ , where $x_h$ and $y_h$ are the coordinates of the hip joint, $\theta_i$ is defined in Fig.2. As aforementioned, the robot has 3 DOFs, so there are 3 redundant coordinates in the generalized coordinates. Thus, constraint functions are added as follows:

$$x_h = -(l_1 + l_2 - r)\sin(\theta_1) - r\cdot\theta_1\,, \tag{2}$$

$$y_h = (l_1 + l_2 - r)\cos(\theta_1) + r\,, \tag{3}$$

$$\theta_5 = (\theta_1 + \theta_3)/2\,, \tag{4}$$

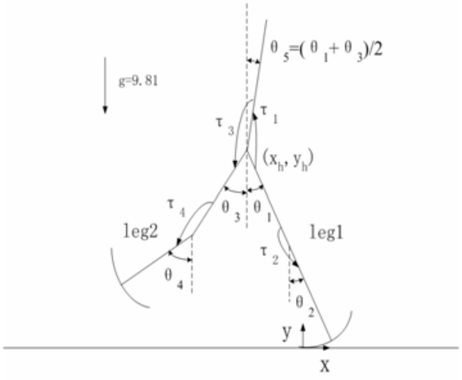

Fig.2 Dynamics model

where r is the radius of the curved foot. To apply the constraints to the dynamics equations, a Lagrange multiplier method is used. Let $\lambda$ be the Lagrange multiplier vector, $\lambda = [\lambda_1, \lambda_2, \lambda_3]^T$ where $\lambda_1$ and $\lambda_2$ denote the x-direction and y-direction forces between the supporting foot and the ground respectively, and $\lambda_3$ denotes the torque between the two legs and the trunk applied by the bisecting mechanism. Combine (1), (2), (3) and (4), the complete dynamics function is derived as:

$$\begin{bmatrix} M & A^T \\ A & 0 \end{bmatrix}\begin{bmatrix} \ddot{\theta} \\ \lambda \end{bmatrix} = \begin{bmatrix} -C(\theta,\dot{\theta})\dot{\theta} - g(\theta) + \tau \\ -B^T\dot{\theta}^2 \end{bmatrix}, \tag{5}$$

where $A$ and $B$ are terms in the second derivative forms of the constraint functions.

Then, the values of the generalized accelerations and the multipliers are solved directly from (5). With these values, the one step walking process is simulated by a numerical integration of the generalized accelerations.

To simulate the complete process of walking, we need to model the collision between the swinging leg and the ground. Suppose the collision is a perfect plastic collision, the swing leg is fully extended and the knee joint is locked when collision occurs. The generalized velocities after collision are derived from the generalized velocities before collision by the law of conservation of momentum. The equation is as follows:

$$\begin{bmatrix} M & A^T \\ A & 0 \end{bmatrix}\begin{bmatrix} \dot{\theta}^+ \\ \lambda \end{bmatrix} = \begin{bmatrix} M\dot{\theta}^- \\ 0 \end{bmatrix}, \tag{6}$$

where $\dot{\theta}^+$ and $\dot{\theta}^-$ are the generalized velocities after and before collision respectively, the Lagrange multiplier $\lambda$ denotes the impulses.

In this simulation, the MACCEPA actuator is modelled as a spring-damper system. The dynamics equation of the actuator at the $i_{th}$ joint is shown as:

$$\tau_i = -k_i \times (\phi_i - \varphi_i) - b_i \times \dot{\phi}_i\,, \tag{7}$$

where $\tau_i$ is the torque applied by the actuator, $\phi_i$ is the relative joint angle between the two links, $\dot{\phi}_i$ is the joint velocity, $b_i$ is the damper coefficient which is set to 0.1 during the simulation. The stiffness coefficient $k_i$ and the equilibrium angle $\varphi_i$ are actively controlled.

## III. Reinforcement Learning Based Dynamic Walking Control

Reinforcement learning is a series of learning methods which learn from the interaction between the agent and the environment. The learning algorithm maintains an action selecting policy which maps the states to the actions. The agent's sole objective is to adjust the policy according to the reinforcement signals it receives from the environment so as to maximize the total reward it receives in the long run [8]. It explores the state-action space through trial-and-error search to find the optimal policy $\pi$ which maximizes the action-value function $Q(s,a)$:

$$Q^*(s,a) = \max_\pi Q^\pi(s,a), \tag{8}$$

for all $s \in S$, and $a \in A(s)$. In (8), $S$ is the state set, $A(s)$ is the action set of a given state, Q* is the optimal action-value function. Having Q*, we can easily achieve the optimal action choosing.

Our method is based on reinforcement learning. It has 2 steps. Firstly, we divide the walking process of the walker into several phases and learn the desired gait for the robot's walking on flat floor based on a Q-learning algorithm. Then, using the learned gait, we apply a FAL method to control the robot to walk on rough terrain.

### *A. Learning of the target gait*

Q-learning is an off-policy temporal-difference (TD) learning algorithm. In Q-learning the optimal action-value function Q* is directly approximated by the learned action-value function. The update rule of $Q(\lambda)$ algorithm is defined as:

$$Q(s_t,a_t) \leftarrow Q(s_t,a_t) + \beta\left[r_{t+1} + \gamma \max_a Q(s_{t+1},a) - Q(s_t,a_t)\right]\Phi(t) \tag{9}$$

where $\beta$ is the learning rate, $\gamma$ is the discount rate, $\Phi(t)$ is the eligibility trace [8].

In this paper a $Q(\lambda)$ algorithm is used to learn the nominal gait and its corresponding control policy for the robot's flat floor walking. As previously noted, the quasi-PDW robot has 3 DOFs. However, if we choose joint angles and joint velocities to be the state vector as in a common robot control task, the state space will have 6 dimensions which are too huge for the trial-and-error based Q-learning algorithm. To avoid the curse of dimensionality, we propose a novel phase division for the walking process to add prior knowledge into the learning algorithm. The walking course is divided into 3 phases by 3 events. The collision (C) event occurs when the swing leg impact to the ground. The mid-stance (M) event occurs when the hip joint angle of the swinging leg equals that of the supporting leg. The fully-extended (F) event occurs when the swing leg is fully extended and the knee joint is locked. The definition of the events and phases is shown in Fig.3.

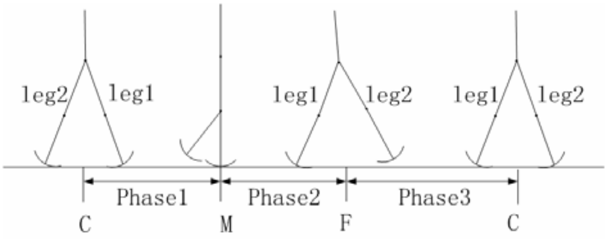

Fig.3 Definition of walking phases

We define the state vector S as $S = [p, K]$, where p is the phase number and K is the kinetic energy of COM. With this definition, the dimension of the state space is reduced. But the drawback of this definition is that the Markov property of the algorithm is broken. However, as the purpose of our method in this stage is to give out a target gait and the corresponding control strategy for the FAL method, this definition is able to tell different gaits apart and the simulation results verify the effectiveness. The action vector A is set as $A = [k_1, k_3, k_4, \varphi_1, \varphi_3, \varphi_4]$, when leg1 is the supporting leg. In our method the stiffness and the equilibrium angle of the actuators are tuned at the phase transforming moment. During one walking phase, the walking process is completely dependent on the passive dynamics of the robot and the actuators. When the walking phase transforms, a failure detection approach is also executed based on a set of thresholds of the joint angles for each phase. With this approach, the undesired posture of the robot leads to failure immediately, which increases the learning speed.

In our algorithm, the reward is set as follows. If the robot falls down or fails to walk, a negative reward of -1 is given. If the robot achieves one step, a positive reward of +1 is given. And we set the final goal of the learning to find a fixed point in the generalized coordinate space. If the generalized coordinates after collision are same as with the generalized coordinates at the beginning of a step, a fixed point is found, and a big positive reward of +100 is given. At the same time, the achieved gait and its control actions are recorded as the target gait and control strategy for the FAL learning control.

### *B. FAL learning control*

As the state-action space of the walking robot is continuous, the most difficult problem for the learning controller is the generalization problem. To introduce generalization into the state presentation, several function approximators have been used, including CMAC, NN and FIS. Among them, FIS is considered to be the best due to the capability of embedding prior knowledge into fuzzy rules, the inherent locality property leading to faster local learning, and the strong generalization ability of fuzzy partition of state spaces [9]. In this paper we use FAL [9] as the learning controller which combines the generalization ability of FIS and the on-line learning ability of advantage learning.

The FIS of the FAL algorithm serves as an advantage function evaluator and a continuous action generator. It is composed of N rules in the following form:

$R_i$: If $s_1$ is $L_1^i$, $s_2$ is $L_2^i$,...,and $s_n$ is $L_n^i$, then

$$\begin{bmatrix} U^i \\ V^i \end{bmatrix} = \begin{bmatrix} p_i \\ w_i \end{bmatrix}. \quad (10)$$

where $S = [s_1,..., s_n]$ is the input variables vector, $L_1^i...L_n^i$ are the membership functions, $U^i$ and $V^i$ are the output of the $i_{th}$ rule. $U^i$ is the local action. $V^i$ is the local advantage value. $p_i, w_i \in R$ are the action value and advantage value which are chosen from the local action set $P_i$ and local advantage parameter set $W_i$ respectively. Each value in $P_i$ is associated with an advantage parameter in $W_i$. The local actions compete with each other to be the selected action of the rule according to their advantage values.

FAL includes two components: the critic evaluating the advantage function and the actor generating the control action. The output of the actor and critic are the weighted sum of the local action values and local advantage values :

$$U(S_t) = \sum_{R_i \in \Omega_t} (\alpha_{R_i} p_i), \quad (11)$$

$$V(S_t, U_t) = \sum_{R_i \in \Omega_t} (\alpha_{R_i} w(p_i(t))), \quad (12)$$

where $S_t$ is the input vector at t moment, $U(S_t)$ is the continuous action, $V(S_t, U_t)$ is the output of the critic which appraises the advantage value for the current state-action pair, $\alpha_{R_i}$ is the truth value of the rule, $\Omega_t$ is the set of activated rules. The structure of FAL algorithm is shown in Fig.4.

To improve the policy of the learning algorithm, the advantage value of the rule $V_t^*(S_t)$ is defined as the advantage value of the local optimal actions. Then, the TD error is defined as follows:

$$e_{t+1} = \Gamma[r_{t+1} + \gamma V_t^*(S_{t+1})] - (\Gamma - 1)V_t^*(S_t) - V_t(S_t, U_t) \quad (13)$$

where r is the reward received at the t+1 moment, $\Gamma$ is the relative scaling factor. Then the update rule of the critic is written as:

$$w_{t+1}(p_i^l) = w_t(p_i^l) + \beta e_{t+1}\Phi_w(t), \forall w_t(p_i^l) \in W_i. \quad (14)$$

With the learned target gait and the corresponding control actions of the flat floor walking, the FAL algorithm is applied to control the robot to walk on uneven floor. Let $\theta^d = [\theta_p^d]$ be the learned gait, and $A^d = [A_p^d]$ be the corresponding actions, where p is the phase number, $\theta_p^d$ and $A_p^d$ are the state vector and action vector for phase p respectively. When the robot is walking in uneven terrain or an environment with disturbances, there will be errors in the state variables. Define the error of phase p as $\theta_e = \theta_p^d - \theta$, where $\theta$ is the state vector. Let $\theta_e$ be the input of FAL, then the output of the system is:

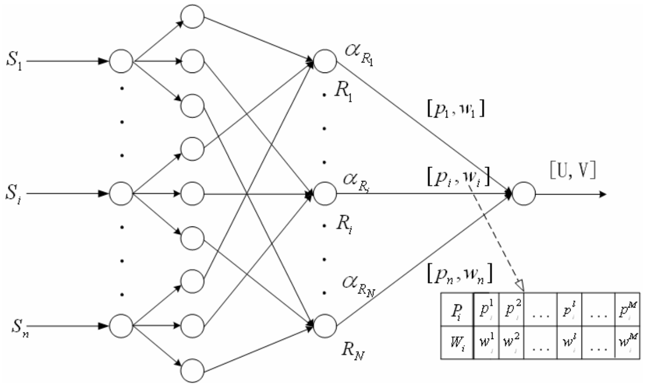

Fig.4 FAL scheme

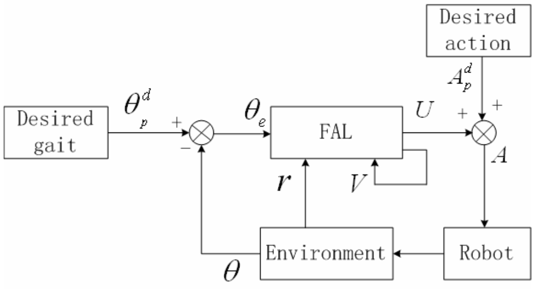

Fig.5 Control System

$$A = A_p^d + U. \quad (15)$$

The structure of the system is shown in Fig.5. To control the robot's walking, we choose the input state vector as $\theta_e = [\theta_e^1, \theta_e^2, \dot{\theta}_e^1, \dot{\theta}_e^2]$, where $\theta_e^1$ is the angle error of the hip joint of the supporting leg, $\theta_e^2$ is the angle error of the hip joint of the swinging leg, $\dot{\theta}_e^1$ and $\dot{\theta}_e^2$ are their velocity errors respectively. At the same time we choose the output action of FAL as $U = [\varphi_1^{act}, k^{act}, \varphi_2^{act}]$, where $\varphi_1^{act}$ is the equilibrium angle action for the hip joint, $k^{act}$ is the stiffness action for the hip joint, $\varphi_2^{act}$ is the equilibrium angle action for the swinging knee joint.

The reward of the learning process is defined as:

$$r = \begin{cases} -1, & \text{fail} \\ (exp^{-E} - 1), & \text{step} \end{cases}, \quad (16)$$

where $E = E_{cur} - E_{last}$ is the difference between the state error in current step and the state error in last step.

In [9], FAL algorithm was successfully applied to control an inverted pendulum system. The local advantage value sets of the algorithm in [9] were initialized with random values. However in our case, the walking robot has more input dimensions and output dimensions than the inverted pendulum system. Randomly initialized values will cause low efficiency of the learning process. Thus, we follow the following approach to initialize the local advantage values. We first construct a data set $T = \{t_1, t_2..t_{N_R}\}$ according to the fuzzy partition of the input variables, where $N_R$ is the number of rules in FIS. We guarantee each $t_i \in T$ is associated with a rule $R_i$, and only activate $R_i$ when is used as input data in FAL. Suppose $t_i$ is the input variables, we choose each action $p_j$ in the local action set of $R_i$ as the output action and simulate the walking process. When the robot encounters a phase transformation, the state error is recorded as $e_{ij}$. Then, the local advantage value of $R_i$ is initialized as:

$$w(p_i^j) = -e_{ij} / \sum_{k=1}^{N_A} e_{ik}, \tag{17}$$

where $N_A$ is the number of actions in the local action set.

## IV. Simulation Results

### A. Learning of the target gait

We first apply the first stage of our method to learn the target gait for the simulated robot. We use the $\varepsilon$ -greedy policy as the exploration policy. The simulation parameters are set as follows $\beta = 0.2$ , $\varepsilon = 0.1$ , $\gamma = 0.95$ , $\lambda = 0.5$ . In our method, the policy improvement only occurs when the phase transforms. So we set the discount rate of the eligibility trace to be a fairly low value to conclude most of the reward to the recent selected actions. After an average of 2,000 trials a stable walking gait is found.

The trajectories of the hip joints of the learned gait are shown in Fig.6. The trajectories of the knee joints are shown in Fig.7. The phase portrait of a 15 steps robot walking is shown in Fig.8. The x-axis and y-axis represent the joint angle and joint velocity of the hip respectively. The figures show a walking episode in which the robot begins with a predefined initial condition, and walks under the control of the learning controller. After several steps the robot's walking is adjusted to the stable walking gait gradually.

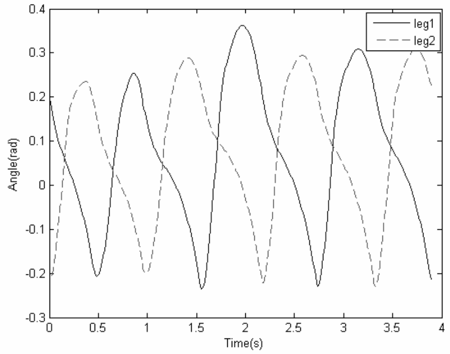

Fig.6 Simulated hip joint angles

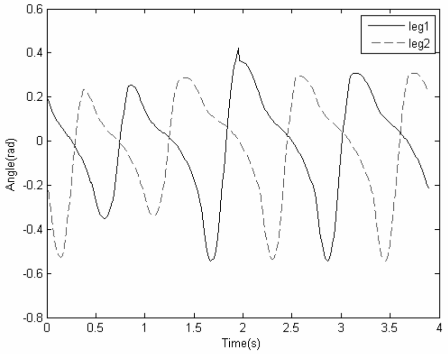

Fig.7 Simulated Knee joint angles

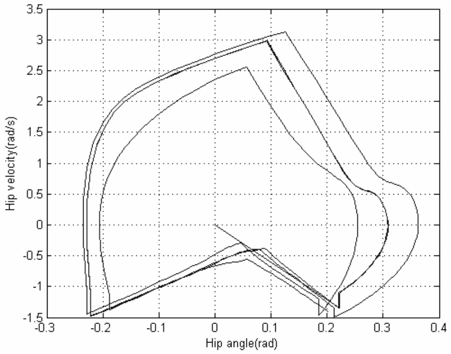

Fig.8 Simulated phase portrait of the hip joints

### B. Walking on uneven floor

Based on the learned target gait, we apply the FAL algorithm to control the robot's walking on uneven floor. As aforementioned, FAL has the capability to add prior knowledge into the fuzzy rules. Before learning, we must define the fuzzy membership function and the local action sets based on the prior task knowledge. In this paper, triangular fuzzy membership functions and a strong fuzzy partition are used. The fuzzy partitions of the input variables are listed in Table.II. The values of the local action sets are defined in Table.III. With these parameters, the local advantage value sets are initialized. Then we control the robot's walking on an uneven floor with the disturbance from -4cm to 2cm. The walking gait is shown in Fig.9. The trajectories of the hip joints are shown in Fig.10.

The accumulated reward received by the learning algorithm is shown in Fig.11. The accumulated reward data is filtered with moving average of 20 trials. The robot learns a walking gait on the uneven floor within 100 trials.

TABLE II
Fuzzy partitions of input variables

| Input | Fuzzy membership function |
|---|---|
| $\theta_e^1(rad)$ | (-0.08,0,0.08) |
| $\theta_e^2(rad)$ | (-0.08,0,0.08) |
| $\dot{\theta}_e^1(rad \cdot s^{-1})$ | (-0.24,0,0.24) |
| $\dot{\theta}_e^2(rad \cdot s^{-1})$ | (-0.24,0,0.24) |

TABLE III
VALUES OF LOCAL ACTION SETS

| Action | Local action sets |
|---|---|
| $\varphi_1^{act}(rad)$ | {-1.2,-0.6,-0.2,0,0.2,0.6,1.2} |
| $\varphi_2^{act}(rad)$ | {-0.4,0,0.4} |
| $k^{act}(N\cdot m\cdot rad^{-1})$ | {-0.8,-0.4,-0.2,0,0.2,0.4,0.8} |

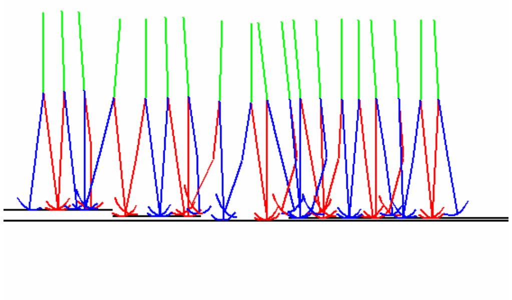

Fig.9 Uneven floor walking

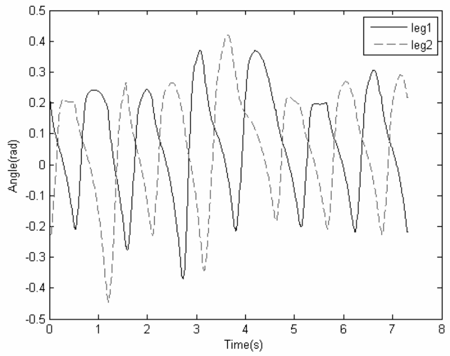

Fig.10 Simulated hip joint angles

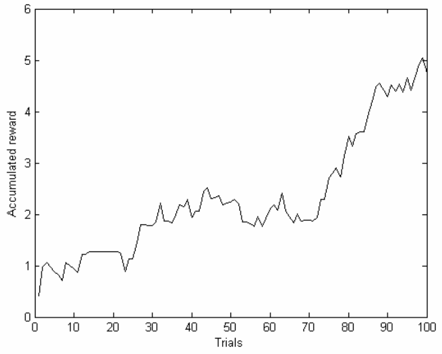

Fig.11 Accumulated reward

## V. CONCLUSIONS

In this paper, a planar quasi-PDW robot is demonstrated and a reinforcement learning based method to control the robot's simulation walking process is presented. We first apply a Q-learning method to learn a nominal gait of the robot's walking on flat floor. Then, a FAL algorithm is used to control its walking on an uneven floor. With our method, the robot quickly adapts to small disturbances in terrain.

Our work in this paper is focused on the control of simulated walking. Our future work will apply this methodology to a physical robot.

## REFERENCES

[1] T.McGeer. "Passive dynamic walking," The International Journal of Robotics Research, 1990, Vol.9, No.2, pp: 62-82.

[2] M.Wisse, A.L.Schwab. "First steps in Passive Dynamic Walking," CLAWAR, 2004.

[3] S.Collins, A.Ruina, R.Tedrake, M.Wisse. "Efficient Bipedal Robots Based on Passive-Dynamic Walkers," Science, 2005, Vol.307, pp: 1082-1085.

[4] E.Schuitema,D.G.E.Hobbelen, P.P.Jonker, M.Wisse, J.G.D.Karssen. "Using a controller based on reinforcement learning for a passive dynamic walking robot," Proceedings of IEEE International Conference on Humanoid Robots, 2005, pp: 232-237

[5] J.Morimoto, J.Nakanishi, G.Endo,C.G. Atkeson et.al. "Poincaré-Map-Based reinforcement learning for biped Walking," Proceedings of IEEE International Conference on Robotics and Automation, 2005, pp: 2381-2386.

[6] S.Collins, A.Ruina. "A Bipedal Walking Robot with Efficient and Human-Like Gait," Proceedings of IEEE International Conference on Robotics and Automation, 2005, pp: 1983-1988.

[7] R.V.Ham, B.Vanderborght, B.Verrelst, et.al. "MACCEPA: the Mechanically Adjustable Compliance and Controllable Equilibrium Position Actuator used in the 'Controlled Passive Walking' biped Veronica," CLAWAR, 2005, pp: 759-766.

[8] R.S.Sutton, A.G.Barto. Reinforcement learning: an introduction. The MIT Press, Cambridge, MA, 1998. ISBN 0-262-19398-1

[9] X.W.Yan, Z.D.Deng, Z.Q.Sun. "Fuzzy advantage learning," Proceedings of IEEE International Conference on Fuzzy Systems, 2000, pp: 865-870.