Proceedings of the 2004 IEEE
International Conference on Robotics & Automation
New Orleans, LA • April 2004

A Simple Reinforcement Learning Algorithm For Biped Walking

Jun Morimoto, Gordon Cheng,
Department of Humanoid Robotics and
Computational Neuroscience
ATR Computational Neuroscience Labs
xmorimo@atr.co.jp, gordon@atr.co.jp
http://www.cns.atr.co.jp/hrcn

Christopher G. Atkeson, and Garth Zeglin
The Robotics Institute
Carnegie Mellon University
cga@cs.cmu.edu, garthz@ri.cmu.edu
http://www.ri.cmu.edu

***Abstract*— We propose a model-based reinforcement learning algorithm for biped walking in which the robot learns to appropriately place the swing leg. This decision is based on a learned model of the Poincare map of the periodic walking pattern. The model maps from a state at the middle of a step and foot placement to a state at next middle of a step. We also modify the desired walking cycle frequency based on online measurements. We present simulation results, and are currently implementing this approach on an actual biped robot.**

I. Introduction

Our long-term goal is to understand how humans learn biped locomotion and adapt their locomotion pattern. In this paper, we propose and explore the feasibility of a candidate learning algorithm for biped walking. Our algorithm has two elements, learning appropriate foot placement, and estimating appropriate walking cycle timing. We are using model-based reinforcement learning, where we learn a model of a Poincare map and then choose control actions based on a computed value function. Alternative approaches applying reinforcement learning to biped locomotion include [1], [13], [2].

An important issue in applying our approach is matching the desired walking cycle timing to the natural dynamics of the biped. In this study, we use phase oscillators to estimate appropriate walking cycle timing [19], [14], [15].

To evaluate our proposed method, we use simulated 3 link and 5 link biped robots (Figs. 1 and 2). Physical parameters of the 3 link simulated robot are in table I. Physical parameters of the 5 link simulated robot in table II are selected to model an actual biped robot fixed to a boom that keeps the robot in the sagittal plane (Fig. 2). Our bipeds have a short torso and point or round feet without ankle joints. For these bipeds, controlling biped walking trajectories with the popular ZMP approach [20], [8], [22], [12] is difficult or not possible, and thus an alternative method for controller design must be used.

In section II-A, we introduce an estimation method of natural biped walking timing by using the measured walking period and an adaptive phase resetting method. In section III, we introduce our reinforcement learning method for biped walking. The robot learns appropriate foot placement through trial and error. In section IV-B, we propose using the estimation method for natural biped walking timing to assist the learned controller. In section IV-C, we analyze the stability of the learned controller.

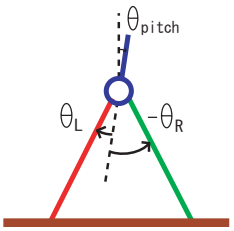

Fig. 1. Three link robot model

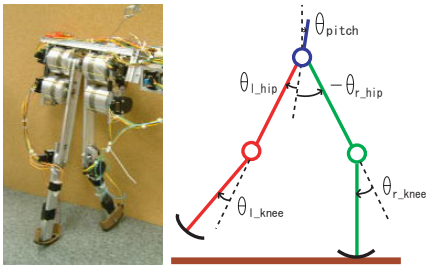

Fig. 2. Five link biped robot

TABLE I
PHYSICAL PARAMETERS OF THE THREE LINK ROBOT MODEL

	trunk	leg
mass $[kg]$	2.0	0.8
length $[m]$	0.01	0.4
inertia $[kg \cdot m^2]$	0.0001	0.01

II. Estimation of Natural Biped Walking Timing

In order for our foot placement algorithm to place the foot at the appropriate time, we must estimate the natural biped walking period, or equivalently, frequency. This timing changes, for example, when walking down slopes. Our goal is to adapt the walking cycle timing to the dynamics of the robot and environment.

TABLE II

PHYSICAL PARAMETERS OF THE FIVE LINK ROBOT MODEL

	trunk	thigh	shin
mass $[kg]$	2.0	0.64	0.15
length $[m]$	0.01	0.2	0.2
inertia ($\times 10^{-4}$ $[kg \cdot m^2]$)	1.0	6.9	1.4

A. Estimation method

We derive the target walking frequency ω^* from the walking period T which is measured from an actual half-cycle period (one foot fall to another):

$$\omega^* = \frac{\pi}{T} \tag{1}$$

The update rule for the walking frequency is

$$\hat{\omega}_{n+1} = \hat{\omega}_n + K_\omega(\omega^* - \hat{\omega}_n), \tag{2}$$

where K_ω is the frequency adaptation gain, and ω_n is the estimated frequency after n steps. An interesting feature of this method is that the simple averaging (low-pass filtering) method (Eq. 2) can estimate appropriate timing of the walking cycle for the given robot dynamics. This method was also adopted in [14], [15].

Several studies suggest that phase resetting is effective to match walking cycle timing to the natural dynamics of the biped [19], [24], [14], [15]. Here we propose an adaptive phase resetting method. Phase ϕ is reset when the swing leg touches the ground:

$$\bar{\phi} \leftarrow \bar{\phi} + K_\phi(\phi - \bar{\phi}) \tag{3}$$

$$\phi \leftarrow \bar{\phi}, \tag{4}$$

where $\bar{\phi}$ is the average phase, and K_ϕ is the phase adaptation gain.

B. A simple example of timing estimation

We use the simulated three link biped robot (Fig. 1) to demonstrate the timing estimation method. A target biped walking trajectory is generated using sinusoidal functions with amplitude $a = 10^\circ$ and a simple controller is designed to follow the target trajectories for each leg:

$$\tau_l = k(a\sin\phi - \theta_l) - b\dot{\theta}_l \tag{5}$$

$$\tau_r = k(-a\sin\phi - \theta_r) - b\dot{\theta}_r, \tag{6}$$

where τ_l denotes the left hip torque, τ_r denotes the right hip torque, $k = 5.0$ is a position gain, $b = 0.1$ is a velocity gain, and θ_l and θ_r are left and right hip joint angles. Estimated phase ϕ is given by $\phi = \hat{\omega}_n t$, where t is the current time.

For comparison, we apply this controller to the simulated robot without using the timing estimation method, so $\hat{\omega}$ is fixed and ϕ increases linearly with time (The walking period was set to $T = 0.63sec$ and frequency $\hat{\omega} = 10rad/sec$). The initial average phase was set to $\bar{\phi} = 1.0$ for the right leg and $\bar{\phi} = \pi + 1.0$ for the left leg, the frequency adaptation gain was set to $K_\omega = 0.3$, and the phase adaptation gain was set to $K_\phi = 0.3$.

With an initial condition which has a body velocity of $0.2m/s$, the simulated 3 link robot walked stably on a 1.0° downward slope (Fig. 3(Top)). However, the robot could not walk stably on a 4.0° downward slope (Fig. 3(Bottom)). When we used the online estimate of $\hat{\omega}$ and the adaptive phase resetting method, the robot walked stably on the two test slopes: 1.0° downward slope (Fig. 4(Top)) and 4.0° downward slope (Fig. 4(Bottom)). In figure 5, we show the estimated walking frequency.

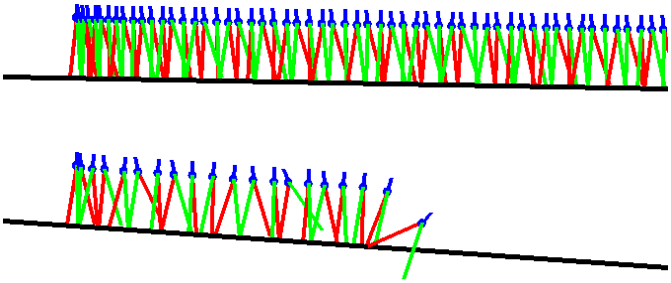

Fig. 3. Biped walking pattern without timing adaptation: (Top) 1.0° downward slope, (Bottom) 4.0° downward slope

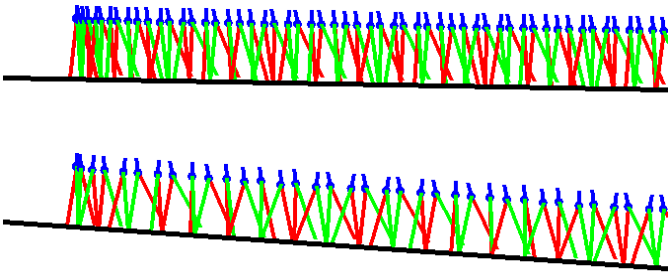

Fig. 4. Biped walking pattern with timing adaptation: (Top) 1.0° downward slope, (Bottom) 4.0° downward slope

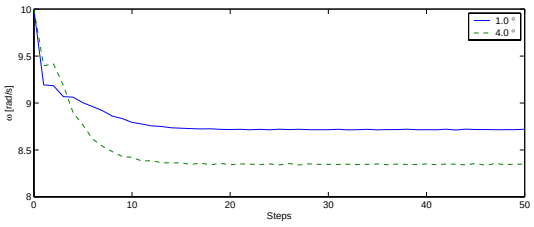

Fig. 5. Estimated walking frequency

III. MODEL-BASED REINFORCEMENT LEARNING FOR BIPED LOCOMOTION

To walk stably we need to control the placement as well as the timing of the next step. Here, we propose a learning method to acquire a stabilizing controller.

A. Model-based reinforcement learning

We use a model-based reinforcement learning framework [4], [17]. Reinforcement learning requires a source of reward. We learn a Poincare map of the effect of foot placement, and then learn a corresponding value function for states at phase $\phi = \frac{1}{2}\pi$ and $\phi = \frac{3}{2}\pi$ (Fig. 6), where we define phase $\phi = 0$ as the right foot touchdown.

1) Learning the Poincare map of biped walking: We learn a model that predicts the state of the biped a half cycle ahead, based on the current state and the foot placement at touch down. We are predicting the location of the system in a Poincare section at phase $\phi = \frac{3\pi}{2}$ based on the system's location in a Poincare section at phase $\phi = \frac{\pi}{2}$. We use the same model to predict the location at phase $\phi = \frac{\pi}{2}$ based on the location at phase $\phi = \frac{3\pi}{2}$ (Fig. 6). Because the state of the robot drastically changes at foot touchdown ($\phi = 0, \pi$), we select the phases $\phi = \frac{\pi}{2}$ and $\phi = \frac{3\pi}{2}$ as Poincare sections. We approximate this Poincare map using a function approximator with a parameter vector $\mathbf{w}^m$,

$$\hat{\mathbf{x}}_{\frac{3\pi}{2}} = \hat{\mathbf{f}}(\mathbf{x}_{\frac{\pi}{2}}, \mathbf{u}_{\frac{\pi}{2}}; \mathbf{w}^m), \tag{7}$$

where the input state is defined as $\mathbf{x} = (d, \dot{d})$. d denotes the horizontal distance between the stance foot position and the body position (Fig. 7). Here, we use the hip position as the body position because the center of mass is almost at the same position as the hip position (Fig. 2). The action of the robot $\mathbf{u} = \theta_{act}$ is the target knee joint angle of the swing leg which determines the foot placement (Fig. 7).

2) Representation of biped walking trajectories and the low-level controller: One cycle of biped walking is represented by four via-points for each joint (Fig. 6). The output of a current policy θ_{act} is used to specify via-points (Table III). We interpolated trajectories between target postures by using the minimum jerk criteria [6], [21] except for pushing off at the stance knee joint. For pushing off at the stance knee, we instantaneously change the desired joint angle to deliver a pushoff to a fixed target to accelerate the motion.

Zero desired velocity and acceleration are specified at each via-point. To follow the generated target trajectories, the torque output at each joint is given by a PD servo controller:

$$\tau_j = k(\theta_j^d(\phi) - \theta_j) - b\dot{\theta}_j, \tag{8}$$

where $\theta_j^d(\phi)$ is the target joint angle for j-th joint ($j = 1 \cdots 4$), position gain k is set to $k = 2.0$ except for the knee joint of the stance leg (we used $k = 8.0$ for the knee joint of the stance leg), and the velocity gain b is set to $b = 0.05$. Table III shows the target postures.

3) Rewards: The robot gets a reward if it successfully continues walking and gets punishment (negative reward) if it falls down. On each transition from phase $\phi = \frac{1}{2}\pi$ (or $\phi = \frac{3}{2}\pi$) to phase $\phi = \frac{3}{2}\pi$ (or $\phi = \frac{1}{2}\pi$), the robot gets a reward of 0.1 if the height of the body remains above 0.35m during the past half cycle. If the height of the body goes below 0.35m, the robot is given a negative reward (-1) and the trial is terminated.

TABLE III

TARGET POSTURES AT EACH PHASE ϕ : θ_{act} IS PROVIDED BY THE OUTPUT OF CURRENT POLICY. THE UNIT FOR NUMBERS IN THIS TABLE IS DEGREES

	right hip	right knee	left hip	left knee
$\phi = 0$	-10.0	θ_{act}	10.0	0.0
$\phi = 0.5\pi$		θ_{act}		60.0
$\phi = 0.7\pi$	10.0		-10.0	
$\phi = \pi$	10.0	0.0	-10.0	θ_{act}
$\phi = 1.5\pi$		60.0		θ_{act}
$\phi = 1.7\pi$	-10.0		10.0	

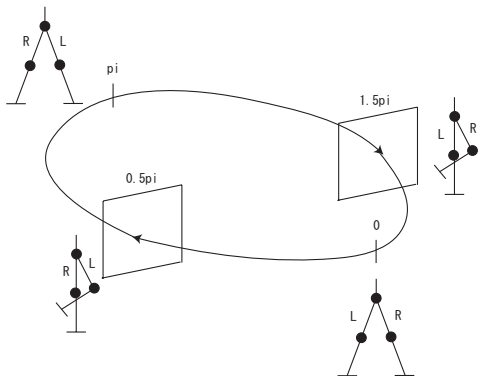

Fig. 6. Biped walking trajectory using four via-points: we update parameters and select actions at Poincare sections on phase $\phi = \frac{\pi}{2}$ and $\phi = \frac{3\pi}{2}$. L:left leg, R:right leg

4) Learning the value function: In a reinforcement learning framework, the learner tries to create a controller which maximizes expected total return. Here, we define the value function for the policy μ

$$V^{\mu}(\mathbf{x}(t)) = E[r(t+1) + \gamma r(t+2) + \gamma^2 r(t+3) + ...], \tag{9}$$

where $r(t)$ is the reward at time t, and γ ($0 \leq \gamma \leq 1$) is the discount factor[1]. In this framework, we evaluate the value function only at $\phi(t) = \frac{\pi}{2}$ and $\phi(t) = \frac{3}{2}\pi$. Thus, we consider our learning framework as model-based reinforcement learning for a semi-Markov decision process (SMDP) [18]. We used a function approximator with a parameter vector $\mathbf{w}^v$ to estimate the value function:

$$\hat{V}(t) = \hat{V}(\mathbf{x}(t); \mathbf{w}^v). \tag{10}$$

By considering the deviation from equation (9), we can define the temporal difference error (TD-error) [17], [18]:

$$\delta(t) = \sum_{k=t+1}^{t_T} \gamma^{k-t-1} r(k) + \gamma^{t_T - t} \hat{V}(t_T) - \hat{V}(t), \tag{11}$$

where t_T is the time when $\phi(t_T) = \frac{1}{2}\pi$ or $\phi(t_T) = \frac{3}{2}\pi$. The update rule for the value function can be derived as

$$\hat{V}(\mathbf{x}(t)) \leftarrow \hat{V}(\mathbf{x}(t)) + \beta\delta(t), \tag{12}$$

where $\beta = 0.2$ is a learning rate. The parameter vector $\mathbf{w}^v$ is updated by equation (19).

[1]We followed the definition of the value function in [17]

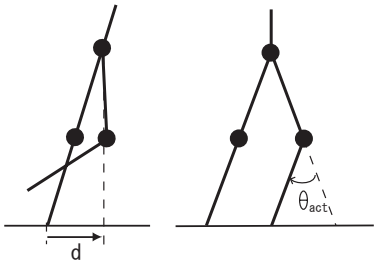

Fig. 7. (left) Input state, (right) Output of the controller

5) Learning a policy for biped locomotion: We use a stochastic policy to generate exploratory action. The policy is represented by a probabilistic model:

$$\mu(\mathbf{u}(t)|\mathbf{x}(t)) = \frac{1}{\sqrt{2\pi}\sigma}\exp\left(-\frac{(\mathbf{u}(t)-\mathbf{A}(\mathbf{x}(t);\mathbf{w}^a))^2}{2\sigma^2}\right), \quad (13)$$

where $\mathbf{A}(\mathbf{x}(t);\mathbf{w}^a)$ denotes the mean of the model, which is represented by a function approximator, where $\mathbf{w}^a$ is a parameter vector. We changed the variance σ according to the trial as $\sigma = \frac{100-N_{trial}}{100} + 0.1$ for $N_{trial} \leq 100$ and $\sigma = 0.1$ for $N_{trial} > 100$, where N_{trial} denotes the number of trials. The output of the policy is

$$\mathbf{u}(t) = \mathbf{A}(\mathbf{x}(t);\mathbf{w}^a) + \sigma\mathbf{n}(t), \quad (14)$$

where $\mathbf{n}(t) \sim N(0,1)$. $N(0,1)$ indicate a normal distribution which has mean 0 and variance 1.

We derive the update rule for a policy by using the value function and the estimated Poincare map.

1) Derive the gradient of the value function $\frac{\partial V}{\partial \mathbf{x}}$ at the current state $\mathbf{x}(t_T)$.
2) Derive the gradient of the dynamics model $\frac{\partial \mathbf{f}}{\partial \mathbf{u}}$ at the previous state $\mathbf{x}(t)$ and the nominal action $\mathbf{u} = \mathbf{A}(\mathbf{x}(t);\mathbf{w}^a)$.
3) Update the policy μ:

$$\mathbf{A}(\mathbf{x};\mathbf{w}^a) \leftarrow \mathbf{A}(\mathbf{x};\mathbf{w}^a) + \alpha\frac{\partial V(\mathbf{x})}{\partial \mathbf{x}}\frac{\partial \mathbf{f}(\mathbf{x},\mathbf{u})}{\partial \mathbf{u}}, \quad (15)$$

where $\alpha = 0.2$ is the learning rate. The parameter vector $\mathbf{w}^a$ is updated by equation (19). We can consider the output $\mathbf{u}(t)$ is an *option* in the SMDP [18] initiated in state $\mathbf{x}(t)$ at time t when $\phi(t) = \frac{1}{2}\pi$ (or $\phi = \frac{3}{2}\pi$), and it terminates at time t_T when $\phi = \frac{3}{2}\pi$ (or $\phi = \frac{1}{2}\pi$).

6) Function approximator: We used Receptive Field Weighted Regression(RFWR) [16] as the function approximator for the policy, the value function and the estimated dynamics model. Here, we approximate the target function $g(\mathbf{x})$ with

$$\hat{g}(\mathbf{x}) = \frac{\sum_{k=1}^{N_b} a_k(\mathbf{x})h_k(\mathbf{x})}{\sum_{k=1}^{N_b} a_k(\mathbf{x})}, \quad (16)$$

$$h_k(\mathbf{x}) = \mathbf{w}_k^T\tilde{\mathbf{x}}_k, \quad (17)$$

$$a_k(\mathbf{x}) = \exp\left(-\frac{1}{2}(\mathbf{x}-\mathbf{c}_k)^T\mathbf{D}_k(\mathbf{x}-\mathbf{c}_k)\right), \quad (18)$$

where $\mathbf{c}_k$ is the center of the k-th basis function, $\mathbf{D}_k$ is the distance metric of the k-th basis function, N_b is the number of basis functions, and $\tilde{\mathbf{x}}_k = ((\mathbf{x}-\mathbf{c}_k)^T, 1)^T$ is the augmented state. The update rule for the parameter $\mathbf{w}$ is given by:

$$\Delta\mathbf{w}_k = a_k\mathbf{P}_k\tilde{\mathbf{x}}_k(g(\mathbf{x}) - h_k(\mathbf{x})), \quad (19)$$

where

$$\mathbf{P}_k \leftarrow \frac{1}{\lambda}\left(\mathbf{P}_k - \frac{\mathbf{P}_k\tilde{\mathbf{x}}_k\tilde{\mathbf{x}}_k^T\mathbf{P}_k}{\frac{a_k}{\lambda} + \tilde{\mathbf{x}}_k^T\mathbf{P}_k\tilde{\mathbf{x}}_k}\right), \quad (20)$$

and $\lambda = 0.999$ is the forgetting factor.

We align basis functions $a_k(\mathbf{x})$ at even intervals in each dimension of input space $\mathbf{x} = (d, \dot{d})$ (Fig. 7) $[-0.2(m) \leq d \leq 0.2(m)$ and $-1.0(m/s) \leq \dot{d} \leq 1.0(m/s)]$. We used 400(=20×20) basis functions for approximating the policy and the value function. We also align 20 basis functions at even intervals in the output space $-0.7(rad) \leq \theta_{act} \leq 0.7(rad)$ (Fig. 7). We used 8000(=20 × 20 × 20) basis functions for approximating the Poincare map. We set the distance metric $\mathbf{D}_k$ to $\mathbf{D}_k = diag\{2256, 90\}$ for the policy and the value function, and $\mathbf{D}_k = diag\{2256, 90, 185\}$ for the Poincare map. The centers of the basis functions $\mathbf{c}_k$ and the distance metrics of the basis functions $\mathbf{D}_k$ are fixed during learning.

IV. Results

A. Learning foot placement

We applied the proposed method to the 5 link simulated robot (Fig. 2). We used a manually generated initial step to get the pattern started. We set the walking period to $T = 0.79sec$ ($\omega = 8.0[rad/sec]$).

A trial terminated after 50 steps or after the robot fell down. Figure 8(Top) shows the walking pattern before learning, and Figure 8(Middle) shows the walking pattern after 30 trials. Target knee joint angles for the swing leg were varied because of exploratory behavior (Fig. 8(Middle)).

Figure 10 shows the accumulated reward at each trial. We defined a successful trial when the robot achieved 50 steps. A stable biped walking controller was acquired after 80 trials (averaged over 5 experiments). The shape of the value function is shown in Fig. 11. The maximum value of the value function is located at positive d (around $d = 0.05(m)$) and negative $\dot{d}$ (around $\dot{d} = -0.5(m/sec)$).

Figure 9 shows joint angle trajectories of stable biped walking after learning. Note that the robot added energy to its initially slow walk by choosing θ_{act} appropriately which affects both foot placement and the subsequent pushoff. The acquired walking pattern is shown in Fig. 8(Bottom).

B. Estimation of biped walking period

The estimated phase of the cycle ϕ plays an important role in this controller. It is essential that the controller phase matches the dynamics of the mechanical system. We applied our timing estimation method described in section II-A to the learned biped controller. The initial average phase was set to $\bar{\phi} = 1.0$ for the left leg and $\bar{\phi} = \pi + 1.0$ for the right leg, the frequency adaptation gain was set to $K_\omega = 0.3$, and the phase adaptation gain was set to $K_\phi = 0.3$.

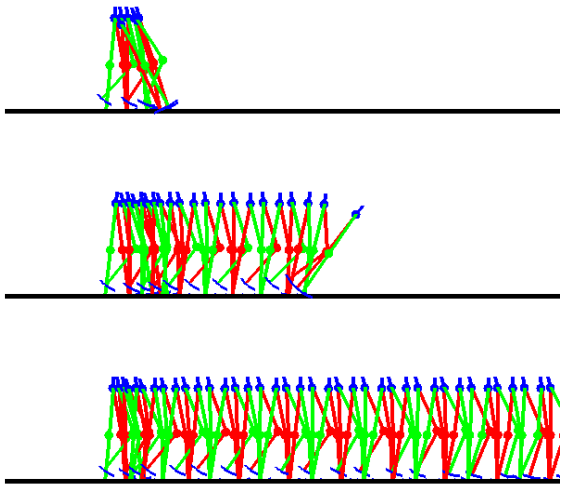

Fig. 8. Acquired biped walking pattern: (Top)Before learning, (Middle)After 30 trials, (Bottom)After learning

We evaluated the combined method on a 1.0° downward slope. The simulated robot with the acquired controller in previous section could not walk stably on the downward slope (Fig. 12(Top)). However, when we used the online estimate of the walking period and the adaptive phase resetting method with the learned controller, the robot walked stably on the 1.0° downward slope (Fig. 12(Bottom)).

C. Stability analysis of the acquired policy

We analyzed the stability of the acquired policy in terms of the Poincare map, mapping from a Poincare section at phase $\phi = \frac{\pi}{2}$ to phase $\phi = \frac{3\pi}{2}$ (Fig. 6). We estimated the Jacobian matrix of the Poincare map at the Poincare sections, and checked if $|\lambda_i(J)| < 1$ or not, where λ_i $(i = 1, 2)$ are the eigenvalues[3], [7]. Because we used differentiable functions as function approximators, we can estimate the Jacobian matrix J based on:

$$J = \frac{d\mathbf{f}}{d\mathbf{x}} = \frac{\partial \mathbf{f}}{\partial \mathbf{x}} + \frac{\partial \mathbf{f}}{\partial \mathbf{u}} \frac{\partial \mathbf{u}(\mathbf{x})}{\partial \mathbf{x}}. \quad (21)$$

Figure 13 shows the average eigenvalues at each trial. The eigenvalues decreased as the learning proceeded, and became stable, i.e. $|\lambda_i(J)| < 1$.

V. Discussion

In this study, we used swing leg knee angle θ_{act} to decide foot placement because the lower leg has smaller mass and tracking the target joint angle at the knee is easier than using the hip joint. However, using hip joints or using different variables for the output of the policy are interesting topics for future work. We also are considering using captured data of a human walking pattern [23] as a nominal trajectory instead of using a hand-designed walking pattern. We are currently applying the proposed approach to the physical biped robot.

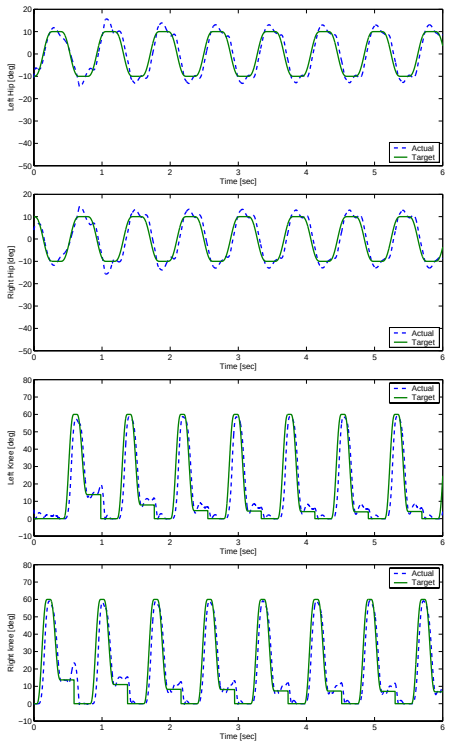

Fig. 9. Joint angle trajectories after learning

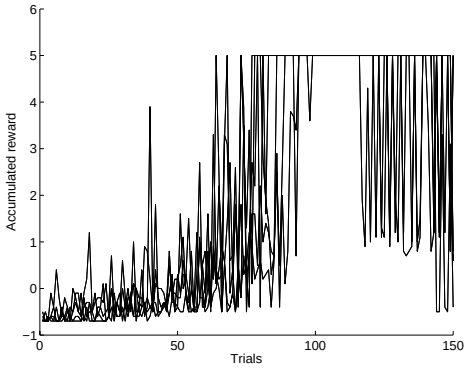

Fig. 10. Accumulated reward at each trial: Results of five experiments

In previous work, we have proposed a trajectory optimization method for biped locomotion [10], [11] based on differential dynamic programming [5], [9]. We are considering combining this trajectory optimization method with the proposed reinforcement learning method.

Acknowledgment

We would like to thank Mitsuo Kawato, Jun Nakanishi, Gen Endo at ATR Computational Neuroscience Laboratories, Japan, and Seiichi Miyakoshi of the Digital Human Research Center, AIST, Japan for helpful discussions.

Atkeson is partially supported by NSF award ECS-0325383.

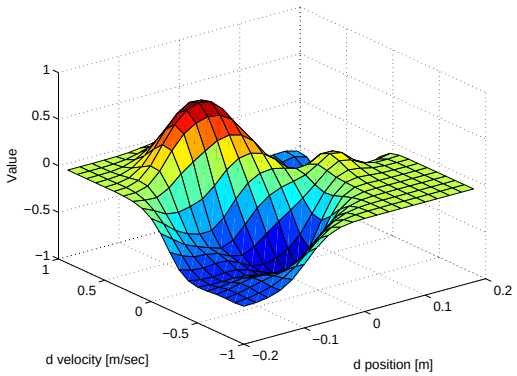

Fig. 11. Shape of acquired value function

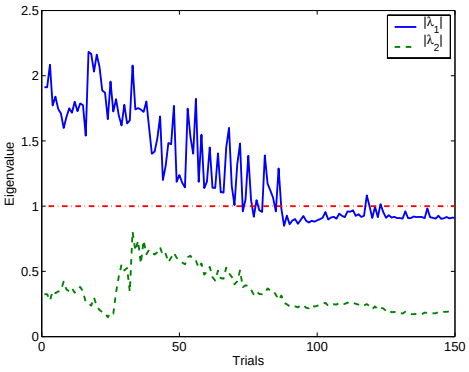

Fig. 13. Averaged eigenvalue of Jacobian matrix at each trial

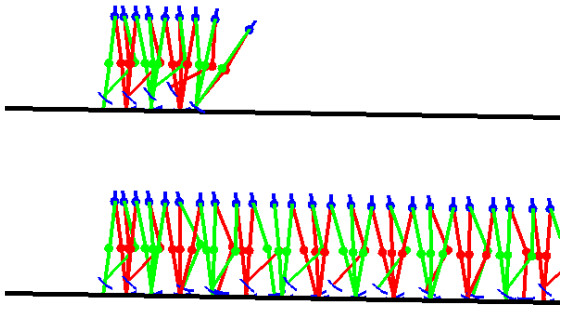

Fig. 12. Biped walking pattern with timing adaptation on downward slope: (Top)Without timing adaptation, (Bottom)With timing adaptation

REFERENCES

[1] H. Benbrahim and J. Franklin. Biped dynamic walking using reinforcement learning. *Robotics and Autonomous Systems*, 22:283–302, 1997.

[2] C. Chew and G. A. Pratt. Dynamic bipedal walking assisted by learning. *Robotica*, 20:477–491, 2002.

[3] R. Q. Van der Linde. Passive bipedal walking with phasic muscle contraction. *Biological Cybernetics*, 82:227–237, 1999.

[4] K. Doya. Reinforcement Learning in Continuous Time and Space. *Neural Computation*, 12(1):219–245, 2000.

[5] P. Dyer and S. R. McReynolds. *The Computation and Theory of Optimal Control*. Academic Press, New York, NY, 1970.

[6] T. Flash and N. Hogan. The coordination of arm movements: An experimentally confirmed mathematical model. *The Journal of Neuroscience*, 5:1688–1703, 1985.

[7] M. Garcia, A. Chatterjee, A. Ruina, and M. J. Coleman. The simplest walking model: stability, complexityu, and scaling. *ASME Jounal of Biomechanical Engineering*, 120(2):281–288, 1998.

[8] K. Hirai, M. Hirose, and T. Takenaka. The Development of Honda Humanoid Robot. In *Proceedings of the 1998 IEEE International Conference on Robotics and Automation*, pages 160–165, 1998.

[9] D. H. Jacobson and D. Q. Mayne. *Differential Dynamic Programming*. Elsevier, New York, NY, 1970.

[10] J. Morimoto and C. G. Atkeson. Robust low-torque biped walking using differential dynamic programming with a minimax criterion. In Philippe Bidaud and Faiz Ben Amar, editors, *Proceedings of the 5th International Conference on Climbing and Walking Robots*, pages 453–459. Professional Engineering Publishing, Bury St Edmunds and London, UK, 2002.

[11] J. Morimoto and C. G. Atkeson. Minimax differential dynamic programming: An application to robust biped walking. In Suzanna Becker, Sebastian Thrun, and Klaus Obermayer, editors, *Advances in Neural Information Processing Systems 15*, pages 1563–1570. MIT Press, Cambridge, MA, 2003.

[12] K. Nagasaka, M. Inaba, and H. Inoue. Stabilization of dynamic walk on a humanoid using torso position compliance control. In *Proceedings of 17th Annual Conference on Robotics Society of Japan*, pages 1193–1194, 1999.

[13] Y. Nakamura, M. Sato, and S. Ishii. Reinforcement learning for biped robot. In *Proceedings of the 2nd International Symposium on Adaptive Motion of Animals and Machines*, pages ThP–II–5, 2003.

[14] J. Nakanishi, J. Morimoto, G. Endo, G. Cheng, S. Schaal, and M. Kawato. Learning from demonstration and adaptation of biped locomotion with dynamical movement primitives. In *Workshop on Robot Programming by Demonstration, IEEE/RSJ International Conference on Intelligent Robots and Systems*, Las Vegas, NV, USA, 2003.

[15] J. Nakanishi, J. Morimoto, G. Endo, G. Cheng, S. Schaal, and M. Kawato. Learning from demonstration and adaptation of biped locomotion. *Robotics and Autonomous Systems (to appear)*, 2004.

[16] S. Schaal and C. G. Atkeson. Constructive incremental learning from only local information. *Neural Computation*, 10(8):2047–2084, 1998.

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

[18] R. S. Sutton, D. Precup, and S. Singh. Between MDPs and semi-MDPs: A Framework for Temporal Abstraction in Reinforcement Learning. *Artificial Intelligence*, 112:181–211, 1999.

[19] K. Tsuchiya, S. Aoi, and K. Tsujita. Locomotion control of a biped locomotion robot using nonlinear oscillators. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1745–1750, Las Vegas, NV, USA, 2003.

[20] J. Vucobratovic, B. Borovac, D. Surla, and D. Stokic. *Biped Locomotion: Dynamics, Stability, Control and Applications*. Springer-Verlag, Berlin, 1990.

[21] Y. Wada and M. Kawato. A theory for cursive handwriting based on the minimization principle. *Biological Cybernetics*, 73:3–15, 1995.

[22] J. Yamaguchi, A. Takanishi, and I. Kato. Development of a biped walking robot compensating for three-axis moment by trunk motion. *Journal of the Robotics Society of Japan*, 11(4):581–586, 1993.

[23] K. Yamane and Y. Nakamura. Dynamics filter – concept and implementation of on-line motion generator for human figures. In *Proceedings of the 2000 IEEE International Conference on Robotics and Automation*, pages 688–693, 2000.

[24] T. Yamasaki, T. Nomura, and S. Sato. Possible functional roles of phase resetting during walking. *Biological Cybernetics*, 88(6):468–496, 2003.