ACQUISITION OF A BIPED WALKING PATTERN USING A POINCARE MAP

Jun Morimoto, Jun Nakanishi, Gen Endo, Gordon Cheng
ICORP Computational Brain Project, JST
ATR Computational Neuroscience Labs
2-2-2 Hikaridai Soraku-gun Seika-cho Kyoto, 610-0288, JAPAN
xmorimo@atr.jp

Christopher G. Atkeson, Garth Zeglin
Carnegie Mellon University
5000 Forbes Ave, Pittsburgh, PA USA

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 a single support phase and foot placement to a state at the next single support phase. We applied this approach to both a simulated robot model and an actual biped robot. We show that successful walking patterns are acquired.

Keywords: Biped Walking; Reinforcement Learning; Poincare map

1. Introduction

In this paper, we propose a learning algorithm to generate appropriate foot placement for biped walking. 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].

First, we use a simulated 5 link biped robot (Fig. 1) to evaluate our proposed method. Physical parameters of the 5 link simulated robot in table 1 are selected to model an actual biped robot fixed to a boom that keeps the robot in the sagittal plane (Fig. 1). We also show that the proposed method can be applied to the actual robot. 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 2, we introduce our reinforcement learning method for biped walking. In section 3, we show simulation results. In section 4, we present an implementation of the proposed method on the real robot, we show that the robot acquire a successful walking pattern within 100 trials.

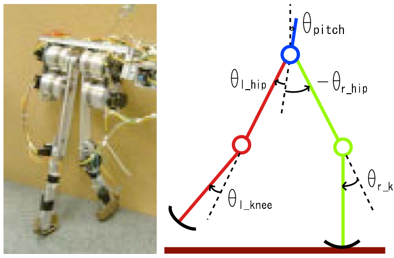

Fig. 1. Five link biped robot

Table 1. 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

2. Model-based reinforcement learning for biped locomotion

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

The input state is defined as $\mathbf{x} = (d, \dot{d})$. where d denotes the horizontal distance between the stance foot position and the body position (Fig. 3). 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. 1). 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. 3).

2.1. *Function approximator*

We use Receptive Field Weighted Regression(RFWR) [16] as the function approximator for the policy, the value function and the estimated Poincare map. We approximate a target function $g(\mathbf{x})$ with

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

$$h_k(\mathbf{x}) = \mathbf{w}_k^T \tilde{\mathbf{x}}_k, \tag{2}$$

$$a_k(\mathbf{x}) = \exp\left(-\frac{1}{2} (\mathbf{x} - \mathbf{c}_k)^T \mathbf{D}_k (\mathbf{x} - \mathbf{c}_k) \right), \tag{3}$$

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 $\tilde{\mathbf{x}}_k = ((\mathbf{x} - \mathbf{c}_k)^T, 1)^T$

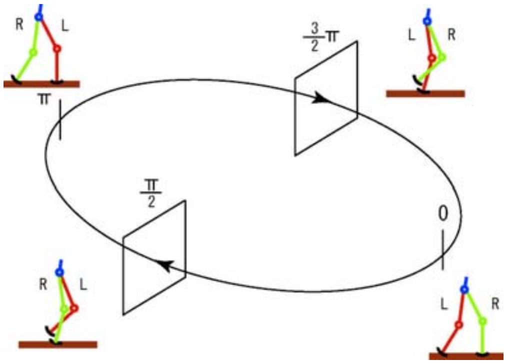

Fig. 2. Biped walking trajectory using four via-points: we update parameters and select actions at Poincare sections at phase $\phi = \frac{\pi}{2}$ and $\phi = \frac{3\pi}{2}$. L:left leg, R:right leg

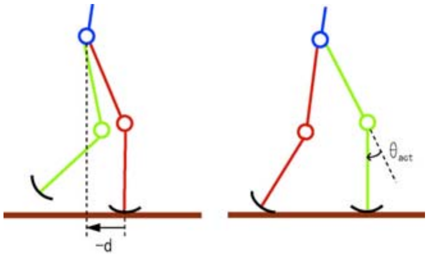

Fig. 3. (left) Input state, (right) Output of the controller

is the augmented state. The update rule for the parameter $\mathbf{w}$ is given by:

$$\Delta \mathbf{w}_k = a_k \mathbf{P}_k \tilde{\mathbf{x}}_k (g(\mathbf{x}) - h_k(\mathbf{x})), \tag{4}$$

where

$$\mathbf{P}_k \leftarrow \frac{1}{\lambda}\left(\mathbf{P}_k - \frac{\mathbf{P}_k \tilde{\mathbf{x}}_k \tilde{\mathbf{x}}_k^T \mathbf{P}_k}{\frac{a_k}{\lambda} + \tilde{\mathbf{x}}_k^T \mathbf{P}_k \tilde{\mathbf{x}}_k}\right), \tag{5}$$

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

In this study, we allocate a new basis function if the activation of all existing units is smaller than a threshold a_{min}, i.e.,

$$\max_k a_k(\mathbf{x}) < a_{\min}, \tag{6}$$

where $a_{\min} = \exp(-\frac{1}{2})$. We initially align basis functions $a_k(\mathbf{x})$ at even intervals in each dimension of input space $\mathbf{x} = (d, \dot{d})$ (Fig. 3) $[-0.2(m) \leq d \leq 0.2(m)$ and $-1.0(m/s) \leq \dot{d} \leq 1.0(m/s)]$. Initial numbers of basis functions are

$400 (=20 \times 20)$ for approximating the policy and the value function. We put 1 basis function at origin for approximating the Poincare map. We set the distance metric $\mathbf{D}_k$ to $\mathbf{D}_k = diag\{2500, 90\}$ for the policy and the value function, and $\mathbf{D}_k = diag\{2500, 225, 1600\}$ 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.

2.2. *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 touchdown. 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 $\phi = \frac{\pi}{2}$(Fig. 2). We use a different model to predict the location at phase $\phi = \frac{\pi}{2}$ based on the location at phase $\phi = \frac{3\pi}{2}$ because the real robot has asymmetric property mainly caused by boom.

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 $\mathbf{w}^m$,

$$\hat{\mathbf{x}}_{\frac{3\pi}{2}} = \hat{\mathbf{f}}_1(\mathbf{x}_{\frac{\pi}{2}}, \mathbf{u}_{\frac{\pi}{2}}; \mathbf{w}_1^m), \tag{7}$$

$$\hat{\mathbf{x}}_{\frac{\pi}{2}} = \hat{\mathbf{f}}_2(\mathbf{x}_{\frac{3\pi}{2}}, \mathbf{u}_{\frac{3\pi}{2}}; \mathbf{w}_2^m), \tag{8}$$

where the input state is defined as $\mathbf{x} = (d, \dot{d})$, and the action of the robot is defined as $\mathbf{u} = \theta_{act}$ (Fig. 3).

2.3. *Representation of biped walking trajectories and the low-level controller*

One cycle of biped walking is represented by four via-points for each joint. The output of a current policy θ_{act} is used to specify via-points (Table 2). 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:

$$\tau_j = k(\theta_j^d(\phi) - \theta_j) - b\dot{\theta}_j, \tag{9}$$

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 use $k = 8.0$ for the knee joint of the stance leg), and the velocity gain b is set to $b = 0.05$. Table 2 shows the target postures.

We reset the phase [19,14] to $\phi = \phi_{reset}$ at right foot touchdown and to $\phi = \pi + \phi_{reset}$ at left foot touchdown, where $\phi_{reset} = 0.3$ rad is empirically determined.

Table 2. Target postures at each phase ϕ : θ_{act} is provided by the output of current policy. The units for numbers in this table are 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	

2.4. *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 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.

2.5. *Learning the value function*

In a reinforcement learning framework, the learner tries to create a controller which maximizes expected total return. We define the value function for the policy μ

$$V^{\mu}(\mathbf{x}(t)) = E[r(t+1) + \gamma r(t+2) + \gamma^2 r(t+3) + ...], \tag{10}$$

where $r(t)$ is the reward at time t, and γ $(0 \leq \gamma \leq 1)$ is the discount factor[a]. 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 use a function approximator with a parameter vector $\mathbf{w}^v$ to represent the value function:

$$\hat{V}(t) = \hat{V}(\mathbf{x}(t); \mathbf{w}^v). \tag{11}$$

By considering the deviation from equation (10), we can define the temporal difference error (TD-error) [17,18]:

$$\delta(t) = \sum_{k=t+1}^{t_T} \gamma^{k-t-1} r(k) + \gamma^{t_T - t} \hat{V}(t_T) - \hat{V}(t), \tag{12}$$

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

$$\hat{V}(\mathbf{x}(t)) \leftarrow \hat{V}(\mathbf{x}(t)) + \beta\delta(t), \tag{13}$$

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

where $\beta = 0.2$ is a learning rate. The parameter vector $\mathbf{w}^v$ is updated by equation (4).

2.6. *Learning a policy for biped locomotion*

We use a stochastic policy to generate exploratory action. The policy is represented by a probabilistic model:

$$\mu(\mathbf{u}(t)|\mathbf{x}(t)) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{(\mathbf{u}(t) - \mathbf{A}(\mathbf{x}(t); \mathbf{w}^a))^2}{2\sigma^2}\right), \tag{14}$$

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 $\sigma = 0.2\left(\frac{150-N_{trial}}{150}\right) + 0.01$ for $N_{trial} \leq 150$ and $\sigma = 0.01$ for $N_{trial} > 150$, where N_{trial} denotes the number of trials. The output of the policy is

$$\mathbf{u}(t) = \mathbf{A}(\mathbf{x}(t); \mathbf{w}^a) + \sigma\mathbf{n}(t), \tag{15}$$

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

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

(1) Predict the next state $\hat{\mathbf{x}}(t_T)$ from the current state $\mathbf{x}(t)$ and the nominal action $\mathbf{u} = \mathbf{A}(\mathbf{x}(t); \mathbf{w}^a)$ using the Poincare map model $\hat{\mathbf{x}}(t_T) = \hat{\mathbf{f}}(\mathbf{x}(t), \mathbf{u}(t); \mathbf{w}^m)$.
(2) Derive the gradient of the value function $\frac{\partial V}{\partial \mathbf{x}}$ at the predicted state $\hat{\mathbf{x}}(t_T)$.
(3) Derive the gradient of the dynamics model $\frac{\partial \mathbf{f}}{\partial \mathbf{u}}$ at the current state $\mathbf{x}(t)$ and the nominal action $\mathbf{u} = \mathbf{A}(\mathbf{x}(t); \mathbf{w}^a)$.
(4) Update the policy μ:

$$\mathbf{A}(\mathbf{x}; \mathbf{w}^a) \leftarrow \mathbf{A}(\mathbf{x}; \mathbf{w}^a) + \alpha \frac{\partial V(\mathbf{x})}{\partial \mathbf{x}} \frac{\partial \mathbf{f}(\mathbf{x}, \mathbf{u})}{\partial \mathbf{u}}, \tag{16}$$

where $\alpha = 0.2$ is the learning rate. The parameter vector $\mathbf{w}^a$ is updated by equation (4). 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 $\phi = \frac{1}{2}\pi$).

3. Simulation results

We applied the proposed method to the 5 link simulated robot (Fig. 1). We use a manually generated initial step to get the pattern started. We set the walking period to $T = 0.79$ *sec* ($\omega = 8.0$ *rad/sec*). A trial terminated after 30 steps or after the robot fell down. Figure 4(Top) shows the walking pattern before learning.

Figure 5 shows the accumulated reward at each trial. We defined a successful trial when the robot achieved 30 steps. A stable biped walking controller was acquired within 200 trials (Fig. 5). The shape of the value function is shown in Figure 6. The minimum value of the value function is located at negative body position d

and negative body velocity $\dot{d}$ because this state leads the robot to fall backward. The maximum value of the value function is located at negative body position d and positive body velocity $\dot{d}$ which leads to a successful walk.

Figure 7 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 Figure 4(Bottom).

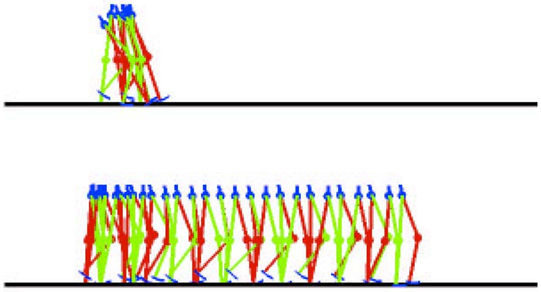

Fig. 4. Acquired biped walking pattern: (Top)Before learning, (Bottom)After learning

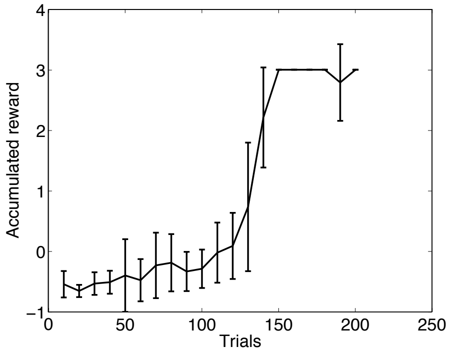

Fig. 5. Accumulated reward at each trial: Results of 10 experiments

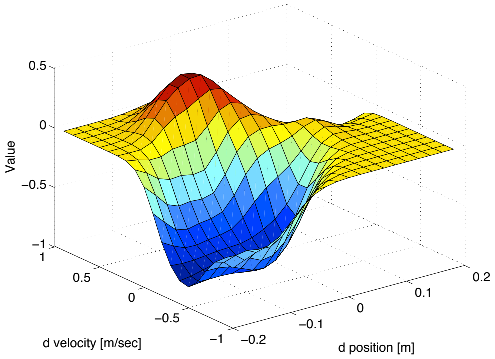

Fig. 6. Shape of acquired value function

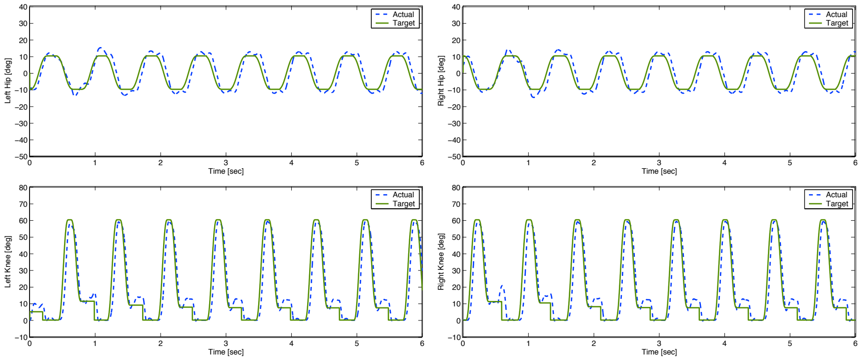

Fig. 7. Joint angle trajectories after learning

4. Real robot implementation

We applied the proposed model-based reinforcement learning scheme to our real biped (Fig. 1). We use a walking pattern generated by a pre-designed state machine controller [15] as the nominal walking pattern. We detect via-points from this nominal walking pattern and manually select via-points which correspond to foot placement (Fig 8). In this framework, control output θ_{act} modulate the selected via-points θ_i^v:

$$\theta_i^v = \bar{\theta}_i^v + \theta_{act} \quad (i = 1, \cdots, n^v), \tag{17}$$

where n^v denotes number of selected via-points, and $\bar{\theta}_i^v$ denotes nominal value of the selected via-points. Each selected via-point is equally modulated by the control output θ_{act}.

We changed the variance σ in equation (14) according to the trial as $\sigma = 0.1\left(\frac{50-N_{trial}}{50}\right) + 0.01$ for $N_{trial} \leq 50$ and $\sigma = 0.01$ for $N_{trial} > 50$, where

N_{trial} denotes the number of trials. We set the walking period to $T = 0.84$ *sec* ($\omega = 7.5$ *rad/sec*). A trial terminated after 30 steps or after the robot fell down. We use the pre-designed state machine for the initial 6 steps.

We also used a phase resetting method for the real robot experiment. We reset the phase to $\phi = \phi_{reset}$ at right foot touchdown and to $\phi = \pi + \phi_{reset}$ at left foot touchdown, where $\phi_{reset} = 0.3$ rad.

Figure 9 shows a biped walking pattern before learning. The robot fell over with the nominal walking pattern. Figure 10 shows a biped walking pattern after learning. After 100 trials in the real environment, the robot acquired a policy which can generates stable biped walking pattern. We applied the acquired controller to a different ground surface. Even on a metal surface, the robot successfully walked using the learned biped walking policy (Fig. 11).

Figure 12 shows joint angle trajectories of the actual robot. The robot generated a stable periodic pattern after 100 trials. During each step, the robot straightened its leg, which is uncommon in the popular ZMP approach due to the necessity of avoiding singularities.

Figure 13 shows the accumulated reward at each trial using the real robot. The robot learned a stable walking pattern within 100 trials.

An acquired value function after 100 trials is shown in Figure 14. The minimum value of the value function is located around zero body position $d = 0.0$ and negative body velocity $\dot{d}$, and the maximum value of the value function is located around zero body position $d = 0.0$ and positive body velocity $\dot{d}$. The difference between shape of the value function acquired in the simulated environment (Fig. 6) and the real environment (Fig. 14) is possibly caused by the effect of the boom.

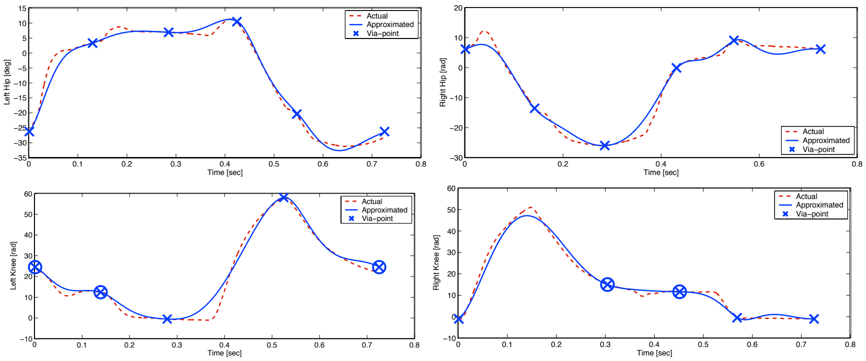

Fig. 8. Nominal joint-angle trajectories and detected via-points represented by cross (×). Manually selected via-points represented by circle (○) are modulated by control output θ_{act}.

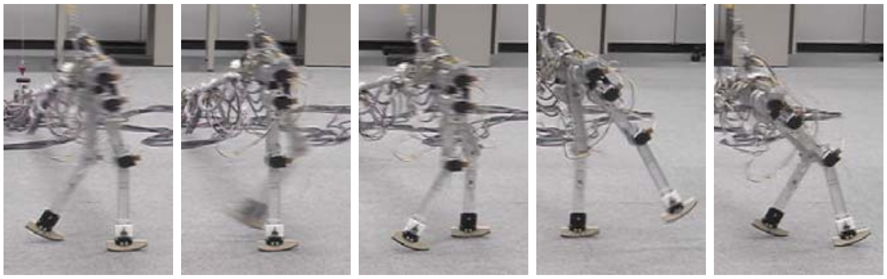

Fig. 9. Biped walking pattern before learning

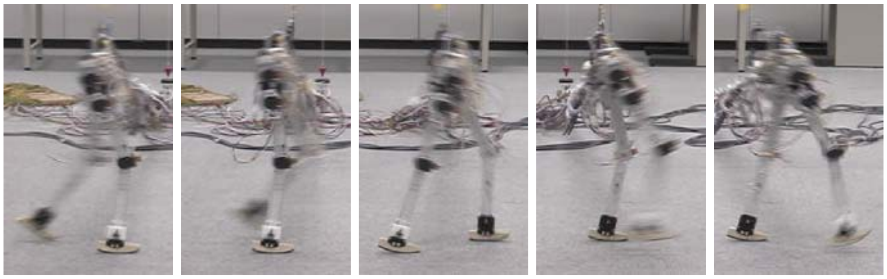

Fig. 10. Biped walking pattern after learning

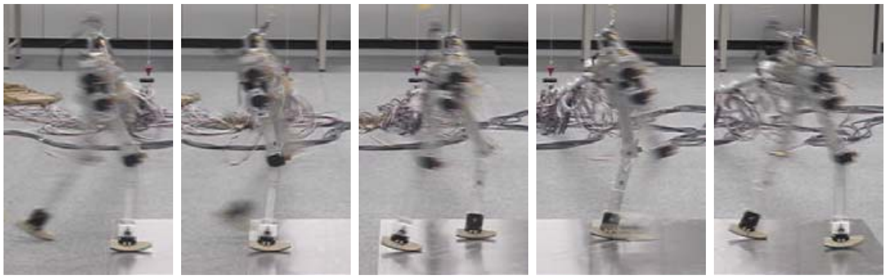

Fig. 11. Biped walking pattern on metal surface

5. Discussion

In this study, we controlled foot placement using the knee because the lower leg has smaller mass and tracking the target joint angle at the knee is easier than tracking using the hip joint. We applied the proposed approach to a physical biped robot and acquired a policy which generates a stable walking pattern. Using hip joints or using different variables for the output of the policy are interesting topics for future

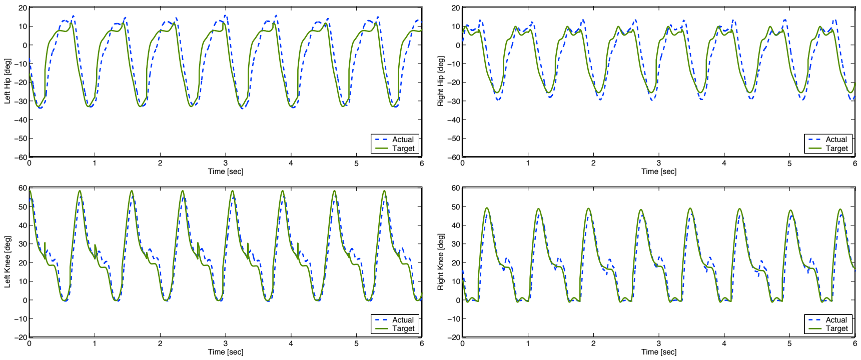

Fig. 12. Joint angle trajectories after learning on real robot

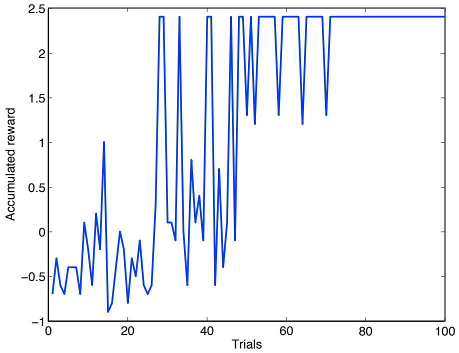

Fig. 13. Accumulated reward at each trial using real robot

work. We are also considering using captured data of a human walking pattern [23] as a nominal trajectory instead of using a hand-designed walking pattern.

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

Acknowledgments

We would like to thank Mitsuo Kawato, 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.

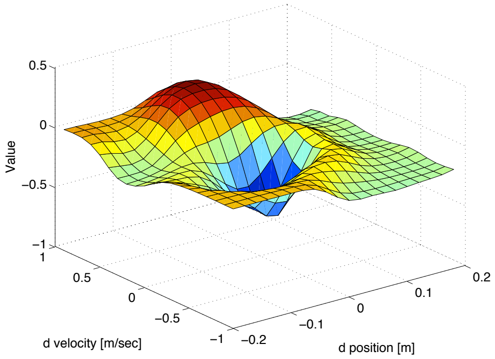

Fig. 14. Shape of acquired value function in real environment

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. *Robotics and Autonomous Systems*, 47:79–91, 2004.
15. M. H. Raibert. *Legged Robots That Balance*. The MIT Press, Cambridge, MA, 1986.
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.