<table>
<thead>
<tr>
<th>Item</th>
<th>Information</th>
</tr>
</thead>
<tbody>
<tr>
<td>Title</td>
<td>Poincare-Map-based Reinforcement Learning for Biped Walking</td>
</tr>
<tr>
<td>Author</td>
<td>Jun Morimoto, Jun Nakanishi, Gen Endo, Gordon Cheng, Christopher Atkeson, Garth Zeglin</td>
</tr>
<tr>
<td>Issue date</td>
<td>2005,</td>
</tr>
<tr>
<td>Copyright</td>
<td>©2005 IEEE. Personal use of this material is permitted. However, permission to reprint/republish this material for advertising or promotional purposes or for creating new collective works for resale or redistribution to servers or lists, or to reuse any copyrighted component of this work in other works must be obtained from the IEEE.</td>
</tr>
</tbody>
</table>
Abstract—We propose a model-based reinforcement learning algorithm for biped walking in which the robot learns to appropriately modulate an observed walking pattern. Via-points are detected from the observed walking trajectories using the minimum jerk criterion. The learning algorithm modulates the via-points as control actions to improve walking trajectories. This decision is based on a learned model of the Poincaré map of the periodic walking pattern. The model maps from a state in the single support phase and the control actions to a state in 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 policies are acquired.

Index Terms—Biped Walking; Reinforcement Learning; Poincaré map

I. INTRODUCTION

We propose a learning algorithm to acquire an appropriate biped walking controllers by modulating an observed walking pattern. We are using model-based reinforcement learning, where we learn a model of a Poincaré map and then choose control actions based on a computed value function. We detect via-points from an observed walking trajectory and use the via-points as control actions.

Several researchers have applied reinforcement learning to biped locomotion [12], [2]. Few studies deal with a physical robots because reinforcement learning methods often require large numbers of trials. The policy gradient method [17] is one of the reinforcement learning methods successfully applied to learn biped walking on actual robots [1], [19]. However, [1] requires hours to learn a walking controller, and [19] requires a mechanically stable robot.

On the other hand, [3] reported that a model-based approach to reinforcement learning is able to accomplish given tasks much faster than without using knowledge of the environment. In our previous work [11], we showed that a model-based approach using an approximated Poincaré map could be applied to learn biped walking in small numbers of trials. However, we used an empirically designed nominal trajectory for the proposed method, and acquired a successful walking pattern only in a simulated environments. In this study, we use observed trajectories, such as those of humans or other robots controlled by this or other algorithms, as nominal trajectories. We show that the proposed method can be applied to an actual robot (Fig. 1).

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 I are selected to model the actual biped robot fixed to a boom that keeps the robot in the sagittal plane (Fig. 1). Our biped has a short torso and round feet without ankle joints. For these bipeds, controlling biped walking trajectories with the popular ZMP approach [7], [22] is difficult or impossible, and thus an alternative method for controller design must be used.

In section II, we introduce our reinforcement learning method for biped walking. In section III, we show simulation results. In section IV, we present an implementation of the proposed method on the real robot, and then demonstrate that the robot acquires a successful walking pattern within 100 trials.

II. POINCARÉ-MAP-BASED REINFORCEMENT LEARNING FOR BIPED LOCOMOTION

We improve biped walking controllers based on an approximated Poincaré map using a model-based reinforce-
moment learning framework [3], [16]. The Poincaré map represents the locus of intersection of the biped trajectory with a hyperplane subspace of the full trajectory state space. In our case, we are interested in the system state at two symmetric phase angles of the walking gait. Modulating via-points affects the locus of intersection and our learned model reflects this effect. Given a learned mapping, we proceed to learn a corresponding value function for states at phases $\phi = \frac{\pi}{2}$ and $\phi = \frac{3\pi}{2}$ (Fig. 2), where we define phase $\phi = 0$ as the left foot touchdown.

The input state is defined as $x = (d, \dot{d})$, where $d$ denotes the horizontal distance between the stance foot position and the body position (Fig. 1). We use the hip position as the body position because the center of mass is nearly coincident with the hips (Fig. 1). We use a human walking pattern in [5] as the nominal trajectory (Fig. 3). The action of the robot $u = \theta^{act}(x) = \left(\theta_{\text{hip}}^{act}, \theta_{\text{knee}}^{act}\right)$ modulates the via-points of the nominal trajectory at each joint:

\[
\theta_{\text{hip}}^{act} = \overline{\theta}_{\text{hip}} + \theta_{\text{hip}}^{v}(x) \quad (i = 1, \cdots, n_{\text{hip}}),
\]

\[
\theta_{\text{knee}}^{act} = \overline{\theta}_{\text{knee}} + \theta_{\text{knee}}^{v}(x) \quad (i = 1, \cdots, n_{\text{knee}}),
\]

where $n_{\text{hip}} = 1$ and $n_{\text{knee}} = 2$ denote the number of selected via-points, and $\theta_{\text{hip}}^{v}$ and $\theta_{\text{knee}}^{v}$ denote the nominal value of the selected via-points. Each selected via-point on a same joint is equally modulated by the control output $\theta^{act}$.

![Fig. 2. Biped walking cycle: we update parameters and select actions at Poincaré sections at phase $\phi = \frac{\pi}{2}$ and $\phi = \frac{3\pi}{2}$. L: left leg, R: right leg.](image)

**A. Function approximator**

We use Receptive Field Weighted Regression (RFWR) [15] as the function approximator for the policy, the value function and the estimated Poincaré map. We approximate a target function $g(x)$ with

\[
\hat{g}(x) = \frac{\sum_{k=1}^{N_b} a_k(x) h_k(x)}{\sum_{k=1}^{N_b} a_k(x)},
\]

\[
h_k(x) = w^T_k \hat{x}_k,
\]

\[
a_k(x) = \exp \left( -\frac{1}{2} (x - c_k)^T D_k (x - c_k) \right),
\]

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

\[
\Delta w_k = a_k P_k \hat{x}_k (g(x) - h_k(x)),
\]

where

\[
P_k = \frac{1}{\lambda} \left( P_k - \frac{P_k \hat{x}_k \hat{x}_k^T P_k}{\hat{x}_k^T P_k \hat{x}_k} \right),
\]

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.,

\[
\max_k a_k(x) < a_{min},
\]

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

![Fig. 3. Nominal joint-angle trajectories observed from a human walking pattern and detected via-points represented by cross ($\times$). Manually selected via-points represented by circle ($\circ$) are modulated by control output $\theta^{act}$. Note that amplitude of the human walking pattern is multiplied by 0.7 to match the small size robot (Fig 1).](image)

**Table I**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Primary Joint</th>
<th>Trunk</th>
<th>Thigh</th>
<th>Shin</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mass (kg)</td>
<td>Primary Joint</td>
<td>2.0</td>
<td>0.64</td>
<td>0.15</td>
</tr>
<tr>
<td>Length (m)</td>
<td>Primary Joint</td>
<td>0.01</td>
<td>0.2</td>
<td>0.2</td>
</tr>
<tr>
<td>Inertia ($\times 10^{-4}$ [kg-m$^2$])</td>
<td>Primary Joint</td>
<td>1.0</td>
<td>0.9</td>
<td>1.4</td>
</tr>
</tbody>
</table>

![Diagram of joint angles and locomotion](image)
functions are 400(=20×20) for approximating the policy and the value function. We put 1 basis function at the origin for approximating the Poincaré map. We set the distance metric \( D_k \) to \( D_k = \text{diag}(2500, 90) \) for the policy and the value function, and \( D_k = \text{diag}(2500, 225, 1600, 1600) \) for the Poincaré map. The centers of the basis functions \( c_k \) and the distance metrics of the basis functions \( D_k \) are fixed during learning.

B. Learning the Poincaré 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 modulated via-points. We are predicting the location of the system in a Poincaré section at phase \( \phi = \frac{\pi}{2} \) based on the system's location in a Poincaré section at phase \( \phi = \frac{\pi}{4} \) (Fig. 2). We use a different model to predict the location at phase \( \phi = \frac{\pi}{2} \) based on the location at phase \( \phi = \frac{3\pi}{4} \) due to the real robot possessing asymmetries caused by a supporting boom.

Because the state of the robot drastically changes at foot touchdown \( (\phi = 0, \pi) \), we select the phases \( \phi = \frac{\pi}{2} \) and \( \phi = \frac{3\pi}{4} \) as Poincaré sections. We approximate this Poincaré map using a function approximator with a parameter vector \( w^m \),

\[
x_{\phi^m} = \hat{f}_1(x_{\phi}, u_{\phi}; w^m_1),
\]

\[
x_{\phi^m} = \hat{f}_2(x_{\phi}, u_{\phi}; w^m_2),
\]

where the input state is defined as \( x = (d, \dot{d}) \), and the action of the robot is defined as \( u = \theta^{act}(x) \).

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

We interpolated trajectories between the via-points by using the minimum jerk criteria [6], [21]. To follow the generated target trajectories, the torque output at each joint is given by a PD servo controller:

\[
\tau_j = k(\theta^j(\phi) - \theta_j) - db_j,
\]

where \( \theta^j(\phi) \) is the target joint angle for \( j \)-th joint \( (j = 1 \ldots 4) \), position gain \( k \) is set to \( k = 4.0 \) except for the knee joint of the stance leg (we use \( k = 9.0 \) for the knee joint of the stance leg), and the velocity gain \( b \) is set to \( b = 0.1 \).

We reset the phase [20], [13] to \( \phi = \phi_{reset} \) at left foot touchdown and to \( \phi = \pi + \phi_{reset} \) at right foot touchdown, where \( \phi_{reset} = 0.7 \) rad is empirically determined.

D. Rewards

The robot gets a reward \( r \) according to the cost \( r_{cost} \) and walking velocity \( r_{vel} \):

\[
r(t) = r_{cost}(t) + r_{vel}(t),
\]

where \( r_{cost}(t) = 0.1 \sum_j \tau_j^2(\phi) \Delta t \), \( r_{vel}(t) = \nu(t) \Delta t \), \( \nu(t) \) m/sec denotes walking speed, and \( \Delta t = 0.001 \) sec. The robot gets punishment (negative reward) if it falls down.

If the height of the body goes below 0.38m, the robot is given a negative reward (-1) and the trial is terminated.

E. 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 \( \mu \):

\[
V^\mu(x(t)) = E[r(t+1) + \gamma r(t+2) + \gamma^2 r(t+3) + ...],
\]

where \( r(t) \) is the reward at time \( t \), and \( \gamma \) \( (0 \leq \gamma \leq 1) \) is the discount factor. In this framework, we evaluate the value function only at \( \phi(t) = \frac{\pi}{2} \) and \( \phi(t) = \frac{3\pi}{4} \).

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 \( w^v \) to represent the value function:

\[
\hat{V}(t) = \hat{V}(x(t); w^v).
\]

By considering the deviation from equation (13), we can define the temporal difference error (TD-error) [16], [18]:

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

where \( t_T \) is the time when \( \phi(t_T) = \frac{\pi}{2} \) or \( \phi(t_T) = \frac{3\pi}{4} \).

The update rule for the value function can be derived as

\[
\hat{V}(x(t)) \leftarrow \hat{V}(x(t)) + \beta \delta(t),
\]

where \( \beta = 0.2 \) is a learning rate. The parameter vector \( w^v \) is updated by equation (6).

F. Learning a policy for biped locomotion

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

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

where \( A(x(t); w^a) \) denotes the mean of the model, which is represented by a function approximator, where \( w^a \) is a parameter vector. We changed the variance \( \sigma \) according to the trial as \( \sigma = 0.1 \left( \frac{150 - N_{trial}}{150} \right) + 0.01 \) 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

\[
u(t) = A(x(t); w^a) + \sigma u(t),
\]

where \( u(t) \sim N(0, 1) \). \( N(0, 1) \) 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 Poincaré map.

1. Predict the next state \( \hat{x}(t_T) \) from the current state \( x(t) \) and the nominal action \( u = A(x(t); w^a) \) using the Poincaré map model \( \hat{x}(t_T) = F(x(t), u(t); w^m) \).

2. Derive the gradient of the value function \( \frac{\partial V}{\partial x} \) at the predicted state \( \hat{x}(t_T) \).
3) Derive the gradient of the dynamics model \( \frac{df}{dx} \) at the current state \( x(t) \) and the nominal action \( u = A(x(t); w^a) \).

4) Update the policy \( \mu \):

\[
A(x; w^a) \leftarrow A(x; w^a) + \alpha \frac{\partial V(x)}{\partial x} \frac{\partial f(x, u)}{\partial u},
\]

where \( \alpha = 0.2 \) is the learning rate. The parameter vector \( w^a \) is updated by equation (6). We can consider the output \( u(t) \) is an option in the SMDP [18] initiated in state \( x(t) \) at time \( t \) when \( \phi(t) = \frac{1}{2} \pi \) (or \( \phi = \frac{3}{2} \pi \)), and it terminates at time \( t_T \) when \( \phi = \frac{1}{2} \pi \) (or \( \phi = \frac{3}{2} \pi \)).

III. SIMULATION RESULTS

We applied the proposed method to the 5 link simulated robot (Fig. 1). We used a manually generated initial step to get the pattern started. We set the walking period to \( T = 0.9 \text{ sec} \) (\( \omega = 7.0 \text{ rad/sec} \)). A trial is terminated after \( 15 \) seconds or after the robot falls down. Figure 4 (Top) shows the walking pattern before learning.

We defined a successful trial when the robot continuously walks for more than \( 15 \) seconds, which approximately corresponds to 30 steps. Figure 5 shows the accumulated reward, averaged over 10 simulation runs, at each trial. Stable walking controllers were acquired within \( 200 \) trials (Fig. 5) for every 10 simulation runs. However, the performance of the acquired controller has a large variance at each simulation run. This is probably because scheduling of the stochasticity of the policy \( \sigma \) in (17) was not appropriate. We need to include \( \sigma \) as a parameter of the policy in our future work [17].

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 \( \dot{d} \) that leads to a successful walk. The number of allocated basis functions are 400 for approximating the value function, 400 for approximating the policy, 443 for the Poincaré map \( f_1 \) in equation (9), and 393 for the Poincaré map \( f_2 \) in equation (10).

The acquired walking pattern is shown in Figure 4 (Bottom). Figure 7 (Left) shows a phase diagram of a successful walking pattern in the state space \( x = (d, \dot{d}) \) after learning. A gradual increase of walking speed can be observed. Figure 7 (Right) shows loci of the walking trajectory at the Poincaré section. The walking trajectory after learning passes through the section at almost same place after a few steps.

IV. REAL ROBOT IMPLEMENTATION

We applied the proposed model-based reinforcement learning scheme to a real biped (Fig. 1). We use a walking pattern generated by a pre-designed state machine controller [14] as the nominal walking pattern (Fig 8).

We detect via-points from this nominal walking pattern and manually select via-points that correspond to foot placement (Fig 8). In this experiment, the control output \( u = \theta^{act}(x) = (\theta_{knee}^{act}) \) modulate the selected via-points:

\[
\theta_{knee}^{act} = \theta_{knee}^{vp} + \theta_{knee}^{act}(x) \quad (i = 1, \cdots, n_{knee}^{v}).
\]

On each transition from phase \( \phi = \frac{1}{2} \pi \) (or \( \phi = \frac{3}{2} \pi \)) to phase \( \phi = \frac{1}{2} \pi \) (or \( \phi = \frac{3}{2} \pi \)), the robot gets a reward of \( 0.1 \), if the height of the body remains above \( 0.38 \)m during the past half cycle. The robot gets punishment (negative reward -1) if it falls down.

We changed the variance \( \sigma \) in equation (17) according to the trial of \( \sigma = 0.1 \left( \frac{50 - N_{trial}}{N_{trial}} \right) + 0.01 \) 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 \( T = 0.84 \text{ sec} \) (\( \omega = 7.5 \text{ rad/sec} \)). A trial is terminated after \( 30 \) steps or after the robot fell down. We use the pre-designed state machine for the initial 6 steps. We set the distance metric \( D_k \) in equation (5) to \( D_k = diag(2500, 90) \) for the policy and the value function, and \( D_k = diag(2500, 90, 1600) \) for the Poincaré map.

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

Figure 8 shows a biped walking pattern before learning. The robot fell over with the nominal walking pattern. After \( 100 \) trials in the real environment, the robot acquired a policy that generated a 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. 10).

Figure 11 shows the accumulated reward at each trial using the real robot. The robot learned a stable walking controller within 100 trials.

An acquired value function after 100 trials is shown in Figure 12. 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. 12) is possibly caused by the effect of the boom. The number of allocated basis functions are 407 for approximating the value function, 401 for approximating the policy, 59 for the Poincaré map \(f_1\) in equation (9), and 59 for the Poincaré map \(f_2\) in equation (10).

V. DISCUSSION

In this study, we proposed Poincaré-map-based reinforcement learning and applied the proposed method to biped locomotion. The simulated robot acquired the biped walking controller using observed human walking pattern as the nominal trajectory. We also applied the proposed approach to a physical biped robot and acquired a policy, which successfully generated a walking pattern. We are currently trying to use a human walking trajectory as the nominal trajectory on the real biped. Automatic selection of the via-points to be used as control actions is part of our future work. In our previous work, we have proposed a trajectory optimization method for biped locomotion [9], [10] based on differential dynamic programming [4], [8]. We are now considering combining this trajectory optimization method with the proposed reinforcement learning method.

REFERENCES

Fig. 9. Biped walking pattern before learning

Fig. 10. Biped walking pattern on metal surface

Fig. 12. Shape of acquired value function in real environment


ACKNOWLEDGMENT

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 awards ECS-0323538 and CNS-0224419.