Auton Robot
DOI 10.1007/s10514-009-9133-z

# Nonparametric representation of an approximated Poincaré map for learning biped locomotion

**Jun Morimoto · Christopher G. Atkeson**

Received: 17 November 2008 / Accepted: 2 August 2009

**Abstract** We propose approximating a Poincaré map of biped walking dynamics using Gaussian processes. We locally optimize parameters of a given biped walking controller based on the approximated Poincaré map. By using Gaussian processes, we can estimate a probability distribution of a target nonlinear function with a given covariance. Thus, an optimization method can take the uncertainty of approximated maps into account throughout the learning process. We use a reinforcement learning (RL) method as the optimization method. Although RL is a useful non-linear optimizer, it is usually difficult to apply RL to real robotic systems due to the large number of iterations required to acquire suitable policies. In this study, we first approximated the Poincaré map by using data from a real robot, and then applied RL using the estimated map in order to optimize stepping and walking policies. We show that we can improve stepping and walking policies both in simulated and real environments. Experimental validation on a humanoid robot of the approach is presented.

**Keywords** Bipedal walking · Reinforcement learning · Poincaré map · Gaussian processes · Humanoid robot

J. Morimoto (✉)
Computational Brain Project, Japan Science and Technology Agency, ICORP, Saitama, Japan
e-mail: xmorimo@atr.jp

J. Morimoto
Department of Brain Robot Interface, ATR Computational Neuroscience Laboratories, Kyoto, Japan

C.G. Atkeson
The Robotics Institute, Carnegie Mellon University, Pittsburgh, USA
e-mail: cga@cs.cmu.edu

Published online: 01 September 2009

## 1 Introduction

Local stability of biped walking patterns has been evaluated using Poincaré maps (McGeer 1990; der Linde 1999). If a dynamical model of a biped robot and a ground contact model are available, a Poincaré map can be derived analytically for a specified periodic orbit (Westervelt et al. 2004; Shiriaev et al. 2005). However, in real environments, the correct robot model and the ground contact model are difficult to identify. Therefore, we consider representing a Poincaré map by using a nonparametric representation based on sampled data from the real environment.

As the amount of data increases, the complexity of the Poincaré map model can be flexibly increased if we use a nonparametric approximation method such as a Gaussian Process (Rasmussen and Williams 2006). We try to locally optimize parameters of a given biped walking controller based on the approximated Poincaré map. We use a reinforcement learning (RL) method as the optimization method.

A Gaussian process model allows us to estimate the probability distribution of a target nonlinear function with a given covariance. Gaussian processes can estimate the accuracy of the approximated function based on the density of the sampled data. This is beneficial, as it is difficult to uniformly collect data from a real robot due to unknown dynamics. By using this stochastic model, RL can take the accuracy of the approximated model into account throughout the learning process.

RL does not require a precise environmental model, and can be a useful technique to improve task performance of real robots. However, one drawback of using RL is that it usually requires a large number of iterations to improve policies. For this reason applications of RL to real environments have been limited (Benbrahim and Franklin 1997;

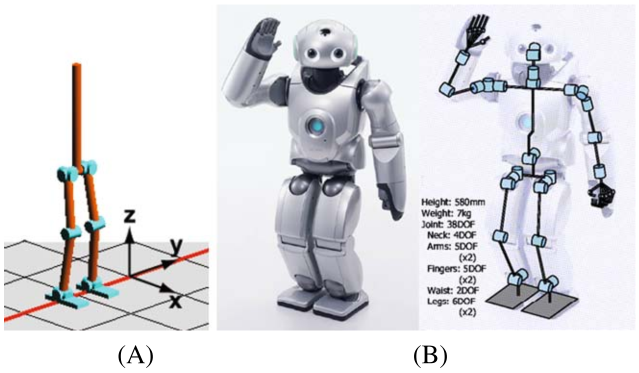

**Fig. 1** (**A**) 3D biped simulation model. Height: 1.6 m, total weight: 95 kg. (**B**) Small humanoid robot used in the experiment

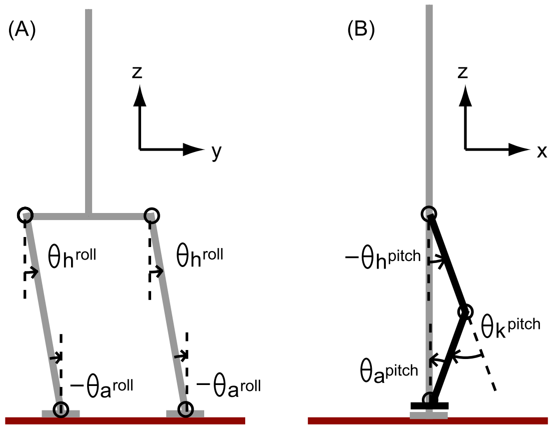

**Fig. 2** Joint index of biped model: (**A**) Roll joints for lateral movements. (**B**) Pitch joints for lateral and forward movements

Morimoto and Doya 2001; Tedrake et al. 2004; Morimoto and Atkeson 2007; Matsubara et al. 2006; Endo et al. 2008; Peters and Schaal 2008).

Methods to improve policy parameters by using inaccurate models have been studied in Atkeson and Schaal (1997), Atkeson (1998), Abbeel et al. (2006). In our approach, we propose the use of a task specific stochastic model instead of using a robot in a real environment to improve a policy.

In this study, we focus on improving the locomotive performance of biped robots as an application of our learning framework. The dynamics of biped robots characteristically include contact and collision with the ground. Modeling the interaction with the ground is difficult in general. Using RL methods can be a suitable approach to improve a walking policy. We directly approximate Poincaré maps of stepping and walking dynamics without explicitly identifying rigid body inertial parameters and without the use of a ground contact model.

To show the learning performance on robots which have different kinematics and dynamics, we apply our learning framework to a 3D biped simulation model (see Fig. 1(A)) and a small humanoid robot (see Fig. 1(B)).

In our approach, 1) we first construct a stepping and a walking controller (Morimoto et al. 2006, 2008) introduced in Sect. 2. We use simple periodic functions (sinusoids) as desired joint trajectories, where the phases of the sinusoids are modulated by the phase of the robot dynamics. Then, 2) we control the amplitude of the sinusoids according to the current state of the robot to improve locomotive performance.

In Sect. 3, our learning method, which uses approximated Poincaré maps of stepping and walking dynamics is introduced. In Sect. 4, we explain how we apply a Gaussian process model to approximate stepping and walking dynamics. In Sect. 5, we describe implementation of a RL method for our learning framework, that uses the dynamics approximated by a Gaussian process.

## 2 Periodic pattern generator

Our biped controller uses a coupled oscillator model to modulate the phase of sinusoidal patterns. The aim of using a coupled oscillator model is to synchronize periodic patterns generated by the controller with the dynamics of the robot. To use the coupled oscillator model, detection of the phase of the robot is needed. We introduce a method to detect robot phase in Sect. 2.1. We briefly explain phase coordination for biped walking in Sect. 2.2. We use simple sinusoidal patterns as nominal trajectories for each joint. We describe the design of the nominal trajectories for stepping movements in Sect. 2.3, and walking movements in Sect. 2.4. Conventions for representing joint movement are presented in Fig. 2.

### 2.1 Phase detection

By using the oscillator model, we can independently design the phase dynamics and amplitude of periodic walking patterns (Tsuchiya et al. 2003). However, to take the sensory information into account in the oscillator system, we need to explicitly detect the phase of the robot dynamics.

As shown by our previous study (Morimoto et al. 2006, 2008), we can use the center of pressure $y_{cop}$ and the velocity of the center of pressure $\dot{y}_{cop}$ to detect the phase of the robot dynamics:

$$\phi(\mathbf{y}_{cop}) = -\arctan\left(\frac{\dot{y}_{cop}}{y_{cop}}\right), \tag{1}$$

where $\mathbf{y}_{cop} = (y_{cop}, \dot{y}_{cop})$ (see Fig. 3). We use a simplified COP detection method introduced in Morimoto et al. (2006, 2008).

### 2.2 Phase coordination

In this study, we use four oscillators with phases $\phi_c^i$, where $i = 1, 2, 3, 4$. We introduce coupling between the oscillators

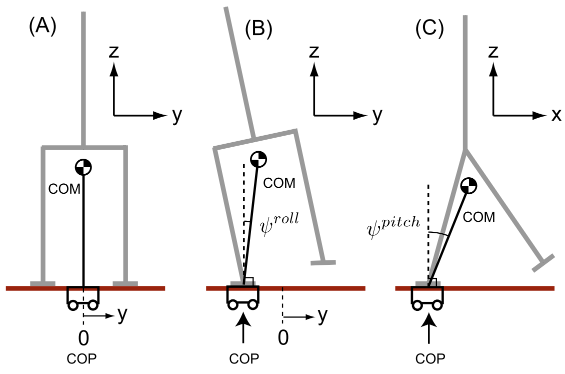

**Fig. 3** (**A**) Inverted pendulum model represented by the center of pressure (COP) and the center of mass (COM). (**B**) $\psi^{roll}$ denotes the roll angle of the pendulum. (**C**) $\psi^{pitch}$ denotes the pitch angle of the pendulum

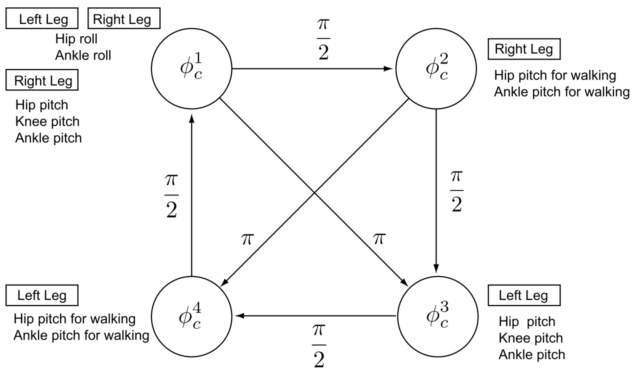

**Fig. 4** Phase coordination. We use four oscillators with phases $\{\phi_c^1, \phi_c^2, \phi_c^3, \phi_c^4\}$ to make symmetric patterns for a lateral movement with the left and right limbs, and also to make symmetric patterns for a forward movement with the left and right limbs. Arrows "$\rightarrow$" indicate phase advance, e.g., $\phi_c^2 = \phi_c^1 + \frac{\pi}{2}$ and $\phi_c^3 = \phi_c^1 + \pi$

and the phase of the robot dynamics $\phi(\mathbf{y}_{cop})$ in (1) to regulate the desired phase relationship between the oscillators:

$$\dot{\phi}_c^i = \omega_c + K_c \sin(\phi(\mathbf{y}_{cop}) - \phi_c^i + \phi_d^i), \tag{2}$$

where $\phi_d^i$ is the desired phase difference, $K_c$ is a coupling constant, and $\omega_c$ is the natural angular frequency of oscillators.

We use four desired phase differences, $\{\phi_d^1, \phi_d^2, \phi_d^3, \phi_d^4\} = \{-\frac{1}{2}\pi, 0.0, \frac{1}{2}\pi, \pi\}$, to make symmetric patterns for stepping movement with the left and right limbs, and also to make symmetric patterns for forward movement with the left and right limbs. Figure 4 shows the phase coordination.

### 2.3 Desired joint angles

1) *Roll joint movements*: We introduce a controller to generate lateral movement. We control the hip joints $\theta_{h_{roll}}$ and the ankle joints $\theta_{a_{roll}}$ (Fig. 2(A)) for this movement. Desired joint angles for each joint are:

$$\theta^d_{h_{roll}}(\phi_c) = A_{h_{roll}} \sin(\phi_c), \tag{3}$$

$$\theta^d_{a_{roll}}(\phi_c) = -A_{a_{roll}} \sin(\phi_c), \tag{4}$$

where $A_{h_{roll}}$ and $A_{a_{roll}}$ are the amplitudes of a sinusoidal function for lateral movements at the hip and the ankle joints, and we use an oscillator with the phase $\phi_c = \phi_c^1$.

2) *Pitch joint movements*: To achieve foot clearance, we generate vertical movement of the feet (Fig. 2(B)) by using simple sinusoidal trajectories:

$$\begin{aligned} \theta^d_{h_{pitch}}(\phi_c) &= A^{pitch} \sin(\phi_c) + \theta^{res}_{h_{pitch}}, \\ \theta^d_{k_{pitch}}(\phi_c) &= -2A^{pitch} \sin(\phi_c) + \theta^{res}_{k_{pitch}}, \\ \theta^d_{a_{pitch}}(\phi_c) &= -A^{pitch} \sin(\phi_c) + \theta^{res}_{a_{pitch}}, \end{aligned} \tag{5}$$

where $A^{pitch}$ is the amplitude of a sinusoidal function. $\theta^{res}_{h_{pitch}}$, $\theta^{res}_{k_{pitch}}$, $\theta^{res}_{a_{pitch}}$ represent the rest posture of the hip, knee, and ankle joints respectively. We use the oscillator with phase $\phi_c = \phi_c^1$ for right limb movement and use the oscillator with phase $\phi_c = \phi_c^3$, which has phase difference of $\phi_c^3 = \phi_c^1 + \pi$, for left limb movement as in Fig. 4.

### 2.4 Additional pitch joint movements for walking behaviors

To walk forward, we use an additional sinusoidal trajectory. Thus, the desired nominal trajectories for the right hip and ankle pitch joints become:

$$\begin{aligned} \theta^{d\_walk}_{h_{pitch}} &= A_{walk} \sin(\phi_c^2) + \theta^d_{h_{pitch}}(\phi_c^1), \\ \theta^{d\_walk}_{a_{pitch}} &= -A_{walk} \sin(\phi_c^2) + \theta^d_{a_{pitch}}(\phi_c^1). \end{aligned} \tag{6}$$

We use the phase $\phi_c^2$, which has a $\frac{1}{2}\pi$ phase difference of $\phi_c^1$ for the right limb. We use $\phi_c^3$ and $\phi_c^4$ for the left limb instead of $\phi_c^1$ and $\phi_c^2$.

## 3 Learning framework

We consider approximating Poincaré maps for stepping and walking dynamics to improve task performance through model-based reinforcement learning (Kuvayev and Sutton 1996). A number of biped walking studies have emphasized that humanoid robots have inverted pendulum dynamics (see Fig. 3), with the top of the pendulum at the center of mass and the base at the center of pressure. Control strategies to stabilize such dynamics have been proposed (Miyazaki and Arimoto 1981; Miura and Shimoyama 1984; Sugihara and Nakamura 2002; Hyon et al. 2007). In this study, we propose using the state of the inverted pendulum

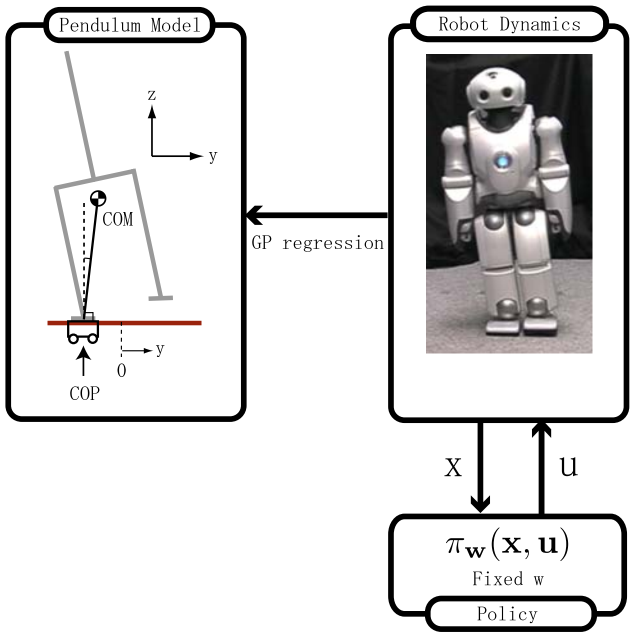

**Fig. 5** Step 2 and 3 in Algorithm 1. Apply the current policy with fixed policy parameters **w** to the actual robot dynamics and sample data. Generate a Gaussian process model which represents the stepping and walking dynamics

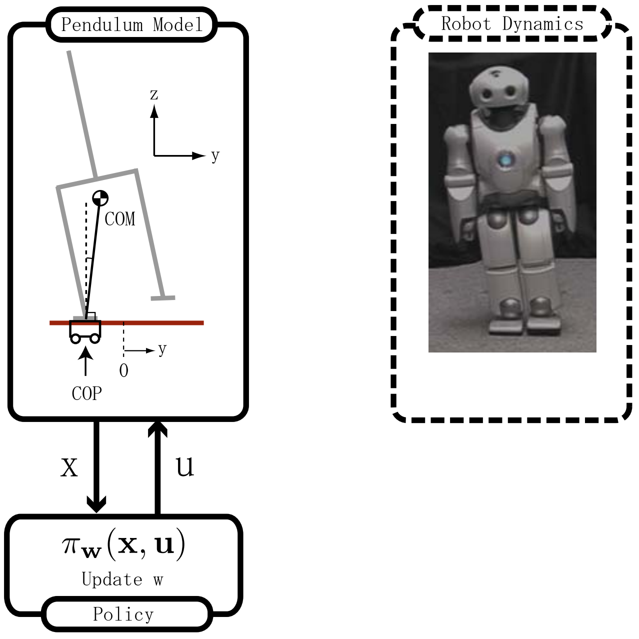

**Fig. 6** Step 4 in Algorithm 1. Update policy parameters **w** by applying a reinforcement learning method to the acquired inverted pendulum model represented by a Gaussian process

**Algorithm 1**

```
1. Initialize policy parameters.
2. Apply the current policy to the actual robot dynamics and
sample data at the defined Poincaré section (see Fig. 5).
3. Generate a Gaussian process model which represents
the stepping and walking dynamics in (7).
4. Update policy parameters by applying a reinforcement
learning method to the acquired Gaussian process (see
Fig. 6).
5. If the policy is not improved, terminate the iteration.
Otherwise, go back to step 2.
```

as the input state for the learning system to make learning tractable.

We assume that nominal stepping and walking controllers are provided (see Sect. 2), and our learning system improves the performance of these controllers. Since the nominal controller can generate periodic movements, we only consider the pendulum state at the Poincaré section.

For example, we consider the dynamics $\dot{\boldsymbol{\xi}} = \mathbf{g}(\boldsymbol{\xi})$ of a state vector $\boldsymbol{\xi} \in \mathbf{R}^n$. The Poincaré map is a mapping from an $n-1$ dimensional surface $S$ defined in the state space to itself (Strogatz 1994). If $\boldsymbol{\xi}(k) \in S$ is the $k$-th intersection, then the Poincaré map $\mathbf{h}$ is defined by $\boldsymbol{\xi}(k+1) = \mathbf{h}(\boldsymbol{\xi}(k))$. In our study, we defined the section for which the roll derivative of the pendulum equals zero ($\dot{\psi}^{roll} = 0$) (see Fig. 3).

The policy of the learning system outputs the next action only at this section. We also assume that we can represent the Poincaré map by a stochastic model. If $\mathbf{x}(k)$ is the $k$-th intersection and $\mathbf{u}(k)$ is the control output at the intersection, the model is defined by:

$$\mathbf{x}(k+1) = \mathbf{f}(\mathbf{x}(k), \mathbf{u}(k)) + \mathbf{n}(k), \tag{7}$$

where $\mathbf{x} = (\psi^{roll})$ for stepping and $\mathbf{x} = (\psi^{roll}, \psi^{pitch}, \dot{\psi}^{pitch})$ for walking (see Fig. 3). $\mathbf{n}(k)$ is the noise input. $\mathbf{f}(\mathbf{x}(k), \mathbf{u}(k))$ represents the deterministic part of the Poincaré map.

To improve task performance, we stochastically modulate the amplitude of the sinusoidal patterns according to the current policy $\pi_{\mathbf{w}}$:

$$\pi_{\mathbf{w}}(\mathbf{x}(k), \mathbf{u}(k)) = p(\mathbf{u}(k) \mid \mathbf{x}(k); \mathbf{w}), \tag{8}$$

where $\mathbf{w}$ is the parameter vector of the policy $\pi_{\mathbf{w}}$. In the following sections, we explain how we approximate the stochastic maps (7), and how we acquire the control policy $\pi_{\mathbf{w}}$.

In our learning framework, we improve the approximated Poincaré map and the policy iteratively (see Algorithm 1). We first sample data from a simulated model or a real robot by using the current policy for a Gaussian process regression as in Fig. 5, and then improve the policy by using the Poincaré map approximated by the Gaussian process as in Fig. 6.

## 4 Non-parametric system identification of Poincaré map

We use a Gaussian process (GP) (Williams and Rasmussen 1996) to approximate the Poincaré map in (7). Gaussian processes provide us a stochastic representation of an approximated function. With Gaussian processes for regression, we assume that the output values $y_i (i = 1, \dots, N)$ are sampled from a zero-mean Gaussian whose covariance matrix is a function of the input vectors $\mathbf{z}_i$ $(i = 1, \dots, N)$:

$$p(y_1, \dots, y_N | \mathbf{z}_1, \dots, \mathbf{z}_N) = \mathcal{N}(y_1, \dots, y_N \mid 0, \mathbf{K}), \tag{9}$$

where $\mathbf{K}$ is an covariance matrix of input vectors with elements $\mathbf{K}_{ij} = \kappa(\mathbf{z}_i, \mathbf{z}_j)$. Here, we used a squared exponential covariance function (Rasmussen and Williams 2006):

$$\kappa(\mathbf{z}_i, \mathbf{z}_j) = v_0 \exp\left( -\frac{1}{2} \sum_{k=1}^{N_d} v_k^d (z_{ik} - z_{jk})^2 \right) + v_1 \delta_{ij}, \tag{10}$$

where $\delta_{ij}$ is Kronecker delta. $z_{ik}$ and $z_{jk}$ are $k$-th element of $i$-th and $j$-th input vectors respectively. $v_0$, $v_1$, and $v_k^d$ are parameters for the covariance matrix. $N_d$ denotes the number of input dimensions. These parameters can be optimized by using a type-II maximum likelihood method (Williams and Rasmussen 1996). A covariance function introduces similarity between data points. By using the squared exponential covariance function, closely placed data points are similar. This similarity measure works well in many applications and is widely used (Rasmussen and Williams 2006; Bishop 2006; Ko and Fox 2009).

Bayesian prediction of an output $y_{N+1}$ corresponding to a new input $\mathbf{z}_{N+1}$ is given as:

$$p(y_{N+1} \mid \mathbf{z}_1, \dots, \mathbf{z}_N, \mathbf{z}_{N+1}, y_1, \dots, y_N) = \mathcal{N}(y_{N+1} \mid \mu, \sigma^2), \tag{11}$$

where

$$\mu = \mathbf{k}(\mathbf{z}_{N+1})^T \mathbf{K}^{-1} \mathbf{y}, \tag{12}$$

$$\sigma^2 = \kappa(\mathbf{z}_{N+1}, \mathbf{z}_{N+1}) - \mathbf{k}(\mathbf{z}_{N+1})^T \mathbf{K}^{-1} \mathbf{k}(\mathbf{z}_{N+1}). \tag{13}$$

Here the vectors $\mathbf{y}$ and $\mathbf{k}(\mathbf{z}_{N+1})$ are defined as $\mathbf{y} = (y_1, \dots, y_N)$ and $\mathbf{k}(\mathbf{z}_{N+1}) = [\kappa(\mathbf{z}_1, \mathbf{z}_{N+1}), \dots, \kappa(\mathbf{z}_N, \mathbf{z}_{N+1})]^T$ respectively.

### 4.1 Simple example

To show how the Gaussian process regression method works, we applied the GP regression to approximate a sinusoidal function:

$$y = \sin(z) + n, \tag{14}$$

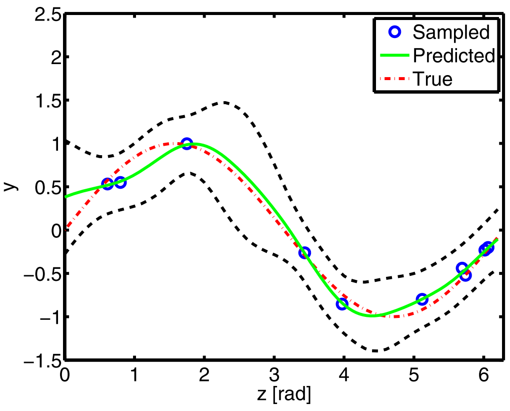

**Fig. 7** Simple example of Gaussian process regression. The *dash-dotted line* shows the target sinusoidal function. *Circles* show sampled data points from the target function with additive noise. The *solid line* shows mean output of the predictive distribution represented by the Gaussian process model. *Dashed lines* show the standard deviation of the predictive distribution

where $n \sim \mathcal{N}(0, 0.01)$ denotes output noise. We used the squared exponential covariance function in (10) with the parameters $v_0 = 1.0$, $v_1 = 0.01$, and $v_k^d = 1.0$.

We sampled ten data points $z_i$ $(i = 1, \dots, 10)$ from a uniform distribution ranged over $0 \leq z \leq 2\pi$. Figure 7 shows a predictive distribution of an approximated sinusoidal function. Since we only sampled ten data points, interpolated regions have large approximation errors. The uncertainty of the interpolated regions are represented by large standard deviations.

An important aspect of the GP regression is that the uncertainty of an approximated function caused by the small number of training samples can be evaluated. As in Fig. 7, the standard deviation of the region where sampled data is not available becomes large.

### 4.2 Application to Poincaré map approximation

We use the GP model to represent how the pendulum model in Fig. 3 behaves with a controller, a robot, actuators, and ground contact models.

The input vector $\mathbf{z}$ for the Gaussian process is composed of the current state $\mathbf{x}(k)$ and control input $\mathbf{u}(k) : \mathbf{z} = (\mathbf{x}(k)^T, \mathbf{u}(k)^T)^T$. The output value $y$ is a component of the state vector at the next intersection $\mathbf{x}(k+1)$, and the control output $\mathbf{u}(k)$ is used to modulate parameters of the stepping and walking controllers. We modulate amplitudes of the sinusoidal patterns that are used as desired joint angles.

The predictive distribution represented by this Gaussian process method is used to improve biped stepping and walking controllers.

## 5 Policy improvement by using a reinforcement learning method

Here, we explain how we applied RL to our biped stepping and walking tasks. RL has been used to improve policies for challenging tasks (Riedmiller et al. 2009; Howard et al. 2009). We used a policy gradient method proposed by (Kimura and Kobayashi 1998), to implement the RL framework. This learning method was used in biped learning studies (Tedrake et al. 2004; Matsubara et al. 2006).

Policy gradient methods are considered as robust learning methods when states for the learning system are partially observable (Jaakkola et al. 1995; Baird and Moore 1999; Meuleau et al. 2001). Convergence properties of policy gradient methods with function approximators have been studied in Sutton et al. (2000), Konda and Tsitsiklis (2003).

Since we applied RL to an approximated Poincaré map instead of a real environment, we did not care much about convergence speed. However, using natural policy gradient methods (Peters and Schaal 2008; Bagnell and Schneider 2003; Kakade 2002) would reduce the time required for the entire learning process.

The basic goal is to find a policy $\pi_{\mathbf{w}}(\mathbf{x}, \mathbf{u}) = p(\mathbf{u} \mid \mathbf{x}; \mathbf{w})$ that maximizes the expectation of the discounted accumulated reward:

$$E\{V(k)|\pi_{\mathbf{w}}\} = E\left\{\sum_{i=k}^{\infty} \gamma^{i-k} r(i) \middle| \pi_{\mathbf{w}}\right\}, \tag{15}$$

where $r$ denotes reward, $V(k)$ is the actual return, $\mathbf{w}$ is the parameter vector of the policy $\pi_{\mathbf{w}}$, and $\gamma$, $0 \le \gamma < 1$, is the discount factor.

In policy gradient methods, we calculate the gradient direction of the expectation of the actual return with respect to the parameters of a policy $\mathbf{w}$. Kimura and Kobayashi (1998) suggested that we can estimate the expectation of the gradient direction as:

$$\frac{\partial}{\partial \mathbf{w}} E\{V(0) \mid \pi_{\mathbf{w}}\} \approx E\left\{\sum_{k=0}^{\infty} (V(k) - \hat{V}(\mathbf{x})) \frac{\partial \ln \pi_{\mathbf{w}}}{\partial \mathbf{w}} \middle| \pi_{\mathbf{w}}\right\}, \tag{16}$$

where $\hat{V}(\mathbf{x})$ is an approximation of the value function for a policy $\pi_{\mathbf{w}}$: $V^{\pi_{\mathbf{w}}}(\mathbf{x}) = E\{V(k) \mid \mathbf{x}(k) = \mathbf{x}, \pi_{\mathbf{w}}\}$.

### 5.1 Value function approximation

The value function is approximated using a normalized Gaussian network (Doya 2000):

$$\hat{V}(\mathbf{x}) = \sum_{i=1}^{N} w_i^v b_i(\mathbf{x}), \tag{17}$$

where $w_i^v$ is a $i$-th parameter of the approximated value function, and $N$ is the number of basis functions $b_i(\mathbf{x})$. An approximation error of the value function is represented by the temporal difference (TD) error (Sutton and Barto 1998):

$$\delta(k) = r(k+1) + \gamma \hat{V}(\mathbf{x}(k+1)) - \hat{V}(\mathbf{x}(k)), \tag{18}$$

We update the parameters of the value function approximator using the TD(0) method (Sutton and Barto 1998):

$$w_i^v(k+1) = w_i^v(k) + \alpha \delta(k) b_i(\mathbf{x}(k)), \tag{19}$$

where $\alpha$ is the learning rate.

### 5.2 Policy parameter update

We update the parameters of a policy $\mathbf{w}$ by using the estimated gradient direction in (16). Kimura and Kobayashi (1998) showed that we can estimate the gradient direction by using the TD error:

$$E\left\{\sum_{k=0}^{\infty} (V(k) - \hat{V}(\mathbf{x}(k))) \frac{\partial \ln \pi_{\mathbf{w}}}{\mathbf{w}} \middle| \pi_{\mathbf{w}}\right\} = E\left\{\sum_{k=0}^{\infty} \delta(k) \mathbf{e}(k) \middle| \pi_{\mathbf{w}}\right\}, \tag{20}$$

where $\mathbf{e}$ is the eligibility trace of the parameter $\mathbf{w}$. Eligibility traces indicate which parameters are eligible for the TD-error and are updated as

$$\mathbf{e}(k+1) = \eta \mathbf{e}(k) + \left.\frac{\partial \ln \pi_{\mathbf{w}}(\mathbf{x}(k), \mathbf{u}(k))}{\partial \mathbf{w}}\right|_{\mathbf{w}=\mathbf{w}(k)}, \tag{21}$$

where $\eta$ is the decay factor for the eligibility trace. Equation (20) can be derived if the condition $\eta = \gamma$ is satisfied. The parameter $\mathbf{w}$ is updated as:

$$\mathbf{w}(k+1) = \mathbf{w}(k) + \beta \delta(k) \mathbf{e}(k). \tag{22}$$

### 5.3 Biped stepping and walking policy

We construct the biped stepping and walking policies based on a normal distribution:

$$\pi_{\mathbf{w}}(\mathbf{x}, \mathbf{u}) = \mathcal{N}(\boldsymbol{\mu}(\mathbf{x}; \mathbf{w}^{\mu}), \boldsymbol{\Sigma}(\mathbf{x}; \mathbf{w}^{\sigma})) \tag{23}$$

where $\mathbf{u}$ is the output vector and $\Sigma$ is the covariance matrix of the policy $\pi_{\mathbf{w}}$. In this study, we defined the covariance matrix as a diagonal matrix, where the $j$-th diagonal element is represented as $\sigma_j$. The $j$-th element of the mean output $\boldsymbol{\mu}$ is modeled by a normalized Gaussian network:

$$\mu_j(\mathbf{x}) = \sum_{i=1}^{N} w_i^{\mu_j} b_i(\mathbf{x}). \tag{24}$$

**Table 1** Parameters of the periodic pattern generator for each experiment

| | Simulated model | | Real robot | |
|---|---|---|---|---|
| | Stepping | Walking | Stepping | Walking |
| $\omega_c$ | 3.5 | 3.5 | 6.3 | 6.3 |
| $K_c$ | 10.0 | 10.0 | 9.4 | 9.4 |
| $A_{h^{roll}}$ | 4.0 | 3.5 | 2.5 | 5.0 |
| $A_{a^{roll}}$ | 4.0 | 3.5 | 8.5 | 9.0 |
| $A^{pitch}$ | 6.0 | 7.0 | 4.0 | 4.0 |

**Table 2** Parameters of policy gradient method

| $\gamma$ | $\alpha$ | $\eta$ | $\beta$ | $\sigma_0$ |
|---|---|---|---|---|
| 0.95 | 0.3 | 0.3 | 0.3 | 0.5 |

Here, $w_i^{\mu_j}$ denotes the $i$-th parameter for $j$-th output of the policy $\pi_{\mathbf{w}}$, and $N$ is the number of basis functions. We represent the diagonal element of the covariance matrix $\Sigma$ using a sigmoid function (Kimura and Kobayashi 1998):

$$\sigma_j(\mathbf{x}) = \frac{\sigma_0}{1+\exp(-\sigma_j^w(\mathbf{x}))}, \quad \text{where } \sigma_j^w(\mathbf{x}) = \sum_{i=1}^{N} w_i^{\sigma_j} b_i(\mathbf{x}), \tag{25}$$

and $\sigma_0$ denotes the scaling parameter. $w_i^{\sigma_j}$ denotes the $i$-th parameter for the $j$-th diagonal element of the covariance matrix. We update the parameters by applying the update rules in (22) and (21).

We use same basis functions $b_i(\mathbf{x})$ in (17), (24), and (25).

## 6 Simulation

We applied our proposed method to the simple 3D simulated biped model in Fig. 1(A). Parameters used in the controllers are summarized in Table 1. Parameters used in the policy gradient method are summarized in Table 2.

### 6.1 Improvement of biped stepping performance

We applied our proposed method to improve stepping in place.

We selected amplitudes of the pitch joint movements as the control output: $\mathbf{u}(k) = A_{step}(k)$. Then, the desired joint angles in (5) become

$$\begin{aligned}
\theta^d_{hpitch}(\phi_c) &= (A^{pitch} + \underline{A_{step}})\sin(\phi_c) + \theta^{res}_{hpitch}, \\
\theta^d_{kpitch}(\phi_c) &= -2(A^{pitch} + \underline{A_{step}})\sin(\phi_c) + \theta^{res}_{kpitch}, \\
\theta^d_{apitch}(\phi_c) &= -(A^{pitch} + \underline{A_{step}})\sin(\phi_c) + \theta^{res}_{apitch}.
\end{aligned} \tag{26}$$

We flip the sign of the roll angle $\psi^{roll}$ in the state vector $\mathbf{x}$ when the sign of COP in the lateral (y) direction (see Fig. 3(A)) changes so that we can use the same policy for the left stance phase and the right stance phase.

We defined the target of the stepping task to keep the desired state at $\psi_d^{roll}$. We use a reward function:

$$r = -K(\psi_d^{roll} - \psi^{roll})^2 \tag{27}$$

for this stepping task, where the desired roll angle $\psi_d^{roll} = 2.0°$ and $K = 0.1$. The learning system also receives a negative reward $r = -1$ if the biped model falls over. We consider that the biped model falls over when the condition $\psi^{roll} < 0.0°$ is satisfied (the COM is outside of the stance foot).

For the Gaussian process (GP) model, we sampled 20 data sets $\{\mathbf{x}(k+1), \mathbf{x}(k), \mathbf{u}(k)\}$ from the simulated environments (at step 2 of Algorithm 1).

We compare learning performance of the proposed method using the Gaussian process model with that of using the full model. The full model means that we run the dynamics simulator of the biped model to learn stepping policies.

Figure 8 compared learning performance on the stepping task by using the GP model and by using the full model. At each visit to the Poincaré section, a reward is given according to the performance of a policy. The average reward (vertical axis) shows the average acquired reward at each visit to the Poincaré section.

Although the variance of the learning performance is larger in early stage of the learning process if we used the GP model, the performance of the acquired policies through the GP model is comparable to the policies acquired through the full model after 10 learning trials. Thus, at least for this particular example, this demonstrates that a full dynamic model is not necessary in order to acquire a policy to achieve sufficient task performance. Especially in extensive simulation studies, we can save computation time, and still may be able to acquire comparable performance.

Furthermore, the acquired stepping policies based on the GP model slightly outperformed the policies acquired by the full model after 8th trial. This difference was significant in terms of the Student's $t$-test with significance level 0.05. Since the gradient estimation of policy parameters can be biased with limited sample data, using the approximated model with a Bayesian estimation method such as the Gaussian process regression would generate better sample data for the gradient estimation. Therefore, it would be interesting to compare Bayesian RL approaches (Dearden et al. 1999; Ghavamzadeh and Engel 2007) with the proposed method as a future study in addition to applying the proposed method to other examples to show how this fact can be generalized to other applications.

Figure 9 shows policies acquired by using the GP model and the full model. Similar policies were acquired in the

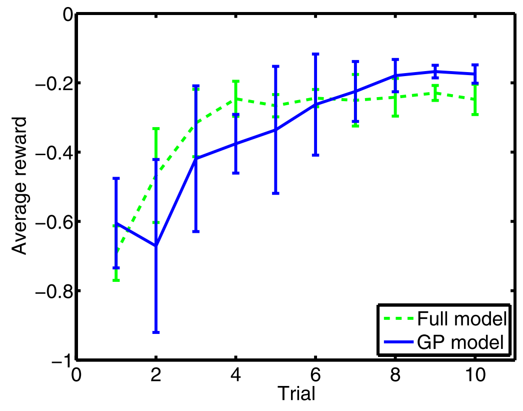

**Fig. 8** A comparison of learning performance. At each visit to the Poincaré section, reward is given according to the performance of a policy. The average reward (*vertical axis*) shows averaged acquired reward at each visit to the Poincaré section. The *solid line* represents the performance of the proposed learning method which used the acquired Gaussian process model. The *dashed line* represents the learning performance of the learning method which used the full dynamics simulation. The result shows the average performance of five simulation runs. *Error bars* show the standard deviation

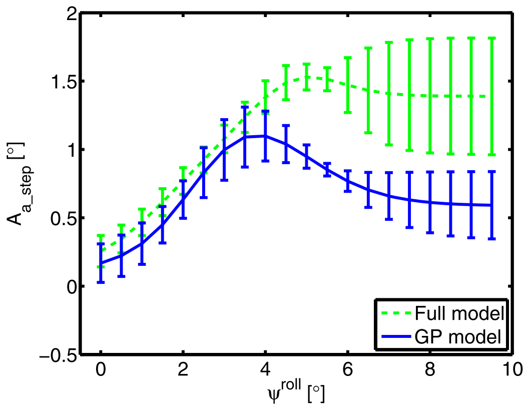

**Fig. 9** Acquired stepping policies. The *solid line* represents acquired policies using the GP model. The *dashed line* represents acquired policies using the full model. The results show an average policy of five simulation runs. *Error bars* show the standard deviation

region $0.0° < \psi^{roll} < 4.0°$, the region of the state space where the policies mainly explored. Outside of the region $4.0° \leq \psi^{roll}$, the policies are different because the data is not frequently sampled from this region. From 3 to 5 basis functions are allocated to represent the mean output of the policies in (24). Therefore, from 3 to 5 parameters were needed to be learned for the mean output. Because we adaptively allocated the basis functions, the number of basis functions are different in different simulation runs (Morimoto and Doya 2001).

We also investigated the robustness of an acquired stepping controller. Figure 10 show the return map of the pendulum state $\psi^{roll}$. Since we change the output $\mathbf{u}(k)$ only at the Poincaré section, it takes two steps to evaluate the control performance of the control output from an acquired policy. We show relationship between $\phi^{roll}(k)$ and $\phi^{roll}(k+2)$. To generate the return map, we disturbed the stepping controller for 0.2 s right after the COP crosses zero by pushing the biped model in the horizontal (y) direction (see Fig. 2) with randomly generated force $F$, where $F$ is sampled from a uniform distribution over $-40 \leq F \leq 40$ N.

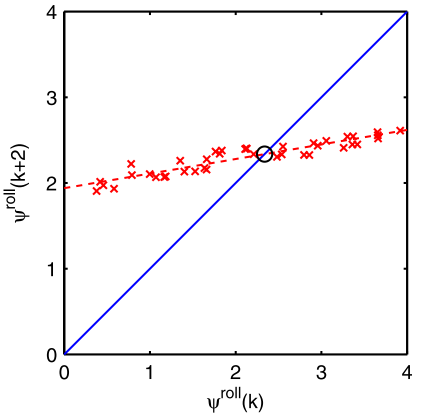

**Fig. 10** Return map of $\phi^{roll}$. We show relationship between $\phi^{roll}(k)$ and $\phi^{roll}(k+2)$. The *dashed line* shows linear approximation of the return map

The dashed line shows the linear approximation of the return map. Since the coefficient of the linear approximation is much smaller than one, this result shows that the acquired policy was able to locally stabilize the stepping dynamics. The approximated fixed point $\phi^{roll} = 2.3°$ is indicated by the circle and is close to the desired $\phi_d^{roll} = 2.0°$.

## 6.2 Improvement of biped walking performance

We also apply our proposed method to improve walking performance. We modulate the amplitude $A_{walk}$ in (6) to generate forward movement for the biped walking task. The target of the walking task is to increase the walking velocity. Since the angular velocity of the pendulum $\dot{\psi}^{pitch}$ (see Fig. 3(C)) at the Poincaré section corresponds to the walking velocity, we use the reward function:

$$r = K(\dot{\psi}^{pitch}) \tag{28}$$

for this biped walking task, where $K = 0.1$. The learning system also receives a negative reward $r = -1$ if the biped model falls over.

For the Gaussian process model, we sampled 100 data sets from the simulated environments. Figure 11 shows the initial performance of the biped walking policy. Figure 12 shows the walking performance after one iteration of the proposed method. One iteration means that 1) acquisition of an approximated Poincaré map from sampled data as in Fig. 5. 2) acquisition of a stepping policy using the acquired

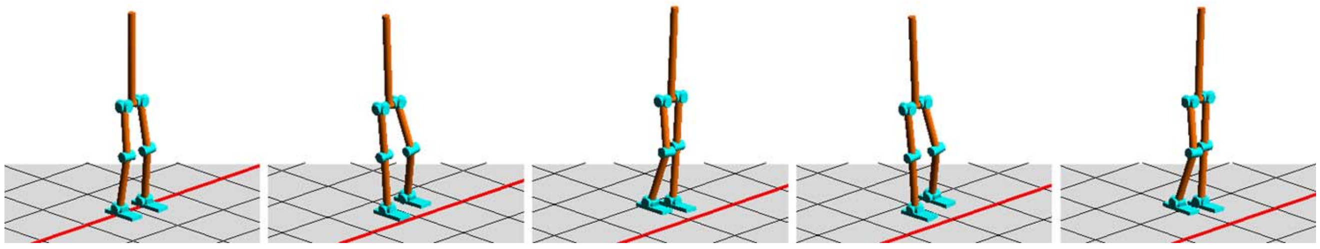

**Fig. 11** Initial walking pattern. The *thick line* represents the starting position. Initially, the simulated robot explores around the starting position. Time proceeds from *left* to *right*

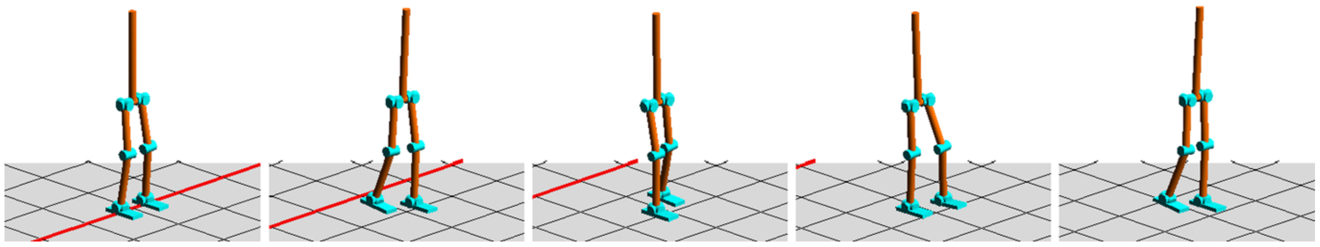

**Fig. 12** Improved walking pattern after one iteration of the proposed learning process. Walking speed is 0.14 m/sec. The *thick line* represents the starting position. Time proceeds from *left* to *right*

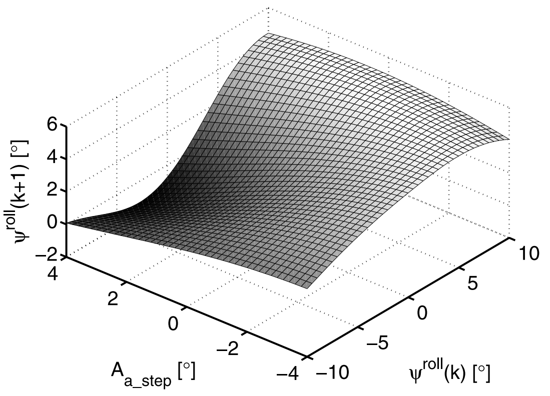

**Fig. 13** Approximated stepping dynamics of the small size humanoid robot by a Gaussian process. The input vector is defined as $\mathbf{z} = (\psi^{roll}(k), A_{a\_step}(k))$, and the output is define as $y = (\psi^{roll}(k+1))$ (see (12))

Poincaré map as in Fig. 6. 3) application of the acquired policy to the simulated robot.

This result suggests that approximated dynamics can be used to improve biped walking performance.

## 7 Experimental results

We applied our proposed method to a small size humanoid robot (see Fig. 1(B)). Parameters used in the controllers are summarized in Table 1.

### 7.1 Improvement of biped stepping performance

We use the roll angle $\mathbf{x} = (\psi^{roll})$ defined by the pendulum model (see Fig. 3) as the state. We selected amplitudes of the ankle roll movements as the control output: $\mathbf{u}(k) = A_{a\_step}$. Thus, the desired joint angle in (4) becomes

$$\theta^d_{a^{roll}}(\phi_c) = -(A_{a^{roll}} + \underline{A_{a\_step}}) \sin(\phi_c). \tag{29}$$

Since the real robot has relatively wide feet, modulating the ankle roll joints is an easy way to control the roll angle $\mathbf{x} = (\psi^{roll})$. Automatic selection of the proper action variables for a given task would be an interesting research topic for future study.

We use the reward function:

$$r = -K(\psi^{roll}_d - \psi^{roll})^2 \tag{30}$$

for this stepping task, where the desired roll angle $\psi^{roll}_d = 0.0°$ and $K = 0.1$. The learning system also receives a negative reward $r = -1$ if the biped model falls over.

For the Gaussian process model, we sampled 20 data sets from the real environment. Figure 13 shows an approximated Poincaré map for stepping dynamics of the small size humanoid robot using a Gaussian process. Here we define the input vector as $\mathbf{z} = (\psi^{roll}(k), A_{a\_step}(k))$ and the output as $y = (\psi^{roll}(k+1))$. We apply the reinforcement learning algorithm to this acquired Poincaré map for stepping dynamics to improve stepping performance.

Figure 14 shows the roll angle $\psi^{roll}$ at the Poincaré section $\dot{\psi}^{roll} = 0$. This result suggests that a stepping policy was acquired by using our proposed method, and it can keep the

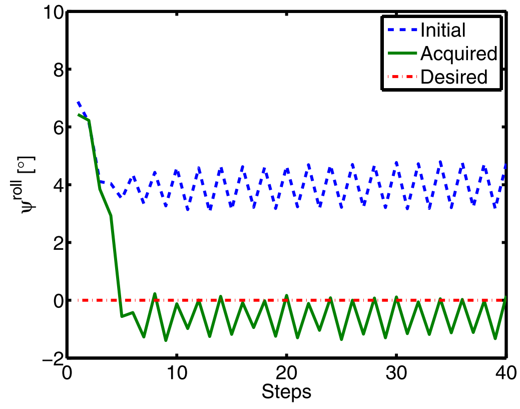

**Fig. 14** The roll angle $\psi^{roll}$ at the Poincaré section $\dot{\psi}^{roll} = 0$ (*solid line*). The *dash-dotted line* represents the desired angle for this stepping task. We used a policy acquired by the proposed learning method

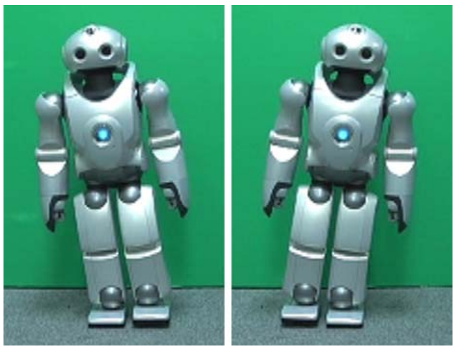

**Fig. 15** Acquired stepping movement of the real robot after one iteration of the proposed learning process. One iteration means that 1) acquisition of an approximated Poincaré map from sampled real data as in Fig. 5. 2) acquisition of a stepping policy using the acquired Poincaré map as in Fig. 6. 3) application of the acquired policy to the real robot

roll state $\psi^{roll}$ around the desired angle (0.0°). An average angle from 10 to 40 steps was 0.12°.

Figure 15 shows the acquired stepping movement of the real robot after one iteration of the proposed learning process.

Again, one iteration means that 1) acquisition of an approximated Poincaré map from sampled real data as in Fig. 5. 2) acquisition of a stepping policy using the acquired Poincaré map as in Fig. 6. 3) application of the acquired policy to the real robot.

### 7.2 Improvement of biped walking performance

We also applied our method to improve biped walking performance of the humanoid robot. We use $\mathbf{x} = (\psi^{roll}, \psi^{pitch}, \dot{\psi}^{pitch})$ as the state and modulate $A_{walk}$ in (6) as the action of the learning system.

We use the reward function based on the pitch derivative $\dot{\psi}^{pitch}$ that corresponds to the walking velocity:

$$r = \frac{1}{T}\int_0^T \dot{\psi}^{pitch}(t)dt \tag{31}$$

for this walking task, where $T$ is the time required for moving from a Poincaré section to the next section, which is approximately the time for one cycle of walking. We used the time integral of the angular velocity $\dot{\psi}^{pitch}$ rather than the instantaneous angular velocity at the defined section because the instantaneous value was not consistently related to the walking velocity in the real environment. Since this reward can not be simply represented by the state $\mathbf{x}$ at the section, we also approximated a stochastic model of the reward function from the data acquired from the real environment by using another Gaussian process. The learning system also receives a negative reward $r = -1$ if the biped model falls over.

For the Gaussian process model, we sampled 100 data sets from the real environments. Figure 16 shows initial performance of the biped walking controller. The initial policy only generates random steps and the humanoid robot could not walk forward.

Figure 17 shows improved walking performance after the first iteration of our learning framework. The humanoid robot could walk forward with the improved policy.

Figure 18 shows improved walking performance after the second iteration of our learning framework. The humanoid robot could walk faster. We also tried additional learning iterations. However, the walking performance did not significantly improve. We may need to use different state representations and/or reward functions to acquire additional improvement of the walking policy.

## 8 Discussion

### 8.1 Stochastic Poincaré maps

In this study, we used a Gaussian process model to represent Poincaré maps. We determined the number of sampled data to represent the covariance matrix in the Gaussian process model so that the prediction error of the state at the next Poincaré section becomes smaller than a given threshold.

Sparse Gaussian process methods (Smola and Bartlett 2001; Snelson and Ghahramani 2006; Candela and Rasmussen 2005) can provide a method to find a sufficient number of data points to model the Poincaré maps.

By using a Gaussian process model, we can stochastically represent accuracy of the Poincaré maps based on the density of the sampled data.

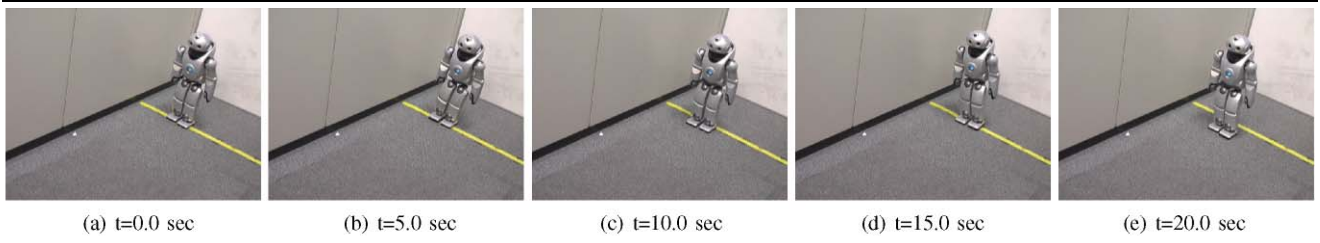

**Fig. 16** Initial walking pattern. The *line on the floor* represents the starting position. Initially, the humanoid robot explores around the starting position. Time proceeds from *left* to *right*

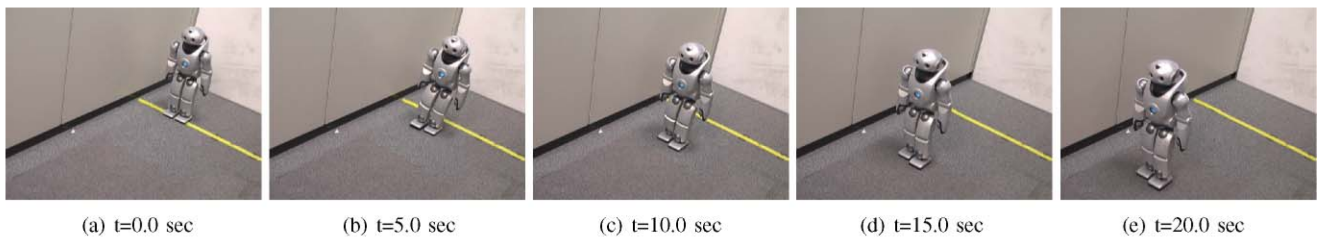

**Fig. 17** Improved walking pattern after the first iteration of the proposed learning process. Estimated walking speed is $2.7 \times 10^{-2}$ m/sec

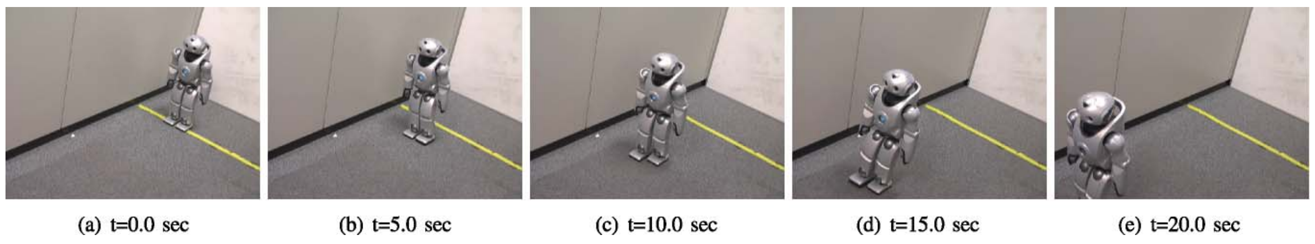

**Fig. 18** Improved walking pattern after the second iteration of the proposed learning process. Estimated walking speed is $4.5 \times 10^{-2}$ m/sec. The robot could walk faster with this policy than after the first iteration

In Byl and Tedrake (2008), a stochastic return map is used to represent uncertainty of the ground contact conditions. The stochastic stability when a compass-gait is walking on randomly generated terrain is studied. As Byl and Tedrake (2008) pointed out, stochastic representation of the walking dynamics can be useful to acquire robust walking policies.

## 8.2 Application to other walking controllers

We applied the proposed learning method to control the amplitude of phase dependent sinusoidal patterns where the phase is modulated according to the center of pressure.

However, application of the learning method is not necessarily limited to the periodic pattern generator model (Morimoto et al. 2006, 2008).

Several biped control methods have been proposed and applied to humanoid robots (Hirai et al. 1998; Kajita et al. 2007; Nagasaka et al. 1999, 2004; Sugihara and Nakamura 2005). Most of them use ZMP-based walking controllers. These control methods rely on accuracy of the robot model and tracking performance of high-gain PD controllers. Since the ZMP-based controllers also use the inverted pendulum model to approximate the full-body dynamics, modeling errors are inevitable and feedback controllers to change desired ZMP are necessary. To design these feedback controllers, parameters need to be hand-tuned through experiments. Our proposed learning method may potentially help to tune parameters of the feedback controllers for ZMP-based approaches.

## 8.3 Selection of reward functions

For the stepping task, the reward function is defined by a squared error from a desired COM roll angle. For the walking task, the reward is given according to walking velocity. In both tasks, negative reward is given when a biped robot falls over. Since controlling the single support phase is the goal for the stepping task, using COM and COP states is a reasonable selection. For the walking task, moving forward at a desired speed is one of the goals, and we can represent this goal by including walking velocity in the reward function.

Other reward functions can be adopted for the stepping and the biped walking task. If we can measure energy consumption of real biped robots, we can consider the energy cost as a negative reward (penalty). Similarity between walking patterns of biped robots and human walking patterns could be used to acquire a policy to generate natural looking walking patterns. Ground reaction force can be used as a negative reward to reduce impact forces when the swing leg contacts the ground.

## 9 Conclusion

We propose using a nonparametric representation of approximated Poincaré maps for biped stepping and walking dynamics and reinforcement learning (RL) to improve task performance. In this study, we first approximate Poincaré maps for stepping and walking dynamics by using data from a simulated model or a real robot, then use the approximated maps for RL to improve stepping and walking policies. We explored using a Gaussian process to approximate the Poincaré maps. By using a Gaussian process, we can estimate a probability distribution of the target nonlinear functions with a given covariance. We showed that we could improve stepping and walking policies by using a RL method with approximated models both in simulated and real environments. Since we used different reward for the stepping and walking tasks, the results showed that our learning framework can improve policies for different objective functions. We applied the proposed control approach to a small humanoid robot.

We evaluated robustness of the resulting stepping policy by pushing the biped model in the horizontal direction and analyzing the return map. We also showed robustness of the resulting walking policy by applying the policy to a real environment. However, by only using one fixed policy, it is difficult to cope with a large environmental change. Therefore, it would be interesting to evaluate how our learning algorithm can quickly adapt to environmental changes.

Compared to other biped learning approaches (e.g., Benbrahim and Franklin 1997; Tedrake et al. 2004; Morimoto and Atkeson 2007), we do not directly use real environments to improve biped walking policies. Therefore, our learning framework requires smaller number of samples from the real environment and can be applied to robots that have may degrees of freedom. On the other hand, our learning method relies on environmental models. We take the reliability of an acquired model into account in a policy improvement process by representing the model with a probability distribution. The estimated reliability of the acquired model can be related to learning performance. We will analyze this relationship in a future study.

In this study, we use a reinforcement learning method (Kimura and Kobayashi 1998) to improve policy parameters. In future work, we will consider using a dynamic programming approach to efficiently update policy parameters using the Gaussian process model (Rasmussen and Kuss 2004) since an analytical update using dynamic programming may reduce the number of iterations to achieve better task performance.

We will also explore convergence properties of the proposed learning method.

**Acknowledgements** We thank Sony Corp. for allowing us the opportunity to use the small humanoid robot. We thank Gen Endo for helping us to setup experiments. We also thank Gordon Cheng for helpful discussions. This material is based upon work supported in part by the National Science Foundation under grants ECS-0325383, EEC-0540865, and ECS-0824077.

## References

Abbeel, P., Quigley, M., & Ng, A. Y. (2006). Using inaccurate models in reinforcement learning. In *Proceedings of the 23rd international conference on machine learning* (pp. 1–8). New York: ACM.

Atkeson, C. G. (1998). Nonparametric model-based reinforcement learning. In M.I. Jordan, M. Kearns & S. Solla (Eds.), *Advances in neural information processing systems 10* (pp. 1008–1014). Cambridge: MIT.

Atkeson, C. G., & Schaal, S. (1997). Robot learning from demonstration. In *Proc. 14th international conference on machine learning* (pp. 12–20). San Mateo: Morgan Kaufmann.

Bagnell, A., & Schneider, J. (2003). Covariant policy search. In *Proceedings of the eighteenth international joint conference on artificial intelligence* (pp. 1019–1024).

Baird, L. C., & Moore, A. W. (1999). Gradient descent for general reinforcement learning. In *Advances in neural information processing systems 11* (pp. 968–974). Cambridge: MIT.

Benbrahim, H., & Franklin, J. (1997). Biped dynamic walking using reinforcement learning. *Robotics and Autonomous Systems*, *22*, 283–302.

Bishop, C. M. (2006). *Pattern recognition and machine learning*. New York: Springer.

Byl, K., & Tedrake, R. (2008). Metastable walking on stochastically rough terrain. In *Proceedings of robotics: science and systems IV*, Zurich, Switzerland, June 2008.

Candela, J. Q., & Rasmussen, C. E. (2005). A unifying view of sparse approximate Gaussian process regression. *Journal of Machine Learning Research*, *6*, 1939–1959.

Dearden, R., Friedman, N., & Andre, D. (1999). Model based Bayesian exploration. In *Proceedings of fifteenth conference on uncertainty in artificial intelligence* (pp. 457–464). San Francisco: Morgan Kaufmann.

der Linde, R. Q. V. (1999). Passive bipedal walking with phasic muscle contraction. *Biological Cybernetics*, *82*, 227–237.

Doya, K. (2000). Reinforcement learning in continuous time and space. *Neural Computation*, *12*(1), 219–245.

Endo, G., Morimoto, J., Matsubara, T., Nakanishi, J., & Cheng, G. (2008). Learning CPG-based biped locomotion with a policy gradient method: application to a humanoid robot. *International Journal of Robotics Research*, *27*(2), 213–228.

Ghavamzadeh, M., & Engel, Y. (2007). Bayesian policy gradient algorithms. In B. Scholkopf, J. Platt & T. Hofmann (Eds.), *Advances in neural information processing systems 19* (pp. 457–464). Cambridge: MIT.

Hirai, K., Hirose, M., & Takenaka, T. (1998). The development of Honda humanoid robot. In *Proceedings of the 1998 IEEE international conference on robotics and automation* (pp. 160–165).

Howard, M., Klanke, S., Gienger, M., Goerick, C., & Vijayakumar, S. (2009). A novel method for learning policies from variable constraint data. *Autonomous Robots* (same special issue, Part B).

Hyon, S., Hale, J. G., & Cheng, G. (2007). Full-body compliant human-humanoid interaction: Balancing in the presence of unknown external forces. *IEEE Transactions on Robotics*, *23*(5), 884–898.

Jaakkola, T., Singh, S. P., & Jordan, M. I. (1995). Reinforcement learning algorithm for partially observable Markov decision problems. In G. Tesauro, D. Touretzky & T. Leen (Eds.), *Advances in neural information processing systems 7* (pp. 345–352). Cambridge: MIT.

Kajita, S., Nagasaki, T., Kaneko, K., & Hirukawa, H. (2007). ZMP-based biped running control. *Robotics and Automation Magazine, IEEE*, *14*(2), 63–72.

Kakade, S. (2002). A natural policy gradient. In *Advances in neural information processing systems 14* (pp. 1531–1536). Cambridge: MIT.

Kimura, H., & Kobayashi, S. (1998). An analysis of actor/critic algorithms using eligibility traces: reinforcement learning with imperfect value functions. In *Proceedings of the 15th int. conf. on machine learning* (pp. 284–292).

Ko, J., & Fox, D. (2009). GP-BayesFilters: Bayesian filtering using Gaussian process prediction and observation models. *Autonomous Robots* (same special issue, Part A).

Konda, V. R., & Tsitsiklis, J. N. (2003). Actor-critic algorithms. *SIAM Journal on Control and Optimization*, *42*(4), 1143–1166.

Kuvayev, L., & Sutton, R. (1996). Model-based reinforcement learning with an approximate, learned model. In *Proceedings of the ninth Yale workshop on adaptive and learning systems* (pp. 101–105).

Matsubara, T., Morimoto, J., Nakanishi, J., Sato, M., & Doya, K. (2006). Learning CPG-based biped locomotion with a policy gradient method. *Robotics and Autonomous Systems*, *54*(11), 911–920.

McGeer, T. (1990). Passive dynamic walking. *International Journal of Robotics Research*, *9*(2), 62–82.

Meuleau, N., Kim, K. E., & Kaelbling, L. P. (2001). Exploration in gradient-based reinforcement learning. Technical report, AI Memo 2001-003, MIT.

Miura, H., & Shimoyama, I. (1984). Dynamical walk of biped locomotion. *International Journal of Robotics Research*, *3*(2), 60–74.

Miyazaki, F., & Arimoto, S. (1981). Implementation of a hierarchical control for biped locomotion. In *8th IFAC* (pp. 43–48).

Morimoto, J., & Atkeson, C. G. (2007). Learning biped locomotion: application of Poincaré-map-based reinforcement learning. *IEEE Robotics and Automation Magazine*, *14*(2), 41–51.

Morimoto, J., & Doya, K. (2001). Acquisition of stand-up behavior by a real robot using hierarchical reinforcement learning. *Robotics and Autonomous Systems*, *36*, 37–51.

Morimoto, J., Endo, G., Nakanishi, J., Hyon, S., Cheng, G., Atkeson, C. G., & Bentivegna, D. (2006). Modulation of simple sinusoidal patterns by a coupled oscillator model for biped walking. In *Proceedings of the 2006 IEEE international conference on robotics and automation* (pp. 1579–1584).

Morimoto, J., Endo, G., Nakanish, J., & Cheng, G. (2008). A biologically inspired biped locomotion strategy for humanoid robots: modulation of sinusoidal patterns by a coupled oscillator model. *IEEE Transaction on Robotics*, *24*(1), 185–191.

Nagasaka, K., Inaba, M., & Inoue, H. (1999). Stabilization of dynamic walk on a humanoid using torso position compliance control. In *Proceedings of 17th annual conference on robotics society of Japan* (pp. 1193–1194).

Nagasaka, K., Kuroki, Y., Suzuki, S., Itoh, Y., & Yamaguchi, J. (2004). Integrated motion control for walking, jumping and running on a small bipedal entertainment robot. In *Proceedings of IEEE 2004 international conference on robotics and automation* (pp. 3189–3194). New Orleans, LA, USA.

Peters, J., & Schaal, S. (2008). Reinforcement learning of motor skills with policy gradients. *Neural Networks*, *21*(4), 682–697.

Peters, J., & Schaal, S. (2008). Natural actor-critic. *Neurocomputing*, *71*(7–9), 1180–1190.

Rasmussen, C. E., & Kuss, M. (2004). Gaussian processes in reinforcement learning. In *Advances in neural information processing systems* (vol. *16*, pp. 751–759). Cambridge: MIT.

Rasmussen, C. E., & Williams, C. K. I. (2006). *Gaussian processes for machine learning*. Cambridge: MIT.

Riedmiller, M., Gablel, R. H. T., & Lange, S. (2009). Reinforcement learning for robot soccer. *Autonomous Robots* (same special issue, Part A).

Shiriaev, A., Robertsson, A., Perram, J., & Sandberg, A. (2005). Periodic motion planning for virtually constrained (hybrid) mechanical systems. In *Proceedings of IEEE conference on decision and control* (pp. 4035–4040).

Smola, J., & Bartlett, P. (2001). Sparse greedy Gaussian process regression. In T. G. Diettrich & V. Tresp (Eds.), *Advances in neural information processing systems 13* (pp. 619–625). Cambridge: MIT.

Snelson, E., & Ghahramani, Z. (2006). Sparse Gaussian processes using pseudo-inputs. In Y. Weiss, B. Scholkof & J. Platt (Eds.), *Advances in neural information processing systems 18* (pp. 1257–1264). Cambridge: MIT.

Strogatz, S. H. (1994). *Nonlinear dynamics and chaos*. Reading: Addison-Wesley.

Sugihara, T., & Nakamura, Y. (2002). Whole-body cooperative COG control through ZMP manipulation for humanoid robots. In *IEEE int. conf. on robotics and automation*, Washington DC, USA, 2002.

Sugihara, T., & Nakamura, Y. (2005). A fast online gait planning with boundary condition relaxation for humanoid robots. In *IEEE int. conf. on robotics and automation* (pp. 306–311). Barcelona, Spain.

Sutton, R. S., & Barto, A. G. (1998). *Reinforcement learning: an introduction*. Cambridge: MIT.

Sutton, R. S., McAllester, D., Singh, S., & Mansour, Y. (2000). Policy gradient methods for reinforcement learning with function approximation. In *Advances in neural information processing systems 12* (pp. 1057–1063). Cambridge: MIT.

Tedrake, R., Zhang, T. W., & Seung, H. S. (2004). Stochastic policy gradient reinforcement learning on a simple 3D biped. In *Proceedings of the 2004 IEEE/RSJ international conference on intelligent robots and systems* (pp. 2849–2854).

Tsuchiya, K., Aoi, S., & Tsujita, K. (2003). Locomotion control of a biped locomotion robot using nonlinear oscillators. In *Proceedings of the IEEE/RSJ international conference on intelligent robots and systems* (pp. 1745–1750). Las Vegas, NV, USA.

Westervelt, E. R., Buche, G., & Grizzle, J. W. (2004). Experimental validation of a framework for the design of controllers that induce stable walking in planar bipeds. *International Journal of Robotics Research*, *23*(6), 559–582.

Williams, C. K. I., & Rasmussen, C. E. (1996). Gaussian processes for regression. In *Advances in neural information processing systems* (vol. *8*, pp. 514–520). Cambridge: MIT.

**Jun Morimoto** is head of Department of Brain Robot Interface at ATR Computational Neuroscience Laboratories and the group leader of Humanoid Motor Learning Group in Computational Brain Project, ICORP, JST. He received his Ph.D. in information science from Nara Institute of Science and Technology (NAIST), Nara, Japan, in 2001. He was a Research Assistant with Kawato Dynamic Brain Project, ERATO, JST, from 1999 to 2001. From 2001 to 2002, he was a postdoctoral fellow at the Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, USA. He Jointed ATR in 2002, and then joined JST, ICORP in 2004.

**Christopher G. Atkeson** is a professor in the Robotics Institute and Human–Computer Interaction Institute at Carnegie Mellon University. His research focuses on humanoid robotics and robot learning, using challenging dynamic tasks such as juggling. Specific research interests include nonparametric learning, memory-based learning including approaches based on trajectory libraries, reinforcement learning and other forms of learning based on optimal control, learning from demonstration, and modeling human behavior. He received the M.S. degree in applied mathematics (computer science) from Harvard University and the Ph.D. degree in brain and cognitive science from MIT. He joined the MIT faculty in 1986, moved to the Georgia Institute of technology College of Computing in 1994, and then joined CMU in 2000. He has received an NSF Presidential Young Investigator Award, a Sloan Research Fellowship, and a Teaching Award from the MIT Graduate Student Council.