Adaptive Control of a Looper-like Robot based on the CPG-Actor-Critic Method

Kenji Makino* Yutaka Nakamura† Tomohiro Shibata* Shin Ishii*

∗ Graduate School of Information Science
Nara Institute of Science and Technology (NAIST)
8916-5 Takayama-cho, Ikoma, Nara 630-0192, Japan
† Graduate School of Engineering, Osaka University
Yamadaoka 2-1, Suita, Osaka, 565-0871, Japan

Abstract

Adaptability to the environment is crucial for mobile robots, because the circumstance, including the body of the robot, may change. A robot with a large number of degrees of freedom possesses potential to adapt to such circumstances, but it is difficult to design a good controller for such a robot. We previously proposed a reinforcement learning (RL) method called the CPG-actor-critic method, and applied it to automatic acquisition of vermicular locomotion of a looper-like robot through computer simulations. In this study, we developed a looper-like robot and apply our RL method to the control of this robot. Experimental results demonstrate fast acquisition of a vermicular forward motion, supporting the real applicability of our method.

1 introduction

To realize a mobile robot that acts in a real environment, adaptability to changes in the environment due to its dynamic nature and various disturbances is necessary. Reinforcement Learning (RL) is a framework for autonomous acquisition of control rules and has been successfully applied to various automatic control problems [1]. Because real robots often have a large number of degrees of freedom (DOF), RL methods to control them need some devices to avoid the problem of "curse of dimensionality".

Motivated by the animal's control mechanism for rhythmic locomotion, which is induced by neural circuits in the spinal cord of vertebrates called central pattern generators (CPGs), robot control schemes using a CPG controller have been studied mainly in the field of robotics [2][3]. Because the parameter of the CPG controller is designed such that the CPG controller and the robot interact with each other and are eventually entrained into a stable limit-cycle attractor, the robot controlled by a CPG controller is robust against possible disturbances from the environment.

Although there have been some studies of designing a CPG controller, autonomous learning framework for a CPG controller is necessary to realize the adaptability to dynamic nature of the environment including its own body. We formerly proposed an RL framework called the CPG-actor-critic model for designing a CPG controller [4]. Since control signals are restricted to be rhythmic in favor of the CPG, this RL method would be able to avoid "curse of dimensionality". In this method, the parameter of the CPG controller is updated according to the gradient of the performance indicator, the average reward per step for example, with respect to the parameter (policy gradient) and this gradient can be obtained by interaction with the environment. After learning, the CPG controller becomes to generate stable locomotion suited to the environment surrounding the robot.

In the current study, we configure a real looper-like robot and its CPG controller. Because a looper has an ability to move by many of simple and rhythmic telescopic motions, a looper-like robot would have an adaptability to various environments. In our previous study, we showed through computer simulations that a good CPG controller can be obtained by the CPG-actor-critic method [5]. In real environments, however, there may arise much difficulty; e.g., it is difficult to approximate contact between the robot and ground, outputs of sensors generally include unknown noise and time delays. In this study, we apply the CPG-actor-critic method to the real looper-like robot we have developed, for achieving automatic acquisition of "vermicular" locomotion.

2 Robot System

Looper-like robot

The looper-like robot we have developed is depicted in Fig.1. This robot is composed by three links, eight actuators and six passive wheels. The first to fourth actuators are linear actuators which are each located between two links, and used to expand-and-contract the body of the robot. We call these actuators **body actuators**. The length of these actuators is about 13cm at minimum and about 18cm at maximum. Black semicircles in Fig.1 denote the ball joint, so that the body of the robot bends when lengths of these actuators are different from one another. The fifth to eighth actuators are also linear actuators which are located to the head or tail link. They move vertically to clamp corresponding links to the ground. We call these actuators **leg actuators**.

In order to make the robot move forward, it is required to repeat expand-and-contract motions, by clamping the head link and the tail link in an appropriate but different timing.

CPG controller

Because a looper-like robot has a substantial potential to move by a rhythmic locomotion, we employ a CPG controller [5] which outputs rhythmic control signals. As shown in Fig. 2, the CPG controller comprises three neural oscillators each of which consists of two neurons, and each actuator is controlled by a single neural oscillator. The $(2i-1)$-th and the $2i$-th neuron's dynamics are defined as

$$\begin{aligned} \tfrac{1}{c}\dot{y}_{2i-1} &= -y_{2i-1} + \tanh(W^s y_{2i-1} + W^I y_{2i} + u_i) \\ \tfrac{1}{c}\dot{y}_{2i} &= -y_{2i} + \tanh(-W^I y_{2i-1} + W^s y_{2i}) \end{aligned}, \quad (1)$$

where y_{2i-1} and y_{2i} denote the $(2i-1)$-th and the $2i$-th neuron's states, respectively. W^S, W^I and c are the self-excitatory connection weight, the mutual inhibitory connection weight and the time constant, respectively. u_i is an input to the i-th neural oscillator, in this study, calculated as the weighted sum of the robot's state variable x and the output of the CPG controller y: as $u_i = \theta_j y_0 + \cdots + \theta_k x_0 + \cdots$. Each neural oscillator outputs sinusoidal control wave whose amplitude and frequency are mainly determined by W^S and W^I, and c, respectively. The phase relation between the neural oscillators and the robot are determined by the input to the CPG neurons $\mathbf{u}$, i.e., the phase can be tuned by changing the weight parameter $\boldsymbol{\theta}$ of the CPG. In this study, this weight parameter $\boldsymbol{\theta}$ is trained by the CPG-actor-critic method (see below).

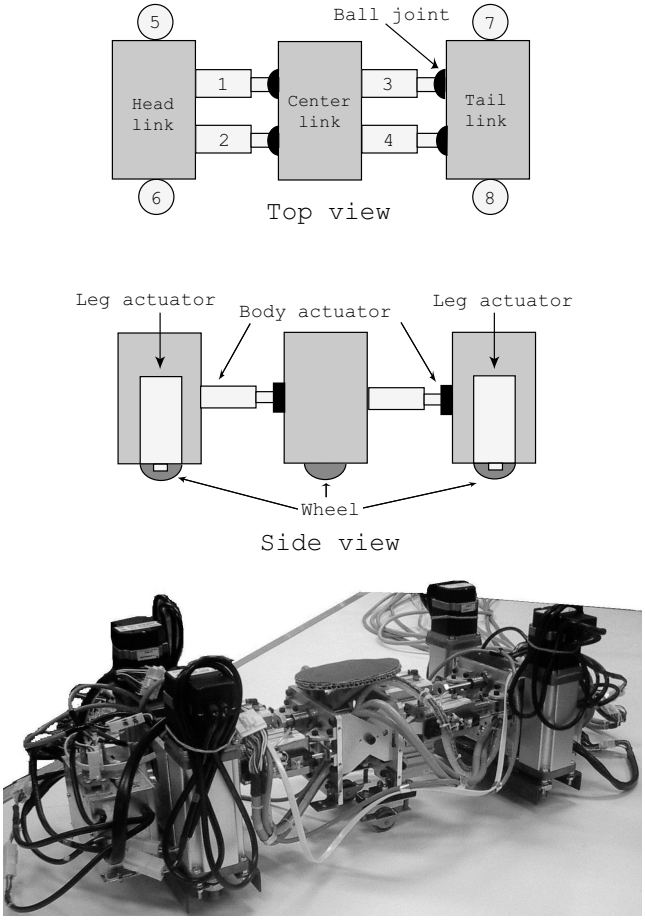

Figure 1: Looper-like robot

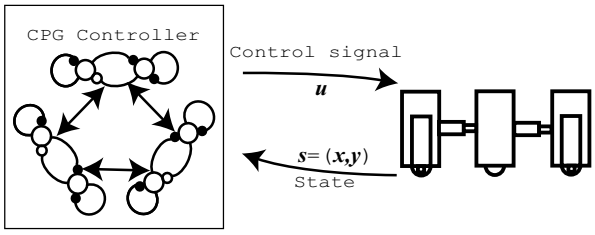

Figure 2: Control scheme using a CPG controller

3 Learning method

Here, we describe our RL algorithm for the CPG controller, which we formerly proposed and called the CPG-actor-critic model [5]. In this controlling by a CPG controller, the control signal depends not only on the state of the target system, but also its own state, because the CPG controller has its own dynamics. This is problematic because most RL algorithms assume the target policy is stationary (time-

independent), and furthermore, heavy computation would be required for training recurrent neural networks like the naive CPG controller. In order to overcome these difficulties, the CPG controller is divided into two parts, the basic CPG and the actor [4]. The basic CPG is a part of the CPG controller with fixed connection parameters, i.e., W^I , W^S, and c are fixed. We treat the physical system and the basic CPG as a single dynamical system to control, and we call this system a CPG-coupled system.

Since the actor turns out to be a feed-forward neural network having no its own dynamics in the CPG-coupled system, we can easily apply usual RL algorithms. The control signal $\mathbf{u}$ for the CPG-coupled system is conceptually represented as

$$\mathbf{u} \sim \pi(\mathbf{u}, \mathbf{s}), \tag{2}$$

where π denotes the control policy of the actor and $\mathbf{s} \equiv (\mathbf{x}, \mathbf{y})$ is a state of the CPG-coupled system. For the sake of simplicity, we assume that Eqs. (1) and (2) are discretized in time by an appropriate method, and the learning system receives an immediate reward $r(\mathbf{s}(t), \mathbf{u}(t))$ at a discrete time step t. The policy $\pi_{\boldsymbol{\theta}}(\mathbf{s}, \mathbf{u})$ is defined by a parametric stochastic policy, i.e., the probability of a control signal $\mathbf{u}$ at a state $\mathbf{s}$ is given by $p(\mathbf{u}|\mathbf{s}; \boldsymbol{\theta})$, where $\boldsymbol{\theta}$ is a parameter vector of the policy. The objective of RL here is to obtain the policy parameter that maximizes the expected reward accumulation defined by $\rho(\boldsymbol{\theta}) \equiv \mathrm{E}_{\boldsymbol{\theta}}[\sum_t \gamma^{t-1} r(\mathbf{s}(t), \mathbf{u}(t))]$, where $\gamma \in (0, 1]$ is a discount factor. The partial differential of $\rho(\boldsymbol{\theta})$ with respect to the policy parameter θ_i is calculated [6][7] as

$$\frac{\partial \rho(\boldsymbol{\theta})}{\partial \theta_i} = \langle \psi_i(\mathbf{s}, \mathbf{u}) Q_\theta(\mathbf{s}, \mathbf{u}) \rangle, \tag{3}$$

where $\psi_i(\mathbf{s}, \mathbf{u}) \equiv \frac{\partial}{\partial \theta_i} \ln \pi_{\boldsymbol{\theta}}$ and $Q_{\boldsymbol{\theta}}(\mathbf{s}, \mathbf{u})$ denotes the action-value function (Q-function). $\langle \cdot \rangle$ stands for the expectation with respect to the stationary distribution of the state-action pair $(\mathbf{s}, \mathbf{u})$. When the Q-function is approximated by a weighted sum of base functions $\boldsymbol{\psi}$: $Q^w_{\boldsymbol{\theta}}((\mathbf{s}, \mathbf{u})) \equiv \sum_i w_i \psi_i(\mathbf{s}, \mathbf{u})$, where $\mathbf{w}$ is the weight vector of the approximate Q-function, the optimal weight in the least square sense, $\tilde{\mathbf{w}} = \arg\min_{\mathbf{w}} \langle (Q_{\boldsymbol{\theta}}(\mathbf{s}, \mathbf{u}) - Q^w_{\boldsymbol{\theta}}(\mathbf{s}, \mathbf{u}))^2 \rangle$, provides the natural policy gradient with no estimation bias for the gradient, so that the policy parameter can be updated [1] as

$$\theta_i := \theta_i + \eta \tilde{w}_i, \tag{4}$$

where η is the learning rate. the optimal weight $\tilde{\mathbf{w}}$ is thus estimated simultaneously based on the least square method [4][1].

4 Experiment

Setup

The aim of the experiment have is to examine if our RL can be applied to a real looper-like robot, we then conducted an experiment attempt to obtain a controller which allows the looper-like robot to move in the forward direction. For the simplicity of the experiment setting, all **body actuators** were controlled by a single neural oscillator, and front **leg actuators** (5-th and 6-th actuators) and hinder **leg actuators** (7-th and 8-th actuators) were controlled by a single neural oscillators, respectively. The output signal $\boldsymbol{\tau}$ to these actuators is defined by

$$\tau_i = \begin{cases} 1, & y_{2i-1} > 0 \\ 0, & \text{otherwise} \end{cases} .$$

When $\tau_i = 1(0)$, the corresponding actuator expands (contracts). The control signal $\mathbf{u}$ to the CPG-coupled system is defined by

$$\begin{aligned} u_1 &= \theta_1 X_1 + \theta_2 X_2 + \epsilon_1 \\ u_2 &= \theta_3 y_1 + \theta_4 y_2 + \epsilon_2 \\ u_3 &= \theta_5 y_1 + \theta_6 y_2 + \epsilon_3 \end{aligned} , \tag{5}$$

where $\mathbf{X} = \{x_2 - x_3 - l, \dot{x}_2 - \dot{x}_3\}$, x_2 and x_3 denote the position of the head link and that of the tail link, respectively. $\epsilon_i (i = 1, 2, 3)$ is a small random noise obeying a normal distribution. l denotes the mean length between the head link and the tail link. The policy parameter $\boldsymbol{\theta}$ determines phase relations among the robot and the neural oscillators and was adjusted by RL. W^S, W^I , and c which are CPG system parameters were fixed at 1.1, 0.7, and 4.0, respectively.

The immediate reward $r(\mathbf{s}(t))$ was given by

$$r(\mathbf{s}(t)) = \dot{x}_1,$$

where x_1 denote the position of the center link. Because the robot currently possesses no sensors to measure its position, an USB camera is placed above the experimental field.

We chose the state representation for RL as follows. Because the phase portrait between two neurons presents a circle whose center is the origin, the angle $\omega_1 = \arctan(y_2/y_1)$ carries the essential feature of the first neural oscillator. Similarly, $\omega_2 = \arctan(y_4/y_3)$ and $\omega_3 = \arctan(y_6/y_5)$ should be useful features of the second and the third neural oscillators, respectively. Furthermore, when the looper-like robot repeats expansion and contraction, $\omega_4 = \arctan((\dot{x}_3 - \dot{x}_2)/(x_3 - x_2))$ is also an important feature of the robot's movements. Because the phase difference seemed to be more important than the phase

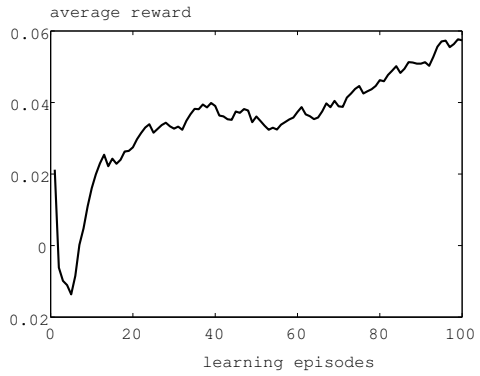

Figure 3: Learning curve. The horizontal axis denotes the number of learning episodes, and the vertical axis denotes the average reward in one episode. The line shows the moving average over 30 episodes.

in this experiment, we employed 49 basis functions for the approximate state-value function: $\phi_{16(i-2)+j}(\mathbf{s}) = \exp(-3\cos(\omega_i - \omega_1 + \pi/16j) - 1)$ for $i = 2, 3, 4$ and $j = 1, 2, ..., 16$ and $\phi_{49} = 1$. At the beginning of an episode, the learning parameter was initialized at random: $\boldsymbol{\theta} = \{0.0006, 0.0001, 0.0012, -0.0005, -0.0004, 0.0011\}$. In each episode during the training, the robot was controlled by the current actor for 45 sec.

Result

Fig.4 shows the learning curve, indicating that appropriate control was achieved after about 10 training episodes. Fig.5 shows actual movements of the looper-like robot before learning and after learning(after 100 learning episodes). Before learning, the robot accidentally moved backward first, and then started to move forward slowly. After learning in contrast, the robot became to move forward much faster. After the learning 100 episodes, the CPG parameters grew as: $\boldsymbol{\theta} = \{0.0007, -0.0064, -0.0027, -0.0011, -0.0102, 0.0017\}$.

5 Conclusion

In this study, we have configured a CPG-based control architecture for a real looper-like robot, and applied RL to obtain a controller for the robot to move in the forward direction. The experiments showed that a good CPG controller for this robot can be efficiently obtained by our RL method. Applying our RL method to many other situations including changes in the ground condition and changes in its body, for example, is our future work.

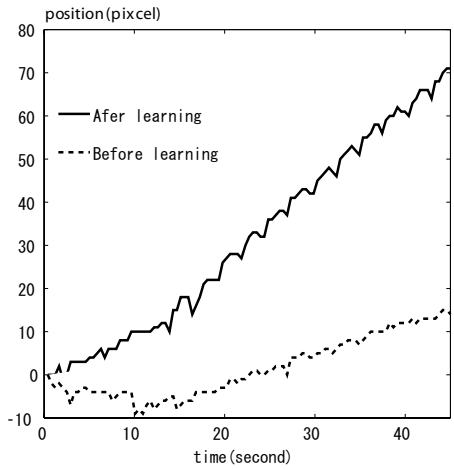

Figure 4: Control result, showing the location of the position of the robot's center.

References

[1] J. Peters, S. Vijayakumar and S. Schaal: "Reinforcement learning for humanoid robotics", Third IEEE International Conference on Humanoid Robotics 2003, Germany (2003).

[2] G. Taga, Y. Yamaguchi and H. Shimizu: "Selforganized control of bipedal locomotion by neural oscillators in unpredictable environment", Biological Cybernetics, 65, pp. 147.159 (1991).

[3] Y. Fukuoka, H. Kimura and A. H. Cohen: "Adaptive dynamic walking of a quadruped robot on irregular terrain based on biological concepts", International Journal of Robotics Research, 22, 3.4, pp. 187.202 (2003).

[4] Y. Nakamura, T. Mori and S. Ishii: International conference on parallel problem solving from nature (PPSN VIII), pp. 972.981 (2004).

[5] Y. Nakamura, T. Mori, S. Ishii: "Natural policy gradient reinforcement learning method for a looper-like robot", International Symposium on Artificial Life and Robotics (AROB 11th '06), GS3-3. (2006)

[6] V. R. Konda and J. N. Tsitsiklis: "Actor-critic algorithms", SIAM Journal on Control and Optimization, 42, 4, pp. 1143.1146 (2003).

[7] R. S. Sutton, D. McAllester, S. Singh and Y. Manour: "Policy gradient method for reinforcement learning with function approximation", Advances in Neural Information Processing Systems, Vol. 12, pp. 1057.1063 (2000).