2004 International Symposium on Nonlinear
Theory and its Applications (NOLTA2004)
Fukuoka, Japan, Nov. 29 - Dec. 3, 2004

# Reinforcement Learning for a Snake-Like Robot

Shuichi Fukunaga†, Yutaka Nakamura‡†, Kazuaki Aso§ and Shin Ishii†‡

†Nara Institute of Science and Technology
8916-5 Takayama-cho, Ikoma, Nara 630-0912, Japan
‡CREST, Japan Science and Technology Agency
§Future Project Division, Toyota Motor Corporation
Email: {shui-f, yutak-na, ishii}@is.naist.jp, aso@ahs.tec.toyota.co.jp

**Abstract**—A snake has large potential to move in various environments by drastically changing its 'gait' pattern, in spite of its simple body. We configured a control scheme for a snake-like robot, using a CPG controller, and developed a learning algorithm to acquire a good control rule in a changing environment. Although the snake-like robot has a large degree of freedom, a computer simulation showed that a control rule that allows the robot to move to a target direction is obtained by our scheme.

## 1. Introduction

There have been a lot of studies of animal locomotion. A snake has large potential to move in various environments by changing its 'gait' pattern drastically according to its environment, in spite of its simple body. Hirose formerly revealed the locomotion mechanism of snakes [4]. The propulsion of a snake is generated by utilizing the difference in the friction coefficients between in the normal-direction and in the tangential-direction. The magnitude of serpentine curvature depends on the friction coefficient.

On the other hand, neurobiological studies have revealed that various rhythmic motor patterns are controlled by neural oscillators referred as central pattern generators (CPGs) [1]. Recently, various control schemes using CPG controllers have been presented; for example, a biped robot [10], a quadruped robot [3] and a salamander [5]. Among those, Ekeberg studied lamprey's locomotion controlled by a CPG, and performed computer simulations [2].

In this study, we configure a control scheme for a snake-like robot, using a CPG controller, and develop a learning algorithm to acquire a good control rule autonomously in a changing environment. Hirose showed that a snake-like robot can be controlled by relatively simple rhythmic control signals. Since lamprey's locomotion is similar to that of a snake, we introduce the Ekeberg's CPG controller to our scheme.

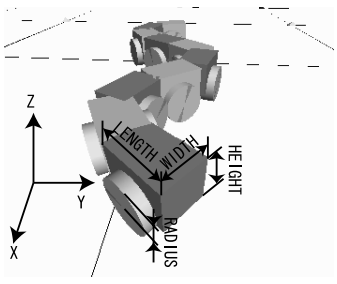
We formerly proposed a reinforcement learning (RL) framework called a CPG-actor-critic model and applied it to the acquisition problem of biped locomotion [7]. RL schemes often suffer from the curse of dimensionality, if the controlled system has a large degree of freedom (DOF). Although a snake-like robot has a larger DOF than a human lower body, it is a redundant system, hence it is useful for RL to utilize such a feature.

## 2. Control scheme for a snake-like robot

### 2.1. Mechanical model

Hirose formerly revealed the locomotion mechanism of snakes [4]. A force applied to a joint between two body compartments can be divided into two elements, one is in parallel with its body axis (tangential-direction) and the other is perpendicular to its body axis (normal-direction). Because in the normal-direction the friction coefficient between the body and the ground is high, the moment to this direction does not occur. In contrast, the friction coefficient in the tangential-direction is low, hence propulsion is produced. The amplitude of serpentine locomotion depends on the friction coefficient. In order to make the friction coefficient of the tangential-direction low and that of the normal-direction high, Hirose designed a snake-like robot possessing passive wheels.

In this study, we employed a snake-like robot composed of ten rigid links each of which possesses passive wheels on its both sides, as depicted in Figure 1. Properties of links and wheels are summarized in the inset of Fig.1. Each joint is able to rotate around the z-axis when a torque is applied. A state is represented by the position of the ten links, $(x_i, y_i)$, $i = 1, 2, \ldots, 10$, the angles of the ten links, $\phi_i$, and their time derivatives, $(\dot{x}_i, \dot{y}_i)$ and $\dot{\phi}_i$. In later simulations, the dynamics of the snake-like robot was calculated by a simulator software called Open Dynamics Engine (ODE) [8].




| parameters | value |
|---|---|
| link mass | 1.0kg |
| link length | 0.15m |
| link width | 0.10m |
| link height | 0.10m |
| wheel mass | 0.5kg |
| wheel radius | 0.07m |

Figure 1: Snake-like robot

### 2.2. CPG model

Ekeberg simulated swimming of a lamprey, by using a mechanical chain model composed of ten rigid links. He used a CPG controller, as depicted in Figure 2; this CPG controller modeled the lamprey's spinal cord, such to have similar property to the real lamprey's CPG. The CPG controller was a recurrent neural network consisting of 100 segments connected one after another, and each segment was locally regulated by eight leaky-integrator neurons. Four were left neurons (LNs) and the other four were right neurons (RNs). There existed a motoneuron in LNs (RNs), which controlled a muscle on the left (right) side of the lamprey's body, and the output of this neuron is denoted by $MN_L$ ($MN_R$) in the followings.

The torque applied to the $i$-th joint was generated according to motoneuron's outputs in the $10i$-th segment ($MN_L(10i)$ and $MN_R(10i)$), and was calculated by

$$\begin{aligned}\tau_i &= K_m(MN_L(10i) - MN_R(10i)) + K_s(MN_L(10i) \\ &\quad + MN_R(10i) + S_t)(\phi_{i+1} - \phi_i) + K_d(\dot{\phi}_{i+1} - \dot{\phi}_i). \quad (1)\end{aligned}$$

$K_m$, $K_s$, $S_t$ and $K_d$ were the gain of muscles, the stiffness gain, the tonic stiffness and the damping coefficient, respectively, and were fixed.

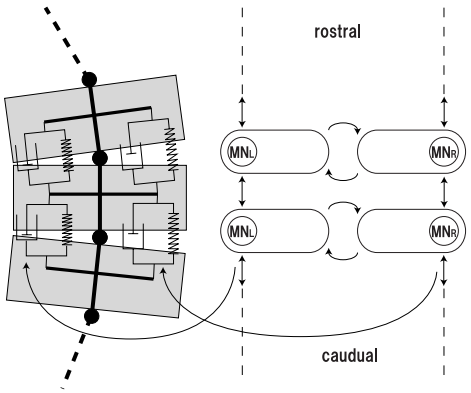



Figure 2: CPG controller

The CPG controller was controlled by a tonic input signal supplied by brainstem. The curvature of the lamprey depends on the strength of the tonic inputs, and asymmetric tonic inputs to LN and RN modify moving direction of the model lamprey.

The swimming of a lamprey has similar characteristics to those of the locomotion of a snake. Then, we employ this CPG controller to control a snake-like robot.

## 3. Learning method

RL schemes are usually formulated as to obtain a stationary policy which depends only on the state of the controlled object. It is not suited for training a CPG controller, because the control signal for the CPG controller is necessarily dependent on its own internal state. We therefore proposed a learning method for a CPG controller, which is called the CPG-actor-critic model [7]. In our scheme, CPG neurons and the controlled object were treated as a single dynamical system called a CPG-coupled system, and RL tried to control the CPG-coupled system. Because it is expected that the controlled object is entrained into a limit cycle produced by the CPG neurons, it is reasonable to treat CPG neurons and the controlled object as a single dynamical system. In addition, although a CPG controller is a kind of recurrent neural networks, and hence its naive training is nonlinear and needs heavy computation, the training of a CPG-coupled system becomes linear; this is profitable for stabilizing RL

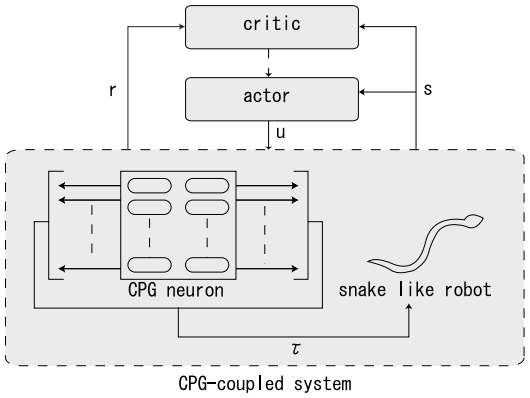



Figure 3: CPG-actor-critic model

### 3.1. Learning algorithm

We here explain the actor-critic method based on stochastic policy gradient [6]. In this method, the critic approximates the state value function which represents the expected accumulation of future rewards. The actor represents a parametric policy, whose parameters are updated based on the policy gradient.

#### *3.1.1. Stochastic policy gradient method*

Parameters are updated according to the following procedure.

1. At time $t$, the actor observes a system state $\mathbf{s}(t)$, and outputs an action $\mathbf{u}(t)$ according to its own policy $\pi$. The system changes its state to $\mathbf{s}(t+1)$.

2. The critic observes a reward $r(t)$ and a new state $\mathbf{s}(t+1)$, and calculates the TD-error $\delta(t)$:

$$\delta(t) = [r(t) + \gamma V(\mathbf{s}(t+1))] - V(\mathbf{s}(t)), \quad (2)$$

where $\gamma$ is the fixed discount rate, and the state value function $V(\mathbf{s})$ is calculated by the current critic.

3. Update the critic's parameters by

$$\mathbf{e}_v = \frac{\partial}{\partial w} V(\mathbf{s}) \quad (3)$$

$$\bar{\mathbf{e}}_v := \gamma\lambda\bar{\mathbf{e}}_v + \mathbf{e}_v \quad (4)$$

$$\mathbf{w}_v := \mathbf{w}_v + \alpha\delta(t)\mathbf{e}_v, \quad (5)$$

where $\mathbf{e}_v$, $\bar{\mathbf{e}}_v$ and $\alpha_v$ are the eligibility of the critic parameter $\mathbf{w}_v$, the eligibility trace, and the fixed learning rate, respectively.

4. Update the actor's parameters by

$$\mathbf{e}_\pi = \frac{\partial}{\partial \mathbf{w}_\pi} \ln\Big(\pi(\mathbf{u}|\mathbf{s})\Big) \quad (6)$$

$$\bar{\mathbf{e}}_\pi := \gamma\lambda\bar{\mathbf{e}}_\pi + \mathbf{e}_\pi \quad (7)$$

$$\mathbf{w}_\pi := \mathbf{w}_\pi + \alpha_\pi\delta(t)\bar{\mathbf{e}}_\pi, \quad (8)$$

where $\mathbf{e}_\pi$, $\bar{\mathbf{e}}_\pi$ and $\alpha_\pi$ are the eligibility of the actor parameter $\mathbf{w}_\pi$, the eligibility trace and the fixed learning rate, respectively.

5. If the actor parameters converge, exit. Otherwise, increment $t$ and go to step 1.

## 4. Simulation

### 4.1. Problem Formulation

In this computer simulation, we aim at obtaining a controller that allows the snake-like robot to move toward a target direction which is parallel with the x-axis (Figure 4). In order to encourage the movement toward the x-direction, an immediate reward was defined as

$$r = \dot{x}_1 - |\dot{y}_1| - \sum_{i=1}^{9} \tau_i^2, \quad (9)$$

where $\dot{x}_1$, $|\dot{y}_1|$ and $\tau_i$ denote the velocity of the head link toward the x-direction, the absolute velocity of the head link toward the y-direction and a torque applied to the $i$-th joint, respectively. The first term encourages the snake-like robot to move fast, the second term suppresses being apart from the target direction and the third term works to reduce the energy consumption.

In order to examine the adaptability to environmental changes, the friction coefficient was set at 1.0 for $t = 0 \sim 1000$(sec) and 0.1 for $t = 1000 \sim 3000$(sec). Before learning, the snake-like robot was set to be straight in parallel with the x-axis and to have no velocity.

### 4.2. Simulation condition

A control signal to the CPG-coupled system was defined as a two dimensional vector $u \equiv (u_{LN}, u_{RN})$ which was given by the probabilistic policy:

$$\pi(\mathbf{u}|\mathbf{s}) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{(\mathbf{u}-\bar{\mathbf{u}})'(\mathbf{u}-\bar{\mathbf{u}})}{2\sigma^2}\right). \quad (10)$$

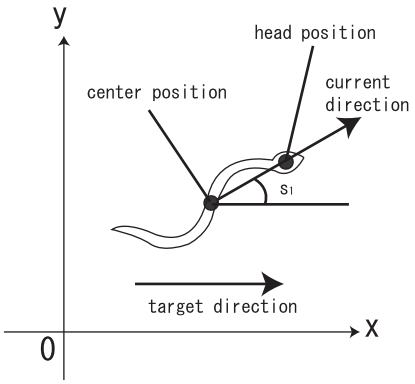



Figure 4: Problem setting

The center variables, $\bar{u}_{LN}$ and $\bar{u}_{RN}$, were calculated by

$$\bar{u}_{LN} = w_1 s_1 + w_3 + 0.5 \quad (11)$$

$$\bar{u}_{RN} = w_2 s_1 + w_3 + 0.5, \quad (12)$$

where $w_1$, $w_2$ and $w_3$ were policy parameters. After a control signal was generated, $u_{LN}$ ($u_{RN}$) was applied as a tonic input to $LN_s$ ($RN_s$). The state variable $s_1$ denoted the angle between the current direction of the snake-like robot and the target direction (see Fig. 4).

### 4.3. Feature extraction

Because the state space of the CPG-coupled system, which consists of the state of the snake-like robot (22 dimensions) and the internal state of the CPG controller (2400 dimensions), is very large, it is difficult to approximate the value function (equation (8)) in the original state space. When the snake-like robot is controlled by the CPG controller, the "posture" of the snake-like robot has an s-shape and the control signal $\mathbf{a}_{\mathrm{TI}}$ mainly affects the magnitude of the s-shape's curvature. Furthermore, all CPG neurons change their states with almost the same frequency. Extracting appropriate features, then, enables the controller to ignore the redundancy within the state variables of the CPG-coupled system. We define three feature variables as inputs to the basis functions:

$$s_1 = \arctan\left(\frac{x_1 - \sum_{j=1}^{10} x_j}{y_1 - \sum_{j=1}^{10} y_j}\right) \quad (13)$$

$$s_2 = \sum_{k=1}^{10} |\phi_{k+1} - \phi_k| \quad (14)$$

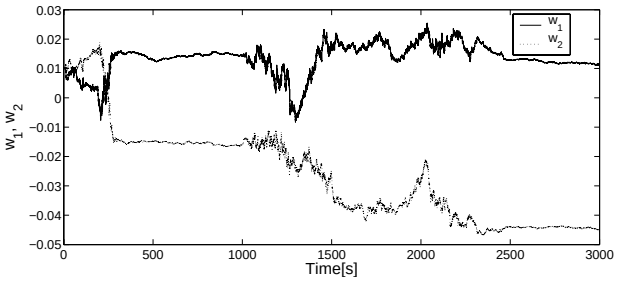
$$s_3 = \sum_{l=1}^{100} MN_L(l) \quad (15)$$

$$s_4 = \sum_{m=1}^{100} MN_R(m). \quad (16)$$

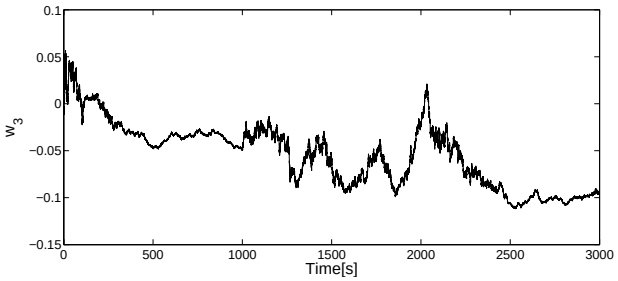
Namely, $s_1$, $s_2$, $s_3$ and $s_4$ represent the direction of the snake-like robot, the magnitude of serpentine curvature, the

summation of left motoneurons' outputs and the summation of right motoneurons' outputs, respectively.

### 4.4. Simulation results




(a) Actor parameters $w_1$ and $w_2$




(b) Actor parameter $w_3$

Figure 5: Actor parameters

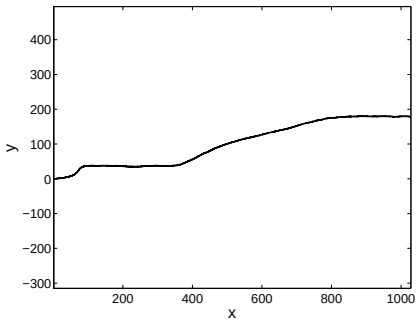



Figure 6: Trajectory of the snake-like robot's head link

Figures 5 and 6 show the actor parameters and the trajectory of the head link, respectively. Until the environment changed at $t = 1000$ (sec), $w_1$, $w_2$ and $w_3$ converged to 0.015, −0.015 and −0.03, respectively. After the environmental change, the policy parameters began to change in order to adapt to the new environment. At about 2500 sec, $w_1$, $w_2$ and $w_3$ converged to 0.013, −0.044 and −0.13, respectively, and then, the snake-like robot came to move toward the target direction stably again (Figure 6).

## 5. Conclusion

In this study, we configured a control scheme for a snake-like robot, based on a CPG controller, and developed a learning algorithm to acquire a good control rule autonomously in a changing environment.

Computer simulation showed that a control rule that allowed the snake-like robot to move toward the target direction was obtained by our scheme.

## References


[1] M. A. Arbib, "*The Handbook of Brain Theory and Neural Networks, Second Edition*" . MIT Press, 2002.

[2] Ö. Ekeberg, "A combined neuronal and mechanical model of fish swimming, "*Biological Cybernetics*, vel.69: pp.363–374, 1993.

[3] Y. Fukuoka, H. Kimura and A. H. Cohen, "Adaptive Dynamic Walking of a Quadruped Robot on Irregular Terrain based on Biological Concepts" , Int. Journal of Robotics Research, Vol.22, No.3-4, pp.187-202, 2003

[4] S. Hirose, *Biologically Inspired Robots (Snake-like Locomotor and Manipulator)*, Oxford University Press, 1993.

[5] A. J. Ijspeert, "A connectionist central pattern generator for the aquatic and terrestrial gaits of a simulated salamander, " *Biological Cybernetics*, Vol.84:5, pp.331–348, 2001.

[6] H. Kimura, and S. Kobayashi, "An Analysis of Actor/Critic Algorithms using Eligibility Traces: Reinforcement Learning with Imperfect Value Function," 15th International Conference on Machine Learning, pp.278–286, 1998.

[7] Y. Nakamura, M. Sato, and S. Ishii, "Reinforcement Learning for biped Robot," 2nd International Symposium on Adaptive Motion of Animals and Machines, 2003.

[8] R. Smith "Open Dynamics Engine," http://ode.org/

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

[10] G. Taga, Y. Yamaguchi and H. Shimizu, "Selforganized control of bipedal locomotion by neural oscillators in unpredictable environment," *Biological Cybernetics* Vol.65(3), 147–159, 1991.