International Journal of Control, Automation, and Systems (2009) 7(3):447-457 http://www.springer.com/12555
DOI 10.1007/s12555-009-0314-5

Central Pattern Generator Parameter Search for a Biped Walking Robot Using Nonparametric Estimation Based Particle Swarm Optimization

Jeong-Jung Kim, Jun-Woo Lee, and Ju-Jang Lee

Abstract: A parameter search for a Central Pattern Generator (CPG) for biped walking is difficult because there is no methodology to set the parameters and the search space is broad. These characteristics of the parameter search result in numerous fitness evaluations. In this paper, nonparametric estimation based Particle Swarm Optimization (NEPSO) is suggested to effectively search the parameters of CPG. The NEPSO uses a concept experience repository to store a previous position and the fitness of particles in a PSO and estimated best position to accelerate a convergence speed. The proposed method is compared with PSO variants in numerical experiments and is tested in a three dimensional dynamic simulator for bipedal walking. The NEPSO effectively finds CPG parameters that produce a gait of a biped robot. Moreover, NEPSO has a fast convergence property which reduces the evaluation of fitness in a real environment.

Keywords: Biped robot, central pattern generator, particle swarm optimization.

1. INTRODUCTION

In the past two decades, there has been growing interest in a biped robot because it has unique advantages compared to other types of robots. The first advantage of a biped robot arises from its mobility. Biped robots can reach places that are inaccessible to wheeled robots, such as over rough terrain, and are a better fit for home and working environments designed for humans. A Second advantage of a biped robot arises from its human-like shape, which is more natural to humans than other types of robots. As such, biped type robots could be used in the future as personal assistants, home assistants, and even as entertainment devices [1].

A major research issue related to biped robots is that of stable walking. One successful realization of a walking biped robot is based on ZMP [2-4]. It is a well defined methodology that guarantees a robot's stability, is easy to implement and can be applied to dynamic walking. However, it requires precise models of the robot and the environment it is designed to perfom in, as well as an extra online control to deal with external perturbation.

In contrast to a model based bipedal walking robot, a biological inspiration based biped walking robot is matter of concern these days. Animals can adapt their locomotion according to specific environments. The locomotion of animals is generated by the Central Pattern Generator (CPG). A mathematical model of the CPG can be applied to generate target torque or joint angle in a robot system.

The CPG model has been widely used in robotic systems such as the snake robot [7], biped locomotion [8-15], quadruped locomotion [16], and arm movement [17,18] because the oscillators have desirable properties such as adaptation to the environment through entrainment.

CPG based bipedal walking does not require information for a robot model in an environment and can smoothly change a gait with little computation burden. However, there is no methodology and there are too many parameters to set for CPG. Therefore, evolutionary computation methods such as Genetic Algorithms (GAs) [11-13], multi-objective Genetic Algorithms [14], and Genetic Programming (GP) [15] are often used to optimize the parameters.

However, when the global optimization method is applied to find CPG parameters, the method evaluates the fitness of application to a robot and numerous fitness evaluations are needed. As such, convergence is an important factor in the selection of a method for preventing a robot from numerous iterations of the method.

Particle Swarm Optimization (PSO) is a population based stochastic optimization method proposed by Kennedy and Eberhart in 1995 and is inspired by social behavior such as flocks of birds or schools of fish [19]. The main advantages of PSO are simple to understand, easy to implement and quick in convergence compared to other global optimization methods such as Genetic Algorithms (GA) and Simulated Annealing (SA) [20]. PSO has been successfully applied in continuous nonlin-

Manuscript received March 10, 2008; revised July 23, 2008 and October 2, 2008; accepted December 15, 2008. Recommended by Editorial Board member Euntai Kim under the direction of Editor Jae-Bok Song.

Jeong-Jung Kim and Ju-Jang Lee are with Division of Electrical Engineering, School of Electrical Engineering & Computer Science, KAIST, 373-1 Guseong-dong, Yuseong-gu, Daejeon 350-701, Korea (e-mails: rightcore@kaist.ac.kr, jjlee@ee.kaist.ac.kr).

Jun-Woo Lee is with Robotics Program, KAIST, 373-1 Guseong-dong, Yuseong-gu, Daejeon 350-701, Korea (e-mail: good791@kaist.ac.kr).

ear function optimization [19], reactive power and voltage control [21], parameter tuning of a controller for a power system [22], PID controller design [23], and feeder reconfiguration [24].

In this paper, nonparametric estimation based PSO (NEPSO) is proposed to search for the parameters of CPG needed for bipedal walking. The method gathers position value and fitness of the particles scattered to each particle and stores it to an experience repository. Consequently, information stored in the experience repository is used for selecting an estimated best position to accelerate the convergence of each particle. The nonparametric estimation method is used to estimate fitness of randomly selected positions in a search space. The NEPSO is suitable for an application requiring a long fitness evaluation time in total computation time.

This paper is organized as follows. In Section 2, a model of a biped robot, the property of the CPG and the arrangement of the CPG for the robot are described. In Section 3, various PSO methods are described. In Section 4, our proposed nonparametric estimation based PSO is introduced. The results of numerical experiments and the CPG parameter search in a 3D dynamic simulator are shown in Section 5, and conclusions and further works are given in Section 6.

2. MODEL OF THE BIPED ROBOT AND CENTRAL PATTERN GENERATOR

In this section, a model of a biped robot and the Central Pattern Generator (CPG) used in this paper are introduced. An arrangement of CPGs for the biped robot is also presented.

2.1. Model of biped robot

The biped robot considered in this paper has only a lower body and consists of a left leg, a right leg and a waist. Each leg has five degrees of freedom (DOF) as shown in Fig. 1. Two DOFs, one DOF and two DOFs are allocated for each hip joint, knee joint and ankle joint respectively.

The parameters l_1, l_2, l_3 and l_4 denote the length of links; and f_f and f_b denote the length of a sole; and f_{s1} and f_{s2} denote the width of a sole. To simplify the dynamic parameter of the robot, each of the robot's links is considered a point mass (i.e., the total mass of each link is located at its center) and each link has m_1, m_2, m_2 and m_4 respectively. Those parameters are summarized in Table 1 and are used for making a robot simulator.

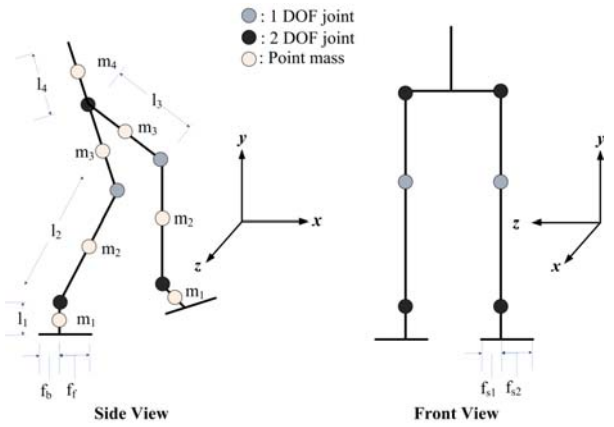

Fig. 1. The model of the biped robot.

Table 1. The parameters of each rigid body.

Parameters	Value
l_1	45.4mm
l_1	116.1mm
l_3	99.7mm
l_4	66.75mm
l_f	51.23mm
l_b	42.03mm
l_{s1}	40.4mm
l_{s2}	52.4mm
m_1	0.13kg
m_2	0.42kg
m_3	0.75kg
m_4	1.36kg

2.2. Central pattern generator

An animal can adapt its locomotion according to a specific environment. The locomotion of the animal is generated by the Central Pattern Generator (CPG). Mathematical models of a CPG are suggested by many researchers and the model proposed by Matsuoka is one of the more popular [6]. The neural oscillator model is used for modeling a CPG that can generate a desired joint angle reference. Every joint of the robot is driven by a neural oscillator that consists of two simulated neurons in mutual inhibition as shown in Fig. 2.

The state variables are determined by (1)- (5).

$$\tau_1 \dot{x}_1 = -x_1 - \beta v_1 - \omega [x_2]^+ - \sum_{j=1}^{n} h_j [g_j]^+ + c, \quad (1)$$

$$\tau_2 \dot{v}_1 = -v_1 + [x_1]^+, \quad (2)$$

$$\tau_1 \dot{x}_2 = -x_2 - \beta v_2 - \omega [x_1]^+ - \sum_{j=1}^{n} h_j [g_j]^+ + c, \quad (3)$$

$$\tau_2 \dot{v}_2 = -v_2 + [x_2]^+, \quad (4)$$

$$y = [x_1]^+ + [x_2]^+, \quad (5)$$

where x_1, x_2, x_3, x_4 are internal states, τ_1, τ_2, c, β, ω, h_j are constant parameters and g_j and

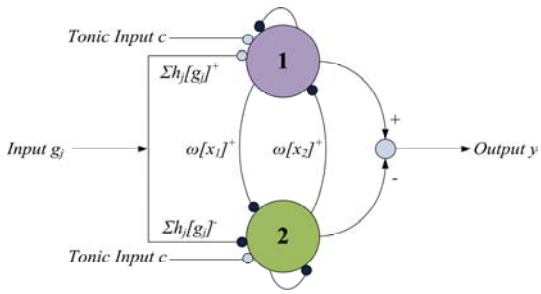

Fig. 2. A neural oscillator model.

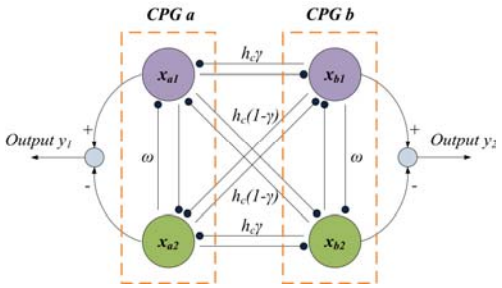

Fig. 3. A network of neural oscillator.

y are input and output signals, respectively. Time constant τ_1 and τ_2 determine the output shape and its frequency, and tonic excitation c modulates the amplitude of output. The input g_j is scaled by a weight h_j and applied to the oscillator. A frequency of the CPG is changed according to parameter τ_1 and τ_2 and an amplitude of the CPG is changed according to parameter c. However, various motions with just two properties of the CPG are not generated. A phase difference of the CPG is realized with a network of neural oscillators [18]. The network of neural oscillators is shown in Fig. 3 and the CPG equation (1) and (2) are changed to (6) and (7) respectively.

$$\tau_1 \dot{x}_1 = -x_1 - \beta v_1 - \omega[x_2]^+ - (h_c\gamma[x_{a2}]^+ + h_c(1-\gamma)[x_{b2}]^+) + c, \quad (6)$$

$$\tau_1 \dot{x}_2 = -x_2 - \beta v_2 - \omega[x_1]^+ - (h_c\gamma[x_{b2}]^+ + h_c(1-\gamma)[x_{a2}]^+) + c. \quad (7)$$

A phase of the CPG is changed according to parameter γ.

2.3. CPG arrangement for biped robot

Based on the three kinds of properties of the CPG in the previous section, the CPGs are arranged for the biped robot model as shown in Fig. 4 and their parameters are summarized in Tables 2 and 3.

Both legs have the same parameter value symmetrically. The target values of the joints are determined by the output of the CPGs. The output of the CPGs are used as target angles for each joint motor to make the robot walk. A control torque of each joint is calculated by the difference between target angle and the current angle of the motor as shown in (8).

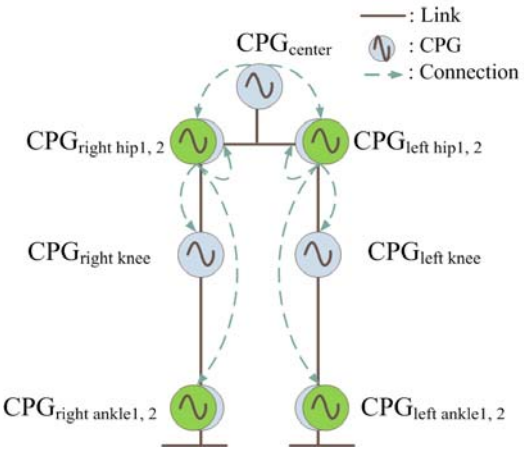

Fig. 4. The CPG arrangement for the biped robot.

Table 2. The parameters of CPG.

Parts	Parameters	Range
Center	τ_1	0.02 ~ 0.8
	c	0.0 ~3/2π
	u_{01}	0.0~1.0
Hip	c	0.0~3/10 π
	left u_{01}	-1.0~1.0
	right u_{01}	-1.0~1.0
Hip2	c	0.0 ~3/2π
	left u_{01}	-1.0~1.0
	right u_{01}	-1.0~1.0
	offset u_{01}	-1/2π ~1/2π
Knee	c	0.0 ~1/2π
	left u_{01}	-1.0~1.0
	right u_{01}	-1.0~1.0
	offset u_{01}	-1/2π~0.0
Ankle1	c	0.0 ~3/10π
	left u_{01}	-1.0~1.0
	right u_{01}	-1.0~1.0
Ankle2	c	0.0 ~3/2π
	left u_{01}	-1.0~1.0
	right u_{01}	-1.0~1.0
	offset u_{01}	-1/2π ~1/2π

Table 3. The parameters of CPG for phase shift.

Parameters	Range
Center to left hip1 phase γ	0.0~1.0
Center to right hip1 phase γ	0.0~1.0
Hip1 to hip2 phase phase γ	0.0~1.0
Hip1 to knee phase phase γ	0.0~1.0
Hip1 to ankle1 phase phase γ	0.0~1.0
Hip1 to ankle2 phase phase γ	0.0~1.0

$$\tau = k_p(\theta_t - \theta) + k_v\dot{\theta}, \quad (8)$$

where k_p and k_v are position and velocity gains, respectively, θ_t is target angle calculated from (5), and θ and $\dot{\theta}$ are the current angle and current angular velocity of each joint.

3. PARTICLE SWARM OPTIMIZATION

3.1. Particle swarm optimization

Particle Swarm Optimization uses the concept called particle and swarm. The particles correspond to an animal, bird, and insect in a herd, flock, and swarm respectively. Each particle has its own position and velocity and is randomly initialized in a search space.

When the particle number in the swarm is N and each particle is a D dimensional vector at an iteration t, the positions and the velocity of each particle is represented

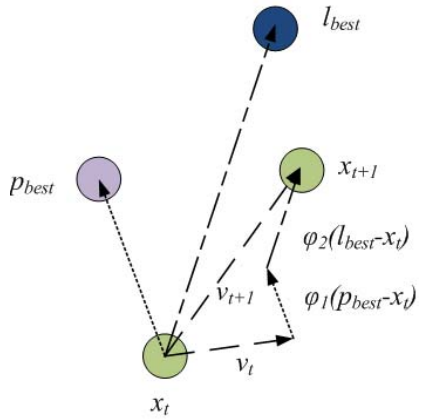

Fig. 5. The position change procedure of a particle in PSO.

as (9) and (10), respectively.

$$\vec{X}(t) = \{\vec{x}_1^D(t), \vec{x}_2^D(t), ..., \vec{x}_{N-1}^D(t), \vec{x}_N^D(t)\}, \tag{9}$$

$$\vec{V}(t) = \{\vec{v}_1^D(t), \vec{v}_2^D(t), ..., \vec{v}_{N-1}^D(t), \vec{v}_N^D(t)\}, \tag{10}$$

where $\vec{x}_n(t)$ and $\vec{v}_n(t)$ are represented in (11) and (12) respectively.

$$\vec{x}(t) = \{x^1(t), x^2(t), ..., x^{D-1}(t), x^D(t)\}, \tag{11}$$

$$\vec{v}(t) = \{v^1(t), v^2(t), ..., v^{D-1}(t), v^D(t)\}. \tag{12}$$

As an iteration progresses, the particles cooperate and finally reach a solution by preserving and sharing their previous best positions. The particles store their best experience during the optimization process and the velocity and the position of each particle is updated by (13) and (14), respectively.

$$\begin{aligned}\vec{V}(t+1) = \vec{V}(t) &+ \varphi_1(\vec{p}_{best} - \vec{X}(t)) \\ &+ \varphi_2(\vec{l}_{best} - \vec{X}(t)),\end{aligned} \tag{13}$$

$$\vec{X}(t+1) = \vec{X}(t) + \vec{V}(t+1), \tag{14}$$

where $\vec{V}(t)$ is a velocity and $\vec{X}(t)$ is a position of the particle at t iteration. $\vec{p}_{best}$ is a previous best position and $\vec{l}_{best}$ is a local best position of each particle obtained so far. φ_1 and φ_2 are determined as $\varphi_1 = rand(0, c_1)$ and $\varphi_2 = rand(0, c_2)$ and c_1 is a cognition learning factor and c_2 is a social learning factor. The position change procedure of a particle in the PSO is shown in Fig. 5.

3.2. Particle swarm optimization variants

The original velocity change equation of the PSO (13) is extended to an inertia weighted version and $\vec{g}_{best}$ version which consequently are (15) and (16), respectively.

$$\begin{aligned}\vec{V}(t+1) = \alpha\vec{V}(t) &+ \varphi_1(\vec{p}_{best} - \vec{X}(t)) \\ &+ \varphi_2(\vec{l}_{best} - \vec{X}(t)),\end{aligned} \tag{15}$$

$$\begin{aligned}\vec{V}(t+1) = \vec{V}(t) &+ \varphi_1(\vec{p}_{best} - \vec{X}(t)) \\ &+ \varphi_2(\vec{g}_{best} - \vec{X}(t)),\end{aligned} \tag{16}$$

where α is the inertia coefficient and $\vec{g}_{best}$ is the best position in the whole swarm. The PSO variant based on (15) is introduced to balance the global search and local search and PSO variant based on (16) is introduced to accelerate the converge speed [25,26].

Krohling [27] suggested the velocity equation based on the Gausssian distribution random number generator in PSO and known as the Gaussian Swarm. It does not require a PSO cognition learning factor c_1 and social learning factor c_2 and converges faster than canonical PSO. The velocity change equation of the Gaussian Swarm is (17)

$$\begin{aligned}\vec{V}(t+1) = &|randn|(\vec{p}_{best} - \vec{X}(t)) \\ &+ |randn|(\vec{g}_{best} - \vec{X}(t)),\end{aligned} \tag{17}$$

where $|randn|$ is positive random numbers generated according to the absolute value of the Gaussian probability distribution, i.e., $abs[N(0,1)]$.

4. NONPARAMETRIC ESTIMATION BASED PARTICLE SWARM OPTIMIZATION

In this section, nonparametric estimation based Particle Swarm Optimization (NEPSO) is introduced. The method is based on an experience repository and an estimated best position to realize fast convergence.

4.1. Experience repository

In PSO, particles cooperate and finally reach a solution by preserving and sharing their previous best positions. In conventional PSO, only the best experience of each particlee and the global experience of the swarm is used to update the velocity of each particle. There remains a chance to use other information to improve the velocity update equation. In this paper we suggest a method that uses the more prior and wide information of the particles than the canonical PSO. PSO is inspired by social behavior, but information in the canonical PSO is scattered to each particle. The place where the experience of particles is stored is defined and named the experience repository.

In the experience repository, not only the experience of whole particles but also experiences of particles at prior iterations are included. The positions of particles and their fitness values are used as information for each particle. This information is added to the experience repository until the number of particles in the experience repository reaches M, which is the size of the experience repository. The large size of an experience repository can contain more information about the particles. An example of the experience repository is shown in Fig. 6. In Fig. 6 the size of the experience repository M is 100. If the size of the swarm in the PSO is 10, the repository can contain the experience of particles during 10 iterations.

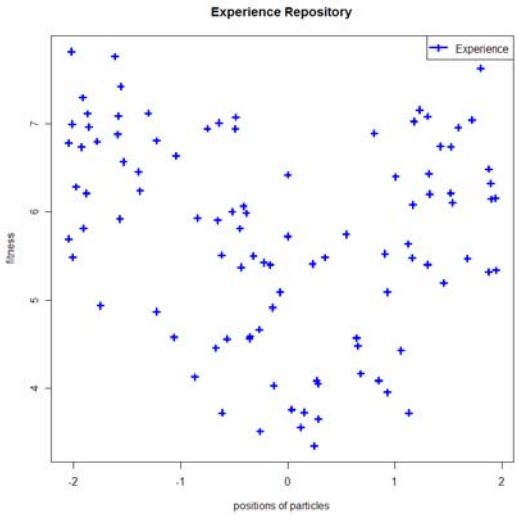

Fig. 6. The experience repository.

4.2. Estimated best position

The information stored in the repository is used to update the velocity of each particle. A modification of the original velocity (18) is suggested to improve the convergence of the optimization process of the PSO.

Equation (18) is used until the current size of an experience repository reaches the maximum size of the experience repository M.

$$\vec{V}(t+1) = |randn|(\vec{g}_{best} - \vec{X}(t)) + |randn|(\vec{e}_{best} - \vec{X}(t)), \quad (18)$$

where $|randn|$ is positive random numbers generated according to the absolute value of the Gaussian probability distribution and $\vec{e}_{best}$ is an estimated best position. Suggested equation (18) is based on the Gaussian Swarm version of the velocity update equation (17) and the $\vec{e}_{best}$ term is added to it. $\vec{e}_{best}$ represents the estimated best position in the experience repository. The estimated best position is selected from the experience repository. A procedure for the selection of the estimated best position in the experience repository is shown in Fig. 7. First, K number of positions within the search space range are sampled as in (19) and their fitness values are estimated as in (20). Finally, the best position is selected among the sampled positions that have the best estimated fitness value as in (21).

$$\vec{x}' = \{x_1', x_2', ..., x_K'\}, \quad (19)$$

$$\vec{g}(x_k') = \{g(x_1'), g(x_2'), ..., g(x_K')\}, \quad (20)$$

$$\vec{x}'^{*} = \arg\min_{x_k'} g(x_k'), \text{ or}$$

$$\vec{x}'^{*} = \arg\max_{x_k'} g(x_k'). \quad (21)$$

A nonparametric estimation method is used to estimate fitness. The nonparametric estimation uses less assumption than a parametric estimation. The nonparametric estimation is divided into nonparametric density estimation and nonparametric regression. The latter is used in NEPSO to estimate the fitness of the position. In parametric regression, we assume that close x have close $g(x)$ values and we find the neighborhood of x and average the r values in the neighborhood to calculate $\hat{g}(x)$ [28]. There are many kinds of methods for defining the neighborhood and averaging in the neighborhood. In the NEPSO, a kernel smooth is used to define the neighborhood and average in the neighbor because it produces smooth curves compared to the mean smoother and the running line smoother. The kernel smoother is defined as (22) and gives less weight to further points.

Fig. 7. The procedure of selecting estimated best position.

$$\hat{g}(x) = \frac{\sum_{t}^{M} H(\frac{x - x^t}{h}) r^t}{\sum_{t}^{M} H(\frac{x - x^t}{h})}, \quad (22)$$

where $H(u) = \frac{1}{\sqrt{2\pi}} \exp[-\frac{u^2}{2}]$. An example of a kernel smoother is shown in Fig. 8. Based on 12 data points, the remaining range is estimated by nonparametric estimation. An example of selecting an estimated best position is shown in Fig. 9. Cross dots represent the experience of the particle in the repository and the round dots are sampled positions of particles and their estimated fitness. Among the K sampled positions, $\vec{e}_{best}$ is selected that has the best estimated fitness value.

4.3. Effect of sampling and estimating fitness

In NEPSO, K positions are sampled and their fitness estimated. The estimated best value is not an exact fitness value, but represents a tendency of the neighborhood. The neighborhood with a good fitness value produces a good estimated fitness value, and a bad fitness value produces a bad estimated fitness value. As such, the selected best position reflects the tendency of the neighborhood and accelerates convergence by adding this term to the Gaussian Swarm velocity equation as in (18).

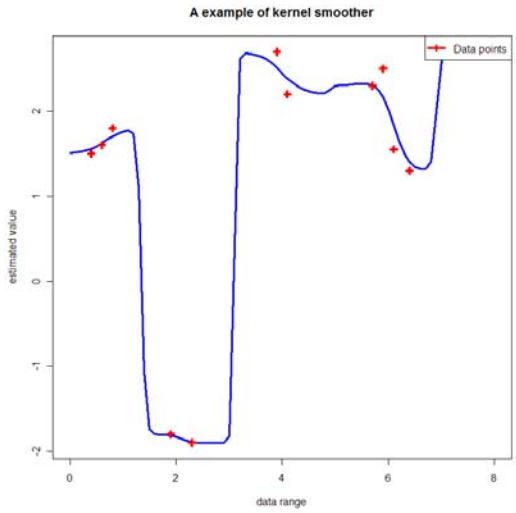

Fig. 8. A example of kernel smoother.

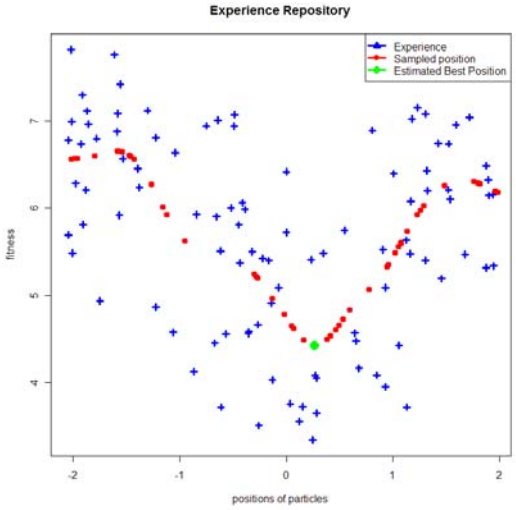

Fig. 9. A example of selecting estimated best position.

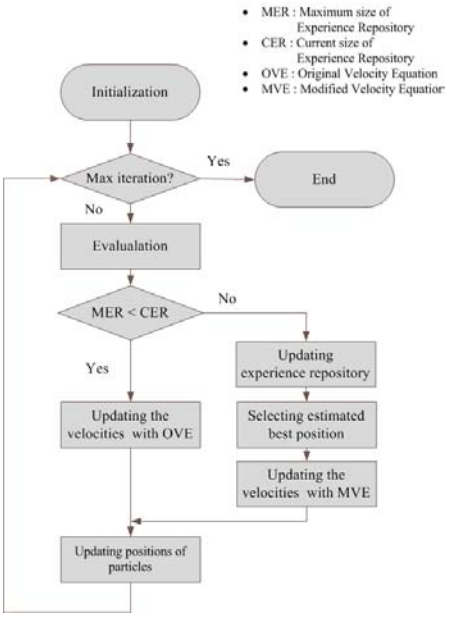

Fig. 10. The NEPSO optimization procedure.

The sampling and estimating of best value can be compared with a tournament selection in a Genetic Algorithm (GA). In the tournament selection, chromosomes are selected randomly and chosen from a previous generation. However, sampling from the experience repository in NEPSO is based on more previous experience and estimated value which is not an exact value. Estimating a fitness covers the range that particles have not been experienced based on the neighbor experience.

4.4. Optimization procedure of NEPSO

The optimization procedure of the NEPSO is summarized in Fig. 10. Updating velocities and position in the canonical PSO procedure are extended to 'Updating experience repository', 'Selecting estimated best position', and 'Updating the velocities and population'. The procedure is repeated until the maximum iteration.

5. SIMULATION

In this section, the proposed method is verified by numerical experiment and a CPG parameter search for the biped walking robot in a 3D dynamic simulator. The simulation environment is summarized in Table 4.

5.1. Numerical experiments

Four nonlinear functions are used for numerical experiments to verify the suggested method. The first function is the Rosenbrock function which has 2 dimensional search space described by (23):

$$f_1(\vec{x}) = 100(x_0^2 - x_1)^2 + (1 - x_0)^2, \quad x_i \in [-2.048, 2.048]. \tag{23}$$

The minimum value of (23) is $f_1(1) = 0$. The second function is the sphere function with 3 dimensional search space described by (24):

$$f_2(\vec{x}) = \sum_{i=0}^{2} x_i^2, \quad x_i \in [-100, 100]. \tag{24}$$

The minimum value of (24) is $f_2(0) = 0$. The third function is the Griewank function which has 10 dimensional search space described by:

$$f_3(\vec{x}) = \frac{1}{400}\sum_{i=0}^{9} x_i^2 - \prod_{i=0}^{9} \cos\left(\frac{x_i}{\sqrt{i+1}}\right) + 1, \quad x_i \in [-400, 400]. \tag{25}$$

The minimum value of (25) is $f_3(0) = 0$. The fourth function is Rastrigin function which has 10 dimensional search space described by:

Table 4. The simulation environment.

CPU Clock	Memory	Operating System
3.2Ghz	1GB	Windows XP

$$f_4(\vec{x}) = \sum_{i=0}^{9} \{x_i^2 - 10\cos(2\pi x_i) + 10\}, \quad x_i \in [-5.12, 5.12]. \tag{26}$$

The minimum value of (26) is $f_4(0) = 0$. The canonical PSO (14), PSO variant 1 based on (15), PSO variant 2 based on (16), and PSO variant 3 based on (17) are compared with NEPSO (18) and their parameters are summarized in Table 5.

A total of 50 runs for each method are conducted and the mean best fitness are summarized in Tables 6-9. In test function 1, none of the methods converged to the solution, but NEPSO found the nearest solution. In test function 2, NEPSO and PSO variant 3 found the solution. In test functions 3 and 4, none of the methods converged to the solution, but NEPSO found the nearest solution. NEPSO requires additional computation time compared to PSO variants.

The results of the test functions are shown in Figs. 11-14. Except for the results in Fig. 11, the convergence speed of NEPSO is faster than other methods.

Table 5. The parameter for numerical experiments.

Method	Parameters	Value
Canonical PSO	Swarm size	30
	Max generation	50
	Cognition learning factor c_1	1.7
	Social learning factor c_2	1.7
PSO Variant 1	Swarm size	30
	Max generation	50
	Cognition learning factor c_1	1.7
	Social learning factor c_2	1.7
	Inertia coefficient α	1.66
PSO Variant 2	Swarm size	30
	Max generation	50
	Cognition learning factor c_1	1.7
	Social learning factor c_2	1.7
PSO Variant 3	Swarm size	30
	Max generation	50
NEPSO	Swarm size	30
	Max generation	50
	Sampling number K	10
	Repository size M	300

Table 6. The optimization result for Function 1.

Method	Mean best ftiness	Average time
Canonical PSO	0.03089038	2047.673 μs
PSO Variant 1	0.00272978	2061.309 μs
PSO Variant 2	0.00962446	2174.429 μs
PSO Variant 3	0.00356192	2219.701 μs
NEPSO	0.00233984	3685.628 μs

Table 7. The optimization result for Function 2.

Method	Mean best ftiness	Average time
Canonical PSO	67.96129	2576.482 μs
PSO Variant 1	0.00278472	2655.964 μs
PSO Variant 2	1.77326	2561.613 μs
PSO Variant 3	0.0	2810.248 μs
NEPSO	0.0	7223.532 μs

Table 8. The optimization result for Function 3.

Method	Mean best ftiness	Average time
Canonical PSO	23.71134	5599.437 μs
PSO Variant 1	1.499977	5379.258 μs
PSO Variant 2	12.95996	5596.306 μs
PSO Variant 3	0.2501918	5791.531 μs
NEPSO	0.1551262	19864.479 μs

Table 9. The optimization result for Function 4.

Method	Mean best ftiness	Average time
Canonical PSO	87.73664	5178.031 μs
PSO Variant 1	50.39771	5233.759 μs
PSO Variant 2	19.4308	5609.016 μs
PSO Variant 3	0.73671	6163.525 μs
NEPSO	0.15512	9850.999 μs

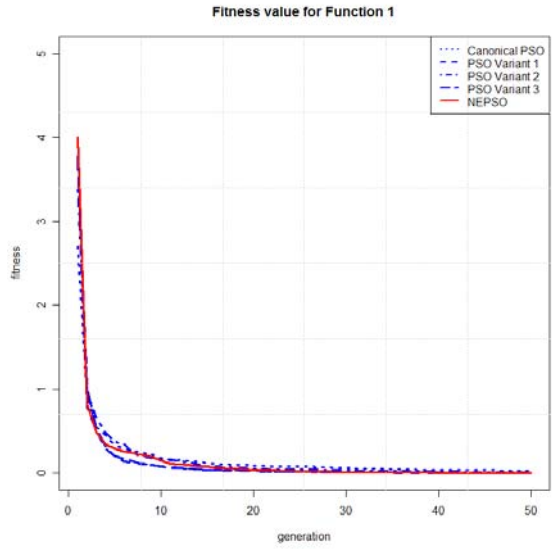

Fig. 11. Convergence performances of various PSOs on Function 1.

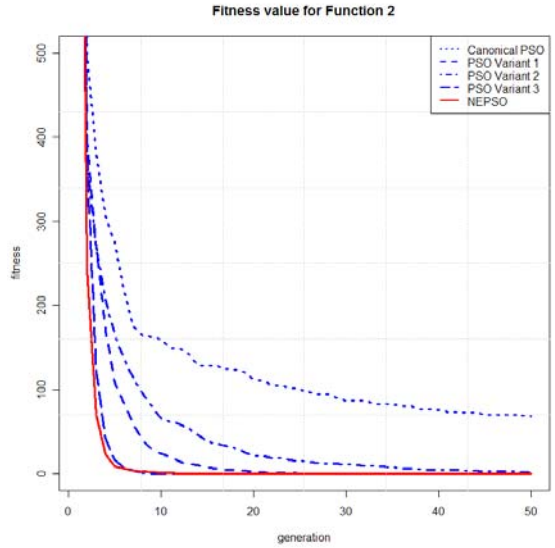

Fig. 12. Convergence performances of various PSOs on Function 2.

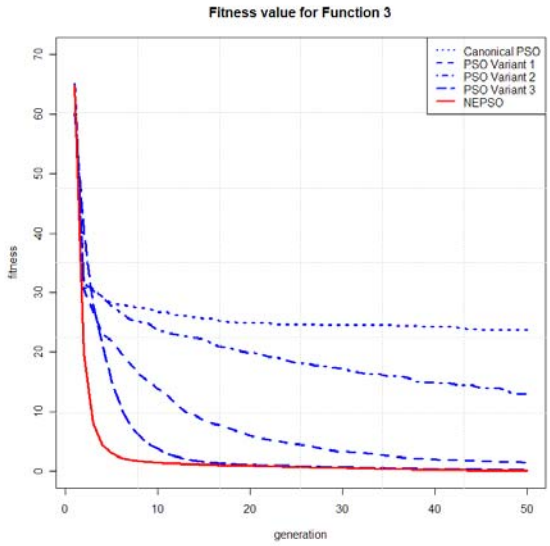

Fig. 13. Convergence performances of various PSOs on Function 3.

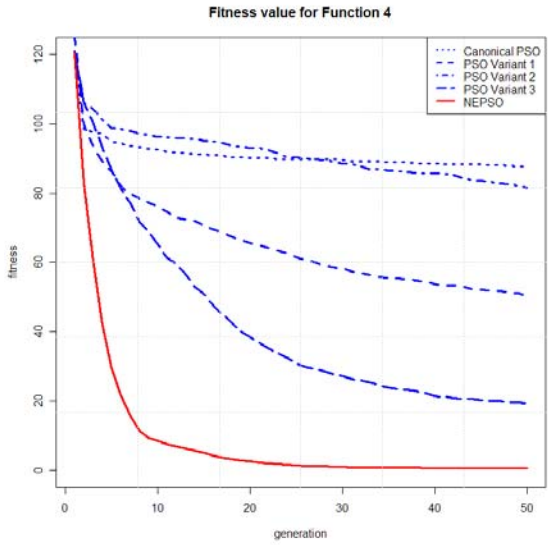

Fig. 14. Convergence performances of various PSOs on Function 4.

5.2. CPG parameter search in 3D dynamic simulator

A simulation is conducted to verify the proposed method using a 3D simulator made with the Open Dynamics Engine (ODE) [29]. The ODE is an open source rigid body dynamics engine and a realistic simulator can be implemented using this engine. The configuration of the biped robot and its parameters in the simulator are shown in Section 2.

Each parameter in PSO is evaluated in the simulator and the evaluation procedure is shown in Fig. 15. It takes about 300ms to evaluate the fitness per iteration in the simulator. A proper set of CPG parameters should be found to make the robot walk. Two fitness functions are designed and one of them is selected as shown in Fig. 16. The first fitness functions are used for the falling of the biped robot and defined as (27).

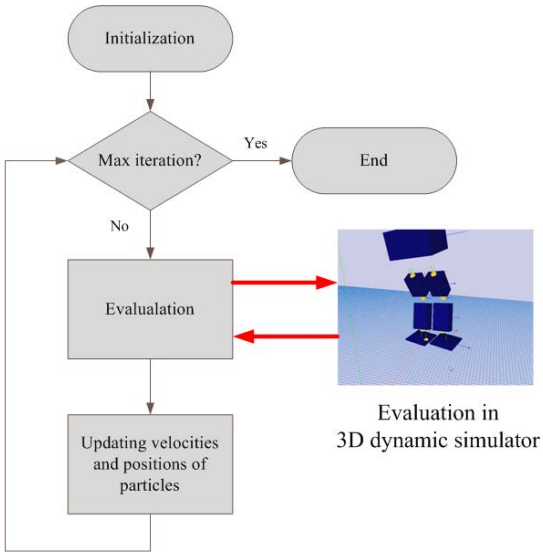

Fig. 15. The evaluation procedure.

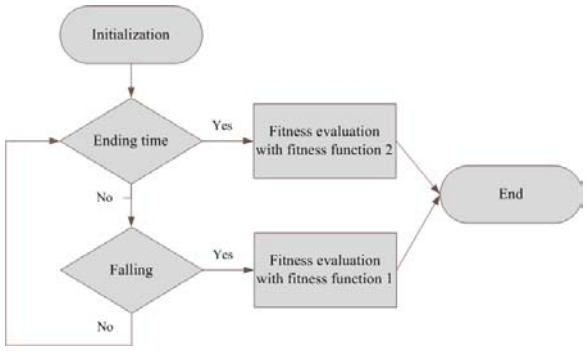

Fig. 16. The procedure of selecting fitness function.

$$fitness = \omega_1 \times \min(torso_x, leftfoot_x, rightfoot_x) + \omega_2 \times S_{\max} + \omega_3 \times T_f, \quad (27)$$

where $torso_x$, $leftfoot_x$, and $rightfoot_x$ are distances from the initial position to the final position of the torso, position of the left foot, and position of the right foot in the x axis, respectively. And $S_{\max}$ and T_f are max step and a final time that the robot has been kept in the predefined y_{stable} in the y axis direction respectively. ω_1, ω_2 and ω_3 are the weighting factors. (27) tries to maximize the distance from the initial position to the final position. For the experiment, the weighting factors ω_1, ω_2 and ω_3 are set to 2, 5, and 1 respectively. y_{stable} is set to 0.27 m. Second fitness functions are used for maximizing the distance from the initial position to the final position within a predefined time T_{pre} and defined as (28).

$$fitness = C_{add} + \omega_4 \times torso_x + \omega_5 \times S_{\max} - \omega_6 \times |torso_z|, \quad (28)$$

where $torso_x$ and $torso_z$ are distances from the initial

Table 10. The parameter for CPG parameter search.

Method	Parameters	Value
Canonical PSO	Swarm size	50
	Max generation	50
	Cognition learning factor c_1	1.7
	Social learning factor c_2	1.7
PSO Variant 1	Swarm size	50
	Max generation	50
	Cognition learning factor c_1	1.7
	Social learning factor c_2	1.7
	Inertia coefficient α	1.66
PSO Variant 2	Swarm size	50
	Max generation	50
	Cognition learning factor c_1	1.7
	Social learning factor c_2	1.7
PSO Variant 3	Swarm size	30
	Max generation	50
NEPSO	Swarm size	50
	Max generation	50
	Sampling number K	10
	Repository size M	500

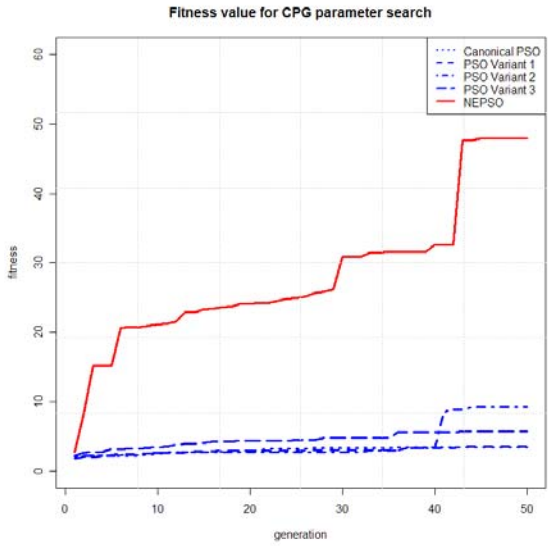

Fig. 17. The parameter search result.

position to the final position of the torso in the x axis and the initial position to the final position of the torso in the z axis. C_{add} is an additional value that distinguishes the state between (27) and (28). ω_4, ω_5 and ω_6 are weighting factors. (28) tries to maximize the distance from the initial position to the final position in the x axis and minimizes the distance from the initial position to the final position in the z axis. For the experiment, the weighting factors ω_4, ω_5 and ω_6 are set to 20, 20 and 5 respectively. C_{add} is set to 20. The parameters for PSOs and NEPSO are summarized in Table 10. The optimization result for the CPG parameter search is shown in Fig. 17 and summarized in Table 11. The NEPSO was the only method that could use fitness function 2 (28) and found the best fitness value as shown in Table 11 and Fig. 17. The NEPSO uses previous knowledge by storing the information of particles into

Table 11. The optimization result for CPG parameter search.

Method	Final ftiness
Canonical PSO	3.4929
PSO Variant 1	3.60935
PSO Variant 2	9.31041
PSO Variant 3	5.74939
NEPSO	47.91904

Table 12. The parameter of CPG found by NEPSO.

Parts	Parameters	Range
Center	τ_1	0.024135
	c	0.000000
	u_{01}	0.000000
Hip	c	0.011863
	left u_{01}	-0.878168
	right u_{01}	-0.030273
Hip2	c	0.114678
	left u_{01}	0.079189
	right u_{01}	0.872256
	offset u_{01}	0.066414
Knee	c	0.010830
	left u_{01}	-0.039831
	right u_{01}	0.020860
	offset u_{01}	0.146311
Ankle1	c	0.035207
	left u_{01}	-0.062312
	right u_{01}	0.279817
Ankle2	c	0.001000
	left u_{01}	0.068465
	right u_{01}	0.007761
	offset u_{01}	0.014332

Table 13. The parameter of CPG for phase shift found by NEPSO.

Parameters	Range
Center to left hip1 phase γ	0.000435
Center to right hip1 phase γ	0.000000
Hip1 to hip2 phase phase γ	0.008514
Hip1 to knee phase phase γ	0.000000
Hip1 to ankle1 phase phase γ	0.039735
Hip1 to ankle2 phase phase γ	0.000000

the experience repository. The knowledge is used to estimate the fitness of the sampled particles. The sampling and the estimation of a particle is not done in the real environment but done in the experience repository that is implemented in a computer. The process accelerates the convergence of NEPSO and finds the CPG parameters of a biped robot within a few iterations. This is the reason that NEPSO outperformed the other PSOs.

The CPG parameters found by NEPSO are summarized in 12 and 13. Those parameters are for CPG equation (1)-(5) and the CPGs produces the target joint angle of the biped robot that makes the robot walk.

6. CONCLUSIONS

In this paper, we have presented a central pattern generator (CPG) parameter search for a biped walking robot using nonparametric estimation based particle swam optimization (NEPSO). The parameter search of CPG is a difficult problem but NEPSO found the parameters effectively by storing the previous knowledge of particles and estimating the best positions.

NEPSO has a fast convergence property which reduces evaluation of a fitness in a real environment and requires additional computation time compared to PSO variants. We expect that NEPSO can be applied to various applications requiring a long evaluation time compared to a calculation time.

REFERENCES

[1] K. Tanie, "Humanoid robot and its application possibility," *Proc. of the IEEE Conference of Multisensor Fusion Integration Intelligent Systems*, pp. 213-214. 2003.

[2] M. Vukobratović, B. Borovac, D. Surla, and D. Stokić, *Biped Locomotion-dynamics, Stability, Control and Application*, Springer-Verlag, 1990.

[3] K. Hirai, M. Hirose, Y. Haikawa, and T. Takenaka, "The development of Honda humanoid robot," *Proc. of IEEE International Conference on Robotics and Automation*, pp. 1321-1326, 1998.

[4] S. Kagami, T. Kitagawa, K. Nishiwaki, T. Sugihara, M. Inaba, and H. Inoue, "A fast dynamically equilibrated walking trajectory generation method of humanoid robot," *Autonomous Robots*, vol. 12, pp. 71-82, 2002.

[5] A. H. Cohen, "Control principle for locomotion looking toward biology," *Proc. of the 2nd International Symposium on Adaptive Motion of Animals and Machines*, pp. 41-51, 2003.

[6] K. Matsuoka, "Sustained oscillations generated by mutually inhibiting neurons with adaptation," *Biological Cybernetics*, vol. 52, pp. 345-353, 1985.

[7] A. J. Ijspeert and A. Crespi, "Online trajectory generation in an amphibious snake robot using a lamprey like central pattern generator model," *Proc. of IEEE International Conference on Robotics and Automation*, pp. 262-268, 2007.

[8] G. Taga, "A model of the neuro-musculo-skeletal system for human locomotion I. emergence of basic gait," *Biological Cybernetics*, vol. 73, pp. 97-111, 1995.

[9] G. Endo, J. Nakanishi, J. Morimoto, and G. Cheng, "An empirical exploration of a neural oscillator for biped locomotion control," *Proc. of the IEEE International Conference on Robotics and Automation*, vol. 3, pp. 3036-3042, 2004.

[10] G. Endo, J. Nakanishi, J. Morimoto, and G. Cheng, "Experimental studies of a neural oscillator for biped locomotion with QRIO," *Proc. of the IEEE International Conference on Robotics and Automation*, pp. 596-602, 2005.

[11] T. Reil and P. Husbands, "Evolution of central pattern generators for bipedal walking in a real-time physics environment," *IEEE Trans. on Evolutionary Computation*, vol. 6, pp. 159-168, 2002.

[12] H. Inada and K. Ishii, "Behavior generation of bipedal robot using central pattern generator (CPG)," *Proc. of the IEEE Conference on Intelligent Robots and Systems*, vol. 3, pp. 2179-2184, 2003.

[13] K. Wolff, J. Pettersson, A. Heralic, and M. Wahde, "Structural evolution of central pattern generators for bipedal walking in 3D simulation," *Proc. of IEEE International Conf. on Systems, Man and Cybernetics*, vol. 1, pp. 227-234, 2006.

[14] J. Shan, C. Junshi, and C. Jiapin "Design of central pattern generator for humanoid robot walking based on multi-objective GA," *Proc. of the IEEE Conference on Intelligent Robots and Systems*, vol. 3, pp. 1930-1935, 2000.

[15] S. Ok and D. S. Kim, "Evolving bipedal locomotion with genetic programming," *Lecture Notes in Computer Science*, vol. 3611, pp. 714-726, 2005.

[16] H. Kimura and Y. Fukuoka, "Biologically inspired adaptive dynamic walking in outdoor environment using a self-contained quadruped robot: Tekken2," *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, vol. 1, pp. 986-991, 2004.

[17] M. M. Williamson, "Neural control of rhythmic arm movements," *Neural Networks*, vol. 11, no. 7-8, pp. 1379-1394, 1998.

[18] M. M. Williamson, *Robot Arm Control Exploiting Natural Dynamics*, Thesis, Massachusetts Institute of Technology, 1999.

[19] J. Kennedy and R.C Eberhart, "Particle swarm optimization," *Proc. of IEEE International Conference on Neural Networks*, vol. 4, pp. 1942-1948, Nov. 1995.

[20] A. Abraham, H. Guo, and H. Liu, "Swarm intelligence: foundations, perspectives and applications," *Studies in Computational Intelligence,* Springer, vol. 26, pp. 3-25, Nov. 2006.

[21] Y. Fukuyama and H. Yoshida, "A particle swarm optimization for reactive power and voltage control in electric power systems," *Proc. of the IEEE Congress on Evolutionary Computation*, vol. 1, pp. 87-93, May 2001.

[22] T. Okada, T. Watanabe, and K. Yasuda, "Parameter tuning of fixed structure controller for power system stability enhancement," *Proc. of IEEE/PES Transmission and Distribution Conference and Exhibition*, vol. 1, pp. 162-167, Oct. 2002.

[23] Y. Zheng, L. Ma, L. Zhang, and J. Qian, "Robust PID controller design using particle swarm optimizer," *Proc. of IEEE International Symposium on Intelligence Control*, pp. 974-979, 2003.

[24] W.-C. Wu and M.-S. Tsai, "Feeder reconfiguration using binary coding particle swarm optimization," *International Journal of Control, Automation, and System*, vol. 6, no. 4, pp. 488-494, 2008.

[25] Y. Shi and R. C. Eberhart, "A modified particle swarm optimizer," *Proc. of the IEEE Congress on Evolutionary Computation*, pp. 69-73, 1998.
[26] J. Kennedy, "Small worlds and mega-minds: effects of neighborhood topology on particle swarm performance," *Proc. of the IEEE Congress on Evolutionary Computation*, pp. 1931-1938. 1999.
[27] R. A. Krohling, "Gaussian swarm: a novel particle swarm optimization algorithm," *Proc. of IEEE Conference on Cybernetics and Intelligent Systems*, vol. 1, pp. 372-376, Dec. 2004.
[28] E. Alpaydin, *Introduction to Machine Learning*, MIT Press, 2004.
[29] R. Smith, *Open Dynamics Engine*, http://www.ode.org, 2007.

Jeong-Jung Kim received the B.S. degree in Electronics and Information Engineering from Chonbuk National University in 2006 and the M.S. degree in Robotics from Korea Advanced Institute of Science and Technology in 2008. He is currently working toward a Ph.D. at the Korea Advanced Institute of Science and Technology. His research interests include biologically inspired robotics and machine learning.

Jun-Woo Lee received the B.S. degree in Electronics, Electrical and Communication Engineering from Pusan National University in 2007. He is currently working toward an M.S. in the Korea Advanced Institute of Science and Technology. His research interests include swarm intelligence and machine learning.

Ju-Jang Lee was born in Seoul, Korea, in 1948. He received the B.S. and M.S. degrees from Seoul National University, Seoul, Korea, in 1973 and 1977, respectively, and the Ph.D. degree in Electrical Engineering from the University of Wisconsin, in 1984. From 1977 to 1978, he was a Research Engineer at the Korean Electric Research and Testing Institute, Seoul. From 1978 to 1979, he was a Design and Processing Engineer at G. T. E. Automatic Electric Company, Waukesha, WI. For a brief period in 1983, he was the Project Engineer for the Research and Development Department of the Wisconsin Electric Power Company, Milwaukee. He joined the Department of Electrical Engineering, Korea Advanced Institute of Science and Technology, Daejeon, in 1984, where he is currently a Professor. In 1987, he was a Visiting Professor at the Robotics Laboratory of the Imperial College Science and Technology, London, U.K. From 1991 to 1992, he was a Visiting Scientist at the Robotics Department of Carnegie Mellon University, Pittsburgh, PA. His research interests are in the areas of intelligent control of mobile robots, service robotics for the disabled, space robotics, evolutionary computation, variable structure control, chaotic control systems, electronic control units for automobiles, and power system stabilizers. Dr. Lee is a member of the IEEE Robotics and Automation Society, the IEEE Evolutionary Computation Society, the IEEE Industrial Electronics Society, IEEK, KITE, and KISS. He is also a former President of ICROS in Korea and a Counselor of SICE in Japan. He is a Fellow of SICE and ICROS. He is an Associate Editor of IEEE Transactions on Industrial Electronics and IEEE Transactions on Industrial Informatics.