International Journal of Advancements in Mechanical and Aeronautical Engineering– IJAMAE
Volume 2: Issue 1 [ISSN : 2372-4153]
Publication Date : 30 April, 2015

Adaptive Control of Hybrid PSO-APGA using Neural Network for Constrained Real-Parameter Optimization

[Hieu Pham, Tam Bui, Hiroshi Hasegawa]

***Abstract*—This paper describes an evolutionary strategy called PSOGA-NN, which uses Neural Network (NN) for self-adaptive control of hybrid Particle Swarm Optimization and Adaptive Plan system with Genetic Algorithm (PSO-APGA) to solve large scale problems and constrained real-parameter optimization. This approach combines the search ability of all optimization techniques (PSO, GA) for stability of convergence to the optimal solution and incorporates concept from neural network for self-adaptive of control parameters. It is shown to be statistically significantly superior to other Evolutionary Algorithms (EAs) on numerical benchmark problems and constrained real-parameter optimization.**

Keywords—Adaptive Plan, Neural Network, Parallel Genetic Algorithm, Particle Swarm Optimization, Real-parameter

I. Introduction

Evolutionary Computation uses iterative process, such as growth or development in a population that is then selected in a guided random search using parallel processing to achieve the desired end. At present, the field of nature-inspired meta-heuristics is dominated by the Evolution Algorithms (EAs) (e.g., Genetic Algorithms (GAs) [1], Evolution Strategies (ESs), and Differential Evolution (DE)) as well as the Swarm Intelligence algorithms (e.g., Ant Colony Optimization (ACO), Particle Swarm Optimization (PSO) [2] [3], Artificial Bee Colony (ABC)). The field also extends in a broader sense to include self-organizing systems, artificial life, memetic and cultural algorithms, harmony search, artificial immune systems, and learnable evolution model.

Unlike most other techniques, GAs maintain a population of tentative solutions, which are competitively manipulated by applying various operators to find a global optimum. However, this process might requires high computational resources such as large memory and search times. To design efficient GAs, a variety of advances by new operators, hybrid algorithms, termination criteria, and more are continuously being achieved. Parallel GAs (PGAs) [4] [5] often leads to superior numerical performance, as well as faster algorithms. However, the truly interesting observation is that the use of structured population, either in the form of a set of islands or a diffusion grid, is responsible for these numerical benefits. A parallel genetic algorithm (PGA) is similar to a serial GA in terms of the representation of the problem parameters, robustness, easy customization, and multi-solution capabilities. However, a PGA is usually faster, less prone to finding sub-optimal solutions only, and capable of cooperating with other search techniques in parallel.

Particle Swarm Optimization (PSO), first introduced by Kennedy and Eberhart [2], is one of the modern meta-heuristics algorithms. It was developed based on the simulation of a simplified social system, and it has been found to be robust in solving optimization problems. PSO uses simple iterative calculations, so it is easy to create the program source. Thus, PSO is applicable to a wide range of optimization problems. However, the performance of the PSO greatly depends on its parameters and it often becomes trapped in local optimum. To resolve this problem, various improvement algorithms have been proposed that can solve a variety of optimal problems [3].

To reduce the cost and to improve the stability, a strategy that combines global and local search methods becomes necessary. As for this strategy, Hieu Pham et al. developed a new approach for Adaptive Plan system of swarm intelligent using PSO with GA (PSO-APGA) [6].

In this paper, we purposed a new evolutionary algorithm called PSOGA-NN, which uses Neural Network (NN) with Migration PGA for Self-adaptive control parameters of hybrid PSO-APGA to solve large scale optimization problems and constrained real-parameter optimization.

The remainder of this paper is organized as follows. Overviews of related works such as PSO-APGA algorithm, Migration PGA concept, and a mathematical model of Neural Network are provided in Section II. Section III describes the algorithm of proposed strategy (PSOGA-NN), and finally Section IV discusses about robustness using numerical benchmark problems and constrained real-parameter optimization.

Hieu Pham
Hanoi University of Science and Technology
Vietnam

Tam Bui
Shibaura Institute of Technology
Japan

Hiroshi Hasegawa
Shibaura Institute of Technology
Japan

II. Related Work

A. Formulation of the Optimization Problem

The optimization problem is formulated as follows:

Design variable:

$$x = [x_1, \ldots, x_D] \tag{1}$$

Objective function:

$$f(x) \rightarrow Min \tag{2}$$

Modified objective function:

$$f^*(x) = f(x) + \gamma P(x) \rightarrow Min \tag{3}$$

Inequality constraint functions:

$$g_j(x) \le 0; \qquad j = [1, \ldots, r] \tag{4}$$

Equality constraint functions:

$$h_k(x) \le 0; \qquad k = [1, \ldots, s] \tag{5}$$

Design range:

$$x^{LB} \le x \le x^{UB} \tag{6}$$

where $f(x)$, $f^*(x)$, γ denote objective function, modified objective function, and penalty coefficient, respectively. $X^{LB} = [x_1^{LB}, \ldots, x_n^{LB}]$, $X^{UB} = [x_1^{UB}, \ldots, x_n^{UB}]$ and D denote the lower boundary condition vectors, the upper boundary condition vectors and the number of design variable vectors (DVs) respectively. r and s are the number of inequality and equality constraints, g_j and h_k are linear or nonlinear real-value functions.

B. Hybrid PSO-APGA

1) Algorithm.

PSO-APGA aims at getting the direction from PSO to adjust in to adaptive system of APGA. This method introduces a design variable (DV) generation formula using the velocity update from PSO operator and sensitivity analysis. Adaptive plan (AP) generates next values of DVs by using control variables (CVs), response variables (RVs), velocity PSO and current values of DVs in the adaptive system (AS) of optimization process. The DV generation process generates a new DV from the current search point via AP according to the formula:

$$x_{i,G+1} = x_{i,G} + AP(C_{i,G}, R_{i,G}) \tag{7}$$

where $AP()$, x, C, R, and G denote a function of AP, DVs, CVs, RVs and generation, respectively.

The flowchart of PSO-APGA is shown in Fig. 1. In this approach, PSO and APGA run individually. The iteration is run by PSO operator for local search, the velocity update is given as initial parameter for APGA process and procedure offspring. As the new population, choose the best position as the global by estimating their fitness. The best scoring individual in the population is taken as the global optimal solution.

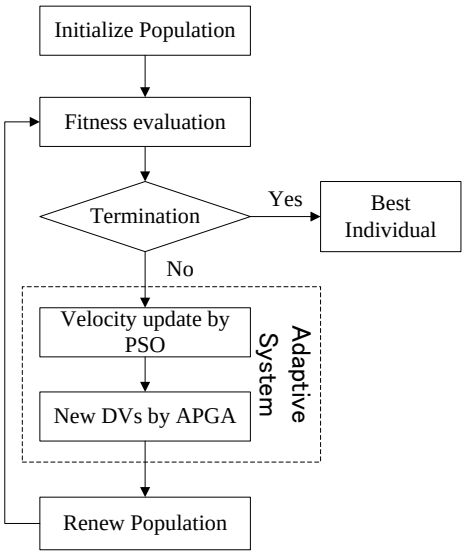

Figure 1. Flow-chart of PSO-APGA.

2) Adaptive Plan.

It is necessary that the AP realizes a local search process by applying various heuristics rules. In this paper, the plan introduces a DV generation formula using velocity update from PSO operator that is effective in the convex function problem as a heuristic rule. This plan uses the following equation:

$$AP(C_{i,G}, R_{i,G}) = ANR \cdot (\nabla R) \tag{8}$$

$$ANR = NR_{i,G} \cdot (scale) \cdot PSO \tag{9}$$

$$PSO = v_{ij,G+1} = \omega v_{ij,G} + c_1 r_1 (pbest_{ij,G} - x_{ij,G}) + c_2 r_2 (gbest_{j,G} - x_{ij,G}) \tag{10}$$

where ∇R denote sensitivity of RVs, *ANR* is adaptive factor defined by multiplying from neighborhood ratio *NR* with constriction factor *scale* and velocity update *PSO*. ω is inertia weight; c_1 and c_2 are cognitive acceleration and social acceleration, respectively; r_1 and r_2 are random numbers uniformly distributed in the range [0.0, 1.0]. *pbest* is best solution has been achieved so far and *gbest* is the best solution of all particles (where j = [1,...,D], D is the dimension of the solution vector)

$C = [c_{i,j},\ldots,c_{i,p}]$; $0.0 \le c_{i,j} \le 1.0$ is used so that it can change the direction to improve or worsen the objective function, and C is encoded into a chromosome by 10 bit strings (shown in Fig. 2). In addition, i, j and p are the individual number, design variable number and its size, respectively.

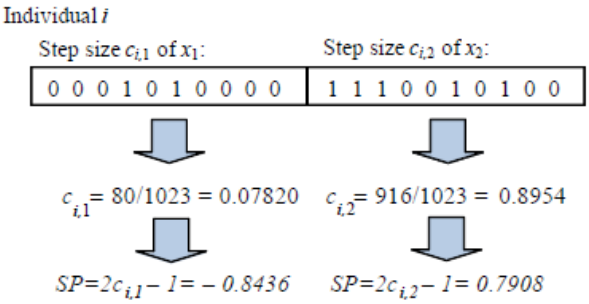

Figure 2. Step size that defined by CVs.

C. *Migration PGA*

Parallel processing has been promoted into evolutionary algorithms due to changes in resources such calculations [4] [5]. This approach can be applied to GA in many different ways, and the literature contains many examples of successful parallel implementations. The classification of parallel GA includes three main different types as global single population master-slave PGA, single population fine-grained PGA, and multiple population coarse-grained PGA [7].

Multiple-population (or multiple-deme) are more sophisticated, as they consist on several sub-populations which exchange individuals occasionally. First, the population is divided into a plurality of sub-populations. Next, in the respective sub-population, the GA repeats selection operators, such as crossover. Then, several search points are drawn from several sub-populations, and then moved to another population. This operation of individuals is called immigration (migration). There are two methods of immigration, where two types of asynchronous population are migrated to other asynchronous source and all subpopulations simultaneously perform the migration, which requires certain conditions. It is necessary to set the migration rate as the path of a number of search points to emigrate or move to a sub-population, which must be synchronous, and the migration interval generation time of the migration period (Fig. 3).

Multi-deme parallel GA are known as distributed GA, because they are usually implemented on distributed computers. Since the high communication ratio, they are occasionally called coarse-grained GAs. Finally, they resemble the "island model" in population genetics which considers relatively isolated demes.

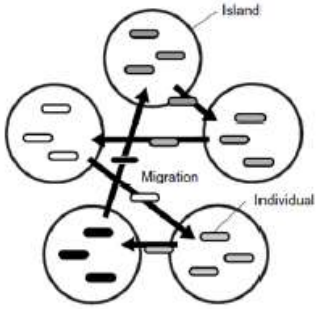

Figure 3. Migration PGA.

III. New Evolution Strategy

A. *Migration PGA in PSOGA-NN*

Migration PGA, such as that described earlier in Section II, is reported compatibility information, a stable design and low computational cost because it deals with GA in parallel. In PSOGA-NN, optimization is conducted by applying GA and PSO in each sub-population. The control variables adjust the vicinity of the output factor *NR* between the sub-populations. The candidate control variables and the new solution come from the other sub-population at the time of immigration, so a diversity of solutions can be expected because the migration destination is determined at random. A schematic diagram of PSOGA-NN with Migration PGA is shown in Fig. 4.

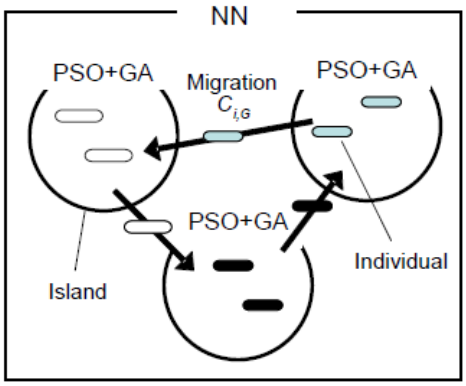

Figure 4. Migration PGA in PSOGA-NN.

B. Self-adaptive using Neural Network

The self-adaptive of neighborhood factor *NR* is used for data clustering of GA control variables using NN, which have been uniquely determined to stabilize their variation. From a viewpoint of excellent parallel processing and to ensure compatibility with multi-point search methods such as PSO and GA also be used in the present method, NN is often used in combination with these techniques [8]. NN may be used to cluster and classify the data without using a signal if it is necessary to learn using a teacher signal that is also a NN. In the present method, the GA variable data clustering is controlled using unsupervised learning to determine the output factor change. The initial neighborhood factor *NR* is set at random and we vary its value based on the NN output. The unsupervised learning method is also a multi-layer NN, so we use NN to perform the feed-forward transfer. The number of layers is determined in a number of search points for each subpopulation. In addition, the NN is configured after it has been sorted in descending order of fitness in the subpopulations to the output side from the input side, where the weight of the transfer equation is as shown in (13). Therefore, many subpopulations have highly adaptive search points with strong effects on other subpopulations. The formulation of the control variable, the transfer equation for each node in the NN and the schematic diagram of the overall NN are as follows.

$$node_t = \sum_{i=1}^{I} SP \cdot w_{i^n j^n} out_i^{n-1} \Big/ I \tag{11}$$

$$SP = 2 \cdot C_{i,G} - 1 \tag{12}$$

$$w_{j^n i^n} = y_{nm} \Big/ y_{(n-1)m} \tag{13}$$

$$NR_{i,G+1} = NR_{i,G} - \nabla NR_i \tag{14}$$

The GA handles control variables (CVs) and *C* is allocated to each search point, which is encoded as a 10-bit string. The order of each search point is allocated to each node of a multi-layer NN, as shown in Fig. 5, on the input side and the output side. The weight of the NN, w_{jnin}, which is determined from the adaption ratio of the search points, is transmitted between the nodes. *C* is the control variable that determines the step size *SP* in (12) (as shown in Fig. 2) and this element determines the amount of the neighborhood factor change ∇NR. Therefore, the neighborhood factor change is an important factor, which determines the width of the overall distribution of the neighborhood of search points. Using the control variable, we can change adaptively *NR* to facilitate more stable solution search and better control of the control variable in the NN. In addition, *n* is the number of hierarchy NN, *m* is the number of sub-populations, *j*, *i* is the number of neurons in NN, *t* is the number of individuals, *S* is the number of search per island, and *I* is the maximum number of islands.

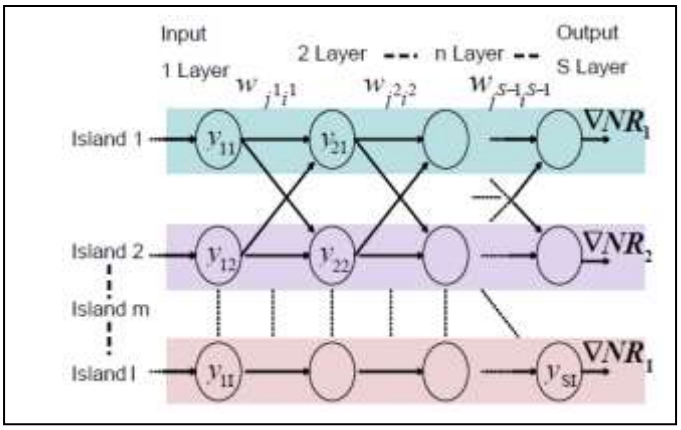

Figure 5. PSOGA-NN Neural Network.

C. Elite Strategy

In this paper, using the diploid genetics is not proper to perform the search using the NN solution [9]. Generally, GA, information has only a single gene for one individual. However, the structure has a double recessive genetic information that does not appear in the dominant phenotype. Here, in NN, genetic information is treated as a control variable. Information dominance for the NN is elite solution closed to the control variable, as shown in the following equation. With the aim of having a strong influence in the form of dominant inheritance, enhancing the effectiveness of the control variable, advantageously advancing the solution search, elite solution against other sub-populations as the PGA with migration.

$$\begin{aligned} &if\ |eSP - SP_1| - |eSP - SP_2| < 0\ \ SP = SP_1 \\ &if\ |eSP - SP_1| - |eSP - SP_2| > 0\ \ SP = SP_2 \end{aligned} \tag{15}$$

D. Reconstruction of PSO Velocity

We carried out the reconstruction of the control variable like considered control variable of PSO-APGA, not only control variable meet the conditions listed below, but also reconstruction of the velocity update by keep performing keep the global search of the search point, the appropriate solution search is always performed.

- The same value adaptation accounted for more than 80% for the entire.
- The same bit-string chromosome occupies more than 80% for the entire.
- The same value of neighborhood factor accounted for 50% of the total.

IV. Experiments

A. Numerical Benchmark Problems

In this section, the numerical experiments are performed to compare with other techniques for the robustness of the optimization approach using five benchmarks with 30 dimensions - Ridge (f_3), Rosenbrock (f_5), Rastrigin (f_9), Ackley (f_{10}), and Griewank (f_{11}). The parameter settings in solving the benchmark tests are given in Table 1.

$$RI: f_3 = \sum_{i=1}^{n} \left(\sum_{j=1}^{i} x_j \right)^2 \tag{16}$$

$$RO: f_5 = \sum_{i=1}^{n-1} \left[100\left(x_{i+1} + 1 - (x_i + 1)^2\right)^2 + x_i^2 \right] \tag{17}$$

$$RA: f_9 = 10n + \sum_{i=1}^{n} \left\{ x_i^2 - 10\cos(2\pi x_i) \right\} \tag{18}$$

$$AC: f_{10} = -20\exp\left(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2}\right) - \exp\left(\frac{1}{n}\sum_{i=1}^{n}\cos(2\pi x_i)\right) + 20 + e \tag{19}$$

$$GR: f_{11} = 1 + \sum_{i=1}^{n} \frac{x_i^2}{4000} - \prod_{i=1}^{n} \cos\left(\frac{x_i}{\sqrt{i}}\right) \tag{20}$$

TABLE I. PARAMETER SETTINGS

Operator	Settings	
	Control Parameter	*Value*
PSO	Inertia Weight	$\omega = 0.7$
	Acceleration Coefficients	$c_1 = c_2 = 0.8$
	Constriction Factor	*scale* = 0.729
GA (NN)	Island number	10
	Imigration rate	0.2
	Selection	1.0
	Crossover	0.8
	Mutation	0.1

B. Constrained Real-parameter Optimization

We shall apply PSOGA-NN to solve real constrained engineering design optimization. Speed reducer design [10] was chosen to evaluate its performance.

The design of a speed reducer is a more complex case study. This problem involves seven design variables, with the face width b (x_1), module of teeth m (x_2), number of teeth on pinion z (x_3), length of first shaft between bearings l_1 (x_4), length of second shaft between bearings l_2 (x_5), diameter of first shaft d_1 (x_6), and diameter of second shaft d_2 (x_7). The objective is to minimize the total weight of the speed reducer. There are nine constraints, including the limits on the bending stress of the gear teeth, surface stress, transverse deflections of shafts 1 and 2 due to transmitted force, and stresses in shafts 1 and 2. The mathematical formulation can be summarized as follows:

Minimize:

$$f(x) = 0.7854x_1x_2^2\left(3.3333x_3^2 + 14.9334x_3 - 43.0934\right) - 1.508x_1\left(x_6^2 + x_7^2\right) + 7.4777\left(x_6^3 + x_7^3\right) + 0.7854\left(x_4x_6^2 + x_5x_7^2\right) \tag{21}$$

subject to:

$$g_1(x) = \frac{27.0}{x_1x_2^2x_3} - 1.0 \le 0 \tag{22}$$

$$g_2(x) = \frac{397.5}{x_1x_2^2x_3^2} - 1.0 \le 0 \tag{23}$$

$$g_3(x) = \frac{1.93x_4^3}{x_2x_3x_6^4} - 1.0 \le 0 \tag{24}$$

$$g_4(x) = \frac{1.93x_5^3}{x_2x_3x_7^4} - 1.0 \le 0 \tag{25}$$

$$g_5(x) = \frac{1.0}{110.0x_6^3}\sqrt{\left(\frac{745.0x_4}{x_2x_3}\right)^2 + 16.9 \times 10^6} - 1.0 \le 0 \tag{26}$$

$$g_6(x) = \frac{1.0}{85.0x_7^3}\sqrt{\left(\frac{745.0x_5}{x_2x_3}\right)^2 + 157.5 \times 10^6} - 1.0 \le 0 \tag{27}$$

$$g_7(x) = \frac{x_2 x_3}{40.0} - 1.0 \le 0 \quad (28)$$

$$g_8(x) = \frac{5.0 x_2}{x_1} - 1.0 \le 0 \quad (29)$$

$$g_9(x) = \frac{x_1}{12.0 x_2} - 1.0 \le 0 \quad (30)$$

$$g_{10}(x) = \frac{1.5 x_6 + 1.9}{x_4} - 1.0 \le 0 \quad (31)$$

$$g_{11}(x) = \frac{1.1 x_7 + 1.9}{x_5} - 1.0 \le 0 \quad (32)$$

with

$$2.6 \le x_1 \le 3.6, 0.7 \le x_2 \le 0.8, 17 \le x_3 \le 28,$$
$$7.3 \le x_4 \le 8.3, 7.3 \le x_5 \le 8.3, 2.9 \le x_6 \le 3.9, \quad (33)$$
$$5.0 \le x_7 \le 5.5$$

Best solution:

$$x^* = (3.5, 0.7, 17, 7.3, 7.8, 3.350214, 5.286683)$$

where $f(x^*) = 2{,}996.384165$

C. *Evaluation*

The experiment results are given in Table II and Table III. "Mean" indicates average of optimum values obtained and "Std. Dev" stands for standard deviation. The results confirmed that this strategy can reduce the computational costs dramatically and significantly improve the stability of convergence on the optimal solution. Overall, PSOGA-NN was capable of attaining robustness, high quality, low calculation on many optimization problems.

TABLE II. AVERAGE RESULT OVER 25 RUNS BY PSOGA-NN (30 DIMENSIONS, POPULATION SIZE 50)

Function	Benchmark Tests			
	Gen.	*NFE*	*Mean*	*Std. Dev*
f_3	54	2,700	0.00E+00	0.00E+00
f_5	1125	56,250	0.00E+00	0.00E+00
f_9	133	6,650	0.00E+00	0.00E+00
f_{10}	159	7,950	4.44E-16	0.00E+00
f_{11}	105	5,250	0.00E+00	0.00E+00

TABLE III. RESULT OF APPLYING PSOGA-NN FOR SPEED REDUCER DESIGN OPTIMIZATION PROBLEM (POPULATION 56)

Speed Reducer Problem			
Solution		*Constraints*	
x_1	3,500008	g_1	-0.073918
x_2	0,700000	g_2	-0.198002
x_3	17,000014	g_3	-0.977783
x_4	7,300133	g_4	-0.901471
x_5	7,800016	g_5	-0.000009
x_6	3,350225	g_6	-4.4E-07
x_7	5,286684	g_7	-0.702500
		g_8	-0.000002
$f(x)$	2996,358	g_9	-0.583332
		g_{10}	-0.051341
NFE	7,840	g_{11}	-0.010854

References

[1] D.E. Goldberg, Genetic Algorithms in Search Optimization & Machine Learning, Addison-Wesley, 1989.

[2] J. Kennedy and R. Eberhart, Swarm Intelligence. Morgan Kaufmann Publishers, 2001.

[3] M. Clerc, Particle swarm optimization, ISTE, 2005.

[4] E. Cantu-Paz, "A survey of parallel genetic algorithms. Calculateurs Paralleles," 10(2), 1998.

[5] E. Alba and J.M. Troya, "A Survey of Parallel Distributed Genetic Algorithms. Complexity," 4(4), 1999, pp.31–52.

[6] H. Pham, H. Hasegawa, "Adaptive system of swarm intelligent with Genetic Algorithm for global optimization," IEEE International Conference on Systems, Man, and Cybernetics (SMC), 2012, pp.171-176.

[7] R. Tanese, "Distributed genetic algorithms," Proc. of 3rd Int. Conf. on Genetic Algorithms, 1989, pp. 434–439.

[8] K. Kobayashi, T. Hiroyasu, and M. Miki, "Mechanism of Multi-Objective Genetic Algorithm for Maintaining the Solution Diversity Using Neural Network," The Science and Engineering Review of Doshisha University, 48(2), 2007, pp.24–33.

[9] M. Kouchi, H. Inayoshi, and T. Hoshino, "Optimization of Neural-Net Structure by Genetic Algorithm with Diploydi and Geographical Isolation Model," Japanese Society for Artificial Intelligence, 7(3), 1992, pp.509–517.

[10] J. Golinski, "An Adaptive Optimization System Applied to Machine Synthesis. Mech. Mach. Theory," 8(4), 1973, pp.419-436.

About Author:

Hieu Pham received the B.E. in Mechatronics from Hanoi University of Science and Technology, Vietnam, in 2007, and the M.E. from Shibaura Institute of Technology, Japan, in 2010. He received his PhD degree in Functional Control Systems from Shibaura Institute of Technology, Japan, in 2013. He is currently working as a lecturer in Hanoi University of Science and Technology, Vietnam. His research interests include optimal systems design, computational intelligence, robotics, and neuroscience.