Parallel Elite Genetic Algorithm and Its Application to Global Path Planning for Autonomous Robot Navigation

Ching-Chih Tsai, *Senior Member, IEEE*, Hsu-Chih Huang, *Member, IEEE*, and Cheng-Kai Chan, *Student Member, IEEE*

***Abstract*—This paper presents a parallel elite genetic algorithm (PEGA) and its application to global path planning for autonomous mobile robots navigating in structured environments. This PEGA, consisting of two parallel EGAs along with a migration operator, takes advantages of maintaining better population diversity, inhibiting premature convergence, and keeping parallelism in comparison with conventional GAs. This initial feasible path generated from the PEGA planner is then smoothed using the cubic B-spline technique, in order to construct a near-optimal collision-free continuous path. Both global path planner and smoother are implemented in one field-programmable gate array chip utilizing the system-on-a-programmable-chip technology and the pipelined hardware implementation scheme, thus significantly expediting computation speed. Simulations and experimental results are conducted to show the merit of the proposed PEGA path planner and smoother for global path planning of autonomous mobile robots.**

***Index Terms*—Elite genetic algorithm, global path planning, mobile robot, navigation, parallel processing.**

I. Introduction

RECENTLY, there has been increasingly interesting in the challenging field of how to apply optimization algorithms to solve global path planning problems. Global path planning is an important problem in many disciplines, including very-large-scale integration design, global positioning system applications, and autonomous robot navigation. These global path planning problems have been solved using many existing approaches, such as cell decomposition [1], [2], skeleton, and potential field [3]. In addition, soft computing approaches, such as fuzzy logic, neural networks, and evolutionary algorithm, have been widely used to solve the global path planning problems. Jung *et al.* [4] presented the neural networks approach for path planning for mobile robots. The fuzzy logic approach to solving the path planning problem was proposed in [5]. Hocaoglu *et al.* [6] used an evolutionary algorithm for solving the optimization problem. Generally, these conventional methods were shown to lack of computational complexity, local minimum, adaption, and nonrobust behavior [7].

Among those existing methods to solve the global path planning optimization problems, genetic algorithm (GA) has been recognized as a heuristic and powerful robust optimization technique by simulating natural evolution over populations [8]. GA was introduced by Holland [9] and has been proven powerful in finding the optimal path by exploiting its strong optimization ability [10]. Such ability hinges on the advantages of both deterministic and probabilistic schemes to improve solutions using simple operators, such as reproduction, crossover, and mutation. Yue *et al.* [11] used GA to point-to-point trajectory planning of flexible redundant robot manipulator and Taharwa *et al.* [7] presented how to use GA to solve the path planning problem in static environment for a mobile robot. An improved GA for path planning of robot under unknown environment is introduced by Lin *et al.* [12]. Moreover, Chen and Zalzala [13] proposed a GA to motion planning of redundant mobile manipulator systems. Manikas *et al.* [14] proposed gyroscopes and global positioning systems incorporated with GA for autonomous robot navigation. However, GA suffers from their complicated computations so that it is not suitable for real-time applications. The field-programmable gate array (FPGA) implementation of GA to shorten the computing time was presented in [15]. However, these conventional GAs [9]–[15] possibly converge to a local optimum and may get stuck for a long time, called premature convergence. This disadvantage of serial algorithm can be avoided by using the parallel processing.

Parallel architectures have been widely used to accelerate various computational algorithms by using distributed processing. They can be roughly divided into four main models: a) global (master-slave) model; b) coarse-grain model; c) fine-grain model; and d) hybrid model (global plus coarse-grain) [16]. Among them, the coarse-grain parallel model has been shown particularly useful and pragmatic for chip-based implementation of a parallel GA, thereby expediting GA computing [16]. Moreover, this kind of parallel model with system-on-a-programmable-chip (SoPC) implementation has potential capability to solve optimal problems much faster than software implementations do. Hence, the premature convergence problem of conventional GAs can be circumvented by combining techniques of the coarse-grain model and the elite GA. Moreover, the kind of parallel computing method, called

Manuscript received February 11, 2010; revised August 22, 2010; accepted December 21, 2010. Date of publication January 28, 2011; date of current version August 30, 2011. This work was supported by the National Science Council, Taiwan, under Grants NSC 97-2628-E-005-004-MY3 and NSC 99-2628-E-241-003.

C.-C. Tsai and C.-K. Chan are with the Department of Electrical Engineering, National Chung-Hsing University, Taichung 402, Taiwan (e-mail: cctsai@nchu.edu.tw; bossmaster_chan@hotmail.com).

H.-C. Huang is with the Department of Computer Science and Information Engineering, HungKuang University, Taichung 433, Taiwan (e-mail: hchuang@sunrise.hk.edu.tw).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TIE.2011.2109332

parallel elite genetic algorithm (PEGA), is helpful not only increasing the diversity of searching space, but also significantly expediting the computation speed.

Although the collision-free path can be easily obtained from the proposed PEGA, the resultant path is composed of a sequence of line segments. Unfortunately, such a discontinuous path is not good from control perspectives for autonomous mobile robots because the robots must stop at each discontinuous corner. Hence, these discontinuous segments must be deformed or smoothed. The B-spline modeling is one of the most efficient curve interpolations and widely applied in many disciplines, including computer graphics [17], computer-aided design [18], image processing [19], signal processing [20], and robotics [21]. With the attractive properties of the B-spline curve fitting method, such as spatial uniqueness, boundedness and continuity, local shape controllability, and structure preservation under affine transformation [17]–[21], this interpolation scheme is practically useful for path smoothing of autonomous mobile robots. However, most studies only focused on the PC- or workstation-based B-splines [17]–[21]. To date, no attempt has been made to integrate PEGA algorithm, coarse-grain parallel architecture, and B-spline path smoothing for developing a new and efficient SoPC-based PEGA to global path planning and path smoothing for autonomous robot navigation.

The objective of this paper is to develop a coarse-grain PEGA and its application to solve the global path planning problem for autonomous robot navigation. Both path planner and B-spline path smoother are constructed in one FPGA chip to find the near-optimal global smooth path. The proposed techniques will be proven more effective to solve these problems than the conventional software-based GAs [9]–[14] or nonparallel FPGA-based GAs [15], [22] do. Overall, the three contributions of the paper are given as follows. First, a coarse-grain PEGA is proposed to increase the diversity of searching space and decrease the probability of convergence in a local optimum; this method is more likely to find the global optimum than the conventional GAs [9]–[15] do. Second, a pipelined PEGA implementation using the hardware/software co-design scheme and the SoPC technique is presented to significantly shorten its computation time. Third, both SoPC-based PEGA path planner and B-spline path smoother are constructed in the same chip to efficiently obtain an optimal collision-free continuous path for autonomous robot navigation.

The rest of this paper is organized as follows. In Section II, the coarse-grain PEGA is proposed to significantly expedite the computing speed and diverse the searching space. Section III elaborates the FPGA implementation of the proposed PEGA using the SoPC technology. Section IV elucidates the procedure of how to apply the PEGA together with the B-spline method to find a globally near-optimal continuous path for autonomous mobile robots. Section V conducts several simulations and experimental results to show the performance and merit of the proposed methods. Section VI concludes this paper.

II. PEGA

This section is aimed at presenting an efficient coarse-grain PEGA for global path planning. The EGA is modified from GA with several key genetic operators, such as selection, crossover, and mutation. In addition, the elite policy and diversity increasing strategy are employed to circumvent the premature convergence in conventional GAs. Furthermore, by taking the advantages of SoPC technology, the coarse-grain parallel EGA will significantly expedite computing speed for global path planning.

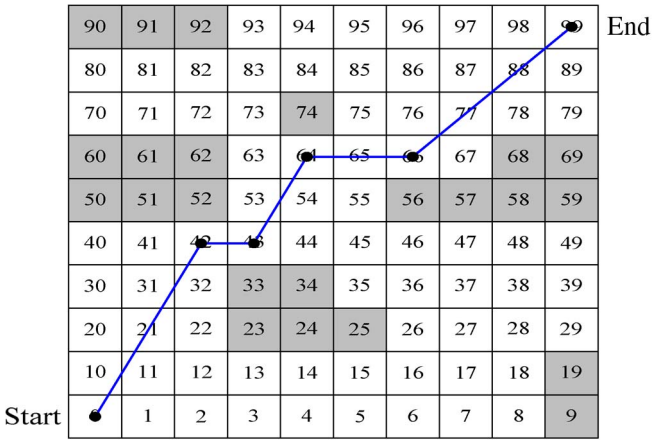

Fig. 1. Grid-based environment with obstacles and the planned collision-free path.

A. EGA Computing for Global Path Planning

In GA-like computing for global path planning, a chromosome contains the set of the grid points of the trajectory, including a start point, via points, and an endpoint shown in Fig. 1. The optimal collision-free path is evolved by the genetic operators, such as selection, crossover, and mutation [9]–[15]. However, the conventional GAs exist the following two drawbacks. The first one is that the best solution may have the chance to become worse in the next generation, thus slowing down the convergent speed. The second one is premature convergence before the global optimal solution has been found. To circumvent these shortcomings, an EGA is proposed in this subsection. It contains reproduction strategy with elite policy and diversity increasing in population pool, thereby making EGA a powerful and effective algorithm for global path planning. The following briefly describes the core genetic operators of the proposed EGA.

1) Reproduction (Selection): The primary objective of the reproduction is to duplicate good solutions and eliminate bad solutions in a population, thus keeping the population size constant. This module is applied to select individuals from the population so that these chosen individuals can be sent to the crossover and mutation modules in order to attain new offsprings. The selection policy is ultimately responsible for ensuring survival of the best fitted individuals. Among several selection methods, such as roulette selection, rank selection, and tournament selection, the tournament selection is used.

2) Crossover: Crossover is the fundamental mechanism of genetic rearrangement and is applied next to the selection in EGA. Although there are various crossover schemes, such as one-point, two-point, uniform, and arithmetic crossover, one-point crossover is adopted throughout this paper due to high-speed operation. The crossover site is randomly determined and

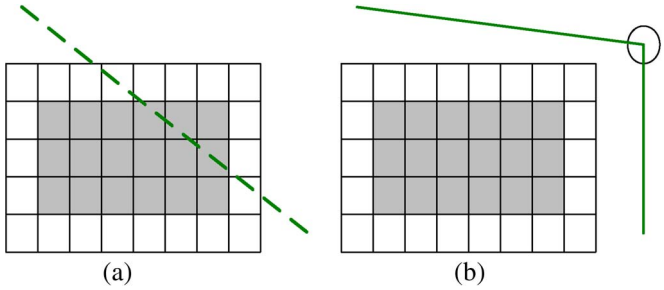

Fig. 2. (a) Path before addition operation. (b) Path after addition operation.

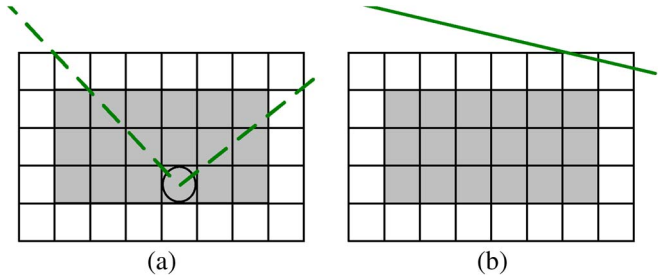

Fig. 3. (a) Path before deletion operation. (b) Path after deletion operation.

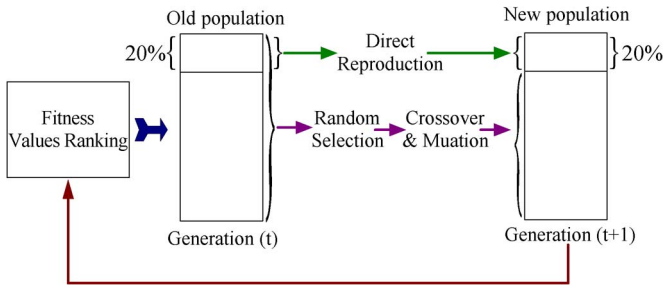

Fig. 4. Reproduction strategy with elite policy.

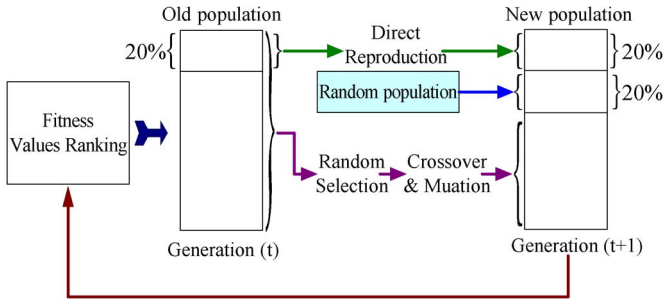

Fig. 5. Diversity increasing in population pool.

some portions of the strings are exchanged between the two solutions to create new solutions.

3) Mutation: Mutation is a process that consists of making small alterations to the bits of the chromosomes by applying some kind of randomized changes. This operator is necessary for maintaining certain diversity in the population, thus avoiding quick convergence to a local minimum. Moreover, two special operations are employed in the proposed EGA, including addition and deletion. Addition is used to repair a path by inserting a node between any two nodes of a segment. Fig. 2 depicts that this operation could make the length of the path longer. Deletion is applied by randomly selecting one node and then deleting it. As shown in Fig. 3, the deletion process is able to attain feasible paths with smaller fitness.

4) Fitness Function: The fitness function is application-specific and is always designed according to the problem to be optimized. The fitness of new chromosomes from genetic operations should be evaluated based on the fitness function. For the global path planning problem, the fitness of each chromosome is formulated in terms of its path length.

5) Reproduction Strategy With Elite Policy: Crossover is an important operator that helps EGA to find the optimal solution. However, the best solution may have a chance to be destroyed. The offsprings may have worse fitness values than their parents do, thus reducing the convergent speed. In order to circumvent the problem, the elite preservation policy is taken into account. As shown in Fig. 4, 20% of the best performing individuals are reproduced directly to the next generation, thereby guaranteeing at least that the offspring will perform as good as their parents. The rest of the new generations are randomly selected from the population pool and produced by the crossover and mutation operations.

6) Increasing Diversity in Population Pool: One of the main problems for conventional GAs is premature convergence before the global optimal solution has been found. The reason of premature convergence is that the individuals in the population pool are too alike. To ensure that members in different generations are not identical in a long run, it is necessary to increase the diversity in population pool by adding random population as shown in Fig. 5. Most importantly, random selection to crossover and mutation makes the remaining population difficult to convergence. Combining diversity increasing and random selection in the proposed EGA, the chromosomes have a more powerful ability to search for the global optimal solution than conventional GAs do.

B. Coarse-Grain PEGA

Although the EGAs mentioned in Section II-A has been shown to find a better optimal solution than conventional GAs do, these algorithms require many computations and iterations that cause enormous time consumption, thus resulting in that such algorithms are not good for real-time applications. Conversely, the computing time can be significantly improved by applying parallel processing which has been widely used to expedite various computational algorithms by using distributed processing [16]. Among several distributed processing models, the coarse-grain parallel model has been shown useful for chip-based implementation of the PEGA [16].

To accelerate the computation time of the coarse-grain PEGA, this subsection intends to design a coarse-grain PEGA for achieving high-speed computations. Fig. 6 depicts the architecture of the proposed high-speed coarse-grain PEGA, comprising two EGAs concurrently executed on two separate EGA processors with data interchange. Compared to conventional EGA, this PEGA gains benefit of distributed computations in which more searching space can be explored and much faster computing speed is employed to find an optimal solution. Hence, this proposed algorithm will significantly increase its

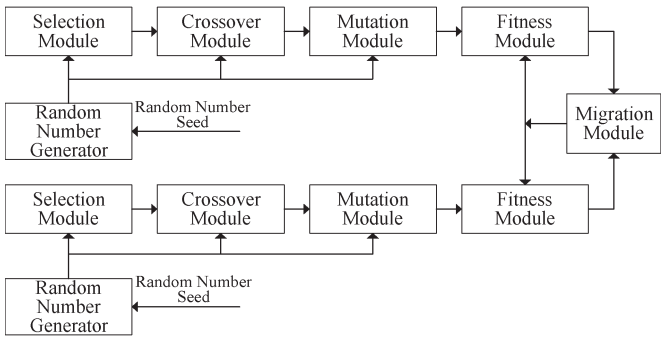

Fig. 6. Block diagram of the proposed coarse-grain PEGA using two nodes.

possibility to search for a global optimum and expedite the computing.

The proposed PEGA has been implemented on two nodes, each of which has its own EGA algorithm. Although the two nodes perform the EGA operation independently and separately, a migration operator is designed to allow them to exchange their best population. The searching spaces for both nodes are independent, and the populations on the nodes evolve separately for a certain number of generations, called isolation time. After the isolation time, a number of the best sub-populations (migration rate) from the two nodes are exchanged via the migration operator. Accordingly, the migration operator effectively increases the diversity among the individuals in each sub-population and significantly decreases the probability of parallel EGA to get stuck into a local optimum.

III. SOPC Implementation

As mentioned before, the software implementation of PEGA for applying large searching spaces may cause unacceptable delays due to enormous computations and iterations. An alternative to this approach is the hardware implementation of PEGA in order to achieve tremendous speedup over software implementation. Accordingly, this section presents the SoPC implementation of the proposed PEGA using hardware/software co-design technique in one FPGA chip.

Fig. 7 depicts the architecture of the SoPC implementation for the proposed coarse-grain PEGA using two nodes. Worthy of mention is that the PEGA fitness module can be implemented into the 32-bit Nios II processor in order to realize a desired fitness function. The user IP cores (custom logic) for this PEGA operators have been developed by VHDL (VHSIC Hardware Description Language), including random number generator (RNG), selection, crossover, mutation, and migration module. The software-based fitness module and hardware-based custom logics for the PEGA are connected to the system interconnect fabric via Avalon memory-mapped interface in one FPGA chip.

The following briefly describes how to design several modules in the proposed SoPC-based PEGA as shown in Fig. 7; they are RNG, selection, crossover, mutation, fitness, and migration modules. The software-based fitness module running in the embedded processor has the flexibility for realizing different fitness functions, whereas the VHDL hardware-based modules in FPGA exploit the features of pipelining and parallelization.

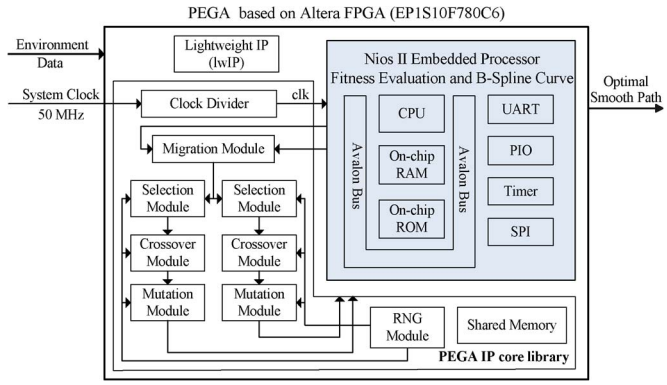

Fig. 7. SoPC-based coarse-grain PEGA using two nodes.

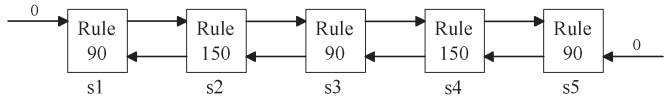

Fig. 8. Example of the cellular automata RNG.

A. RNG Module

This pseudorandom number generator (PRNG) is a crucial module required in the proposed parallel EGA. The module generates a sequence of pseudorandom bits for executing the crossover, mutation, and selection modules. Although there are two methods [linear feedback shift register (LFSR) and cellular automata (CA)] commonly used for the RNG module, the CA method has been shown to generate better random numbers than the LFSR method does [23], and in the paper, it is adopted to produce desired pseudorandom numbers by using several alternating cells which change their states based on the two popular rules: rule 90 and rule 150 [23], given by

$$\text{Rule 90}: \ s_i^+ = s_{i-1} \oplus s_{i+1}$$

$$\text{Rule 150}: \ s_i^+ = s_{i-1} \oplus s_i \oplus s_{i+1}$$

where s_i denotes the current state of cell i; s_i^+ denotes the next state for s_i; $\oplus$ is the exclusive OR operator. Fig. 8 shows the example of CA-based PRNG.

For efficient hardware realization, the VHDL language is employed to implement this 16-bit CA module instead of the software-based RNG [23]. The main advantage of hardware-based cellular automata RNG hinges on its inherent speed. Once the seed value is given, the VHDL-based RNG efficiently generates a random sequence by the cell outputs on every system clock cycle (0.02 μs).

B. Selection Module

The aim of the selection module is to ensure survival of the fittest chromosomes (parents) to create new offsprings. In the proposed PEGA, the selected two chromosomes are sent to the crossover module and then the mutation module in the pipelined parallel architecture in Fig. 6. Roulette selection is the most popular selection scheme; however, it is required to sum up the fitness values for all the genes and to sort all the genes in the current population. Hence, the tournament selection method

is adopted in this module by taking the advantages of both chip size and computing speed to converge into the optimum as fast as possible. The procedure of the selection module in this proposed PEGA is similar to the conventional GA selection modules. In comparison to the conventional GAs, the hardware-based selection module is particularly designed to achieve the selection task. By taking the advantage of hardware implementation, the efficient comparators are employed to select the best chromosomes using VHDL combinational logic in the FPGA chip.

C. Crossover Module

The crossover module is a mechanism of genetic information exchange used in the proposed PEGA. Once the selection process has been completed in Section III-B, the two selected chromosomes are passed to this module. The crossover module generates two new offsprings; this is done by exchanging the strings of the two parent chromosomes. The high-speed one-point crossover scheme is chosen in this PEGA because two offsprings are obtained after the crossover process. The one-point crossover module has been implemented in combinational logic VHDL code to perform the high-speed crossover process more efficiently than other crossover schemes do. Note that the random single crossover point reported in this module comes from the RNG module in Section III-A. Once the crossover position is randomly determined, the genetic information of the two-parent chromosomes is directly exchanged to perform one-point crossover and the altered chromosomes are then sent to the mutation module.

D. Mutation Module

The mutation module has been implemented using combinational logic VHDL codes. After receiving two offsprings from the crossover process, the mutation module starts to proceed with the mutation process. The mutation bit position is randomly generated from the PRNG module in Section III-A. Moreover, both the addition and deletion operations were implemented by VHDL in this module.

E. Fitness Module

As mentioned in Section II-A4, the fitness function can be defined for its corresponding optimal problem. The purpose of this fitness module is to evaluate the population members after mutation and insert them into a new population. Because the fitness evaluation is usually problem-specific and user-defined by different applications, the fitness module has been efficiently implemented by software running in the embedded processor Altera Nios II. With the software-based fitness module, different fitness functions can be implemented with the pre-designed software program. Furthermore, the Nios II C-to-hardware acceleration compiler is employed to create custom hardware accelerators directly from ANSI C fitness module source code, thus significantly improving execution performance of the software-based fitness module.

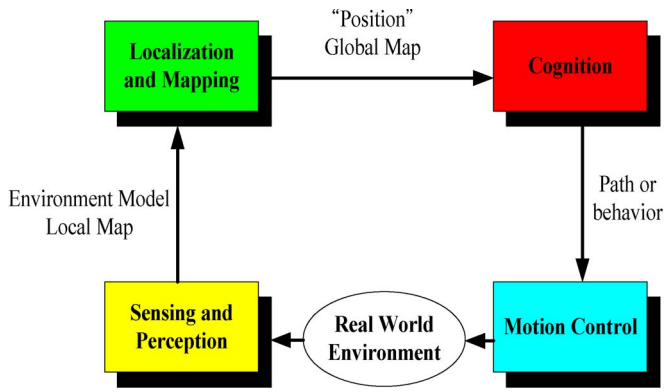

Fig. 9. General control structure of autonomous robot navigation.

F. Migration Module

In the proposed coarse-grain PEGA, there are two separate EGA engines working concurrently. Both EGA engines communicate with each other via the migration module; in particular, the migration module is designed to allow the two separate EGA engines to exchange their two best chromosomes after the isolation time. This module was implemented by VHDL to take the advantages of hardware realization. In this migration module, a VHDL-based timer is used to deal with data exchange from the two EGA engines. This hardware timer was implemented in FPGA by using a simple digital counter with a clock rate of 50 MHz. Once the isolation time is reached, the data exchange task is triggered and the best populations of the two separate EGA engines are directly exchanged using a simple combinational logic.

IV. Application to Global Path Planning for Autonomous Robot Navigation

As proposed in [24], the general control structure of autonomous robot navigation generally includes four main technologies: locomotion and motion control, sensing and perception, localization and mapping, and cognition (including path planning). Fig. 9 depicts the general control structure which can be applicable to most of mobile robots. Global path planning is one of the important tasks in the cognition module, aiming at constructing a collision-free path that satisfies certain optimization criterion between the starting position and the goal. Once the optimal collision-free path is constructed, the mobile robot will be accurately steered to follow the predetermined path toward the desired position using some kind of motion control laws and obstacle avoidance algorithms. In the following, the proposed SoPC-based PEGA and SoPC-based B-spline path smoother are used to solve the global path planning problem for autonomous robot navigation.

A. Fitness Function

In a structured environment with obstacles, the global path planning problem is aimed to find a feasible collision-free path for a mobile robot to move from a start location to a target location. Given a description of the environment, initial, and

final positions, the proposed PEGA is applied to compute a collision-free trajectory.

As shown in Fig. 1, a collision-free path (chromosome) contains a set of the grid points (genes of a chromosome), including the start point, via-points, and the endpoint. The length of a chromosome was variable, ranging from two to maximum length $M_{\max}$. A path can be either feasible (collision-free) or infeasible by evaluating the fitness function expressed by

$$F = \sum_{i=1}^{M_{\max}} (d_i + \alpha_i T) \tag{1}$$

where $M_{\max}$ is the number of line segments of a path, d_i is the distance of the near two nodes forming the line segment, T is a constant, α_i is given by

$$\alpha_i = \begin{cases} 0 & \text{if the ith line segment is feasible} \\ \sum_{k=1}^{N} k & \text{if the line segment intersects obstacle(s)} \end{cases} \tag{2}$$

where N is the number of obstacles that the line segment intersects.

B. SoPC-Based PEGA for Global Path Planning

The SoPC-based PEGA for solving the global path planning problem with the fitness function in (1) is described by the following steps.

- Step 1) Use the CA-based PRNG to randomly generate the population from the start point to the goal point.
- Step 2) Set the two parents from the VHDL-based tournament selection.
- Step 3) Execute the procedure of crossover and also check whether new chromosomes are acceptable. If the new chromosomes do not satisfy the requirement, repeat this procedure until acceptable chromosomes are obtained.
- Step 4) Perform the mutation process (addition and deletion are included) with low mutation rate, and ensure that new chromosomes are reasonable.
- Step 5) Repeat these four steps again until the convergence criterion are met or predetermined number of iterations is reach.

Once the optimal path has been determined based on the fitness function (1), these optimal sequences from the PEGA will be sent to the SoPC-based B-spline path smoother to construct a smooth collision-free path.

C. SoPC-Based Cubic B-Spline Path Smoother

In this subsection, the SoPC-based cubic B-spline modeling is constructed for path smoother of the mobile robots, thus improving the initial path from PEGA planner. The cubic B-spline has been the most commonly used in practice by taking the tradeoff between the computational efficiency and the smoothness of path curves. With this feature, the cubic B-spline path smoother is presented and then integrated into the same PEGA chip via the SoPC technology, as shown in Fig. 7.

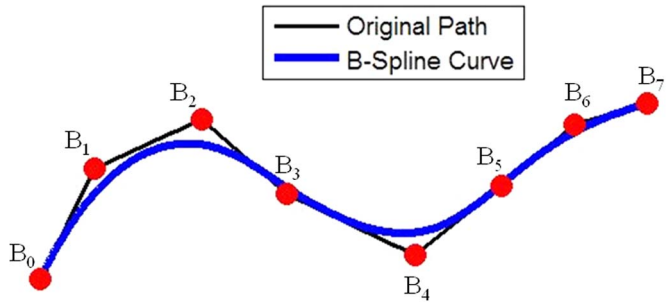

Fig. 10. B-spline curve with via points ($B_0 \sim B_7$).

1) B-Spline Modeling: This subsection briefly introduces the theories of B-spline modeling, which were proposed for approximation theory and have been applied for geometric modeling and etc. B-spline modeling approach has evolved from Bezier curves, aiming to provide local control of the curve shape without using a special set of blending functions. In addition, B-splines can add via points without increasing the degree of the curve. Fig. 10 depicts the path from planner and the B-spline curve (path smoother).

The B-spline curve defined by $n+1$ control points B_i is given by

$$P(t) = \sum_{i=1}^{n+1} B_i N_{i,k}(t), \quad 0 \le t \le t_{\max} \tag{3}$$

where $P(t)$ is $(k-2)$ times continuously differentiable; $N_{i,k}(t)$ are the normalized B-spline basis functions defined by the following Cox–deBoor recursion formulas

$$N_{i,k}(t) = \frac{(t - x_i)N_{i,k-1}(t)}{x_{i+k-1} - x_i} + \frac{(x_{i+k} - t)N_{i+1,k-1}(t)}{x_{i+k} - x_{i+1}} \tag{4}$$

$$N_{i,1}(t) = \begin{cases} 1, & x_i \le t \le x_{i+1} \\ 0, & \text{otherwise.} \end{cases} \tag{5}$$

The points x_i are called parametric knots or knot values. This knot vector must be non-decreasing, i.e., $x_i \le x_{i+1}$. For an open curve

$$x_j = \begin{cases} 0, & j < k \\ j - k + 1, & k \le j \le n \\ n - k + 2, & j > n \end{cases}$$

where $0 \le j \le n + k$ and $0 \le t \le n - k + 2$. Moreover, $N_{i,k}(t)$ is a piecewise polynomial of degree $k-1$ on each interval and $\sum_{i=0}^{n} N_{i,k}(t) \equiv 1$. Since $P(t)$ is $(k-2)$ times continuously differentiable, this means that for cubic B-splines, the curve must be twice continuously differentiable. Note that any indeterminate forms, such as integer division by zero or 0/0 are assumed as zero. The advantages of the B-spline modeling are summarized as follows:

1) Changing to a control point only affects the local curve of the B-spline curve.
2) Any number of points can be added without increasing the degree of the polynomial.
3) The B-spline $P(t)$ and its derivates of order $1, 2, \ldots, k-2$ are all continuously differentiable.

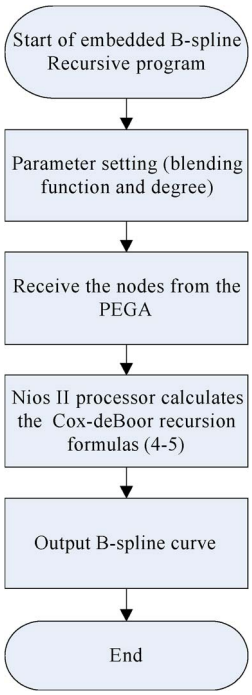

Fig. 11. Flowchart of the SoPC-based B-spline path smoother.

4) A B-spline curve is contained in the convex hull of its control polyline (strong convex hull property).

With these important properties, the B-spline modeling is particularly useful for finding smooth paths of autonomous mobile robots by means of grid-based maps. In the following, the SoPC technology will be employed to implement the B-spline path smoother by its efficient hardware/software co-design, thus significantly reducing realization cost in comparison with conventional software-based B-splines in personal computers or workstations.

2) SoPC-Based Cubic B-Spline Path Smoother: As shown in Fig. 7, the soft-core processor Nios II is embedded to evaluate the fitness functions of the PEGA path planner. After the optimal collision-free path is obtained, the cubic B-spline recursive program will be executed by this Nios II processor to achieve the path smoothing task. The proposed PEGA chip outputs this smoothed path information to the controller of mobile robots, in order to track the optimal path. Fig. 11 depicts the flowchart of the B-spline recursive program for the path smoother.

V. Simulation, Experimental Results, and Discussion

One simulation and two experiments are conducted in this section to illustrate the feasibility, performance, and merit of the proposed PEGA together with the B-spline path smoother. The simulation is performed to compare resultant fitness values between the proposed EGA and conventional GAs, while both experiments are respectively used for illustration of effectiveness of the proposed PEGA path planner together with the B-spline path smoother. In addition, execution performance of the software-based and SoPC-based algorithms is also investigated as well.

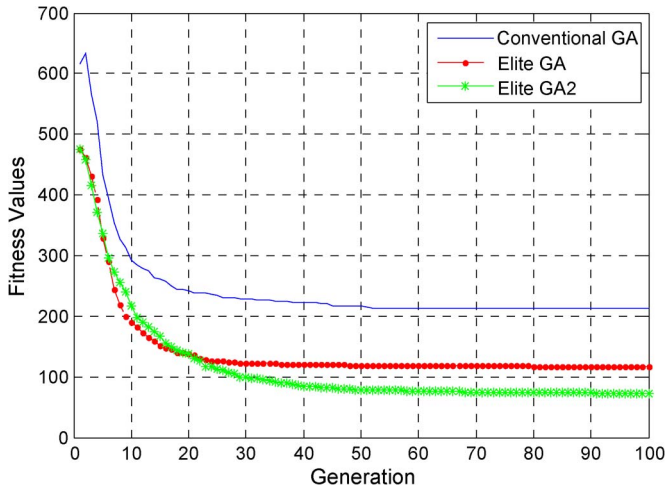

Fig. 12. Fitness values of two EGAs and conventional GA.

A. Simulation Results on the EGA Computing

This subsection aims to conduct simulation to examine the feasibility and efficacy of the proposed EGA in comparison with the conventional GA. The EGA and conventional GA are used to solve the global path planning problem in mobile robots with the fitness function in (1). The number of initial population is set by 50, the crossover probability is 0.8, the mutation probability is 0.1, and the maximal offspring generation is 100. Fig. 12 presents the fitness values of the EGA and conventional GA for solving the global path planning problem in (1). The combination of the proposed EGA with the policy of increasing diversity of population pool is called EGA2. The three algorithms were executed for 100 iterations with the same parameters, including the individual of initial generation, the size of the population, the probability of the crossover, and the mutation. As can been seen in the simulation results, the EGA has a better convergent speed than conventional GA does, but it falls into the local optimization easily. Moreover, the EGA2 overcomes this drawback and has a better fitness value than others do. Through simulation results, the proposed EGA2 has been shown capable of finding better solutions for the mobile robot in comparison with conventional GAs. Note that the processing time of the EGA2 is significantly improved by the parallel mechanism in SoPC-based PEGA.

B. Performance Evaluation of the SoPC-Based PEGA

This subsection compares the proposed SoPC-based PEGA and conventional GAs in terms of fitness values. The software fitness module implemented by the soft-core processor Nios II works with the PEGA IP library in FPGA to find the optimal solution. This pipelined and parallel strategies significantly diverse the searching space and shorten the execution time. To compare with the conventional methods to solve the path planning problem in mobile service robot, Table I presents the detailed analyses for resource usage and average execution time of five different implementations in which the chromosome length is 16 bits for each parameter. Theses algorithms were executed 20 times with distinct random seeds to report the analyses. Software GA and EGA Matlab codes were running

TABLE I
COMPARISON OF RESOURCE USAGE AND EXECUTION TIME OF THE SOFTWARE-BASED AND SOPC-BASED ALGORITHMS

	Software GA	SoPC GA	SoPC Parallel GA	Software EGA	SoPC PEGA
Resource usage (LEs)		5357	8357		9014
Execution time (sec)	45.92	1.5	0.73	69.22	0.76

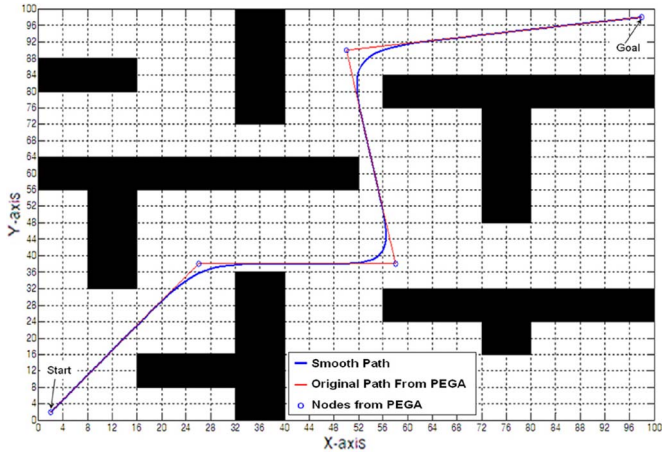

Fig. 13. Experimental results of the proposed PEGA path planning and smoothing in a complex environment.

on a PC with Pentium D 3.4-GHz CPU. The fitness modules in SoPC GA, SoPC parallel GA, and SoPC PEGA were realized in software running on the soft-core embedded processor (Nios II). The software GA, software EGA, and SoPC GA were terminated in 100th generation. The proposed SoPC-based coarse-grain PEGA and SoPC parallel GA terminated at the approximate fitness value for software EGA and software GA, respectively. With efficient FPGA implementation, the proposed coarse-grain PEGA found the optimal configuration and shortened the computation time which is 90 times of its corresponding software EGA computing. Compared with the SoPC parallel GA, the proposed SoPC-based coarse-grain PEGA obtained a better optimal solution (with lower fitness value) by using slightly more LEs and execution time.

C. Experimental Results of Global Path Planning and Smoothing

The subsection investigates the experimental results of the SoPC-based PEGA together with the B-spline path smoother for solving the global path planning problem of autonomous mobile robots. The effectiveness of the proposed PEGA is demonstrated by conducting two experiments with the parameters settings: population size is 50, crossover probability is 0.85, and mutation probability is 0.1. Figs. 13 and 14, respectively, present the experimental path planning results for a complex environment and a double U-shape environment. These results have shown that the proposed PEGA is capable of evolving optimal collision-free paths in different environments for mobile robots.

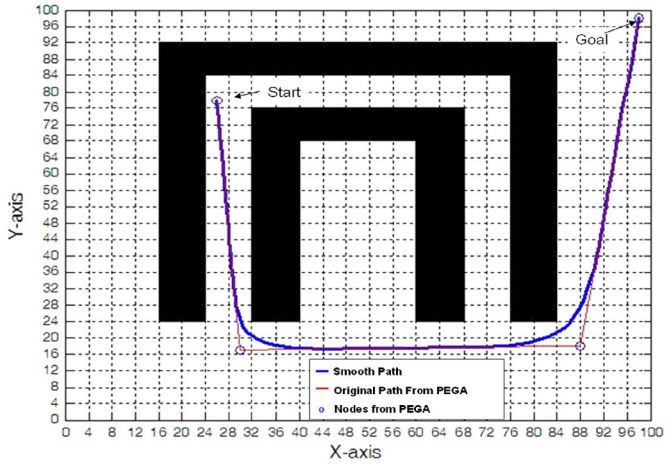

Fig. 14. Experimental results of the proposed path planning and smoothing in a double U-shape environment.

Furthermore, these discontinuous paths are smoothed by the SoPC-based B-spline path smoother as shown in Figs. 13 and 14, where the path information is obtained from the experimental PEGA chip. These experimental results indicate that the proposed SoPC-based PEGA is capable of successfully constructing smooth collision-free paths in the grid-based structured environment with obstacles.

Remark 1: Once the smooth collision-free path is obtained from the proposed PEGA and B-spline path smoother, the mobile robots can follow this path using the motion control laws in [25]–[30].

VI. CONCLUSION

This paper has presented a coarse-grain PEGA with its application to global path planning for autonomous robot navigation. The proposed coarse-grain PEGA has been efficiently implemented into an FPGA chip using the hardware/software co-design technique and the SoPC technique. In order to smoothen planned paths from the proposed PEGA planner, the path smoother has been proposed using the B-spline smoothing technique. The path smoother has also been realized into the same FPGA chip. Through simulations and experimental results, the proposed SoPC-based PEGA has been shown to outperform conventional software-based GAs or parallel GAs. Last but not least, this SoPC-based coarse-grain PEGA along with B-spline path smoother has successfully been applied to address the grid-based global path planning problem of autonomous mobile robots. An interesting topic for future research would be to integrate the proposed method and a local path planning method using techniques of bubble band and dynamic windows for developing a complete path planning system for autonomous robot navigation.

REFERENCES

[1] A. Hourtash and M. Tarokh, "Manipulator path planning by decomposition: Algorithm and analysis," *IEEE Trans. Robot. Autom.*, vol. 17, no. 6, pp. 842–856, Dec. 2001.
[2] C. Cai and S. Ferrari, "Information-driven sensor path planning by approximate cell decomposition," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 39, no. 3, pp. 672–689, Jun. 2009.

[3] E. Rimon and D. Koditschek, "Exact robot navigation using artificial potential functions," *IEEE Trans. Robot. Autom.*, vol. 8, no. 5, pp. 501–518, Oct. 1992.
[4] I. K. Jung, K. B. Hong, S. K. Hong, and S. C. Hong, "Path planning of mobile robot using neural network," in *Proc. IEEE Int. Symp. Ind. Electron.*, Bled, Slovenia, 1999, vol. 3, pp. 979–983.
[5] H. Surmann, J. Huser, and J. Wehking, "Path planning for fuzzy controlled autonomous mobile robot," in *Proc. 5th Int. Conf. Fuzzy Syst.*, New Orleans, LA, Sep. 1996, vol. 3, pp. 1660–1665.
[6] C. Hocaoglu and A. C. Sanderson, "Planning multiple paths with evolutionary speciation," *IEEE Trans. Evol. Comput.*, vol. 5, no. 3, pp. 169–191, Jun. 2001.
[7] I. A. Taharwa, A. Sheta, and M. A. Weshah, "A mobile robot path planning using genetic algorithm in static environment," *J. Comput. Sci.*, vol. 4, no. 4, pp. 341–344, 2008.
[8] J. C. Gallagher, S. Vigraham, and G. Kramer, "A family of compact genetic algorithms for intrinsic evolvable hardware," *IEEE Trans. Evol. Comput.*, vol. 8, no. 2, pp. 111–126, Apr. 2004.
[9] J. H. Holland, *Adaptation in Natural and Artificial Systems*. Ann Arbor, MI: Univ. Michigan Press, 1975.
[10] H. C. Lau, T. M. Chan, W. T. Tsui, and W. K. Pang, "Application of genetic algorithms to solve the multidepot vehicle routing problem," *IEEE Trans. Autom. Sci. Eng.*, vol. 7, no. 2, pp. 383–392, Apr. 2010.
[11] S. Yue, D. Henrich, W. L. Xu, and S. K. Tso, "Point-to-point trajectory planning of flexible redundant robot manipulators using genetic algorithms," *Robotica*, vol. 20, no. 3, pp. 269–280, May 2002.
[12] L. Lin, H. Wang, and Q. Wu, "Improved genetic algorithms based path planning of mobile robot under dynamic unknown environment," in *Proc. IEEE Int. Conf. Mechatron. Autom.*, Luoyang, China, Jun. 2006, pp. 1728–1732.
[13] M. Chen and A. M. Zalzala, "A genetic approach to motion planning of redundant mobile manipulator systems considering safety and configuration," *J. Robot. Syst.*, vol. 14, no. 7, pp. 529–544, 1997.
[14] T. W. Manikas, K. Ashenayi, and R. L. Wainwright, "Genetic algorithms for autonomous robot navigation," *IEEE Instrum. Meas. Mag.*, vol. 10, no. 6, pp. 26–31, Dec. 2007.
[15] P. R. Fernando, S. Katkoori, D. Keymeulen, R. Zebulum, and A. Stoica, "Customizable FPGA IP core implementation of a general-purpose genetic algorithm engine," *IEEE Trans. Evol. Comput.*, vol. 14, no. 1, pp. 133–149, Feb. 2010.
[16] Z. Konfrst, "Parallel genetic algorithms: Advances, computing trends, applications and perspectives," in *Proc. Int. Parallel Distrib. Process. Symp.*, Apr. 2004, pp. 26–30.
[17] Q. Huang, S. Hu, and R. Matin, "Fast degree elevation and knot insertion for B spline curve," *Comput. Aided Geom. Des.*, vol. 22, no. 2, pp. 183–197, Feb. 2005.
[18] Z. Wu, H. S. Seah, and M. Zhou, "Skeleton based parametric solid models: Ball b-spline surfaces," in *Proc. IEEE Int. Conf. Comput.-Aided Des. Comput. Graph.*, Oct. 2007, pp. 47–52.
[19] P. Thévenaz, T. Blu, and M. Unser, "Interpolation revisited medical images application," *IEEE Trans. Med. Imag.*, vol. 19, no. 7, pp. 739–758, Jul. 2000.
[20] Q. Wang and L. Wu, "Translation invariance and sampling theorem of wavelet," *IEEE Trans. Signal Process.*, vol. 48, no. 5, pp. 1471–1474, May 2000.
[21] C. G. Johnson and D. Marsh, "Modelling robot manipulators with multivariate B-splines," *Robotica*, vol. 17, no. 3, pp. 239–247, May 1999.
[22] F. C. Allaire, M. Tarbouchi, G. Labonte, and G. Fusina, "FPGA implementation of genetic algorithm for UAV real-time path planning," *J. Intell. Robot. Syst.*, vol. 54, no. 1–3, pp. 495–510, Mar. 2009.
[23] M. Serra, T. Slater, J. C. Muzio, and D. M. Miller, "The analysis of one-dimensional linear cellular automata and their aliasing properties," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 9, no. 7, pp. 767–778, Jul. 1990.
[24] R. Siegwart and I. R. Nourbakh, *Introduction to Autonomous Mobile Robots*. Cambridge, MA: MIT Press, 2004.
[25] C. C. Tsai, H. C. Huang, and T. Y. Wang, "Simultaneous tracking and stabilization of an omnidirectional mobile robot in polar coordinates," *J. Chinese Inst. Eng.*, vol. 32, no. 4, pp. 569–575, Jun. 2009.
[26] H. C. Huang and C. C. Tsai, "Adaptive robust control of an omnidirectional mobile platform for autonomous service robots in polar coordinates," *J. Intell. Robot. Syst.*, vol. 51, no. 4, pp. 439–460, Apr. 2008.
[27] H. C. Huang and C. C. Tsai, "Simultaneous tracking and stabilization of an omnidirectional mobile robot in polar coordinates: A unified control approach," *Robotica*, vol. 27, no. 3, pp. 447–458, May 2009.
[28] R. Marin, G. Leon, R. Wirz, J. Sales, J. M. Claver, P. J. Sanz, and J. Fernandez, "Remote programming of network robots within the UJI industrial robotics telelaboratory: FPGA vision and SNRP network protocol," *IEEE Trans. Ind. Electron.*, vol. 56, no. 12, pp. 4806–4816, Dec. 2009.
[29] C. F. Juang and C. H. Hsu, "Reinforcement ant optimized fuzzy controller for mobile-robot wall-following control," *IEEE Trans. Ind. Electron.*, vol. 56, no. 10, pp. 3931–3940, Oct. 2009.
[30] L. L. Menegaldo, G. A. N. Ferreira, M. F. Santos, and R. S. Guerato, "Development and navigation of a mobile robot for floating production storage and offloading ship hull inspect," *IEEE Trans. Ind. Electron.*, vol. 56, no. 9, pp. 3717–3722, Sep. 2009.

Ching-Chih Tsai (S'90–M'92–SM'00) received the Diplomat in electrical engineering from National Taipei Institute of Technology, Taipei, Taiwan, the M.S. degree in control engineering from National Chiao-Tung University, Hsinchu, Taiwan, and the Ph.D. degree in electrical engineering from Northwestern University, Evanston, IL, in 1981, 1986, and 1991, respectively.

He has published and coauthored over 280 technical papers, and he is currently a Distinguished Professor in the Department of Electrical Engineering, National Chung-Hsing University, Taichung, Taiwan. From 2003 to 2005, he served as the Director at Center for Research Development and Engineering Technology, College of Engineering, National Chung-Hsing University. In 2006, he served as the Chair, Taipei chapter, IEEE Robotics and Automation Society, and the Director at Center for Advanced Industry Technology and Precision, National Chung Hsing University. Since 2007, he has been the President, Taichung Chapter, the Chinese Institute of Engineers, Taiwan, and since 2009, he has been the Chair, Taichung Chapter, IEEE Systems, Man, and Cybernetics Society. His current research interests include mechatronics, advanced control methods, intelligent control, embedded control systems, and their applications to mobile robots and industrial control.

Dr. Tsai is an IET Fellow and a Fellow of Chinese Automatic Control Society.

Hsu-Chih Huang (S'08–M'09) received the M.S. degree in Institute of Biomedical Engineering from National Cheng-Kung University, Tainan, Taiwan, and the Ph.D. degree in electrical engineering from National Chung-Hsing University, Taichung, Taiwan, in 1999, and 2009, respectively.

He is currently an Assistant Professor in the Department of Computer Science and Information Engineering, HungKuang University, Taichung, Taiwan. His current research interests include intelligent control, mobile robots, embedded systems, SoPC, and nonlinear control.

Cheng-Kai Chan (S'08) received the B.S. degree in electrical engineering from National Taiwan University of Science and Technology, Taipei, Taiwan, in 1994, and the M.S. degree in electrical engineering from National Chung Hsing University, Taichung, Taiwan, in 2004, where he is currently working toward the Ph.D. degree in the Department of Electrical Engineering.

His current research interests include mechatronics, nonlinear control, adaptive control and their applications to industrial processes and machines.