Proceedings of the 2009 IEEE International Conference on Systems, Man, and Cybernetics
San Antonio, TX, USA - October 2009

SoPC-Based Parallel Elite Genetic Algorithm for Global Path Planning of an Autonomous Omnidirectional Mobile Robot

Hsu-Chih Huang, *Member, IEEE*
Department of Computer Science and Information Enginnering
HungKuang University
Taichung, Taiwan, R. O. C.
hchuang@sunrise.hk.edu.tw

Ching-Chih Tsai, *Senior Member, IEEE*
Department of Electrical Engineering
National Chung-Hsing University
Taichung, Taiwan, R. O. C.
cctsai@nchu.edu.tw

Shui-Chun Lin, *Member, IEEE*
Department of Electronic Engineering
National Chin-Yi University of Technology
Taichung, Taiwan, R. O. C.
lsc@ncut.edu.tw

***Abstract*—This paper presents an efficient parallel elite genetic algorithm (PEGA) for global path planning of an omnidirectional mobile robot moving in a static environment expressed by a grid-based map. This efficient PEGA, consisting of two parallel EGAs along with a migration operator, is proposed for global path planning of the mobile robots. The PEGA takes advantages of maintaining better population diversity, inhibiting premature convergence and keeping parallelism than conventional GAs do. The generated collision-free path is optimal in the sense of the shortest distance. The pipelined hardware implementation of IP (Intellectual Property) core library on a field-programmable gate array (FPGA) chip is employed to significantly speedup the processing time. Furthermore, a soft-core processor and a real-time operating system (RTOS) are embedded into the same chip to perform the global path planning using hardware/software co-design technique and SoPC (System-on-a-Programmable-Chip) concept. The merit and performance of the proposed SoPC-based PEGA are illustrated by conducting several simulations and experiments.**

***Keywords*—Elite genetic algorithm, embedded, omnidirectional, parallel, system-on-a-programmable-chip.**

I. INTRODUCTION

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

Among those existing methods to solve the global path planning optimization problems, GA algorithms have been recognized as heuristic and powerful robust optimization techniques for solving the global path planning optimization problems [7]. It takes the advantage of both deterministic and probabilistic approaches by using simple operators to improve solutions, such as crossover and mutation. Genetic algorithm was introduced by Holland [8] and has been proven to find the optimum path by taking the advantage of its strong optimization ability. Yue *et al.* [9] used GA to point-to-point trajectory planning of flexible redundant robot manipulator and Taharwa *et al.* [6] presented how to use GA to solve the path planning problem in static environment for a mobile robot. The improved GA for path planning of robot under unknown environment is introduced by Lin *et al.* [10]. Moreover, Chen and Zalzala [11] proposed a genetic approach to motion planning of redundant mobile manipulator systems. Manikas *et al.* [12] used gyroscopes incorporated with GA for autonomous robot navigation. However, the GAs suffer from their complicated computations so that they are not suitable for real-time applications. The FPGA implementation of GAs to speedup the processing time is presented in [13]. However, these conventional GAs [8-13] possible converge to a local optimum and get stuck in the solution for a long time, called premature convergence. The disadvantage can be avoided by using FPGA of the proposed PEGA. Moreover, the parallel computing method is presented not only to increase the diversity of searching space but also to speedup the processing time.

Recently, the new generation of FPGA technology enables an embedded processor IP and application IPs to be integrated into a SoPC developing environment. This new SoPC technology has been bringing a major revolution in the design of integrated circuits [14-16]. Since both software and hardware are integrated into a single programmable logic device, the designers can easily combine the flexibility of software unit and high performance of hardware unit for building a system on a chip [14-16]. Within an SoPC, the circuits requiring fast processing are suitable to be implemented by hardware in FPGA, and the highly sophisticated algorithms with heavy computations can be realized by software in FPGA [14-16]. The proposed FPGA-based PEGA using SoPC technology is more efficient than software-based GAs [8-12] or non-parallel FPGA-based GAs [13] do. As the authors' best understanding, SoPC-based PEGA has not been proposed to solve the global path planning problems of the omnidirectional mobile robot yet.

The objective of this paper is to develop a SoPC-based PEGA to solve the global path planning problem of the omnidirectional mobile robot. A pipelined hardware implementation of the PEGA is employed to speedup the

978-1-4244-2794-9/09/$25.00 ©2009 IEEE

processing time. The rest of this paper is organized as follows. In Section II, the coarse-grain PEGA is proposed to significantly accelerate the computing and diverse the searching space. Section III elaborates the FPGA implementation of the proposed parallel elite genetic algorithm using the SoPC technology. Section IV elucidates the procedure of how to apply the PEGA to solve the global path planning problem for the omnidirectional mobile robot. Section V conducts several simulations and experimental results to show the effectiveness and merit of the proposed methods. Section VI concludes this paper.

II. PARALLEL GENETIC ALGORITHM

This section aims to develop an efficient coarse-grain parallel elite genetic algorithm, including selection, crossover, mutation, elite policy and diversity increasing strategy. The elite policy and diversity increasing strategy are employed to avoid the premature convergence. With the benefits of EGA, the PEGA is proposed by using the coarse-grain parallel architecture, which takes the advantages of FPGA-based implementation.

2.1 Elite GA Computing for Global Path Planning

In GA computing for global path planning, a chromosome contains the set of the grid points of the trajectory including start point, via points and end point shown in fig.1. The optimal collision-free path is evolved by the GA operators, such as selection, crossover and mutation [6]. However, the conventional GAs exist a few drawbacks as following. The first one is that the best solution may have the chance to become worse in the next generation; thus the convergence speed will be slowed down. The second one is premature convergence.

To circumvent these shortcomings, an elite genetic algorithm is proposed in this Section. It contains reproduction strategy with elite policy, and diversity increasing in population pool. Those make EGA a powerful and effective search algorithm. In what follows the core operators of EGAs are introduced.

2.1.1 Selection

The main task of selection module is to select individuals from the populations so that these individuals can be sent to the crossover and mutation module in order to attain new offsprings. Selection is one of the key operators that ensure survival of the fitness.

2.1.2 Crossover

Crossover is the fundamental mechanism of genetic rearrangement in EGAs. This is done by the exchange of strings between two parent chromosomes. Crossover occurs when two chromosomes break and then reconnect but to different end piece.

2.1.3 Mutation

Mutation is a process which 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 preventing quick convergence to a local minimum. Moreover, there are two special operations are employed in the proposed EGA,

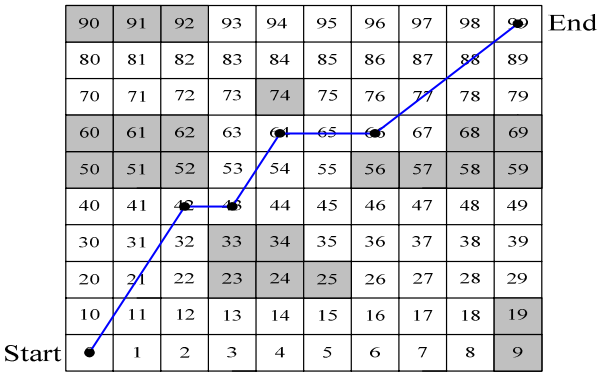

Fig.1. A grid-based environment and the collision-free path.

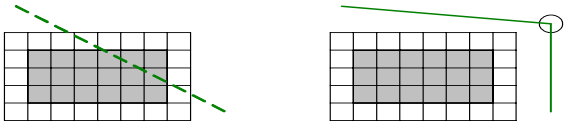

Fig.2. (a)Path before addition. (b)After addition operation.

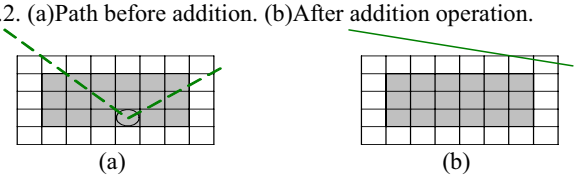

Fig.3. (a)Path before deletion. (b)After deletion operation.

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 may make the length of the path longer, but it may also make the path to be feasible by avoiding going through obstacles. Deletion is applied by randomly to choose one node, and delete it. As shown in Fig.3, the deletion process can make the path have the capability to reduce the fitness value by shorting the nodes and avoiding passing through obstacles.

2.1.4 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, such as crossover and mutation, should be evaluated based on the fitness function. For complex problems, the computation time becomes dominant in the overall performance.

2.1.5 Reproduction Strategy with Elite Policy

Crossover is an important operator that helps GAs to find the optimal solution. The chromosome crossovers with another one based on the crossover probability. However, the best solution has the chance to be destroyed. The offspring may have worse fitness than parents and it will slow down the convergent speed. To circumvent the problem, one takes the elite preservation policy into account. As shown in Fig.4, 20% of the best-performing individuals are reproduction directly to the next generation. It can be ensured that the offspring will perform as least as good as their parents. The rest of the new generation are randomly selection from the population pool and produced by crossover and mutation operations.

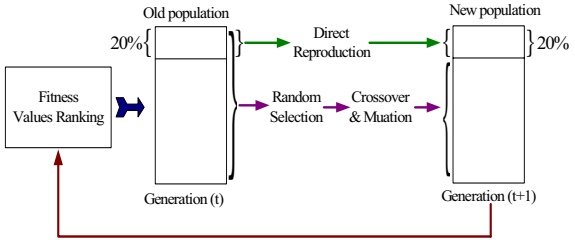

Fig.4. Reproduction strategy with elite policy.

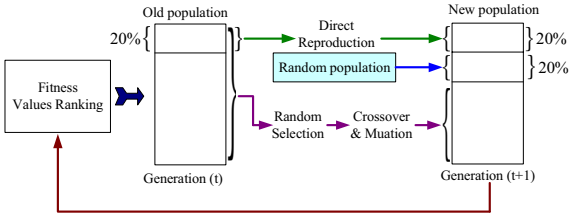

Fig.5. Diversity increasing in population pool.

2.1.6 Diversity Increasing in Population Pool

One of the main problems of the GA process is premature convergence before the global optimal solution has been found. The reason for premature convergence is that the individuals in the population pool are too alike. To ensure that members in different generation are not identical in a long run, one can 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 rest of the population do not converge easily. Combining diversity increasing and random selection in EGAs, the chromosomes have the ability to search the global solution than the conventional GAs do.

2.2 Coarse-Grain Parallel Elite Genetic Algorithm

Although EGAs have been shown to find a better optimal solution than GAs do, they require many computations and iterations that cause enormous time consumption. The parallel EGA has a more powerful capability to solve optimal problems much faster than conventional or serial EGAs.

Parallel architectures have been widely used to accelerate various computational algorithms by using distributed processing [17]. The coarse-grain parallel model is particularly useful and pragmatic for chip-based implementation of parallel EGA, thereby speeding up its computing. Moreover, this kind of parallel model with SoPC implementation has potential capability to solve optimal problems much faster than software implementations do. This subsection intends to design a useful coarse-grain PEGA based on the coarse-grain model [17]. Fig.6 depicts the architecture of the proposed coarse-grain PEGA. Such a PEGA is composed of two EGA algorithms which are concurrently executed on two processors with data interchange. Compared with 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 possibility to seek for a global optimum, and accelerate its computation.

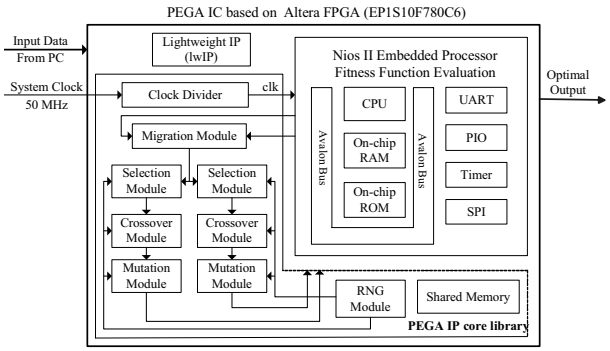

Fig.6. FPGA-based coarse-grain PEGA using two nodes.

The parallel EGA has been implemented on two nodes, each of which has its 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 members. 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 subpopulations (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 in a local optimum.

III. SOPC IMPLEMENTATION

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 [18]. This subsection is dedicated to design the PEGA in FPGA for solving optimal problems. Compared with [18], not only hardware circuits of genetic operators are implemented in FPGA but also an embedded processor and a real-time OS are embedded in the same FPGA. Fig.6 depicts the architecture of the FPGA implementation for the proposed coarse-grain PEGA using two nodes. Worthy of mention is that the fitness module for the PEGA has been implemented into the 32-bit Nios II processor to realize the 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) module, selection module, crossover module, mutation module and migration module. The software-based fitness module and hardware-based custom logics for PEGA are connected to the system interconnect fabric via Avalon-MM in one FPGA chip. The used FPGA chip is Altera Stratix EP1S10F780C6 with 10,570 LEs (Logic Element), 426 user I/O pins, 6 DSP blocks, 920,448 RAM bits memory, 6 PLLs (Phase-Lock Loop) and an embedded Nios II 32-bit RISC (Reduced Instruction Set Computer) processor.

In what follows, special efforts will be paid to design several modules in the proposed FPGA-based PEGA shown in Fig.6, such as RNG module, selection module, crossover module, mutation module, fitness module and migration module. 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 so that they can be further synthesized to application-specific integrated circuits (ASICs). Moreover, with the SoPC technology, a real-time operating system (RTOS) can be ported into the same FPGA chip for embedded applications, such as many kinds of network service.

3.1 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 module, mutation module and selection module. 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 [18]. Thus, the CA method is adopted to produce desired pseudorandom numbers by using several alternating cells [18].

For efficient hardware realization, the VHDL language is adopted to implement this CA module instead of the software-based RNG [19]. 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 output.

3.2 Selection, Crossover and Mutation

These three genetic operators are implemented by combinational logic using VHDL. The tournament selection and one-point crossover are adopted by taking the advantages of both chip size and computing speed. Worthy of mention is that low mutation rate is preferred so that the two offsprings will resemble to their parents.

3.3 Fitness Module

Since 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. 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 (C2H) 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.

3.4 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.

IV. GLOBAL PATH PLANNING FOR OMNIDIRECTIONAL MOBILE ROBOT

Global path planning is one of the important tasks in intelligent control for autonomous mobile robots. The global path planning problem for mobile robots is essentially an optimal problem for constructing a collision-free path route that satisfies certain optimization parameters between the starting position and the goal. Once the optimal collision-free path is constructed, the mobile robot accurately follows the pre-determined path toward the desired position using some kinds of control law. In what follows, special efforts will be paid to solve the global path planning problem for omnidirectional mobile robot using the proposed PEGA.

4.1. Global Path Planning for Mobile Robot Using PEGA

In an environment with obstacles, global path planning is to find a suitable 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 o compute a collision-free trajectory.

As shown in Fig.6, a collision-free path (chromosome) contains a set of the grid points (genes of a chromosome), including the start point, via points and end point. The length of a chromosome was variable and between 2 and maximum length $M_{\max}$. A path can be either feasible (collision-free) or infeasible by evaluating the fitness function expressed as below.

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

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.

4.2 FPGA-Based PEGA for Global Path Planning

This section is aimed to apply the FPGA-based PEGA for the global path planning problem of the mobile robot. The PEGA for solving the global path planning problem with the fitness function in (1) is described by the following steps.

Step1. Use the CA-based PRNG to randomly generate the population from the start point to the goal point.

Step2. Set the two parents from the VHDL-based tournament selection.

Step3. Execute the procedure of crossover and also check whether new chromosomes are acceptable. If the new chromosomes do not satisfy the requirement, this procedure must be repeated until acceptable chromosomes are obtained.

Step4. Perform the mutation process (addition and deletion are included) with low mutation rate, and ensure that new chromosomes are reasonable.

Step5. Repeat these four steps again until the convergence

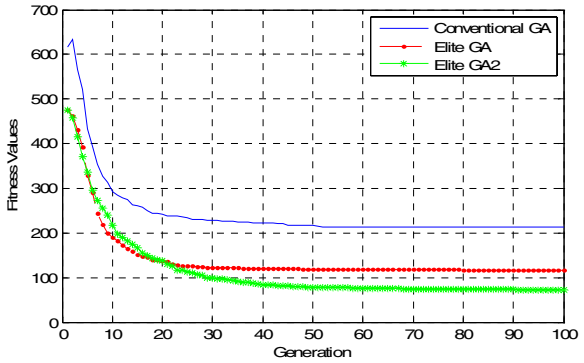

Fig.7. The fitness value of EGA and conventional GA.

criterions is met or pre-determined number of iteration is reach.

Once the optimal path is determined based on the fitness function (1), these optimal sequences, from the PEGA chip can be sent to the SoPC-based platform controller via TCP/IP protocol for tracking the collision-free path [16]. The resource usage of the proposed PEGA is 9,014 LEs (85% of total LEs), 690,152 memory bits (75% of total RAM bits).

V. SIMULATION, EXPERIMENTAL RESULTS AND DISCUSSION

5.1 Simulation Result of Fitness Value in Elite GA Computing

This subsection aims to conduct simulation to examine the feasibility and efficacy of the proposed elite genetic algorithm compared with the conventional GA. The elite GA and GA are used to solve the global path planning problem in omnidirectional mobile robot with the fitness function in (1). The number of initial population is supposed to 50, the crossover probability is 0.8, the mutation probability is 0.1, and the maximal offspring generation is 100. Fig.7 presents the fitness values of the proposed elite GA 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 are executed are 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. From the simulation results, the *EGA* has a better convergence speed than conventional GA does, but it seeks the local optimization easily. The *EGA2* overcomes this drawback and has a better fitness value than others do. Through simulation results, the proposed EGAs have been shown capable of finding better solutions for the mobile robot in comparison with conventional GAs. Note that the processing time of EGA is significantly improved by parallel mechanism in FPGA-based PEGA.

5.2. Experimental Results of Global Path Planning

This section is devoted to investigate the experimental results of the SoPC-based PEGA for the path planning problem of the mobile service robot. The effectiveness of the proposed PEGA is demonstrated by conducting two

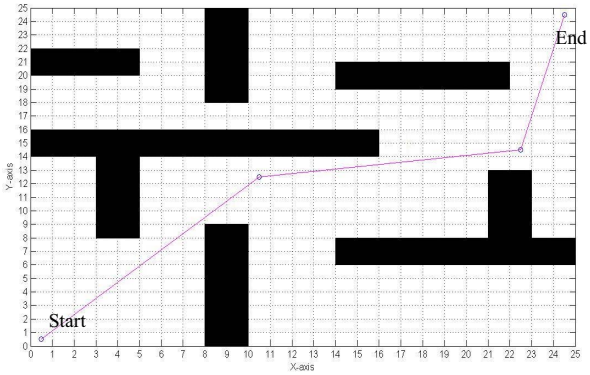

Fig.8. Experimental result of the FPGA-based PEGA path planning in a complex environment.

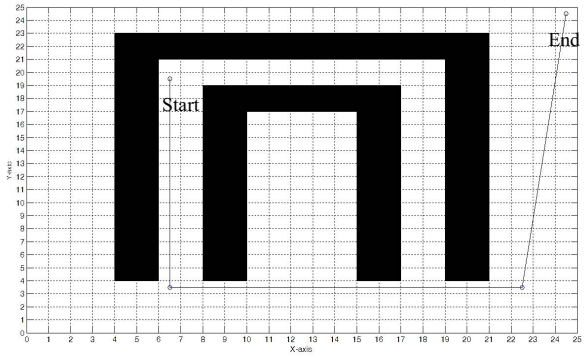

Fig.9. Experimental result of the FPGA-based PEGA path planning in a double U-shape environment.

Table I. Comparison of resource usage and execution time of the software-based and SoPC-based algorithms.

	Software GA	SoPC GA [7]	SoPC Parallel GA [7]	Software EGA	SoPC PEGA
Resource usage (LEs)		5357	8357		9014
Execution time (sec)	45.92	1.5	0.73	69.22	0.76

experiments with the parameters settings, population size is 50, crossover probability is 0.85, and mutation probability is 0.1. Fig.8, 9 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 the mobile robot.

5.3 Performance Evaluation of the FPGA-based PEGA

The aim of the second experiment is to examine the merit and performance of the proposed coarse-grain PEGA. 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 redundant problem in mobile service robot, Table I

presents the detailed analyses for resource usage (LEs) 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 on a PC with Pentium D 3.4GHz CPU. The fitness module in SoPC GA and coarse-grain PEGA were realized in software running on the soft-core embedded processor (Nios II) which evaluates the fitness of each chromosome and passes the randomly generated chromosomes with two other chromosomes (parents) to the selection module. The selection unit compared these chromosomes and obtained two better chromosomes passed to the next generation. The software GA, software EGA and SoPC GA were terminated in 100th generations. The proposed coarse-grain PEGA and SoPC parallel GA [17] 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 [17], the proposed FPGA-based coarse-grain PEGA obtained a better optimal solution (with lower fitness value) by using slightly more LEs and execution time.

VI. CONCLUSION

This paper has presented a coarse-grain parallel elite genetic algorithm for path planning of an omnidirectional mobile robot. The proposed coarse-grain PEGA has been efficiently implemented into a FPGA chip using the hardware/software co-design technique and SoPC technique to solve the path planning problem. The coarse-grain PEGA has been rapidly developed in the FPGA chip by incorporating with the embedded processor and the RTOS in the same chip. Through simulations and experimental results, the proposed FPGA-based PEGA outperforms conventional software-based GAs or parallel GAs. Last but not least, this FPGA-based coarse-grain PEGA has been applied to the service robot to perform global path planning task.

ACKNOWLEDGMENT

The authors gratefully acknowledge financial support from the National Science Council, Taiwan, R.O.C., under the grant NSC 95-2213-E-005-002.

REFERENCES

[1] A. Hourtash and M. Tarokh, "Manipulator path planning by decomposition: algorithm and analysis," *IEEE Transactions on Robotics and Automation*, vol.17, no.6, pp.842-856, 2001.

[2] E. Rimon and D. Koditschek, "Exact robot navigation using artificial potential functions," *IEEE Transactions on Robotics and Automation*, vol.8, no.5, pp.501-518, October 1992.

[3] I. K. Jung, K. B. Hong, S. K. Hong and S. C. Hong, "Path Planning of mobile robot using neural network," *Proceedings of the IEEE International Symposium on Industrial Electronics,* vol.3, pp.979-983, Slovenia 1999.

[4] H. Surmann, J. Huser and J. Wehking, "Path planning for fuzzy controlled autonomous mobile robot," *5th International Conference on Fuzzy Systems*, vol.3, pp.1660-1665, New Orleans, September, 1996.

[5] C. Hocaoglu, and A. C. Sanderson, "Planning multiple paths with evolutionary speciation," *IEEE Transactions on Evolutionary Computation,* vol.5, no.3, pp.169-191, 2001.

[6] I. AL. Taharwa and A. Sheta and M. AI. Weshah, "A mobile robot path planning using genetic algorithm in static environment," *Journal of Computer Science*, vol.4, no.4, pp. 341-344, 2008.

[7] J. C. Gallagher, S. Vigraham and G, Kramer, "A family of compact genetic algorithms for intrinsic evolvable hardware," *IEEE Transactions on Evolutionary Computing*, vol.8, no.2, pp.111-126, 2004.

[8] J. H. Holland, "Adaptation in natural and artificial systems," *University of Michigan Press*, Ann Arbor, 1975.

[9] 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.

[10] L. Lin, H. Wang and Q. Wu, "Improved genetic algorithms based path planning of mobile robot under dynamic unknown environment," *Proceeding of the IEEE International Conference on Mechatronics and Automation*, pp.1728-1732, Luoyang, China, June 2006.

[11] M. Chen and A. M. S. Zalzala, "A genetic approach to motion planning of redundant mobile manipulator systems considering safety and configuration," *Journal of Robotic Systems*, vol.14, no.7, pp.529-544, 1997.

[12] T. W. Manikas, K. Ashenayi and R. L. Wainwright, "Genetic algorithms for autonomous robot navigation," *IEEE Instrumentation & Measurement Magazine*, vol.10, no.6, pp. 26-31, December 2007.

[13] B. Shackleford, G. Snider, R. J. Carter, E. Okushi, M. Yasuda, K. Seo and H. Yasuura, "A high-performance, pipelined, FPGA-based genetic algorithm machine," *Genetic Programming and Evolvable Machines*, vol.2, pp.33-60, 2001.

[14] Y. F. Chan, M. Moallem and W. Wang, "Design and implementation of modular FPGA-based PID controllers," *IEEE Transactions on Industrial Electronics*, vol.54, no.4, pp. 1898-1906, August 2007.

[15] D. Zhang and H. Li, "A stochastic-based FPGA controller for an induction motor drive with integrated neural network algorithms," *IEEE Transactions on Industrial Electronics*, vol. 55, no.2, pp.551-561, February 2008.

[16] H. C. Huang and C. C. Tsai, "FPGA implementation of an embedded robust adaptive controller for autonomous omnidirectional mobile platform," *IEEE Transactions on Industrial Electronics*, vol.56, no.5, pp.1604-1616, May 2009.

[17] M. S. Jelodar, M. Kamal, S. M. Fakhraie, M. N. Ahmadabadi, "SOPC-based parallel genetic algorithm," *IEEE Congress on Evolutionary Computation*, pp.2800-2806, 2006.

[18] M. Serra, T. Slater, J. C. Muzio and D. M. Miller, "The analysis of one-dimensional linear cellular automata and their aliasing properties, " *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol.9, no.7 pp.767-778, 1990.