IV International Conference on Computational Methods for Coupled Problems in Science and Engineering
COUPLED PROBLEMS 2011
M. Papadrakakis, E. Oñate and B. Schrefler (Eds)

A HARDWARE-BASED MULTI-DISCIPLINARY DESIGN OPTIMISATION METHOD FOR AERONAUTICAL APPLICATIONS

J. KOK*, F. GONZALEZ*, N. KELSON† AND T. GURNETT†

*Australian Research Centre for Aerospace Automation (ARCAA)
Queensland University of Technology (QUT)
Eagle Farm, 4009 Queensland, Australia
e-mail: info@arcaa.aero, www.arcaa.aero/

†High Performance Computing (HPC) and Research Support Group
Queensland University of Technology (QUT)
Brisbane, 4000 Queensland, Australia
e-mail: qut.itshpc@qut.edu.au, www.itservices.qut.edu.au/hpc/

Key words: Aeronautics, Field Programmable Gate Array, Multi-disciplinary Design Optimisation, Unmanned Aerial Vehicle

Abstract. There are many applications in aeronautics where there exist strong couplings between disciplines. One practical example is within the context of Unmanned Aerial Vehicle (UAV) automation where there exists strong coupling between operation constraints, aerodynamics, vehicle dynamics, mission and path planning. UAV path planning can be done either online or offline. The current state of path planning optimisation online UAVs with high performance computation is not at the same level as its ground-based offline optimizer's counterpart, this is mainly due to the volume, power and weight limitations on the UAV; some small UAVs do not have the computational power needed for some optimisation and path planning task. In this paper, we describe an optimisation method which can be applied to Multi-disciplinary Design Optimisation problems and UAV path planning problems. Hardware-based design optimisation techniques are used. The power and physical limitations of UAV, which may not be a problem in PC-based solutions, can be approached by utilizing a Field Programmable Gate Array (FPGA) as an algorithm accelerator. The inevitable latency produced by the iterative process of an Evolutionary Algorithm (EA) is concealed by exploiting the parallelism component within the dataflow paradigm of the EA on an FPGA architecture. Results compare software PC-based solutions and the hardware-based solutions for benchmark mathematical problems as well as a simple real world engineering problem. Results also indicate the practicality of the method which can be used for more complex single and multi-objective coupled problems in aeronautical applications.

1 INTRODUCTION

Multi-disciplinary design optimisation (MDO) has been actively applied across engineering disciplines over a wide range of problems, the majority of which lies within the field of aeronautics [1]. The complexity of modern systems has directed research focus towards improving MDO search algorithms and analysis tools [2–4]. One method to speed up the runtime of an MDO search algorithm is to implement its features in hardware, where concurrent data processing is possible. MDO search algorithms that take several hours to run could be executed in fractions of a second, impacting significantly on the design time of a project.

One such hardware which has advanced extensively in technology is the Field Programmable Gate Array (FPGA). An FPGA implementation in the field of aeronautics will also contribute to the Unmanned Aerial Vehicle (UAV) automation community where it is necessary to have a high level of integration within the capabilities of a vehicle. One practical example is within the context of Unmanned Aerial Vehicle (UAV) automation where there exists strong coupling between operation constraints, aerodynamics, vehicle dynamics, mission and path planning. UAV path planning can be done either online or offline. The current state of path planning optimisation online UAVs with high performance computation is not at the same level as its ground-based offline optimizer's counterpart, this is mainly due to the volume, power and weight limitations on the UAV; some small UAVs do not have the computational power needed for some optimisation and path planning task. In this paper, we describe an optimisation method which can be applied to Multi-disciplinary Design Optimisation problems and UAV path planning problems. Hardware-based design optimisation techniques are used. The power and physical limitations of UAV, which may not be a problem in PC-based solutions, can be approached by utilizing a Field Programmable Gate Array (FPGA) as an algorithm accelerator. The inevitable latency produced by the iterative process of an Evolutionary Algorithm (EA) is concealed by exploiting the parallelism component within the dataflow paradigm of the EA on an FPGA architecture.

In this paper, an FPGA implementation of the search algorithm of an MDO is proposed. Our proposed design inherits the diversity and Pareto optimality features necessary for an effective MDO search algorithm. Simulation results showing speed up of 43 times for discrete and discontinuous test problems, and 244 times for the constrained test problem.

2 BACKGROUND

Previous work incorporating search algorithm design onto FPGA [5–10] have ascertained this concept. One of the limitations was the application to single objective search algorithms only, which is not realistically applicable to real-world scenarios where conflicting objectives often arise. An important point to note is that the paradigm behind an MDO search algorithm design is very different from that of a single objective optimisation [11]; the MDO search algorithm is aimed to find multiple trade-off optimal solutions with a wide range of

values across the objectives, whereas a single objective optimisation search algorithm is directed towards one optimal solution. Bonissone and Subbu [12] presented an Evolutionary Multi-objective Optimisation on an FPGA but were restricted to the basic filtering of Pareto fronts and not addressing the core features such as solution diversity and Pareto optimality.

3 METHODOLOGY

The methodology applied to this research is as depicted in Figure 1 below. Coinciding with the methodology, the rest of the paper is organized as follows. Section 4 summarises the features of EAs which contribute to a robust MDO search algorithm. Section 5 presents the proposed FPGA-based MDO implementation. Section 6 reports on the test problems conducted to verify the functionality of the FPGA-based MDO and demonstrate its effectiveness. Section 7 concludes with a brief summary and highlighting future work.

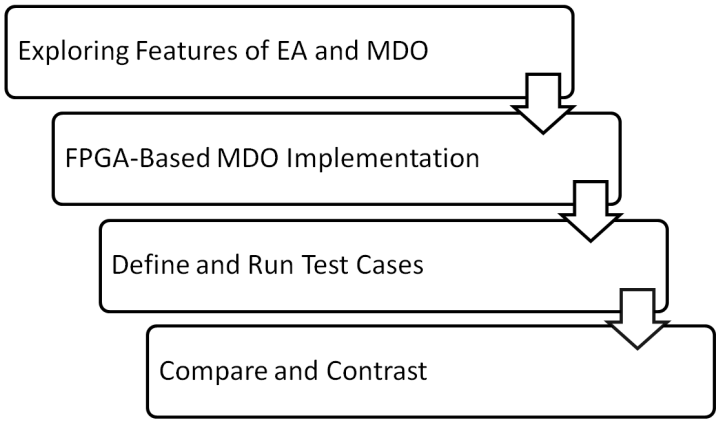

Figure 1: Methodology for developing an FPGA-based MDO

4 MULTI-DISCIPLINARY DESIGN OPTIMISATION

The two main objectives in multi-objective optimisation and MDO is to find a set of solutions which are as diverse as possible and as close as possible to the set of optimal solutions within the feasible search space, known as the Pareto-optimal front [11]. There are a number of search methods used in MDO, such as NSGA-II [13], DPGA [14], SPEA [15], TDGA [16] and PAES [17]. One class of the techniques actively applied on MDO is Evolutionary Algorithms (EAs) [18, 19], which is a Metaheuristic optimisation algorithm inspired by the theory of natural evolution. The attractiveness of EAs lies in its population-based characteristic, where manipulating multiple solutions leads to the simultaneous discovery of a Pareto set [4]. There are a number of EAs and MOEAs with different features. In this paper we use and explore the features of a well known multi-objective EA, NSGA-II [13]. Even though NSGA-II has some drawback [13], it also has very good

features such as diversity maintenance technique using a crowding distance assignment and Pareto ranking for preserving elitists.

5 PROPOSED FPGA-BASED MDO

5.1 Algorithm Overview

The architecture of the FPGA-based MDO is depicted below in Figure 2. The algorithm incorporates the key features, fixed-point representation, random number generator, crossover, mutation, Pareto front ranking, crowding distance assignment, selection and evaluation, which are needed for securing diverse Pareto-optimal fronts. The population, consisting of candidate solutions, is stored on the block RAM onboard the FPGA. Tournament selection is randomly carried out across the population, determining better candidate solutions to be genetically altered, which produces the offspring for that current generation. After the offspring have been crossovered, mutated and evaluated, they are concatenated with the parent population to undergo Pareto front ranking and crowding distance assignment. The higher ranking and wider spread solutions are updated back into the population block RAM. It is stressed that the data flow arrow in Figure 2 denotes parallel processing. The following subsections will describe the functionality of the basic features and modules of the proposed FPGA-based MDO.

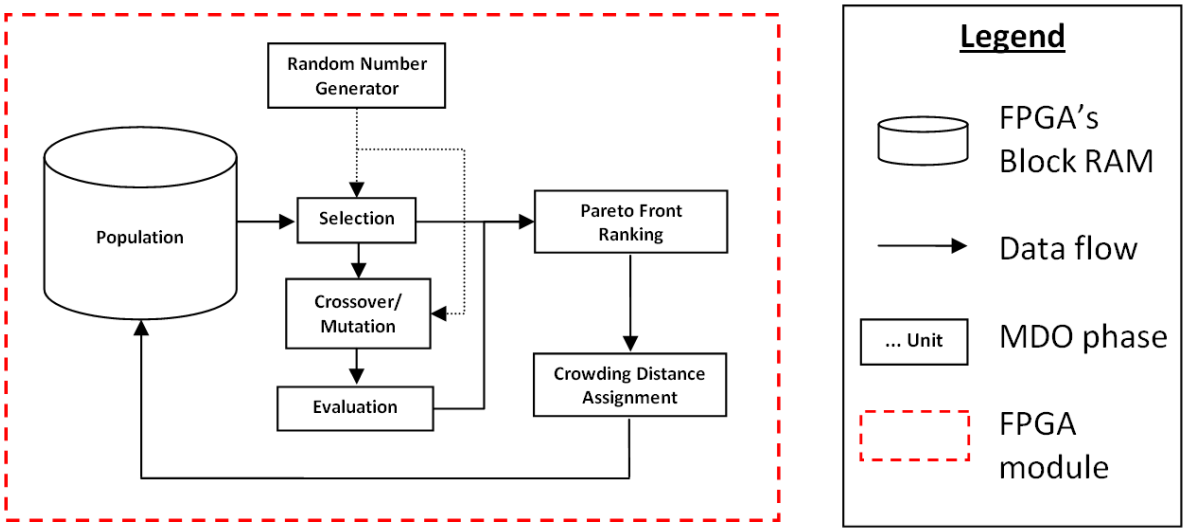

Figure 2: Architecture of the FPGA-based MDO search algorithm.

5.2 Representation

One of the first design decisions is determining the representation of candidate solutions. These could be coded either in real-coded binary, floating-point or fixed-point numbers. Michalewicz [19] experimented and concluded that floating-point and fixed-point represen-

tation is faster, more consistent and higher in precision than real-coded representation. Since the complexity of floating-point arithmetic consumes a larger logic footprint and is not as efficient as fixed-point arithmetic, therefore the candidate parameters are encoded in fixed-point representation.

5.3 Random Number Generator

When deciding on a logic level Random Number Generator (RNG) for an FPGA implementation, two main factors are taken into consideration. They are the randomness and its period. Matsumoto and Nishimura [20] proposed a pseudo RNG called Mersenne Twister, which was argued to be as fast and random as the standard ANSI-C "rand()". The Mersenne Twister is essentially a uniformly distributed pseudo RNG based on a matrix linear recurrence over a large finite binary field. Another advantage of FPGA implementation of a Mersenne Twister is its low resource consumption and its capability to generate new random sequences every clock cycle.

5.4 Crossover/Mutation

A simulated binary crossover and a polynomial mutation proposed by Deb and Agarwal [21] are implemented. The intention of a crossover operation is to exchange useful information between two candidate solutions, whereas a mutation operation is aimed to slightly alter a candidate solution. Thus, crossover and mutation can be seen as exploitation and exploration respectively. A balance between them is applied to guide a search algorithm.

5.5 Pareto Front Ranking

Pareto front ranking is based on the non-dominance feature of a candidate solution [11]. Solution A is said to dominate solution B if all of solution A's fitness is better than solution B's, else solution B is non-dominated. Non-dominated solutions are allocated higher rank than dominated ones, hence ensuring the preservation of Pareto fronts.

5.6 Crowding Distance Assignment

A technique proposed by Deb(2001) [11] known as crowding distance assignment is implemented to maintain the diversity of the Pareto fronts. The advantage of this technique lies in the nature by which it operates, whereby it does not require any performance dependent parameter.

5.7 Selection

Tournament selection based on the Pareto front rank and crowding distance is used as a competition winning criteria for the next generation of offspring. Higher ranking solutions wins over lower ranking solutions. If two solutions are of the same rank, the solution with higher crowding distance wins. This method ensures the survival of the fittest.

5.8 Evaluation

The evaluation module is the only module that is application dependent. It would contain the necessary objective functions to be optimized.

6 EXPERIMENTS AND RESULTS

6.1 Test problems

Three different types of test problems were used as a test bench. These three test problems were specifically chosen to analyze the functionality of the algorithm when applied on discrete, discontinuous and constrained problems. The first test problem is SCH1 [22]:

$$\begin{aligned} &\text{Minimize} \quad f_1(x) = x^2, \\ &\text{Minimize} \quad f_2(x) = (x-2)^2. \end{aligned} \tag{1}$$

This problem has a convex set of Pareto-optimal solution. It is aimed to highlight the diversity of solutions and whether or not they lie on the convex Pareto-optimal front.

The second test problem is SCH2 [22] :

$$\text{Minimize} \quad f_1(x) = \begin{cases} -x & \text{if} \quad x < 1, \\ x-2 & \text{if} \quad 1 < x \leq 3 \\ 4-x & \text{if} \quad 3 < x \leq 4 \\ x-4 & \text{if} \quad x > 4 \end{cases}$$

$$\text{Minimize} \quad f_2(x) = (x-5)^2. \tag{2}$$

This problem has a Pareto-optimal front consisting of two discontinuous regions. The aim for this test problem is to verify if the algorithm will capture the true Pareto-optimal front and not get trapped in a local front.

The third test problem is a well-studied constrained welded beam design problem [23]:

$$\begin{aligned} &\text{Minimize} \quad f_1(x) = 1.10471h^2l + 0.04811tb(14+l), \\ &\text{Minimize} \quad f_2(x) = \frac{2.1952}{t^3b}. \end{aligned} \tag{3}$$

$$\begin{aligned} \text{Subjected to} \quad & g_1(x) \equiv 13,600 - \tau(x) \geq 0, \\ & g_2(x) \equiv 30,000 - \sigma(x) \geq 0, \\ & g_3(x) \equiv b - h \geq 0, \\ & g_4(x) \equiv P_c(x) - 6,000 \geq 0, \\ & 0.125 \leq h, b \leq 5, \\ & 0.1 \leq l, t \leq 10. \end{aligned} \tag{4}$$

where:

$$\begin{aligned}
\tau(x) &= \sqrt{(\tau')^2 + (\tau'')^2 + \frac{l\tau'\tau''}{\sqrt{0.25(l^2+(h+t)^2}}}, \\
\tau' &= \frac{6,000}{\sqrt{2}hl}, \\
\tau'' &= \frac{6,000(14+0.5l)\sqrt{0.25(l^2+(h+t)^2)}}{2(0.707hl(\frac{l^2}{12}+0.25(h+t)^2))}, \\
\sigma(x) &= \frac{504,000}{t^2b}, \\
P_c(x) &= 64,746.022(1-0.0282346t)tb^3. \qquad (5)
\end{aligned}$$

This optimisation problem consists of four design variables and five inequality constraints. It is aimed to test the search algorithm functionality under constrains.

6.2 Experiment Setup

The PC-based implementation for comparison is the NSGA-II [13], which is fundamentally based on an elitist genetic algorithm. NSGA-II was simulated though MATLAB on a Intel(R) Core(TM)2 Duo CPU E8600 @ 3.33GHz, 3.49 GB of RAM. The proposed FPGA-based MDO was simulated on a Xilinx Virtex 4 (xc4vlx200-11ff1513). This section illustrates the FPGA-based MDO and NSGA-II results on the three tests problems. The parameters used for both algorithms and all the test problems are as follows:

- population size = 20,
- number of generations = 100,
- crossover rate = 0.9,
- mutation rate = 0.1.

6.3 Discussion of the Results

Figure 3 and Figure 4 shows the Pareto fronts captured for SCH1 and SCH2 respectively. The proposed FPGA-based MDO was demonstrated to be able to converge to the true Pareto-optimal front and not being trapped in a local front for the discontinuous problem (SCH2). In both problems the FPGA-based MDO found diverse set of solutions, about of the same quality as those found by the NSGA-II.

Figure 5 shows the performance of the algorithms of the constrained welded beam problem. Both algorithms were not able to converge to the true Pareto-optimal front after a low specified number of generations; the true Pareto front is found after 10000 generations of the PC-based NSGA-II. The FPGA-based MDO is able to find 4 solutions closer and better than NSGA-II after 100 generations. The fronts captured by both algorithms are also diverse; no clustering occurred.

Table 1 shows the computation time results for the test problems. The computation time was significantly enhanced, in particular for the welded beam problem where there was a speed improvement of 244 times. The NSGA-II, which is processing sequentially, is problem complexity dependent; as the objectives becomes more complex, the algorithm takes a longer time to process the information. Whereas the FPGA-based MDO search algorithm have no correlation with the problem definition because of its parallel processing, which accounts to the significant speed improvement when faced with the more complex welded bream problem.

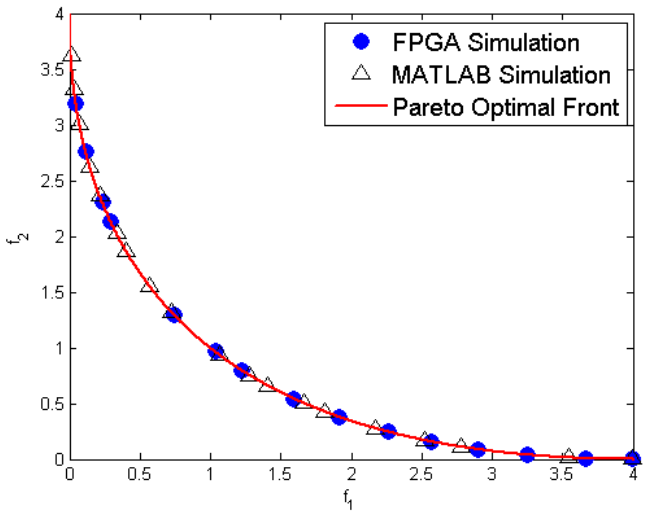

Figure 3: Pareto fronts obtained using MATLAB and FPGA for the SCH1 problem showing both algorithms converging to the true Pareto-optimal front.

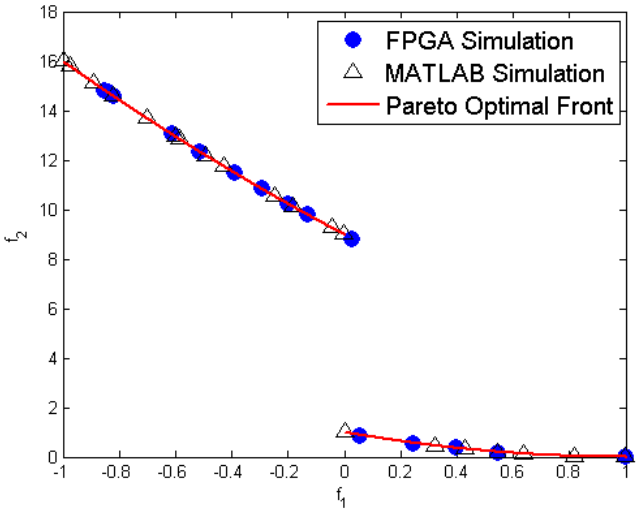

Figure 4: Pareto fronts obtained using MATLAB and FPGA for the SCH2 problem showing both algorithms converging to the true Pareto-optimal front.

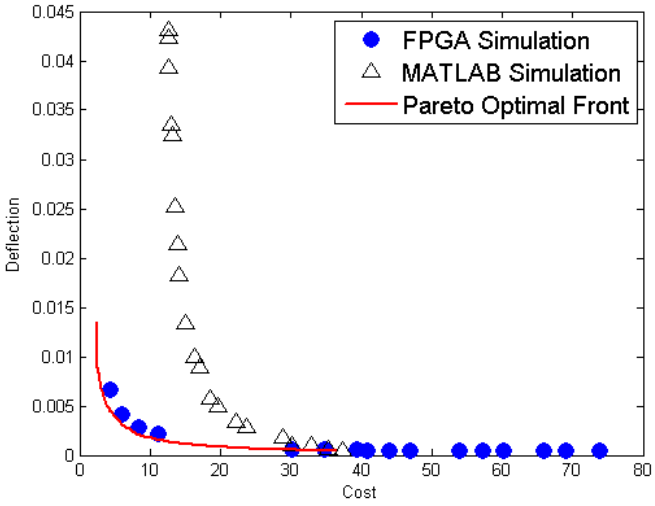

Figure 5: Pareto fronts obtained using MATLAB and FPGA for the welded beam problem showing both algorithms not converging to the true Pareto-optimal front.

Table 1: Computation time results showing the speed improvements

Test Problem	Computation Time (ms)		Speed Improvement (X)
	NSGA-II	FPGA-Based MDO	
SCH1	232	7.542	30
SCH2	354	8.128	43
Welded Beam Problem	1805	7.392	244

7 CONCLUSIONS

We have proposed a computationally fast and effective search method to address the increasingly complex field of Multi-disciplinary Design Optimisation (MDO). On three different types of test problems, which pertained to discrete, discontinuous and constrained problems, the proposed FPGA-based MDO was able perform computationally better than NSGA-II, with speed improvements of over 30 times, and producing results of about similar quality. However, with the welded beam problem, the FPGA-based MDO with its data represented in predefined 32-bit fixed-point widths was not able to attain a wider Pareto-front. In the aeronautics community, this optimisation method can be used on the ground for design problems (aerodynamics, structural etc.) and also practicable for onboard aeronautics application, where equipment size and power consumption are limiting current onboard capabilities.

Future work will look into improving the solution quality by analysing how fixed-point representation and resolution affects the Pareto-front found, especially when objective space of the problem is unknown beforehand. One possible research development is to dynamically alter the width of the fixed-point representation based on the search progress. Additionally, the computation time can be drastically improved by pipelining the flow of FPGA modules.

ACKNOWLEDGEMENTS

Computational resources and services used in this work were provided by the Australian Research Centre for Aerospace Automation (ARCAA) and the High Performance Computing and Research Group of the Queensland University of Technology (QUT), Brisbane, Australia.

REFERENCES

[1] J. Sobieszczanski-Sobieski and R. T. Haftka, "Multidisciplinary aerospace design optimization: Survey of recent developments," in *Structural and Multidisciplinary Optimization*, vol. 14, no. 1, 1997, pp. 1–23.

[2] J. Sobieszczanski-Sobieski, "Multidisciplinary design optimization: An emerging new engineering discipline," Rio de Janeiro, Brazil, 1993.

[3] S. Venkataraman and R. T. Haftka, "Structural optimization complexity: What has Moore's Law done for us?" in *Structural and Multidisciplinary Optimization*, vol. 28, no. 6, 2004, pp. 375–387.

[4] D. S. Lee, L. F. Gonzalez, J. Periaux, and K. Srinivas, "Robust design optimisation using multi-objective evolutionary algorithms," *Computers and Fluids*, vol. 37, no. 5, pp. 565–583, June 2008.

[5] S. D. Scott, A. Samal, and S. Seth, "HGA: A hardware-based genetic algorithm," in *Proceedings of the 1995 ACM/SIGDA Third International Symposium on Field-Programmable Gate Array*, 1995, pp. 53–59.

[6] C. Aporntewan and P. Chongstitvatana, "A hardware implementation of the compact genetic algorithm," in *Proceedings of the 2001 Congress on Evolutionary Computation*, vol. 1, 2001, pp. 624–629.

[7] 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, no. 1, pp. 33–60, March 2001.

[8] W. Tang and L. Yip, "Hardware implementation of genetic algorithms using FPGA," in *Proceedings of the 2004 47th Midwest Symposium on Circuits and Systems*, 2004, pp. 549–552.

[9] 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 Transactions on Evolutionary Computation*, vol. 14, no. 1, pp. 133–149, 2010.

[10] J. Kok, L. F. Gonzalez, R. A. Walker, T. Gurnett, and N. A. Kelson, "A synthesizable hardware evolutionary algorithm design for unmanned aerial system real-time path planning," in *Proceedings of the 2010 Australasian Conference on Robotics and Automation*, Brisbane, Australia, December 2010.

[11] K. Deb, *Multi-objective optimization using evolutionary algorithms*, 1st ed., ser. Wiley-Interscience series in systems and optimization. John Wiley and Sons, 2001.

[12] S. Bonissone and R. Subbu, "Evolutionary multiobjective optimization on a chip," in *Proceedings of the 2007 Workshop on Evolvable and Adaptive Hardware*, Honolulu, Hawaii, September 2007, pp. 61–66.

[13] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," *IEEE Transaction on Evolutionary Computation*, vol. 6, no. 2, April 2002.

[14] A. Osyczka and S. Kundu, "A new method to solve generalized multicriteria optimization problems using the simple genetic algorithm," *Structural Optimization*, vol. 10, no. 2, pp. 94–99, 1995.

[15] E. Zitzler and L. Thiele, "An evolutionary algorithm for multiobjective optimization: The strength pareto approach," Zurich, Switzerland, 1998.

[16] H. Kita, Y. Yabumoto, N. Mori, and Y. Nishikawa, "Multi-objective optimization by means of thermodynamical genetic algorithm," in *Proceedings of Parallel Problem Solving from Nature IV (PPSN-IV)*, 1996, pp. 504–512.

[17] A. the non-dominated front using the Pareto archived evolution strategy, "Evolutionary computation journal," *Computers and Fluids*, vol. 8, no. 2, pp. 149–172, 2000.

[18] D. Goldberg, *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison-Wesley, 1989.

[19] Z. Michalewicz, *Genetic Algorithms + Data Structures = Evolution Programs*. Springer-Verlag, 1992.

[20] M. Matsumoto and T. Nishimura, "Mersenne twister: A 623-dimensionally equidistributed uniform pseudo-random number generator," *ACM Transactions on Modeling and Computer Simulation*, vol. 8, no. 1, pp. 3–30, January 1998.

[21] K. Deb and R. B. Agarwal, "Simulated binary crossover for continuous search space," *Complex Systems*, vol. 9, pp. 115–148, April 1995.

[22] J. D. Schaffer, "Multiple objective optimization with vector evaluated genetic algorithms," in *Proceedings of the First International Conference on Genetic Algorithms*, 1985, pp. 93–100.

[23] G. V. Reklaitis, A. Ravindran, and K. M. Ragsdell, *Engineering Optimization Methods and Applications*. New York: Wiley, 1983.