Soft Computing Testing in Real Industrial Platforms for Process Intelligent Control

Larzabal E., Cubillos J.A., Larrea M., Irigoyen E., and Valera J.J.

University of the Basque Country (UPV/EHU), Intelligent Control Research Group
(GICI - gici.drupalgardens.com)
{ekaitz.larzabal,eloy.irigoyen}@ehu.es

Abstract. By testing advanced control techniques based on Soft Computing into industrial platforms is possible to analyse the feasibility and reliability of these implementations for being subsequently used in real industrial processes. In many cases, this fact is not taken into account for several reasons concerning with the complexity of performing hardware implementations. Hence, simulation testing becomes the last step before showing an implemented solution. The main objective of this work is to give a step beyond for achieving a more realistic test of the Intelligent Control techniques. For this reason, a first approximation of a Genetic Algorithm controller (NSGA-II) is implemented, tested, studied and compared in the stages of the controller design, and simultaneously in different industrial platforms. Most relevant results obtained in software simulation and in Hardware In the Loop (HIL) implementation are finally shown and analysed.

1 Introduction

Since many years, Intelligent Systems area has been presented as a new chance to solve tricky problems, being particularly relevant in coping with the intricacy and the complexity of the real world industrial process control [1]. As Rudas and Fodor present in their work Intelligent Systems [1], new fields emerged in this area developing computational solutions for the new approaches based on intelligence, such as Computational Intelligence [2], Soft Computing [3], and combining techniques from both fields, Hybrid Systems [4].

In both fields, Computational Intelligence (CI) and Soft Computing (SC), the GAs have recently appeared in the developed solutions in industrial control applications [5]. These solutions might be classified into two categories: one group for analysing and off-line design; other group for adapting and on-line controller tuning.There exist a short number of proposals where GAs directly calculate an on-line and real time control action, due to: (1) non-reliable computation, (2) high computational cost, and (3) problems in obtaining convergence and stability, such as discussed works in Fleming et al. [6] and Valera et al [7].

Currently, there exist lots of studies for solving industrial control problems based on ANN, as the work presented by Bose in Motor Drives [8], and based on FL, as the one presented by Precup in a survey on industrial applications of fuzzy control [9]. The principal gap of these works is to develop a real implementation in real industrial processes. Usually, all the performed studies are tested in an experimental stage with industrial

V. Snasel et al. (Eds.): SOCO Models in Industrial & Environmental Appl., AISC 188, pp. 221–230.
springerlink.com © Springer-Verlag Berlin Heidelberg 2013

process models which can include complex dynamics, but they are carried out in a simulation framework. Although there are multiple proposals based on SC techniques in the literature, few real developments can be found [8]. All of them are developed with laboratory equipment, without using real industrial platforms, such as industrial computers, embedded computers or programmable logic controllers. Some works are laboratory implementations with FPGA/DSP, or with a Host-PC configuration, testing communication and control issues, but not proving the robustness of a laboratory equipment in real industrial processes [8].

This work takes a step forward using GAs for intelligent control, providing a framework for the rapid prototyping and testing using industrial and usually used HW platforms in the Industry. In subsequent sections, promising results with different real platforms are presented. These results were obtained in several tests for controlling processes with complex dynamics and in solving optimization problems based on GAs with high computational cost requirements. Real tests have been carried out with two different industrial platforms; a robust industrial PC controller with Peripheral Component Interconnect (PCI) bus, and a PAC (Programmable Automation Controller). Beyond using GAs for solving optimization problems focused on tasks as planning, scheduling, tracking and calibration, this work introduces first results in intelligent controlling of processes with complex dynamics. Specifically, this development computes future values for control actions by an optimization process using GAs. This future values are computed for a predictive control scheme where, depending on the sample time, the computational cost could be very demanding.

There exist two fundamental problems on applying Model Predictive Control (MPC) strategies: (1) the accuracy of the model to approach the process or plant to be controlled, and (2) the control optimization problem to solve (specially for non-linear model predictive control, NMPC) during each controller sample time. Therefore, GAs as a part of Evolutionary Algorithms can offer a relevant possibility in high complexity optimization problem solving when process models with high non linearity are chosen and the selected cost functions are non-quadratic and non-convex [7].

Moreover, if the used prediction horizon is long, new difficulties appear in NMPC problem solving which have to be considered in real applications. On the one hand, the computational cost in solving the control problem with such horizons is high. On the other hand, this control strategy produces convergence problems in the optimization process, so in a general control formulation for working out a solution the computational cost will grow up as well. Furthermore, if it is necessary to control process systems with faster dynamics, a new problem appears when trying to perform an implementation in a Real Time (RT) control scheme, because of time requirements to be reached in each short sample time. This work tries to show how a control algorithm based on GAs can be executed in RT, on several industrial platforms and in shorter sample times. To this end, a Hardware In the Loop (HIL) testing configuration was prepared in the laboratory where controllers were implemented into two different industrial platforms.

In this work we present in section 2 the multiobjective GA approach used in our NMPC strategy. Subsequently, in order to implement the GA in real time with Matlab/Simulink and xPC-Target tools, several modifications and adjustments have been carried out and presented in section 3. Section 4 shows in a simulation context, two

different controlled systems which will be later used in real tests. The two real platforms tested in this work are presented in section 5. Results with both real platforms in RT tests are showed and explained in section 6. Finally, section 7 contains the last conclusions and future works.

2 The Genetic Algorithm NSGA-II

The optimization of multiple objectives problems, where any improvement in one of the objectives makes other objectives worse, has been a extensively explored research area. The optimal solutions obtained in such problems are denominated non-inferior solutions and all of them belong to the set of Pareto [10]. There exist several methods to search for the non-inferior (set of Pareto) solutions in the multi-objective optimization context. Among them the ones based on evolutionary algorithms stand out. Some of these contributions can be found in [11][6] and in [7]. Examples of efficient evolutionary algorithms such as Nondominated Sorting Genetic Algorithm (NSGA) and Micro-Genetic Algorithm (l-GA) are presented in [12] and in [13] respectively. The main drawbacks of these techniques are their high computational cost and the need of a decision maker to select one solution among the Pareto set. Other drawbacks related to the control context are the difficulty of demonstrating the stability, the convergence to a near global optimal and the robustness of the final solution

The NSGA-II used in this work is the one proposed by Deb et al. in [14]. The NSGA-II is the evolution of the NSGA originally proposed by Srinivas and Deb in [12]. This second version of the algorithm arose to answer the main criticisms (high computational complexity, lack of elitism and the need for specifying the sharing parameter) the NSGA received [14].

The possibility to tackle the multi-objective problems in the context of NMPC makes very interesting the NSGA-II algorithm. The introduction of the elitist mechanisms in the NSGA-II algorithm that improves the convergence time of the Pareto solutions is especially useful for control, where there exist specific problems as the time requirements (directly linked to the computational complexity of the controller algorithm). In [15] and [7], the authors takes the multi-objective NMPC scheme approach by using NSGA-II [12] to search the set of Pareto non-inferior solutions at each sampling time of the controller. For these applications, the NSGA-II works as the optimization solver of the NMPC problem at each sampling time.

The NSGA-II flowchart explained briefly starts with the initialization of the population (size N), evaluates the objective functions and ranks the population. The next step consists of a loop that ends when the stopping criteria is met. Inside the loop the following steps are taken in the following order; selection, crossover, mutation, evaluation of the objective functions, combination of population, ranking and finishes with the selection of N individuals. If the stopping criteria is met the final population is presented. See [14] for a detailed description of the NSGA-II algorithm.

NSGA-II was proposed as a Multiobjective GA although for this article it has been used with SISO plants with only one objective. It should be addressed that the scope of this work has been the preparation of a framework for the rapid prototyping and testing of intelligent control techniques in industrial control platforms. Future work will lead

to the implementation of multiobjective control strategies using GAs in RT as other use cases.

3 RT NSGA-II Programming for Predictive Control

The NSGA-II original code provided by Deb et al. in [14] was written in C. This code has been rewritten in a new s-function code for implementing in the Matlab/Simulink development environment. Some code reduction was made in order to minimize the coding size: e. g. lines related to the binary coding was removed because only real number coding was required. Also the controller time performance has been improved by making some modifications to adequate the NSGA-II in order to enhance its execution in short control sampling times. Finally, it is important to note that a new stop criteria has been added to include in the control strategy not only the GA iterations, but also the execution time.

In order to perform the predictive control with the NSGA-II algorithm, the objective function should be evaluated for the entire prediction horizon. The function evaluation is related to the error produced in the controlled variable of the control loop. To calculate this error the algorithm needs a model of the system to be controlled that can be obtained by mathematical approximation, neural network identification, etc. In this work, the first RT implementations have been performed by using mathematical models presented below. Furthermore, some tests have been made with neural network models performing a satisfactory control as well. Once the GA ends the searching (with a time based or a limited generation number stopping criteria) the last solution is used as predictive control action at each controller sampling time.

4 Simulation Results

In this section some simulation results are presented. For these experiments the two following nonlinear systems have been chosen from different benchmark systems:

System 1 (This is a modification of the system presented in [16]):

$$y_{k+1} = 0.8\left[u_k^3 + \frac{y_k y_{k-1}}{1+y_k^2}\right] \tag{1}$$

This system has been created for increasing the non-linear dynamics near the origin of coordinates of the original system.

System 2 (This system is presented in [17]):

$$y_{k+1} = \frac{1.5 y_k y_{y-1}}{1+y_k^2+y_{k-1}^2} + 0.7\sin\left[0.5\left(y_k+y_{k-1}\right)\right]\cos\left[0.5\left(y_k+y_{k-1}\right)\right] + 1.2u_k \tag{2}$$

For the simulation stage, the controller has been implemented in Matlab/Simulink. A batch of 50 simulations have been performed for each system. Figure 2 and Figure 3 show the evolution of the mean value in each batch of simulations. The system 1

performs with a 25% overshoot (M_p) while the system 2 performs with a 45% M_p. It is obvious this is not the best possible performance, but the implemented controller is quite simple and does not include any kind of constraint.

The system 1 output signal shows the following statistical values; $M_{p\_Max} = 0.3674$; $M_{p\_Min} = 0.3648$; $M_{p\_Mean} = 0.3661$; $M_{p\_Variance} = 2.537e-7$. It should be mentioned that the control action max. value never exceeds 0.7223.

The system 2 output signal shows the following statistical values; $M_{p\_Max} = 1.4984$; $M_{p\_Min} = 1.4941$; $M_{p\_Mean} = 1.4960$; $M_{p\_Variance} = 6.8509e-7$. It should be mentioned that the control action max. value never exceeds 0.8344.

The simulations show that both systems can be controlled with the proposed NSGA-II predictive controller. The simulation results are very satisfactory despite the implemented controller is quite simple and does not include any kind of restriction.

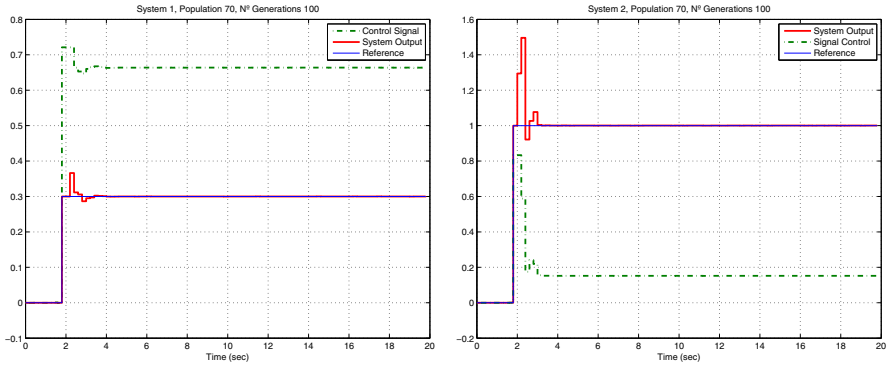


Fig. 1. Sample time 0.2 seg, Initial population 70, n generations 100. a) System 1 and b) System 2

Although the results may not be as satisfactory as they should, the scope of this work should not be lost. The HIL implementation of this first controller is challenging enough to continue with this first controller before trying to develop a very accurate one. All in all, more research has to be done in order to improve the controller and the results that it provides.

5 Rapid Control Prototyping and HIL Testing

Nowadays, the software development platforms provide accurate simulation results, but these simulations are not always enough to understand the real behaviour of the system. The next step to take should be the testing of the development in a real system. But the risk of a malfunction during the real testing is still there, and sometimes that risk is simply unacceptable. Hardware In the Loop arises to fill the gap between the software simulations and the real system implementation. The simulation gives a step towards the real implementation using the HIL test with the externalization of the signals. The HIL

can be used to simulate a single component or even the whole system replacing some of the parts by mathematical models [18]. The benefits of the HIL have been tested in many different industrial fields as automotive [19], electronics [20], wind energy systems [21], etc. Therefore, the Intelligent Control Research Group (GICI) of the UPV/EHU has adopted HIL methodology to test the different controllers under more real situations.

In the case of study of this article, the HIL test implies the externalization of the control signal u(k) and the system output signal y(k), as shown in Figure 2. The controller is going to be hosted in one platform and the system to be controlled will be placed in another platform. The configuration of the platforms and the data acquisition systems are listed in the table 1.

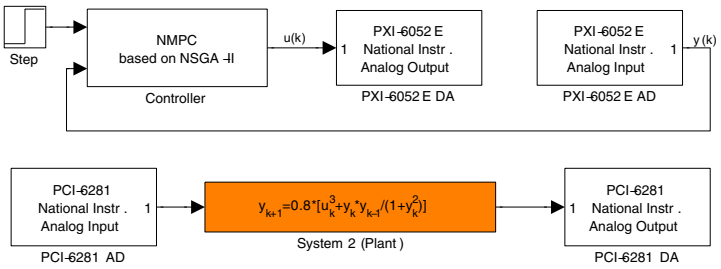


Fig. 2. Mathworks Matlab/Simulink block diagram

Table 1. Platform specification

Industrial PC / Kontron			
Host	Kontron	xPC-Target 4.0	NI pxi-6052 (16bits)
Target	PC	xPC-Target 4.0	NI pci-6281(18bits)
PAC / Beckhoff			
Host	Beckhoff	Xubuntu 8.04 / RTAI 3.8	K-Bus / KL3404 (AI) / KL4034 (AO) (12bits).
Target	PC	xPC-Target 4.0	NI pci-6221(16bits)

Both platforms have in common the target configuration for the system to be controlled. This system is composed by a PC with a National Instruments PCI Data Acquisition card described above. All the configurations and the SW/HW installation steps are explained in the *howtos* hosted in the GICI webpage.

6 Results (HIL Real Time Simulations)

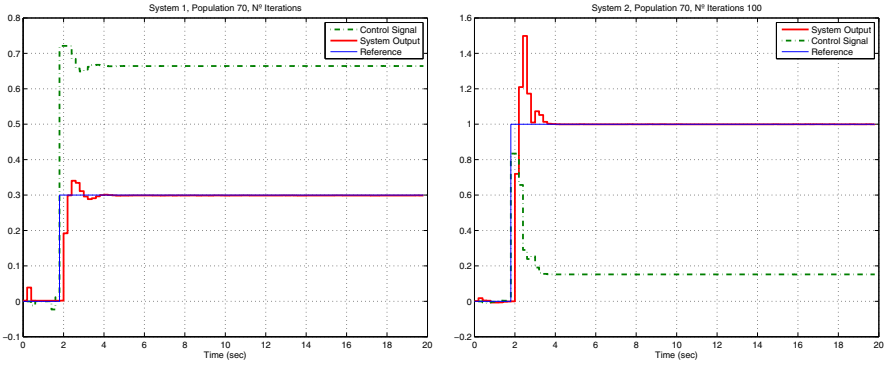
In this section some results are shown. The HIL experiments have also been performed with the two systems (System 1 and System 2) in 50 simulations, both running in the two platforms presented in the previous section: Industrial PC and PAC.

It should be remarked the difference in the bit resolution of the two platforms. The industrial PC analog I/O card has a 16bit resolution, whereas the PAC K-Bus analog I/O cards have a 12bit resolution. Also the NI PCI-6281 DAQ card has a 18bit resolution, while the NI PCI-6221 DAQ card has 16bit resolution. This difference is one of the reasons, but not the unique, of the noisier signal of the PAC platform.

Note that the system to be controlled is already running before the controller starts in all the results shown above.

6.1 Industrial PC (PCI Bus Based On)

The Industrial PC is a very powerfull and robust platform, with a similar core as the PCs but with an industrial bus (PCI). The results of this platform with the two systems under testing (1) and (2) can be seen in Figure 3(a) and Figure 3(b).



(a) (b)

Fig. 3. Sample time 0.2 seg, Initial population 70, n generations 100. a) System 1 and b) System 2

The system 1 output signal shows the following statistical values; $M_{p\_Max} = 0.3769$; $M_{p\_Min} = 0.3622$; $M_{p\_Mean} = 0.3649$; $M_{p\_Variance} = 1.3517e-5$. It should be mentioned that the control action max. value never exceeds 0.7229.

The system 2 output signal shows the following statistical values; $M_{p\_Max} = 1.7932$; $M_{p\_Min} = 1.4679$; $M_{p\_Mean} = 1.5673$; $M_{p\_Variance} = 0.0163$. It should be mentioned that the control action max. value never exceeds 0.8478.

The results of the Industrial PC are very similar to the ones presented in the simulation section.

6.2 PAC / Beckhoff: K−Bus

The PAC can be described as a less powerfull platform in comparison with the industrial PC, but the differences between both platforms are becoming less and less significant.

The PAC has the capability of being programmed with PLCs programming language, and it can be seen as a bridge between the PLCs and the industrial PCs. The results of this platform with the two systems can be seen in Figure 4(a) and Figure 4(b).

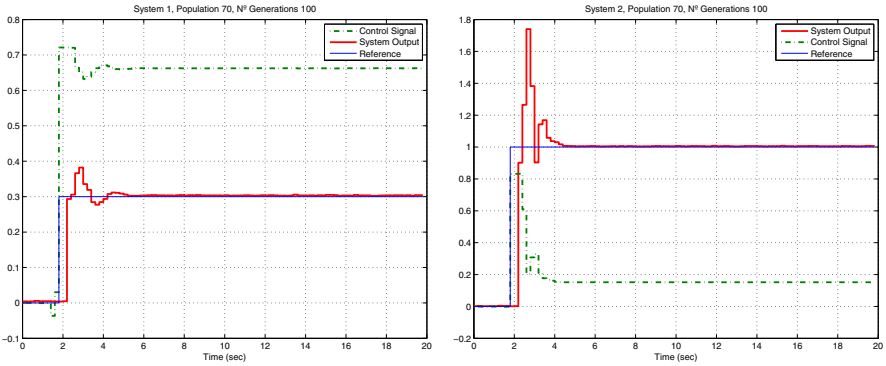


Fig. 4. Sample time 0.2 seg, Initial population 70, n generations 100. a) System 1 and b) System 2

The system 1 output signal shows the following statistical values; $M_{p\_Max} = 0.4006$; $M_{p\_Min} = 0.3713$; $M_{p\_Mean} = 0.3829$; $M_{p\_Variance} = 2.7136e-5$. It should be mentioned that the control action max. value never exceeds 0.7223.

The system 2 output signal shows the following statistical values; $M_{p\_Max} = 1.8117$; $M_{p\_Min} = 1.5871$; $M_{p\_Mean} = 1.7777$; $M_{p\_Variance} = 0.0020$. It should be mentioned that the control action max. value never exceeds 0.8404.

The results of the PAC are similar to the ones presented in the simulation section. Compared with the Industrial PC results, it can be seen that the PAC performs with a noisier signal. All in all, the results of the two platforms and the simulations are very similar.

The two platforms are capable of performing the 100 generations of the GA in the required time each sample time. In case of not performing the 100 generations a stop criteria has been implemented based in the required time to guarantee the RT and deterministic performance of the controller.

7 Conclusions

In this work the first step to implement Soft Computing techniques in industrial platforms have been taken in a Hardware In the Loop structure. A basic NMPC controller has been prototiped and tested in real industrial hardware platforms with promising results. These results demonstrate that present industrial platforms have enough computational capability to run advanced control strategies using intelligent and Soft Computing computation techniques. The rapid prototyping and testing framework presented in this article is very useful to make fast improvements in the algorithms in order to satisfy the hard real time and computational cost requirements.

The application of the NSGA-II algorithm in a NMPC strategy has been also presented as an example of prototyping and testing. Different steps for developing the control algorithm, from off-line simulations to rapid control prototyping and HiL testing, have been done, laying the basis for the implementation of future intelligent control strategies over real industrial controllers. All these steps have shown satisfactory results as the ones presented in this paper.

Future work will lead us to the implementation of multivariable and multiobjective intelligent-expert controls for highly non-linear and complex controlled plants. The combination of Neural Networks, Genetic Algorithms and Fuzzy Logic in advanced strategies is being a promise solution to optimize highly complex control problems. With the framework presented in this article the hybridization of those techniques is being investigated, implemented and tested easier.

Another research line will guide us to the implementation of a neural network identification system. Firstly, the offline identification should be implemented in RT (it has already been implemented in simulations) and secondly the identification should be done online. This improvement will help the controller to be less dependant of a mathematical model of the system.

Acknowledgement. This work comes under the framework of the project titled Implementation / Integration of Intelligent Control Techniques on Real Time Control Systems ATICTA-2 with reference S-PE10UN70 granted by the Basque Regional Government (GV / EJ). The work has also been funded by the Education, University and Research Department of the Basque Government (GV/EJ) BFI-2010-118 research fellowship.

References

1. Rudas, I.J., Fodor, J.: Intelligent systems. Proceedings of Int. J. of Computers, Communications & Control 3, 132–138 (2008)
2. Duch, W.: What is computational intelligence and where is it going. Challenges for Computational Intelligence 63, 1–13 (2007)
3. Bonissone, P.P.: Soft computing: the convergence of emerging reasoning technologies. Soft Computing 1, 6–18 (1997)
4. Sahin, S., Tolun, M., Hassanpour, R.: Hybrid expert systems: A survey of current approaches and applications. Expert Systems with Applications 39(4), 4609–4617 (2012)
5. Saridakis, K., Dentsoras, A.: Soft computing in engineering design – a review. Advanced Engineering Informatics 22(2), 202–221 (2008)
6. Fleming, P., Purshouse, R.: Evolutionary algorithms in control systems engineering: A survey. Control Engineering Practice 10(11), 1223–1241 (2002)
7. Valera, J., Irigoyen, E., Gómez, V., Artaza, F., Larrea, M.: Intelligent multi-objective nonlinear model predictive control (imo-nmpc): Towards the 'on-line' optimization of highly complex control problems. Expert Systems with Applications 39(7), 6527–6540 (2012)
8. Bose, B.K.: Neural network applications in power electronics and motor drives- an introduction and perspective. IEEE Trans. on Industrial Electronics 54(1), 14–33 (2007)
9. Precup, R.E., Hellendoorn, H.: A survey on industrial applications of fuzzy control. Computers in Industry 62(3), 213–226 (2011)
10. Pareto, V.: Cours d'Économie Politique, vol. I & II. Université de Lausanne (1897)

11. Deb, K.: Multi-objective optimization using evolutionary algorithms. John Wiley & Sons, Ltd. (2001)
12. Srinivas, N., Deb, K.: Multiobjective optimization using nondominated sorting in genetic algorithms. Evolutionary Computation 2, 221–248 (1994)
13. Toscano Pulido, G., Coello Coello, C.A.: The Micro Genetic Algorithm 2: Towards Online Adaptation in Evolutionary Multiobjective Optimization. In: Fonseca, C.M., Fleming, P.J., Zitzler, E., Deb, K., Thiele, L. (eds.) EMO 2003. LNCS, vol. 2632, pp. 252–266. Springer, Heidelberg (2003)
14. Deb, K., Pratap, A., Agarwal, S., Meyarivan, T.: A fast and elitist multiobjective genetic algorithm: Nsga-ii. IEEE Trans. on Evolutionary Computation 6(2), 182–197 (2002)
15. Laabidi, K., Bouani, F., Ksouri, M.: Multi-criteria optimization in nonlinear predictive control. Mathematics and Computers in Simulation 76(5-6), 363–374 (2008)
16. Narendra, K.S., Parthasarathy, K.: Identification and control of dynamical systems using neural networks. IEEE Trans Neural Networks 1(1), 4–27 (1990)
17. Harris, C. (ed.): Advances in Intelligent Control. CRC Press (1994)
18. Hanselmann, H.: Hardware-in-the-loop simulation as a standard approach for development, customization, and production test of ecu's. Technical Report 931953, SAE Int. (1993)
19. Kendall, I., Jones, R.: An investigation into the use of hardware-in-the-loop simulation testing for automotive electronic control systems. Control Engineering Practice 7(11) (1999)
20. Lu, B., Wu, X., Figueroa, H., Monti, A.: A low-cost real-time hardware-in-the-loop testing approach of power electronics controls. IEEE Trans. on Industrial Electronics 54(2) (2007)
21. Li, H., Steurer, M., Shi, K., Woodruff, S., Zhang, D.: Development of a unified design, test, and research platform for wind energy systems based on hardware-in-the-loop real-time simulation. IEEE Trans. on Industrial Electronics 53(4), 1144–1151 (2006)