Proceedings of the 21st International Conference on Modelling and Applied Simulation (MAS), 026
19th International Multidisciplinary Modeling & Simulation Multiconference

2724-0037 © 2022 The Authors.
doi: 10.46354/i3m.2022.mas.026

Real-time wind turbine simulation for pitch control purposes by using a hardware-in-the-loop approach

David Kenko[1] and Adrian Gambier[1,*]

[1]Fraunhofer IWES, Fraunhofer Institute for Wind Energy Systems, Am Seedeich 45, Bremerhaven, 27572, Germany

[*]Corresponding author. Email address: adrian.gambier@iwes.fraunhofer.de, agambier@ieee.org.

Abstract

Because of the fact that large wind turbines are not available for research experiments and downscaled models are characterized by having different behaviors than the large ones, simulation tools become significant not only for the study of the machine dynamics but also for the control system design. However, pure digital simulation alone is insufficient for studying the performance of control algorithms since real-time operation is required for such an analysis. As a result, an interacting approach that combines wind turbine simulation and Hardware-in-the-Loop (HiL) control emerges as a promising solution. Thus, the current work proposes a system for real-time simulation and control of large-scale wind turbines based on a Hardware-in-the-Loop approach, in which a simulation tool is combined with commercial control hardware for wind turbines. The hardware-software architecture is discussed. Finally, the practical convenience of the developed system is illustrated by means of a numerical study.

Keywords: Large wind turbines; pitch control; hardware in the loop; real-time simulation

1. Introduction

The constant size growth of wind turbines (see the illustrative Fig. 1), which results in a major increase in machine and system complexity, entails, as a consequence, a more advanced and demanding control system.

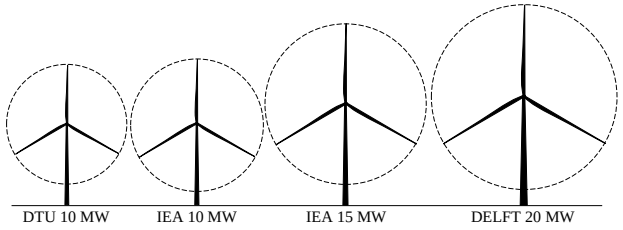

Figure 1. Comparative description of several large reference wind turbines.

An important limitation to design the wind turbine control system is the fact that there are no experimental wind turbines of such a size that allows studies to be carried out with a realistic perspective.

One way to overcome this type of challenge is provided by HiL systems, which basically consist of a simulation environment that mixes purely mathematical models with real physical components. HiL techniques for control purposes comprise a mathematical dynamic model implemented in a simulation software and physical components embedded in the control loop, in this instance, the control hardware.

Although the origins of HiL systems are unknown, there are many references that suggest that they have existed for a very long time. The first indications of the presence of this technology date back to the beginning of the twentieth century, when it contributed to the development of aeronautics by means of flight simulators (see, e.g., Evans and Schilling, (1984), Bailey and Doerr, (1996)). A survey with an historic perspective is given in Brayanov and Stoynova, (2019).

However, HiL systems are nowadays used in practically all engineering areas, for instance, in the in the automotive industry (see Kiffmeier, (1996), Hanselmann, (1996) and in power systems (e.g., Faruque and Dinavahi, (2010), Viehweider et al., (2011)). They have also been used to study power systems (wind energy converters from the electrical point of view, Steurer et al., (2004) and grid phenomena, Roscoe et al., (2010), Viehweider et al., (2011). Very large hardware-in-the-loop simulators for the investigation of wind turbines nacelles have been presented in Leisten et al., (2017) and in Neshati et al., (2016).

Hardware-in-the-loop controllers for wind energy systems are not easy to find in the literature. A preliminary work of this HiL simulator for control has been presented by Basilios and Gambier, (2020). The basic idea of this HiL simulator is to study new approaches for the control of wind energy converters by using a more realistic environment. In the present contribution, the focus is set on the description of the control strategies and the control problems of very large wind turbines. Thus, Section 2 is devoted to describe the wind turbine control. In Section 3, HiL systems for control purposes are presented. In particular, real-time simulation and real-time control are contrasted. The hardware-in-the-loop design is the subject of Section 4. In Section 5, a very large wind turbine of 20 MW is studied. Finally, conclusions are drawn in Section 6.

2. Fundamentals on Wind Turbine Control

2.1. Control Strategies and Operational States

A wind turbine has many operation states depending on the wind speed. However, only the two production states are of interest for the present work, namely the partial load operation and the full load operation.

Depending on the wind speed, the operation of the wind turbine can be separated into four regions. The machine is unable to produce in the first region because the wind speed is less than the design *cut-in* value. It enters into the second region, i.e., the partial load operation, when the wind speed exceeds the cut-in value. Here, the wind speed is sufficient to produce but the machine is not able to reach the nominal values. Thus, the control objective is to generate as much power as possible. The control variable is the electromagnetic torque, which is manipulated by means of power converters.

If the wind speed surpasses the nominal value, the machine moves to the full load states, also known as Region III. The control goal in this region is to keep the rotational speed (and indirectly, the power) constant by pitching the rotor blades in the feather direction. If the wind speed exceeds the cut-out value, the wind turbine reaches the fourth region and has to shut down.

2.2. Pitch Control in Full Load Operation

The simplest pitch control system is called collective pitch control (CPC). It uses a measurement of the generator rotational speed to build the control error from the nominal rotational speed used as a set point. The controller delivers a unique pitch angle that is passed to the three pitch actuators. As a controller, a PI control (proportional integral) law with gain scheduling and an anti-windup mechanism for saturation in magnitude and rate is used.

2.3. Active Tower Damping Control

The pitch activity introduces oscillation on the tower, Bossanyi, (2000), which can be reduced by using active tower damping control (ATDC). The ATDC uses the concept of damping injection, Takegaki and Arimoto, (1981), which can be provided by a PD controller (proportional derivative), Visioli, (2006). However, the P part is set here to zero in order to avoid shifting the first natural frequency of the tower dynamic.

2.4. Control laws

In full load operation, at least three control loops become active. First, the torque control continues working in the same way as it does in the partial load operation. The typical control law is a proportional quadratic one with inertia injection based on an additional derivative term Burton et al., (2011). The formulation is given by

$$T_g = K_1 \omega_g^2 + K_2 \dot{\omega}_g \,, \tag{1}$$

where ω_g is the rotational generator speed, T_g is the reference torque delivered to the power converter, K_1 is the gain for power extraction, and K_2 is used in very large machines for the reduction of the rotor inertia.

The second control loop limits the rotational speed by collectively pitching the blades. The standard control law used for this operation is the PI controller with anti-windup technique given by

$$U(s) = f_p E(s) + \frac{1}{s}\left[f_i E(s) - f_a [U(s) - U_a(s)] \right] \tag{2}$$

and depicted in Figure 2.

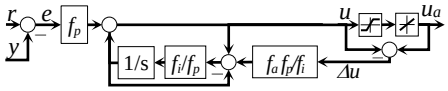

Figure 2. PI controller in the automatic reset configuration with

Due to the fact that the system has three actuators an only one controller the anti-windup mechanism is more complex. The scheme. proposed in Gambier, (2022) is illustrated in Figure 3.

The third control law is for the ATDC according to Gambier, (2022), namely

$$\Delta\beta_{atdc}(t) = -\frac{(\gamma_t - 1) D_t}{(\partial F_t / \partial \beta)_{\beta_0}} \dot{x}_t(t) \,, \tag{3}$$

where D_t is tower modal damping, $x_t(t)$ tower top displacement and $\gamma_t > 1$ is a design parameter. $\partial F_t/\partial\beta$ is the sensibility function at the current pitch angle, which works as a scheduling parameter to correct the gain when the wind speed changes, providing a gain-scheduling adaption.

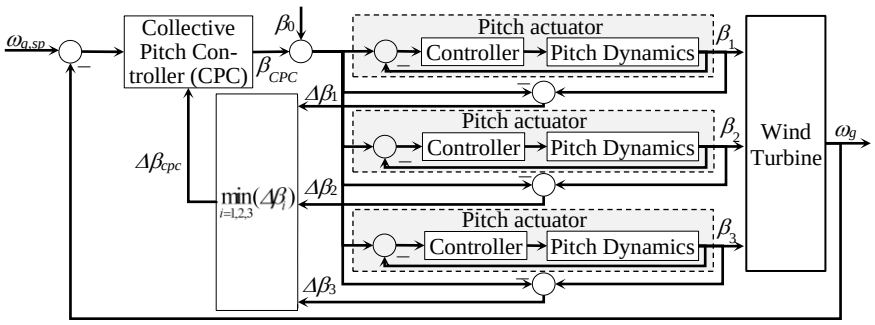

Figure 3. Anti-windup technique for the collective pitch control.

3. Hardware in the loop for real-time control

The concepts of simulation, modeling, real time, and hardware in the loop are broadly used with varying definitions and meanings depending on the context and discipline in which they are employed. To minimize misperceptions and confusions, this section introduces and clarifies such terms according to their use in this work.

3.1. Modeling for control purposes and simulation

In the present framework, a model is defined as a mathematical description of a physical component or subsystem. Depending on the nature of the internal states of a dynamic system, they can be described by algebraic-differential equations, if only continuous states are present, by a discrete formalism (e.g., state machines) if they consist only of discrete states, or by a hybrid formalism (e.g., a hybrid automaton) if the system has continuous and discrete states. Time-discrete systems belong to the first group, but difference equations are used instead of differential ones.

The differential equations of a model with continuous states, which can be found in a wind turbine, can be linear, nonlinear or with partial derivatives. The term "simulation" is used here to refer to the process involved in the numerical solution of a set of the above-mentioned equations by using a computer program. The computer code is called "solver" and implements the numerical algorithm used to solve the differential and algebraic equations.

3.2. Comparing real-time simulation and real-time control

From the numerous definitions of real time portrayed in the specialized literature, one is chosen to be used here that emphasizes time as an essential variable and timing limitations affect the delivery of results:

> *The correctness of an operation in a real-time environment is determined not only by the operative correctness of the outcome but also by fulfilling a preestablished deadline by which this outcome must be available.*

The aforementioned tasks are frequently concerned with the control system or with the system's response in "real time" to events occurring outside of it. As a result, real-time activities have to respond without delay to requests triggered by both planned internal tasks and external associated events (see Gambier, 2004). Hence, the system responds deterministically in all situations, even when the task answers are constrained by time restrictions set by rigorous deadlines.

Multitasking real-time systems can be developed by using one of two concepts. One uses a concurrent real-time programming language, while the other is based on an ordinary programming language, with real-time duties commissioned to a real-time operating system (RTOS, Burns and Wellings, (2009)).

Thus, for the realization of real-time systems, RTOS provide services such as multitasking (i.e., concurrency and parallelism), scheduling, and inter-task communication procedures. For example, QNX, Kim et al., (2010), VxWorks, Liu et al., (2017), RT-Linux, Abbott, (2018), and LynxOS, Garcia, (2017) are recognized RTOS products. There is also embedded real-time software, which offers real-time services for specific hardware, such as for example, cellphones and programmable logic controllers (PLC).

When the term real time is used to describe simulation software, the strict meaning is different. According to Isermann et al., (1999), simulation software operates in real time if the inputs and outputs of the simulating and real systems are governed by the same dependence in time. Nevertheless, the solver of the simulation software must be run inside a real-time task to fulfill the condition of the same time dependence. The time-step of the solver is thus governed by the dynamics of the simulated system. Hence, the deadline in the real-time task must be established at the end of the solver's integration step in order to synchronize both simulation and real time.

3.3. Solvers for real-time simulation

A solver is a computer implementation of an algorithm for the numerical integration of ordinary differential equations. Runge-Kutta (RK), Adams-Bashforth (AB), Adams-Moulton (AM), or Adams-Bashforth-Moulton (ABM) are examples of two-pass predictor-corrector algorithms used by modern solvers (see Heath, (1997)).

The Adams-Bashforth algorithm is normally chosen for solvers in real-time simulation environments, Howe, (1991), because of its characteristic of real-time inputs, i.e., it does not require the inputs $\mathbf{u}(n+1)$, which

are unknown in a real-time context, to calculate the derivatives in step n+1. In contrast, Adams-Moulton, Adams-Bashforth-Moulton and Runge-Kutta algorithms do require knowledge of these inputs. However, Howe, (1989), introduced modifications in ABM and AM such that the algorithms could also be used in real-time simulation.

The errors caused by the solver are of the order n of the approximation given by the integral time-step t_i, i.e., $O(t_i^n)$. Solvers frequently use order four for their algorithms. This implies an error in the order $O(t_i^4)$. Fourth order algorithms are called AB4, ABM4 and RK4.

3.4. Simulation timing and real-time control

The determinist behavior of the system is achieved through the use of a constant integration step, which avoids recalculations and additional iterations that are common in adaptive algorithms, resulting in a more predictable computing time.

At the moment of selecting the integration step, stability conditions must be considered. This is reached choosing the time step to be less or even negligible in comparison to the system dynamics represented by the highest eigen frequency, Khaled-El Feki, (2014). Under this consideration, Balla, (2011), select the time-step as

$$t_i = 1/(\pi f_{max}). \tag{4}$$

The maximum frequency is given by $f_{max} = \omega_{max}/(2\pi)$, where ω_{max} represents the maximum undamped eigen frequency of the system. Thus, the integration step becomes

$$t_i = 2/\omega_{max}. \tag{5}$$

On the other hand, the system presented here combines real-time simulation and control, and therefore, the Shannon's theorem should also be satisfied. Hence, the sampling time T_0 for practical applications of control systems (see e.g., Åström and Wittenmark, (1997)) has to satisfy

$$T_0 = 1/(\kappa f_{max}) \tag{6}$$

for $2 < \kappa \leq 10$. Since the solver should return many values of the solution within a sampling time to ensure that the variables are "continuous", the time step can be set to

$$t_i = \eta T_0 = (\eta/\kappa)(1/f_{max}), \tag{7}$$

for $\eta \leq 0.1$. Thus, in order to preserve continuous time emulation, the constant time-step should be chosen to be (κ/η) times lower than the smallest time constant of the simulated dynamic system.

Because the estimation of the time-step is an empirical procedure and is closely linked to the application, the time-step can be set as a prioritized balance between stability, precision, and computational burden.

Advanced integration algorithms are implemented in two stages. In the first one, a prediction is carried out, and, in the second one, a correction is applied in order to improve the accuracy.

From (7), an AB4 algorithm will produce errors in the order given by

$$O(t_i^4) = \eta^4/(\kappa^4 f_{max}^4) \tag{8}$$

and considering the empiric range $\pi \leq \eta/\kappa \leq 100$, it follows

$$10^{-8}/f_{max}^4 \leq O(t_i^4) \leq 10^{-2}/f_{max}^4. \tag{9}$$

Thus, a compromise can be found for κ in the selection of the sampling period such that, for the maximum natural frequency, an acceptable integration error is obtained.

3.5. Hardware-in-the-loop control concept

Different concepts of Hardware-in-the-Loop can be found in the literature. They depend mainly on the discipline and, within it, on the subject. More detailed descriptions are provided, for instance, in in Sarhadi and Yousefpour, (2015), Bacic, (2005), Bélanger et al., (2010).

Since the interest in the present work is to study advanced control approaches in a realistic real-time environment, the hardware that is installed "in the loop" is precisely the real-time hardware that executes the control software, including the low-level control and the supervisory control.

As a large wind turbine is generally not available for experimental studies, it is replaced by a virtual machine simulated in a real-time environment using a high-resolution model. The concept obtained following this idea is depicted in Figure 4.

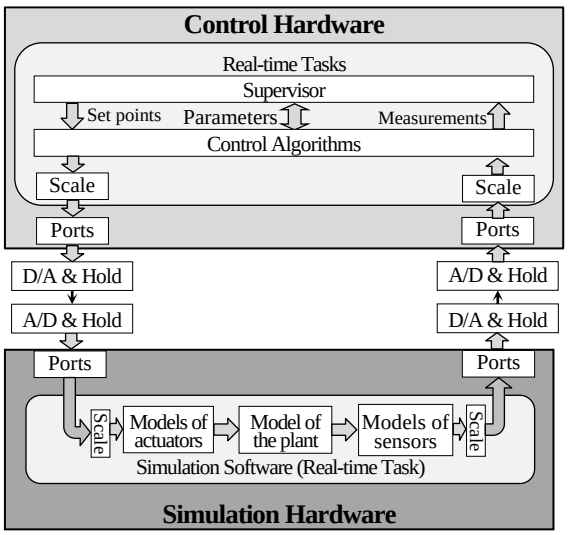

Figure 4. Control system in the hardware-in-the-loop concept

The control and simulation systems must be synced due to the fact that they run on different machines. As a result, the simulation must run in real time. That is, the numerical integration must satisfy deadlines.

4. Hardware-in-the-loop design

The design of the hardware-in-the-loop architecture includes the concept, the hardware configuration, the software selection and the synchronization. These are explained in the subsections bellow.

4.1. General architecture

The general architecture is depicted in Figure 5.

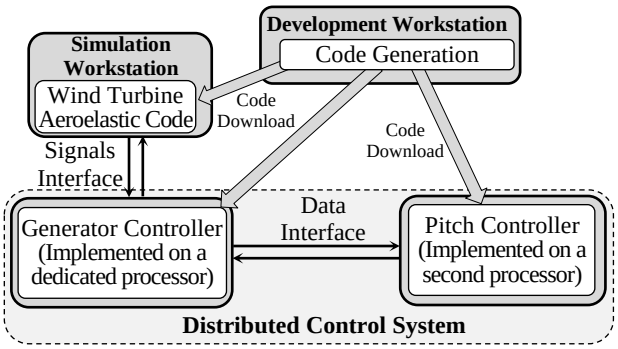

Figure 5. HiL architecture concept for wind turbine control

The overall system architecture consists of a workstation used with the dual function, wind turbine simulator and development system, and a dedicated hardware for the control operations. Both systems need interfaces and real-time capability.

4.2. Concepts for hardware design

As simulator and development system, a Windows® Workstation with an Intel® Core™ i7 Extreme edition processor is chosen. This decision leads to the need for compatible embedded interface cards, which, in addition, must satisfy the condition of having several inputs and outputs (A/D, D/A and I/O channels) and, moreover, being supported by the Simulink Desktop Target. Finally, two identical cards from the company Humusoft® were chosen, resulting in 16 channels of each type.

For the control hardware, several possibilities are available on the market as well. However, it is of interest to reduce, at a minimum, the development burden. Hence, equipment that meets the requirements

- support for Simulink (implementation based on a blockset and code generation for the hardware),
- standard hardware used in real wind turbines, and
- real time provided by a standard real-time operating system

is prioritized in the analysis. All these features are provided by Bachmann®'s M1 industrial platform, which includes with two CPUs (MC210 and MX213) and implements VxWorks® from Wind River as an RTOS. The interconnection between the interface cards and the Bachmann modules needs signal conditioning adapters. The system is shown in Figure 6.

Figure 6. Hardware-in-the-loop system for wind turbine control

4.3. Software for the real-time simulation

The dynamic simulation of wind energy converters requires a very complex and aero-servo-elastic code, whose development and maintenance involves many years. In addition, it has to run in a real-time task. Although there exist several tools for this purpose (see, e.g., Cp-Lambda from Politecnico di Milano, Bottasso and Croce, (2009), Bladed from DNV GL, Bossanyi, (2003), HAWC2 from the Denmark Technical University, Larsen and Hansen, (2014), and QBlade from the Technical University of Berlin, Pechlivanoglou et al., (2010), the code FAST (and more recently OpenFast) from National Renewable Energy Laboratory, Jonkman and Buhl Jr., (2005), offers several attractive characteristics for the current implementation:

- the source code is available and can be modified,
- it is modular organized and documented,
- a Matlab/Simulink interface is provided, and
- the B4 algorithm is implemented as solver.

The Matlab/Simulink interface is an important feature because some tools like Simulink Desktop Real-Time and the Simulink Coder can be used to generate real-time code from the FAST library that may run in a soft real-time task where the integration step can be set up to 1 ms.

Hence, despite the time restrictions imposed on the simulation, the wind turbine can be simulated in the real-time like environment offered by Simulink Desktop Real-Time. A limitation at present is given by the fact that the interface uses calls to API (Application Programming Interface) functions that the Simulink Coder does not yet support, and hence, a hard real time implementation is not possible.

4.4. Configuration and system synchronization

The synchronization between real-time control and simulation must be carried out on both sides. On the control hardware side, VxWorks offers the required synchronization service. Contrarily, the simulation code does not provide a synchronization facility, and therefore, an alternative solution has to be provided.

To this end, the FAST output variable "simulation time", which is characterized as a monotonic increas-

ing piecewise constant signal (see Figure 7), is passed through an edge detector to generate a pulse train to wake-up the control tasks at the end of the time-step, which are in a waiting state in the real-time control hardware.

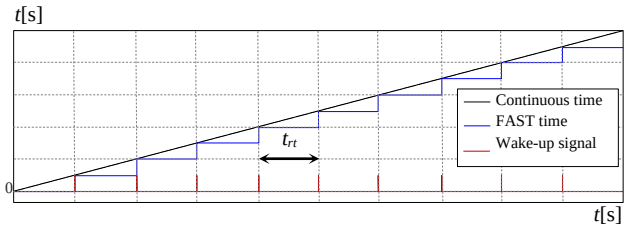

Figure 7. Synchronization procedure for the HiL system

The integration time step defined inside the solver is now denoted by t_{FAST} and the computational time that the solver needs to calculate the solution of the differential equations for the time step t_{FAST} is symbolized by t_{rt}. The real-time operation requires meeting $t_{rt} < t_{FAST}$.

The solver needs the input signals at the start of the integration time-step and then they are kept constant throughout the time-steps until the sampling period T_0 ends. To continue the simulation during the next sample period, new values of the input signals must be available at this time point. Hence, it is necessary that the entire sequence of time-steps be completed within the sampling period while leaving enough margin for the controller to compute the next values of the control signals before moving on to the next simulation step.

In addition, the solver provides the new output values after $m\ t_{rt}$ steps, which means that the outputs are available to compute the new values of the control variables when the wake-up signal triggers the waiting control tasks. The consequence is that control tasks must provide the new values of the control variables in the interval $\Delta t = T_0 - m\ t_{rt}$. The sequence is exemplified in Figure 8.

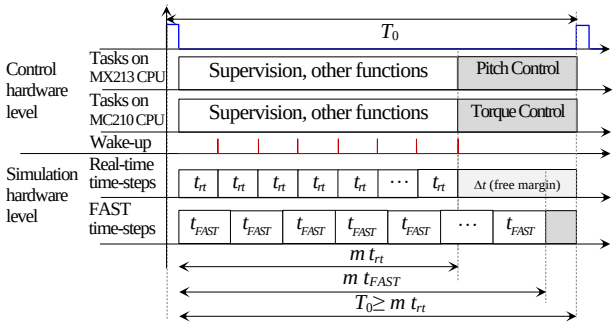

Figure 8. Time diagram for the HiL system

5. Application study

The hardware-in-the-loop architecture has been tested with several examples based on reference wind turbines. A particular application study is presented in the following subsections.

5.1. Description of the wind turbine

The study is based on a three-bladed, horizontal axis, variable-speed and variable-pitch, upwind wind turbine, whose reference model is presented in Ashuri et al., (2016). The control of this machine is studied by Gambier and Meng, (2019).

The rotor is 276 meters in diameter, the blades are 135 meters long, and the hub is 6 meters in diameter (see Figure 9). Each blade with a mass of 259 tonnes is split into 20 sections and includes six distinct airfoils. The total mass of the rotor is 839.3 tonnes, with a rotational inertia of 2.92×10^9 kg m^2. The nacelle is characterized by a mass of 252.8 tonnes and is supported by a tower whose hub height is 160.2 m, the

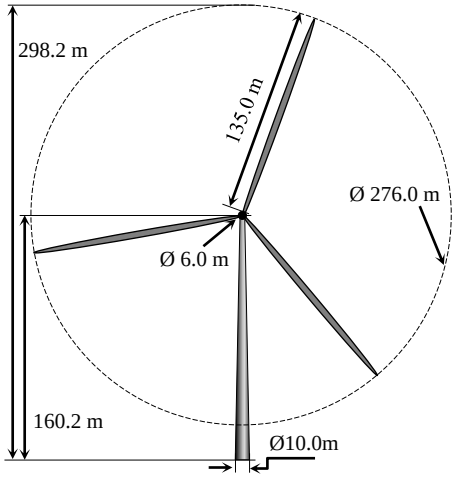

Figure 9. Descriptive scheme of the 20 MW wind turbine

bottom diameter is 10 m and the top diameter is 6.2 m.

The drive train has a damping constant of 4.97×10^7 Nm/(rad/s) and an equivalent spring constant of 6.9411 Nm/rad. A rated electrical power of 20 MW equates to a rated mechanical power of 21.19 MW for a generator efficiency of 0.944. At a tip-speed ratio of 9.51, the maximum power factor $C_{p,max}$ lies by 0.47268. For a rated rotor speed of 7.16 rpm, the gearbox ratio provides a rated generator speed of 1173.7 rpm, yielding a rated wind speed of 10.715 m/s.

Natural angular frequencies are localized between

$$0.0174\,\text{rad/s} \leq \omega_n \leq 11.7596\ \text{rad/s}\ , \qquad (10)$$

and therefore, the maximum natural frequency is f_{max}=11.7596/(2π) = 1.8716 Hz. This leads to a sampling period of T_0 = 0.05 s. Following (7), a value κ = 0.25 is calculated for an integration time-step t_{FAST} = 0.0125. This means that in one sampling period, four integration time-steps can be completed.

The real computational time of the simulation system has been measured to be 2.6 ms, i.e., t_{rt} = 0.0026 s. Hence, the control hardware has 42 ms to calculate the new values of the control variables, and the AB4 algorithm will work with an error of the order of $O(t_i{}^4) = 3.18\ 10^{-8}$.

5.2. Experiment for the Simulation Environment

The high-resolution model of the above-presented 20 MW wind turbine is simulated by using FAST running in the real-time environment. The rated wind speed is 10.715 m/s, which leads to a rotor speed of 7.157 rpm.

Hence, the wind turbine is driven by stochastic wind, whose effective wind speed, which corresponds to a Kaimal spectrum with turbulence of 10 %, wind shear and tower shadow, is used as profile of the effective wind speed. The wind profile is set for the operation in Region III, where the wind speed is over the rated value of 10.715 m/s and bounded by 22 m/s. Figure 10 illustrates 10 minutes of it.

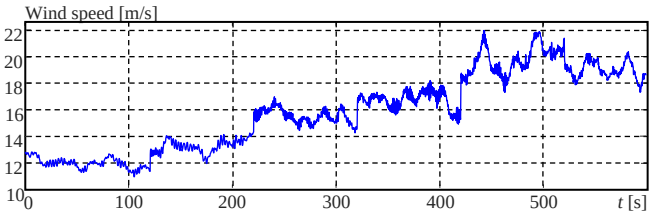

Figure 10. Effective wind speed signal with a turbulence value of 10%.

5.3. Simulation results

The FAST library embedded in the real-time simulation environment is employed as a virtual very large wind turbine for control experiments. For this purpose, the high-resolution model of a 20 MW wind turbine is used.

Nonetheless, the major goal of this study is to ensure that the developed hardware-in-the-loop system operates properly. Namely, a high-resolution model can be run in a real-time setting in order to support research and development in wind turbine control systems. It could be verified that the communication between control and simulation hardware is accurate, maintaining the real-time conditions. Moreover, there is sufficient time available for the calculation of the control signals.

Finally, the control signals in the pitch control loop with anti-windup technique are calculated by the MX213 and delivered to the simulation hardware to regulate power in the case of over-rated wind speed.

On the other hand, torque control, pitch control and active tower damping control are designed for the aforementioned goal and can still be enhanced and improved. The simulation curves are shown in Figure 11 (a, b and c).

The pitch control system is activated when the wind speed is over-rated. Since the main control objective of the pitch control system is to keep the power constant and limited to the rated value for all values of the wind speed over its rated value (10.715 m/s), the pitch controller works correctly as shown in Figure 11 a.

6. Conclusions

In this work, a hardware-in-the-loop controller for the study of control systems of very large wind turbines is described. Control strategies and control problems are presented and the HiL design is analysed. Moreover, the synchronisation of the real-time simulator and the real-time control hardware has been studied. Finally, a numerical example is presented.

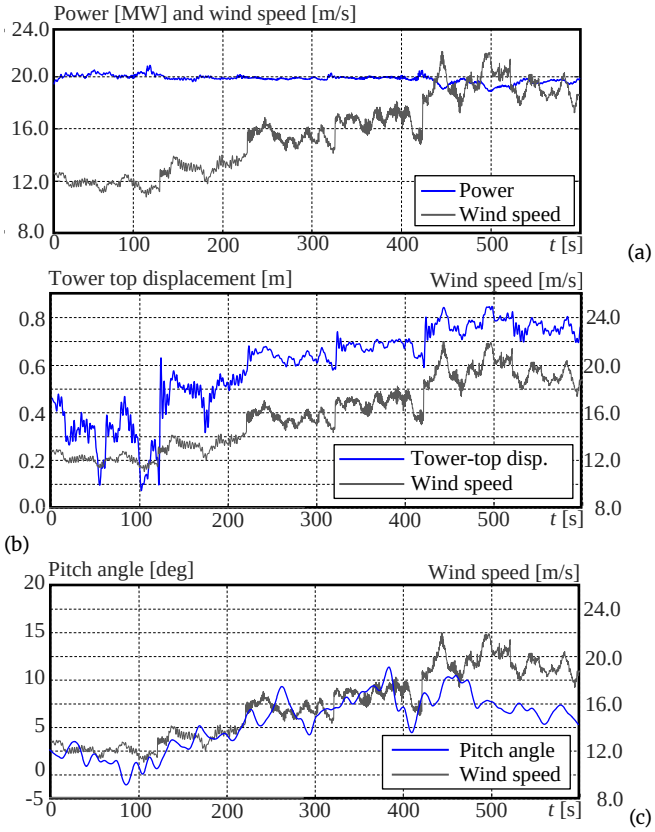

Figure 11. (a) Generated power with collective pitch control. (b) Tower top oscillations with tower damping control. (c) Pitch control signal

The example shows that all requirements are satisfied and that the HiL simulator is useful for the purpose of testing new control approaches as well as developing new control concepts. The next step in the development is to implement the supervisory control system in order to obtain a complete control system for all operational states.

Funding

This work has been financed by the Federal Ministry of Economic Affairs and Climate Action (BMWK).

References

Abbott, D. (2018). *Linux for Embedded and Real-time Applications 4th Edition*, Newnes (Elsevier), Oxford, UK.

Ashuri, T., Martins, J. R. R. A., Zaaijer, M. B., van Kuik, G. A. M. and van Bussel, G. J. W. (2016). Aeroservoelastic design definition of a 20 MW common research wind turbine model. *Wind Energy*, 19: 2071-2087.

Åström, K. and Wittenmark, B. (1997). *Computer controlled systems*, 2nd edn, Prentice Hall International.

Bacic, M. (2005). On hardware-in-the-loop simulation. Proceedings of the *44th IEEE Conference on Decision and Control and European Control Conference 2005*, Seville, 3194-3198, 2005.

Bailey, M. and Doerr, J. (1996). Contributions of hardware-in-the-loop simulations to Navy test and evaluation. *Proceedings of the Society of Photo-optical Instrumentation Engineers*, 2741: 33-43.

Balla, J. (2011). "Dynamics of mounted automatic cannon on track vehicle". *International Journal of Mathematical Models and Methods in Applied Sciences*, 5: 423 – 432.

Basilios, R. and Gambier, A. (2020). Hardware-in-the-Loop simulation and control for developing very large wind energy systems. *IFAC-PapersOnLine*, 53: 12127 – 12132.

Bélanger, J., Venne, P. and Paquin, J. N. (2010). The what, where and why of real-time simulation. Proceedings of the *PES General Meeting*, 37-49, 2010.

Bossanyi, E. A. (2000). The design of closed loop controllers for wind turbines. *Wind Energy*, 3: 149 – 163.

Bossanyi, E. A. (2003). GH Bladed, Version 3.6 User Manual, Garrad Hassan & Partners Limited, Bristol.

Bottasso, C. L. and Croce, A. (2009). Cp-Lambda user manual, Dipartimento di Ingnegneria Aerospaziale, Politecnico di Milano, Milano.

Brayanov, N. and Stoynova, A. (2019). Review of hardware-in-the-loop - a hundred years progress in the pseudo-real testing. *Journal Electrotechnica and Electronica*, 54: 70 - 84.

Burns, A. and Wellings, A. (2009). *Real-time systems and programming languages*, Fourth Edition edn, Addison Wesley, Essex.

Burton, T., Jenkins, N., Sharpe, and Bossanyi, E. (2011). *Handbook of Wind Energy*, Wiley.

Evans, M. B. and Schilling, L. J. (1984). The role of simulation in the development and flight test of the HiMAT vehicle, NASA, NASA, Hampton.

Faruque, M. O. and Dinavahi, V. (2010). Hardware-in-the-loop simulation of power electronic systems using adaptive discretization. *IEEE Transactions on Industrial Electronics*, 57: 1146-1158.

Gambier, A. (2022). *Control of Large Wind Energy Systems*, Springer Nature, Basel, Switzerland.

Gambier, A. and Meng, F. (2019). Control system design for a 20 MW reference wind turbine, Hong Kong, 2019.

Garcia, L. W. (2017). Real-time operating systems. Case study: LynxOS vs. VxWorks, Florida Atlantic University, Boca Raton.

Hanselmann, H. (1996). Hardware-in-the-Loop simulation testing and its integration into a CACSD toolset. Proceedings of the *1996 IEEE International Symposium on Computer-Aided Control System Design*, Dearborn, USA, 152 – 156, 15–18 September 1996.

Heath, M. T. (1997). *Scientific Computing*, McGraw-Hill, NewYork, USA.

Howe, R. M. (1989). An improved numerical integration method for flight simulation. Proceedings of the *AIAA Flight Simulation Technologies Conference and Exhibit*, Washington D.C., 310-316, 1989.

Howe, R. M. (1991). A new family of real-time redictor-corrector integration algorithms. *Simulation*: 177-186.

Isermann, R., Schaffnit, J. and Sinsel, S. (1999). Hardware-in-the-loop simulation for the design and testing of engine-control systems. *Control Engineering Practice*, 7: 643-653.

Jonkman, J. M. and Buhl Jr., L. M. (2005). FAST User's Guide, NREL, Battelle.

Khaled-El Feki, A. B. (2014). Distributed real-time simulation of numerical models: application to power-train, Université de Grenoble, NNT: 2014GRENT033, Université de Grenoble, Grenoble.

Kiffmeier, U. (1996). A Hardware-in-the-Loop testbench for ABS controllers. Proceedings of the *on Control and Diagnostics in Automotive Applications*, Genova, 3-4 October 1996.

Kim, J. Y., Lee, Y. J., Cheon, S. W., Lee, J. S. and Kwon, K. C. (2010). A Commercial-off-the-shelf(COTS) dedication of a QNX real time operating system (RTOS), Mumbai, 2010.

Larsen, T. J. and Hansen, A. M. (2014). How 2 HAWC2, the user's manual, v.4.5, Risø, Roskilde.

Leisten, C., Jassmann, U., Balshüsemann, J., Hakenberg, M. and Abel, D. (2017). Design and analysis of a MPC-based mechanical hardware-in-the-loop system for full-scale wind turbine system test benches. *IFAC-PapersOnLine*, 50: 10985 – 10991.

Liu, J., Gao, X., Jiang, B., Yang, S. and Zhang, Z. (2017). Deterministic replay for multi-core VxWorks applications, Beijing, 2017.

Neshati, M., Zuga, A., Jersch, T. and Wenske, J. (2016). Hardware-in-the-loop drive train control for realistic emulation of rotor torque in a full-scale wind turbine nacelle test rig. Proceedings of the *2016 European Control Conference*, Aalborg, 1481-1486, 2016.

Pechlivanoglou, G., Marten, D., Nayeri, C. N. and Paschereit, C. O. (2010). Integration of a wind turbine blade design tool in xfoil/xflr5, Bremen, 2010.

Roscoe, A. J., Mackay, A., Burt, G. M. and McDonald, J. R. (2010). Architecture of a network-in-the-loop environment for characterizing AC power-system

behavior. *IEEE Transactions on Industrial Electronics*, 57: 1245-1253.

Sarhadi, P. and Yousefpour, S. (2015). State of the art: hardware in the loop modeling and simulation with its applications in design, development and implementation of system and control software. *International Journal on Dynamics and Control*, 3: 470-479.

Steurer, M., Li, H., Woodruff, S., Shi, K. and Zhang, D. (2004). Development of a unified design, test, and research platform for wind energy systems based on hardware-in-the-loop real-time simulation. Proceedings of the *IEEE 35th Power Electronics Specialists Conference*, Aachen, Germany, 3604 – 3608, 20–25 June 2004.

Takegaki, M. and Arimoto, S. (1981). A new feedback method for dynamic control of manipulators. *Journal of Dynamic Systems, Measurement and Control*, 103: 119 – 125.

Viehweider, A., Lauss, G. and Lehfuss, F. (2011). Stabilization of power hardware-in-the-loop simulations of electric energy systems. *Simulation Modelling Practice and Theory*, 19: 1699 - 1708.

Viehweider, A., Lehfuss, F. and Lauss, G. (2011). Power hardware-in the-loop simulations for distributed generation. Proceedings of the *International Conference on Electricity Distribution*, Frankfort, Germany, 1 - 4, 2011.

Visioli, A. (2006). *Practical PID Control*, Springer, London, UK.