Chapter 8

Processor-in-the-Loop Simulations Applied to the Design and Evaluation of a Satellite Attitude Control

Luiz S. Martins-Filho, Adrielle C. Santana,
Ricardo O. Duarte and Gilberto Arantes Junior

Additional information is available at the end of the chapter

http://dx.doi.org/10.5772/57219

1. Introduction

The design of a controller requires a mathematic modeling followed by the adjusting of some model parameters. However to overcome the controller to a single piece of hardware, involves the codification of this mathematical based model into an appropriate firmware description suited to work correctly in a specific platform. Recently a model driven development approach, firstly defined by system engineers, has been frequently used as way to reduce the time of development of embedded systems, producing rapid and reliable product in a short time development cycle. This model driven approach is basically used for test and is known as X-In-The-Loop. These tests provide four levels of testing configurations: MIL (Model-In-The-Loop), SIL (Software-In-The-Loop), PIL (Processor-In-The-Loop) and HIL (Hardware-In-The-Loop). Each of the configuration levels provides some advances and reduces the gap in the development process that initiate with the mathematical model and ends at the firmware running in a stand-alone microprocessor platform [1].

A model-driven system approach starts from the MIL configuration, where basic simulation is performed to analyze the controller model along with the simulated plant model. The basic idea behind MIL is to generate and validate some test cases of your model in a computing high precision float pointing arithmetic platform, providing the behavior and the quality of your model under a certain computer platform. The proposal of this step is to generate the reference output test results of your controller to the following steps. Any problem with the mathematical controller model can be rapidly found and corrected, taking the MIL step a first step of development.

In Software-In-The-Loop (SIL) the model used in the MIL test is replaced by an executable code running at the same computer platform in a fixed-point arithmetic manner. This step helps the system developer to find fast some wrong memory sizing choices. Normally these two steps are made using a single integrated platform in a PC. The SIL step can be bypassed if your controller system will run in a piece of hardware that has a floating point dedicated unit in its CPU datapath.

The following test consists in the PIL (Processor-In-The-Loop) that goes beyond the PC platform. This step introduces some hardware features that permit to achieve more realistic situations where the control algorithm will run. In PIL the target processor is a non-real time environment and the communication with the external processors is given by using specific functions installed in a simulation integrated environment installed in the host PC.

PIL requires drivers to communicate the computer platform with the aimed hardware. The resulting object code generated in the PC links with other test-management functionality and is then downloaded, typically to an off-the-shelf evaluation board with the target processor. The simulation tool, running on the PC machine, then communicates with the downloaded software, typically via a serial communication link.

The PIL simulation follows the simulation tool installed in the computer send test values to the firmware installed in the processor of the evaluation board, through a serial link and waits for the processor response through the same or another communication channel.

The software real time operation cannot be tested in the PIL, this step is done by the HIL test. Although at first sight, this can be seen as a limitation, in fact this permits to break the simulation problem in two parts that can be verified before we have the certainty that the controller firmware will run correctly in the standalone processor platform. PIL permits to test if the compile optimizations effects in a non-real time execution platform with the presence of an off-the-shelf processor platform where the firmware will finally run.

As the last step of an embedded controller system development HIL is presented. HIL simulation must include electrical emulation of sensors and actuators in a real time target platform before the controller be validated with real sensors and actuators of the plant. These electrical emulations act as the interface between the plant simulation and the embedded system under test all of them in the same platform. The value of each electrically emulated sensor is controlled by the plant simulation and is read by the embedded system under test bringing a real time feedback next to the real situation that the controller will face before to be installed to control a physical plant.

Nowadays there are some model-driven platforms that allows the engineer develop the above simulation step of the controller. Latelly, Mathworks© and National Instruments© are the most known platforms. This work concentrates in presenting the development steps of an attitude control model in the MATLAB – Simulink tool from Mathworks©.

2. The processor in the loop (PIL) and hardware in the loop (HIL) simulation approach

In the PIL application example described in section 3, we use MATLAB/Simulink environment both for a design procedure, code generation and for perform a PIL co-simulation together with a device. In the following paragraphs we describe some works that has some relationship with our work regarding the use of a development environment, code generation and co-simulation. The focus was in works with aerospace application where we presented their proposition, the hardware and tools used as well as the co-simulation scheme done.

In [2], simulations SIL and PIL were performed to obtain an attitude determination and control system (ADCS) for the microsatellite CKUTEX from Cheng Kung University. The SIL simulation is made using the MATLAB software and after, the PIL simulation is implemented using a PIC microcontroller to the implantation of the ADCS algorithm while the satellite dynamics is implemented in NI-PXI platform and coded by Labview software. According to the authors, the attitude determination and control system that was obtained and tested, provided good results once the plant dynamic response obeyed the project specifications. In [3], the MATLAB software is used with the purpose of generate the code of an entire control system and the dynamics of two satellites (represented by two robots). In this work a physical simulator using two industrial robots is assembled for a simulation of proximity operations between satellites as *on-orbit servicing* (OOS) activities. A model of the satellites dynamics, its control, actuators and sensors (constituting the named Application Control System - ACS) is made in MATLAB/Simulink by the tool named Real-Time Workshop (RTW). It generates a code supported by the operating system VxWorks that on the other hand, operate according this code, the monitoring and control system of the facility where the movement commands desired are sent to the robots. It is a HIL simulation where controllers, actuators, state observers, among other modeled modules in the MATLAB/Simulink can be removed of the ACS and included in their own hardware if necessary.

In [4], a true co-simulation approach is studied for control application running on a Field Programmable Gate Array processor (FPGA). The proposed approach adopts the LABVIEW Real-Time environment (from National Instruments©) to perform the simulation of an artificial satellite' dynamic model. This study simulates a reaction wheel and its control in Simulink exclusively for the controller design. Subsequently, the generated code is implemented on the FPGA. The entire system model attitude control is previously built and simulated in Simulink. The RTW generates the code that represents the satellite dynamics' mathematical model inside LABVIEW environment. In the case of the reaction wheel and its control, RTW generates the C code to simulate the dynamic model that must be adapted to the LABVIEW rules for FPGA programming.

In [5], the Simulink is used together with another MATLAB toolbox called xPC Target. This toolbox allows to perform prototyping, testing and development of real-time systems for running in general-purpose computers. The MATLAB/Simulink tools were used to generate executable code for the target computer from the model built in Simulink. The goal is to achieve simulation and real-time implementation of an algorithm integrating inertial navigation and GPS, using the validation and testing of the proposed algorithm in the FlightGear flight simulator (inside MATLAB), running on the host computer.

In order to exemplify the possibility of use of different development environment to perform SIL and PIL simulations of a satellite control system; in the work of [6], a SIL simulation is made in MATRIX$_x$ environment where is obtained and tested the control algorithm, simulated the space environment and the satellite dynamics. This same software is used to generate the controller code, that after is downloaded in the satellite processor while its model is ported to a DSP that is controlled by a computer. This computer runs the real-time simulation together with the satellite processor, controlling the actuators and sensors signals too, in a PIL simulation. An interface allows the user to interact with the simulation, monitoring and changing simulation parameters in real-time.

In the cited works, is observed as the simulations SIL, PIL and HIL can be useful in controllers project and test to aerospace applications and as this simulations can be implemented in different ways, different virtual development environments and together with several hardware platforms. The results are not only obtained much more quickly but come from tests very cheap that those made in a real platform, which is often not accessible to the researchers. Furthermore, the interaction with the user in real-time and the analysis instruments, available by co-simulation tools, make the tests much more dynamic and enables the analysis of the system response in situations more realistic.

Today there are many co-simulation tools and many are the possibilities of combination between these tools and the several processors hardware platforms. Such tools are in continuous development and making possible the coupling with several hardware platforms from different manufacturers, including additional functional facilities to become easier the task of development.

In these studies, the Simulink is used as a friendly graphical tool for systems development, design and generation of executable code for various devices and applications. The following section presents more details of a control system design procedure using the PIL approach, Simulink tools and a DSP processor.

3. Example of PIL simulation application

As an example of PIL simulation scheme, we present the application of this strategy on the attitude control problem of an artificial satellite. This case study consists of an embedded digital attitude control system (ACS) design for three axis stabilization [7]. The information obtained by the sensors is processed by a Digital Signal Processor (DSP). The controller design is based on Linear-quadratic Gaussian (LQG) optimal control approach, a well known control theory in terms of space technology applications.

3.1. The satellite attitude control problem

The mathematical model of the satellite attitude considers two reference systems: the orbital frame and the satellite body frame. The orbital frame moves jointly with the satellite. Its origin coincides with the satellite's center of the mass. The axis z_0 of the orbital frame is defined by the satellite radius vector, the axis y_0 by the orbital normal, and the axis x_0 completes a right handed coordinate frame.

The body frame (x, y, z) has its origin in the center of mass of the satellite. Their axes coincide with the satellite's principal axes of inertia. We consider the case where the nominal attitude has axes of the body frame aligned with the axes of the orbital frame [8]. The Euler angles are adopted as parameters of attitude, considering the sequence of rotations 3-2-1. In this case, the rotation matrix is given by [9].

$$R_x^X = \begin{bmatrix} \cos\psi\cos\theta & \cos\psi\cos\theta & -\sin\theta \\ -\cos\phi\sin\psi + \sin\phi\sin\theta\cos\psi & \cos\phi\cos\psi + \sin\phi\sin\theta\sin\psi & \sin\phi\cos\theta \\ \sin\phi\sin\psi + \cos\phi\sin\theta\cos\psi & -\sin\phi\cos\psi + \cos\phi\sin\theta\sin\psi & \cos\phi\cos\theta \end{bmatrix} \tag{1}$$

The kinematics equation is obtained by the time derivative of the rotation matrix equation [8], then:

$$\dot{R}(t) = S(\omega(t))R(t) \tag{2}$$

The kinematics equation can be simplified if one considers small angle maneuvers, in this case we can approximate: $\sin\phi \approx \phi$, $\sin\theta \approx \theta$, $\sin\psi \approx \psi$, $\cos\phi \approx 1$, $\cos\theta \approx 1$ and $\cos\psi \approx 1$. Moreover, the non-linear terms $(\psi\theta, \psi\phi, \theta\phi)$ are very small compared to the linear ones. The velocities are also small compared to the orbital velocity [10]. Considering those approximations the kinematic equation is given by:

$$\omega_{ib}^b = \begin{bmatrix} \dot{\phi} \\ \theta \\ \dot{\psi} \end{bmatrix} + \omega_0 \begin{bmatrix} -\psi \\ -1 \\ 0 \end{bmatrix} \tag{3}$$

In this work the satellite is modeled as a rigid body. Therefore the dynamic model can be obtained with the Euler equation that describes the rotation of a rigid body [9, 11]. The dynamic equation is given by:

$$J\dot{\omega}_{ib}^b + S(\omega_{ib}^b)J\omega_{ib}^b = \tau_d^b + \tau_p^b \tag{4}$$

where J is the inertia matrix of the satellite, τ_d^b represents the torques from external perturbations acting on the satellite, and τ_p^b represent the control torques (τ_x, τ_y, τ_z), all vectors described in the body frame, ω_{ib}^{b} is the angular velocity of the body with respect to the inertial frame written in the body frame. The gravity gradient is the only external perturbation considered in this work. Its effect cannot be neglected in a low-orbit satellite; an asymmetric body subject to the Earth gravitational field will experience a torque tending to align the axis of minor inertia with the field direction. The gravity gradient torque is modeled as:

$$\tau_g^b = 3\omega_0^2 \begin{bmatrix} (J_z - J_y)\phi \\ (J_x - J_z)\theta \\ 0 \end{bmatrix} \tag{5}$$

Substituting the applied torques and the kinematic equation (Eqs. 5 and 3) into Eq. 4, and representing this dynamics in the state space form, we have [10]:

$$\begin{aligned} \dot{x} &= Ax + Bu \\ y &= Cx + Du \end{aligned} \tag{6}$$

3.1.1. The LQG control approach

The Linear Quadratic Regulator is designed upon the linearization of the dynamic model. The theory is developed for linear systems. However, the system simulation takes into consideration the complete non-linear model of the satellite. This optimal control approach consists of minimizing a quadratic cost function and computing a feedback gain matrix [12]. The optimization problem aims to obtain a control law expressed by a linear relationship between the state variable x (expressing here the Euler's angles and the angular velocities) and the control variable u (the applied torques), i.e. $u = -Kx$. The matrix of gain K is obtained by the minimization of a quadratic cost function formulated as follows:

$$J_{lqr} = \int_0^T \left(x^T Q_c x + u^T R_c u \right) dt \tag{7}$$

where Q_c and R_c are the weight matrices of state and control vectors, respectively. The value of K results from solving the algebraic matrix Riccati equation for a time-invariant system and considering an infinite horizon context:

$$A^T P + PA - PBR_c^{-1} B^T P + Q_c = 0 \tag{8}$$

where A and B are the matrices of the linearized attitude dynamic model. The optimal control gain is then given in terms of the solution P of the Riccati equation:

$$K = R^{-1} B^T P \tag{9}$$

Due to the presence of noises in sensors measurements and uncertainties in models, a filter is needed to obtain more reliable information about the states. For the inclusion of signals filtering, we adopted the controller design based on the theory of Linear Quadratic Gaussian control [12]. We consider in this work the presence of white noise in the observations, and that the system is observable. In the LQG problem, we want to minimize the cost function:

$$J_{lqg} = \int_0^T \left(x^T Q_f x + u^T R_f u \right) dt \tag{10}$$

where Q_f is the covariance matrix of the measurements noise, and R_f is the covariance matrix of the dynamics noise (or model uncertainties). The LQG control law determination will be given by solving two problems: setting the controller for the linear quadratic deterministic problem, and setting a Kalman-Bucy filter for optimum estimation $\hat{x}$ of state x. The formulation of the Kalman-Bucy filter is given by:

$$\hat{\dot{x}} = A\hat{x} + Bu + L\ (y - C\hat{x}) \tag{11}$$

where L is the Kalman filter gain that is obtained by solving the algebraic Riccati matrix equation:

$$A^T S + SA - SCR_f^{-1} C^T S + Q_f = 0 \tag{12}$$

The gain of the optimal filter is given by $L\ = R_f^{-1} C^T S$. After obtaining L, it is possible to obtain the transfer function of open loop LQG controller according to [10]:

$$K_{lqg} G(s) = K(sI - A + BK + LC)^{-1} LG(s) \tag{13}$$

where $G(s) = C(s)(sI - A)^{-1} B$ is the transfer function of the attitude dynamics.

The linear quadratic controller action can be implemented using actuators such as reaction wheels and magnetic actuators for a continuous control command. However, during orbital operations, such as rapid detumbling maneuvers, the required torqueses are usually too high for reaction wheels. Therefore, on-off propulsion strategies are used for such operations [9]. The choice of cold gas jets actuation in the present study aims to test the control in a most critical situation in terms of the small attitude adjustments difficulties.

The firing of the gas jets is controlled by a pulse-width pulse-frequency modulator (PWPF) [13]. The PWPF is an interesting option for the thrusters control system due to its advantages over other types of pulse modulators. PWPF is designed to provide propulsion output proportional to the input command. The modulator optimizes the use of propellants; it provides a smoother control and increases the equipment life. The PWPF structure is shown in Fig. 1 [10].

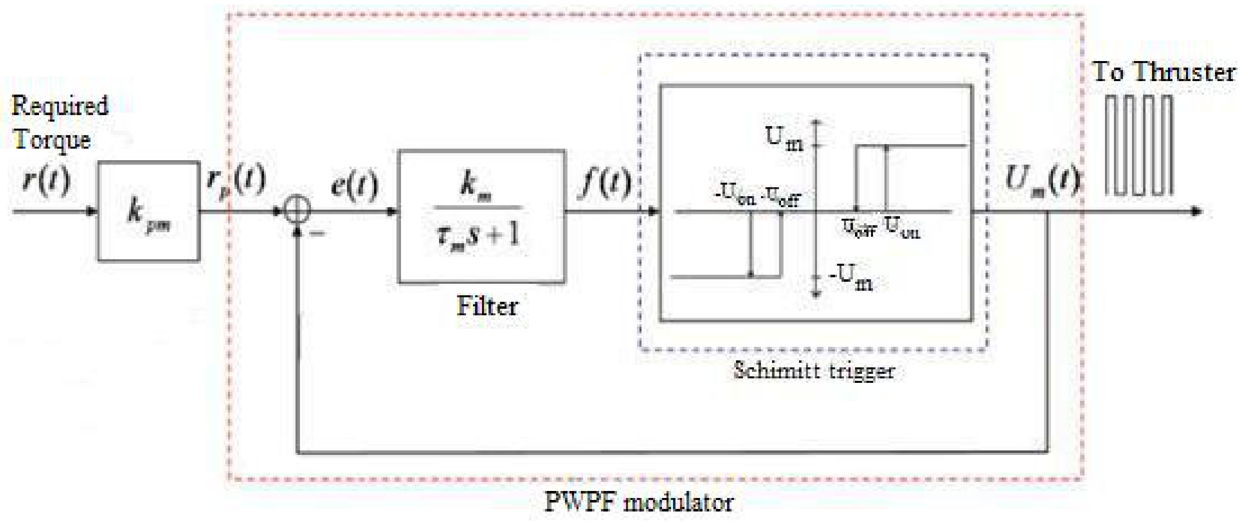


Figure 1. Scheme of the PWPF modulator.

When the positive input in the Schmitt trigger is greater than U_{on}, the trigger output is U_m. If the input falls below U_{off}, the trigger output becomes null. This response is also reflected for negative inputs. The error signal *e(t)* is the difference between the output of Schmitt trigger U_{on} and system input *r(t)*. This error is sent to a pre-filter whose input *f(t)* feeds the Schmitt trigger [10].

3.1.2. The digital LQG controller

The LQG controller, originally designed for continuous time systems, must be adapted to discrete time application. The appropriate selection of the sampling period *T* is a crucial factor in digital controller design, since if this period is too large there are problems in the signal reconstruction, and if it is too small, system instability and processing capacity problems can occur. In principle, one can believed that smaller sampling period is the best digital approximation. However, if the sampling period is too small, the controller poles approach the unit, causing instability. According to the equation mapping from *s* and *z* spaces, where, we can see that if *T* is very small, tending to zero, the poles of the controller in *z* tends to 1, which are the poles of a marginally unstable system, making the closed-loop system unstable. However, it is not necessary that *T* approaches to zero to start the problems, if it is small enough the poles at *z* cannot be anymore distinguished by the computer unit. Furthermore, small sampling periods can introduce significant distortions in the system dynamics behavior [14].

Very large sampling periods may result in violation of the rule established by the sampling theorem, which says that the sampling frequency ω_s must be twice greater than the highest signal component frequency ω_M [15]. If the condition of the theorem is not satisfied, there are information losses in the signal reconstruction. A reasonable choice for the sampling rate is 10 to 30 times the bandwidth of the system ω_B in closed loop [15]. A suggestion of [16] is to adopt a sampling frequency greater than 20 ω_B, in order to have a fairly smooth control response. In [17], the indication is to adopt the sampling period $T = 1/(10 f_B)$, where $f_B = \omega_B / 2\pi$ and f_B is the bandwidth of the closed loop continuous system. System dynamics frequencies' analysis and recursive tuning adjustments led to the value of 10ms as a quite satisfactory value, considering the desired pointing accuracy. Furthermore, since the maximum speed of the adopted DSP core is 600MHz, a sampling period of 10ms allows the processing of the received signal and computing the control signal in the co-simulation in the interval between samples.

The design of control systems in the continuous domain is mathematically simpler and allows the use of a large set of tools. In the case of control system design in discrete domain, the mathematical problem is quite more complicated. In addition, in the continuous time domain, the visualization of the relationship between physical reality and mathematical representation of a control system is more evident. Therefore, the usual starting point for a discrete time control design is the continuous time control system study, followed by discretization procedures.

There are several methods of discretization of a given continuous time system, and they are basically divided into open loop methods and closed loop methods. In the open-loop methods the discretization essentially consists of turning the transfer function *G(s)* in *G(z)*. This transformation is performed by substituting the terms in *s* by *z* terms in order to satisfy some criterion. In the closed loop methods, the discretization of a controller function *G(s)* is obtained taking into account information from the operation of the closed-loop system, and also the knowledge of all the transfer functions involved in the system, including the plant that is is intrinsically continuous.

Several methods of discretization have been proposed and evaluated [18]. We adopted the open loop method of transformation of Tustin, also called bilinear transformation, giving satisfactory results even when compared to closed loop methods (which often have better results by taking into account the whole system), when applied to systems of low order. This method is based on the approximation of the integral represented by the factor *1/s*. The integral can be approximated by trapezoidal integration method in order to obtain:

$$s \approx \frac{2z-1}{Tz+1} \tag{14}$$

The dynamics of the satellite plant in the case of discrete control remains the same as in the continuous time case. Figure 2 illustrates the LQG controller scheme after its discretization. The main change observed in this scheme is the addition of a zero-order holder in the output signal which is sent to the satellite's block, as indicated by the output named "torque". The sensors signals are received by the "measured states" gateway. Finally, the controller subsystem was changed by the discretization in terms of its integration function.

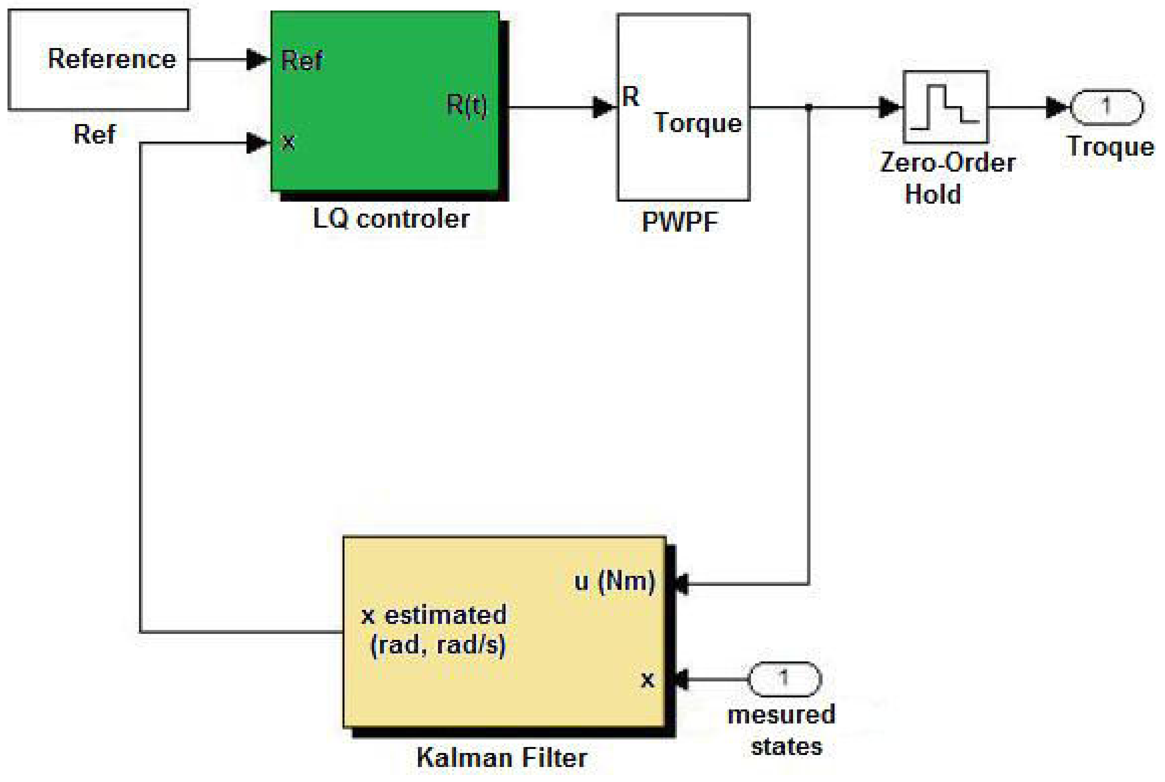


Figure 2. Scheme of the discret LQG controller.

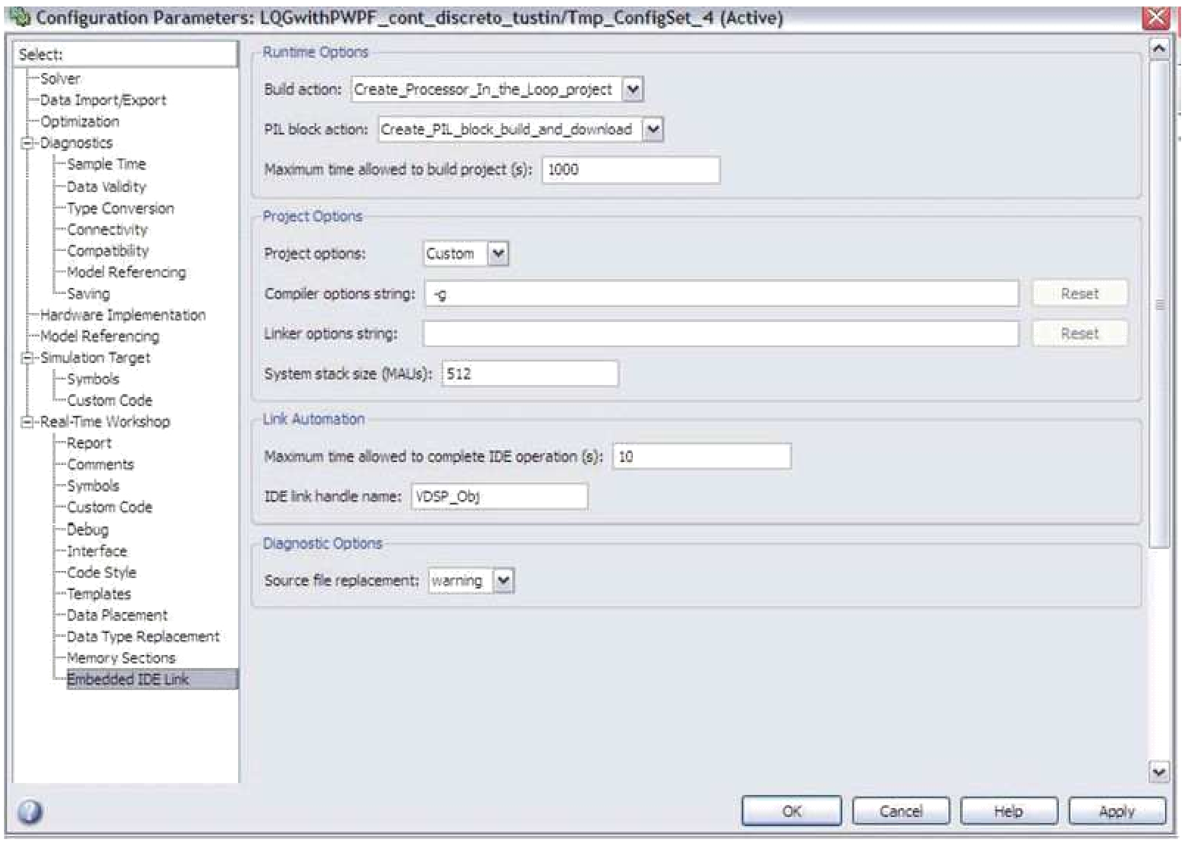
3.2. Numerical simulations

The validation of the digital attitude controller was carried through a scheme of co-simulation, where a computer performs the simulation of the satellite's motion, in MATLAB/Simulink, using models of attitude kinematics and dynamics, and a Blackfin 537 DSP device (installed on ADSP-BF537 kit, Analog Devices) plays the digital LQG controller processing, and also the PWPF modulator [7].

The interface with the computer uses the Real-Time Workshop tool, through the Embedded IDE link, as shown in the Fig. 3.

The validation tests comprise two distinct scenarios. In the first one, the tests consider the same case studied in the precedent sections, i.e. the three axes attitude stabilization. In the second scenario, the attitude control is aimed to perform a maneuver to achieve a new orientation, i.e. a task of attitude tracking.

The co-simulation scheme is based on the computer communication with the DSP (cf. Fig. 4), which is facilitated by Simulink tool, which allows integration of several external processors, including the BF537, by creating a block *Processor-In-the-Loop* (PIL) in the simulation diagram. This communication is made possible through the interaction of the Simulink with the BF537 development environment, the Visual DSP ++ (Fig. 5).



Configuration Parameters: LQGwithPWPF_cont_discreto_tustin/Tmp_ConfigSet_4 (Active)

Select:
- Solver
- Data Import/Export
- Optimization
- Diagnostics
  - Sample Time
  - Data Validity
  - Type Conversion
  - Connectivity
  - Compatibility
  - Model Referencing
  - Saving
- Hardware Implementation
- Model Referencing
- Simulation Target
  - Symbols
  - Custom Code
- Real-Time Workshop
  - Report
  - Comments
  - Symbols
  - Custom Code
  - Debug
  - Interface
  - Code Style
  - Templates
  - Data Placement
  - Data Type Replacement
  - Memory Sections
  - Embedded IDE Link

Runtime Options
Build action: Create_Processor_In_the_Loop_project
PIL block action: Create_PIL_block_build_and_download
Maximum time allowed to build project (s): 1000

Project Options
Project options: Custom
Compiler options string: -g
Linker options string:
System stack size (MAUs): 512

Link Automation
Maximum time allowed to complete IDE operation (s): 10
IDE link handle name: VDSP_Obj

Diagnostic Options
Source file replacement: warning

Figure 3. The PIL interface screen of the Real-Time Workshop (MATLAB/Simulink).

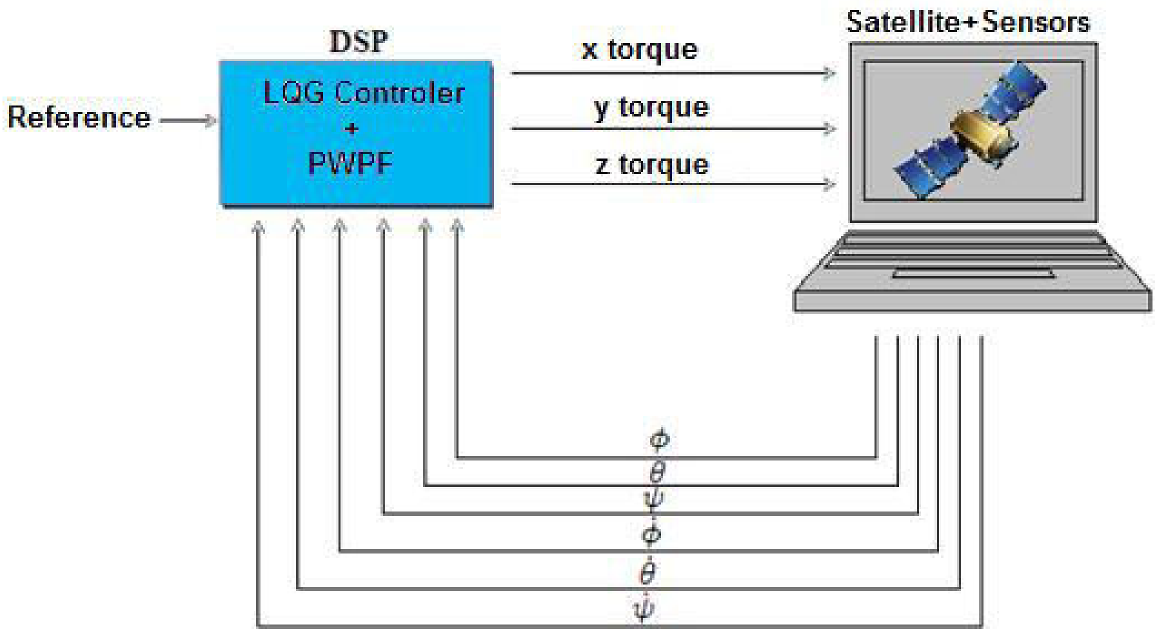
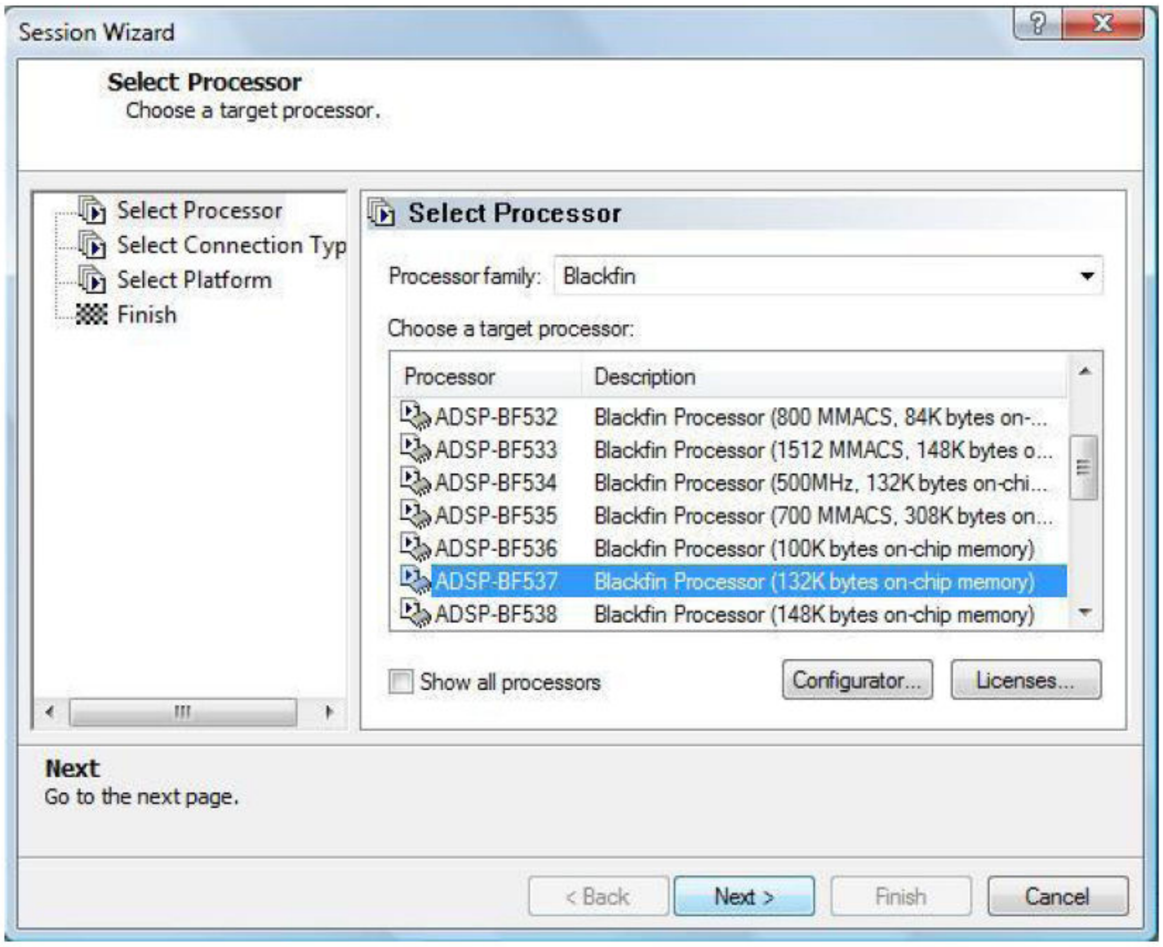


Figure 4. Co-simulation scheme: the DSP processes the LQG controller and the PWPF modulator, and computer simulates the motion of the satellite attitude.



Processor family: Blackfin

Choose a target processor:

Processor	Description
ADSP-BF532	Blackfin Processor (800 MMACS, 84K bytes on-...
ADSP-BF533	Blackfin Processor (1512 MMACS, 148K bytes o...
ADSP-BF534	Blackfin Processor (500MHz, 132K bytes on-chi...
ADSP-BF535	Blackfin Processor (700 MMACS, 308K bytes on...
ADSP-BF536	Blackfin Processor (100K bytes on-chip memory)
ADSP-BF537	Blackfin Processor (132K bytes on-chip memory)
ADSP-BF538	Blackfin Processor (148K bytes on-chip memory)

Figure 5. The session opening in Visual DSP++ environment.

The block PIL is inserted in the block diagram developed in the Simulink environment. It is responsible for the communication with the DSP, i.e. in charge of sending the information of the satellite attitude and of receiving the commands related to the control action, i.e. the gas jets driving from the PWPF modulator.

Figure 6 shows the results for the three Euler angles, of the validation tests for the case of three-axis stabilization scheme using co-simulation. The considered initial deviation is 10 degrees for each one of the three Euler angles, as in the case of the previous simulations.

A comparative analysis of the obtained results in the case of continuous LQG controller (in a simulation made only in MATLAB/ Simulink environment), and those obtained in the scheme of co-simulation (Fig. 6), can be made from a plot of the results differences. This differences plot is shown in Fig. 7.

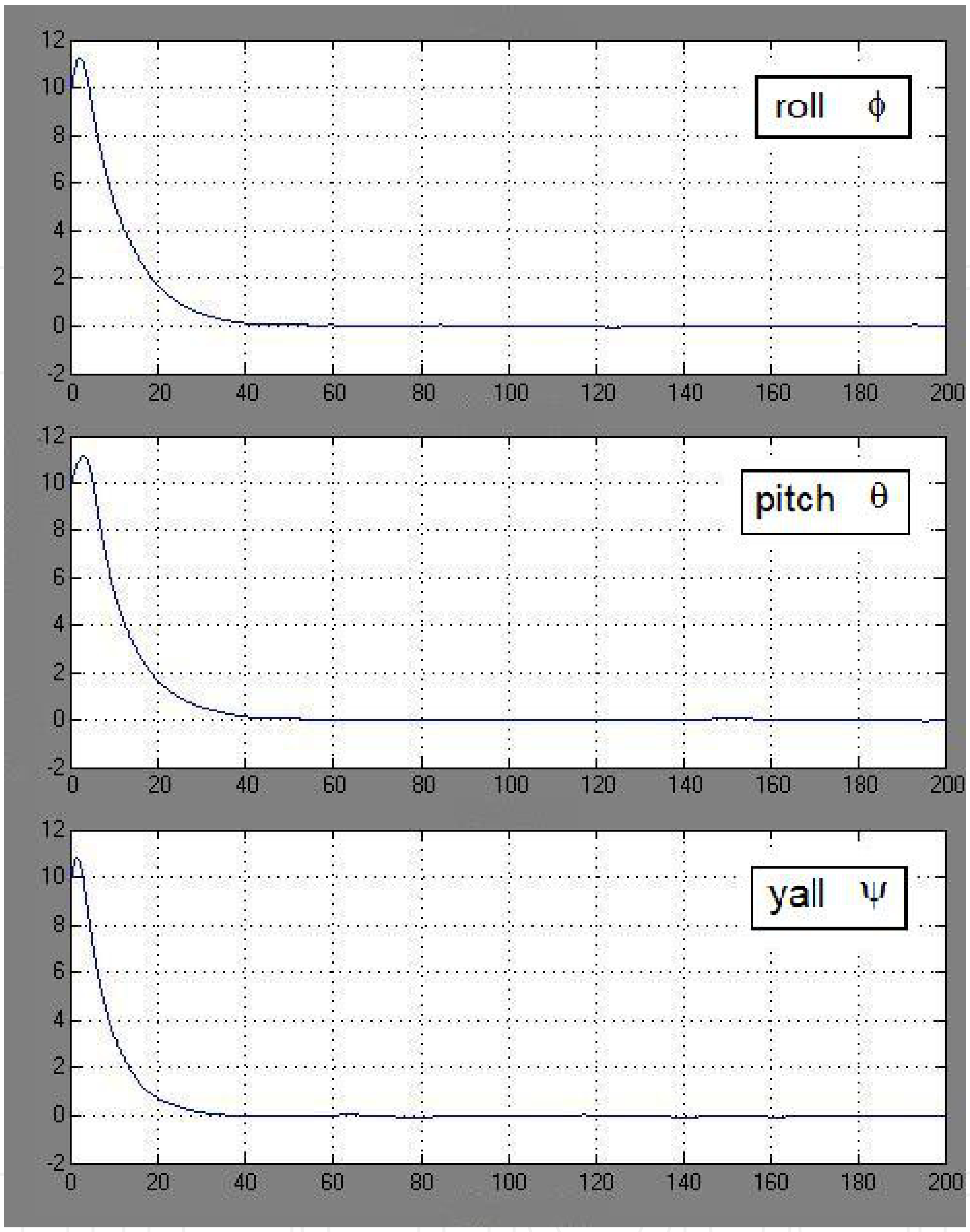


Figure 6. Co-simulation results of attitude control using a DSP (for ϕ, θ, ψ, in degrees).

The differences are smaller than 0.1 degree for the three angles. However, we cannot conclude about the better precision scenario. Both simulations met the accuracy specifications, nevertheless the simulation of continuous-time system lacks of realism for an experimental application. In fact, the small difference between the two cases shows only that the co-simulation scheme works very similarly to the idealized case, which may suggest an optimistic outcome in relation to the expectations of an experimental application.

In a second test case, the satellite is commanded to change its attitude, initially with three angles (roll, pitch, and yaw) at the same value of 10°, for a new attitude defined by the values

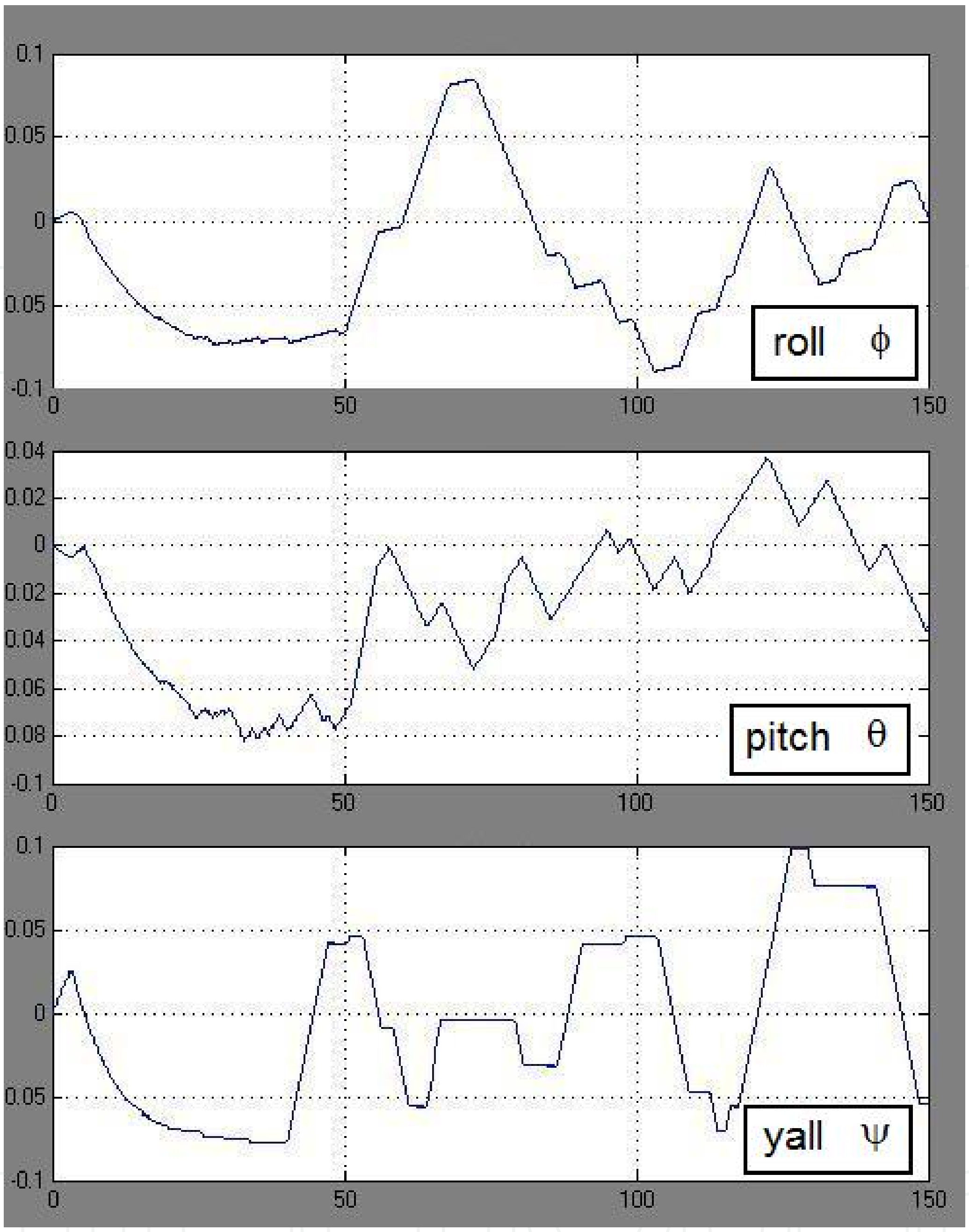


Figure 7. Difference between the simulation results of the continuous controller and co-simulation of digital control (for ϕ, θ, ψ, in degrees).

50, 60 and 45°, respectively. The results are shown in Fig. 8. It was observed that the controller performs satisfactorily its task, with tracking maneuver time of about 45 seconds. An important remark: the gap angles between initial and final attitudes are relatively large for the considered approximations in the synthesis of LQG control law. In the controller design, we adopted a linearized model for small angle values, whereas in the simulation of the motion of the satellite attitude we used nonlinear models of kinematics and dynamics. These models, in the case of large angles, exhibit very different behaviors of the linearized model, especially for higher angular velocities. It shows that the adopted controller achieves adequate performance even with this difference between models, and it presents interesting features of robustness.

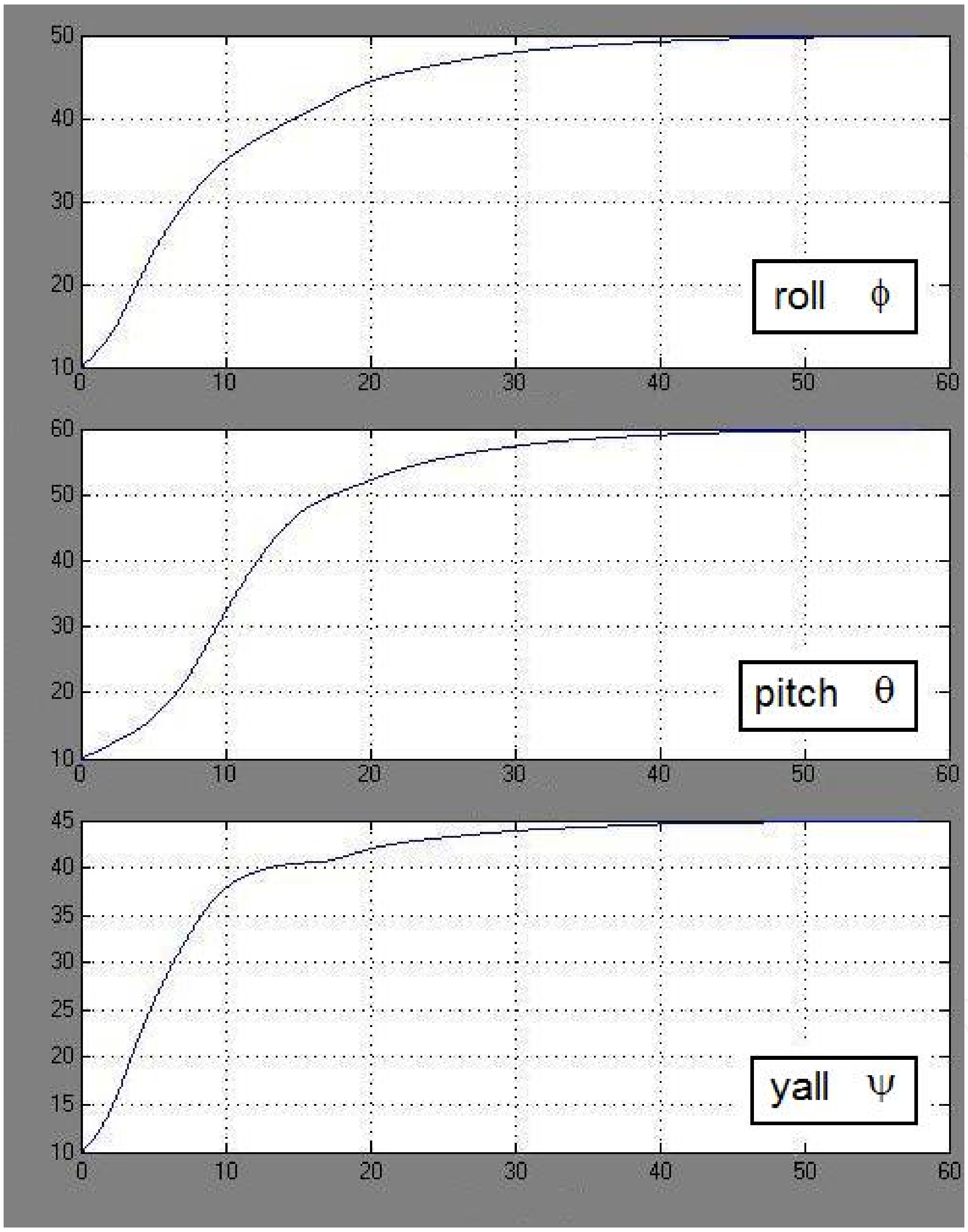


Figure 8. Co-simulation results of attitude control for the three Euler's angles, considering a tracking maneuver.

In terms of adopted sampling time, it was observed that the simulation in MATLAB/ Simulink waits the DSP processing to continue the calculations, preventing it from some problem related to the interval between two controller processing and actuation. However, there is a DSP development platform tool that allows the measurement of processing time and data traffic.

Consequently, it is possible to verify if the DSP processing time remains inferior to the maximum time period provided for sampling (10ms in this study). This tool is the *Cycle Counter*, which considers the frequency of the DSP core, 500MHz. The obtained result was processing and traffic time, much smaller than the time available due to the sampling period. It's possible to conclude that there are no problems in this application related to the aspect of the controller discretization and the use of a digital processor to perform the function of controlling and modulating the actuators.

4. Conclusion

The development of applications for embedded systems, as well the design of controllers to be performed by dedicate processors, can be immensely facilitated using the co-simulation approaches such as the Processor-In-The-Loop and Hardware-In-The-Loop. This validation scheme has been used as a way to move beyond on strictly computer simulations, opening the studies to realistic problems related to communication and exchange of data between embedded processor and controlled system, e.g. time delays, reliability of transmitted and received data, and processing time of the controller. These features are particularly useful for the design of artificial satellite embedded systems. That is the case of the example discussed here, a module function for the attitude control to be performed by an embedded digital processor.

Acknowledgements

The authors acknowledge the support of the Federal University of ABC - UFABC (Brazil), and of the *Coordenação de Aperfeiçoamento de Pessoal de Nível Superior* – CAPES / Program BJT – Science without Borders (Brazil).

Author details

Luiz S. Martins-Filho[1*], Adrielle C. Santana[2], Ricardo O. Duarte[3] and
Gilberto Arantes Junior[1]

*Address all correspondence to: luizsmf@gmail.com

1 Federal University of ABC, Brazil

2 Federal University of Ouro Preto, Brazil

3 Federal University of Minas Gerais, Brazil

References

[1] Shokry H, Hinchey M. Model-based verification of embedded software. Computer 2009; 2(4) 53-59.

[2] Juang JC, Chong CY, Tsai YF, Miau JJ, Tsai JR, Pan HP. Design, Implementation and Verification of Microsatellite Attitude Determination and Control Subsystem Based on Processor-in-the-loop. The 3rd Nano-Satellite Symposium: conference proceedings, Kitakyushu, Japan. 2011.

[3] Gaias G, D'Amico S, Ardeans JS, Boge T. Hardware-in-the-loop Multi-satellite Simulator for Proximity Operations. The 11th International Workshop on Simulation & EGSE facilities for Space Programmes: conference proceedings, Noordwijk, Netherlands, 2010.

[4] Seelaender G. Emulation and co-simulation of attitude control system for the PMM satellite and electro-hydraulic system for an aircraft using FPGAs. MSc. Thesis. Brazilian Institute for Space Researches, 2009.

[5] França Jr JA, Morgado JA. Real time implementation of a low-cost INS/GPS system using xPC Target. Journal of Aerospace Engineering, Sciences and Applications 2010, II(3) 29-38.

[6] Sabatini P, De Marchi E, Lupi T. Development and validation of attitude control systems using advanced off-the-shelf computer-based tools. Data Systems in Aerospace – DASIA: conference proceedings, Sevilla, Spain, 1997.

[7] Santana AC, Martins-Filho LS, Duarte RO, Arantes Jr G, Casella IRS. Satellite attitude control using a digital signal processor. Journal of Aerospace Technology and Management 2012, 4(1) 15-24.

[8] Wie HWB, Arapostathis A. Quaternion feedback regulator for spacecraft eigenaxis rotations. Journal of Guidance Control and Dynamics 1989 12(3) 375-380.

[9] Shuster, M.D. A survey of attitude representations. The Journal of the Astronautical Sciences 1993 41(4) 439-517.

[10] Arantes Jr G, Martins-Filho LS, Santana AC. Optimal on-off attitude control for the Brazilian multi-mission platform satellite. Mathematical Problems in Engineering V. 2009, ID 750945.

[11] Awrejcewicz J, Koruba Z. Classical Mechanics. Applied Mechanics and Mechatronics. New York: Springer, 2012.

[12] Dorato P, Abdallah CT, Cerone V. Linear Quadratic Control: an Introduction. New Jersey, USA: Prentice Hall, 1998.

[13] Buck NV. Minimum vibration maneuvers using input shaping and pulse-width, pulse-frequency modulated thruster control. MSc. Thesis. Monterey Naval Postgraduate School, 1996.

[14] Smith SW. The Scientist and Engineer's Guide to Digital Signal Processing. San Diego, California, USA: California Technical Publishing, 1997.

[15] Ǻstrom KJ, Witternmark B. Computer Controlled Systems: Theory and Design, New Jersey, USA: Prentice Hall, 1997.

[16] Franklin GF, Powell JD, Workman ML. Digital Control of Dynamic Systems, Manio Park, California, USA: Addison Wesley Longman, 1998.

[17] Dorf RC, Bishop RH. Modern Control Systems. Upper Saddle River, New Jersey, EUA: Pearson Prentice Hall, 2008.

[18] Soares PMOR. Discretization of continuous controllers. MSc Thesis. University of Porto, 1996.