Applied Mechanics and Materials Vol 783 (2015) pp 43-56
© (2015) Trans Tech Publications, Switzerland
doi:10.4028/www.scientific.net/AMM.783.43

Submitted: 2015-03-17
Accepted: 2015-04-13

An Hardware-In-the-Loop Tool for the Design of Complex Mechanical Systems Controllers

Manuel Beschi[1, a], Davide Colombo[2,b] , Paolo Grande[2,c], Fabrizio Padula[3,d] and Antonio Visioli[4,e]

[1]ITIA, National Research Council, Milan, Italy

[2]Gefran SpA, Gerenzano (VA), Italy

[3]Dipartimento di Ingegneria dell'Informazione, University of Brescia, Italy

[4]Dipartimento di Ingegneria Meccanica e Industriale, University of Brescia, Italy

[a]manuel.beschi@itia.cnr.it, [b]davide.colombo@gefran.com, [c]paolo.grande@gefran.com, [d]fabrizio.padula@unibs.it, [e]antonio.visioli@unibs.it

Keywords: Hardware-In-the-Loop, mechatronic systems, rapid control prototyping, industrial drives.

Abstract. In this paper we propose an Hardware-In-the-Loop (HIL) system for industrial drives. The system allows the user to optimize the design of control strategies by emulating complex mechanical applications. In fact, with this approach it is possible to simulate the behavior of the real mechanical system, and therefore to verify a priori the effectiveness of a control strategy and to achieve a rapid prototyping of the mechatronic system. The structure of the system consists of a couple of brushless motor which are connected by a mechanical joint. In one drive the control functions related to a specific application are implemented while the other one is used to replicate the mechanical system model of the application. A modular approach has been selected in order to allow a rapid development of a given application. In particular, a library of components has been implemented both in Simulink and in an IEC61131-3 language.

Introduction

Hardware-In-the-Loop (HIL) systems have been widely employed in the last years in order to develop real-time embedded systems [1]. In particular, they allow the user to verify the correctness of the design of the control algorithms without having the final application directly available and, in addition, they allow the execution of a large number of tests without the danger of damaging the application. In fact, they are one of the major tool for the rapid control prototyping and the reduction of the time-to-market of a new product.

HIL techniques have been employed first in the aviation industry [2] but its use have then been spread also in other sectors such as automotive [3], power electronics systems [4] and industrial machines. In particular, their use in the design of robotic systems have been also proposed [5,6,7,8].

In general, it is recognized that HIL systems are fundamental tools for the development of mechatronic systems, where the integration of the mechanical, electronic and automatic control parts is a key factor for the quality of the final product [9,10,11,12].

In this paper we present an HIL system for the simulation of (possibly complex) mechanical systems. The main idea is provide a tool for the well-known V-model approach for the design of engineering systems [13,14], which implies that the following steps are followed: analysis of the system and definition of the requirements and architecture, detail design, implementation, integration, test, verification and validation.

In particular, the devised HIL system allows the designer to perform various tasks, in particular:

- analyze the dynamics of the mechanical system and optimize the control system (by also easily perform trial-and-error procedures);
- validate the control software;
- validate the safety system;

- perform simulations of different scenarios in order to validate the control system in all the possible operating conditions.

The setup consists of two brushless motors, each one with its own controller unit, connected by shaft. One of the motor is actually the motor under test and it is used to actuate the system, while the other one represents the physical simulator according to the dynamic model of the mechanical system. Design choices of the HIL system will be discussed in the next sections, as well the solution of practical issues. Illustrative results are then presented to demonstrate the effectiveness of the system.

System Architecture

The system architecture consists of standard hardware and software components. As mentioned in the previous section, two brushless motors (Gefran SBM series, whose characteristics are shown in Table 1) are rigidly connected by shaft (see Fig. 1). Each motor has its own power inverter and controller unit (Gefran ADV200), with 3 kW and 5.5 kW of nominal power for the motor under test (MUT) and the simulation motor (SM) respectively. Then, to measure the position and the velocity of the motors, the MUT and the SM are equipped with a high-resolution EnDat and SinCos encoders respectively. The control algorithm is written in structured text, supported by the IEC 61131-3 standard. Finally, the human-machine interface is implemented in LabView on a standard PC (see Fig. 2). The communication between the drives and the PC is performed via Modbus. Control loops run at a 1 [kHz] frequency.
It is worth stressing that, in general, the size of the motors that are employed is not very relevant, as the mechanical system to be simulated can be scaled. The only important issue is that the size of nominal power of the SM is greater than that of the MUT (for example, a SM where the nominal power is about twice that of the MUT can be considered a good rule of thumb). Actually, using standard industrial hardware implies a reduction of the overall costs which can be relevant in many industrial fields.

Table 1. Characteristics of the motor used in the HIL system

	MUT	SM
Momentum of inertia	$2.75 \cdot 10^{-4}$ [kgm^2]	$11.6 \cdot 10^{-4}$ [kgm^2]
Maximum torque	15 [Nm]	38 [Nm]
Continuous stall torque	5.4 [Nm]	15.3 [Nm]
Nominal velocity	3000 [rpm]	3000 [rpm]

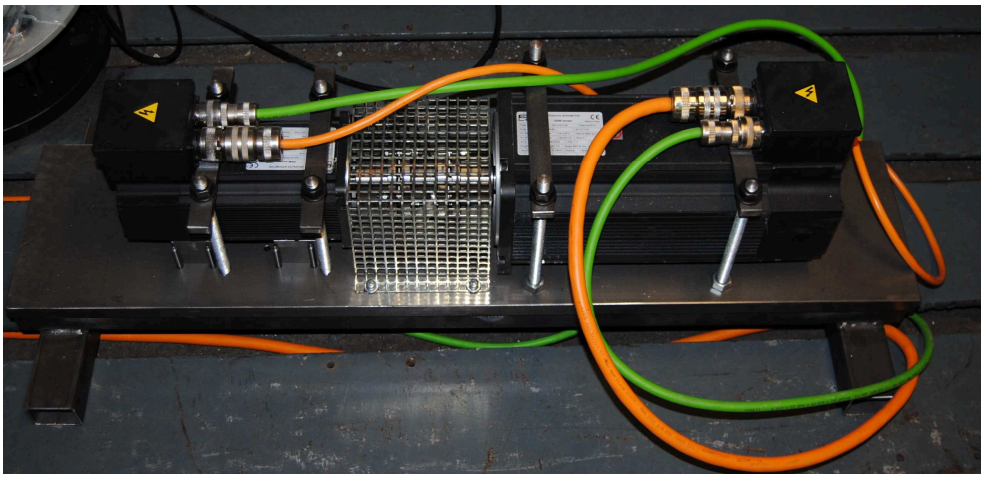

Figure 1. A picture of the two motors of the HIL system.

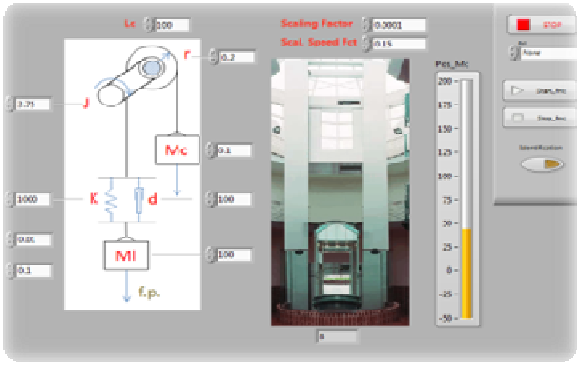

Figure 2. A screenshot of the human-machine interface.

Modelling and Identification

The HIL system is a dynamic system that can be modeled by means of the following equations:

$$\begin{cases} J_t \dot{\omega}(t) = T_m(t) + T_s(t) - T_{ms}(\omega(t)) \\ \dot{\phi}(t) = \omega(t) \end{cases} \tag{1}$$

where ϕ and ω are, respectively, the position [rad] and velocity [rad/s] of the motor shaft, J_T is the overall moment of inertia [kgm^2] at the shaft (which includes the moments of inertia of the two motors and that of the joint), T_m and T_s are, respectively, the torques [Nm] of the MUT and of the SM and $T_{ms}(\omega)$ is the friction torque that acts on the shaft and depends on the velocity of the two motors which are both present.

In order to simulate the behavior of a given mechanical load as accurately as possible, the SM must compensate its dynamics with the aim of transforming (1) into the following system of equations:

$$\begin{cases} J_t \dot{\omega}(t) = T_m(t) + T_c(t) - T_f(\omega(t)) \\ \dot{\phi}(t) = \omega(t) \end{cases} \tag{2}$$

where J_m is the moment of inertia [kgm^2] of the MUT, T_c is the resistive torque [Nm] produced by the mechanical system to be simulated and $T_f(\omega)$ is the friction torque that acts on the shaft when only the MUT is present. The value that T_s has to assume to obtain (2) from (1) results to be

$$T_s(t) = T_c(t) - T_m(t) + T_{ms}(\omega(t)) + (J_T + J_m)\dot{\omega}(t) \tag{3}$$

The block scheme of the overall system is represented in Fig. 3. It appears that the dynamic compensation requires the estimation of the motor acceleration which can be achieved by using a high-pass filter

$$F(s) = \frac{s}{\tau_f s + 1}$$

where τ_f is the filter time constant which can be conveniently selected in order to avoid to influence the dynamics of the mechanical system. In this context it is worth noting that the transfer function from ω to T_m (note that T_c=0 as there is no simulated load) is

$$G(s) := \frac{\omega(s)}{T_m(s)} = \frac{1}{J_m s} \frac{\tau_f s + 1}{\frac{J_t}{J_m}\tau_f s + 1} \tag{4}$$

where it appears that the pole and zero values depend on the filter time constant.
In order to obtain a correct compensation of the dynamics of the system, an accurate estimation of the parameters of the system (1) is necessary. For this reason, an automatic identification procedure can be employed for the estimation $\hat{T}_{ms}(\omega)$ of the friction function $T_{ms}(\omega)$. It consists in making the motor shaft rotating at different constant velocities. A practical solution is to make the SM rotating at a constant speed while no torque is applied to the MUT. Then, the values of the torque that is necessary to keep each set-point velocity is determined and the friction function can be obtained by interpolating the results (see Fig. 4). In practice, the friction function can be implemented as a look-up table in the function block that computes the overall dynamic compensation. Once the friction term has been estimated, the moment of inertia J_T can be easily obtained by applying a constant torque to the system and by subtracting the friction term. In particular, a negative maximum torque $-T_{max}$ can be applied to the SM until the maximum negative speed $-\omega_{max}$ is attained. Then, a maximum torque T_{max} is applied (note that $T_s(t) = T_{\max} + T_{ms}(\omega(t))$ as $T_m(t)=0$) and the time interval t_J until the maximum speed ω_{max} is attained is computed. The moment of inertia can be therefore estimated as

$$\hat{J}_t = \frac{T_{\max} \cdot t_J}{2\omega_{\max}} \tag{5}$$

The experimental results related to the identification method are shown in Fig. 5. The moment of inertia results to be $\hat{J}_t = 29 \cdot 10^{-4}$ [kgm^2].

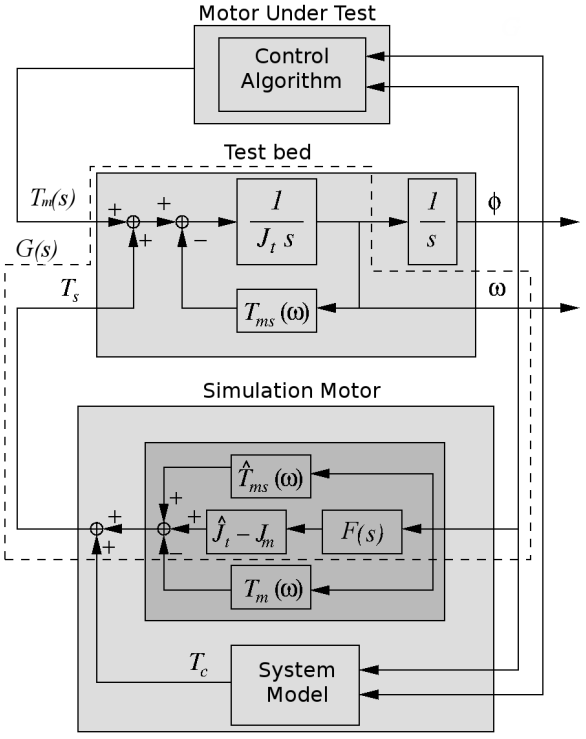

Figure 3. The block scheme of the HIL system.

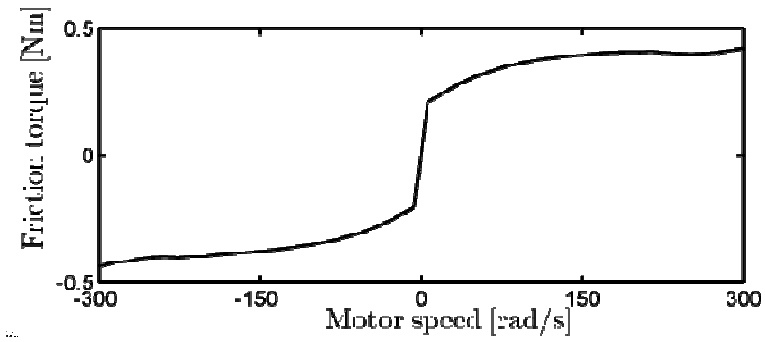

Figure 4. The estimated friction torque.

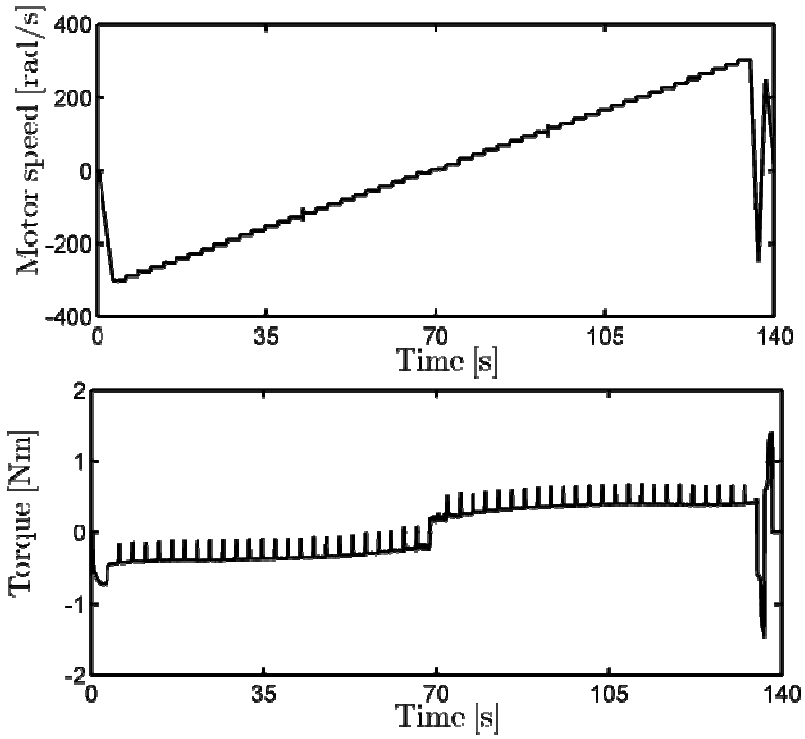

Figure 5. The experiment for the friction and inertia estimation.

Library of Mechanical Components

A library of simple mechanical components has been created so that they can be easily connected in order to built a complex mechanical system to be simulated. Each component has been implemented both directly in the ST language and in Simulink. In the latter case it can be employed to simulate the system via software and, most of all, it can be used to obtain the ST code by means of the Simulink PLC Coder toolbox. Indeed, the presence of such a library facilitates porting the software to other platforms.

A description of the various blocks is provided hereafter (obviously, other blocks can be implemented depending on the applications to be simulated). In order to facilitate the user in building the (possibly complex) system to be simulated, each block has a standard interface (see Fig. 6), that is, it requires, as inputs, the position A_u and the velocity ω_u of the component ahead of the block and the sum of the torques (or forces) T_d that act on the systems and that are caused by the elements behind the block. As outputs, each block has the position A_d and the velocity ω_d of the simulated component and the sum of the torques (or forces) T_u that the component applies to the elements above the block.

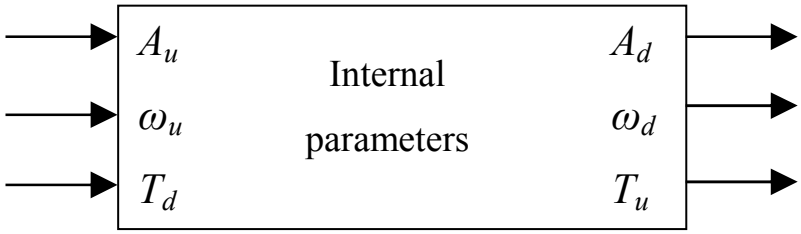

Figure 6. The inputs and outputs configurations of a block of the library.

Joint and Axle. The *JointAndAxle* block consists of an inertial element connected to the system by means of an elastic transmission, which is characterized by the presence of an elasticity and a damping that can be constant or variable. The component is depicted in Fig. 7 and is represented by the following equations:

$$\begin{cases} \dot{A}_d(t) = \omega_d(t) \\ \dot{\omega}_d(t) = \dfrac{1}{J}\left[\begin{array}{l} k\left(A_u(t) - A_d(t)\right) + h\left(\omega_u(t) - \omega_d(t)\right) + c\left|\omega_u(t) - \omega_d(t)\right| \\ + k_g A_d(t) - h_g \omega_d(t) - c_g \left|\omega_d(t)\right| + T_d(t) \end{array}\right] \\ T_u(t) = -\left(k\left(A_u(t) - A_d(t)\right) + h\left(\omega_u(t) - \omega_d(t)\right) + c\left|\omega_u(t) - \omega_d(t)\right|\right) \end{cases} \tag{6}$$

where A_d and ω_d are, respectively, the position and the velocity of the simulated body, J is the inertia (or the mass) of the system, A_u and ω_u are, respectively, the position and the velocity of the element ahead of the considered one. Then, k, h and c are, respectively, the constants that model the elasticity, the viscous friction and the sliding friction that can be present in the joint between the simulated body and the system ahead of it. For the sake of generality, also the constants k_g, h_g and c_g have also been introduced in order to model the elasticity, the viscous friction and the sliding friction term that can be present between the simulated body and the reference system. Finally, as already mentioned, T_d is the sum of the torques that act on the systems and that are caused by the elements behind the component while T_u is the sum of the torques (or forces) that the component exerts to the elements above the block.
It is worth stressing that the *JointAndAxle* block is capable to model a lumped parameter system. However, placing two or more of them in a cascade configuration gives to the user the chance to satisfactorily represent also a distributed parameter system.

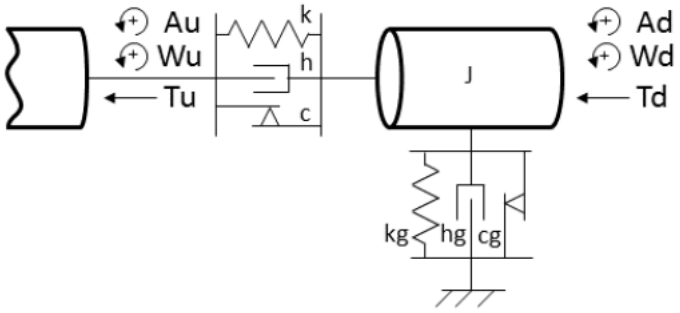

Figure 7. Model of the *JointAndAxle* component.

Pulley. The *Pulley* block allows the user to convert a rotational motion (generated by a motor) into the linear motion of the ropes and it plays a key role in the simulation of hoisting systems. The pulley can be connected directly to a motor or not. The component is shown in Fig. 8 and is represented by the following equations:

$$\begin{cases} P_l(t) = A_u(t)r \\ P_c(t) = L_f(t) - A_u(t)r \\ v_l(t) = \omega_u(t)r \\ v_c(t) = -\omega_u(t)r \\ T_u(t) = r\left(F_l(t) - F_c(t)\right) \end{cases} \quad (7)$$

where A_u and ω_u are, respectively, the position and the velocity of the input shaft of the pulley (which are the same of the possibly connected electrical motor), P_l, P_c, ω_l, ω_c are the positions and the velocities of the ropes (note that the pedices *l* and *c* are, respectively, for *load* and *counterweight*, thinking about a hoisting system), r is the radius of the pulley, L_f is the length of the rope (in an elevator, it is the distance between the cabin and the counterweight), T_u is the torque exerted by the pulley to the ahead block and F_l and F_c are the forces acting on the rope and exerted by the two masses.
By considering the equations (7) it appears that the block does not implement any dynamics but it just provides a kinematic relation between the input and the output.

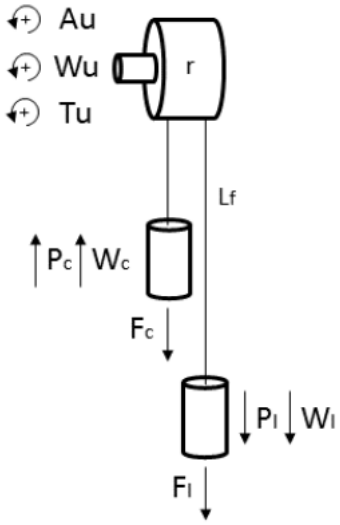

Figure 8. Model of the *Pulley* component.

Transmission. The *Transmission* block basically implements the functions of a speed reducer (see Fig. 9), by taking into account also its efficiency in direct and retrograde motion. Thus, the following equations are implemented:

$$\begin{cases} A_d(t) = A_u(t)\tau \\ \omega_d(t) = \omega_u(t)\tau \end{cases} \quad (8)$$

where τ is the speed ratio (note that the block can be easily used also to model a speed multiplier).

Then,

$$\begin{cases} T_d(t)\omega_d(t) = \eta T_u(t)\omega_u(t) \\ T_d(t) = \eta \dfrac{T_u(t)}{\tau} \end{cases} \quad (9)$$

where η is the mechanical efficiency, that is, $\eta=\eta_d$ in case of direct power flow or $\eta=1/\eta_r$ in case of reverse power flow.

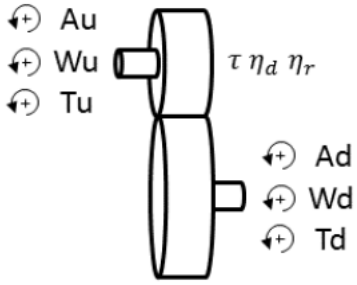

Figure 9. Model of the *Transmission* component.

Motor. The *Motor* block implements an electrical motor. Actually, this component does not need to be simulated in the HIL system, but it has been in any case implemented in order for a software-in-the-loop simulation to be available. As it is the first element of a kinematic chain, it does not need of inputs from ahead blocks and therefore its layout is different from the general one shown in Fig. 6. The inputs of the block are therefore the reference torque C_{m_ref} and the resistant torque C_{res} dependent on the blocks behind the motor one. Then, the momentum of inertia of the motor J_m and the friction terms have also to be taken into account. That is, the static friction term c and the viscous friction term F_{visc} have to be defined. In order to avoid algebraic loops, a difference equation has to be implemented and in this context it is necessary to consider a sampling period t_s and the motor position and velocity (A_{d_prec} and ω_{d_prec}, respectively). The output of the block consists of the actual position A_d and velocity ω_d. The (Matlab) code that implements the overall function is as follows:

```
function[Ad,Wd]=MotorFunction(Jm,c,ts,Cm_ref,C_res,Ad_prec,...
                              Wd_prec,F_visc)
  Wd=Wd_prec+(Cm_ref+C_res-c*sign(Wd_prec)-F_visc)*(ts/Jm);
  if (Wd*Wd_prec)<=0
     C_att=Cm_ref+C_res+Wd_prec*Jm/ts;
     if abs(C_att)<=c
        Wd=0;
     end
  end
  Wd_prec=Wd;
  Ad=Ad_prec+ts*(Wd_prec);
  Ad_prec=Ad;
```

Note that the structure of the function is quite simple. The value of the actual velocity is computed based on its previous value and on the other internal parameters and variables of the motor. The new value of the torque is then computed. If this value does not exceed the static friction value, then the velocity is set to zero. Finally, the value of the position is determined and assigned to the output. It is worth stressing that the accurate estimation of the friction and inertia parameters of the motor is essential to correctly implement the block.

Brake. The *Brake* block allows the designer to implement the braking functionality in a mechanical system so that the motion of a body can be slowed down or stopped. The model of a brake basically consists of two shaft connected with an elastic link with a constant or variable stiffness and damping (see Fig. 10). Basically, when the brake is enabled, a torque with a sign opposite to the actual one is generated. This torque T_b has a maximum value chosen as the maximum value of the torque of the moving body (for example, a motor). The equations that describe the systems are

$$\begin{cases} \dot{A}_d(t) = \omega_d(t) \\ \dot{\omega}_d(t) = \dfrac{1}{J_b}\left[k_b\left(A_u(t) - A_d(t)\right) + h_b\left(\omega_u(t) - \omega_d(t)\right) + c_b\left|\omega_u(t) - \omega_d(t)\right| + T_d(t) + T_b(t)\right] \\ T_u(t) = -\left(k_b\left(A_u(t) - A_d(t)\right) + h_b\left(\omega_u(t) - \omega_d(t)\right) + c_b\left|\omega_u(t) - \omega_d(t)\right|\right) \end{cases} \quad (10)$$

where J_b is the momentum of inertia (or the mass) of the brake and k_b, h_b, and c_b, are internal parameters of the brake, that is, its elastic constant, viscous friction coefficient and sliding friction coefficient, respectively. In order to avoid chattering problems when the velocity is close to zero, the velocity at the next sampling time is estimated and, based on this value and the actual one (namely, based on their sign), the output torque is evaluated. The procedure is implemented by means of the following (Matlab) code:

```
function T_out = Brake_Function(T_in,T_brake,Wd,Enable,ts,Jb)
  T_out=T_in;
  if Enable
     if Wd > 0
          T_out=T_in-T_brake;
     elseif Wd < 0
          T_out=T_in+T_brake;
     end
    Wd_next=Wd+(T_out*ts)/Jb;
    if Wd*Wd_next <= 0
       if T_in > T_brake
            T_out=T_in-T_brake;
       elseif T_in < -T_brake
            T_out=T_in+T_brake;
       else
            T_out=-Wd*ts/Jb;
       end
    end
  else
     T_out=T_in;
  end
```

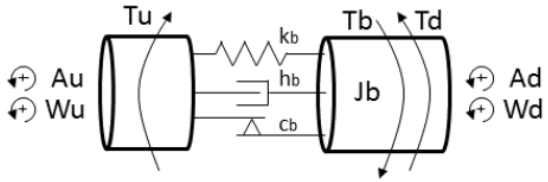

Figure 10. Model of the *Brake* component.

Validation

The HIL system have been tested and validated by using a Fourier analysis, in order to understand its capability to simulate a real system. Indeed, an input signal consisting of the sum of five sinusoids have been applied to the system (that is, it has been applied as torque signal to the MUT) and the results have been compared with those obtained with a purely Matlab simulation. In particular, different *JointAndAxle* systems have been implemented in the SM, that is, different second-order systems with different values of undamped natural frequency ω_n and of damping factor ξ. In fact, with the addition of a *JointAndAxle* component implemented in the SM, the transfer function between the torque of MUT and its velocity is

$$G'(s) := \frac{\omega(s)}{T_m(s)} = \frac{1}{Js^2 + hs + k} = \frac{\mu\omega_n^2}{s^2 + 2\xi\omega_n s + \omega_n^2} \tag{11}$$

where it appears that μ=1/J, ω_n=$\sqrt{K/J}$, ξ=h/(2ω_n) and J, K, and h are the *JointAndAxle* parameters (the others are set to zero). Thus, the following input torque signal has been applied:

$$T_m(s) = 10\sin(0.1\omega_n t) + 2\sin(0.5\omega_n t) + \sin(\omega_n t) + 0.5\sin(2\omega_n t) + 0.1\sin(10\omega_n t)$$

and the Fourier analysis has been performed on the measured (MUT) velocity signal and on the signal resulting from a simple software simulation of the system (11). As illustrative examples, results related to the case ξ=0.5 and ω_n=10 and ω_n=100 respectively are shown in Fig. 11-12 and in Fig. 13-14 where both the time domain and the frequency domain have been considered. It appears that the Fourier coefficients are similar until the frequency (100 [Hz]) become closer to the control frequency, where a distortion is expected. In this way, a clear idea of the working area of the HIL system has been obtained. Actually, it appears that most of the typical mechanical systems that are of interest for testing are in the applicability range.

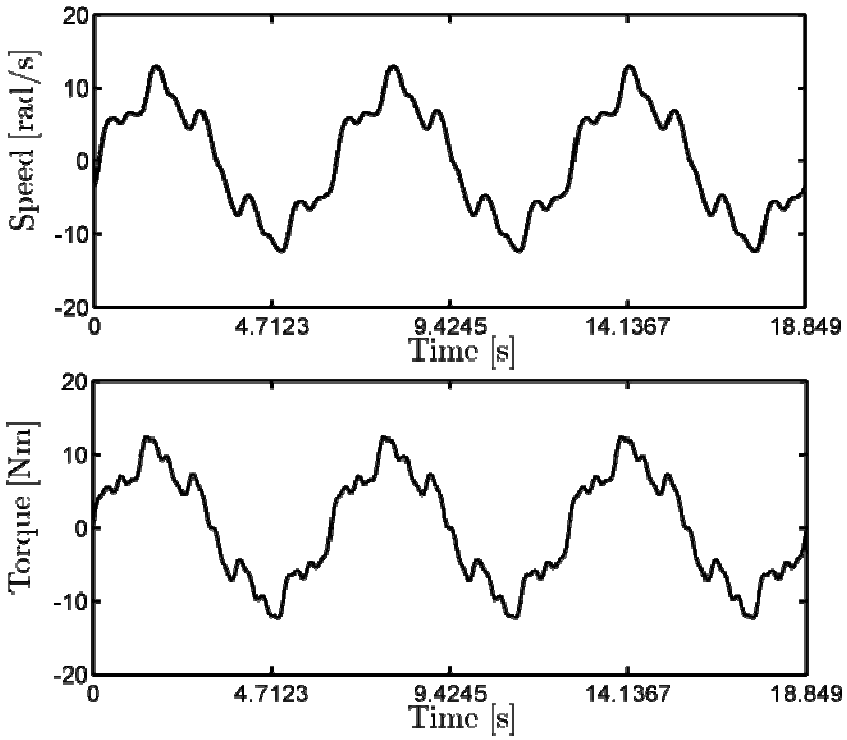

Figure 11. Results in the time domain of the Fourier analysis with ω_n=10 [rad/s]. Solid line: response of the HIL system. Dashed line: response obtained with Simulink.

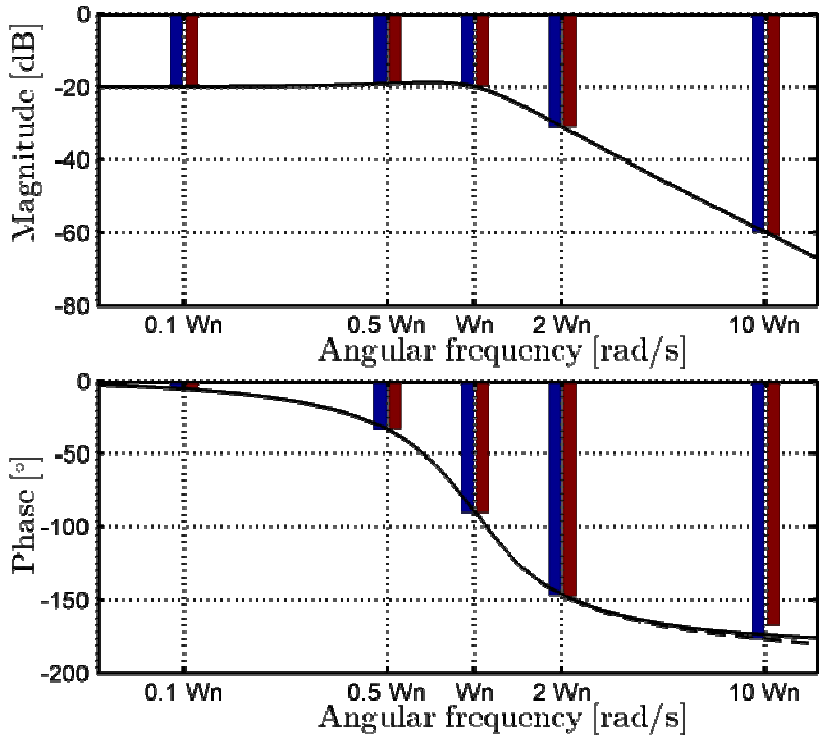

Figure 12. Results in the frequency domain of the Fourier analysis with ω_n=10 [rad/s]. Blue line: response of the HIL system. Red line: response obtained with Simulink. Solid line: Bode plot of the continuous time system (11). Dashed line: Bode plot of the discretized system.

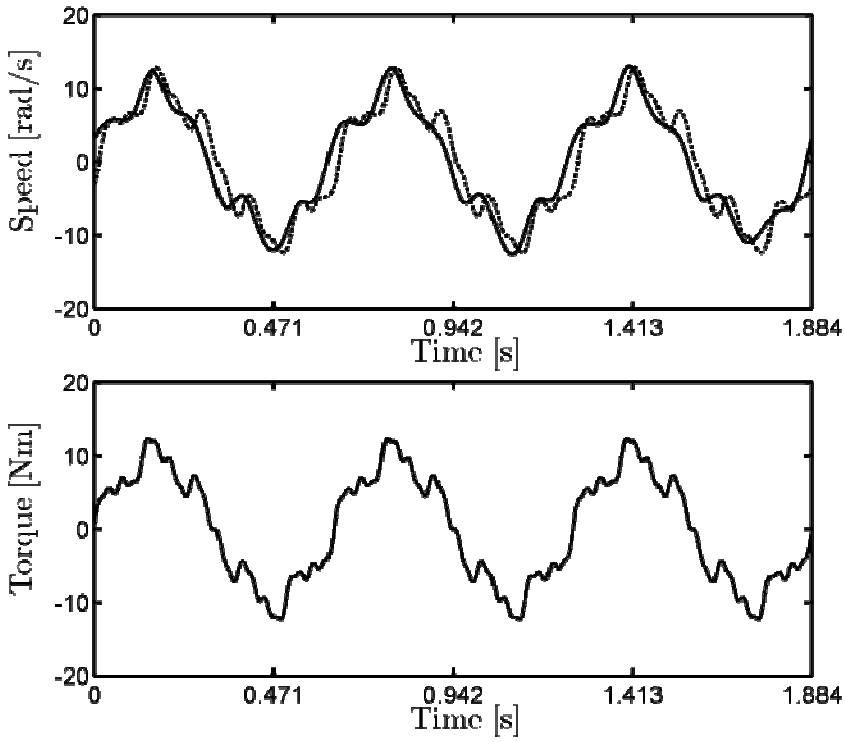

Figure 13. Results in the time domain of the Fourier analysis with ω_n=100 [rad/s]. Solid line: response of the HIL system. Dashed line: response obtained with Simulink.

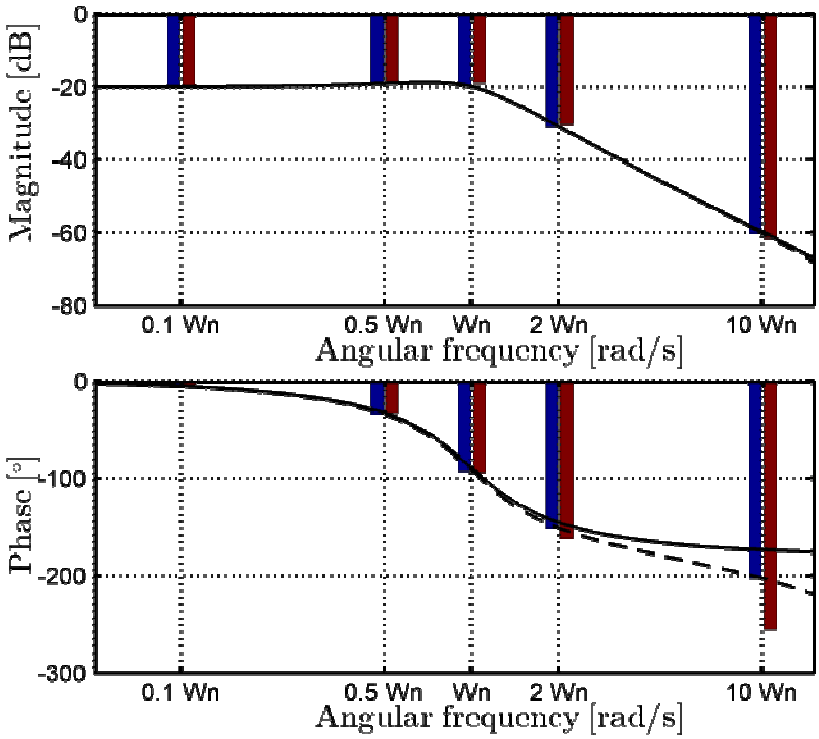

Figure 14. Results in the frequency domain of the Fourier analysis with ω_n=100 [rad/s]. Blue line: response of the HIL system. Red line: response obtained with Simulink. Solid line: Bode plot of the continuous time system (11). Dashed line: Bode plot of the discretized system.

Illustrative example: simulation of an elevator

In order to illustrate the use of the devised HIL system, an elevator system for a high skyscraper has been considered. The system is sketched in Fig. 15 and it has been conveniently scaled in order to be in the ranges of torque and velocity of the two motors. For example, the actual mass of the cabin is 500 [kg] but it has been reduced to 0.1 [kg]. The other parameters have the following values. The moment of inertia of the motor and of the pulley (which are supposed to be rigidly connected) is equal to $3.75 \cdot 10^{-4}$ [kgm^2], the height of the elevator (that is, the length of the rope) is 100 [m], the radius of the pulley is 0.03 [m], the mass of the counterweight is 0.1 [kg], the elastic constant of a 100 [m] rope is 1 [Nm/rad], its damping coefficient is 0.01 [Nms/rad] and, finally, the connection between the cabin (or the counterweight) and the guide rails has an elastic coefficient equal to 0 [Nm/rad] and a damping coefficient of 0.01 [Nms/rad]. Note that the last two values are expressed by the coefficients h_g and c_g of a *JointAndAxle* component.

The response of the system is shown in Fig. 16 where the position of the cabin virtually coincides with its reference signal. It is interesting to note the MUT torque presents small variations when the velocity is close to zero. These are because of the compensation of the static frictions that works properly.

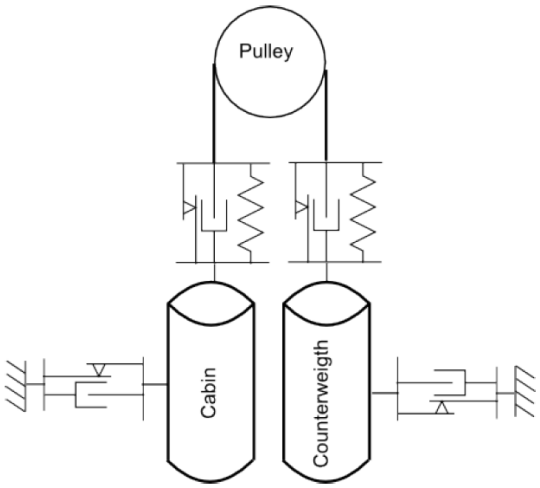

Figure 15. The elevator system used as an illustrative example.

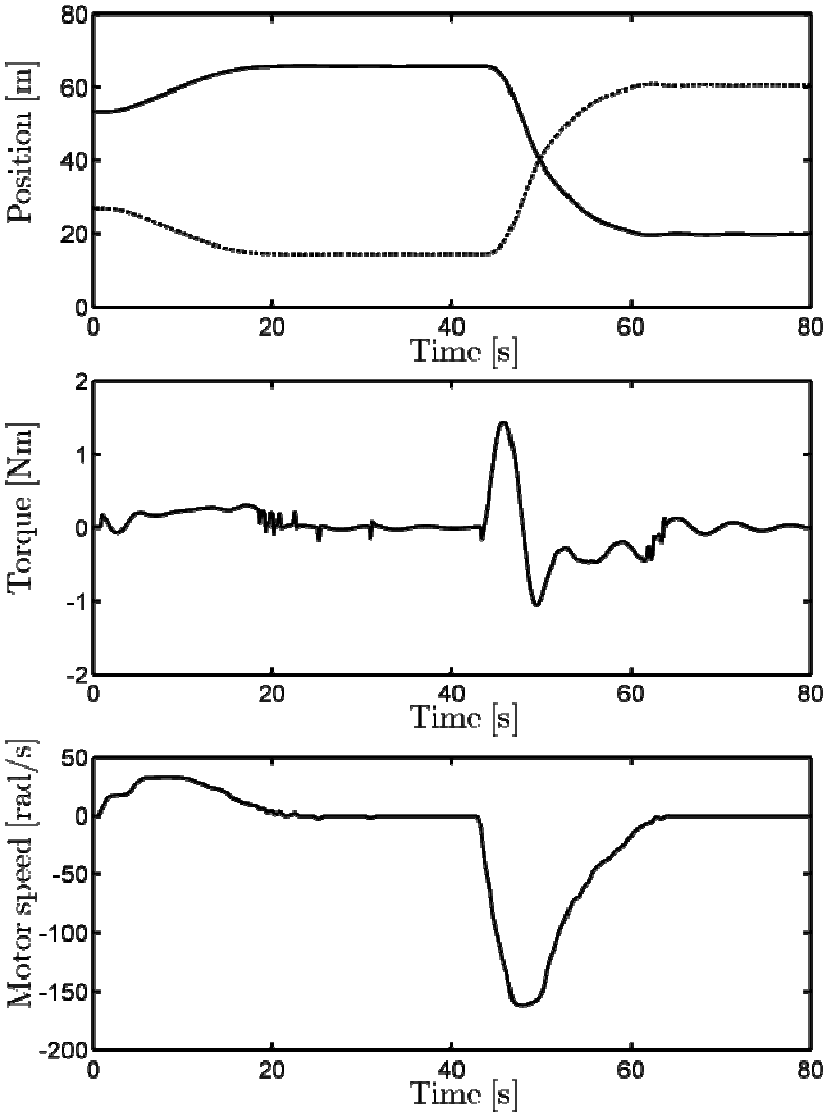

Figure 16. Response of HIL system with the elevator example. Top: position of the cabin (solid line) and of the counterweight (dashed line). Middle: torque of the MUT (including friction compensation). Bottom: velocity of MUT.

Conclusions

An HIL system for the simulation of mechatronic systems has been presented. The availability of a library of simple mechanical elements allow the designer to build a complex mechanical system in a relatively easy way. The tool is therefore suitable for the rapid control prototyping. In fact, the control system can be designed by taking into account the possible configurations of the system and the possible values of the parameters, thus allowing the simulation of different scenarios without possibly time-consuming, expensive and dangerous experiments on the real system.
Future work will include the creation of a library of components also for hydraulic systems, in order to simulate machines consisting both of mechanical and hydraulic components such as plastic injection molding machines.

References

[1] J.A. Ledin: Embedded Systems Programming (1999), p. 42

[2] A. Bittar and N.M. Franco de Oliveira: International Conference on Unmanned Aircraft Systems (2013), p. 134.

[3] T. Brendecke and F. Kucukay: International Journal of Vehicle Design Vol. 28 (2012), p. 84

[4] B. Lu, X. Wu, H. Figueroa and A. Monti: IEEE Transactions on Industrial Electronics Vol. 54 (2007), p. 919.

[5] H. Temeltas, M. Gokasan and O.S. Bogosyan: Proceedings of the 27th Annual Conference of the IEEE Industrial Electronics Society (2001), p. 357

[6] A. Martin and M.R. Emani: Proceedings of the 2006 IEEE International Conference on Mechatronics and Automation (2006), p. 2162

[7] A. Martin and M.R. Emani, in: Robot Manipulators, edited by M. Ceccarelli, chapter, 19, InTech (2008).

[8] R. Chhabra and M.R. Emani: Mechatronics Vol. 23 (2013), p. 335

[9] R.H. Bishop: *Mechatronics: an Introduction* (CRC Press, USA 2005).

[10] D. Shetty, R.A. Kolk, J. Kondo and C. Campana: Proceedings of the 2001 IEEE International Conference on Advanced Intelligent Mechatronics (2006), p. 1005

[11] T. Le, F.-M. Renner and M. Glesner: Proceedings of the 8th IEEE International Workshop on Rapid Systems Prototyping (1997), p. 116

[12] L. Racchetti and C. Fantuzzi: Proceedings of the 18th IEEE Conference on Emerging Technologies and Factory Automation (2013)

[13] D. Michalek, C. Gehsat, R. Trapp and T. Bertram: Proceedings of the IEEE/ASME International Conference on Advanced Intelligent Mechatronics (2005), p. 1065

[14] R. Isermann: *Mechatronics Systems: Fundamentals* (Springer-Verlag, UK 2005).