Nuno Miguel Fonseca Ferreira · José António Tenreiro Machado *Editors*

Mathematical Methods in Engineering

Springer

Nuno Miguel Fonseca Ferreira • José António
Tenreiro Machado
Editors

Mathematical Methods in Engineering

Springer

Editors
Nuno Miguel Fonseca Ferreira
Coimbra Institute of Engineering
Coimbra
Portugal

José António Tenreiro Machado
Dept of Electrical Engineering
Institute of Engineering of Porto
Porto
Portugal

ISBN 978-94-007-7182-6 ISBN 978-94-007-7183-3 (eBook)
DOI 10.1007/978-94-007-7183-3
Springer Dordrecht Heidelberg New York London

Library of Congress Control Number: 2014938246

Printed on acid-free paper

Springer is part of Springer Science+Business Media (www.springer.com)

Mathematical Methods in Engineering

Preface

This book addresses, in a single volume, some of the contributions that are carefully selected according to the reports of referees, presented at the International Symposium, MME10 Mathematical Methods in Engineering, held in Polytechnic Institute of Coimbra- Engineering Institute of Coimbra (IPC/ISEC), Portugal, October 21–24, 2010.

The Symposium provided a setting for discussing recent developments issues about theoretical and applied areas of mathematics and engineering. The conference was intended to be an international forum where an effective exchange of knowledge and experience amongst researchers active could take place.

The members of the organizing committee were Micael Couceiro and Nuno Ferreira.

We would like to thank all the referees and other colleagues who helped in preparing this book for publication. Our thanks are also due to all participants for their contributions to the symposium and to this book.

Our special thanks are due to Nathalie Jacobs and Cynthia Feenstra from Springer, for their continuous help and work in connection with this book.

Contents

Mathematical Modeling for Software-in-the-Loop Prototyping of Automated Manufacturing Systems

Claudio Bonivento, Matteo Cacciari, Andrea Paoli and Matteo Sartini

Abstract Nowadays automated manufacturing systems are designed as the complex interconnection of components belonging to different engineering domains. Actually high performances are required in order to satisfy market needs and standards. In this framework the validation via simulation plays a crucial role as it allows to verify the system during the design phase. Software-in-the-loop architectures represent a good practice to take into account also technological side-effects that represent a classical cause of long time-to-market or, in the worst case, to project failure. In this paper we present a mathematical simulator to be used within a software-in-the-loop prototyping system.

Keywords Multi-domain simulator · Mechatronics · Rapid prototyping · Validation by simulation

1 Introduction

Recently in most of industrial processes an ever increasing degree of automation has been observed. This is motivated by new request of systems with high performances in terms of quality of products and services, productivity, efficiency and low costs in the design, realization and maintenance. This trend in the growth of complex automation systems is rapidly spreading over Automated Manufacturing Systems (AMS). Nowadays automation is based on the integration between different areas: automatica and mathematical control theory, mechanics, electrical devices and electronics and

C. Bonivento (✉) · M. Cacciari · A. Paoli · M. Sartini
CASY—Center for Research on Complex Automated Systems, University of Bologna,
Viale Carlo Pepoli 3/2, Bologna 40136, Italy
e-mail: claudio.bonivento@unibo.it

M. Cacciari
e-mail: matteo.cacciari@unibo.it

A. Paoli
e-mail: andrea.paoli@unibo.it

M. Sartini
e-mail: matteo.sartini@unibo.it

N. M. Fonseca Ferreira, J. A. Tenreiro Machado (eds.), *Mathematical Methods in Engineering*, DOI 10.1007/978-94-007-7183-3_1,

computer engineering (see [1–3]), and this makes automation a tough task. Time-to-market is a crucial issue in developing any industrial product and, consequently, the request of reducing development time in realizing automated industrial plants is getting more and more tight. Toward this purpose, many tools have been developed to speed-up the testing phase of control logic in the development life-cycle. One way to improve development time is to develop hardware and software in parallel. Usually, this approach involves separate hardware and software professional teams development that perform their work simultaneously and independently. As soon as an hardware prototype and a substantial portion of the embedded code become available, such hardware and software are combined in a system integration phase and the testing task begins. Too frequently, serious problems arise during this system integration process and this typically causes significant hardware reconfiguration or software workarounds which worst the time to market. To develop software independently from hardware while avoiding integration problems an important key factor is the availability of simulation tools. In fact integration problems are due to side effects deriving from technological aspects which cannot be considered if software is developed independently from hardware. Besides this, the simulation is not only important to reduce software development time, but also to analyze system properties such as, fault diagnosis, fault reconfiguration and safety that cannot be tested on the field (see [4, 5]). From this brief discussion it turns out that a good simulation tool is such if is based on a properly detailed mathematical model of the system, and it is capable to capture all technological aspects linked to the field device (mechanical implementation, communications etc.).

Several software tools exist to model and simulate separately all different aspects. In particular we refer to physical domain simulator of the system dynamics, logic control CACSD tools, electronic and mechanics CAD. So, the integration problem still remain open. An approach to test the control software algorithm while considering technological aspects is the Software-In-the-Loop (SIL) simulation. In this approach the control unit interacts with a mathematical simulator of the plant evaluating both logic correctness and the side effects coming from the implementation (see [6–8]).

The aim of this work is to present a SIL technological architecture for rapid prototyping that embeds a multi-domain mathematical simulator implemented on a PC, a logic control running on a PLC and a communication infrastructure between the simulator and the controller.

This paper is organized as follows. In Sect. 2 we present a description of our proposed architecture and, in Sect. 3, the mathematical model of the system. In Sect. 4 we show the application to a micro FMS.

2 A SIL Technological Architecture for Rapid Prototyping

In order to present the needs and requirements for the proposed SIL architecture, we present a simple FMS that has motivated in this work. This system will be used as a testbed along the the work.

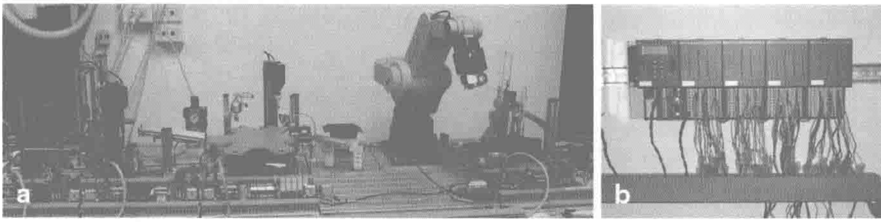

Fig. 1 Testbed hardware. **a** Flexible manufacturing system. **b** Control hardware

The testbed is a miniaturized flexible manufacturing system (FMS) produced by FESTO-DIDACTIC (see Fig. 1a); the plant is devote to produce short-stroke cylinders each of them composed by a basic body, a piston, a spring and a cap. In particular the system starts from raw pieces which are worked to realize the bodies and assemblies them with the other parts to obtain the desired cylinder. Thanks to the use of different basic bodies it is possible to realize different diameter cylinders. In the following cylinders' bodies will be referred as workpieces. The FMS is composed by four stations (see Fig. 1a): the first station is the distribution station, where the workpiece is picked from the raw materials warehouse and moved to the second station, the testing station. In testing station the workpiece is measured and its color and height is identified. According to this measurements the workpiece is discarded or moved to the processing station; in this station the workpiece is tested to verify if it can be worked or not. If the workpiece positively passes the test, it is drilled and then moved to the last station, the assembly station, where workpieces are assembled by a robotic manipulator to realize the cylinder. For a complete description of the system the reader is referred to [9]. The control of the FMS is implemented on a ABB PLC belonging to AC500 family equipped with CPU PM581-ETH with four input/output modules DC523 (see Fig. 1b)

The FESTO-FMS is equipped with sensors and actuators related with pneumatic and electric technology and is driven by control logic (event-driven) implemented on the real time computational architecture (PLC). The interaction of all these different fields makes the simulation of the system a complex task.

An important remark should be done on the communication between the control logic and the field; The state of the plant is read by the set of sensors and communicated to the PLC. On the basis of "picture" that freeze the state of the plant, the control logic compute the actual control action that is communicated to the set of actuators. This scenario must be reproduced also in the SIL architecture: the mathematical model should also simulate the logic sensor readings that will be send to the PLC in the form of a data vector. In the same way, the control action computed by the control logic (all boolean values) is sent back to the simulator that must accept it as input of the drivers of the simulated physical components. The communication is performed via ethernet using OPC communication. The main advantage of OPC is that is realize data exchange between different hardware nodes guaranteeing interoperability and flexibility (see [10–12]). In our architecture both the simulator and the control logic

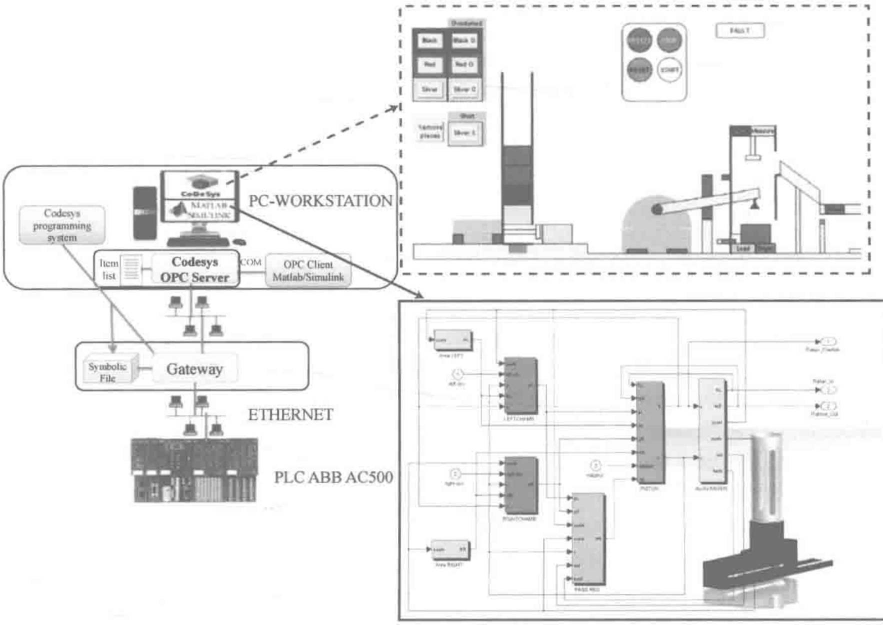

Fig. 2 SIL technological architecture

interface with an OPC client (the first is installed on the PC, the latter on the PLC), while an OPc server (installed in the PC) realize the communication between the two clients. More in details the OPC server manage symbolic files containing variables from time to time read or written by PC or PLC. At the same time is guarantied the data synchronization using time-stamps of data. it is important to stress at this stage the OPC do not realize a real time communication. This extension belongs to our ongoing research and can be realized physically (using a real time network) or virtually (managing time at both simulator and controller sides).

The task of the simulator implemented on the PC is to simulated components dynamics and prepare input data vector for PLC. We implemented the mathematical model of the components with Matlab/Simulink. Matlab/Simulink supports different toolboxes to help the designer to model systems, using a multi-domain approach. Using Matlab/Simulink we have an easy integration between different components and it is possible to build open model, changing in easy way the level of accuracy of the mathematical model. Moreover Matlab/Simulink embeds the OPC client that is used for communicated. The control logic has been developed with Codesys, a logic CACSD tool, the same that also run in the PLC. Codesys can also be used to design a logic simulator when all components are modeled as discrete integrators. Having Codesys also installed in the PC where the simulator runs (besides Matlab), it is possible, from time to time to decide which simulator should be used, changing the level of details of the simulator see (Fig. 2). Finally, since Codesys embed an OPC client, also the communication can be performed without the needs of any other tool.

3 Mathematical Modeling of Automation Systems Components

In this section we are going to show the mathematical models and within the simulator. For the sake of brevity here we present only the model of a pneumatic component. The intrinsic difficulty of modeling a system of this type is the strong interaction between mechanical and pneumatic physical domains. In a pneumatic system the potential energy generated by compressed fluid is converted to mechanical energy using valves and actuators. For this reason we can speak of multi-domain modeling.

Before seeing our implementation it is usefull to introduce some pneumatic basics. It is well known that, like others physical domains, a pneumatic circuit can be associated to an equivalent electric circuit (see [13]). The pressure (p) can be assimilated to a voltage and the mass flow rate (G), different from the volume flow rate because of air compressibility to the electric current. As in electrical circuits, also in pneumatic domain can be introduced the concepts of resistance (R), inductance (L_p) and capacitance (C_p) of a component.

Resistance is defined as the pressure derivative respect to mass flow. Analytically deducing this value is very difficult because is variable with the mass flow rate. Instead of using a complex formula like $G = f(\Delta p)/R(\Delta p)$, it is preferable to use two experimentally deducible constant parameters (sonic conductance C and critical pressure ratio b) that are characteristic of each pneumatic resistive component.

Exploiting these constants and assuming polytropic transformations, the value of G can be found as:

$$G = \rho_a \cdot Q = \rho_a \cdot C \cdot p_1 \cdot \sqrt{1 - \left(\frac{\frac{p_2}{p_1} - b}{1 - b}\right)^2} \cdot \sqrt{\frac{293}{T}} \qquad \textit{subsonic case} \tag{1}$$

$$G = \rho_a \cdot Q = \rho_a \cdot C \cdot p_1 \cdot \sqrt{\frac{293}{T}} \qquad \textit{sonic case} \tag{2}$$

where p_1 and p_2 are respectively the upstream and downstream pressure of any components, ρ_a is the air density and Q is the volume flow rate. According to the rate between downstream and upstream pressure ($b = p_2/p_1$) we can be in subsonic case ($b < \theta$) or in sonic case ($b \geq \theta$) where θ is a constant function of the gas specific heat ratio. In few words, due to the Venturi effect, when upstream pressure is larger than θ times the downstream pressure ($b = 0.5283$), the flow is independent from the downstream pressure (in this case we say that the flow is chocked or that the flow has sonic velocity).

Pneumatic capacitance is typical of components with non negligible volume and, assuming a perfect gas, it is linked with the pressure by the equation:

$$\frac{dp}{dt} = \frac{G_{in} - G_{out}}{C_p} \tag{3}$$

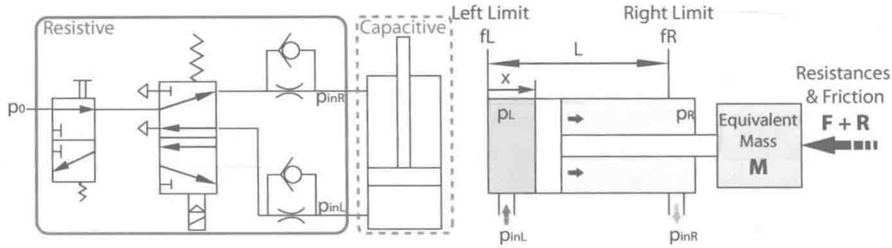

Fig. 3 Pneumatic system

Finally the pneumatic inductance is linked with the pressure by the equation:

$$p_1 - p_2 = L_p \cdot \frac{dQ}{dt}$$

Every pneumatic component has a resistance, a capacitance and an inductance. We consider a pneumatic component as resistive if the resistive effect is dominant respect to the others. Besides it is important to notice that usually pneumatic circuits are represented as simple RC circuits, in fact inertial phenomena are usually negligible with respect to resistive and capacitive ones. In rough words the dynamical behavior of a pneumatic circuit is equivalent to filling/emptying capacitive elements (volumes) through resistive elements at the inputs and outputs.

In the following we will study the mathematical model of the system depicted in Fig. 3. The model can be split in two parts: a resistive part composed by a set of resistive elements and a capacitive part including the pneumatic cylinder.

The pressure source p_0 feeds the first part composed by distribution valves, unidirectional flow control valves, pneumatic fits and pipes. All these components are resistive elements characterized by a sonic conductance C and a critical pressure ratio b constant. As in an electric circuits, the resistive element in series can be substituted with an equivalent resistive element with an equivalent C_{eq} and an equivalent b_{eq}. The relations are:

$$C_{eq} = \sqrt[3]{\frac{1}{\sum \frac{1}{C_j^3}}} \qquad b_{eq} = 1 - C_{eq}^2 \cdot \left(\sum \frac{1 - b_j}{C_j^2} \right)$$

Concerning the second part of the scheme, the pneumatic cylinder can be seen as two chambers element with a sliding common wall determining the piston movements. The dynamical behavior of the piston in a double effect cylinder can be described by:

$$p_L \cdot A_L - p_R \cdot A_R - (F + R) = M \cdot \ddot{x} \tag{4}$$

where p_L and p_R are the chambers pressures, A_L and A_R are the areas of the piston face subject to the pressure, M is the piston equivalent mass (the mass of the moving parts) and F and R are respectively the force due to the friction and to the payload.

Since the volume of the chambers is variable, the pneumatic capacitances of the pneumatic cylinder are variable and can be computed as:

$$C_{pL} = (A_L \cdot x + V_{0s}) \cdot \frac{\rho_i}{n \cdot p_{atm}} \cdot \left(\frac{p_L}{p_{atm}}\right)^{\frac{1-n}{n}} + \rho_i \cdot \left(\frac{p_L}{p_{atm}}\right)^{\frac{1}{n}} \cdot A_L \cdot \dot{x} \quad (5)$$

$$C_{pR} = (A_L \cdot [L - x] + V_{0d}) \cdot \frac{\rho_i}{n \cdot p_{atm}} \cdot \left(\frac{p_R}{p_{atm}}\right)^{\frac{1-n}{n}} - \rho_i \cdot \left(\frac{p_R}{p_{atm}}\right)^{\frac{1}{n}} \cdot A_R \cdot \dot{x} \quad (6)$$

where C_{pL} and C_{pR} are respectively the left and right chamber capacitances, L is the length of the cylinder, p_{atm} is the atmospheric pressure, n is the polytropic index ($n = 1$) and ρ_i is the air density at the initial condition.

Knowing C_{eq}, b_{eq} and the pressure at the extremities of the duct (p_{inL} or p_{inR} and p_L or p_R), remembering Eqs. (1) and (2), we can find the mass flow rate entering or outgoing the chambers:

$$G_{in} = \rho_a \cdot C_e \cdot pL \cdot \sqrt{1 - \left(\frac{\frac{p_L}{p_{inL}} - b}{1 - b}\right)^2} \cdot \sqrt{\frac{293}{T}} \quad \textit{Subsonic entering flow} \quad (7)$$

$$G_{in} = \rho_a \cdot C \cdot p_{inL} \cdot \sqrt{\frac{293}{T}} \quad \textit{Sonic entering flow} \quad (8)$$

For the outgoing flow the equations are the same. The only difference is that, since $p_L > p_{inL}$, the two pressures must be swapped.

Using Eqs. (1), (2), (5) and (6), and exploiting the relation (3), we can find the p_L and p_R evolution.

Having p_L, p_R, the piston areas (A_L and A_R), the payload and the friction force, it is possible to find the piston position (x in Fig. 3). This model has been implemented using Matlab/Simulink (see Fig. 4) to simulate a controlled device of the FMS described in Sect. 2.

4 Application to the Experimental Setup

In order to show the application of the SIL architecture to the FESTO-FMS (see Fig. 5), we present here the model of the cylinder warehouse whose aim is to distributed workpieces to testing station using a singular pneumatic cylinder. The Cylinder is fed by PLC using digital signal to force the movement and a different digital signal to enable the air supply. The device is equipped with two sensor that read the two limit of its stroke. These two digital signal are send to the PLC.

It is possible to define the parameters C and b of the resistive elements and the characteristics of the pneumatic cylinder like: piston diameter, rod diameter (right

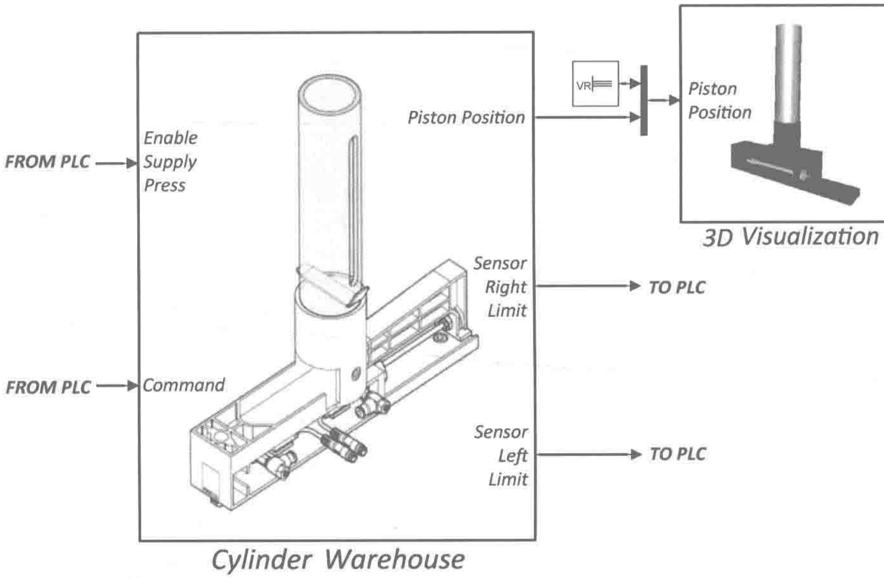

Fig. 4 Simulink scheme

and left), the friction parameters, the extraction delay, the payload during the entering and the outgoing of the piston and the equivalent mass of the piston (calculated using CAD) etc.

The presented Simulink block is directly connected to the PLC and to the 3D visualization. The Cylinder Warehouse receives as inputs from the PLC the signals that enable the supply pressure and that command the distribution valves (hence the direction of the piston move). On the other hand the PLC receives from Simulink the value of the limit switch sensors. These are virtual sensors created inside the block that give 1 if the piston reach the stroke limit or 0 otherwise. The 3D visualization has been realized importing a CAD model of the system into the Matlab toolbox Virtual Reality Toolbox. This toolbox allows the interaction with the 3D object using signals coming from Simulink. In this case the signal of interest is the piston position x.

To tune the model we have used producer (where possible) data sheet, but data like piston seal friction coefficient or the regulation of the flow valves are unknown. Obtaining these data requires experimental identification of parameters with expensive and not easy to find instruments. Due to this problem and due to the high number of parameters using a grey box identification approach, we preferred tuning the parameters to have a comparable response with the response obtained with Festo ProPneu simulator. Since the parameters are linked each other, the tuning is not easy. For example the mass of the components has been calculated using CAD tools, the friction has been achieved multiplying the weight of the moving components for the friction coefficient of the material and the resistance associated to the flow valves depends on

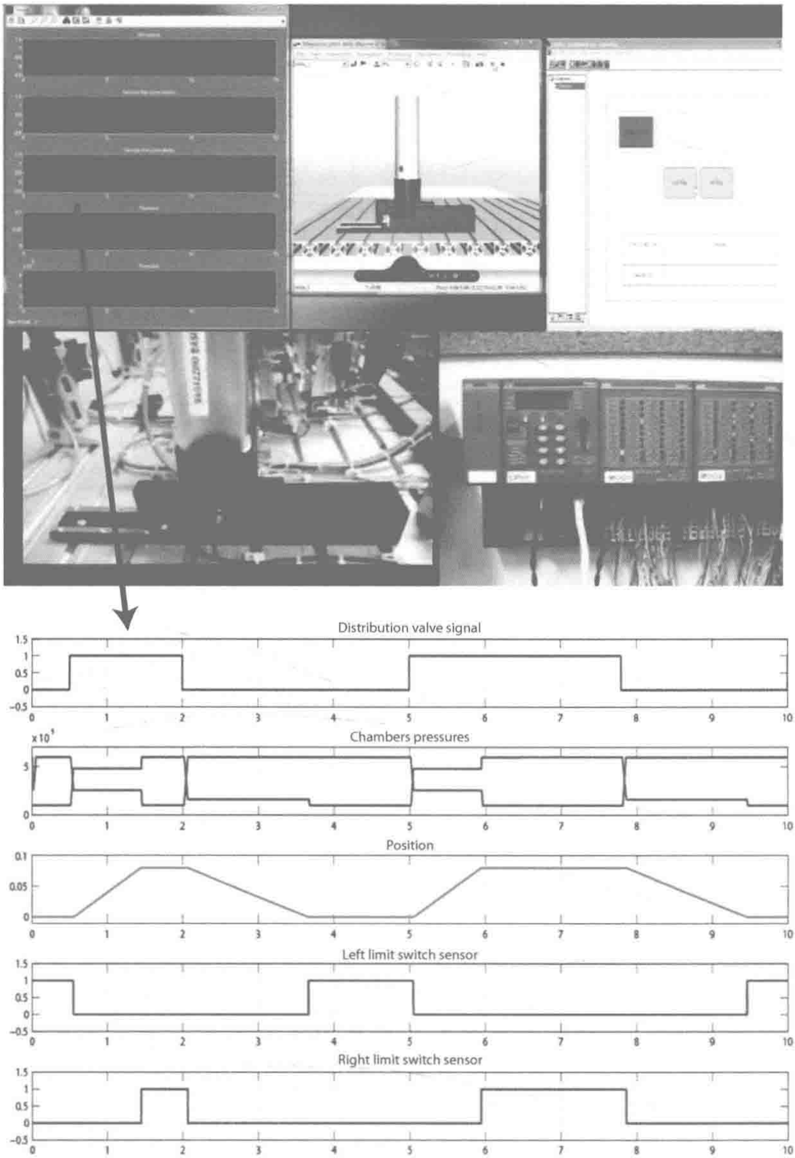

Fig. 5 Application to testbed: user front-end (*above*) and simulation results (*below*)

the mechanical resistance due to the friction. The tested control logic is very simple and roughly realize a cyclic run of the device along its stroke.

In Fig. 6 the comparison of position, velocity and chambers pressure are presented. The position and the velocity are very similar to the results obtained with ProPneu, although the mass and the loads are very little (0.06 Kg and 0.2/0.3 N).